Genetic Programming for Symbolic Regression on Incomplete Data

by

Baligh Al-Helali

A thesis
submitted to the Victoria University of Wellington
in fulfilment of the
requirements for the degree of
Doctor of Philosophy
in Computer Science.

Victoria University of Wellington
2021

Abstract

Symbolic regression is the process of constructing mathematical expressions that best fit given data sets, where a target variable is expressed in terms of input variables. Unlike traditional regression methods, which optimise the parameters of pre-defined models, symbolic regression learns both the model structure and its parameters simultaneously.

Genetic programming (GP) is a biologically-inspired evolutionary algorithm, that automatically generates computer programs to solve a given task. The flexible representation of GP along with its "white box" nature makes it a dominant method for symbolic regression. Moreover, GP has been successfully employed for different learning tasks such as feature selection and transfer learning.

Data incompleteness is a pervasive problem in symbolic regression, and machine learning in general, especially when dealing with real-world data sets. One common approach to handling data missingness is data imputation. Data imputation is the process of estimating missing values based on existing data. Another approach to deal with incomplete data is to build learning algorithms that directly work with missing values.

Although a number of methods have been proposed to tackle the data missingness issue in machine learning, most studies focus on classification tasks. Little attention has been paid to symbolic regression on incomplete data. The existing symbolic regression methods are only applicable when the given data set is complete.

The overall goal of the thesis is to improve the performance of symbolic regression on incomplete data by using GP for data imputation, instance selection, feature selection, and transfer learning.

This thesis develops an imputation method to handle missing values for symbolic regression. The method integrates the instance-based similarity of the k-nearest neighbour method with the feature-based predictability of GP to estimate the missing values. The results show that the proposed method outperforms existing popular imputation methods.

This thesis develops an instance selection method for improving imputation for symbolic regression on incomplete data. The proposed method has the ability to simultaneously build imputation and symbolic regression models such that the performance is improved. The results show that involving instance selection with imputation advances the performance of using the imputation alone.

High-dimensionality is a serious data challenge, which is even more difficult on incomplete data. To address this problem in symbolic regression tasks, this thesis develops a feature selection method that can select a good set of features directly from incomplete data. The method not only improves the regression accuracy, but also enhances the efficiency of symbolic regression on high-dimensional incomplete data.

Another challenging problem is data shortage. This issue is even more challenging when the data is incomplete. To handle this situation, this thesis develops transfer learning methods to improve symbolic regression in domains with incomplete and limited data. These methods utilise two powerful abilities of GP: feature construction and feature selection. The results show the ability of these methods to achieve positive transfer learning from domains with complete data to different (but related) domains with incomplete data.

In summary, the thesis develops a range of approaches to improving the effectiveness and efficiency of symbolic regression on incomplete data by developing a number of GP-based methods. The methods are evaluated using different types of data sets considering various missingness and learning scenarios.

Acknowledgements

I would like to express my gratitude to those who gave me the assistance and support I needed to complete this thesis. Above all, I thank Allah for blessings, wellness, and abilities, that He has given me during this work and in all my life.

First and foremost, I would like to express my very great appreciation to my supervisors Prof. Bing Xue, Prof. Mengjie Zhang, and Dr. Qi Chen. They have spent many dedicated hours and efforts to train my research skills and provide encouraging and challenging feedbacks to improve my research work. They patiently and supportivlly paved my way from far away to reach this moment of my research journey. I am profoundly indebted to Prof. Bing Xue, who, besides being a brilliant academic supervisor, is a caring human being. Her academic guidance was a corner stone in my research. Her positivity, even when I am not doing well, kept me hopeful in very difficult times. I am completely grateful to Prof. Mengjie Zhang, for his constant support during my research. He had been generously giving me from his unlimited knowledge and extensive expertise. I am deeply beholden to Dr. Qi Chen for her inexhaustible efforts in shaping my research abilities. She had been providing valuable and constructive feedback to improve my research work. To all my supervisors, you have been more than just academic supervisors to me.

I sincerely thank my dear friends Mohammed Nofal, Mohammed Al-Shaboti, and Mahdi Abdollahi, for being good friends, and for sharing delicious food with their flatmate :). I would like to offer my thanks to the members of the Evolutionary Computation Research Group (ECRG) for creating an active and interesting research environment. I also would like to thank my officemates in Cotton and Maru buildings for putting up with my annoyance. Special thanks go to my friends Abubakar Siddique, Abdulmalik Hasanain, Harisu Abdullahi Shehu, Joao Costa, Mazhar Ansari Ardeh, Hiroshika Hinduramage, Duleeka Munasinghe, Ghassem Narimani, Fangfang Zhang, Shabbir Abbasi, Ying Bi, Muru Raj Odiathevar, Apsara Wimalasiri, Junaid Haseeb, Kosisochukwu Madukwe, Atefeh Talebian, Kirita-Rose Escott, Isabella Pimentel Pincelli, and so many unlisted, who had been of a great help to ease my life during stressful times.

I would like to express my thanks to Victoria University of Wellington and Ibb university for their invaluable funding. I also would like to acknowledge the efforts of all staff members of the School of Engineering and Computer Science. I would like to thank Dr Harith Al-Sahaf, Dr. Aaron Chen, Dr. Cao Truong Tran, Dr. Andrew Lensen, and Dr. Grant Dick, whose discussions and works helped me in different stages of my research. I also want to show my gratitude to those who have taught me throughout my study journey. Especially, to Prof. Rashad Al-Jawfi, who has been always my example and without him I would never reach this moment.

Last, but most important, I would like to dedicate this work to my family. To my parents, Mohammed and Yaqut for working so hard to bring up their son. To my wife, Sena for being my love and for sharing a lot of happiness and difficulty in the life with me. To my children, Mohammed, Buthainah, and Nuha for growing up away from their dad for the past few years. I would like also to thank my sisters (Manal and Maha) who are my two eyes and my brothers (Helal, Emad, Morad, and Wasim) who are the rest of my senses. Finally, this work is dedicated to the soul of my late brother Raed. After you, I am desperately faking happiness to forget your loss. I know you wanted to be proud of your big brother and I hope I can see this when we meet, hereafter.

List of Publications

1. Al-Helali, B., Chen, Q., Xue, B., & Zhang, M. Multi-Tree Genetic Programming with Feature-based Transfer Learning for Symbolic Regression on Incomplete Data. Submitted to IEEE Transactions on Cybernetics.

2. Al-Helali, B., Chen, Q., Xue, B., & Zhang, M. Interval-valued Genetic Programming-based Feature Selection for Symbolic Regression with High-dimensional Incomplete Data. To be submitted to Evolutionary Computation Journal (ECJ).

3. B. Al-Helali, Q. Chen, B. Xue and M. Zhang, "Multi-Tree Genetic Programming with New Operators for Transfer Learning in Symbolic Regression with Incomplete Data," in IEEE Transactions on Evolutionary Computation 2021, doi: 10.1109/TEVC.2021.3079843.

4. Al-Helali, B., Chen, Q., Xue, B., & Zhang, M. (2021). A new imputation method based on genetic programming and weighted KNN for symbolic regression with incomplete data. Soft Computing, 25(8), 5993-6012.

5. Al-Helali, B., Chen, Q., Xue, B., & Zhang, M. GP with a Hybrid Tree-vector Representation for Instance Selection and Symbolic Regression on Incomplete Data. In 2021 IEEE Congress on Evolutionary Computation (CEC 2021) (Accepted). IEEE.

6. Al-Helali, B., Chen, Q., Xue, B., & Zhang, M. (2020, December). GP-based Feature Selection and Weighted KNN-based Instance Selection for Symbolic Regression with Incomplete Data. In 2020 IEEE Symposium Series on Computational Intelligence (SSCI 2020) (pp. 905-912). IEEE.

7. Al-Helali, B., Chen, Q., Xue, B., & Zhang, M. (2020, December). Data Imputation for Symbolic Regression with Missing Values: A Comparative Study. In 2020 IEEE Symposium Series on Computational Intelligence (SSCI 2020) (pp. 2093-2100). IEEE.

8. Al-Helali, B., Chen, Q., Xue, B., & Zhang, M. (2020, November). Genetic Programming-Based Selection of Imputation Methods in Symbolic Regression with Missing Values. In Australasian Joint Conference on Artificial Intelligence (AI 2020) (pp. 163-175). Springer, Cham.

9. Al-Helali, B., Chen, Q., Xue, B., & Zhang, M. (2020, June). Multi-tree genetic programming for feature construction-based domain adaptation in symbolic regression with incomplete data. In Proceedings of the 2020 Genetic and Evolutionary Computation Conference (GECCO 2020) (pp. 913-921).

10. Al-Helali, B., Chen, Q., Xue, B., & Zhang, M. (2020, July). Genetic programming with noise sensitivity for imputation predictor selection in symbolic regression with incomplete data. In 2020 IEEE Congress on Evolutionary Computation (CEC 2020) (pp. 1-8). IEEE.

11. Al-Helali, B., Chen, Q., Xue, B., & Zhang, M. (2020, July). Multi-tree genetic programming-based transformation for transfer learning in symbolic regression with highly incomplete data. In 2020 IEEE Congress on Evolutionary Computation (CEC 2020) (pp. 1-8). IEEE.

12. Al-Helali, B., Chen, Q., Xue, B., & Zhang, M. (2020, April). Hessian complexity measure for genetic programming-based imputation predictor selection in symbolic regression with incomplete data. In European Conference on Genetic Programming (EuroGP 2020) (pp. 1-17). Springer, Cham.

13. Al-Helali, B., Chen, Q., Xue, B., & Zhang, M. (2019, December). A Genetic Programming-based Wrapper Imputation Method for Symbolic Regression with Incomplete Data. In 2019 IEEE Symposium Series on Computational Intelligence (SSCI 2019) (pp. 2395-2402). IEEE.

14. Al-Helali, B., Chen, Q., Xue, B., & Zhang, M. (2019, December). Genetic programming for imputation predictor selection and ranking in symbolic regression with high-dimensional incomplete data. In Australasian Joint Conference on Artificial Intelligence (AI 2019) (pp. 523-535). Springer, Cham.

15. Al-Helali, B., Chen, Q., Xue, B., & Zhang, M. (2019, November). Genetic programming-based simultaneous feature selection and imputation for symbolic regression with incomplete data. In Asian Conference on Pattern Recognition (ACPR 2019) (pp. 566-579). Springer, Cham.

16. Al-Helali, B., Chen, Q., Xue, B., & Zhang, M. (2018, December). A hybrid GP-KNN imputation for symbolic regression with missing values. In Australasian Joint Conference on Artificial Intelligence (AI 2018) (pp. 345-357). Springer, Cham.

List of Figures

List of Tables

Contents

Chapter 1

Introduction

1.1 Problem Statement

Artificial intelligence is the branch of computer science that aims at providing computers the ability to behave like humans [200]. This includes developing data mining techniques for constructing models based on given data. The process of building these models involves methods from different fields such as database systems, machine learning, and statistics [39]. Machine learning (ML) is a sub-field of artificial intelligence that enables computers to "learn" from given samples [204]. It focuses on designing and analysing algorithms that learn models from observed data and apply the learned models on unseen data [158].

Symbolic Regression (SR) is the process of producing a symbolic mathematical model that best fits a given data set [123]. There are two important advantages of symbolic regression compared with traditional regression methods. The first one is that it does not require pre-assumptions on the desired model structures [48]. Symbolic regression searches for both the model structure and its parameters simultaneously. The second advantage is its "white-box" nature, which makes it more interpretable than many other methods [189]. Due to these attractive advantages, the application of symbolic regression becomes more and more popular in many fields [235]. However, compared to other learning tasks, e.g. classification, data quality challenges in symbolic regression have not received adequate attention [48].

Nowadays, there is a rise in the awareness of the importance of data quality in ML [195]. While an ML system = code + data (where code means model/algorithm), data is an under-valued part of ML. Fortunately, recently, there is an increasing trend to move from model-centric to data-centric learning, which is a new paradigm for AI development — focused on data quality. New terminologies such as "DataPrepOps" and "MLOps" are emerging in the data science community. DataPrepOps refers to data science and ML culture and practice that includes a set of steps that aims to build a training data set (DataPrep) for ML system operations (Ops) and MLOps [7, 217] is a compound of machine learning and operations. According to the survey results presented by Andrew Ng [1], the answer to the question of whether teams should improve the code or the data, around 80% of participants think data deserve more focus. However, only a relatively small portion of the published ML research papers has been dedicated for improving the data quality.

One challenging data quality issue is data missingness [141]. Many real-world data sets have instances with missing values due to some common reasons such as badly designed questionnaires and failures in data collection devices. For instance, in the UCI machine learning repository [62], more than 30% of the available regression data sets are annotated as having missing values. The way of handling missing values should be taken seriously as this may affect the overall performance of the learning process [13]. Data sets that contain missing values are referred to as incomplete data sets [40]. Data incompleteness is a pervasive problem in symbolic regression. Unfortunately, most existing symbolic regression methods cannot work directly on incomplete data sets.

The methods for dealing with data incompleteness can be classified

[1]https://www.youtube.com/watch?v=qnCJvQL9wcg , Mar 26, 2021

into three main approaches. The first approach is to delete all instances (features) that contain missing values then learn using only the remaining complete data [90]. Secondly, some learning methods can directly deal on incomplete data without explicitly estimating the missing values such as C4.5 [191], fuzzy-based approach [26], and learn/sup ++ ensemble method [127]. The third approach is called *imputation*, where the missing values are firstly replaced by estimated values then the learning is carried out using the complete imputed data set [4, 8]. Unlike classification, the investigation of symbolic regression on incomplete data has not received adequate efforts. In fact, the most common approach to dealing with data missingness in the existing symbolic regression methods is the deletion approach.

Data imputation is a well-established approach to mitigating the incompleteness problem in data mining [72]. Two ways to make these predictions for missing values are data-driven imputation and model-based imputation [236]. In data-driven imputation, e.g. K-nearest neighbour (KNN), each instance that has missing values is completed based on existing values in the most similar instances. The model-based imputation, e.g. regression-based imputation, uses the complete data to build a regression model for a target incomplete feature based on other features, and then the regression model is used to predict missing values in this feature.

Evolutionary computation (EC) is inspired by the biological Darwinian evolution [261]. Genetic programming (GP) is an EC algorithm that automatically generates computer programs for performing a user-defined task [21]. It typically starts with a population of random programs then refines them progressively using variation and selection strategies until getting a satisfactory solution [157]. Due to the symbolic nature of its solutions besides being free from prior knowledge, GP and its variations are the most popular techniques for symbolic regression [42].

1.2 Motivations

In data science, missingness is a serious challenge [219]. Although several approaches have been presented to tackle data incompleteness in machine learning, most studies focus on classification tasks rather than regression tasks, and only a small number of studies have been carried out on symbolic regression on incomplete data.

1.2.1 Lack of Effective Data Imputation Methods for Symbolic Regression on Incomplete Data

Imputation methods can improve the treatment of missing values [24]. This approach is widely adopted as it aims to produce complete data, which enables the use of the learning algorithms. For example, several imputation methods are developed for classification with missing values [79, 90]. Regression-based imputation has been widely used to estimate the missing values in incomplete data sets. Its idea is to build regression models for the incomplete features based on the other features using regression algorithms such as linear regression, ridge regression, decision trees, and random forests [8].

GP-based imputation has been investigated on classification tasks and has shown better performance than some popular imputation methods [223, 225, 230]. However, the applicability of imputation methods for symbolic regression has rarely been investigated. This includes examining whether the imputation methods used for handling missing values in classification can be used for symbolic regression. More importantly, new GP-based imputation methods can be designed particularly for symbolic regression tasks.

1.2.2 Unavailability of Instance Selection Methods for Symbolic Regression on Incomplete Data

The data-driven imputation methods depend commonly on the instance-based relevance and replace the missing values in incomplete instances using existing data in similar instances. One widely used example of these methods is KNN-based imputation [151, 25]. It works by finding K nearest instances to the incomplete instance, then the missing values are replaced with the mean (mode) of the numerical (categorical) values of the same feature in the retrieved instances [23]. This method is called weighted KNN (WKNN) when the retrieved instances are weighted based on their distances to the incomplete instances.

When imputing test instances, KNN searches for similar instances from the training data. This process can be time-consuming especially with a large number of instances [231]. As a solution, instance selection can be used before applying imputation [234]. KNN has been successfully applied to handle the missingness issue in different learning tasks such as clustering [3, 59] and classification [51, 79]. However, instance selection has not been investigated for the task of symbolic regression on incomplete data.

1.2.3 High-dimensional Data

Commonly, data sets with a large number of features are referred to as high-dimensional data [182]. In statistical learning, this is the case when the number of features is much larger than the number of instances [146]. High-dimensionality may cause several problems in machine learning [182]. For example, it increases the risk of over-fitting, which implies a poor generalisation ability of learned models [48]. Moreover, it might lead to a high computational learning cost and over-complicated models [182]. This situation is much more challenging when the data are incomplete as there is a need to handle the missingness problem in addition to the dimensionality issue. Most of the existing methods for symbolic regression with high-dimensional data only consider complete data [48]. To the best of our knowledge, no methods have been introduced to perform direct feature selection for symbolic regression on incomplete data.

One way to mitigate the high-dimensionality issue is feature selection (FS). Feature selection aims at choosing a useful subset of features from a given set of features [259]. The usefulness can be measured in the sense of improving the learning performance while reducing the computation cost. EC techniques have been used successfully for feature selection to improve the performance of different learning tasks such as clustering and classification [257]. Due to its natural selection ability, GP is one of the EC techniques that have attained more investigations for feature selection in recently published studies [244, 164, 28]. For symbolic regression, feature selection methods for improving the generalisation ability of GP are proposed [48, 17]. However, such studies use complete data sets. The situation is much more challenging when the data are incomplete. In this case, the methods to be developed should consider the treatment of missing values in addition to handling the dimensionality issue. This work presents an attempt to fill this gap by developing feature selection methods for symbolic regression on incomplete high-dimensional data sets.

1.2.4 Small Sized Data

Lack of data is a problematic issue when learning from real-world data. All the aforementioned approaches to dealing with data missingness require enough data to train reliable models. Thus, in the case of suffering from a shortage of data, transfer learning can be utilised. Transfer learning is the learning paradigm in which knowledge learnt on a specific domain is reused to improve the learning process in another domain [174]. The giving domain is called a source domain (SD) while the receiving domain is called a target domain (TD). The learning process in the domain is identified by the domain itself, denoted as $\mathcal{D}$, and the task to be learned in this domain, $\mathcal{T}$. The domain $\mathcal{D}$ is determined by a feature space $\mathcal{X}$ and its marginal probability distribution $P(\mathcal{X})$, while the task $\mathcal{T}$ consists of a target function $f(.)$ and a label space $\mathcal{Y}$. In transfer learning, at least one of the aforementioned SD learning components should differ from the corresponding one(s) in the TD. Transfer learning has been intensively investigated for classification and clustering [193, 190, 70, 218]. However, only a few studies have been conducted on transfer learning for symbolic regression [97, 68, 104]. Moreover, none of these studies consider the existence of missing values. Therefore, there is a need to investigate transfer learning for symbolic regression on incomplete data. This thesis aims at developing transfer learning methods to reuse knowledge extracted from symbolic regression in complete domains (domains with complete data) for improving the symbolic regression learning in different yet related incomplete domains (domains with missing values).

It can be noticed that data incompleteness represents a serious problem in machine learning and the investigations of addressing it focus on classification tasks. Moreover, incompleteness and high dimensionality provide additional challenges to symbolic regression tasks. An extra challenge that can occur with data missingness is data shortage. Missingness is more problematic when the data amount is inadequate, which limits the ability to learn effective symbolic regression models. Therefore, how to deal with missing values in symbolic regression is still an open issue. This thesis aims to fill this gap by developing novel methods to address these challenges when performing symbolic regression.

1.3 Research Goals

The main goal of this thesis is to develop new GP-based methods that can improve symbolic regression on incomplete data. This includes methods to address challenges related to data missingness, high dimensionality, and data shortage. To achieve this goal, a number of objectives have been established to guide this research as follows.

1. *Developing GP-based imputation methods for symbolic regression on incomplete data.*

   This thesis plans to develop GP-based data imputation methods for symbolic regression on incomplete data. Such methods try to utilise the GP advantages in constructing regression-based imputation models. Moreover, both the instance-based similarity and the feature-based predictability will be combined to come up with a new imputation method. This objective aims at improving the imputation accuracy on predicting the missing values, which, consequently, improves the symbolic regression on incomplete data.

2. *Developing GP-based instance selection methods for symbolic regression on incomplete data.*

   Data-driven imputation depends on the instance similarity to fill in the missing values. To improve this approach, the thesis proposes instance selection methods. The selected instances should be representative of the original data set. This work proposes methods that combine both GP and KNN to enhance imputing the missing values. Such a hybridisation is expected not only to improve the effectiveness but also the efficiency of symbolic regression on incomplete data. Moreover, instead of using an explicit, separate instance selection process, this objective proposes integrating the selection of the instances with the process of constructing symbolic regression models. It aims to develop an evolutionary method that selects the instances in order to improve the imputation process while optimising the symbolic regression performance.

3. *Developing GP-based feature selection methods for symbolic regression on high-dimensional incomplete data sets.*

In regression-based imputation, the features having missing values are considered as target variables while using other features as imputation predictors. However, some features might be redundant/irrelevant and they are less likely to be useful for improving the prediction accuracy. This thesis aims at developing a GP-based method that can be used for selecting both imputation predictors and regression features when performing symbolic regression on incomplete data. The goal is to develop a method that can select a good feature set from incomplete data directly. As some of the selected features might be incomplete, this work proposes another selection step to select their imputation predictors. Usually, data imputation is used as a pre-processing method before performing further learning steps. Adopting such a scheme may result in imputing incomplete features that might not benefit the learning process. This scheme performs learning based on all the available features regardless of their relatedness or redundancy, which can be unprofitably costly when dealing with high-dimensional data. To address this issue, this thesis aims to develop an approach that uses interval-valued GP to select the features before applying data imputation. The method proposed in this objective is expected to improve the quality of the used feature set, which in turn improves the performance of imputation and symbolic regression on incomplete data.

4. *Developing GP-based transfer learning methods for symbolic regression on incomplete data.*

   In addition to data missingness, some domains suffer from the shortage of available data, which limits the ability to handling the missing values and constructing symbolic regression. This objective aims to utilise transfer learning for symbolic regression on incomplete data in such domains. The idea of this objective is to transfer knowledge learned from symbolic regression in a complete domain with adequate knowledge (source domain) into a related, but different, symbolic regression task in an incomplete domain with scarce data (target domain). The goal of this objective is use GP to construct transformations from source domains to target domains. For this purpose, the thesis proposes utilising the ability of multi-tree GP to construct multiple features for building the transformations between domains. These transformations are used to benefit the learning in the incomplete target domain from the learning in the complete source domain. This approach is expected to compensate for the data shortage in the target domain by reusing the knowledge extracted from the source domain such that the symbolic regression in the target domain is improved.

5. *Integrating feature selection with transfer learning for symbolic regression on incomplete data.*

   In some cases, incomplete domains with limited data might have relatively a large number of features. This situation adds an extra challenge to the symbolic regression learning process. To deal with this situation, this objective aims at integrating feature selection with transfer learning for symbolic regression on incomplete data. The idea of this objective is to propose a feature selection mechanism that can be utilised during the transfer learning process to benefit the symbolic regression process in a target incomplete domain. The feature selection aims at utilising the ability of GP to pick the important features. The feature selection combines GP-based feature importance from both the source domain and the target domain. The feature selection outcomes are integrated with the multi-tree GP-based transfer learning process. This approach has the potential to extend the applicability of GP-based transfer learning to incomplete domains with limited data, where the number of features is large.

1.4 Major Contributions

This thesis makes the following major contributions.

1. The thesis shows how GP can be utilised for enhancing data imputation in symbolic regression on incomplete data. The thesis develops a new GP-based imputation method for handling missing values in symbolic regression on incomplete data. This method is based on combining GP with WKNN is proposed for symbolic regression on incomplete data. It works by constructing GP-based models to predict the missing values of incomplete features using other available features, where the instances used for constructing such models are selected using WKNN. The experimental results on real-world data sets show that the proposed method outperforms a number of state-of-the-art methods with respect to the imputation accuracy and the symbolic regression performance. This work is one of the first studies on developing GP-based imputation methods for symbolic regression on incomplete data.

   Parts of this contribution have been published in:

   Al-Helali, B., Chen, Q., Xue, B., & Zhang, M. (2021). A new imputation method based on genetic programming and weighted KNN for symbolic regression with incomplete data. Soft Computing, 25(8), 5993-6012.

   Al-Helali, B., Chen, Q., Xue, B., & Zhang, M. (2019, December). A Genetic Programming-based Wrapper Imputation Method for Symbolic Regression with Incomplete Data. In 2019 IEEE Symposium Series on Computational Intelligence (SSCI 2019) (pp. 2395-2402). IEEE.

   Al-Helali, B., Chen, Q., Xue, B., & Zhang, M. (2018, December). A hybrid GP-KNN imputation for symbolic regression with missing values. In Australasian Joint Conference on Artificial Intelligence (AI 2018) (pp. 345-357). Springer, Cham.

2. The thesis shows how instance selection can help handle missing values in GP-based symbolic regression on incomplete data. This contribution presents an instance selection method that combines both GP and KNN to enhance imputing the missing values, which in turn improves the symbolic regression performance on incomplete data. This method proposes a mixed tree-vector representation for GP to perform instance selection and symbolic regression on incomplete data. In this representation, each individual has two components: an expression tree and a bit vector. While the tree component constructs symbolic regression models, the vector component selects the instances that are used to impute missing values by the WKNN imputation method. The obtained experimental results show the applicability of the proposed method on real-world data sets with different missingness scenarios. When compared with existing methods, the proposed method not only produces more effective symbolic regression models but also achieves more efficient imputations. This thesis provides a first instance selection method for symbolic regression on incomplete data.

   Parts of this contribution have been published in:

   Al-Helali, B., Chen, Q., Xue, B., & Zhang, M. (2020, December). GP-based Feature Selection and Weighted KNN-based Instance Selection for Symbolic Regression with Incomplete Data. In 2020 IEEE Symposium Series on Computational Intelligence (SSCI 2020) (pp. 905-912). IEEE.

   Al-Helali, B., Chen, Q., Xue, B., & Zhang, M. GP with a Hybrid Tree-vector Representation for Instance Selection and Symbolic Regression on Incomplete Data. IEEE Congress on Evolutionary Computation (CEC 2021), 2021, Accepted.

3. The thesis shows how GP-based feature selection can be utilised to improve the performance of symbolic regression on high-dimensional incomplete data. This contribution proposes a new GP-based method for feature selection for high-dimensional symbolic regression on incomplete data. The method developed in this thesis is based on interval-valued GP (IGP) to select features directly from incomplete data. In this method, instead of selecting the imputation predictors for all incomplete features, IGP is used to select the set of incomplete features that should be considered. This method is applied to high-dimensional symbolic regression on various types of data sets considering different missingness scenarios. In addition to the comparison with the case of using all the available features, the proposed method is compared with state-of-the-art feature selection methods that have the ability to select features directly from incomplete data. The results show that the proposed method not only improves the symbolic regression performance, but also selects fewer yet more relevant features. This work is one of the first studies on developing GP-based feature selection methods for symbolic regression on incomplete data. Such methods will be useful to avoid several problems linked to the high-dimensionality challenge when performing symbolic regression on incomplete data.

   Parts of this contribution have been published in:

   Al-Helali, B., Chen, Q., Xue, B., & Zhang, M. Interval-valued Genetic Programming-based Feature Selection for Symbolic Regression with High-dimensional Incomplete Data. To be submitted to Evolutionary Computation Journal (ECJ).

   Al-Helali, B., Chen, Q., Xue, B., & Zhang, M. (2020, July). Genetic programming with noise sensitivity for imputation predictor selection in symbolic regression with incomplete data. In 2020 IEEE Congress on Evolutionary Computation (CEC 2020) (pp. 1-8). IEEE.

Al-Helali, B., Chen, Q., Xue, B., & Zhang, M. (2020, April). Hessian complexity measure for genetic programming-based imputation predictor selection in symbolic regression with incomplete data. In European Conference on Genetic Programming (EuroGP 20202) (pp. 1-17). Springer, Cham.

Al-Helali, B., Chen, Q., Xue, B., & Zhang, M. (2019, December). Genetic programming for imputation predictor selection and ranking in symbolic regression with high-dimensional incomplete data. In Australasian Joint Conference on Artificial Intelligence (AI 2019) (pp. 523-535). Springer, Cham.

4. The thesis shows how GP-based transfer learning can help improve symbolic regression performance on incomplete data. This contribution utilises transfer learning to reuse knowledge learnt from different yet related but complete domains. Due to its powerful feature construction ability, GP is used to construct feature-based transformations that map the feature space of the source domain to that of the target domain such that their differences are reduced. Particularly, this work proposes a new multi-tree GP-based feature construction approach to transfer learning in symbolic regression on incomplete data. It transfers knowledge related to the importance of the features and instances in the source domain to the target domain to improve the learning performance. Moreover, new genetic operators are developed to encourage minimising the distribution discrepancy between the transformed domain and the target domain. The experimental results show that the proposed method achieves better performance compared with a set of traditional learning methods on real-world data sets with various incompleteness and transfer learning scenarios. This contribution helps in advancing the use of transfer learning for dealing with data incompleteness and data shortage problems in symbolic regression.

Parts of this contribution have been published in:

B. Al-Helali, Q. Chen, B. Xue and M. Zhang, "Multi-Tree Genetic Programming with New Operators for Transfer Learning in Symbolic Regression with Incomplete Data," in IEEE Transactions on Evolutionary Computation, doi: 10.1109/TEVC.2021.3079843.

Al-Helali, B., Chen, Q., Xue, B., & Zhang, M. (2020, July). Multi-tree genetic programming-based transformation for transfer learning in symbolic regression with highly incomplete data. In 2020 IEEE Congress on Evolutionary Computation (CEC 2020) (pp. 1-8). IEEE.

Al-Helali, B., Chen, Q., Xue, B., & Zhang, M. (2020, June). Multi-tree genetic programming for feature construction-based domain adaptation in symbolic regression with incomplete data. In Proceedings of the 2020 Genetic and Evolutionary Computation Conference (GECCO 2020) (pp. 913-921).

5. This thesis shows how to integrate GP-based feature selection with GP-based transfer learning to improve symbolic regression on incomplete data, where the number of features is large. This contribution proposes a feature selection mechanism in a multi-stage transfer learning method such that the process of handling missing values is enhanced in each stage. Feature selection is based on the implicit feature selection ability of GP when performing symbolic regression in both the source domain and the target domain. Transfer learning is based on multi-tree GP refined gradually in each stage according to the feature selection feedback. This method is applied to symbolic regression tasks on incomplete domains with limited data. The obtained results show positive transfer learning with a notable efficiency improvement. This work initiates the efforts in conducting high-dimensional symbolic regression on domains with small data sets that have missing values.

   Parts of this contribution have been submitted as:

Al-Helali, B., Chen, Q., Xue, B., & Zhang, M. Multi-Tree Genetic Programming with Feature-based Transfer Learning for Symbolic Regression on Incomplete Data. Submitted to IEEE Transactions on Cybernetics, June 2021.

1.5 Organisation of Thesis

The rest of this thesis is organised as follows. Chapter 2 presents essential basic background concepts along with a literature review of the related studies. Chapters 3-7 present the main contributions of the thesis. Figure 1.1 shows the overall structure of the contributions of this thesis. Each chapter addresses one of the objectives described in Section 1.3. Finally, Chapter 8 concludes the thesis and provides some potential future research directions.

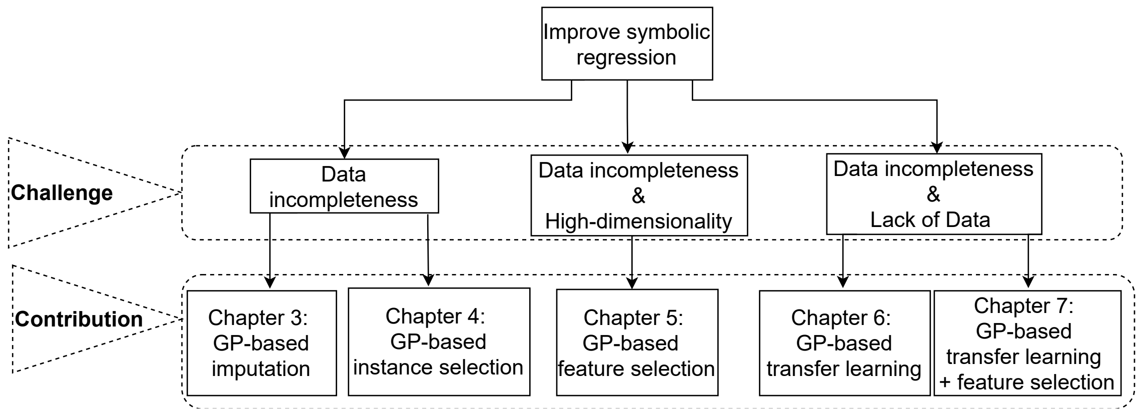

Improve symbolic regression
Challenge
Data incompleteness
Data incompleteness & High-dimensionality
Data incompleteness & Lack of Data
Contribution
Chapter 3: GP-based imputation
Chapter 4: GP-based instance selection
Chapter 5: GP-based feature selection
Chapter 6: GP-based transfer learning
Chapter 7: GP-based transfer learning + feature selection

Figure 1.1: The overall structure of the contributions of the thesis.

Chapter 2 presents basic concepts and essential background of symbolic regression, data missingness, genetic programming, feature selection, and transfer learning. The main focus of this study is GP-based symbolic regression on incomplete data. Therefore, a detailed literature review for the related works on GP-based symbolic regression is provided with a focus on highlighting the ways of treating missing values. It also visits advances in data imputation, feature selection, and transfer learning for symbolic regression. It then discusses open questions and current challenges that form the motivations of the thesis.

Chapter 3 presents a GP-based method for imputing incomplete data for symbolic regression on incomplete data. This method is based on GP and WKNN. The imputation methods are evaluated for symbolic regression on synthetic and real-world incomplete data sets. In addition to the imputation performance, the regression performance is also used for evaluating the imputation models.

Chapter 4 gives an instance selection method for symbolic regression on incomplete data. This chapter proposes a mixed tree-vector representation for GP to perform instance selection and symbolic regression on incomplete data. While the tree component constructs symbolic regression models, the vector component selects the instances that are used to impute missing values by the WKNN imputation method. The experimental work is conducted on real-world data sets with different missingness scenarios.

Chapter 5 presents a feature selection method for symbolic regression on incomplete data. This chapter proposes an interval-valued GP-based approach to select features directly from incomplete data. This method is applied to high-dimensional symbolic regression on incomplete data. The experimental works have been conducted on various types of data sets considering different missingness scenarios.

Chapter 6 provides a GP-based transfer learning method for symbolic regression on incomplete data. This method utilises transfer learning to reuse knowledge from different yet related complete domains to improve learning in incomplete domains. The proposed method is based on multi-tree GP to construct feature transformations from a source domain to a target domain. The method is examined on symbolic regression on incomplete data using real-world data considering different transfer learning scenarios.

Chapter 7 integrates feature selection with a GP-based transfer learning method for symbolic regression on incomplete data. This method employs feature selection to improve transfer learning from complete source domains to incomplete target domains. The feature selection provides feedback during the learning process to filter the used features gradually.

Finally, Chapter 8 summaries the work and draws the overall conclusions of the thesis. It also identifies key research points and contributions of the thesis. It then suggests some possible future research directions.

Chapter 2

Literature Review

This chapter introduces the basic concepts of relevant topics in this thesis along with a literature review of the related studies. After introducing machine learning and evolutionary computation, more detailed discussions on GP and symbolic regression are provided. Moreover, the main concepts related to incompleteness, feature selection, instance selection, and transfer learning are given. For the related work, this chapter discusses recent proposed methods that are related to the contributions of this thesis highlighting their advantages and limitations.

2.1 Background

2.1.1 Machine Learning

Machine learning is the process of developing methods that enable machines to detect patterns automatically from given data, and then use the discovered patterns to predict unseen data [197]. Machine learning is considered as a crucial field of artificial intelligence as it facilities building different kinds of intelligent applications, such as computer vision and data modelling [188]. However, there are broad fields that intersect with machine learning such as mathematics, statistics, physics, and computer science [201].

The main goal of machine learning is to develop practical algorithms considering several factors such as accuracy, time and space efficiency, the data amount, and the interpretability [9]. For a given task, rather than programming the computer to solve it explicitly, the developed methods in machine learning enable the computer to come up with its own solution based on the provided instances [208]. The learning is usually done on instances of data (also called examples and observations), direct experience, or instruction. Therefore, machine learning paradigm can be viewed as "programming by example" [9].

2.1.1.1 Supervised Learning

In supervised machine learning, the desired outputs for the training data are known, i.e. the training of the algorithms is performed using labeled instances [122]. These labels are used to detect the errors of the model by comparing them with the predicted ones. The learning is then refined to minimise the obtained errors. The learning process stops when achieving an acceptable performance. The built model can be applied to predict the labels of unseen test data. Commonly, supervised learning is adopted when there is available historical data to predict future outcomes. Two main supervised learning tasks are classification and regression [197]. The labels in the classification task are categorical while the labels are numerical in the regression task. Symbolic regression is a supervised learning process that aims at discovering the mathematical expressions that best fit data in regression tasks.

2.1.1.2 Unsupervised Learning

For unsupervised learning, the labels of the data instances are not available. The model is trained by exploring the data to detect hidden patterns and structures [92]. Unsupervised machine learning is suitable when it is difficult to annotate the data or when there is no agreement on what the right labels should be. It is usually used to detect group of instances that are similar or relevant in some manner. Clustering and association rule learning are common tasks of unsupervised learning.

2.1.1.3 Semi-supervised Learning

In semi-supervised learning, some of the available data are labeled but unlabeled data are also used to construct the desired model [271]. This case is useful when labeling the whole data is too costly or even unrealistic sometimes. An example of such situations is the classification of web pages as it is easy to get labels for many web pages but it is difficult to get labels for all web pages. Semi-supervised learning utilises a fully labeled data along with the unlabeled data for the training process. Co-training algorithm is a common example of semi-supervised learning. Co-training starts with learning separate classifiers for two views of the data using labeled examples. The best predictions of the classifiers are then used to iteratively generate additional labeled training data.

2.1.1.4 Reinforcement Learning

In reinforcement learning, a trial and error strategy is used to determine the rewards yielded by the training process [215]. There are three main components of reinforcement learning: 1) an agent who is the primary decision maker, 2) the environment which constitutes everything the agent interacts with and 3) actions which dictate what the agent does. The aim of reinforcement learning is to perform actions leading to high rewards or driving optimal outcomes. This kind of learning is widely utilised in robotics and gaming.

2.1.2 Evolutionary Computation and Genetic Programming

2.1.2.1 Evolutionary Computation

Evolutionary computation (EC) is an artificial intelligence field which interests in designing, building, applying and analysing algorithms based on the Darwinian natural selection principles [76]. The main idea in EC is that: given a population of individuals, the natural selection process caused by the environmental pressure grows the generations' population fitness over time [76]. Given an objective function, a set of candidate solutions are created randomly and this function is used as a quality measure to select some better candidates, parents, to seed the next generation by applying variation operations (e.g. crossover and/or mutation). This process results a set of new candidates, the offspring, and can be repeated until a satisfactory solution is found or pre-defined stop criteria is reached. The selection process can be performed to select fitter individuals as parents or to survive, but typically weak individuals should have a chance to be selected as well.

EC algorithms can be mainly categorised to evolutionary algorithms (EAs) and swarm intelligence (SI) algorithms [20]. EAs can be classified to evolutionary programming [82], evolutionary strategies (ESs) [27], genetic algorithms (GAs) [100] and genetic programming (GP) [123]. Popular SI algorithms are particle swarm optimisation (PSO) [111] and ant colony optimisation (ACO) [73]. There are also other popular EC algorithms such as differential evolution (DE), learning classifier systems (LCS), artificial immune systems(AIS), and artificial bee colony (ABC) algorithms [20].

In an evolutionary process, many components are stochastic. The crossover and mutation are based on information that will be exchanged during evolution and usually performed in a random manner. However, the selection operators can be either stochastic, or deterministic. The general scheme of an EA can be given as in Algorithm 1 [76].

The main components of evolutionary algorithms include [76]:

Algorithm 1: The general scheme of an EA.

```
1 Initialise A population of individuals of the first generation;
2 while not stop do
3     Evaluate all individuals of the current generation (computing the fitness);
4     Select parent(s) from the current generation population;
5     Create new individual(s) (offspring) using genetic operators on the selected
      parents;
6     Evaluate the offspring;
7     Form new generation population consisting of new offspring (some selected
      parents can be included) ;
8 end
```

- Representation

  When using an EA to solve a problem, the representation of the individuals should be specified. Such individuals can be seen as complex structured phenotypes and represented by corresponding genotypes. There are common representations such as bit-strings, permutations of integers, real-valued vectors, or trees.

- Fitness function

  To evolve the population, there is a need to measure how good the solutions are. Therefore, designing a suitable fitness function is very important. Evaluating such functions may be computationally expensive.

- Initialisation

  The initial set of solutions is usually created randomly by sampling of the search space. Such sampling can be performed uniformly. However, uniform sampling might not be well-defined in some search spaces such as parse-tree spaces. Another common practice is to include some pre-known good solutions in the initial population.

- Selection

To generate a new generation out of the current one, selected individuals (parents) are used to produce new individuals. However, how these parents are chosen? The answer of this question defines the selection component of EA. Selection is usually based on the fitness values of the individuals. Usually, fitter individuals have higher possibility to be selected than others. However, some chances should be given for selecting less fitted individuals.

- Crossover

  Crossover operators exchange the information between selected parents to generate new offspring. Its idea is that parents that can achieve good fitness values include building blocks, i.e. good parts (partial solutions), and recombining such parts might increase the fitness of their children. Nevertheless, crossover can be performed in many other ways.

- Mutation

  Mutation is the process of changing parts of the selected individual to produce a new one. Mutation maintains genetic diversity from the current generation to the next. It is inspired from biological mutation, where one or more genes are altered in a chromosome. Although mutation could be implemented in different ways, it is common to modify the genotype with a small probability (e.g. flip some bits of a bitstring).

- Termination criterion

  To have a practical EA algorithm, a reasonable stopping criterion should be used to avoid the time consuming optimisation infinite process when the optimal solution can not be found. A common used strategy is to state a maximum number of generations. Another criterion is to stop if there is no considerable improvement achieved after a pre-defined amount of time or generations.

- Setting the parameters

  To design an EA, a number of parameters should be specified. Examples of these parameters are selection size, population size, crossover probability, and mutation probability. The main challenge here is how to set such parameters. The combined influence of all parameters should be considered as the effect of one parameter is often unpredictable. Most studies rely on empirical selection for the values of the parameters.

Exploration and exploitation are two crucial concepts in EAs [75]. The idea behind exploration is that the current best-so-far solutions might be local optima and there is a need to explore different new regions of the search space aiming to find better ones. However, the exploitation idea is to keep searching around the best-so-far solutions hoping that their neighbourhood contains even better solutions.

2.1.2.2 Genetic Programming

Genetic programming (GP) is an EC technique that generates computer programs for solving a given problem [21]. GP has a wide range of machine learning applications including classification [137, 5, 6, 150, 117, 77], and regression [18, 95, 42]. Moreover, GP has many important applications [253] such as optimisation, scheduling, systems modelling, and signal processing. There are several advantages of GP. Firstly, there is no need for prior knowledge on the structure of the solution. Another advantage is that it can represent the solutions using a flexible representation which is suitable for human reasoning.

GP works on a set of individuals called population, each of them represents a potential solution to the underling problem. To use GP for solving a problem, Koza [124] states that there is a need to specify:

- The terminal set: inputs that can be constants or variables.

- The function set: functions that are domain specific and combined with the terminal set for constructing potential solutions to the considered problem.
- The fitness function: it is an objective function that measures how the solution is appropriate to the problem in question.
- The control parameters set: This set defines the GP settings such as probabilities of the crossover and the mutation and the size of the population.
- The termination criterion: It is a predefined parameter to state when the evaluation process should be stopped. Commonly used criteria include the number of the generations and the fitness error tolerance.

There are different variations for GP such as strongly-typed GP [160], Linear GP [33], and Grammatical GP [251, 156]. A description of the key components of standard tree-based GP is given below.

- **Representation**

  Commonly, GP individuals are represented as trees. The terminal nodes (leaves) of such trees take their values from the terminal set which can be randomly generated constants, or selected from a set of given feature values. Whereas the non-terminal nodes are taken from the function set representing operations to be performed on the output values of their child nodes which can be simple (e.g., arithmetic operators such as +, -, and *), or more complex (e.g., loops).

- **Initialisation**

  The GP process starts with generating an initial population of a number of initial individuals considering the minimum- and maximum-depth of the generated trees. There are three popular initialisation methods. The first one is called *full* method which generates an individual by randomly selecting elements from the function set until reaching a pre-defined maximum depth. The leaf nodes are then populated randomly by elements from the terminal set. The second one is the *grow* method which is similar to the full method, however, it selects from both function and terminal sets. Therefore, the branch stops growing at that point where a terminal node is selected. To emphasise the diversity of the initial individuals, the two methods are combined in the *ramped half-and-half* method. In this method, half of the individuals are generated using the full method and the other half is generated using the grow method.

- **Fitness evaluation**

  The evaluation is very essential in GP to measure how good an evolved program is. The specification of the fitness function is critical as it is used to assess the performance of the whole evolutionary process. The fitness function is related to the problem to be solved. For example, for a classification problem, the fitness function can be the accuracy and it can be the prediction error for a regression problem.

- **Selection**

  The selection is the process of identifying which individuals from the current generation will be used to produce new individuals for the subsequent generation. The fitness value is used to assess the chances of an individual to be selected. Hence, fitter individuals are more likely to be selected than others. However, involving poor individuals might help produce a better generation as it increases the diversity of the individuals. One of the selection methods is the **roulette wheel selection** method. It works by randomly selecting a program from the current generation, where the selection probability is based on the fitness values. This step is repeatedly performed to select a number of individuals. Another popular selection method is the **tournament selection** method. In this method, a predefined number (tournament size) of individuals are randomly sampled from the population and then the best one is selected which gives a chance for all programs to be drawn in the first step regardless of their fitness values. However, it requires to set an extra parameter which is the tournament size.

- **Genetic Operators**

  A number of genetic operators are used to produce the subsequent generation programs (children) from the programs of the current generation (parents). There are three genetic operators in GP: 1) reproduction, 2) crossover, and 3) mutation.

  1. **Reproduction**

     Reproduction copies the selected individuals from the current generation to the subsequent one [125]. The copied individuals are selected from the current population based on the fitness values [126]. However, the selection of the best programs is not guaranteed. If it is asserted that the best individuals are selected, it is called *elitism*, which aims to maintain the achieved level of performance and prevent it from degrading in the subsequent generations. This operation ensures that the next generation is at least as good as the current one.

  2. **Crossover**

     The crossover operator exchanges genetic materials from individuals of the current generation (parents) to create new individuals (children) for the subsequent generation. The parent individuals are selected using one of the selection methods. After that, a crossover point is chosen in each tree, and then sub-trees are swapped at the crossover points. There are several methods for the crossover operator such as one-point, two-point, uniform and half-uniform, cut and splice, and three-parent crossover [183].

3. **Mutation**

   The mutation operator uses only one parent from the current population to produce the offspring. After selecting the parent from the current population, a mutation point on this parent tree is randomly selected, and another generated parent tree replaces the sub-tree at that point in order to generate the new child. The main difference between crossover and mutation is that crossover combines the existing genetic material, whereas mutation allows producing new genetic material by using newly generated sub-trees.

2.1.3 Incomplete Data

2.1.3.1 Data Missingness

There are different categorisations of missing values [72] including missing completely at random (MCAR), missing not at random (MNAR), and missing at random (MAR). In MCAR missingness, the missing values are a random subset of the given data set with a completely random reason. In this kind of missingness, the missingness probability of a value is not related to any other value. When missing values are MCAR, most handling missing techniques give unbiased results [72]. This is because no gain can be guaranteed by analyzing the available data to estimate associations with missing values. For MNAR missingness, the probability that a value is missing is related to non-observed information [198]. Due to the lack of valuable information, there is no universal method to handle this type of missingness. Usually, missing values are neither MCAR nor MNAR [72]. Instead, the probability of missingness is commonly related to information that is present, i.e., missingness depends on other existing values. This type of missingness is called missing at random (MAR). This term is confusing, however, it can be justified as missing values may indeed be considered random conditional on other values [198]. When the missingness is MAR, most statistical techniques for handling missing values give biased results [198]. However, more sophisticated methods (e.g. regression-based multiple imputation) may give good results [72].

2.1.3.2 Handling missing values

The methods for handling missing values can be categorised into three different approaches as follows [90].

1. Deletion of incomplete cases and learning using only the complete data portion [90]. This approach is simple but it might cause a loss of informative data.
2. Imputation of missing values and learning using a complete data set with the imputed ones [206, 142, 8]. This approach enables the use of different learning methods but usually, it is time-consuming.
3. Machine learning implicit procedures that can directly deal with incomplete instances in the application process without explicitly estimating the missing values such as decision trees [191, 249], fuzzy approaches [26, 167], and ensemble methods [127, 106]. This approach avoids the overhead of imputation but this might come at the expense of the accuracy that could be gained if the missing values are recovered effectively.

2.1.3.3 Data Imputation

This thesis will focus on the imputation approach. Here we provide a brief background on missing value imputation. For more details, the reader is refereed to [219], where a systematic review of machine learning-based incomplete data imputation techniques is presented.

There are two main types of imputation: single imputation and multiple imputation. Single imputation directly replaces missing values by estimated ones based on observed data. Popular single imputation methods include: mean imputation, mode imputation, cold deck imputation, hot deck imputation, KNN imputation, and regression imputation [142, 60]. The common problem with single imputation is that it ignores uncertainty and underestimates the variance [199]. Multiple imputation addresses this problem by considering both within-imputation uncertainty and between-imputation uncertainty [216].

In multiple imputation, multiple complete copies of the data are produced each for independent estimates for the missing values. Conditional on observed data, a posterior distribution of missing data is modeled to draw a random sample and create several imputed data sets reflecting the uncertainty about imputation [207]. Then, standard statistical analysis is conducted on the imputed data sets separately and the results are combined to get an estimated value [216].

Some methods are widely used for imputation [22, 8]. K-nearest neighbour (KNN) is used to impute the missing values by averaging the k most similar instances. Another method is random forest (RF), which is used for imputation by employing decision trees to predict the missing values based on the non-missing ones. It starts by replacing the missing data with the average of the corresponding complete values and then iteratively improves the missing imputation using proximity.

2.1.4 Feature Selection and Instance Selection

2.1.4.1 Feature Selection

Feature selection is the process of finding a small necessary and sufficient representative subset of the available features [116]. Typically, this can be done by choosing a subset leads to a better performance than all other subsets [166]. The performance can be the prediction accuracy [119].

In feature selection, the two main components are the search strategy and the evaluation criteria [256]. The search strategy tries to find the best feature subset(s). This procedure might start with an empty subset, full feature set, or a random subset of features and then applies a search strategy for finding an optimal feature subset [134]. Traditional feature selection methods such as sequential forward selection (SFS) [252], sequential backward selection (SBS) [152], sequential floating forward selection (SFFS), and sequential floating backward selection (SFBS) [186] have the disadvantage of getting into local optima easily. Therefore, more efficient global search methods are utilised.

EC techniques such as PSO, GA, and GP have been applied successfully to feature selection [168, 258, 211, 136]. This is due to the ability of these methods to perform a global search on a population of possible solutions. However, applying these evolutionary computation methods on high-dimensional problems is still a challenge in this field [256].

Based on the evaluation criteria, feature selection methods can be classified as wrapper, filter and embedded approaches based on the way of involving a learning algorithm in evaluating the quality of the feature subsets [145]. Filter feature selection methods do not require learning algorithms during the selection process as the features are selected based on data properties such as distance, information, dependency and consistency [55]. Wrapper methods use a learning algorithm for evaluating the features during the selection process. For the embedded feature selection methods, the features are selected while building the learning models simultaneously.

In comparison between the different methods, wrapper methods usually obtain better performance. However, filter methods are usually computationally less expensive. Feature selection methods are also categorised into single objective approaches and multiobjective approaches based on the number of considered objectives, e.g. accuracy maximisation, dimensionality minimisation, and complexity minimisation [221].

2.1.4.2 Instance Selection

In learning tasks, a training set of instances is used to build a model that can then predict the label of new unseen test instances later. Sometimes, not all training instances are useful for the learning algorithm. Therefore, a better performance (or at least acceptable with respect to some measures) can be achieved when ignoring some of the available instances. Such process is called instance selection [169].

The goal of the instance selection is to obtain a subset of the training instances such that the performance using the selected instances is better or at least not significantly worst than the original ones [35]. As instance selection reduces the number of training instances, the training time might be decreased. Instance selection methods can be classified into incremental methods and decremental methods [144]. In the incremental approach, the selection starts with an empty subset and augments the selected instances incrementally by including instances from the training set during the selection process. However, the decremental methods start with the full training set and reduce them decrementally by removing instances along the selection [169]. Similar to feature selection, the instance selection methods can be categorised based on the evaluation criteria into wrapper, filter and embedded methods according to how a learning algorithm is involved [169].

Many wrapper methods are usually based on the k-NN classifier [53] such as the Condensed Nearest Neighbour (CNN) [96], Selective Nearest Neighbour rule (SNN) [196], Generalised Condensed Nearest Neighbour rule (GCNN) [50]. Unlike wrapper methods, filter instance selection methods are not based on a learning method for selecting the instances from the training set. The border instances of a class are useful for preserving the class discrimination regions [254, 35]; therefore selecting border instances is the main focus in many filter instance selection methods.

The use of clustering for instance selection is adopted in [159, 242, 243]. The main idea is to split the training data set into clusters and select the instances at the centers of the clusters. In [159], the Generalised-Modified Chang Algorithm (GCM) is presented. This method works by merging the same-class nearest clusters and then selecting the centres of the new merged clusters. In [242], a method called the Nearest Sub-class Classifier (NSB) is proposed by allowing the selection of different number of instances for each class using the maximum variance cluster method presented in [243].

Several EC-based methods have been used for instance selection [130, 129, 38]. The main idea is as follows. Provide individual representation such that each individual determines a set of instances and evaluate the individuals according to a fitness function (e.g. classification accuracy in wrapper instance selection context). For example, in GA-based instance selection, the individuals are represented as bit strings. The length of these individuals equals the number of instances. The bit with a "1" value indicates that the corresponding instances is selected and the "0" bit means it is not selected. The instances that correspond to 1-bits are then used to evaluate the individual by a learning algorithm. The best individuals are then selected and used to generate new ones using evolutionary operators. The process is repeated iteratively (generations) and the best individual from the final generation is selected. In [89], a memetic algorithm for instance selection and an approach named Clonal Selection Algorithm (CSA) is proposed.

2.1.5 Transfer Learning

Transfer learning (TL) is a learning paradigm that extracts knowledge gained while solving one problem/domain (called a source problem/domain) and uses it to solve a different but related problem/domain (called a target problem/domain). Three main questions need to be considered when developing a transfer learning method [174]:

1. What to transfer?

2. How to transfer?

3. When to transfer?

2.1.5.1 Transfer Learning Categorisation

Based on how the first question is addressed, transfer learning methods can be further categorised into four categories [250]: instance-based transfer learning, relational-based transfer learning, parameter-based transfer learning, and feature-based transfer learning. Instance-based transfer learning implies that the knowledge transferred from the source domain to the target domain consists of data instances. In relational-based transfer learning, this knowledge represents relationship between the data in the source and target domains. Parameter-based transfer learning works by transferring parameters that are share between the source and the target tasks. Finally, feature-based transfer learning is when the knowledge that is transferred are the learned feature transformations.

One way to transfer knowledge between different domains is by constructing feature-based transformations [174]. These transformations work by finding suitable feature representations to make different domains close to each other. The closeness can be measured based on the performance of the algorithm, the domain distribution mismatch, or both. Feature-based transformations can be symmetric or asymmetric [84]. The symmetric transformations map both domains to a third domain with a common feature space, and the asymmetric transformations map one domain to the other. When the feature space is similar in the different domains transfer learning is called homogeneous transfer learning and it is called heterogeneous otherwise. The transfer learning outcome is positive (negative) if it leads to a better (worse) performance compared with the case of learning in the target domain alone (i.e. traditional learning without transfer learning).

2.1.5.2 Transfer Learning with Incomplete Data

There are a few studies that employ transfer learning for handling the missingness issue. In [269], a transfer learning-based deep neural network imputation method is proposed to estimate missing gene expression values from DNA methylation data. This work is validated on a biological domain achieving a better performance than the case of not using transfer learning. In [248], a transfer-based additive least squares support vector machine (LS-SVM) classifier for handling missing values is proposed. It simultaneously identifies the contribution of each incomplete instance in the classification error using a fast leave-one-out cross validation strategy.

This method is evaluated on seven data sets showing at least comparable, if not better, results compared to the complete case strategy, mean imputation, and k-nearest neighbor imputation based on the standard LS-SVM and support vector machine classifiers performance. However, there are some notes on this work.

First of all, the source and target data sets are subsets of the original data, where the complete instances form the source domain and the incomplete ones represent the target domain. This setting may not guarantee the distribution difference between the domains as the missingness is synthetically imposed on instances that come from the same domain of the complete ones. Moreover, this artificial missingness was inserted in only the first three top ranked feature(s) based on wrapper and filter techniques, then modified their values to unknown. However, there are some issues.

It seems to misconsider the missingness mechanism, viz. MAR, MCAR, and MNAR, that affects the control of the experiment settings, and, subsequently, the validity of the obtained results. Another issue is the dependency on the ranking methods, which could be avoided by considering more features to impose the missingness.

2.2 Related Work

2.2.1 Symbolic Regression

Symbolic regression is a methodology that automatically generates mathematical models that describe relationships on given data.

There are several techniques that have been used to perform symbolic regression. Although GP has been considered as the dominating approach for symbolic regression, other EC-based methods have been also proposed. Moreover, several non-EC approaches have been utilised for developing symbolic regression methods.

2.2.1.1 EC-based Symbolic Regression

GP has been typically applied to solving symbolic regression problems [123, 124]. After many years GP is still by far the most dominating approach for symbolic regression. This is due to several several advantages of symbolic regression via GP including [247]

- No assumptions on the model structure are required.
- An implicit sensitivity analysis of the inputs and feature selection is performed.
- Independencey of input features is not pre-assumed.

Different variations of GP-based methods have been proposed for symbolic regression. Grammatical evolution (GE) is used for symbolic regression in [173]. The application of GE requires employing a search strategy (e.g. simulated annealing, hill climbing, random search, and GAs) and the performance relies on the used search strategy. In [260], analytic programming (AP) is used for symbolic regression. It is based on GP and Hilbert spaces aiming to address the representation of a symbolic model. The reported comparisons show that AP is equivalent to GP on the considered tasks. However, it needs to be expanded with more comparative analysis focusing on the complexity of the addressed problems.

A large-scale GP cloud computing system called FlexGP is presented in [210]. GP is improved via massive data-parallel ensemble learning and employed for symbolic regression on large-scale data sets on the cloud. FlexGP is a decentralised, fault tolerant parallelisation framework. It runs many multiple regression GP copies. Each copy is for a different data sample and different parameters. A fused model is created as these individual GP learners are evolving in an ensemble on demand manner. In [16], Multiple Regression GP (MRGP) is proposed by combining all possible subtrees of an individual tree using LASSO [220].

In [268], a methodology called self-learning gene expression programming (SL-GEP) for improving the efficiency and search accuracy is proposed. One of the main features of SL-GEP is to represent each chromosome with sub functions that can be used for constructing the final solution. Another important feature of the proposed SL-GEP is using differential evolution to develop a new search mechanism. SL-GEP is evaluated using fifteen symbolic regression problems. The experimental results show that SL-GEP outperforms several state-of-the-art algorithms in terms of the search efficiency and the accuracy of symbolic regression.

Vanneschi et.al. [238] presented a new method for population initialisation in GP named Evolutionary Demes Despeciation Algorithm (EDDA). This method is then applied to Geometric Semantic GP (GSGP). The performance of the presented method is evaluated experimentally on six complex real-life symbolic regression problems. On all the considered problems, the proposed initialisation technique has been shown to allow generating solutions with comparable or even better generalisation ability, than the ramped half-and-half method, with significantly smaller size of the solutions.

In [239], a parallel and distributed GP system called Multi-Population Hybrid GP (MPHGP) is presented. In this system, two sub populations run in parallel: one for a Multi Objectives GP (MO-GP) algorithm and the other sub population for Geometric Semantic GP (GSGP). The two evolutionary processes run simultaneously and the individuals are migrated among them based on a predefined synchronisation time. The proposed system MPHGP is evaluated on five symbolic regression real-life applications, showing its ability to get the advantages of both MO-GP and GSGP as well as mitigating the problems of the two methods.

In [181], a new general GP approach is developed to solve recursive nature problems [180]. One of the methods is for solving common recursive problems (GPCR) and the other is domain independent GP method that aims to solve generic problems with recursive structures (GPGR). The performance of the new method is compared against nine algorithms on sixteen different problems of 4 different types where one of them is symbolic regression. Kushida [131] proposes a modified CAP to introduce symbiotic relationship between aphids and ants. The performance of the proposed method is examined on symbolic regression problems.

In [192], a hybrid method for symbolic regression is proposed by combining a variation of GP called Kaizen programming and relevance vector machine. Unlike traditional EC algorithms where a single individual is a complete solution, this method proposes a solution based on linear combination of basis functions built from individuals during the evolving process. Basis functions are essential functions that can be used to composite new functions.

Several studies focus on using geometric semantics in GP proposing different operators such as semantic-aware initialisation, search, and selection. Chen et.al. [43] investigated the geometric crossover with angle-awareness and its impact on improving the learning ability performance, generalisation performance, the program size, and the computational cost. For evaluation, six popular symbolic regression benchmark problems are considered and the results showed that the proposed method has better learning performance and more generalisation ability.

In [12], a continuous neural encoding approach is proposed to improve linear representation in GP for symbolic regression. A Model-based GP approach is proposed for symbolic regression of small expressions in [245]. This work uses a gene-pool optimal mixing evolutionary algorithm (GOMEA) model-based EA framework for symbolic regression. In [31], Zoetrope GP (ZGP) algorithm is proposed based for symbolic regression. ZGP representation employs repeated fusions starting by partial expressions and gradually generating more complex expressions to end up with new features. The produced features are then combined to fit the given data in linearly. In [212], standard gradient boosting is modified by replacing the embedded weak learner in favor of a strong(er) one to present symbolic-regression boosting (SyRBo). Experimental results on about 100 regression data sets show that adding a small number of boosting stages—between 2 and 5—to a symbolic regressor, significantly improves the symbolic regression performance.

Bloat is a problematic issue in GP, where unnecessarily large programs are evolved without much gain in the fitness. This causes excessive use of computing resources (e.g. processor and memory) and evolves models with poor generalisation ability. To address this issue, a multi-objective (MO) technique is combined with local search for symbolic regression in [140]. In [128], adding constraints on the function and its derivatives for the incorporation of prior knowledge in symbolic regression is investigated. This is done by proposing two algorithms. The first algorithm is an extension of tree-based GP, where infeasible solutions are discarded in the selection step. The second one is a two population evolutionary algorithm that separates the feasible from the infeasible solutions. The methods are evaluated on a set of synthetic and real-world symbolic regression tasks. Shape-constrained polynomial regression produces the best results for the test set but also significantly larger models.

EC-based method called an artificial immune system (AIS) is used for symbolic regression [107]. AIS is similar to GP as the programs are represented as trees, however different components of the evolutionary process of GP are translated into the immune metaphor. The experimental results show that the AIS method typically converges slightly quicker than the standard GP method. However, the size average of trees evolved using AIS is larger than that of using GP.

In [135], a variation of Grammatical Evolution (GE) is proposed by incorporating imitation learning. This study presented an introduction to social learning and evaluated the proposed method on a set of benchmark symbolic regression problems. The results obtained are promising comparing to canonical GE. To improve the existing algorithm, meta-frameworks of social learning could be used in the design of new search algorithms. In [132], Feature Engineering Automation Tool (FEAT) is developed. FEAT is a stochastic optimisation that provides a representation with explicit variable dependencies. Recently, a method called Interaction-Transformation Evolutionary Algorithm (ITEA) is proposed in [58]. This method constructs generalised additive models that include interactions between features.

2.2.1.2 Non-EC Methods for Symbolic Regression

A non-EC method called Fast Function Extraction (FFX) is proposed for symbolic regression in [155]. FFX uses a path-wise regularised learning technique for pruning a large set of candidate basis functions down to a smaller compact set of models. In [41], another non-EC real-time algorithm for symbolic regression is proposed. This algorithm is called Elite Bases Regression (EBR) and it works by generating a set of candidate basis functions coded with parse-matrix by specific mapping rules. Some of the elite bases are updated iteratively based on the correlation with the target model. The regression model is then spanned by the elite bases. A comparison between EBR and FFX is conducted and the reported results indicate that EBR is able to solve symbolic regression problems more effectively. However, several control parameters should be tuned in the EBR method. Moreover, the method needs to be modified to improve its applicability to complicated real-world applications (e.g. data with nonlinear correlation).

Deep learning is arguably the most active and successful machine learning field in the last decade. Research in deep learning led to revolutionary advances in different AI areas demonstrating ground-breaking performances on many learning tasks such as computer vision and natural language processing. Conversely, due to its lack of interpretability, deep learning was not a favourable approach for symbolic regression. However, recently, the success story of deep learning has been extended to symbolic regression. Lample and Charton [133] show that neural networks can be successfully utilised for tasks in mathematics, such as solving symbolic integration and differential equations. They propose a syntax for representing mathematical problems, and methods for generating large data sets that can be used to train sequence-to-sequence models. In [54], a general approach, based on Graph Neural Networks (GNNs), to purify symbolic representations of the learned deep models is developed by introducing strong inductive biases.

In [114, 113], a demonstration system is presented for deep symbolic regression based on reinforcement learning. The system visualises the best discovered expressions during training and allows the user to guide the process by changing hyper-parameters and selecting expressions manually. In [29], a fully-convolutional sequence-to-sequence model is applied to discover symbolic equations from numerical data. The goodness of this approach is shown by both a qualitative and a quantitative analysis on a large set of mathematical expressions. In [115], a neural network-based architecture called the equation learner (EQL) network is used for symbolic regression. The EQL architecture is integrated with other deep learning architectures such that the whole system can be trained end-to-end through backpropagation. To show the effectiveness of such systems, different symbolic regression tasks are conducted.

In [36], a supervised machine learning tool called Feyn is introduced for symbolic regression. The simulation engine that powers this tool is called QLattice. QLattice is a supervised machine learning tool inspired by Richard Feynman's path integral formulation, that explores many potential models to solve a given problem. QLattice formulates these models as graphs that can be interpreted as mathematical equations, allowing the user to completely decide on the trade-off between interpretability, complexity and model performance.

2.2.2 Evolutionary Computation-based Imputation

Evolutionary computation algorithms have been used for estimating missing values [222]. The used techniques include GAs, particle swarm optimisation, GP, and hybridisation between EC and non-EC methods.

There are several GA-based imputation methods for classification on incomplete data. In [88], GA is used to impute missing values in multivariate data. In [147], a multi-objective GA is used to build an imputation method which can deal with both numeric and nominal features. In [185], a GA is used to impute missing values in discrete features by searching for the most suitable value for each missing value. In [176], an imputation method based on GA that uses domain values for the feature as pool of solutions is proposed. The fitness function is the classification accuracy on decision tree classifier. In [148] a GA-based imputation method is proposed for pattern classification on incomplete data.

In [81], GAs for missing value imputation in time series is proposed. It minimises an error function derived from considering the auto-correlation, the mean, and the variance. GA-based imputation for missing data in time series is also presented in [81, 87]. In [57], an imputation method using GA for clustering is presented. This method is called EACImpute and it is based on the assumption that clustering can be useful for the imputation purposes.

In [91], two imputation methods are proposed by combining PSO, autoassociative extreme learning machine (AAELM), and evolving clustering method (ECM). PSO which simultaneously minimises two errors is used for the selection of the values. The first error is the mean square error between the covariance matrix of the complete data and the covariance matrix of all data including the imputed ones. The second one is the absolute error between the determinants of the two covariance matrices. In [203], a hybrid imputation method combining PSO and Fuzzy C-Means (FCM) is implemented. The FCM is used to extract the similar complete instances. Then, PSO is applied to optimise the records based on information from the incomplete instances. In [202], a Fuzzy Swarm imputation method is proposed. A fuzzy-based clustering of the complete instances is firstly applied as a preprocessing stage to fill in the missing values, and then PSO is used to optimise the imputation. These methods showed better performance than popular non-EC methods.

2.2.2.1 GP for Incomplete Data

GP-based imputation methods have been successfully used for classification. A GP-based multiple imputation method that utilises the symbolic regression prediction ability of GP to estimate missing values in classification data sets is proposed in [223]. To impute incomplete test instances, the whole data (including the training set) are used to build regression models whose target is the feature that contains missing values. The model with the smallest fitness value is then applied to estimate the missing values. This process is repeated for each incomplete feature. In [225], the GP-based imputation is modified to have two separate processes; the training process and the test process. In the training process, imputation regression models are constructed using the training data set. The test process is then performed to impute single test instances by applying the constructed imputation regression models. In [230], multiple imputation and GP are combined to evolve classifiers on data with missing values. Common patterns of missing values are firstly extracted and GP is then used to construct a classifier for each pattern.

2.2.2.2 Interval-valued GP for Incomplete Data

Interval arithmetic is a technique that enables calculating arithmetic operations (e.g. addition) on interval-valued elements instead of single-valued numbers. With interval arithmetic, GP outputs intervals rather than point predictions in [205]. In [110], interval-valued GP (IGP) is used to make sure that the models produced by GP do not result in undefined values. In [65, 64], IGP is extended with interval-aware search operators to reduce the number of invalid solutions obtained during the search process. Recently, some attempts have been taken for utilising IGP in dealing with incomplete data.

In [226], GP-based multiple feature construction (GPMFC) is successfully extended by using IGP to directly construct features for classification on incomplete data. However, this method did not solve the problem of incompleteness directly and the classifiers still need to be able to classify incomplete data. The classifiers used in [226] are tree-based classifiers that have the ability to implicitly select the features while constructing the models. They compared their proposed method with the approach of combining imputation and feature selection. Their method showed better performance than the simple imputation of using the mean value but not better than the case of using more advanced imputation method (e.g. multivariate imputation by chained equations, MICE). In [224], IGP is utilised to directly construct classifiers for incomplete data. The results show that IGP has more effectiveness and efficiency than the case of combining traditional GP with imputation. Moreover, it outperforms some popular classifiers that can directly classify data with missing values such as C4.5 and CART. IGP is improved in [232] by incorporating ensemble learning.

Existing IGP-based methods are specifically designed for classification and did not achieve acceptable results when we tried to adopt them for symbolic regression on incomplete data. This is because the designed fitness functions are based on comparing specific points of the produced IGP intervals with the desired class labels. Therefore, there is a need for a new error measure that fits the symbolic regression case. Besides, the feature selection, which can bring benefits to the symbolic regression performance, is not considered in these methods.

2.2.2.3 Symbolic Regression on incomplete data

In symbolic regression research, the most common strategy to deal with incomplete data is to delete the instances having missing values [63, 74, 48]. However, there are some studies that uses different approaches. The application of evolutionary system combining GP with Evolutionary Strategy (GPAES) for symbolic regression on incomplete data is presented in [34]. The method employs many heuristics. Evolutionary strategy is used in each GP cycle for each individual for parameters optimisation. The experimental results are obtained using synthetic data using a dynamic model called the Lorenz attractor system. The main limitation of the presented method is the enormous computational complexity.

In [246], the missingness is treated as having imbalanced data in certain regions of mathematical functions. A framework for automatic weighting of data samples is suggested taking into account the relative importance of the samples. This importance is by four schemes related to proximity, remoteness, surrounding, and nonlinear deviation from nearest neighbours. The methods are used to balance synthetic data drawn from mathematical functions. The applicability of this methods should be validated on real-world incomplete data.

As far as we know, the only studies that consider imputation for symbolic regression are [178, 34]. However, they have some limitations. [34] considered only artificial functions, while in [178], missing values are simply replaced with corresponding feature values from other instances.

Most of the proposed EC-based imputation methods are developed for the classification task. There is no much work on symbolic regression. Moreover, the existing imputation methods might not be applicable to symbolic regression effectively as they are developed for other tasks (e.g. classification). Therefore, more investigations should be done for symbolic regression tasks.

2.2.3 High-dimensional Symbolic Regression

Dealing with high-dimensional data is still a challenge, limiting the practical applicability of symbolic regression on several real-world problems. Therefore, several studies have been conducted on mitigating the dimensionality issue in symbolic regression.

In [154], symbolic regression on data sets with hundreds of features is addressed using non-traditional GP technique. This technique is designed based on GP, latent variables, and nonlinear sensitivity analysis. The technique is verified on 24 circuit modeling problems. There is a need to validate such methods on different real-world problems with higher dimensions. In [214], the requirements for feature selection on high-dimensional data sets with respect to various feature importance aspects are explored using symbolic regression and random forests. Feature selection is validated on several problems to identify the conceptual differences in the generated feature importance.

For high-dimensional symbolic regression, a two-stage feature selection method is presented in [44]. The evolutionary process is split into two phases. The first phase provides a set of candidate important features. This set is then used in the second phase to improve the generalisation of GP models. With the same goal, a feature selection method is proposed in [48]. This method works by integrating a permutation measure, used in random forest regression, to obtain the importance of features that appear in the GP models. A similar approach is proposed in [66], where the permutation measure is employed for feature selection based on the influence of a feature within a GP model.

In [49], a feature construction method is developed based on GP for high-dimensional symbolic regression. This method is named GP with embedded feature construction (GPEFC). In GPEFC, new informative building blocks on fitter individuals are tracked and used to construct new features that dynamically augment the terminal set of GP. The reported experimental results showed that GPEFC is able to evolve more compact models efficiently. Moreover, it outperforms the standard GP with respect to both generalisation performance and learning ability. However, effect of the detected building blocks and the relationship between the generalisation ability and the new features has not been investigated.

In [121], deep learning-based feature selection is utilised in classification based on symbolic regression process. Feature selection is handled in the deep learning layer by performing a series of individual linear discriminant analysis including sequential minimal optimisation (SMO) and bees algorithm enhancements. However, none of these studies considers incomplete data in symbolic regression.

As can be seen, the existing methods to mitigate the high-dimensionality issue in symbolic regression use complete data sets. These methods might need to be adapted for dealing with incomplete data. Moreover, new methods are required for handling the missing values in addition to the high-dimensionality when performing symbolic regression on high-dimensional incomplete data.

2.2.4 Feature Selection and Instance Selection on incomplete data

2.2.4.1 Feature Selection on Incomplete Data

Feature selection is an important preprocessing step in machine learning but it is usually performed on complete data. However, there are only a small number of feature selection methods that can be applied to data sets with missing values.

In [19], a method for selecting features for classification with incomplete data sets is introduced. The method works by determining the Markov blanket of the target variable, which reflects the worst case assumption of the missingness mechanism. In [149], a method that performs feature selection directly without imputation is proposed for classification based on maximising each instance's uncertainty margin in its own relevant subspace. A method that simultaneously carries out classification and feature selection on incomplete data in an online manner is presented in [109]. For classification 1-NN with a Gaussian mixture model is used. In the application stage on test data, the missing attributes are discarded and the available ones are ranked by feature selection based the trained model.

In [187], mutual information is combined with rough sets to select features from incomplete data. In most considered cases, the proposed algorithm showed more effective feature selection results compared with existing feature selection algorithms on different real-world incomplete data sets. In [61], a feature selection and classification method is proposed for incomplete multi-modal high-dimensional data. The method is shown to produce better performance than the complete case strategy on incomplete multimodal data.

In [37], an algorithm based on feature selection and partial distance strategy is proposed for clustering on incomplete high-dimensional big data and reported results better than existing clustering methods in both accuracy and computation time. In [266], a framework for selecting features on incomplete data is introduced. It starts with relieving the influence of outliers and then an indicator matrix is employed to avoid unobserved data to take participation in numerical computation of feature selection. Experimental results show that the proposed method outperformed existing feature selection methods in terms of clustering performance on both real and artificial incomplete data sets.

In [227, 228], PSO is combined with a classifier that has the ability of classifying incomplete data (e.g. C4.5) for feature selection on incomplete data. PSO is used to search feature subsets and the classifier is used to evaluate these subsets. This work is extended by integrating feature selection with bagging in [229]. The experimental results show that combining bagging and feature selection provides better classification accuracy than three methods that are using C4.5/REPTree as classifiers and PSO as a feature selection method. Moreover, it produces less complex models compared to the case of bagging without feature selection.

In [231], an approach that integrates imputation, clustering and feature selection for classification on incomplete data is proposed. The number of instances used by the imputation is reduced by clustering, while relevant features are selected using differential evolution (DE). However, feature selection is performed after imputing the missing values in the training data, which implies the stake of unnecessary overhead of imputing irrelevant features, especially with high-dimensional data.

Some feature selection methods can be applied directly to incomplete data sets for classification tasks [19, 149, 109]. In [209], a hybrid method is proposed for missing data imputation. The method is based on a fuzzy c-means, regression, and mutual information. For missing values imputation, mutual information is used to select the features in each cluster. On the other hand, the impact of feature selection on incomplete medical data imputation is studied in [143]. These studies concluded that the feature selection reduces the learning time and improves the imputation accuracy.

2.2.4.2 Instance Selection with Incomplete Data

Instance weighting is employed for transfer learning in GP for symbolic regression in [45, 46, 47]. In [45], an implicit instance selection is performed to transfer data between different yet related domains. This method is extended in [47]. However, all these methods consider only complete data. In [11], principal component analysis (PCA) is used as an instance selection mechanism and integrated with fully conditional specification and

Markov chain Monte Carlo (MCMC) for multiple imputation.

In [80], a locality constrained sparse representation is used for instance selection for data imputation. In [234], the impact of instance selection on data imputation is investigated. They presented and compared four different methods for combining instance selection and missingness imputation. In the first strategy, imputation is performed then then instance selection is conducted. In the second strategy, the imputation is done after performing instance selection. The third and fourth strategies carry out one more instance selection after the first two strategies. The results suggest that performing instance selection first then imputation outperforms the other three methods. In [194], different imputation methods are reviewed including evidence chain estimation, class centre-based approach, decision tree, and random forests, combining instance selection, fuzzy rough set, and stochastic semi-parametric model.

An investigation on the impact of instance selection on the imputation performance is also investigated in [102]. This is done by considering the algorithms DROP3, IB3, and GA, for instance selection and the algorithms KNNI, MLP, and SVM, for imputation. The data is pre-split into complete and incomplete data sets and then the selection is performed based on the complete data. The results show that instance selection can improve the imputation over numerical and mixed data but not pure categorical data. However, restricting the selection on complete data only prevents learning from the knowledge related to the missingness in the training data to improve imputing new incomplete test data.

2.2.4.3 Feature Selection and Instance Selection on incomplete data

In [52], a method based on the co-evolution approach is proposed to perform both instance selection and feature selection. Instance selection and feature selection are treated as independent subproblems. This method is developed in a wrapper manner, which utilises a random forest classifier in the co-evolution process to remove less relevant features and instances, thereby improving the overall performance.

The lack of studies that consider performing data pre-processing before imputation is highlighted in [101]. In this regard, their study aims to examine the impact of two pre-processing steps, namely, feature selection and instance selection, on data imputation. IB3 ([2]) is used for instance selection due to its efficiency while providing reasonable performance.

From our review above, there is no studies considered using both feature selection and instance selection for symbolic regression on incomplete data. Therefore, we present this thesis to fill this gap.

2.2.5 Transfer Learning for GP and Symbolic Regression

Evolutionary transfer learning has gained a rapid interest recently [267]. One approach to transfer learning with GP is to extract knowledge from the evolutionary process in the source domain to be utilised in improving the evolutionary learning process in the target domain. This knowledge can be related to the population such as full trees [69, 98] or subtrees [69, 98, 170, 14]. Another kind of knowledge is related to the feature weights as in [15]. In [105, 103], the transferred knowledge is represented as code fragments. This approach is related to layered learning and population seeding [163].

In [86], several classifiers are learned from multiple source domains and then utilised to vote for document classification in the target domain. In [46], the source instances are weighted based on their impact on the target GP-based symbolic regression (GPSR) learning task. Another instance-based transfer learning method for GPSR is presented in [45], where the optimal weights for source domain instances are obtained using differential evolution search. This method is time-consuming when the source domain has a large number of instances. The work in [45] is extended in [47], where a more effective and efficient instance weighting framework for transfer learning in GPSR is proposed. Meanwhile, a distribution estimation is utilised for providing better starting points for the search process.

In [163] and [233], GP-based feature transformations evolved in a source task are transferred to a target task. These transformations are constructed using multidimensional multiclass GP with multidimensional populations [162]. The incompleteness issue is referred to as a possible source of an important impact, however, it was not considered in their study.

In [68], several transfer learning methods for GP are proposed. The main idea is to transfer a set of good trees or sub-trees from the source problem to the target one. The methods are evaluated using symbolic regression tasks and the use of transfer learning methods improved the GP performance on both training and unseen data. Furthermore, the code growth problem in GP is reduced. Such methods do not handle data problems such as high-dimensionality and incompleteness.

A transfer learning method for GP called Common Subtrees from Related Problems (CSRP) is proposed in [170]. This method is dynamic, non-random, and broadly applicable. The evaluation of the proposed method is conducted experimentally on symbolic regression problem and Boolean domain problems. The reported results show that CSRP significantly outperformed the baseline methods on most considered symbolic regression problems. The CSRP method should be expanded and another methods should be implemented and tested to address different tasks (e.g. Boolean problems). Moreover, some of the investigations on CSRP should be addressed such as selecting optimal weights of the subtrees and how general the method. This in turn requires testing CSRP on more problems especially those utilising real-world datasets, multi-variable symbolic regression problems, and symbolic regression problems using non-commutative operators.

In [97], the ability of GP with transfer learning is investigated on different transfer scenarios. The impact of transfer learning on the evolutionary training process is investigated along with an analyses of how to utilise the knowledge learnt from the source domain for improving the learning process on the target domain. The reported results show that GP with transfer learning has encouraging performance on different transfer scenarios. The transferred knowledge is helpful in providing a good initial population for the target domain, speeding up the convergence, and obtaining better final solutions. However, these advantages varies in different scenarios.

In summary, transfer learning has recently received an increasing interest in the GP community. Some of the published works adopt the transfer learning paradigm to address the case of having a small number of instances in symbolic regression tasks. However, these methods deal with only complete data.

2.3 Summary

The key basic concepts of topics in this thesis are presented in this chapter. A discussion of the machine learning tasks, evolutionary computation, incomplete data, feature selection, instance selection, transfer learning, and related works are provided. As GP and symbolic regression represent core components of this thesis, this chapter provided a detailed explanation of them. This chapter also discusses proposed methods on symbolic regression and analyses the advantages and limitations of those methods.

In the literature, only a few studies have been conducted on symbolic regression with high-dimensional and large-scale data. However, none of them considered the incompleteness issue. In fact, symbolic regression on incomplete data has not been studied adequately even on small data sets. Therefore, how to deal with missing values in symbolic regression is still an open research question.

In light of the literature, there are some limitations in the related work that need to be addressed. These limitations form the motivations of this thesis and they can be summarised as follows:

- Imputation is the most effective approach to deal with missing data in machine learning, however, it has seldom been studied in symbolic regression with incomplete data. Although many imputation methods (some are GP-based methods) have been developed for handling the missingness issue, they focus mainly on classification tasks. There is no well-established study to investigate the utilisation of data imputation in symbolic regression with incomplete data.

- Instance selection is one of the classic techniques to deal with data quality issues (e.g. noise). It has been successfully used for classification with incomplete data. However, it has not been used for symbolic regression on incomplete data.

- Feature selection has been successfully used to mitigate the high-dimensionality issue. GP has a powerful implicit feature selection ability, but it has not been considered as a feature selection method for symbolic regression on incomplete data. Furthermore, the high-dimensionality issue has been barely investigated in symbolic regression on incomplete data.

- Transfer learning has been an emerging paradigm to improve learning in domains with limited knowledge. Despite the success story of utilising transfer learning to compensate for the lack of data in different tasks, no existing work in transfer learning considers the data incompleteness in symbolic regression.

The following chapters of this thesis are dedicated to showing how we utilised GP to address these issues.

Chapter 3

Genetic Programming-based Imputation for Symbolic Regression on Incomplete Data

3.1 Introduction

One popular approach to handle incomplete data is data imputation. Imputation works by estimating the values of the missing data [202]. The imputed data can then be used by any learning algorithm. Existing research on dealing with missing values focuses mainly on classification tasks. Many imputation methods have been developed for classification on incomplete data [90]. However, in symbolic regression, only a few studies consider imputation on incomplete data [178]. This chapter aims to fill this gap by developing an imputation method to handle the missing values in symbolic regression tasks. GP-based imputation (GPI) is adopted successfully for classification on incomplete data in [223, 225, 230]. This chapter shows how GP-based imputation can be utilised for symbolic regression on incomplete data.

Data imputation can be done based on instance similarity basis. A popular method that follows this approach is KNN-based imputation. Another way to impute the missing values is based on utilising regression algorithms to predict the incomplete features. This approach is based on feature predictability. An example of this approach is GP-based imputation. Each of these approaches has the advantage of utilising one aspect of the data. The first one utilises the relationships of the instances and the second one employs the relationships between features. However, each one undermines the other factor.

The design of the proposed method in this chapter aims at combining both the instance-wise similarity approach and the feature-wise predictability approach. It utilises the KNN-based instances proximity, while employing the GP-based feature regression predictability. This strategy increases the accuracy of estimating the missing values, which in turn improves the modelling of the symbolic regression.

3.1.1 Chapter Goals

The main goal of this chapter is to develop a new imputation method to improve the performance of symbolic regression on incomplete data. This method works by combining weighted K nearest neighbour (WKNN) and GP. WKNN increases the contributions of instances which are close to the incomplete instances when building the imputation models. Meanwhile, GP is used to build imputation models to estimate the missing values using the instances retrieved by WKNN. Specifically, the objectives of this Chapter include:

1. Developing a hybrid imputation method by combining feature-based prediction using GP and instance-based similarity using WKNN;

2. Utilising the proposed method for symbolic regression on incomplete data with different missingness scenarios; and

3. Investigating whether the proposed method can outperform state-of-the-art imputation methods with respect to the imputation accu-

racy, the symbolic regression performance, and the imputation time.

3.1.2 Chapter Organisation

This chapter is organised as follows. In Section 3.2, the proposed method is presented with a description of how it is used for symbolic regression on incomplete data. After that, the experiment design is presented in Section 3.3. The experimental results are given and analysed in Section 3.4. Finally, Section 3.5 draws the conclusions of this chapter.

3.2 The Proposed Method

In this section, the details of the proposed method are presented in terms of the overall design, the training process, and the test process.

3.2.1 The Overall Design

Two main approaches for imputation are data-driven and regression-based [90]. In data-driven imputation, the relationships between the data instances are explored to fill in the missing values. The data-driven approach is powerful when there is a proximity similarity between the instances in the data set. However, it does not adequately utilise the feature-wise relationships. On the other hand, the regression-based imputation predicates the missing values in an incomplete feature using regression models built based on other features. This approach is successful in employing the regression predictability between features. However, it uses all available instances without considering their closeness to the incomplete instances.

In this chapter, a new imputation method is designed to be both data-driven and regression-based. Such a combination is intended to gain the advantages of each approach and mitigate the disadvantages of both. The imputation models are generated by GP using instances selected by WKNN. Therefore, this method is called weighted KNN-GP (WKNN-GP). The WKNN-GP method has two processes: the WKNN-GP training process and the WKNN-GP test process.

Figure 3.1 shows the overall design of the proposed method. As it shows, the input of the WKNN-GP training process is an incomplete training data set while the outputs are sets of imputation models and a complete imputed training set. After the construction of imputation models, the WKNN-GP test process uses these models to impute the missing values in the test set and produces a complete imputed test set.

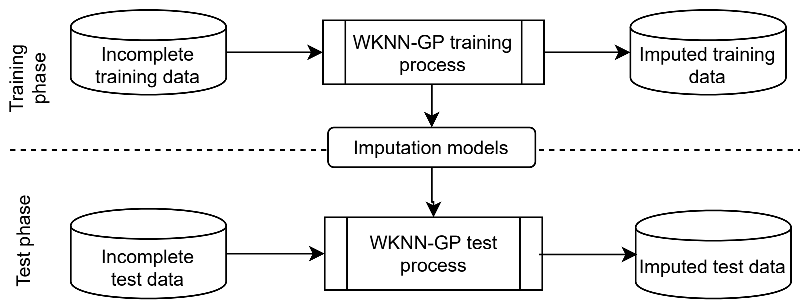


Figure 3.1: The overall design of using the proposed WKNN-GP method for imputation.

3.2.2 The WKNN-GP Training Process

3.2.2.1 The main idea

The training process aims to provide imputation models for the missing values in a given incomplete data set. There are two main steps for creating these models. The first step is to utilise weighted KNN (WKNN) to extract K nearest instances to the incomplete instance. The second step employs GP imputation to build prediction models using the retrieved K instances. These models are constructed by fitting the features with missing values (i.e. the imputation target) and involving the weights of the K instances in the GP fitness function. In WKNN, the similarity between instances is measured using the normalised Euclidean distance given in Equation (3.1).

$$\begin{aligned} distance(\mathbf{p}, \mathbf{q}) &= \sqrt{\frac{(q_1 - p_1)^2}{r_1^2} + \frac{(q_2 - p_2)^2}{r_2^2} + \cdots + \frac{(q_n - p_n)^2}{r_n^2}} \\ &= \sqrt{\sum_{i=1}^{n} \frac{(q_i - p_i)^2}{r_i^2}}, \end{aligned} \tag{3.1}$$

where $\mathbf{p}$, $\mathbf{q}$ are real-valued n-dimensional instances, and r_i is the range of the i^{th} feature.

The use of WKNN gives more importance to the instances that are close to the incomplete ones. The closer the instance, the higher contribution. On the other hand, GP is used to generate imputation models due to its successful application for classification on incomplete data [223, 225].

Example 1. *To clarify the high-level steps of the proposed method, a simple example is given. This example is also helpful to explain the detailed algorithm later. Considering the incomplete data set shown in Table 3.1 (the missing values are represented as ?), two main steps are applied to perform WKNN-GP imputation for the missing value at* $(i, j) = (1, 2)$*, i.e. the value of the feature* F_2 *in the instance* I_1*.*

Firstly, the existing values in the first instance are extracted to form a complete instance. WKNN is used to get its K *(let* $K = 2$*) nearest instances. The output of this step is shown in Table 3.2. After that, GP is used to build imputation models using the data in Table 3.2 as input instances and the corresponding values of the* 2^{nd} *feature as target values. In other words, GP constructs imputation models for a new problem whose data set is shown in Table 3.3. The best constructed GP model is then used to estimate the missing value at* $(1, 2)$*. This model is also used to impute similar instances such as* I_{11}*.*

Table 3.1: An incomplete data set.

Instance	F_1	F_2	F_3	F_4	T: regression target
I_1	300	**?**	1300	?	33.7
I_2	310	10	1200	12	35.87
I_3	200	40	1700	?	30.73
I_4	?	30	1100	15	39.17
I_5	320	?	1600	13	39.11
I_6	300	60	1400	17	35.01
I_7	200	40	1200	11	29.77
I_8	290	80	1400	11	31.1
I_9	?	?	1000	14	39.7
I_{10}	230	?	1800	?	37.3
I_{11}	290	?	1400	?	30.1
I_{12}	280	90	1500	15	31.3

Table 3.2: Complete sub-instances: 2-nearest instances to the first instance.

$Instance$	F_1	F_3
I_2	310	1200
I_6	300	1400

Table 3.3: The selected 2-nearest instances with the new target variable.

$Instance$	F_1	F_3	Imputation target (from F_2)
I_2	310	1200	10
I_6	300	1400	60

3.2.2.2 The training algorithm

The pseudocode of the training procedure is shown in Algorithm 2. It receives an incomplete training data set $\mathcal{X}$ and outputs different sets that are used to capture the imputation models for the missing values.

Algorithm 2: WKNN-GP Imputation: The Training Algorithm

```
Input  : Training data set $\mathcal{X}$ with missing values.
Output : Complete data set $\mathcal{X}^C$, $G$: GP imputation models, $R$: reference set, $P$:
         missingness patterns, and $S$: statistics of the training features.
1  Let $G = R = P = D = S = \phi$;
2  foreach missing value $\mathcal{X}[i, j]$ do
3      Obtain the corresponding missingness pattern $P_{i,j}$;
4      Extract the non-missing values from the $i^{th}$ instance to form a complete
         instance $V_{i,j}$;
5      if ($\exists P_{\hat{i},j} \in P$ s.t. $P_{i,j} = P_{\hat{i},j}$) and ($\exists R_{\hat{i},j} \in R$ and $D_{\hat{i},j} \in D$ s.t.
         $distance(R_{\hat{i},j}, V_{i,j}) \leq D_{\hat{i},j}$) then
6          $\mathcal{X}^C[i, j] \leftarrow G_{\hat{i},j}(V_{i,j})$
7      else
8          Use $V_{i,j}$ as a new reference instance $R_{i,j}$ ($R_{i,j} = V_{i,j}$) ;
9          Get a data subset $\mathcal{X}_{i,j}^{tmp}$ for the features included in $P_{i,j}$ after excluding
             the instances that have missing values at the $j^{th}$ feature;
10         Obtain $\mathcal{X}_{i,j}$ as a complete data subset by taking the complete instances
             from $\mathcal{X}_{i,j}^{tmp}$;
11         $K \leftarrow min(max(|J_{i,j}|, \lfloor |I_{i,j}|/4 \rfloor), |I_{i,j}|)$, where $I_{i,j}$ and $J_{i,j}$ are the
             instance and feature indexes of $\mathcal{X}_{i,j}$, respectively;
12         $\mathcal{X}_{i,j}^K, \mathcal{D}_{i,j}^K, \mathcal{W}_{i,j}^K \leftarrow KNN(\mathcal{X}_{i,j}, R_{i,j}, K)$, where $\mathcal{X}_{i,j}^K$ contains the $K$ nearest
             instances, $\mathcal{D}_{i,j}^K$ is the corresponding distance set, and $\mathcal{W}_{i,j}^K$ is the
             corresponding distance-based weights set w.r.t the reference $R_{i,j}$;
13         Set the imputation target as $T_{i,j} \leftarrow \mathcal{X}[I_{i,j}^K, j]$, where $I_{i,j}^K$ is the instance
             indexes of $\mathcal{X}_{i,j}^K$;
14         for $g = 1$ to $N$ do
15             $G_g^{tmp} \leftarrow$ GP($\mathcal{X}_{i,j}^K$, $T_{i,j}$, $\mathcal{W}_{i,j}^K$);
16         end
17         Let $G_{\hat{g}}^{tmp}$ be the regression function with the least fitness value, i.e.
             $fitness(G_{\hat{g}}^{tmp}) \leq fitness(G_g^{tmp}), g = 1, ...N$;
18         Set $G_{i,j}$ to be $G_{\hat{g}}^{tmp}$ and use it to impute $\mathcal{X}[i, j]$ (i.e.
             $\mathcal{X}^C[i, j] \leftarrow G_{i,j}(R_{i,j})$);
19         $D_{i,j} \leftarrow \max \mathcal{D}_{i,j}^K$;
20         Append $G_{i,j}, R_{i,j}, D_{i,j}$, and $P_{i,j}$, to $G, R, D$, and $P$, respectively.
21     end
22     Calculate $S$ the statistics of $\mathcal{X}^C$ (the mean for the numerical features and the
         mode for the categorical features);
23 end
```

The set G is for the constructed GP-based imputation models. R is a set of complete reference sub-instances extracted from the incomplete instances under processing. P is a set of missingness patterns representing different forms of incompleteness. The set S contains the means of the numerical features and the modes of the categorical features.

For each missing value, the missingness pattern of the containing instance is obtained. The missingness pattern consists of a bit string with the same number of features, where the missing feature is encoded as 0 while the observed is 1. The complete values in the instance are extracted and compared with the already constructed references of models with the same pattern. If a similar model is found, the corresponding imputer is used to estimate the missing value. Otherwise, KNN is used to obtain the K nearest complete instances with their weights according to the normalised Euclidean distance. The obtained instances are used to build GP imputation models involving the weights of instances in the GP fitness function. The GP imputation models are built by considering the feature containing the missing value as a regression target. Finally, the best model (i.e. the one with the best fitness value) is selected.

3.2.2.3 Data preparation

For each missing value at position (i, j), a missingness pattern $P_{i,j}$ is formed to reflect which features have missing values in the i^{th} instance. Such patterns are bit strings formed by putting zeros at the positions of missing values and ones for the existing values. Non-missing values are extracted to form a complete instance $V_{i,j}$. This instance is compared with the existing reference instances in the set R that have the same missingness pattern.

If there is a similar $R_{\hat{i},j}$, the corresponding imputation model $G_{\hat{i},j}$ is used to impute $\mathcal{X}[i, j]$ directly, i.e. $\mathcal{X}^C[i, j] = G_{\hat{i},j}(V_{i,j})$. Otherwise, new models are constructed and $V_{i,j}$ is used as a reference instance $R_{i,j}$. The features included in $P_{i,j}$ are used to get a data subset $\mathcal{X}^{tmp}_{i,j}$ after removing the instances that have missing values at the j^{th} feature. A complete data subset $\mathcal{X}_{i,j}$ is then formed by including only the complete instances from $\mathcal{X}_{i,j}^{tmp}$.

In Example 1, the missingness pattern for the missing value $\mathcal{X}[1,2]$ is $P_{1,2} = 1010$, where "1" indicates feature with existing value while "0" indicates the missingness. From the 1^{st} instance, extract the complete instance $V_{1,2}$. As there are no previously learned imputation models, no need to compare $V_{1,2}$ with the reference set R and $V_{1,2}$ is used as a new reference instance $R_{1,2}$ ($R_{1,2} = V_{1,2}$). This vector is shown in Equation (3.2).

$$V_{1,2} = \begin{pmatrix} 300 & 1300 \end{pmatrix} \tag{3.2}$$

A data subset $\mathcal{X}_{1,2}^{tmp}$ is formed by excluding the features F_2 and F_4, and the instances I_1, I_5, I_9, I_{10}, and I_{11} (i.e. the instances with missing values in the targeted feature, F_2). This sub data set is shown in Equation (3.3). A complete data subset $\mathcal{X}_{1,2}$ is then obtained by removing instances that have missing values from $\mathcal{X}_{1,2}^{tmp}$, as shown in Equation (3.4).

$$\mathcal{X}_{1,2}^{tmp} = \begin{pmatrix} 310 & 1200 \\ 200 & 1700 \\ ? & 1100 \\ 300 & 1400 \\ 200 & 1200 \\ 290 & 1400 \\ 280 & 1500 \end{pmatrix} \tag{3.3}$$

$$\mathcal{X}_{1,2} = \begin{pmatrix} 310 & 1200 \\ 200 & 1700 \\ 300 & 1400 \\ 200 & 1200 \\ 290 & 1400 \\ 280 & 1500 \end{pmatrix} \tag{3.4}$$

3.2.2.4 WKNN settings

Rather than using all instances in $\mathcal{X}_{i,j}$ to build the imputation models, the WKNN method is employed to acquire $\mathcal{X}_{i,j}^K$, which contains the K nearest instances to $R_{i,j}$ from $\mathcal{X}_{i,j}$. The corresponding distances $\mathcal{D}_{i,j}^K$ and distance-based weights $\mathcal{W}_{i,j}^K$ are also calculated.

The value of K is determined by Equation (3.5). This equation restricts the lower bound of K as the number of available features $|J_{i,j}|$ to avoid having fewer instances than the used features. The upper bound of K is chosen empirically as one-fourth the number of the retrieved instances $\lfloor |I_{i,j}|/4 \rfloor$. However, if these constraints can not be satisfied, i.e. in case of a small complete subset, all instances obtained are used $|I_{i,j}|$. $\mathcal{D}_{i,j}^K$ is calculated according to Equation (3.6), and $\mathcal{W}_{i,j}^K$ is computed using Equation (3.7). The distance used to measure the similarity in KNN is the normalised Euclidean (Equation (3.1)).

$$K = min(max(|J_{i,j}|, \lfloor |I_{i,j}|/4 \rfloor), |I_{i,j}|) \tag{3.5}$$

$$\mathcal{D}_{i,j}^K[k] = distance(\mathcal{X}_{i,j}^K[k], V_{i,j}), k = 1, ...K. \tag{3.6}$$

$$\mathcal{W}_{i,j}^K[k] = 1 - \frac{\mathcal{D}_{i,j}^K[k]}{D_{i,j}}, k = 1, ...K, \tag{3.7}$$

where $D_{i,j} = \max\limits_{k=1,...K} \mathcal{D}_{i,j}^K[k]$.

The application of WKNN to get the K instances nearest to $R_{1,2}$ in Example 1 results in $\mathcal{X}_{1,2}^K$ (Equation (3.8)) and their weights are $\mathcal{W}_{1,2}^K$=[0.25 1]. The number of the valid features $|J_{1,2}|$ is 2 and the number of available complete instances $|I_{1,2}|$ is 6, hence $K = 2$ as $K = min(max(2, \lfloor 6/4 \rfloor), 6)$.

$$\mathcal{X}_{1,2}^K = \begin{pmatrix} 310 & 1200 \\ 300 & 1400 \end{pmatrix} \tag{3.8}$$

3.2.2.5 GP modeling

The subset $\mathcal{X}_{i,j}^{K}$ is then used to build N GP regression functions $\{G_g\}_{g=1}^{N}$ with the j^{th} feature as the target variable $T_{i,j}$. The weighted relative squared error (Equation (3.9)) is used as a fitness function to measure the quality of the individuals in the process of building GP regression models. The value $\mathcal{W}_{i,j}^{K}[k]$ is considered when evaluating the fitness function as a distance penalty for the k^{th} instance. The closer the instance to the targeted incomplete one, the larger contribution in the fitness function.

$$WRSE_{i,j}^{K} = \frac{\sum_{k=1}^{K} \mathcal{W}_{i,j}^{K}[k](Y_{i,j}[k] - T_{i,j}[k])^2}{\sum_{k=1}^{K}(T_{i,j}[k] - \bar{T}_{i,j})^2} \tag{3.9}$$

where K is the number of instances, $Y_{i,j}$ is a vector of the predicted values, $T_{i,j}$ is the corresponding vector of the desired values, and $\bar{T}_{i,j}$ is the average of the desired values $T_{i,j}[k]$, $k = 1, 2, 3..., K$.

For Example 1, GP is used to build N regression models G_g^{tmp}, $g = 1, ..., N$ where $\mathcal{X}_{1,2}^{K}$ is the input and the 2^{nd} feature of $\mathcal{X}$ is the target variable, i.e. $G_g^{tmp}(\mathcal{X}_{1,2}^{K}) \approx T_{1,2}$, where $T_{1,2}$ is shown in Equation (3.3).

$$T_{1,2} = \begin{pmatrix} 10 \\ 60 \end{pmatrix} \tag{3.10}$$

Finally, the GP model $G_{\hat{g}}^{tmp}$ with the best fitness value is selected as the imputation model $G_{1,2}$ for the reference $R_{1,2}$ and it is used to estimate the missing value $\mathcal{X}[1, 2]$ as follows:

$$\mathcal{X}^{C}[1, 2] = G_{1,2}(R_{1,2}) \tag{3.11}$$

The instances that have the same missingness pattern as $P_{1,2} = 1010$ (i.e. I_{10} and I_{11}) are compared to the reference instance $R_{1,2}$ considering the associated distance $D_{1,2}$. Since the instance I_{11} is similar to I_1, the imputation model $G_{1,2}$ is used to impute the missing value $\mathcal{X}[11, 2]$. For the instance I_{10}, it is not close enough to use these models. Therefore, new models $P_{10,2}$, $R_{10,2}$, and $G_{10,2}$ are constructed for the missing value $\mathcal{X}[10,2]$ by using the same process that is used to build the models for $\mathcal{X}[1,2]$.

3.2.3 The WKNN-GP Test Process

To impute test instances, the constructed models during the training process are used as in Algorithm 3. The inputs of the test process are the learned imputation models and an incomplete test instance. This process outputs an imputed complete instance. For each missing value, if there is a similar imputation pattern, the imputation models associated with this pattern are considered for imputing this missing value. Otherwise, the missing value is replaced by the mean value if the feature is numerical or the mode value if the feature is categorical.

Algorithm 3: WKNN-GP Imputation: The Test Algorithm.

```
Input  : Test instance $V_{test}$, $R$: the instance-based reference set, $G$: the
         GP-based imputation models, $P$: the missingness pattern set, and $S$
         contains the means and modes of the training features.
Output : Complete instance $V^C_{test}$
1  Set $V^C_{test} \leftarrow V_{test}$ ;
2  foreach missing value $V^C_{test}[j]$ do
3      Extract the non-missing values to form a complete instance $V_{test,j}$ ;
4      Obtain the corresponding missingness pattern $P_{test,j}$;
5      if there is a training pattern $P_{\hat{i},j} \in P$ which is similar to $P_{test,j}$ then
6          From the training reference instances that have the same missingness
             pattern $P_{\hat{i},j}$, get $R_{\hat{i},j}$ as the nearest one to $V_{test,j}$ , i.e. get $R_{\hat{i},j} \in R$ s.t.
             $distance(R_{\hat{i},j}, V_{test,j}) \leq distance(R_{i,j}, V_{test,j}), \forall i \neq \hat{i}$;
7          Set $V^C_{test}[j] \leftarrow G_{\hat{i},j}(V_{test,j})$; where $G_{\hat{i},j}$ is the corresponding GP imputer;
8      else
9          Use $S$ to estimate $V^C_{test}[j]$ by the mean if the $j^{th}$ feature is numerical or
             the mode if it is categorical.
10     end
11 end
```

The process of imputing the values is done iteratively for each feature using other features as predictive inputs. For example, if there are n features, the first one with missing values is firstly imputed based on the complete ones. The remaining features are then imputed in order by involving both the complete and the already imputed features.

Considering Example 1, let V_{test} be an incomplete test instance shown in Equation (3.12). The WKNN-GP imputation test process starts with the first missing value $V_{test}[2]$. To impute this value, all imputation models whose missingness patterns match $P_{test,2} = 1010$ are considered. Among the reference instances of this pattern, the nearest one to the test instance is $R_{1,2}$, hence, the GP imputation model $G_{1,2}$ is applied to impute this missing value, i.e. $V^C_{test}[2] = G_{1,2}(V_{test,2})$, where $V_{test,2} = [310\ 1400]$. A similar process is performed to impute the last missing value but with the missingness pattern $P_{test,4} = 1110$ as the previous missing value is already imputed.

$$V_{test} = \begin{pmatrix} 310 & ? & 1400 & ? \end{pmatrix} \tag{3.12}$$

3.3 Design of Experiments

In this section, the settings used for the experimental evaluations are presented. It includes the benchmark data sets, the baseline methods, the parameters of the methods, and the performance evaluation metrics.

3.3.1 Data Sets

For empirical evaluation, two types of data sets are considered: real-world data sets with synthetic incompleteness and real-world data sets with actual incompleteness. These data sets are split into $70 : 30$ training-test data sets following the widely used evaluation approach [48, 93, 240].

3.3.1.1 Real-world Data Sets with Synthetic Incompleteness

The first collection of data sets is shown in Table 3.4 and more details can be found in the data repositories UCI [62] and OpenML [241]. To introduce synthetic incompleteness to the complete data sets, MAR missingness mechanism is used to impose missing values with 10%, 30%, and 50% missingness ratios. This ends up with 90 training/test incomplete data sets for each complete data set (30 synthetic incomplete pairs for each of the three missingness ratios). The MAR mechanism is used as it is more suitable to evaluate the imputation performance [270]. This is because imputation estimates the missing values based on existing data, which suits the MAR mechanism that the missingness is related to existing data. It is implemented based on the simsem R package [184], which generates missingness in a feature depending on randomly selected set of covariates (other features) using a threshold method.

Table 3.4: The complete data sets used for **synthetic** missingness.

Data set	#Features	#Instances	Repository
Yacht	6	308	UCI
Housing	13	506	UCI
Forest	12	517	UCI
ENB2012	7	768	UCI
Concrete	8	1030	UCI
Weather izmir (Wizmir)	9	1461	OpenML
Debutanizer (PHP)	7	2394	OpenML
Kin8nm	8	8191	OpenML

3.3.1.2 Real-world Data Sets with Real Incompleteness

For the evaluation on real-world incompleteness, data sets that originally have missing values are used. Four real-world data sets having different ratios of missing values are used to examine the different methods. The statistics of these data sets are shown in Table 3.5 and more details can be found in the UCI machine learning repository [62].

Table 3.5: Statistics of the **real** incomplete data sets.

Data set	#Features	#Instances	#Incomplete Instances		Repository
			Number	Ratio%	
Auto-mpg	7	398	6	1.58	UCI
SkillCraft1	19	3395	57	1.68	UCI
Imports-85	15	205	54	26.34	UCI
Fishcatch	7	158	87	55.06	OpenML
Kdd_coil_2 (KDD)	11	316	34	10.75	OpenML
PBC	18	418	142	33.97	OpenML

3.3.2 Benchmark Methods

To investigate the imputation performance of the proposed method, it is compared with a set of popular imputation methods including: K-nearest Neighbour (KNN), Linear Regression (LR), Decision Tree (DT), Multi-layer Perceptron (MLP), Epsilon-Support Vector Regression (SVR), Bayesian Ridge Regression (BR), Random Forests (RF), and GP Imputation (GPI). These methods represent algorithms from the five tribes of machine-learning, namely: connectionism; evolutionism; Bayesianism; analogism; and, symbolism [71]. Except for GPI, the benchmark imputation methods utilise the corresponding regressors in the sklearn imputer using default parameters [30]. For GPI, the regression model in the GPlearn package [213] is employed in the sklearn imputer.

The symbolic regression process is carried out using the GP approach. The GP framework provided by distributed evolutionary algorithms in python (DEAP) [83] is used for the implementation of GP-based methods. The GP settings including terminals, functions, parameter values are shown in Table 3.6, where the number of generations represents the termination criteria.

Table 3.6: The used values for GP parameters

Parameter	**Value**
Generations	100
Population size	512
Crossover probability	0.8
Mutation probability	0.2
Elitism	Top 10
Selection Method	Tournament Selection
Tournament size	7
Minimum depth	2
Maximum depth	8
Initialisation	Ramped half-and-half
Function set	+, -, *, protected / (%)
Terminal set	features and constants $\in U[-1, 1]$

3.3.3 Performance Measures

To evaluate the imputation methods, there are three different measures. The imputation error, the symbolic regression error, and the computational time [222].

The imputation error is a measure that evaluates the accuracy of the prediction of the missing values. This measure requires the presence of the original values as ground truth to be compared with the predicted ones. The imputation performance evaluation method is shown in Figure 3.2.

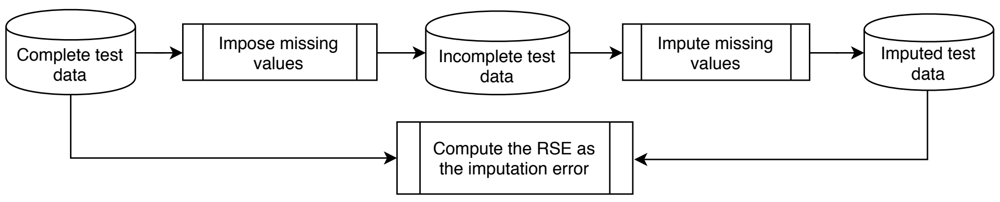


Figure 3.2: The imputation performance evaluation.

The original test data are compared with the imputed test data after applying the imputation model. The differences between the predicted values and the ground truth values are computed by the Relative square error (RSE) shown in Equation (3.13).

$$RSE = \frac{\sum_{i=1}^{n}(y_i - t_i)^2}{\sum_{i=1}^{n}(t_i - \bar{t})^2} \tag{3.13}$$

where n is the number of data values, y_i is the i^{th} predicted value, t_i is the i^{th} desired value (ground truth), and $\bar{t}$ is the average of the desired values t_i, $i = 1, 2, 3..., n$.

The second evaluation measure (symbolic regression error) is also based on testing incomplete data. This scenario simulates the real-life situation when missing values can occur also in the test data when predicting future instances. The adopted evaluation is shown in Figure 3.3. Symbolic regression error is also measured using RSE. Note that for the imputation error, the desired values in RSE are the values of input features, while these values are target variable values for the symbolic regression error.

It evaluates the symbolic regression error using the imputed complete test set after applying the imputation method. For each data set, the imputation methods are used to produce complete training and test sets. The training data are then fed into the symbolic regression process resulting in a symbolic regression model. This model is applied to the test data and the obtained regression error is used for comparisons.

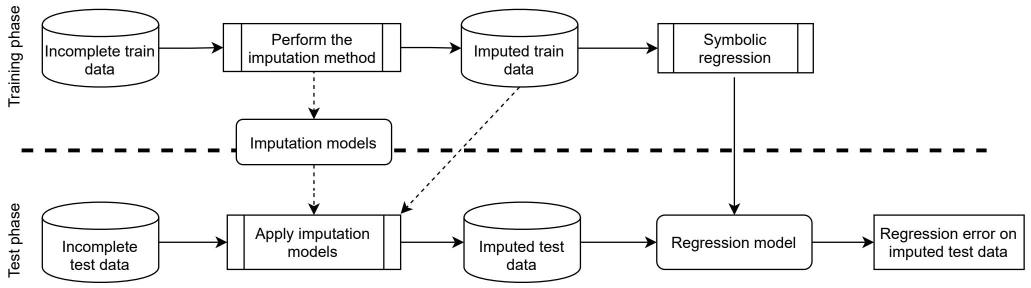


Figure 3.3: Symbolic regression performance evaluation.

For regression-based imputation methods (e.g. GP), the imputation models constructed during the training process are used when imputing the test data, whereas the imputed training data set itself is used in the data-driven methods (e.g. KNN).

3.4 Results and Discussions

This section shows the experimental results of the proposed imputation method, WKNN-GP. The results are analysed and compared with state-of-the-art imputation methods. The comparisons are carried out in terms of the imputation error, the symbolic regression performance, and the computational time. This section starts with the results of using the synthetic incomplete data sets showing the imputation errors, the symbolic regression errors, the significance test, and the imputation time. After that, the results for real-world incomplete data sets are presented.

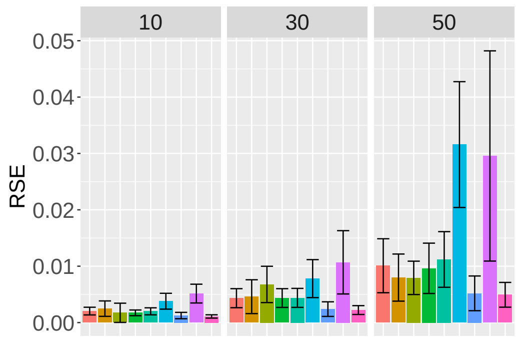
3.4.1 Imputation Error on Synthetic Incomplete Data

Following the imputation performance evaluation method shown in Figure 3.2, summary of the imputation errors is shown in Figure 3.4. The y-axis is the average of the imputation error measured by the RSE measure (Equation (3.13)) over the 30 synthetically generated incomplete data set for each missingness ratio.

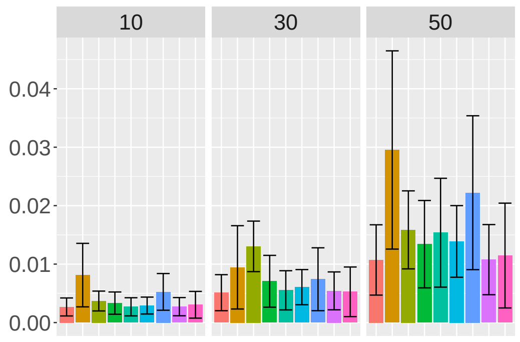
Figure 3.4 shows that the proposed GP-based method achieves better or at least similar results compared to the benchmark methods in most cases. WKNN-GP has the lowest imputation errors on the Housing, Yacht, ENB2012, and Kin8nm, and it is comparable to the best method on the other data sets. Even when it is not the top method, WKNN-GP still manages to achieve acceptable imputation results in comparison with the other methods. For example, while not the best on the Forest data set, WKNN-GP has a better imputation than three other methods.

Figure 3.4 shows the standard deviation for each imputation method shown as vertical lines. The longer the line is, the higher standard deviation is. The methods with higher standard deviations are less stable than those with lower standard deviations. As shown in these figures, the proposed method shows more stable results than other methods in most considered cases. This stability also differs based on the used data set. For example, despite the used imputation method, the Kin8nm data set shows stable imputation results whereas the Yacht data set shows unstable imputation results. This might be due to the nature of the data in each data sets. Some methods are less stable than others. For instance, the GPI method shows extremely more unstable imputation on the PHP and Concrete data sets than other methods. This could be due to the stochastic nature of the GPI method. This stochasticity is mitigated in WKNN-GP by integrating the WKNN method.

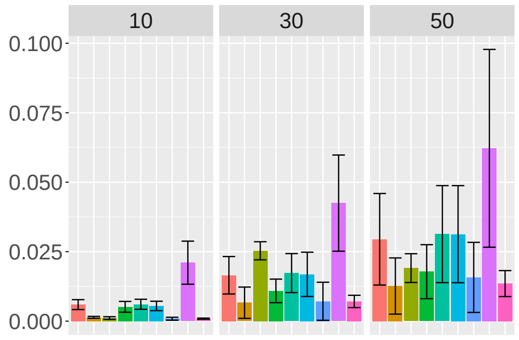
The WKNN-GP method advances the GPI and WKNN methods in almost all cases. This indicates that combining GPI and WKNN has a better imputation than using them separately. On data sets where GPI has a poor imputation performance (e.g. the Yacht data set), WKNN-GP has the best imputation performance. Similarly, WKNN-GP achieves the best imputation even on data sets when WKNN is not working well such as on the Kin8nm data set. This might be because WKNN-GP combines the advantages and mitigates the weakness of each method at the same time.



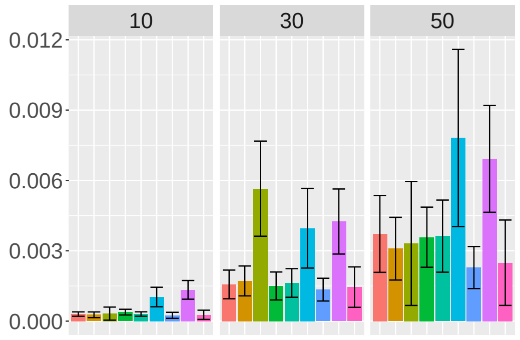
(a) Housing data set.



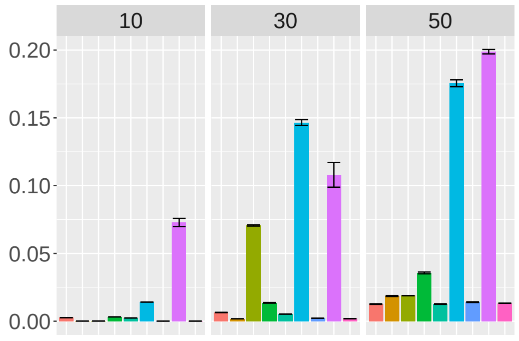
(b) Yacht data set.



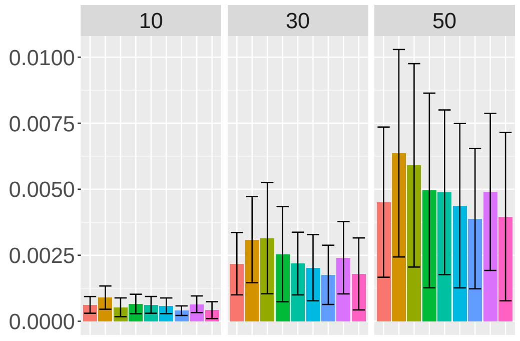
(c) Forest data set.



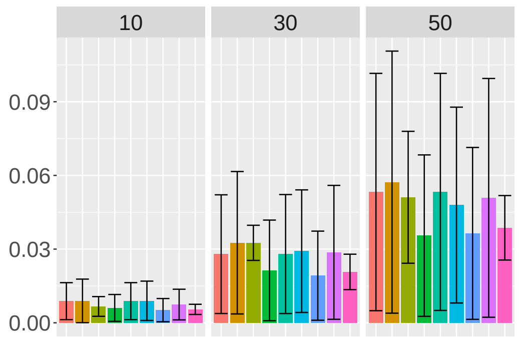
(d) Concrete data set.



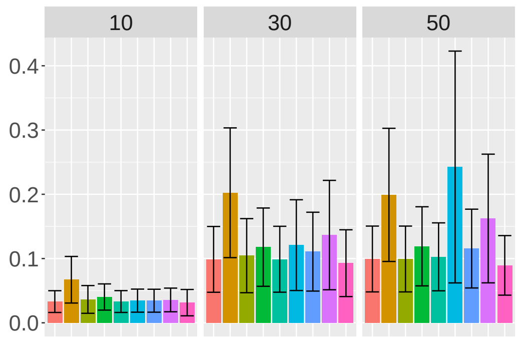
(e) ENB2012 data set.



(f) Wizmir data set.



(g) PHP data set.



(h) Kin8nm data set.

Method 🟥 BR 🟧 DT 🟨 GPI 🟩 WKNN 🟩 LR 🟦 MLP 🟦 RF 🟪 SVR 🟪 WKNN-GP

Figure 3.4: The imputation errors of different imputation methods on the synthetic incomplete data sets with different missingness ratios.

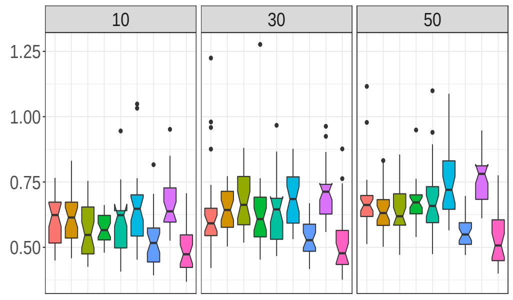
It can be noticed that the imputation error bars are monotonically increasing with respect to the missingness ratio. The errors increase with the increase of the missingness ratio. This pattern is common regardless of the imputation method or the used data set. This means that with higher missingness ratio, it is less likely to recover the missing values accurately. This is expected as there are less available data to build the imputation models. Moreover, the differences between the performance of the different methods is affected by the missingness ratio. With less missingness ratio, these differences are less likely to be high.

3.4.2 Symbolic Regression with Synthetic Incompleteness

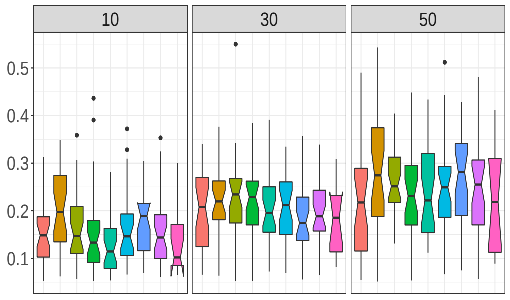
For each synthetic incomplete data set, 90 pairs of incomplete training and test data sets are imputed by each imputation method and the imputed complete pairs are then used for symbolic regression. For each pair, 30 independent symbolic regression experiments are conducted after using each imputation method. The symbolic regression test error results on the synthetic incomplete data sets are shown in Figure 3.5.

Methods with better imputation accuracy lead to better symbolic regression models. This can be noticed as the proposed method WKNN-GP has achieved the best symbolic regression on the Housing, Yacht, ENB2012, and Kin8nm data sets, where it achieved the best imputation results. Similarly, the method RF leads to the best results on the Concrete, Wizmir, and PHP data sets in terms of both the symbolic regression and the imputation performance. However, even on these data sets, WKNN-GP has competitive results.

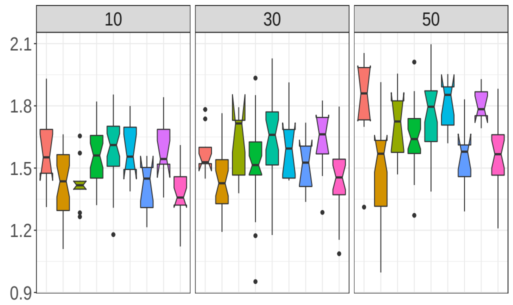
The detailed comparisons on each data set considering the missingness ratios show that more significant differences are observed when having higher missingness ratios. This is because the higher missingness ratio, the more likely to get different predictions for the missing values which in turn produces different symbolic regression results.



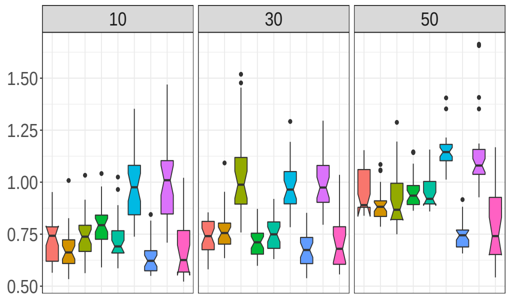
(a) Housing data set.



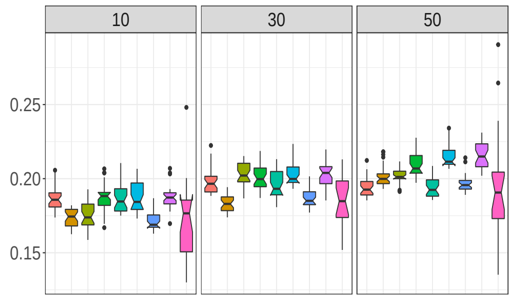
(b) Yacht data set.



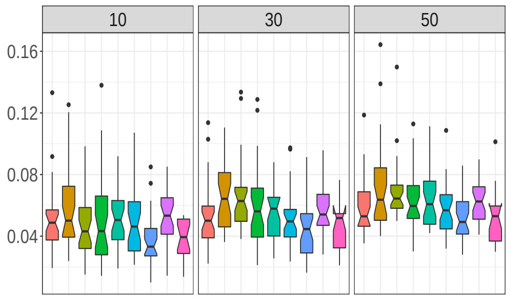
(c) Forest data set.



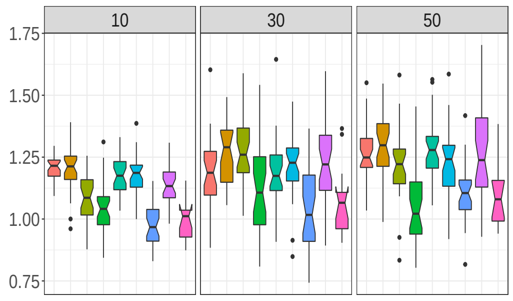
(d) Concrete data set.



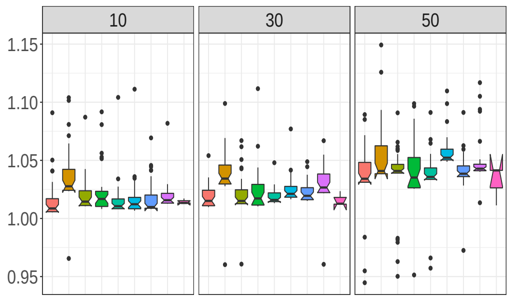
(e) ENB2012 data set.



(f) Wizmir data set.



(g) PHP data set.



(h) Kin8nm data set.

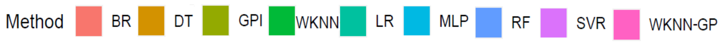


Figure 3.5: The symbolic regression results for using different imputation methods on the synthetic incomplete data sets with different missingness ratios.

3.4.3 Significance Test Results of Symbolic Regression with Synthetic Incompleteness

To examine the significance of the profitability of the proposed method with respect to the symbolic regression performance, the obtained results for each incomplete data set are used by a significance test to compare the benchmark methods with the proposed method. The significance test results between the proposed method and the benchmark imputation methods are shown in Table 3.7. The comparisons are carried out using the Wilcoxon non-parametric statistical significance test with a significance level of 0.05 on the symbolic regression results from 30 independent runs.

For each data set, there are 30 incomplete versions for each of the three missingness ratios, which sums up to 90 comparisons. The symbol "+" means WKNN-GP is significantly better, the "-" symbol means WKNN-GP is significantly worse, and "=" indicates no significant difference. The "Total" column shows the total number of "+"/win ("-"/loss) comparisons for the proposed method on the corresponding data sets, while the "Total" row shows the totals against the benchmark imputation methods.

Table 3.7 shows that the proposed GP-based method outperforms most other methods in most cases. On four of the eight data sets, WKNN-GP has the best imputation performance, while it was outperformed by only one method on three data sets. The comparisons show that WKNN-GP achieves better symbolic regression performance compared with the other methods in most cases. Except for the RF method, WKNN-GP has more win comparisons against all other methods. On seven of the eight data sets, the number of having significantly better performance when using WKNN-GP is more than that of having significantly worse results compared to all other methods. The best benchmark method is RF. This is expected as it is an ensemble learning method. On the other hand, the methods MLP and SVM are associated with the worst results in most data sets.

Table 3.7: The significance test results of the imputation performance, where the numbers refer to the amount of win(+)/loss(-)/draw(=) comparisons of WKNN-GP against each method.

Method	LR			DT			WKNN			MLP			SVR			BR			RF			GPI			Total			Best Method
ST	+	-	=	+	-	=	+	-	=	+	-	=	+	-	=	+	-	=	+	-	=	+	-	=	+	-	=	
Housing	57	6	27	48	6	36	51	9	30	69	0	21	78	0	12	54	0	36	27	27	36	42	6	42	426	54	240	WKNN-GP
Yacht	48	15	27	81	0	9	48	6	36	45	6	39	30	12	48	30	12	48	75	0	15	60	0	30	417	51	252	WKNN-GP
Forest	84	0	6	21	27	42	48	12	30	84	0	6	84	0	6	84	0	6	27	33	30	66	0	24	498	72	150	DT
ENB2012	33	15	42	42	15	33	81	0	9	72	0	18	81	0	9	39	3	48	27	36	27	69	0	21	444	69	207	WKNN-GP
Concrete	45	12	33	39	9	42	45	12	33	72	0	18	75	0	15	48	6	36	21	57	12	60	0	30	405	96	219	RF
Wizmir	24	39	27	45	9	36	30	21	39	21	36	33	27	18	45	24	33	33	15	66	9	33	12	45	219	234	267	RF
PHP	63	0	27	75	0	15	18	24	48	60	0	30	60	0	30	63	0	27	9	33	48	60	0	30	408	57	255	RF
Kin8nm	45	6	39	84	0	6	51	6	33	69	3	18	69	3	18	36	21	33	33	21	36	30	24	36	417	84	219	WKNN-GP
Total	354	93	189	351	66	219	372	90	258	492	45	183	504	33	183	378	75	267	234	273	213	420	42	258	3234	717	1809	

Table 3.8: The computation time (in milliseconds E-03) for each imputation method on the test data sets.

Method	Ratio	Yacht	Housing	Forest	ENB2012	Concrete	Wizmir	PHP	Kin8nm
LR	10	0.07531	0.1164	0.1215	0.1432	0.1876	0.2041	0.2193	0.2853
	30	0.07743	0.1284	0.1361	0.1714	0.1954	0.234	0.2953	0.3953
	50	0.07893	0.1371	0.1478	0.1967	0.2125	0.2587	0.3193	0.5349
DT	10	0.07246	0.1137	0.1577	0.1714	0.2134	0.2596	0.3543	0.4534
	30	0.0954	0.1348	0.1736	0.1954	0.2135	0.339	0.4651	0.6121
	50	0.1058	0.1537	0.1848	0.2335	0.2644	0.4117	0.6224	0.9244
WKNN	10	9.3127	15.155	18.833	19.6393	25.1855	34.917	112.385	125.3242
	30	10.438	21.2536	21.6672	26.0451	38.9654	47.761	155.8892	215.9783
	50	21.379	36.4654	37.6553	41.6741	49.2722	61.135	167.9085	233.2535
MLP	10	0.1069	0.2744	0.2871	0.2965	0.2817	0.317	0.3408	0.4564
	30	0.1259	0.2975	0.2936	0.3127	0.3824	0.3943	0.5408	0.7425
	50	0.1449	0.3278	0.3468	0.3965	0.5135	0.6453	0.7815	1.0615
SVR	10	0.1132	0.1414	0.1664	0.1845	0.2178	0.357	0.4608	0.6582
	30	0.1528	0.1628	0.1972	0.2386	0.3067	0.4774	0.6015	0.8689
	50	0.1829	0.2138	0.2436	0.2965	0.3278	0.5745	0.9615	1.3255
BR	10	0.0548	0.1313	0.1476	0.1641	0.1713	0.2065	0.2122	0.3735
	30	0.0848	0.1829	0.1731	0.2034	0.2112	0.2715	0.3148	0.4952
	50	0.1052	0.2238	0.2165	0.2785	0.3154	0.3915	0.4243	0.6191
RF	10	1.6648	4.939	5.6838	6.9178	7.7239	8.7738	21.5583	26.9764
	30	3.6369	6.7049	8.4839	9.03151	10.1413	12.1944	27.6523	31.7724
	50	4.5561	9.2128	10.2434	12.6577	15.6439	17.2716	28.8338	35.1255
GPI	10	0.1562	0.2537	0.2679	0.2765	0.2965	0.3234	0.4213	0.4891
	30	0.1839	0.2747	0.2855	0.3454	0.3554	0.4113	0.5443	0.5891
	50	0.2142	0.2953	0.3213	0.4256	0.4847	0.5145	0.6177	0.7225
WKNN-GP	10	0.2769	0.3438	0.3476	0.3619	0.3172	0.3478	0.5018	1.0723
	30	0.4351	0.4437	0.4674	0.5247	0.5374	0.6292	0.8077	1.3325
	50	0.5358	0.6157	0.6175	0.5362	0.6071	0.7493	0.9269	1.7156

3.4.4 Imputation Time on Synthetic Incomplete Data

Table 3.8 shows the average computation time of applying the imputation methods to the test data sets.

GPI and WKNN-GP build imputation models in the training process and the built models are used directly to impute new incomplete instances. Thus, the imputation time of the unseen data is decreased dramatically. The BR method has the shortest time, whereas the slowest one is KNN as it needs using the training data set to impute the test data. WKNN-GP is more efficient for imputing new incomplete instances than the methods with competitive imputation accuracy, especially RF. On the other hand, WKNN-GP is slightly slower than GPI. This is a small cost for the much better symbolic regression accuracy and imputation performance.

As seen in Table 3.8, once GP-based models are learned, they can be applied efficiently for imputing values in new instances. This is because GP typically evolves expression trees that are very quick to evaluate compared with other methods (e.g., random Forest). However, constructing GP-based imputation models is a time consuming process.

3.4.5 Symbolic Regression with Real Incompleteness

To validate the applicability of the proposed method in real-world tasks, data sets with real missing values are considered in this set of experiments. The symbolic regression results for different methods on real incomplete data sets are shown in Table 3.9.

Except for the imputation error, the measures that are used to evaluate the performance on synthetic incomplete data earlier are presented here. For the evaluation on real incomplete data, it is not possible to use the imputation error measure as the actual (ground truth) values of the missing data are unknown. Therefore, the evaluation is based on the symbolic regression results and the imputation time.

Table 3.9 shows the mean and standard deviation of the test symbolic regression RSEs achieved by each method along with the significance test when compared with the proposed method. The significance test sign “ST” refers to significance test result of the comparison between the proposed method, WKNN-GP, and each corresponding benchmark method.

Table 3.9: The symbolic regression results for the different methods on **real incomplete data sets**.

Method		Auto-mpg	SkillCraft1	Imports-85	KDD	PBC	Fishcatch
LR	mean	0.2514	0.6567	0.3815	1.099	1.0312	0.3317
	stdv	0.0784	0.1447	0.0411	0.1264	0.0798	0.2984
	ST	=	=	+	=	+	+
DTR	mean	0.2452	0.6478	0.3786	1.1971	0.9836	0.1629
	stdv	0.066	0.1567	0.0714	0.2165	0.0187	0.0953
	ST	=	=	+	+	+	+
WKNN	mean	0.2451	0.6506	0.3784	1.1724	0.9786	0.1673
	stdv	0.0621	0.1077	0.0719	0.3645	0.0184	0.0172
	ST	=	=	+	+	+	+
MLPR	mean	0.2491	0.6523	0.4109	1.0996	1.0059	0.2031
	stdv	0.0635	0.1295	0.056	0.2966	0.0314	0.0344
	ST	=	=	+	+	+	+
SVR	mean	0.2437	0.6531	0.4149	1.2958	1.0458	0.1865
	stdv	0.0719	0.1256	0.0534	0.4697	0.0598	0.0374
	ST	=	=	+	+	+	+
BR	mean	0.2453	0.6516	0.3906	1.1837	0.98	0.2805
	stdv	0.0643	0.1378	0.0365	0.1615	0.0495	0.146
	ST	=	=	+	+	+	+
RF	mean	0.2401	0.6321	0.3445	1.0015	0.9383	0.1526
	stdv	0.0551	0.1286	0.0812	0.2683	0.0328	0.0341
	ST	=	=	+	-	-	=
GPI	mean	0.2419	0.6411	0.3956	1.2522	0.9945	0.1721
	stdv	0.0625	0.1362	0.0947	0.2191	0.1101	0.0919
	ST	=	=	+	+	+	+
WKNN-GP	mean	0.2441	0.6331	0.3272	1.0944	0.9706	0.1578
	stdv	0.0665	0.1285	0.0304	0.2315	0.0661	0.0882

The sign "+"/"-"/"=" means that the results of the WKNN-GP method are significantly better/worse/equal when compared with those of the benchmark methods.

Similar to the case of synthetic incompleteness, the proposed method achieves remarkable results when applied to data sets with real incompleteness. Except for the RF method, WKNN-GP significantly outperforms all the other methods on different data sets. WKNN-GP is only outperformed on two data sets when compared with RF, which is an ensemble method. For the Auto and Skill data sets, the ratio of the missing values is too low, which marginalises the difference between the different methods on these data sets.

3.4.6 Further analysis: Outlier Analysis

In this work, the validation of the developments is through objective evaluation, primarily through the RSE metric. The typical values of the RSE measure is in the range of $[0, 1]$. When RSE is 0 this means that the model is perfectly fitting the task, while 1 implies that the model provides an overall performance worse than simply taking the mean estimation as a prediction model. Therefore, as there are many RSE measures that are greater than one, there is a need to conduct some further analysis to figure out the cause of such predictions. For this purpose, we consider a symbolic regression model on the popular data set Concrete. Let's take the following produced symbolic regression model:

$y = add(div(X7, mul(0.251, add(div(X7, mul(X3, 0.175)), add(div(X7, mul(0.251, mul(X3, 0.175))), sub(div(X6, add(div(X7, mul(0.251, add(div(X7, div(X6, mul(0.251, X3))), sub(div(X6, mul(0.251, X3)), div(X7, add(div(X7, mul(0.251, mul(X3, 0.175))), sub(div(div(X4, -0.477), mul(0.251, X3)), div(X4, -0.477)))))))), mul(X3, 0.175))), mul(0.251, mul(X3, 0.175))))))), sub(div(X6, sub(mul(0.251, X3), div(X4, -0.477))), div(X4, -0.477)))$

The corresponding simplified expression is shown as:

$$y = 2.096x_4 + \frac{x_6}{0.251x_3 + 2.096x_4} + \frac{x_7}{-0.011x_3 + \frac{0.251x_6}{0.175x_3 + \frac{x_7}{\frac{0.063x_3x_7}{x_6} - \frac{0.251x_7}{2.096x_4 - \frac{8.352x_4}{x_3} + \frac{22.766x_7}{x_3}} + \frac{1.0x_6}{x_3}}} + \frac{7.149x_7}{x_3}} \tag{3.14}$$

Table 3.10: Samples of the predictions on the Concrete data set.

ix	y_test	prds	$(y\_test - mean)^2$	$(y\_test - prds)^2$
17	27.04	20.33410362	80.12005	44.9690462
18	36.15	43.41008185	0.02528724	52.7087884
19	35.76	36.16100676	0.05335194	0.16080642
20	30.08	-115.797227	34.9396892	21280.1653
21	14.14	14.04340336	477.465344	0.00933091
22	13.29	39.57825622	515.334511	691.072415
23	37.68	39.47232546	2.85278724	3.21243056
87	45.37	45.26693228	87.9660088	0.01062296
88	52.42	36.14111061	269.912685	265.00224
89	26.26	134.2052671	94.6919794	11652.1807
90	28.3	32.33296068	59.1511794	16.2647718
91	46.8	50.26754314	116.834905	12.0238554

This symbolic regression model has a training RSE of 0.685, however, the test RSE is 1.646. This means that this model is worse than the trivial mean prediction model by more than 60%. To analyse this situation, we take a deeper look at the predictions at the instance level. We have noticed that the symbolic regression model produces extreme outliers. For example, we show samples of the obtained predictions as shown in Table 3.10. The instances highlighted with a blue colour show extremely off predictions of the symbolic regression model.

Note that the shown instances are samples of the instances where the instance "ix" (first column) refers to the instance order in the test data set. The y_test value refers to the desired true value while the $prds$ refer to the prediction values obtained using the symbolic regression model. The $(y\_test - mean)^2$ is the residual error of the true values from their mean value and $(y\_test - prds)^2$ is the residual error of the true values from the corresponding symbolic regression predictions. Although the mean value of the predictions (35.805) is close to the mean value of the true values (35.991), the sum of the residuals from the mean value ($\sum(y\_test - mean)^2 \approx 32074$) is significantly less than the sum of the residuals from the predictions ($\sum(y\_test - prds)^2 \approx 52795$). Since RSE is computed as $RSE = \sum(y\_test - prds)^2 / \sum(y\_test - mean)^2$, it is greater than 1 (1.646), which indicates a performance worse than the mean indicator. To show the impact of the outliers we remove them one by one then show the impact of their removal on the test error. Interestingly, when removing the first outlier (the 20^{th} instance), the RSE error went down to 0.984. This represents around 40% decrease in the overall error. Furthermore, when removing the 89^{th} instance the regression error is further decreased to about 0.622.

Now, to analyse the cause of getting these outlier predictions, we take a look at the input feature values of these instances and how they were predicted using the symbolic regression model. For the 20^{th} instance, both features x_6 and x_7 were incomplete and the missing values are imputed before performing regression. The errors in imputing these two features propagated the prediction outputs of symbolic regression, which leads to an extreme outlier in the predictions. On the other hand, for the 89^{th} instance, only the feature x_3 was incomplete. But it also leads to an outlier output in the regression prediction. Although many other instances have missing values the accuracy of recovering these missing values was considerably better than that of the two shown outlier instances. This means that the poor predictions of the missing values might have a big impact on the overall regression outputs.

The oultier issue represents a limitation of the proposed method that can be addressed in future work in different ways. One of these ways is that the range of the prediction outputs of GP can be bounded by the range of the training data. This could be done by employing interval arithmetic GP. The impact of the outlier issue is also related to the evaluation metric used to measure the regression accuracy. Similar to mean square error

(MSE) and root mean square error (RMSE), RSE is based on squaring the error residuals of the predictions. This makes RSE sensitive to outliers. This issue is less significant when measuring the accuracy using metrics that are less sensitive to outliers such as mean absolute error (MAE).

3.5 Chapter Summary

In this chapter, a new imputation method to improve symbolic regression performance on incomplete data is proposed. This method is called Weighted KNN-GP (WKNN-GP) which integrates two imputation methods: weighted K-nearest neighbour (WKNN) and genetic programming imputation (GPI). Such an integration is to utilise both the instance-based proximity of WKNN and the feature-based predictability of GP. The evaluation is conducted on real-world data sets considering the imputation accuracy, the symbolic regression performance, and the imputation time. The experimental results show that WKNN-GP outperforms both the GPI method and the WKNN method. Moreover, it is significantly better than some state-of-the-art imputation methods. WKNN-GP is more efficient and effective for imputing new incomplete instances than almost all other methods.

This chapter shows how WKNN can be utilised to work with GP for regression-based imputation. WKNN is powerful in exploiting the similarity between instances that have common missingness patterns. On the other hand, GP is powerful in building models for incomplete features based on other features. Combining these abilities was the main motivation of this chapter. It shows that such a hybridisation brings several benefits to symbolic regression on incomplete data. This includes efficient data imputation and effective symbolic regression when dealing on incomplete data. However, it performs the two processes, data imputation and symbolic regression, separately. The imputation method should be applied first to produce complete data then the symbolic regression can be performed on the imputed data. This means that the imputation process does not exploit the impact on the regression performance to be improved. The next chapter will develop methods that can impute data during the symbolic regression learning process, i.e. performing imputation and symbolic regression at the same time.

Chapter 4

GP-based Instance Selection for Symbolic Regression on Incomplete Data

4.1 Introduction

Instance selection finds a set of data instances that is useful for improving the learning performance. It is useful for getting rid of noisy and outlier instances and improving the effectiveness and the efficiency of the learning process [254]. However, very few existing instance selection methods are applicable to incomplete data. Instance selection methods usually use algorithms that work only on complete data. Moreover, to the best of our knowledge, before this work, no study has been done on instance selection for symbolic regression on incomplete data.

A common approach to handling the missingness situation is data imputation [90]. It works by estimating missing values based on existing data. However, not all existing data is useful. Therefore, which existing data should be used for imputing the missing values? The answer to this question is important when dealing with incomplete data. One popular imputation method is based on K-nearest neighbour (KNN). This method requires the use of the training data for imputing the unseen data. This requirement usually indicates high computation time when imputing each incomplete instance.

This chapter investigates integrating instance selection into data imputation for symbolic regression on incomplete data. To this aim, this chapter proposes a tree-vector mixed representation for GP to perform instance selection and symbolic regression on incomplete data. In this representation, each individual has two components: an expression tree and a bit vector. While the tree component constructs symbolic regression models, the vector component selects the instances for imputation by the weighted k-nearest neighbour (WKNN) imputation method introduced in Chapter 3. The complete imputed instances are then used to evaluate the GP-based symbolic regression model.

Chapter 3 proposes performing the imputation first then the symbolic regression is done on the imputed data. On the contrary, this chapter suggests performing data imputation and symbolic regression concurrently. This strategy is intended to increase the interaction between the models that impute the missing values and the symbolic regression modelling, which guides the imputation towards improving the symbolic regression performance. Consequently, the accuracy of both estimating the missing values and modelling symbolic regression will be increased simultaneously.

4.1.1 Chapter Goals

The main goal of this chapter is develop a GP-based instance selection method for symbolic regression on incomplete data. Such a method aims at improving both the effectiveness and efficiency of the data imputation and symbolic regression on incomplete data. Specific objectives of this chapter are:

- Proposing a hybrid representation that combines the tree representation and the vector representation to perform symbolic regression and instance selection concurrently.

- Proposing a new GP-based method that utilises the hybrid tree-vector representation to improve the WKNN imputation for symbolic regression on incomplete data.

- Investigating the effect of the proposed method for symbolic regression on data sets considering both synthetic and real missingness scenarios.

4.1.2 Chapter Organisation

This chapter is organised as follows. In Section 4.2, the proposed method is presented. The experiment design is presented in Section 4.3 and the corresponding results are described in Section 4.4. Finally, Section 4.5 presents the conclusions of this chapter.

4.2 The Proposed Method

This section presents the proposed method for instance selection and symbolic regression on incomplete data. It starts by introducing the overall system. After that, the learning process that selects the instances and produces symbolic regression models is presented. Finally, the testing process is given to use the selected components for symbolic regression prediction on new incomplete instances.

In this method, the main contribution is to integrate the vector representation with the GP-based tree representation for symbolic regression on incomplete data. The goal of this integration is to perform instance selection and symbolic regression simultaneously. The tree component is used to evolve the symbolic regression model and the vector component is responsible for selecting instances that could be used by WKNN for imputation to improve the performance of the constructed symbolic regression model.

4.2.1 The Overall Framework

The overall framework of this method is shown in Figure 4.1. Firstly, the data set is divided into a training set and a test set. The training data set is used by the proposed GP with hybrid tree-vector representation (GPTV) method to select a set of instances and construct symbolic regression models.

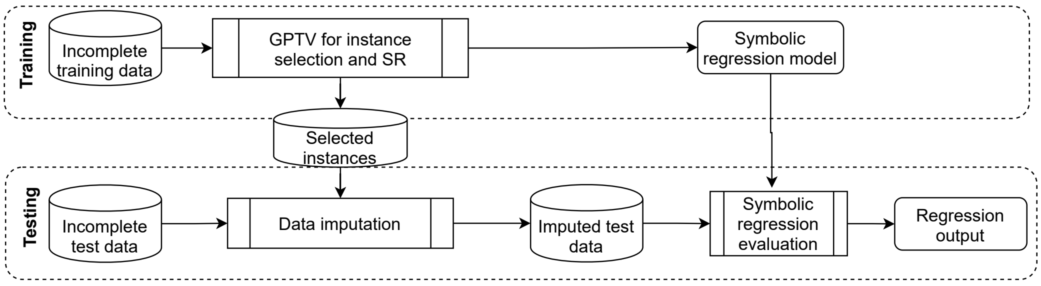


Figure 4.1: The overall framework of GPTV instance selection and symbolic regression on incomplete data.

During the evolutionary process of GPTV, symbolic regression is modelled by the expression tree and the instances are selected by the bit vector. The selected instances are used for imputing the missing values and the imputed/complete data are used to evaluate the evolved symbolic regression models. This means that the symbolic regression tree and the instance selection vector are learnt concurrently and they impact each other with the utilisation of WKNN-based imputation. In the test stage, the selected instances are used by the WKNN imputation method to impute the incomplete test data. After that, the learned symbolic regression models are applied to predict the imputed test data.

4.2.2 GP with Tree-vector (GPTV) Representation

The GPTV individual is represented using two components. The first one is the tree-based representation for GP that constructs symbolic regression models and the second one is the bit vector representation that is responsible for instance selection. Figure 4.2 shows an example of this representation.

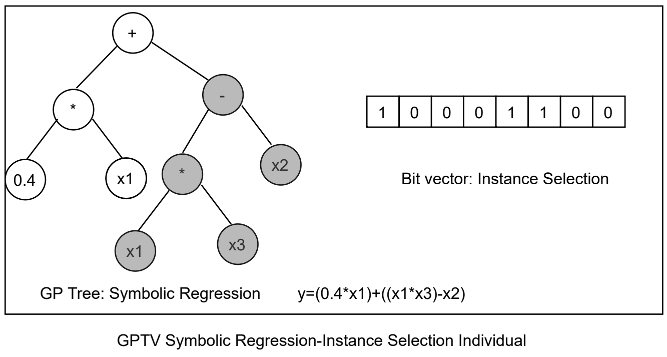

+
*
-
0.4
x1
*
x2
x1
x3
1 0 0 0 1 1 0 0
Bit vector: Instance Selection
GP Tree: Symbolic Regression
y=(0.4*x1)+((x1*x3)-x2)
GPTV Symbolic Regression-Instance Selection Individual

Figure 4.2: A GPTV individual for instance selection and symbolic regression.

The tree component of the GPTV individual, standard GP tree representation is used to evolve symbolic regression models. For the instance selection part of the GPTV representation, the vector is formed as a bit vector, whose length equals the number of available training instances. Each bit corresponds to one training instance such that when the bit has a value of "1", the corresponding instance is selected, otherwise, this instance is omitted.

Based on this representation, each individual delivers both a mathematical expression representing a symbolic regression model and a set of selected instances. The symbolic regression model and the selected instances are evolved and evaluated together as a single solution. Therefore, this representation can consider the impact of the selected instances on the modelling of the symbolic regression instantly.

4.2.3 GPTV Fitness Function

Each GPTV individual has two objectives that should be achieved simultaneously, building a good symbolic regression model and selecting a useful set of instances. Therefore, the fitness function should reflect these two objectives. The fitness function of GPTV is designed as in Equation (4.1). It has two parts to minimise both the symbolic regression error and the ratio of selected instances.

$$fitness = SR_{err} + SR_{err} * \alpha * I_{ratio} \tag{4.1}$$

where the SR_{err} is the symbolic regression error computed using the RSE defined in Chapter 3, which is shown in Equation (4.2).

$$RSE = \frac{\sum_{i=1}^{n}(y_i - t_i)^2}{\sum_{i=1}^{n}(t_i - \bar{t})^2} \tag{4.2}$$

where n is the number of instances, y_i is the i^{th} predicted value, t_i is the i^{th} desired value, and $\bar{t}$ is the average of the desired values t_i, $i = 1, 2, 3..., n$.

I_{ratio} is the instance reduction ratio introduced to reduce the number of selected instances by GPTV and it is computed as in Equation (4.3).

$$I_{ratio} = \frac{|\mathcal{I}_s|}{|\mathcal{I}|} \tag{4.3}$$

where $|\mathcal{I}_s|$ is number of selected instances and $|\mathcal{I}|$ is the number of all the available instances.

The parameter α is used to control the relative impact amount of the instance reduction on the regression error. As shown in Equation (4.1), the selection pressure is combined with the regression error to encourage the fitness function to minimise the number of selected instances in addition to minimising the regression error. The parameter $\alpha \in (0, 1)$ is used to control the balance between the instance selection pressure and the prediction error, which is set empirically to 0.2 to give more importance to the regression performance. This means that, for individuals that have very similar regression errors, the ones with fewer instances are favoured. Different values of α are tried during the preliminary experiments. The higher the value of α, the higher reduction in the number of the instances, however, this comes at the expense of the regression performance.

To prevent the selection pressure from dominating the fitness function, especially when the regression error is too small, the selection pressure is multiplied by the regression error. This strategy is also useful when the regression error is big as it is, theoretically, not bounded, while the selection pressure is limited between 0 and 1. However, this multiplication operation will preserve the selection pressure to be a ratio of the regression error regardless of how big/small it is.

4.2.4 GPTV Operators

As each GPTV individual consists of a tree and a bit vector, genetic operators that can be applied to such individuals should be designed. The traditional selection operator in GP can be used in GPTV. In this work, the tournament selection approach is adopted to select the fittest individual among a number of randomly picked individuals. This process is performed to select the individuals that are used as parents for crossover and mutation.

For the GPTV crossover operator, the two selected parents are used to produce two new offspring. This operation is done by performing both GP-like and GA-like crossover operations between the two individuals at the same time. An example of performing crossover between two GPTV individuals is shown in Figure 4.3. For trees, the single crossover point is used, while the two-point crossover is used for vectors. The crossover operator results in two new offspring, each contains a new bit vector and a new tree. As shown in the figure, the genetic contents of the grey coloured individual and the white coloured one are exchanged to form new mixed offspring.

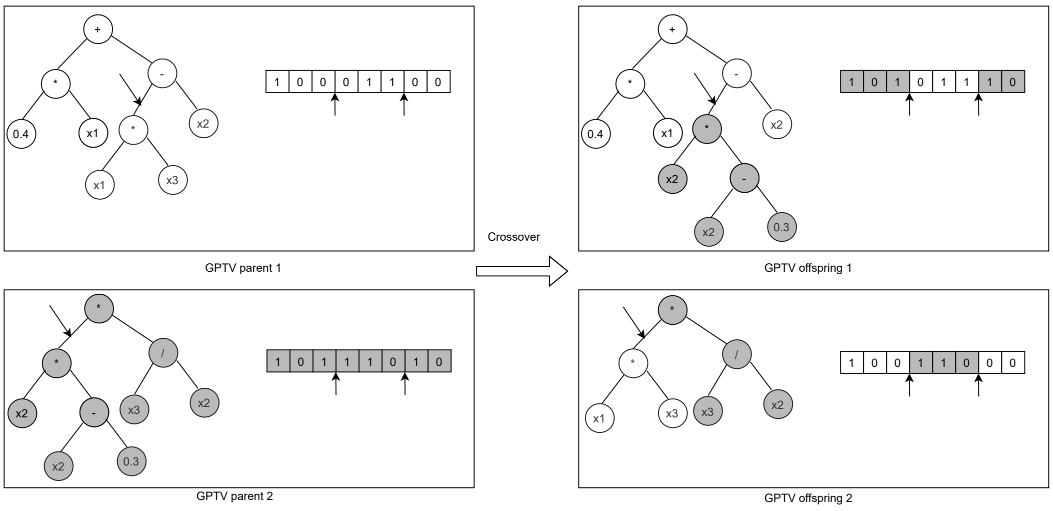


Figure 4.3: The crossover of GPTV individuals, where the arrows point to the crossover points.

Similarly, the GPTV mutation operator also performs both GP-like mutation and GA-like mutation at the same time. For a selected GPTV parent, the tree component is mutated by replacing a random subtree with a newly generated one. For the vector, the mutation is done by flipping bits, where the probability of each bit to be flipped equals the mutation rate. An example of GPTV mutation is given in Figure 4.4.

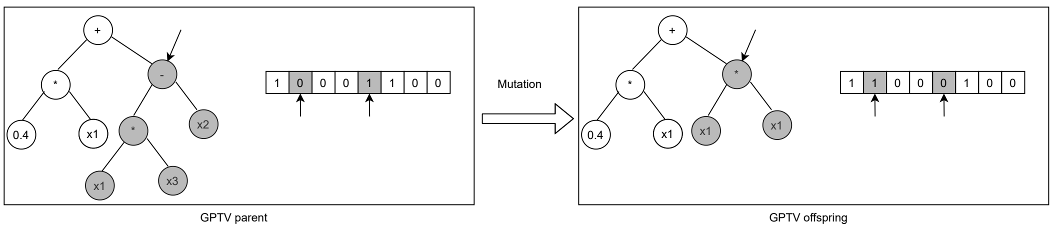


Figure 4.4: The mutation of GPTV individuals, where the arrows point to the mutation points.

4.2.5 GPTV Training

In the training stage, the training data set containing incomplete data is used to select a set of instances and build a symbolic regression model. A diagram of the data flow of the training process of GPTV is shown in Figure. 4.5.

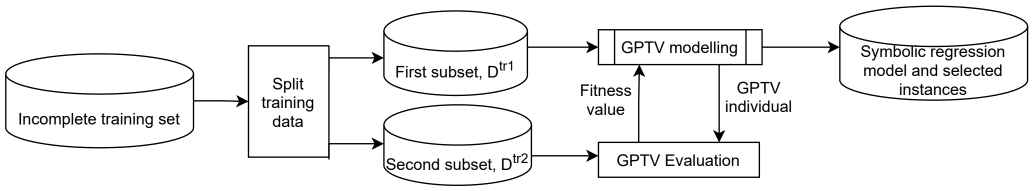


Figure 4.5: Data flow GPTV training.

This figure shows how the training data is used for learning GPTV models. The training data set is divided into two subsets for evaluation purposes. While the first subset is considered for determining the selected instances, the second data set is used to evaluate the models on different instances.

The pseudo-code of the training process is shown in Algorithm 4. First, the training data set is split into two sets; training subset 1 (referred to as D^{tr1}) and training subset 2 (D^{tr2}). The second training subset serves as a validation set to prevent over-fitting during instance selection. The validation set is important in instance selection methods [120], which are easy to over-fit the training data without holding out a validation set. This is because the evolutionary process is prone to models that fit a set of selected instances getting high training performance, while this set might not be representative of the whole data set.

Algorithm 4: GPTV instance selection and symbolic regression on incomplete data.

```
   Input  : Training data set with missing values, $D$.
   Output: A set of selected instances, $\mathcal{I}_s$
           A symbolic regression model, $SR$.
 1 Divide the training data set into two training subsets $D^{tr1}$ and $D^{tr2}$;
 2 Initialise a population of GPTV individuals;
 3 $gen = 0$;
 4 Use WKNN to impute $D^{tr1}$ and get an imputed complete data set
     $\hat{D}^{tr1}$;
 5 while $gen < genMax$ do
 6     foreach individual $ind$ in the current population do
 7         From $\hat{D}^{tr1}$, select a set of instances, $\mathcal{I}_s$;
 8         Using the selected instances, WKNN imputes the missing
             values in $D^{tr2}$ getting a complete data $\hat{D}^{tr2}$;
 9         Combine $\mathcal{I}_s$ and $\hat{D}^{tr2}$ getting a set of complete data $\hat{D}^{fit}$;
10         Compile the GP tree getting a symbolic regression model
             $SR$;
11         Evaluate $SR$ on the data $\hat{D}^{fit}$ getting a regression error
             $SR_{err}$;
12         Compute the instance reduction ratio $I_{ratio}$;
13         Calculate the fitness of the individual $ind$ using Equation
             (4.1);
14     end
15     Produce the next generation using GPTV genetic operators
         (Subsection 4.2.4);
16     $gen = gen + 1$;
17 end
18 Obtain the best individual in the last generation and use its tree
     component to get $SR$ and its vector component to get $\mathcal{I}_s$;
19 Return the selected instances, $\mathcal{I}_s$, and the symbolic regressor, $SR$.;
```

After obtaining the two training subsets, the GPTV evolutionary process is performed. It starts with an initialised population then a generational enhancement is done to obtain the required solution. The WKNN method is first used to impute the instances in the first training subset. For each GPTV individual, the vector component is used to select a set of instances from the complete first training subset. These selected instances are used to impute the missing values in the second subset. The two data sets are then combined to get a complete data set. After that, the GP tree of the GPTV individual is that represents a symbolic regression model is evaluated on the complete data. Finally, from the best individual of the last generation, the algorithm returns the symbolic regression model and the set of selected instances. For instance selection, the instances that are encoded in the vector of the best GPTV individual are selected. For the symbolic regression, the mathematical expression represented by the tree of the best GPTV individual is obtained.

4.2.6 WKNN Imputation

As seen above, the evaluation of GPTV individuals requires applying the constructed symbolic regression models to complete data. For this purpose, the widely used imputation method weighted k-nearest neighbour (WKNN) is used.

WKNN works by finding k nearest instances among the selected instances for each incomplete instance to be imputed. These instances are the nearest distance-based neighbours and they are called imputation donors. Each instance has assigned a weight based on the distance between the incomplete instances and the complete instances. WKNN is chosen as it is a good commonly used imputer. It is an instance-based imputer, where instance selection is crucial. The weighted mean of the values of these neighbours is then used to replace the missing values in the same features.

While the tree component constructs symbolic regression models, the vector component selects the instances based on their contribution to impute missing values by the WKNN imputation method. The instance selection idea is to select instances that better impute the missing values in other instances. This can be considered as the selection of imputation donors.

4.2.7 GPTV Testing

Algorithm 5: The GPVT testing process.

```
Input  : A test instance $I$
         $SR$: a learned symbolic regression model
         Set selected instances, $\mathcal{I}_s$.
Output: A prediction output for the instance $I$.
1 if $I$ is incomplete then
2     Use the selected instances in $\mathcal{I}_s$ to impute $I$ by the WKNN
        imputation method;
3     Apply the symbolic regression $SR$ to predict the regression
        output of $\hat{I}$, i.e. $p = SR(\hat{I})$;
4 end
5 else
6     Apply the symbolic regression $SR$ to predict the regression
        output of $I$, i.e. $p = SR(I)$;
7 end
8 Return the symbolic regression output, $p$;
```

In the test stage, the outcomes of the training process are applied to new test instances as shown in Algorithm 5. The selected instances are used to impute the missing values then the symbolic regression model is used to predict the regression output of the complete imputed instance. For an incomplete test instance, I, the WKNN imputation method is used to impute the missing values based on the set of the selected instances,

$\mathcal{I}_s$. This operation results in a complete instance, $\hat{I}$. After that, the learned symbolic regression model SR is applied on $\hat{I}$ to predict its regression output.

4.3 Design of Experiments

In this section, the settings used for the experimental evaluations are presented. For empirical evaluation, the data sets used in Chapter 3 are considered. The method proposed in this chapter, GP with tree-vector representation (GPTV), is compared with methods presented in Chapter 3. However, to avoid repeating the results presented previously, only the following methods are considered as benchmarks.

- **WKNN**: In this method, the weighted KNN imputation method is applied to estimate the missing values then symbolic regression is performed on the imputed complete data. The WKNN method considers all the available training instances. This method is considered to show the improvement gained by integrating the instance selection in the proposed method over the main WKNN imputation.

- **RF**: In this method, the random forest regressor is employed for imputation then symbolic regression is performed on the imputed complete data. This method is considered because it was the best benchmark method in Chapter 3. Therefore, the comparisons with this method are intended to measure the performance of the proposed method against a powerful existing method.

- **GPI**: In this method, GP is used for data imputation then symbolic regression is performed on the imputed complete data. The GPI method is considered compare with the case of using GP for imputation before symbolic regression.

- **WKNN-GP**: This method is presented in Chapter 3, where weighted KNN is combined with GP to estimate the missing values. WKNN is used to retrieve a set of instances that are close to the incomplete instances to be imputed. After that, GP is used to estimate the missing values in this incomplete instance. This method is considered to show that combining instance selection and GP can improve the direct imputation using WKNN and GP.

The benchmark methods are compared based on their associated symbolic regression performance. For example, when showing the results for WKNN, these are symbolic regression results after using WKNN imputation to estimate the missing values in the underlying data. As the evolutionary process is stochastic, 30 independent runs are performed on each data set. The difference between the results of the benchmark methods and the proposed method is measured using the Wilcoxon non-parametric statistical significance test with a significance level of 0.05.

The setting of the benchmark methods are the same as in Chapter 3. The proposed method is implemented using the DEAP package [83] with settings shown in Table 4.1. For the vector part of GPTV, a classic GA-based implementation via DEAP is followed. The settings of both tree and vector parts are set empirically after several pilot experiments. The evolutionary process is terminated when reaching the generation number.

4.4 Results and Discussions

This section presents the experimental results along with discussion and analysis. It starts with the results of symbolic regression on data sets with synthetic missing values. After that, the results for symbolic regression on data sets with real-world incompleteness are presented. Then the instance selection results and the associated imputation times are given.

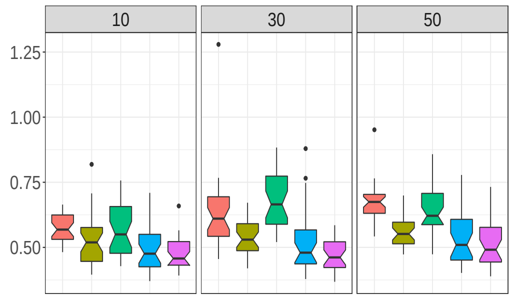
Table 4.1: The GPTV settings.

Parameter	Value
Generations	100
Population size	512
Crossover rate	0.8
Mutation rate	0.2
Elitism	Top 10 individuals
Selection method	tournament (7)
Vector representation	binary
Vector crossover type	two-point
Tree minimum depth	2
Tree maximum depth	8
Tree initialisation	Ramped-half and half
Tree function set	+, -, *, protected %
Tree terminal set	features and constants $\in U[-1, 1]$

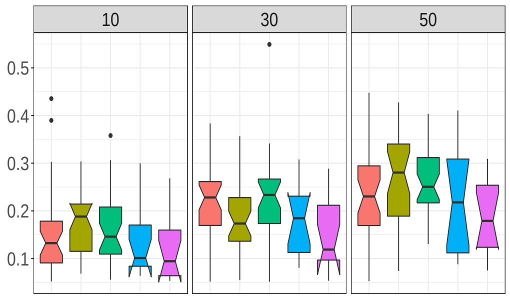
4.4.1 Symbolic Regression with Synthetic Incompleteness

The symbolic regression results in terms of the RSEs on the test sets for the different methods on the synthetic incomplete data sets are shown in Figure 4.6. The symbolic regression performance is measured using RSE (as shown in Equation (4.2)).

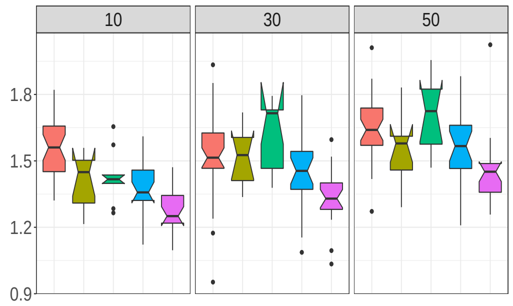
As can be seen from the shown results, the proposed method brings improvements over the compared methods. In particular, the GPTV method outperforms the other methods in almost all cases. This is because the GPTV method considers improving the symbolic regression performance when selecting the instances that are used for imputation. These instances serve as imputation donors and they are then used later to impute the test data. The soundness of this approach might be because the instances selected during the training process help improve handling the missing values in the test data as the missingness cause in the training data is similar to that in the tests data.



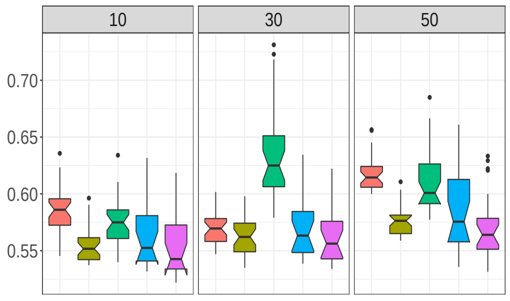
(a) Housing data set.



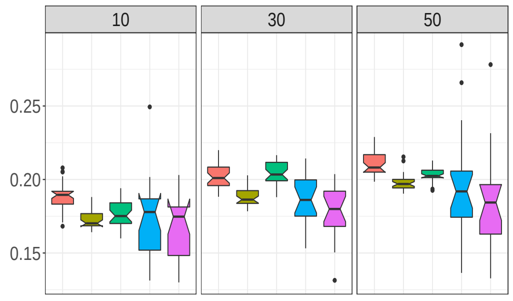
(b) Yacht data set.



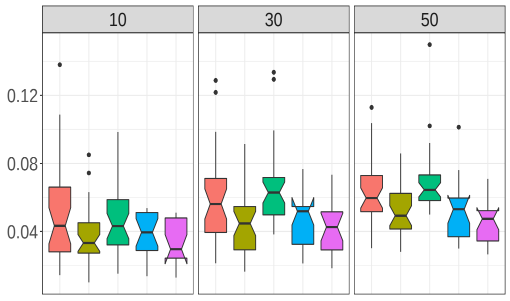
(c) Forest data set.



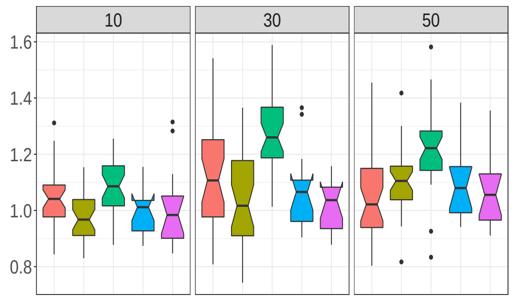
(d) Concrete data set.



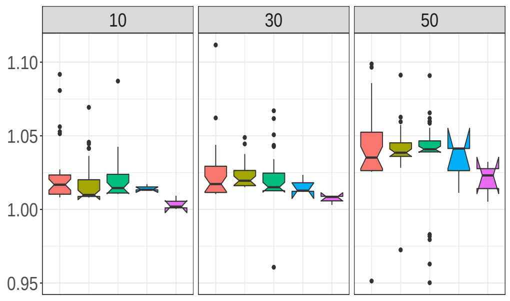
(e) ENB2012 data set.



(f) Wizmir data set.



(g) PHP data set.



(h) Kin8nm data set.

Method 🟥 KNN 🟨 RF 🟩 GPI 🟦 WKNNGP 🟪 GPTV

Figure 4.6: The symbolic regression results for the different methods on **synthetic incomplete data sets** with different missingness probabilities.

The combination of both of the instance selection and the symbolic regression modelling bring improvements over using underling methods WKNN and GP respectively. The results obtained by GPTV are better than those obtained by using GPI and WKNN individually. Moreover, GPTV shows better performance compared with WKNN-GP. This is because the instances used for imputation are selected in a way that improves the symbolic regression performance.

The results show that GPTV has a superior performance compared to the benchmark method in most cases. The main difference between the benchmark method and the GPTV method is that, the benchmark imputation methods perform missingness estimation in an unsupervised manner. That is, the imputation is carried out regardless of its impact on the symbolic regression performance. In contrast, GPTV performs imputation while constructing the symbolic regression models. The selected imputation instances are guided by the goodness of both of the produced symbolic regression model.

To show the significance of the obtained symbolic regression results, the Wilcoxon statistical test is performed between the GPTV method and each one of the other methods. These comparisons are conducted for the 30 synthetic incomplete data sets for each of the three missingness probabilities. So, there are 90 comparisons between GPTV and each method on each data set. The significant test results are shown in Table 4.2.

Although the proposed method shows good results in all considered data sets, it can be noticed that the significance of the improvement differs depending on the used data set. This can be induced from the total number of wins/losses on each data set. The GPTV method has less improvement when applied to data sets that have a relatively small number of instances e.g. the Yacht data set as there is no enough data to learn robust models. The results suggest that the GPTV method is most suitable for data sets with medium size. This is shown in the high number of wins on the Housing, Forest, and ENB2012 data sets.

Table 4.2: The significance test results of the symbolic regression performance, where the numbers refer to the amount of win(+)/loss(-)/draw(=) comparisons of GPTV against each method.

Method	WKNN			RF			GPI			WKNN-GP			Total		
Data	+	-	=	+	-	=	+	-	=	+	-	=	+	-	=
Housing	74	4	12	44	23	23	78	1	11	45	16	29	241	44	75
Yacht	58	7	25	32	0	58	77	2	11	32	21	37	199	30	131
Forest	81	0	9	57	12	21	81	0	9	74	4	12	293	16	51
ENB2012	84	3	3	47	36	7	69	4	17	51	14	25	251	57	52
Concrete	69	8	13	36	44	10	70	0	20	71	5	14	246	57	57
Wizmir	50	4	36	35	44	11	74	4	12	68	8	14	227	60	73
PHP	21	8	61	28	36	26	71	0	19	66	3	21	186	47	127
Kin8nm	66	12	12	41	33	16	62	11	17	58	7	25	227	63	70
Total	503	46	171	320	228	172	582	22	116	407	78	152	1870	374	636

On the other hand, the improvement of the method is less significant on data sets that have a relatively large number of instances e.g. the Kin8nm data set. This may be due to the use of bit vector representation to evolve instance sets. This means that the individuals will contain longer vectors for data sets with a larger number of instances, which hinders the convergence of the evolutionary process [120]. On the other hand, the best results are achieved in medium-scale data sets.

4.4.2 Symbolic Regression with Real Incompleteness

The symbolic regression results for the examined methods on real incomplete data sets are shown in Table 4.3. This table shows the mean and standard deviation of test RSEs of the symbolic regression achieved by each method along with the significance test when compared with the proposed method.

The sign "+" is used to refer to that the proposed method, GPTV, is significantly better than the compared benchmark method. The sign "-"

means that the benchmark method has significantly better performance than GPTV. If there is no significant difference between the methods, the test sign is "=".

Table 4.3: The symbolic regression results for the different methods on **real incomplete data sets**.

Method		Auto-mpg	SkillCraft1	Imports-85	KDD	PBC	Fishcatch
WKNN	mean	0.2451	0.6506	0.3784	1.1724	0.9786	0.1673
	stdv	0.0621	0.1077	0.0719	0.3645	0.0184	0.0172
	ST	=	=	+	+	+	+
RF	mean	0.2401	0.6321	0.3445	1.0015	0.9383	0.1526
	stdv	0.0551	0.1286	0.0812	0.2683	0.0328	0.0341
	ST	=	=	+	=	-	=
GPI	mean	0.2419	0.6411	0.3956	1.2522	0.9945	0.1721
	stdv	0.0625	0.1362	0.0947	0.2191	0.1101	0.0919
		=	=	+	+	+	+
WKNN-GP	mean	0.2441	0.6331	0.3272	1.0944	0.9706	0.1578
	stdv	0.0665	0.1285	0.0304	0.2315	0.0661	0.0882
	ST	=	=	+	=	+	+
GPTV	mean	0.2431	0.6297	0.3175	1.0772	0.9561	0.153
	stdv	0.0843	0.0489	0.0537	0.3622	0.3013	0.1054

Similar to the case of synthetic incompleteness, GPTV achieves the best results compared with the other methods on most data sets with real incompleteness. GPTV significantly outperforms all the other methods on the Imports data set. On the other hand, GPTV manages to obtain equivalent, if not better, results on the Fishcatch and KDD data set. Moreover, except for the comparison with RF, GPTV was the best on the other data sets. RF achieved significantly better results on the PBC data set. This is expected as RF is an ensemble method. However, on the Auto and Skill data sets, the differences between methods are not significant because these data sets have a small amount of missing values.

4.4.3 Instance Selection and Computation Time

Another comparison that could be conducted between data-driven imputation methods is the data reduction ability of the methods. This reduction plays an important role in the efficiency of applying these methods in symbolic regression on incomplete data. Table 4.4 shows the number of instances ($\#Instances$) used in each method along with the corresponding symbolic regression computation time (in milliseconds per test instance).

Table 4.4: The **average computation time** for each method (in milliseconds per instance) along with the average of used instances.

Data set	WKNN		GPTV	
	#Instances	Time	#Instances	Time
Yacht	205	13.7099	113	6.9466
Housing	337	24.29123	158	13.4779
Forest	345	26.05183	207	21.8698
ENB2012	512	29.1195	266	27.7723
Concrete	687	37.8077	385	31.8323
Wizmir	974	47.93767	506	49.9945
PHP	1596	145.3942	878	75.1859
Kin8nm	5461	191.5189	2621	79.9697

As seen from Table 4.4, the instance selection method leads to a decrease in the number of selected instances, which in turn decreases the computation time of the WKNN imputation method. WKNN uses all training data to impute missing values in test instances. On the other hand, GPTV uses only a subset of the instances for imputation. This makes GPTV more efficient than WKNN.

4.5 Chapter Summary

The goal of this chapter was to propose a new instance selection method to improve the imputation for symbolic regression on incomplete data. To achieve this goal, we proposed a method, GPTV, which uses a tree-vector mixed representation to perform instance selection for imputation while constructing GP-based symbolic regression models.

To evaluate the proposed method, two sets of real-world data sets considering two incompleteness scenarios are used. The first scenario imposes synthetic missingness into complete data sets and the second scenario considers dealing with data sets that have actual missing values. The results show that the proposed method can improve the symbolic regression performance.

GPTV has the ability of selecting representative training instances and learn a symbolic regression model that can be used to predict incomplete test data. The experimental results show the superiority of the GPTV method in handling the incompleteness for symbolic regression. The combination of tree and vector representations not only improves symbolic regression effectiveness but also reduces the computation time when applying the learned models to new instances.

This chapter shows how instance selection can be utilised to improve symbolic regression on incomplete data. It presents a method that builds imputation models and symbolic regression models simultaneously. The proposed method in this chapter helps speed up and enhance the imputation process of the WKNN method. However, this method might not be effective in the case of high-dimensional data. Incomplete data sets with a large number of features need methods that can address the curse of dimensionality issue in addition to handling the missing values. This is the subject of the next chapter in which a method for symbolic regression on high-dimensional incomplete data will be presented.

Chapter 5

GP-based Feature Selection for Symbolic Regression on High-dimensional Incomplete Data

5.1 Introduction

High dimensionality is one of the serious data challenges in symbolic regression and it is more challenging if the data are incomplete. Although feature selection can successfully tackle the high dimensionality issue, unfortunately, most feature selection methods do not work in the presence of missing values. GP has the natural feature selection ability but it is not directly applicable to incomplete data. Instead, to deal with incompleteness, it is common to impute the missing values first then perform GP on the imputed complete data. However, in the case of having many irrelevant incomplete features, it would be better to avoid the unnecessary costly imputation of such features. This could be done if feature selection is performed before data imputation. This chapter shows how GP can be utilised for feature selection for symbolic regression on incomplete data.

Interval arithmetic is a technique that enables calculating arithmetic operations (e.g. addition) on interval-valued elements instead of single-valued numbers. With interval arithmetic, GP outputs intervals rather than point predictions in [205]. In [110], interval-valued GP (IGP) is used to make sure that the models produced by GP do not result in undefined values. In [65, 64], IGP is extended with interval-aware search operators to reduce the number of invalid solutions obtained during the search process. Recently, some attempts have been taken for utilising IGP in dealing with incomplete data.

In [226], GP-based multiple feature construction (GPMFC) is successfully extended by using IGP to directly construct features for classification on incomplete data. This method showed better performance than the simple imputation of using the mean value but not better than the case of using more advanced imputation method (e.g. multivariate imputation by chained equations, MICE). In [224], IGP is utilised to directly construct classifiers for incomplete data. IGP is improved in [232] by incorporating ensemble learning. However, we have tried to adapt this approach to perform symbolic regression directly on incomplete data but the results are disappointing. Although the predictions are not accurate, we noticed that the IGP evolved models are able to pick features that are predictive than others.

In this work, an IGP-based method is proposed to select features directly from incomplete data. This method is applied to high-dimensional symbolic regression on incomplete data. The main reason of using IGP is that replacing missing values with feature intervals can reflect the uncertainty associated with the missingness in these features. Therefore, IGP feature selection might provide a feature set that benefits the learning process more than the approach of traditional GP-based feature selection after imputation.

5.1.1 Chapter Goals

The goal of this paper is to develop a new GP-based method that can *directly* select features from incomplete data. To this end, an interval-valued GP-based approach is proposed for selecting features. The selected features with incomplete data are then imputed and fed into the symbolic regression learning process. This work has the following main objectives.

- Proposing a GP-based method for selecting features from incomplete data for symbolic regression. Instead of imputing the data then selecting the features, this method provides the ability to select the features **directly** from incomplete data. Although some attempts have been done on GP for dealing with incomplete data, none of them has investigated its ability to select features directly from incomplete data. To do so, a new interval-valued GP (IGP) approach is presented in our work, which not only selects features but also selects their imputation predictors.

- Proposing a fitness function for IGP, which includes a point-to-interval error metric and a feature selection pressure factor for IGP. For this purpose, the mid-point metric which can measure the distance between an interval and a real value is used to form an evaluation metric. In addition, the fitness function is regularised by a feature selection pressure to enforce reducing the number of selected features during the evolutionary process.

- Utilising the proposed approach for symbolic regression on high-dimensional incomplete data. Several experiments have been conducted to evaluate the applicability of the proposed method to symbolic regression in different missingness scenarios. The obtained results provide empirical evidence that having a GP-based feature selection method that can work directly on incomplete data benefit both the efficiency and the effectiveness of symbolic regression on incomplete data.

5.1.2 Chapter Organisation

This chapter is organised as follows. In Section 5.2, the proposed method is presented. Then, the experiment design is presented in Section 5.3. The experimental results are described in Section 5.4. Finally, Section 5.5 concludes this chapter.

5.2 The Proposed Method

5.2.1 The Overall System

This method selects useful features before imputation for symbolic regression on incomplete data as shown in Figure 5.1. The feature selection process is highlighted by a dashed frame.

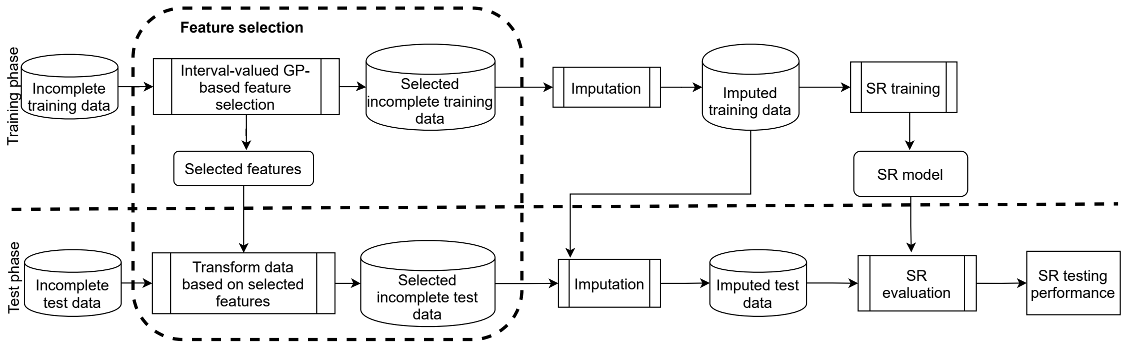


Figure 5.1: Utilising feature selection for imputation in symbolic regression on incomplete data.

This figure shows two phases: training and testing. The incomplete training data are used for feature selection directly without imputing the data. The output is a set of selected features, and both the training set and test set are transformed by keeping only the selected features and removing the other features. After that, an imputation method is applied to produce complete data sets. Popular imputation methods are performed on the transformed data. The imputed training data are then used in the test phase by the imputation method to estimate the missing values in test instances. The imputed training data are fed into standard GP learning process to construct symbolic regression models while the prediction performance of these models is examined on the imputed test data.

5.2.2 Interval-valued GP

Interval arithmetic refers to dealing with intervals that are defined as sets of real numbers as below [161].

$$[x^-, x^+] = \{x \in \mathbb{R} \,|\, x^- \leq x \leq x^+\}, \tag{5.1}$$

where $x^-(x^+)$ is the lower (upper) bound of the interval and it is unbounded if $x^- = -\infty$ or $x^+ = +\infty$, while the whole set of real numbers, $\mathbb{R}$, is obtained when both are infinite.

Similar to real-valued arithmetic, which deals with arithmetic operators on real numbers, interval arithmetic operators are defined on intervals. For practical applications, mostly, the following four basic operators are used [179]. Intervals can be combined with real numbers where the real number $x \in \mathbb{R}$ is represented by an interval as $[x, x]$ [56]. For practical applications, mostly, the following four basic operators are used and they can be used to derive complicated functions.

- Addition:
$$[x_1^-, x_1^+] + [x_2^-, x_2^+] = [x_1^- + x_2^-, x_1^+ + x_2^+]. \tag{5.2}$$

- Subtraction:
$$[x_1^-, x_1^+] - [x_2^-, x_2^+] = [x_1^- - x_2^+, x_1^+ - x_2^-]. \tag{5.3}$$

- Multiplication:
$$\begin{aligned}[x_1^-, x_1^+] * [x_2^-, x_2^+] =& [\min(x_1^- x_2^-, x_1^- x_2^+, x_1^+ x_2^-, x_1^+ x_2^+), \\ & \max(x_1^- x_2^-, x_1^- x_2^+, x_1^+ x_2^-, x_1^+ x_2^+)].\end{aligned} \tag{5.4}$$

- Division:

$$\frac{[x_1^-, x_1^+]}{[x_2^-, x_2^+]} = [x_1^-, x_1^+] \cdot \frac{1}{[x_2^-, x_2^+]}, \tag{5.5}$$

where

$$\begin{aligned} \frac{1}{[x_2^-, x_2^+]} &= \left[\frac{1}{x_2^+}, \frac{1}{x_2^-}\right], 0 \notin [x_2^-, x_2^+] \\ \frac{1}{[x_2^-, 0]} &= \left[-\infty, \frac{1}{x_2^-}\right] \\ \frac{1}{[0, x_2^+]} &= \left[\frac{1}{x_2^+}, \infty\right] \\ \frac{1}{[x_2^-, x_2^+]} &= \left[-\infty, \frac{1}{x_2^-}\right] \cup \left[\frac{1}{x_2^+}, \infty\right] = [-\infty, \infty], 0 \in (x_2^-, x_2^+) \end{aligned} \tag{5.6}$$

The use of interval arithmetic extends standard real-valued GP to interval-valued GP (IGP) [205]. In IGP, the terminal set elements are intervals. That is, instead of single-valued features, intervals of feature values are used. Similarly, the real-valued operators in the function set of standard GP are replaced with interval-valued arithmetic operators defined in Equations (5.2), (5.3), (5.4), and (5.5).

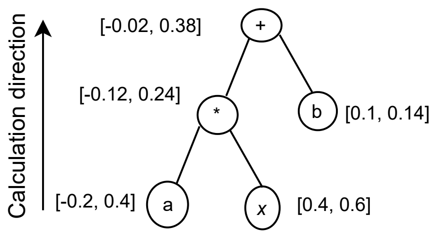


Figure 5.2: A tree-based representation for IGP, $g = a * x + b$, where $a = [-0.2, 0.4], b = [0.1, 0.14]$ and $x = [0.4, 0.6]$.

An IGP tree is shown in Figure 5.2. Given two constant terminals $a = [-0.2, 0.4]$ and $b = [0.1, 0.14]$ and a feature $x = [0.4, 0.6]$, the expression $g(a, b, x) = a * x + b$ is evaluated as $g(a, b, x) = ([-0.2, 0.4] * [0.4, 0.6]) + [0.1, 0.14] = [-0.2 * 0.6, 0.4 * 0.6] + [0.1, 0.14] = [-0.02, 0.38]$.

5.2.3 IGP for Feature Selection in Symbolic Regression on Incomplete Data

GP has the ability of selecting features while constructing the desired models. The features involved in a GP program represent a set of selected features. However, traditional GP can only work on complete data. To select features from incomplete data, a two-stage IGP based method is proposed in this work. The procedure of IGP for feature selection is shown in the following listing. The outputs are two lists of features selected in two stages. The first one is $IGPFS$, which is a set of features selected. The second list $IGPPS$ contains the selected imputation predictors for the incomplete features that are included in $IGPFS$.

The first stage starts by preparing the data set to be IGP-compatible, i.e. convert it from real-valued data set into an interval-valued data set by replacing each entry with an interval. Missing values in a feature are replaced with an interval of the feature estimated as its minimum and maximum existing values, while the existing values are also replaced with an interval whose lower and upper bounds equal to the existing value itself. For example, assuming we have 3 features, the interval of which are $[-1, 2]$, $[-2, 3]$, and $[-3, 2]$, respectively. So, the original single-valued instances $(?, 1.5, 1)$, $(?, ?, 1)$, and $(?, ?, ?)$ are converted to the interval-valued instances $([-1, 2], [1.5, 1.5], [1, 1])$, $([-1, 2], [-2, 3], [1, 1])$, and $([-1, 2]\,, [-2, 3], [-3, 2])$, respectively.

As the process of feature selection in the first stage considers the symbolic regression performance on the target variable, some features that are not useful for the target variable but can be useful for imputing incomplete features might not be selected. For this reason, the second stage of feature selection is presented to select a good set of predictors for the selected incomplete features. For each incomplete feature selected in stage 1, IGP selects imputation predictors. The selected features and predictors are used for imputing the data.

Listing 1. The main steps of the proposed approach

```
S1. Stage 1: selecting the features.
    S1.1. Form an interval-valued data set, $[D]$, in which each single value in $D$ is
          replaced with an interval.
    S1.2. Construct IGP model, $G$, to predict $T$ using the data set $[D]$.
    S1.3. Append all features in $G$ to $IGPFS$.
S2. Stage 2: selecting imputation predictors for the selected features.
    S2.1. $IGPPS = \phi$.
    S2.2. For each incomplete feature $f$ in $IGPFS$.
        S2.2.1. $IGPPS_f = [f]$.
        S2.2.2. Form an interval-valued data set in which the target variable is $f$,
                $[D]_f$, taking only the instances that does not have missing values
                in $f$.
        S2.2.3. Construct IGP model, $G_f$, to predict $f$ using the data set $[D]_f$.
        S2.2.4. Append all predictors appear in $G_f$ to the set of $f$ imputation pre-
                dictors $IGPFS_f$.
        S2.2.5. Append the list $IGPPS_f$ to $IGPPS$.
```

5.2.4 IGP Fitness Function

IGP individuals are applied to predict the outputs of interval-valued features and produce interval-valued predictions. Given a data set with n instances, each IGP individual produces n interval predictions $[y_i^-, y_i^+], i = 1, 2, 3..., n$. However, in order to evaluate these outputs against the desired target values, $t_i, i = 1, 2, 3..., n$, a metric that measures the distance between interval-valued outputs and single-valued targets is needed.

The overall prediction error of an IGP individual is computed using the mean point to interval error (MPIE) shown in Eq (5.7).

$$MPIE = \frac{\sum_{i=1}^{n} d(t_i, [y_i^-, y_i^+])}{n} \tag{5.7}$$

where n is the number of the instances, $[y_i^-, y_i^+]$ is the i^{th} predicted interval, t_i is the i^{th} desired value, and d is the distance metric between $[y_i^-, y_i^+]$ and t_i. To define d, different metric errors are used as follows.

Existing work that used IGP for incomplete data evaluates the quality of the evolving individuals by comparing mid-points of the produced intervals with the desired values [224, 232]. This metric is used for classification with incomplete data and it is defined as in Eq.(5.8).

$$d(t_i, [y_i^-, y_i^+]) = |t_i - \frac{y_i^- + y_i^+}{2}|. \tag{5.8}$$

Meanwhile, a selection pressure which emphasises on reducing the number of involved features in the evolved IGP individuals is considered in the fitness function of IGP. The selection pressure is based on the generation number and the number of selected features as in Equation (5.9).

$$selection\_pressure = \frac{g}{G} * \frac{\#F_s}{\#F} \tag{5.9}$$

The first component introduces the impact of generation, which is defined by the current generation number, g, divided by the total number of generations, G. The weight of feature reduction increases when g increases, which allows including more features in the early generations and the more advanced generations have larger contributions to feature reduction. The second component is defined by the number of features in the individual, $\#F_s$, divided by the number of all available features, $\#F$. If two individuals at a generation have the same error, a lower (better) fitness value is given to the one with fewer number of features.

As shown in Equation (5.10), the selection pressure is combined with the regression error MPIE to form the IGP fitness function to minimise both the number of features and the regression error. A parameter $\alpha \in$

$(0, 1)$ is used for controlling the balance between the feature selection pressure and the regression error, which is set empirically to be 0.3 to weigh the regression performance more. To prevent the selection pressure from dominating the fitness function, especially when the error is too small, the selection pressure is multiplied by this error. This strategy is also useful when the error MPIE is big as it is, theoretically, not bounded, while the selection pressure is limited between 0 and 1. However, this multiplication operation will preserve the selection impact to be a ratio of the whole error regardless of how big/small it is.

$$fitness = MPIE * (1 + \alpha * selection\_pressure) \tag{5.10}$$

5.3 Design of Experiments

To investigate the performance of the proposed approach, a set of experiments has been conducted. This section presents the design of these experiments including the used data sets, parameters, evaluation metrics, and benchmark methods.

5.3.1 Data Sets

To evaluate the proposed method, different types of data sets are used. This consideration allows providing more solid conclusions regarding the performance of the proposed approach under different circumstances.

5.3.1.1 Real-world Data Sets with Synthetic Incompleteness

Four high-dimensional real-world data sets are used. Each data set has more features than instances. Table 5.1 shows the information of these data sets and more details can be found in [241]. In this work, each data set is split randomly into training and test (sub)sets with the ratio of $70 : 30$. Similar to previous chapters, incomplete data sets are generated by imposing missingness ratios of 10%, 30%, and 50% to each complete data set.

Table 5.1: Statistics of the data sets.

Data	# Features	#Training Instances	#Test Instances
Selwood	54	21	10
Pah	113	54	26
Pdgfr	321	54	25
Mtp2	1143	183	91

5.3.1.2 Artificial Data Sets with Synthetic Incompleteness

Artificial data sets are also considered for comparing the algorithms. This allows systematic experiments, which enables more solid analysis. For this purpose, the well-known Friedman function [85] is used. This function includes both linear and non-linear relationships between the independent variables and the dependent variable. A normalised noise, ϵ, is also added to as shown in Equation (5.11).

$$y = 10 * sin(\pi * x_1 * x_2) + 20 * (x_3 - 0.5)^2 + 10 * x_4 + 5 * x_5 + \epsilon \tag{5.11}$$

Following previous research on regression presented in [10], in our work, artificial data sets based on Friedman function are generated. In addition to the 5 input features, additional independent randomly generated features are added into the data sets to measure ability of the feature selection methods to handle irrelevant features. Moreover, to measure the impact of colinearity, three colinearity degrees (0, 2, and 4) are used when generating data sets of 1000 instances. The colinearity degree refers to the number of features that are dependent on other features. These artificial data sets are named as $fri\_cC\_I\_F$, where C, I, and F are the colinearity degree, number of instances, and number of features, respectively. For example, the data set $fri\_c0\_1000\_10000$ consists of 1000 instances and 10000 features with zero colinearity. The data sets used in this chapter are formed by adding 9050 random features to the data sets of 1000 instances, and colinearity degrees 0, 2, and 4, that are used in [10] and can be found in [241]. This forms the data sets $fri\_c0\_1000\_10000$, $fri\_c2\_1000\_10000$, and $fri\_c4\_1000\_10000$. For shortcut, we will refer to these data sets as fri_c0, fri_c2, and fri_c4, respectively. After that, the strategy used previously on real-world data is adopted on the artificial data for splitting the data and imposing missingness.

5.3.1.3 Real-world Data Sets with Real Incompleteness

To validate the applicability of the proposed approach on realistic incompleteness, two real-world data sets that originally have missing values are used. Table 5.2 shows statistics of these data sets and more details can be found in the UCI machine learning repository [62].

Table 5.2: Statistics of the used incomplete data sets

Data set	#Features	#Instances	#Incomplete instances
CCN	128	1994	1676
CCUN	147	2215	1763

The CNN data set is related to the communities and crime data within the United States. The CCUN data set is similar to CCN but it has more variables. Another difference is that the CCUN data set is unnormalised. These data sets are used for high-dimensional symbolic regression in [49, 44] after deleting the incomplete entries. Note that, the string features are removed (1 from CNN and 2 from CCUN) and the instances that have missing target values are removed as well (97 from CCUN). Although the extra variables in CCUN are provided as potential goals for prediction, we are using them as predictive features. The considered target variable in CNN is "ViolentCrimesPerPop" and it is "nonViolPerPop" in CCUN.

5.3.2 Methods and Parameters

5.3.2.1 Evaluation Methodology

The feature selection approaches are evaluated based on their impact on three popular imputation methods for symbolic regression on incomplete data. The imputation methods include mean imputation (MeanIM), K-nearest neighbour imputation (KNNIM), and random forest imputation (RFIM) [118]. These methods represent different strategies. MeanIM is a uni-variate ad hoc imputation method. KNNIM is an instance-based proximity imputation method while RFIM is a feature-based regression imputation method. This consideration enables the validation of the applicability of the feature selection methods to various incompleteness situations. The imputation methods are based on the packages missingpy [1] and simputation [236], and the symbolic regression is carried out using the gplearn package [213] keeping default settings.

5.3.2.2 Parameter Settings

The proposed approach produces two feature sets from the two stages: IGPFS and IGPPS. These sets are compared with two other approaches. The first one is to use all the available features without any feature selection strategy and it is denoted as "All". This approach is used to evaluate the difference brought when feature selection is employed. Actually, it aims to examine the consideration of feature selection in the first place. The second method is based on decision tree (DT) that has the ability to select features from incomplete data. This one is denoted as DTFS and it is considered to compare the GP-based feature selection method with a non-GP feature selection method where both can work directly on incomplete data.

The IGP-based feature selection is implemented using strongly typed GP utilising the DEAP python package [83] with the setting shown in Table 5.3. The evolutionary process stops when reaching the maximum number of generations or getting a zero fitness function. Note that, as the division operation in IGP is not well defined on intervals spanning zero, the trees which violate the IGP assumptions are eliminated during the search process.

Table 5.3: IGP settings

Parameter	Value
Generations	50
Population size	512
Crossover rate	0.8
Mutation rate	0.2
Elitism	Top-10 individuals
Selection method	Tournament selection
Tournament size	7
Maximum depth	9
Initialization	Ramped-half and half
Function set	Interval functions (Equations (5.2), (5.3), (5.4), (5.5))
Terminal set	Interval-valued features

For DTFS, the R library rpart [265] which is an implementation of classification and regression trees (CART) is used, while the synthetic incomplete data sets are generated using the R package SIMSEM [184]. The metric used for computing the regression error on the test set is relative squared error (RSE) shown in Equation (5.12), which is the same as previous chapters.

$$RSE = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{5.12}$$

where n is the number of instances, y_i $(\hat{y}_i)$ is the target (predicted) value of the i^{th} instance, and $\bar{y}$ is the average of the target values.

The evaluation of the methods is based on the symbolic regression performance. Features are selected from incomplete data producing transformed data that are imputed to get complete data, which can be used in symbolic regression. The feature selection methods are All, DTFS, IGPFS, and IGPPS. When using IGPFS, it refers to the use of the features selected by IGP, however, IGPPS implies adding the imputation predictors of each incomplete feature in the IGPFS set. These predictors are also selected by IGP. The imputation methods are MeanIM, KNNIM, and RFIM. For the symbolic regression, it is based on standard GP, and its regression error is measured using RSE of the predictions for the test data. As GP is stochastic, the experiments are repeated 30 times independently, with a different random seed every time, for each method then the collective results are compared with corresponding ones of the other methods. The significance of the difference between the results is measured based on pair-wise Wilcoxon test with a significance level of 0.05.

5.4 Results and Discussions

5.4.1 Performance on Symbolic Regression

In this section, the experimental results are presented. It starts by showing the symbolic regression performance on different types of data sets. After that, the feature reduction and the computation time are analysed.

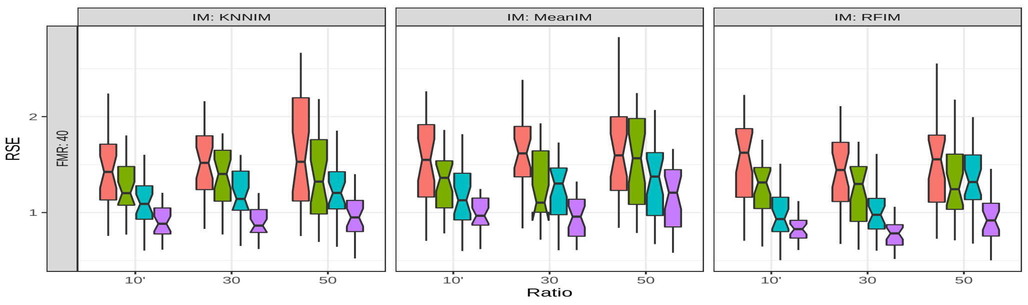
5.4.1.1 Symbolic Regression on Real-world Data Sets with Synthetic Incompleteness

The first set of experiments is conducted on complete real-world data sets after imposing different ratios of missing values. Figure 5.3 shows the symbolic regression results with synthetic incompleteness on real-world data sets. Each subfigure shows the results for one data set where the imputation methods are shown vertically and the feature-based missingness ratio is given horizontally. The y-axis represents the symbolic regression error measured using RSE on each test set, whereas the x-axis stands for the instance missingness ratio.

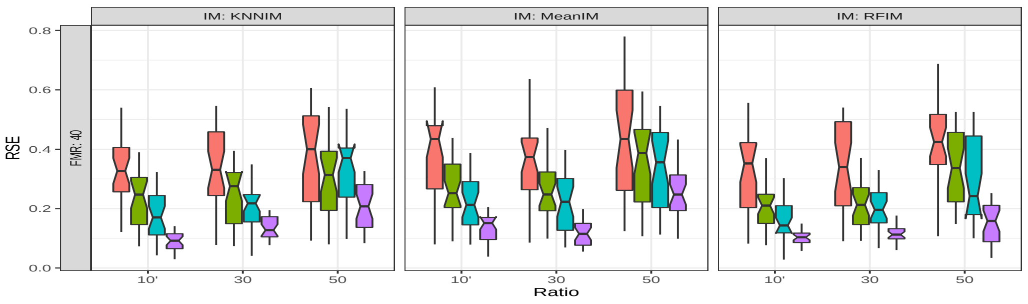
From the shown results, it is clear that IGPPS has the best symbolic regression results compared with the other methods when using different imputation methods on all the data sets. It tends to improve the results obtained by IGPFS, which produces, mostly, better results than DTFS. Such results are due to the high compatibility of GP-based feature selection methods with symbolic regression compared with non-GP feature selection. The results also show that feature selection techniques are better than the case of using all features in most cases.

For the impact of the missingness ratio, it can be noticed that the ratio of missingness has a considerable effect on symbolic regression performance. In fact, the higher the missingness ratio, the worse the symbolic regression performance. Moreover, models constructed using highly incomplete data are unstable. The variance with such incompleteness ratios indicates the stability of the corresponding models. For example, the case of 50% missingness ratio has the worst results on each data set. On the other hand, the 10% missingness ratio is accompanied with better symbolic regression results compared with higher ratios.

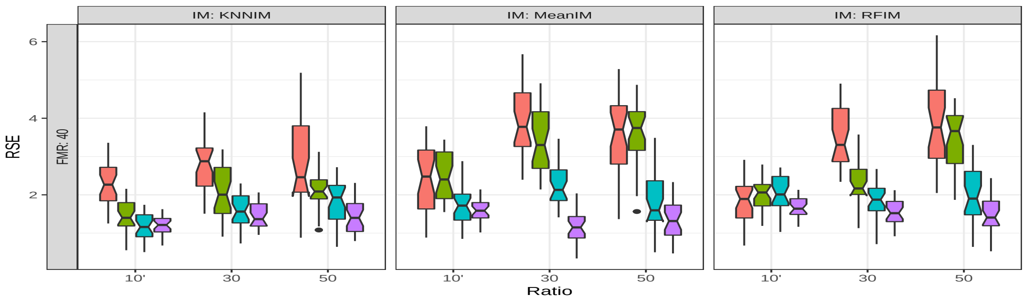
(a) Selwood



(b) Pah



(c) Pdgfr



(d) Mtp2

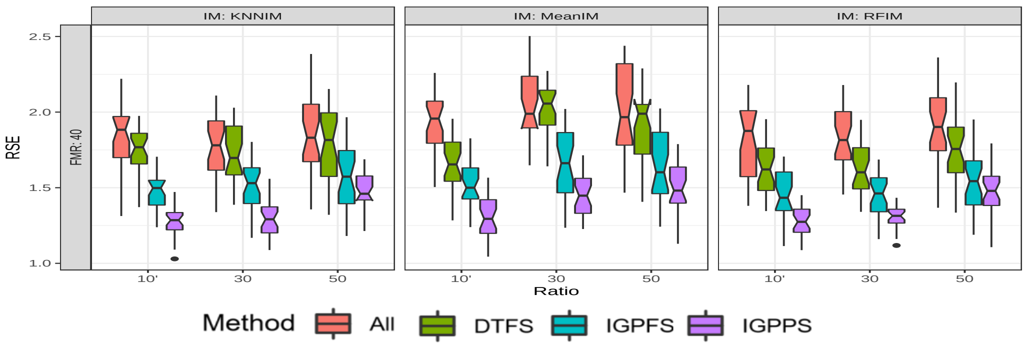


Figure 5.3: The test RSEs with different feature selection and imputation methods.

Table 5.4: The test RSEs on Artificial Data with Synthetic Incompleteness.

Method	Data	fri_c0			fri_c2			fri_c4		
	IM%	10	30	50	10	30	50	10	30	50
	All	1.7838	1.7412	1.8727	1.7804	1.7225	1.6269	1.8718	1.9997	1.9909
	DTFS	1.5144	1.5969	1.6053	1.4795	1.5242	1.6593	1.6224	1.8532	1.6384
MeanIM	IGPFS	1.2992	1.3611	1.4212	1.3198	1.2722	1.35109	1.28269	1.620304	1.41505
	IGPPS	1.1952	1.2387	1.3075	1.2538	1.1323	1.2835	1.1801	1.5231	1.2735
	ST	+,+,+	+,+,+	+,+,+	+,+,+	+,+,+	+,+,+	+,+,+	+,+,+	+,+,+
	All	1.7378	1.7405	1.8721	1.7262	1.6409	1.7345	1.6323	1.9269	2.0487
	DTFS	1.4517	1.5568	1.5058	1.4226	1.593	1.6242	1.698	1.568	1.7371
KNNIM	IGPFS	1.2412	1.1079	1.2896	1.2121	1.1893	1.2672	1.6258	1.3184	1.3602
	IGPPS	1.1047	1.0525	1.2251	1.1393	1.1179	1.1912	1.4632	1.2261	1.2922
	ST	+,+,+	+,+,+	+,+,+	+,+,+	+,+,+	+,+,+	+,+,+	+,+,+	+,+,+
	All	1.611	1.8077	1.7814	1.7154	1.7105	1.8117	1.3755	1.7772	1.8083
	DTFS	1.4944	1.5244	1.5447	1.5275	1.4237	1.4108	1.2007	1.5204	1.314
RFIM	IGPFS	1.1932	1.2374	1.2955	1.278	1.3024	1.3073	1.1046	1.1808	1.1701
	IGPPS	1.0739	1.1755	1.2307	1.2013	1.2117	1.1635	1.0051	1.1217	1.0998
	ST	+,+,+	+,+,+	+,+,+	+,+,+	+,+,+	+,+,+	+,+,+	+,+,+	+,+,+

5.4.1.2 Symbolic Regression on Artificial Data with Synthetic Incompleteness

Table 5.4 shows the symbolic regression results with synthetic incompleteness on artificial data. In this table, IM% refers to the instance-based missingness ratio, and column "ST" is for the significance test of the difference between the $IGPPS$ and the other methods. The symbol "+" ("-") means that IGPPS outperforms (is outperformed by) the compared method, whereas "=" means no significant difference. These symbols are in 3-tuple according to the comparisons with the other methods in the same order shown in the table.

From this table, similar conclusions to those obtained on the real-world data with synthetic completeness can be easily noticed. Namely, the use of feature selection leads to considerable improvements in the symbolic regression results regardless of the used imputation method. When it comes to comparing the feature selection methods, the GP-based approach achieves better results than the DT-based one. Specifically, the IGPPS method is the best in all considered cases. These results can be used to conclude that the proposed IGP-based feature selection approach selects features that are more beneficial for symbolic regression. Eventually, IGP-based feature selection also performs symbolic regression but with interval-valued space instead of the single-valued space used in GP-based symbolic regression. Therefore, the usability of the features selected by IGP-based feature selection seems intrinsically justifiable.

For the cases of non-significance difference between feature selection methods, interestingly, they mostly come with either high or low missingness ratios. That is, in the case of low missingness ratios, their impact is too small to clarify the differences between the methods. On the other hand, for high ratios of missing values, the performance is poor due to the lack of data needed by the imputation methods. In this case, the improvement brought by feature selection is not notably significant.

5.4.1.3 Symbolic Regression on Real-world Incomplete Data

Table 5.5 shows the symbolic regression results with real-world incomplete data sets. The results represent the mean of RSE (over 30 runs) for the test data when performing symbolic regression after imputing the missing values with each feature selection method.

Based on the experimental results, the proposed selection approach (IGPFS and IGPPS) provides better symbolic regression results compared to the other methods on the considered data sets. Interestingly, the features selected by DTFS may reduce the performance that can be obtained from the use of the full set of features. This is probably because DTFS might exclude some features that are relatively important for constructing good symbolic regression models. Such results indicate that the use of GP to feature selection provides features that are more compatible with the symbolic regression process compared with non-GP feature selection.

Table 5.5: The symbolic regression test RSEs on real-world incomplete data sets.

		CCN			CCUN		
		Mean	Std	ST	Mean	Std	ST
MeanIM	Full	0.5246	0.063	(+,=,=)	0.0344	0.0070	(-,-,-)
	DTFS	0.5314	0.0866	(-,-,-)	0.0325	0.0046	(+,-,-)
	IGPFS	0.5013	0.0654	(+,+,-)	0.0304	0.0063	(+,+,=)
	IGPPS	0.4886	0.0563	(+,+,+)	0.03001	0.0075	(+,+,=)
KNNIM	Full	0.5217	0.0549	(=,-,-)	0.0244	0.0102	(-,-,-)
	DTFS	0.5236	0.0691	(=,-,-)	0.0237	0.0056	(+,-,-)
	IGPFS	0.5101	0.0563	(+,+,-)	0.02104	0.0046	(+,+,-)
	IGPPS	0.4973	0.0643	(+,+,+)	0.0201	0.0038	(+,+,+)
RFIM	Full	0.5134	0.0531	(=,-,-)	0.0398	0.0059	(-,-,-)
	DTFS	0.5167	0.0559	(=,-,-)	0.0353	0.0046	(+,-,-)
	IGPFS	0.4955	0.0469	(+,+,-)	0.0313	0.0049	(+,+,-)
	IGPPS	0.4758	0.0561	(+,+,+)	0.0301	0.0058	(+,+,+)

5.4.2 Feature Reduction

Table 5.6 shows the feature reduction of the feature selection techniques on the considered data sets. In this table, feature reduction is measured by the number of distinct features selected by each method over different settings. The obtained results show that all the three feature selection methods can bring a huge reduction in the number of features.

Table 5.6: Feature reduction of the methods on different data sets.

	Selwood	Pah	Pdgfr	Mtp2	fri_c0	fri_c2	fri_c4	CNN	CCUN
All	54	113	321	1143	10000	10000	10000	126	144
DTFS	21	26	23	43	20	32	37	8	7
IGPFS	19	23	34	39	21	23	37	16	17
IGPPS	28	45	66	71	32	54	56	25	32

In most cases, IGPFS leads to the highest reduction in the number of features. This is because IGPFS only selects the features that are useful for the target variable based on IGP, while IGPPS selects more features as it adds predictor features to those ones selected by IGPFS. Although DTFS can provide fewer features in several cases such as on the real incomplete data sets CNN and CUNN, these features might not be the best ones for symbolic regression. This is clear from the symbolic regression performance comparison given in the previous section (see Table 5.5).

The feature reduction is not only related to the feature selection method but also the nature of the data sets. It seems that there are more redundant features in some data sets than in others. For example, among the real-world data sets, the Mtp2 data set has the most irrelevant/redundant features, which results in the highest feature reduction ratio. On the other hand, the artificial data sets show high reduction ratios due to the large amount of randomly added features.

5.4.3 Computation Time

Table 5.7 shows the computation time of the used feature selection techniques with different imputation methods on the considered data sets. In this table, the computation time shows the average of processing time per test instance on a machine with 1.7Mhz CPU and 8GB RAM.

The processing time is computed for performing the whole process to reach predicting the regression output for a test instance. This implies imputing the missing values in the test instance when employing the imputed training data by imputation methods. Consequently, selecting fewer features means less overload. Therefore, the processing time is related to the feature reduction showed in Table 5.6. The obtained results show that the use of feature selection reduces the required processing time considerably when compared to the case of using all features. IGPFS leads to more efficient models than the other feature selection methods. In most cases, it has the lowest computation time.

Table 5.7: The computation time of the symbolic regression evaluation (milliseconds per instance) for the compared feature selection techniques when used by different imputation methods.

		Selwood	Pah	Pdgfr	Mtp2	fri_c0	fri_c2	fri_c4	CNN	CCUN
MeanIM	All	4.73215E-08	1.14E-07	1.8E-07	6.26E-07	4.67E-06	4.67E-06	4.67E-06	2.39E-06	3E-06
	DTFS	3.76573E-08	8.2E-08	8.23E-08	2.6E-07	1.34E-06	1.34E-06	1.35E-06	2.35E-06	2.96E-06
	IGPFS	3.56578E-08	8.36E-08	8.6E-08	2.58E-07	1.34E-06	1.34E-06	1.35E-06	2.36E-06	2.96E-06
	IGPPS	4.1323E-08	9.1E-08	9.66E-08	2.7E-07	1.35E-06	1.35E-06	1.35E-06	2.36E-06	2.97E-06
KNNIM	All	0.01268865	0.027962	0.04394	0.149052	1.09843	1.09843	1.098117	0.56441	0.706805
	DTFS	0.010103925	0.020834	0.019973	0.062268	0.315806	0.317373	0.317138	0.553914	0.696388
	IGPFS	0.009320675	0.01966	0.021461	0.061955	0.317138	0.316355	0.318391	0.554541	0.697171
	IGPPS	0.0109655	0.022636	0.023654	0.064461	0.317686	0.318783	0.319879	0.556499	0.698032
RFIM	All	0.001514075	0.003306	0.005299	0.018198	0.134273	0.134312	0.134312	0.068938	0.086475
	DTFS	0.001197844	0.002587	0.00252	0.00758	0.038638	0.038906	0.038877	0.067808	0.0852
	IGPFS	0.001217009	0.002444	0.002549	0.007542	0.038647	0.038666	0.038839	0.067923	0.085181
	IGPPS	0.001303254	0.002769	0.002856	0.007963	0.038868	0.039002	0.038983	0.068047	0.085363

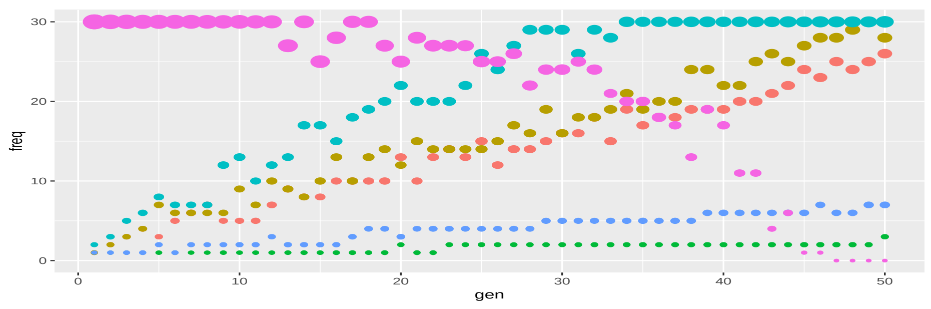
5.4.4 Further Analysis

Feature selectability refers to the ability of a feature to be selected. For the artificial data sets, the generator of the data is already known, which can be used to induce the relationships between the features and the regression output. This makes it easier to analyse the ability of the feature selection method to select the relevant features.

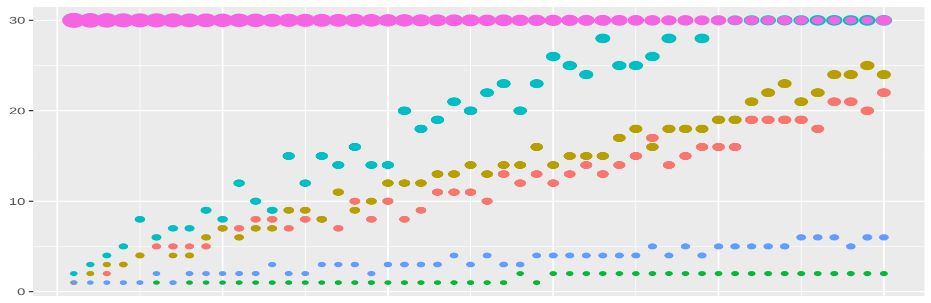
Figure 5.4 shows the relevant features ($f1, f2, f3, f4, f5$) and the non-relevant collective features (nrf) during the IGPFS evolutionary learning process on artificial data sets with synthetic incompleteness. This figure shows the selectability of these features measured by the frequency of the occurrence of the features and the ratio of this occurrence in each generation over the 50 generations. The frequency refers to how many times, out of the 30 runs, the feature appeared in the best-of-generation individual. However, the ratio means the appearance percentage of the feature in the generation with respect to all the features involved in the IGP program. This measure is indicated by the size of the shown circles in the figure.

It can be seen that, feature $f4$ has a higher occurrence frequency and

(a) The data set fri_c0.



(b) The data set fri_c2.



(c) The data set fri_c4.

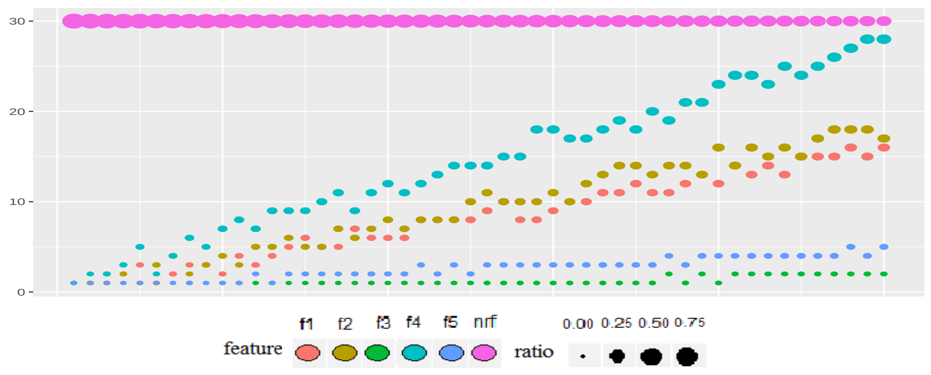


Figure 5.4: The irrelevant features (nrf) selected together with five relevant features during the evolutionary process with synthetic incompleteness on artificial data sets.

higher ratio compared with the other features. This indicates that $f4$ is more important than the others. In fact, according to the target function

(Equation (5.11)), $f4$ seems to have the highest impact on the output variable. When comparing the selection of the relevant features collectively, the advanced generations have more ability to select these features than the premature ones. This is clear from the decreasing curve of the irrelevant features, nrf, with the increase of the generation number.

On the other hand, the colinearity factor, which refers to the number of features that are dependent on other features, has a notable impact. The higher colinearity, the more likely to select non-relevant features, which is shown in the pink nrf nodes in Figure 5.4. For the fri_c0 data set, there is no colinearity between features, while the fri_c2 and fri_c4 data sets have colinearity degrees of two and four, respectively. This could be noticed as more irrelevant features are selected on the fri_c2 and fri_c4 data sets than on the fri_c0 data set.

5.4.4.1 Analysis of selectability of incomplete features

To have a deeper look, the selectability of the incomplete features is analysed. That is, we aim at examining how likely the incomplete features to be selected by the feature selection methods. Although, theoretically, for higher incomplete features ratio, it is more likely for incomplete features to be selected, but this also depends on the importance of these features. For this purpose, we intentionally imposed the incompleteness into the relevant features in the synthetic Friedman data set fri_c0 (with zero colinearity to avoid its possible influence), then study their selectability by the feature selection approaches.

This time, we used a wider range of instance-based missingness ratios (viz. 5%, 10%, 15%, 20%, 25%, 30%, 35%, 40%, 45%, 50%, 55%, 60%) in order to increase the chance of drawing solid conclusions on the selectability of features with respect to the incompleteness ratio. Note that, as we already know the features that will be affected by the missingness, the feature-based missingness ratio is not considered. Each missingness ratio has imposed randomly for 30 times and the obtained incomplete data are fed to the two feature selection approaches, i.e. the GP-based IGPPS method and the non-GP DTFS method. The probability of an incomplete feature to be selected represents the ratio of times it got selected. It is calculated by dividing the number of times the feature is selected by the total number of runs. Only IGPPS is considered since all incomplete features selected by IGPFS are included in IGPPS. The results corresponding to these experiments are shown in Figure 5.5.

Figure 5.5 shows that, although the GP-based approach is generally better, the difference between the two approaches is not notable for extreme missingness ratios, i.e. less than 15% and more than 55%. This is because both methods can easily select the relevant features from data with neglectable low missingness ratios. However, for high ratios of missing values, the data set is highly corrupted which hinders the applicability of the two approaches.

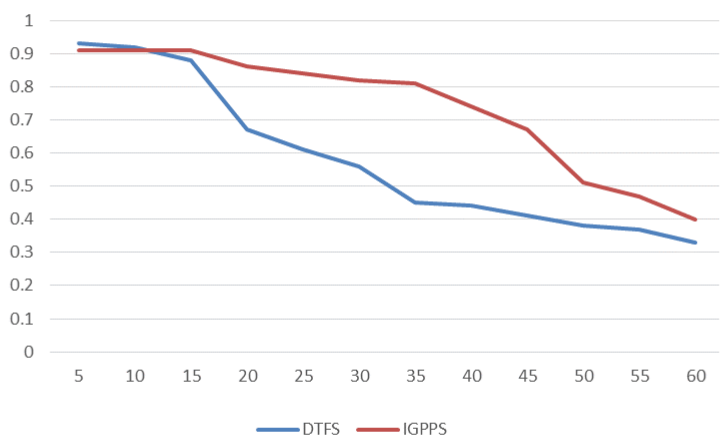


Figure 5.5: The probability of a relevant incomplete feature to be selected by the GP-based IGPPS method compared with the non-GP DTFS method considering different ratios of missingness on the synthetic Friedman data set fri_c0.

5.5 Chapter Summary

In this chapter, a GP-based approach that has the ability to select features from incomplete data for symbolic regression is presented. It employs interval-valued GP to select the features and their imputation predictors. As this approach deals with intervals rather than single-valued entries, an interval-based error metric is derived and used in the fitness function of the IGP-based feature selection to better guide the search process.

One important finding in this work is that performing feature selection before imputing the incomplete data can bring several benefits. It can lead to more accurate predictions while consuming less time. Another conclusion is that, for GP-based symbolic regression on incomplete data, GP-based feature selection is more beneficial than the decision tree-based feature selection. This can be observed from comparing the two approaches when employed for symbolic regression on incomplete data.

The proposed approach is utilised for improving imputation in symbolic regression on real and artificial data with different incompleteness ratios. The obtained results show that the proposed IGP approach outperforms the commonly used decision tree approach in directly selecting features from incomplete data for symbolic regression. This approach can not only achieve better symbolic regression performance but also select fewer features which in turn reduces the required computation time. Moreover, in addition to improving the performance by a better feature selection ability for regression, this performance is further improved when IGP is used to select predictors for imputation as well.

On the other hand, although the proposed method achieves efficient testing performance, the training stage efficiency could be improved. Furthermore, the proposed methods assume that there is enough data to achieve reasonable learning. The situation when there is limited data will be addressed in the next chapter.

Chapter 6

GP-based Transfer Learning in Symbolic Regression on Incomplete Data

6.1 Introduction

In the previous chapters where standard supervised machine learning is adopted, there was an implicit assumption that the existing data are enough to build models that can provide reasonable estimations for the missing values. However, what if the task in hand is in a domain with limited data (e.g. a small number of instances)? This is an extra challenge in addition to the data incompleteness. The existing data in the domain might not be adequate to provide reliable models to handle the missing data. To deal with such a situation, one promising approach is to transfer knowledge from different (yet related) complete domains. This can be achieved via transfer learning, which reuses learned knowledge in source domains to improve the learning in target domains [250, 262]. To perform transfer learning, feature-based transformations can be constructed to map feature spaces between domains such that their differences are reduced [177]. However, this has not been adequately investigated in symbolic regression on incomplete data. The related methods assume that the used data sets are complete. As transfer learning is powerful in dealing with limited domains, there is a need to utilise it to compensate for the shortage of data due to having both missing values and a small number of instances.

6.1.1 Chapter Goals

The main goal of this chapter is to develop a method that utilises transfer learning to deal with incomplete data in symbolic regression on domains with limited data. In this chapter, we aim to propose a novel multi-tree GP (MTGP) based transfer learning method to transfer knowledge from a complete source domain to improve symbolic regression in incomplete target domains. The specific objectives of this chapter include:

1) Developing a new mechanism to transfer both feature-based and instance-based *importance weighting knowledge* from the source domain to the target domain.

2) Developing new MTGP crossover and mutation operators based on a probabilistic selection of the best trees for crossover and the worst tree for mutating. The new genetic operators aim to maintain a higher level of exploration and exploitation to benefit the search effectiveness and accelerate the convergence of the evolutionary process. A new tree-wise selection method is also proposed for producing the next generation.

3) Designing an MTGP modelling process that produces feature transformations, which map complete source domains to incomplete target domains. The designed MTGP modelling considered two metrics for transfer learning. The first one is about domain-wise similarity. The Kolmogorov-Smirnov statistic, which tests the distribution similarity between data sets, is used to measure the domain mismatch. The second metric is related to the task learning performance. The transformation aims at improving the SR performance when handling the incompleteness in the target domain.

6.1.2 Chapter Organisation

This chapter is organised as follows. In Section 6.2, the proposed method is presented with a description of how it is used for symbolic regression on incomplete data. After that, the experiment setup is presented in Section 6.3, then the obtained results are given in Section 6.4. Finally, Section 6.5 draws the conclusions of this chapter.

6.2 The Proposed Method

Transfer learning provides a good solution when learning in domains with limited number of instances. In this work, a transfer learning-based method is developed to deal with the case that some of these (few) instances are incomplete. The main idea of the proposed method is to build a transformation that maps a source domain to a different (but related) target domain. Such a mapping is carried out using MTGP by constructing multiple features from the source features such that the transformed data has a similar representation to that of the target domain and leads to better target SR learning. The aim of the transformation is to compensate for the lack of knowledge in the target domain due to both missingness and shortage in the number of instances.

6.2.1 Overall Approach

Figure 6.1 shows the overall framework proposed in this work. First, an SR learning process is performed in the source domain with complete data. The outcomes of this process is SR models that are used to provide estimations for the weights of the features and instances in the source domain. This process is highlighted with a green colour background in Figure 6.1 and it will be detailed in Section 6.2.2. The extracted weights knowledge is then employed by the MTGP-based transformation to benefit the learning process in the target domain.

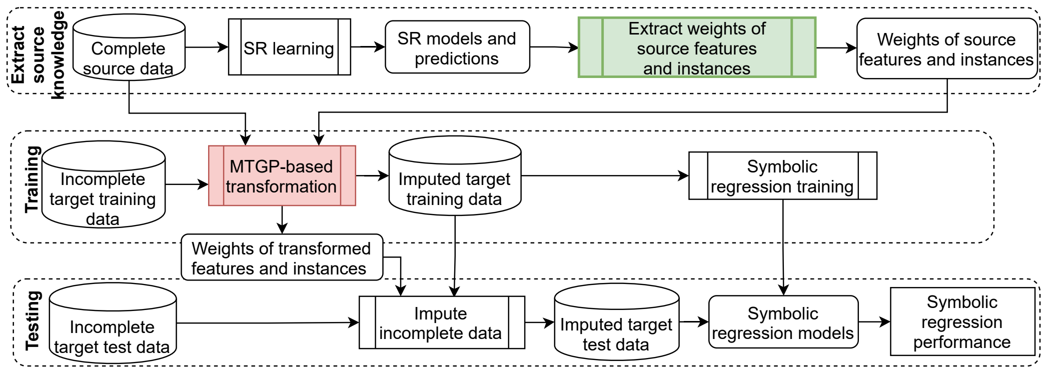

Extract source knowledge
Complete source data
SR learning
SR models and predictions
Extract weights of source features and instances
Weights of source features and instances
Training
Incomplete target training data
MTGP-based transformation
Imputed target training data
Symbolic regression training
Weights of transformed features and instances
Testing
Incomplete target test data
Impute incomplete data
Imputed target test data
Symbolic regression models
Symbolic regression performance

Figure 6.1: The Proposed Framework of Utilising Multi-tree GP-based transfer learning in Symbolic Regression on Incomplete Data.

The MTGP-based transformation (the component with red background) represents the core of this work and it involves other modules as will be presented in Subsection 6.2.3. It constructs an asymmetric mapping from the source feature space to the target feature space using MTGP by aligning the source data and the target training data. This mapping is used to transform the source data in a way that helps impute the missing values in the target domain effectively. It produces transformed features and instances along with their weights.

The transformed knowledge is utilised to impute the incomplete training and test sets in the target domain. After that, normal SR is carried out on the imputed target data. That is, the imputed training target data set is used for building SR models that are then applied to predict the outputs of the imputed test data.

In the next subsections, a detailed formalisation of the proposed MTGP transformation is presented. For this purpose, the notations given in Table 6.1 are used. Superscripts of t, s, and c will be used to refer to target, source, and constructed domains, respectively. For example, the symbols $\mathcal{D}^s$, $\mathcal{D}^t$, and $\mathcal{D}^c$ denote the source domain, the target domain, and the constructed domain, respectively.

Table 6.1: Summary of the used notations

Symbol	Meaning	Symbol	Meaning
$\mathcal{T}$	task	$\mathcal{D}$	domain
$\mathcal{X}$	feature space	$\mathcal{L}$	loss function
$\mathcal{M}$	MTGP individual	f	regression learned model
Y	desired value	$\hat{Y}$	predicted output
X	data matrix	I	instance vector
m	number of features	n	number of instances
X_j	the j^{th} feature	I_i	the i^{th} instance
W_{X_j}	weight of the j^{th} feature	W_{I_i}	weight of the i^{th} instance

6.2.2 Extracting Knowledge from the Source Domain

The learning process in the source domain is SR on complete data. The outcomes of this process are the learned SR models $\{f_l^s(\mathcal{D}^s)\}_{l=1}^r$, where r is the number of the independent runs (set to 30 in this work). These models are used to estimate the weights for the source domain features and instances based on their impact according to the learning process in the source domain. The more important features/instances in the source domain, the more likely they benefit the target domain, and therefore, the larger weight they get in the transformation construction process. The main steps of extracting knowledge based on the learning in the source domain are shown in Algorithm 6.

a) Source-wise feature weighting: The weights of the features are calculated based on their frequency in the best-of-run SR models as in Equation (6.1). To find the weight of each feature, the ratio of its frequency to the sum of the total frequency of all m^s source-domain features are taken, then the average of these ratios over the r runs is computed by dividing by r.

$$W^s_{X^s_j} = \frac{1}{r}\sum_{l=1}^{r} \frac{frq(X^s_j, f^s_l)}{\sum_{p=1}^{m^s} frq(X^s_p, f^s_l)} \tag{6.1}$$

where $W^s_{X^s_j}$ is the weight of the j^{th} source feature, X^s_j, according to the

Algorithm 6: Extracting source knowledge.

```
Input  : Complete source data, $\mathcal{D}^s$.
Output : Weights for the source features and instances.
1 for $l = 1$ to $r$ do
2     Construct an SR model, $f_l^s$, on $\mathcal{D}^s$;
3     For each feature, $X_j^s$, count its frequency $frq(X_j^s, f_l^s)$ in the evolved $f_l^s$ SR
        model;
4     For each instance, $I_i^s$, obtain its prediction error according to the model $f_l^s$
        as $(Y_i^s - f_l^s(I_i^s))^2$;
5 end
6 Compute the weight of source features by Equation (6.1);
7 Compute the weight of source instances by Equation (6.2);
8 Return the weights of source features and instances;
```

source domain learning process, and $frq(X_j^s, f_l^s)$ is the number of appearances of the j^{th} feature in the l^{th} learned GP program, f_l^s, where m^s is the number of source features.

b) Source-wise instance weighting: On the other hand, the prediction errors of the learned models on each instance is used to calculate their weights as follows.

$$W^s_{I_i^s} = \frac{1}{r}\sum_{l=1}^{r}\left(1 - \frac{(Y_i^s - \hat{Y}_{i,l}^s)^2}{\sum_{o=1}^{n^s}(Y_o^s - \bar{Y}^s)^2}\right) \qquad (6.2)$$

where $W^s_{I_i^s}$ is the weight of the i^{th} source instance, I_i^s, according to the source domain learning process, where n^s is the number of source instances, $\hat{Y}_{i,l}^s$ is the i^{th} predicted value by applying the l^{th} SR model to predict the i^{th} instance in the source domain, i.e. $\hat{Y}_{i,l}^s = f_l^s(I_i^s)$, Y_i^s is the i^{th} source desired value, and $\bar{Y}^s$ is the average of these desired values.

The weighting strategy of the source-domain features and instances assumes that the more impact of features/instances on the SR learning in the source domain the more gain they provide for the transformation process. The weights of the features are measured based on how frequent a feature is involved in the learned SR models. The higher frequency, the higher weights. The weighty features are expected to have more contributions when mapping the source domain to the target domain. For the instances, those with lowest errors in the source domain are given larger weights than others. This implies that these instances will have more contributions in the target domain. Such a strategy is based on that, the noisy and outlier instances tend to have more prediction bias, which means they might not be much useful in mitigating the data missingness issue in the target domain.

6.2.3 MTGP-based Transformation

Multi-tree GP is basically a standard tree-based GP method, except that each individual is represented using multiple trees instead of a single tree. It is used for constructing multiple features by assigning the expression evolved with each tree as a constructed feature. This mechanism is utilised to find a transformation that maps a source feature space to a new feature space, which is compatible with a target feature space. As all trees in an individual are evolving simultaneously, a set of produced features is constructed such that they perform well together.

Let $\mathcal{D}^s$ be a given source domain with m^s features, $\mathcal{X}^s = \{X^s_j\}^{m^s}_{j=1}$, and $\mathcal{M}$ is an MTGP individual with m^c trees, $\mathcal{M} = \{\mathcal{M}_j\}^{m^c}_{j=1}$. $\mathcal{M}$ is used to construct m^c features, $\mathcal{X}^c = \{X^c_j\}^{m^c}_{j=1}$, based on the source feature space, which transforms the source domain $\mathcal{D}^s$ into a new constructed domain $\mathcal{D}^c$. An illustration of this method is shown in Figure 6.2. The source domain features, $X^s_1, X^s_2, \ldots, X^s_7$, are used as inputs for the terminal nodes of each tree in the evolved individual. These trees represent the constructed features, X^c_1, X^c_2, X^c_3, and X^c_4, that are mapped one-to-one to the target domain features, X^t_1, X^t_2, X^t_3, and X^t_4, respectively.

The feature construction process is based on the source features (and GP function set), and aims to minimise the distribution mismatching between the source domain and the target domain. The constructed features

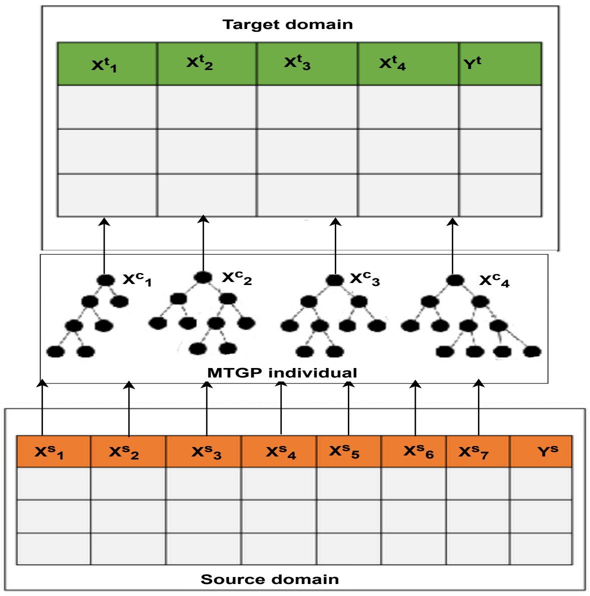


Figure 6.2: Multi-tree genetic programming-based asymmetric mapping from a source domain onto a target domain.

represent a transferred feature space, which has a reduced distribution mismatch between the two domains. The constructed features are used for transforming the features from the source domain to the target domain as follows. Each constructed feature is a variable expressed in terms of the source features. That is, $X^c_j = f(X^s_1, X^s_2, \ldots, X^s_{n^s})$. This constructed feature is meant to match the corresponding target feature, X^t_j, by minimising the distribution mismatch between the constructed feature and the target feature. As will be seen later, this is done by minimising a domain distribution distance, Ω, which implies minimising a Kolmogorov-Smirnov distribution distance, $KS(X^c_j, X^t_j), \forall j$.

The proposed method is designed as a wrapper-based method, where the feature transformation is evolved based on the performance feedback from the target regression task while reducing the distribution mismatch between the constructed domain and the target domain. The main steps of constructing the MTGP transformation are shown in Algorithm 7. As all trees in the individuals are evolving simultaneously, the whole set of produced features is constructed such that they perform well together.

Algorithm 7: The MTGP-based transformation.

```
Input  : Complete source data, $\mathcal{D}^s$, incomplete target training data, $\mathcal{D}^t$, and
         weights of source features and instances, $\{W^s_{X^s_j}\}^{m^s}_{j=1}, \{W^s_{I^s_i}\}^{n^s}_{i=1}$.
Output : Transformed data $\mathcal{D}^c$, imputed target training data, $\hat{\mathcal{D}}^t$, and weights
         for transformed features and instances.
1  Generate the initial population;
2  while not terminated do
3      foreach individual $\mathcal{M}$ in the population do
4          Apply $\mathcal{M}$ to $\mathcal{D}^s$ getting a constructed data $\mathcal{D}^c$;
5          Compute the weights for transformed features, $\{W^c_{X^c_j}\}^{m^c}_{j=1}$, and
             instances, $\{W^c_{I^c_i}\}^{n^c}_{i=1}$ (Subsection 6.2.3.1);
6          Normalise the computed weights getting $\{NW^c_{X^c_j}\}^{m^c}_{j=1}$ and $\{NW^c_{I^c_i}\}^{n^c}_{i=1}$;
7          Impute $\mathcal{D}^t$ getting a complete $\hat{\mathcal{D}}^t$ with the help of $\mathcal{D}^c$ and the
             normalised weights (Subsection 6.2.3.2);
8          Evaluate the fitness of $\mathcal{M}$ based on regression RSE on $\hat{\mathcal{D}}^t$ and
             distribution distance between $\hat{\mathcal{D}}^t$ and $\mathcal{D}^c$ (Subsection 6.2.3.3);
9      end
10     Select individuals from the current generation;
11     Produce the next generation using genetic operators (crossover, elitism, and
         mutation) (Subsection 6.2.3.4);
12 end
13 Obtain the best individual in the last generation (best-of-run) to be the
     constructed transformation, $\mathcal{M}$;
14 Return transformed data $\mathcal{D}^c$, imputed complete target training data, $\hat{\mathcal{D}}^t$, and
     weights for transformed features and instances $\{NW^c_{X^c_j}\}^{m^c}_{j=1}, \{NW^c_{I^c_i}\}^{n^c}_{i=1}$ based
     on the transformation $\mathcal{M}$;
```

6.2.3.1 Weighting Transformed Features and Instances

The features and instances of the source domain $\mathcal{D}^s$ are transformed by MTGP to a constructed domain $\mathcal{D}^c$. The source learning-based weights are used to construct weights for the transformed features and instances.

a) Transformation-wise feature weighting: $W^c_{X^c_j}$ is the weight of this feature in the constructed domains computed based on the weights of the source features that are used to construct the feature X^c_j as below.

$$W^c_{X^c_j} = \sum_{\forall X^s_p \in terminal(\mathcal{M}_j)} W^s_{X^s_p} \tag{6.3}$$

where $W^c_{X^c_j}$ is the summation of the weights of source features that contributed to constructing this feature, where $\mathcal{M}$ is the MTGP model used to construct the transformation and $terminal(\mathcal{M}_j)$ refers to the terminal set (leaf nodes) of the j^{th} tree in $\mathcal{M}$, which eventually constructs the j^{th} feature, X^c_j. These weights are then normalised to get $\{NW^c_{X^c_j}\}^{m^c}_{j=1}$, where $NW^c_{X^c_j} = W^c_{X^c_j} / \sum_l W^c_{X^c_l}$.

b) Transformation-wise instance weighting: The set of the transformed instances are also weighted. As these instances can be from the transformed domain or the target domain, different weighting criteria are used. The weight of neighbour instance that belongs to the target domain is set to one, while the weights of the instances that are from the constructed domain are calculated based on their weights in the source domain. These weights are then normalised getting $\{NW^c_{I^c_i}\}^{n^c}_{i=1}$.

$$W^c_{I_i} = \begin{cases} W^s_{I_i} & \text{if } I_i \in \mathcal{D}^s \\ 1 & \text{otherwise.} \end{cases} \tag{6.4}$$

6.2.3.2 Data Imputation

The extracted domain knowledge is a set of normalised feature weights, $\{NW^c_{X^c_j}\}^{m^c}_{j=1}$, and another set of normalised instance weights, $\{NW^c_{I^c_i}\}^{n^c}_{i=1}$. Such knowledge along with the transformed data are used to impute the target data by a modified weighted k-nearest neighbour (WKNN) imputation as shown in Algorithm 8. Note that, this modified WKNN is different from that used in previous chapters as it involves using the extracted feature importance weights and instance importance weights as shown in the algorithm. Although several imputation methods could be used, WKNN is selected because it can work as an implicit weighting scheme for the transferred source instances. In addition to calculating the weights of the source instances based on their impact on the prediction error, they are weighted also according to their contributions to imputing missing values in the target domain.

Algorithm 8: The imputation process.

```
Input  : Transformed weights, training data $\mathcal{D}^t$, transformed data $\mathcal{D}^c$, and an
         incomplete instance $I_a$.
Output : The imputed complete instance, $\hat{I}_a$.
1 foreach instance $I_i$ in $\mathcal{D}^t \bigcup \mathcal{D}^c$ do
2     Compute the weighted Euclidean distance between $I_a$ and $I_i$, using
        Equation (6.5);
3 end
4 Sort the distances and select the nearest $k$ instances;
5 Impute the missing values using the weighted average of the retrieved $k$ nearest
    neighbours using Equation (6.7);
6 Return the imputed complete instance, $\hat{I}_a$;
```

A modified weighted euclidean distance is used to measure the distances between instances, which are then used for another weighting in the WKNN imputation method. The instances retrieved by KNN are combined in a weighted average summation to impute the missing values. Only the features with non-missing values in the compared instances are used to calculate the distance. Let I_a be an incomplete instance to be imputed and $\mathcal{J}$ is the set of indexes of the features that have non-missing values in I_a, and let I_i be another instance, which also has non-missing values in the $\mathcal{J}$ features, then the distance between I_a and I_i is computed as below.

$$d(I_a, I_i) = \sqrt{\sum_{j \in \mathcal{J}} W^c_{X^c_j} (I_a[j] - I_i[j])^2} \tag{6.5}$$

where $I_a[j]$ and $I_i[j]$ are the values of j^{th} feature in the two instances I_a and I_i, respectively.

The weighted distance between the instances I_a and I_i is denoted as $\omega(I_a, I_i)$ and it is obtained as follows:

$$\omega(I_a, I_i) = \frac{1}{d(I_a, I_i)} \tag{6.6}$$

For the incomplete instance I_a, k nearest instances are retrieved and the weighted distances are obtained then normalised getting $N\omega(I_a, I_1), \ldots,$ $N\omega(I_a, I_k)$. If the distance between I_a and one of the other instances, say I_b, is zero, then their weighted distance is set to 1 and the weights of other instances become zeros, i.e. $N\omega(I_a, I_b) = 1$ and $N\omega(I_a, I_c) = 0, \forall c \neq b$. The estimation of the missingness of the instance I_a is then taken as the weighted average of the retrieved neighbours as follows.

$$\hat{I_a} = \sum_{i=1}^{k} \frac{(NW^c_{I_i} + N\omega(I_a, I_i))}{2} \cdot I_i \tag{6.7}$$

6.2.3.3 MTGP Fitness Function

MTGP aims at making the transformed domain, $\mathcal{D}^c$, as close as possible to the target domain, $\mathcal{D}^t$. This closeness is measured based on two metrics. The first one is the learning loss, $\mathcal{L}$, which refers to how the transformed domain benefits the target SR task after imputation, i.e. SR on $\hat{\mathcal{D}}^t$. The second metric is the distribution mismatch, Ω, which measures how similar the distribution of the transformed domain is to the distribution of the target domain Ω. The two measures are put together in the fitness function given Equation(6.8) and its details are given below.

$$fitness = \mathcal{L}(\mathcal{T}^t(\hat{\mathcal{D}}^t)) + \lambda\Omega(\mathcal{D}^t, \mathcal{D}^c) \tag{6.8}$$

where λ is used as a regularisation factor to balance the inter-dominance between the two measures $\mathcal{L}$ and Ω. λ is set to 0.3, which gives more importance to the learning task part than the domain distribution part.

a) The Loss Error $\mathcal{L}$: Once the target data are imputed, regression is performed on the resulted complete data and its performance is measured by the relative squared error (RSE) (shown in Equation (6.9)).

$$RSE(\hat{Y}, Y) = \frac{\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2}{\sum_{i=1}^{n}(Y_i - \bar{Y})^2} \tag{6.9}$$

where n is the number of the instances, $\hat{Y}_i$ is the i^{th} predicted value, Y_i is the i^{th} desired value, and $\bar{Y}$ is the average of the desired values.

The learning quality from the target domain side is based on the RSE of the SR process on the imputed target training data as in Equation (6.10).

$$\mathcal{L}(\mathcal{T}^t(\hat{\mathcal{D}}^t)) = RSE(f^t(\hat{X}^t), Y^t), \tag{6.10}$$

where $\hat{X}^t$ is the imputed data, Y^t is the target domain desired output and f^t is a regression model on the imputed target domain.

b) Distribution Mismatch, Ω: For the domain distribution mismatch part of the fitness function (Equation(6.8)), the distribution mismatch measure, Ω, needs to be identified. For this purpose, a statistical distance metric that measures the difference between domain distributions, i.e. between $\mathcal{D}^t$ and $\mathcal{D}^c$ is required.

Kolmogorov-Smirnov (KS) is a statistic that measures the difference between the empirical distribution function of the sample and the cumulative distribution function of the reference distribution, or between the empirical distribution functions of two samples [78]. The KS measure can be used to quantify the probability distribution difference between two underlying one-dimensional samples, S_1 and S_2 as in Equation (6.11).

$$KS(S_1, S_2) = \sup_x |F_{S_1}(x) - F_{S_2}(x)|, \tag{6.11}$$

where F_{S_1} and F_{S_2} are the empirical distribution functions of S_1 and S_2 respectively, and $\sup$ is the supremum function.

To derive the Ω distance, the KS distance that works on one-dimensional data is extended to measure the distribution distance between domains with multiple features. This is achieved by simply averaging the KS distances between the symmetric features in the two domains (Equation (6.12)). This measure represents the distribution-wise part of the regularised fitness function (6.8).

$$\Omega(\mathcal{D}^c, \mathcal{D}^t) = \frac{\sum_{j=1}^{m^t} KS(X_j^c, X_j^t)}{m^t}, \tag{6.12}$$

where X_i is the data extracted based on the i^{th} feature in the $\mathcal{D}$ domain.

6.2.3.4 The Proposed Genetic Operators

The genetic crossover and mutation operations are applied to individuals that are selected from the population based on their fitness values. In this selection process, better individuals are favoured over inferior individuals. In standard single-tree GP, the genetic operators are implemented at the subtree level. As MTGP individuals consist of multiple trees, the genetic operators can be performed by the means of standard subtree operators applied to trees selected from these individuals. Therefore, other selection mechanisms at the tree level are needed. In this work, the proposed tree-wise operators are based on the tree-wise quality, which represents the distribution mismatch between the constructed features and the target features.

Let $\mathcal{M}$ be an MTGP individual with m^t trees, $\{\mathcal{M}_k\}_{k=1}^{m^t}$, the feature X_k^c is constructed by the k^{th} evolved tree, $\mathcal{M}_k$, applied on the source feature space, i.e. $X_k^c = \mathcal{M}_k(X^s)$. The quality of this tree is measured by the distribution difference between the corresponding constructed feature and the k^{th} target feature, i.e. $\Omega_{\mathcal{M}_k} = KS(X_k^c, X_k^t)$. In this way, the distribution-wise part of the fitness function of the whole individual, Ω_M, can be written as follows.

$$\Omega_M = \frac{\sum_{k=1}^{m^t} \Omega_{\mathcal{M}_k}}{m^t}. \tag{6.13}$$

Based on that, the proposed mutation and crossover operators are described as below.

a) Mutation: In the standard subtree mutation, a randomly picked subtree of the selected individual is changed. This operator is extended by applying it to a tree in the selected MTGP individual. However, the index of the tree to be mutated should be selected. In this work, an operator called probabilistic worst index mutation (PWIM) is proposed. In this operator, although the worst trees are encouraged to be considered for mutation, other trees still have some possibility to be mutated. The possibility of picking the idx^{th} tree for mutation is based on a mutation probability calculated as in Equation (6.14).

$$PWIM(\mathcal{M}, idx) = \frac{\Omega_{\mathcal{M}_{idx}}}{\sum_{k=1}^{m^t} \Omega_{\mathcal{M}_k}} = \frac{\Omega_{\mathcal{M}_{idx}}}{m^t * \Omega_M} \tag{6.14}$$

where $\mathcal{M}$ is an MTGP individual, $\mathcal{M}_k$ is the k^{th} tree in $\mathcal{M}$, m^t is the number of trees, and idx is the index of the tree to be selected.

b) Crossover: Similar to mutation, MTGP crossover can be performed on the trees in the selected MTGP individuals. Some operators might allow crossover between trees that have different indexes in the individuals. For example, random-index crossover (RIC) [139] selects a tree from each individual randomly and performs standard GP crossover between them. Such operators allow exchanging genotype information between trees that reconstruct different target features, which in turn can limit the ability of each tree to specialise. For this reason, we only allow crossover between trees that are meant to construct the same target feature. This operator is referred to as same-index crossover (SIC) in [138], where the crossover is always performed to trees at the same index in each individual. This operator encourages individual trees to specialise while interacting with different individuals through the crossover operation. However, what index(s) should be considered? In this work, an operator called probabilistic best same index crossover (PBIC) is proposed.

Given two MTGP models, $\mathcal{M}$ and $\mathcal{M}'$, PBIC favours better trees to be considered for crossover, but does not entirely exclude poorly evolved trees. This is done by assigning a crossover probability for each pair of trees based on their average quality. The probability of the idx^{th} pair to be considered for the crossover is calculated using Equation (6.15).

$$PBIC(\mathcal{M}, \mathcal{M}', idx) = \frac{2 - (\frac{\Omega_{\mathcal{M}_{idx}}}{m^t * \Omega_M} + \frac{\Omega_{\mathcal{M}'_{idx}}}{m^t * \Omega_{\mathcal{M}'}})}{2(m^t - 1)} \tag{6.15}$$

where $\mathcal{M}, \mathcal{M}'$ are MTGP individuals, $\mathcal{M}_k$ is the k^{th} tree in $\mathcal{M}$, m^t is the number of trees, and idx is the index of the tree to be selected.

The rationale behind these operators is that, for the crossover, the best trees are mated to encourage producing better solutions. The crossover between good trees in the selected individuals tends to produce good trees in the new offspring individuals. However, as it will be desired for all trees (features) to be well constructed, the poorly evolved trees are targeted to be mutated during the evolutionary process. The mutation operator aims at arousing such trees to get different solutions. In both cases, the other trees should have the chance to be mated/mutated and this is why the operators are designed in a probabilistic manner,

6.3 Design of Experiments

6.3.1 Data Sets

The proposed method is evaluated on six real-world data sets that are frequently used in the existing transfer learning regression research [272, 175, 46]. Two of these data sets, namely, the Auto-mpg and Imports data sets, contain missing values, whereas the others are complete. In these data sets, the incomplete instances are holdout during the modelling process, but these instances are considered later to validate the proposed method when dealing with real incompleteness.

To simulate the transfer learning scenarios, preparation steps similar to those in the related work are applied [47]. Each data set is divided into source and target domains with different distributions. To achieve that, the data are sorted according to a certain continuous feature then the source domain is obtained by selecting the first two-thirds of the data and the remaining instances are used as target domain data. Then, different normal distributions are used to perturb the features with random numbers in the two domains to guarantee the difference in the distribution between the two domains. The statistics of the outcomes of this process are shown in Table 6.2, where m is the number of features in the data set and n^s (n^t) is the number instances in the source (target) domain after data split. For more details on the original data sets, the reader can refer to the machine learning repository UCI [62].

Table 6.2: Statistics of the used data sets

Data set	m	n^s	n^t	Split feature	Split value
Housing	13	369	137	TAX	600
Concrete	9	687	343	fine	803.7
Yacht	6	198	110	Froude	0.35
Forestfires	13	347	170	DC	694.8
Auto-mpg	6	258	134	acceleration	16.5
Imports	14	125	70	length	176.6

For the heterogeneous case, it is necessary to have source and target domains with different feature spaces. For this purpose, a feature-wise split is considered such that the set of features used in the target domain is different from the one in the source domain. The feature space is split into two halves: Half1 and Half2. Half1 (Half2) is the first (second) half of the features as in the order of the original data set. These sets are mutually exclusive and if the total number of features is odd, the first half will have one more feature. Specifically, the sets are defined as in Equation (6.16). Similar to [263, 264], the first half of the features is used for the source domain while all features are used for the target domain. We refer to this setting as "Half1-All". Moreover, in this work, two other possibilities, "All-Half2" and "Half1-Half2", are also considered. Then, different normal distributions are used to perturb the features with random numbers in the domains to guarantee the differences between scenarios and domains.

$$\begin{aligned} Half1 &= \{X_j : j \leq \lceil m \rceil\} \\ Half2 &= \{X_j : j > \lceil m \rceil\} \end{aligned} \tag{6.16}$$

On the other hand, to simulate the missingness situation, for each target data set, incomplete data sets are synthetically generated by imposing MAR missingness ratios. The MAR mechanism is used as it is more suitable to evaluate the imputation performance [270]. This process is repeated 30 times for the missingness ratios $10\%, 30\%$, and 50% considering 40% of the features randomly. Eventually, 90 incomplete data sets are generated for each target data set (3 missingness ratios $\times$ 30 random versions).

6.3.2 Benchmark Methods and Settings

The proposed method is compared with the following methods.

WKNNIM [25]: This method follows traditional learning without using transfer learning where only the target domain is considered in the learning process. It is done by first imputing the target incomplete data using the WKNNIM imputation method then the imputed complete data are used for SR learning. This method was described and used in early chapters as well. The goal of considering this method is to investigate whether the achieved transfer learning is positive or not.

WKNNIMDC: This method is learning after applying WKNNIM on training data resulted from combining the target training data and the source data without any adaption between the domains. It simply combines source instances and the target domain training instances then the combined data are used for learning the target task. This approach is considered to examine the effectiveness of the proposed transformation. However, this method is not applicable in the heterogeneous case as the feature spaces are different, which disallows the concatenation of the corresponding two data sets.

RFIM [99]: To examine whether the proposed transfer learning method can also achieve better performance when compared with other imputation approaches rather than the underlying KNNIM, another popular imputation method is also considered in the case of no transfer learning. This method is random forest imputation (RFIM) and it is chosen because it represents a different imputation strategy than WKNNIM. For the same reason, **RFIMDC** is also considered. It works by learning after applying the imputation method RFIM on training target data combined with the source data without any adaption.

MTGPTL: This method builds an MTGP-based transformation considering only the loss function part of the fitness function (regression error in the target domain Equation (6.10)). This transformation is performed on selected source domain instances then the transformed ones are utilised in the target domain learning. It does not consider importance weights for instances and features. Moreover, the genetic operators are based on a random selection of the trees to be considered for mutation and crossover.

MTGPTL uses the same-index crossover (SIC) approach for individuals' crossover [139]. It works by randomly picking a tree index and then applying a standard single-tree GP crossover on the two trees at this index in the two multi-tree individuals. This approach increases the ability of each constructed feature to specialise while improving the performance of being used with the other evolved features. For the mutation, it is performed by applying a standard single-tree GP mutation on a randomly chosen tree. As each individual is intended to be a transformation from the source domain to the target one, the number of trees is set to the number of the features in the target feature space. In addition to random constants, the source problem features form the terminal set of MTGP.

MTGPDA: In this method, in addition to minimising the loss error as in MTGPTL, the source-target domain distribution mismatch shown in Equation (6.12) is minimised. That is, the fitness function in MTGPDA is the same as in pMTGPDA (Equation (6.8)). However, the other components of MTGPDA are similar to MTGPTL. The genetic operators in MTGPDA are based on a random selection of the trees and the importance weights of features/instances are not considered.

Table 6.3: The used values for MTGP parameters.

Parameter	Value
Generations	50
Population size	512
Crossover rate	0.8
Mutation rate	0.2
Elitism	5
Selection	Tournament selection with a size of 7
Maximum depth	8
Initialisation	Ramped-half-and-half
Function set	+, -, *, protected %, sine, exp
Terminal set	features and constants $\in (-1, 1)$

The proposed method is referred to as **pMTGPDA** in which MTGPDA is extended such that feature/instance weight knowledge is transferred between the domains and the genetic operators are performed in a probabilistic manner.

The performance of the different methods is evaluated based on SR performed on the imputed data sets measured using the RSE metric (Equation (6.9)). As GP is stochastic, 30 independent runs are performed for each experiment. In the target domain, the data set is split into $30 : 70$ training-test subsets. This split is common in transfer learning research to emphasise the shortage of data available for learning in the target domain [47]. To assess the difference between the results, a Wilcoxon non-parametric statistical test with a significance level of 0.05 is used. The proposed method is implemented using the DEAP python package with the settings shown in Table 6.3. The evolutionary process stops when reaching the maximum generation number. These algorithmic parameters are chosen empirically based on validation data sets (subsets of the training data keeping the test data unseen). For SR, imputation (WKNN and RFIM), and Kolmogorov-Smirnov, the python packages gplearn, missingpy, and SciPy are used respectively with default settings.

6.4 Results and Discussions

In this section, the experimental results are presented and discussed. We only show the test results without the corresponding training performance as the focus is on the ability of the methods to generalise on unseen data.

6.4.1 Symbolic Regression in *Homogeneous* Domains

In homogeneous transfer learning, the source and target feature spaces are the same but their marginal distributions are different, i.e. $\mathcal{X}^s = \mathcal{X}^t$ but $P(\mathcal{X}^s) \neq P(\mathcal{X}^t)$. Figure 6.3 shows the homogeneous transfer learning results of the proposed method compared with other methods on different data sets with three missingness ratios. After imputing the missing values, SR models are trained using the imputed training data sets and the RSE results of applying these models on the imputed test data sets are shown in Figure 6.3.

From the experimental results, all MTGP-based methods show the ability to achieve positive transfer learning in most cases. The results obtained when using MTGP-based transfer learning are better than those of using the traditional learning-based methods that only consider the target domain, i.e. WKNNIM and RFIM. Moreover, MTGP-based methods outperform the methods of combining the source domain with the target domain without transformation, WKNNIMDC and RFIMDC.

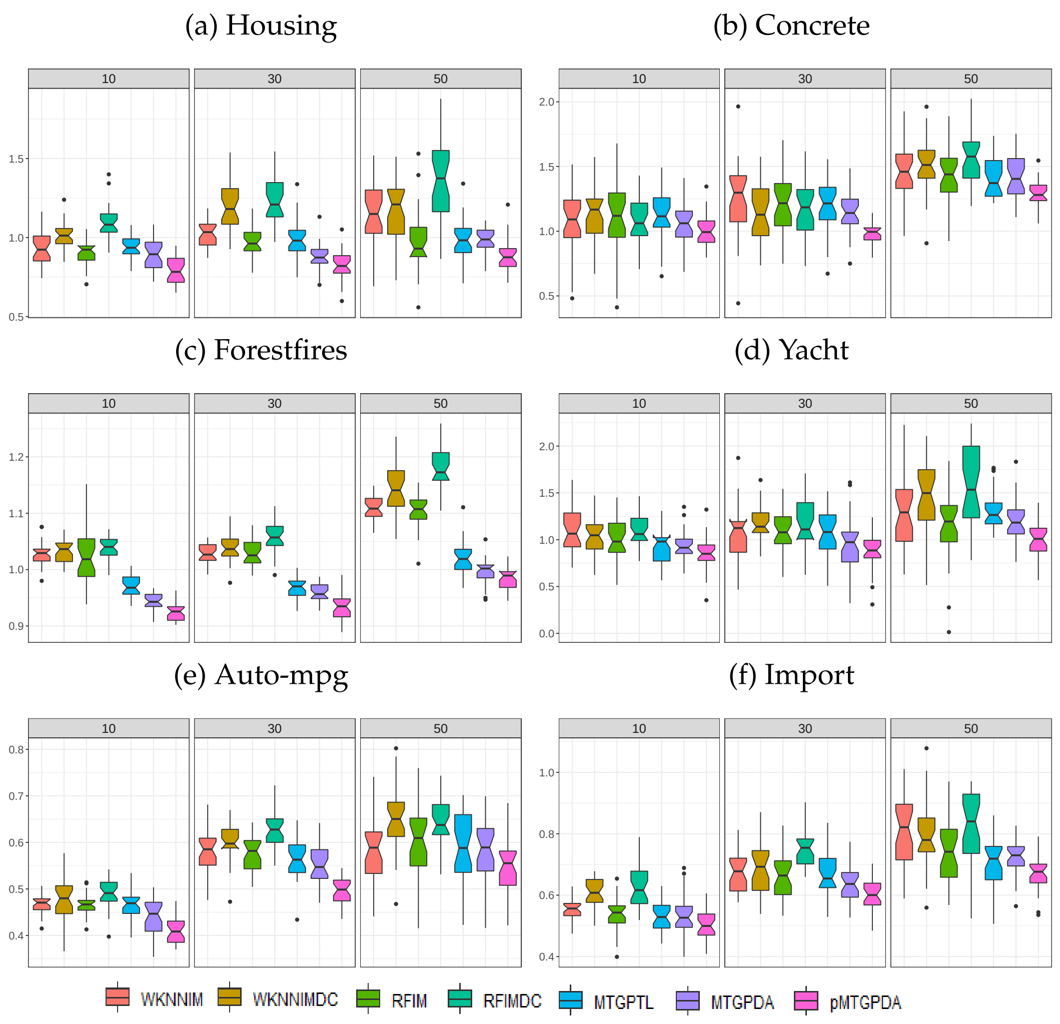


Figure 6.3: The SR test results (in RSE) on real-world data with synthetic incompleteness in the **homogeneous** transfer learning scenario per missingness ratio (10,30,50).

Among the MTGP-based methods, the proposed method, pMTGPDA, tends to provide the best SR performance. MTGPDA is better than MTGPTL as, in addition to improving the regression performance of MTGPTL, it bridges the distribution gap between the two domains. On top of that, pMTGPDA advances MTGPDA by utilising feature/instance weight knowledge and involving genetic operators that guide the distribution mismatch reduction process.

Although FRIM is better than KNNIM when learning based on the target domain alone, the figures differ when domains' data are combined as KNNIMDC outperforms RFIMDC. RFIMDC has the worst performance among all used methods. This is because RFIMDC learns from data sets that do not have consistent distribution and it works by building feature-based imputation models for the incomplete features. Consequently, these models that are constructed considering source domain data that have different distribution (in addition to the target data) will be used to estimate the missing values in the target domain. The results of both WKNNIMDC and RFIMDC show that transferring the data from the source domain to the target domain without domain adaptation might not improve the target learning, rather, it is more likely to degrade the learning performance. These results highlight the superiority of the MTGP-based transfer learning.

6.4.2 Symbolic Regression in Heterogeneous Domains

For the heterogeneous transfer learning scenario, the feature space of the source domain is different from the feature space of the target domain, i.e. $\mathcal{X}^s \neq \mathcal{X}^t$ and $P(\mathcal{X}^s) \neq P(\mathcal{X}^t)$. According to the settings mentioned earlier, we have three cases: "Half1-All", "All-Half2", and "Half1-Half2". The results of the "Half1-All","All-Half2", scenarios are shown in Figures 6.4, 6.5, and 6.6, respectively.

In the "Half1-All" case, the feature space of the source domain consists of only the first half of the original data while the target domain contains all the available features. This means that SR in the target domain will still use a feature set similar to that used in the homogeneous scenario. However, there are fewer features in the source domain to be used for constructing the MTGP transformations from the source domain to the target domain. Nevertheless, as can be seen from the results shown in Figure 6.4, MTGP-based methods manged to advance the SR performance in most cases. In particular, the proposed pMTGPDA method showed the best SR performance compared with any other method.

In the "All-Half2" scenario, the feature space of the source domain consists of all the available original features while the target domain has only the second half of the features. In this case, the full set of features are considered to construct the MTGP transformations from the source domain to the target domain. However, the target domain itself has a limited number of features to be used for building the target SR models. Therefore, as can be noticed from the results shown in Figure 6.5, the SR performance of all methods is generally poor. Such results are expected as the performance is calculated in the target domain, which, in this case, has only a subset of the original features. For such a difficult situation, pMTGPDA still managed to enhance the SR performance in the target domain. This enhancement is due to the knowledge transferred from the source domain, which has all the available features.

For the extremest heterogeneity scenario, "Half1-Half2", both domains have completely different feature sets. Besides the limited features used for SR in the target domain, there is a limited set of features in the source domain to be utilised for constructing the target features by MTGP-based transfer learning methods. However, even in this extreme scenario, the proposed pMTGPDA method still achieved considerable improvements in the target SR performance.

Overall, it is clear that all the MTGP-based methods provide positive transfer learning even when the feature space of the source domain differs from that of the target feature space. Regardless of the considered transfer learning scenario, the pMTGPDA method was always able to advance the results of the benchmark methods.

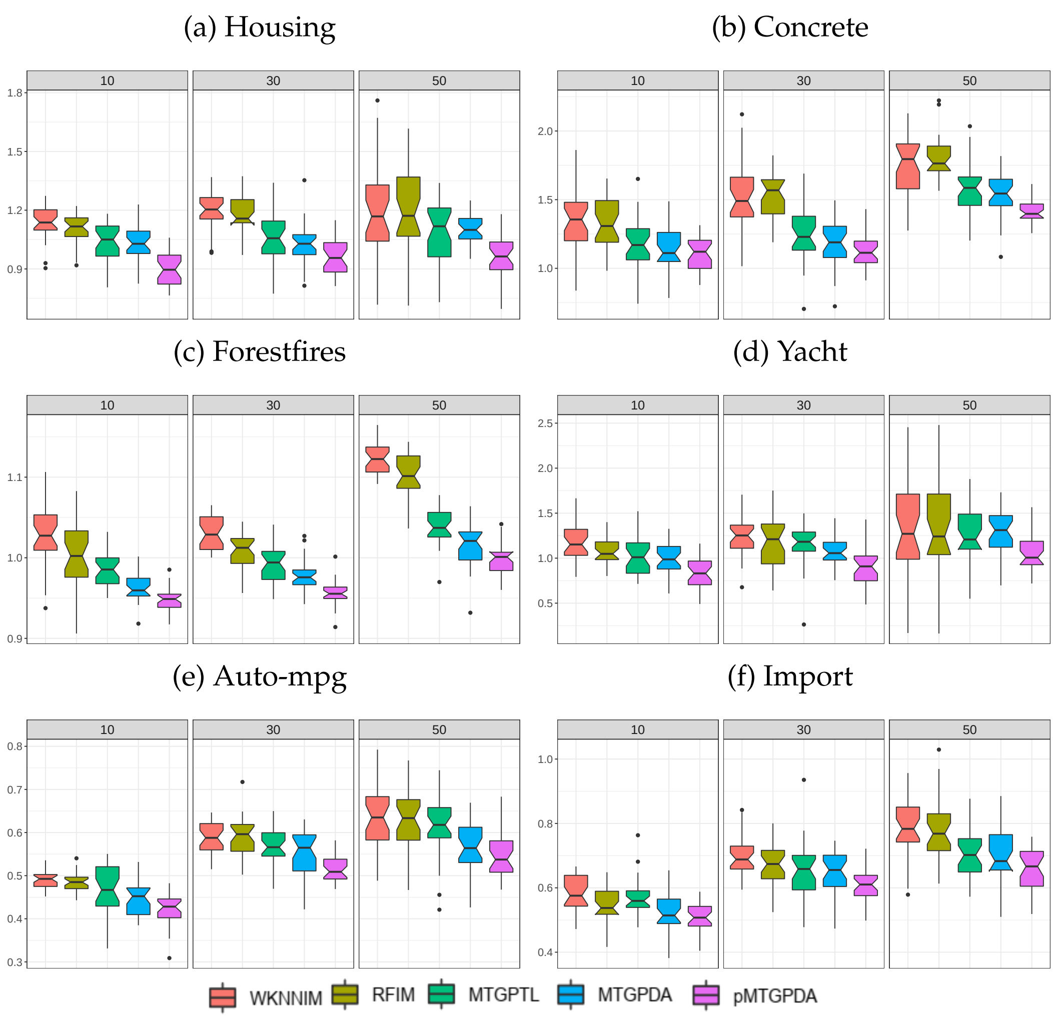


Figure 6.4: The SR test results (in RSE) on real-world data with synthetic incompleteness in the **heterogeneous** transfer learning scenario (**Half1-All**).

6.4.3 Symbolic Regression with Real Incompleteness

Since the data sets Auto-mpg and Imports are originally incomplete, they can be used to examine the validity of the proposed method on real-world incompleteness. As mentioned earlier, the incomplete instances are separated and the modelling is done on the remaining complete data. How-

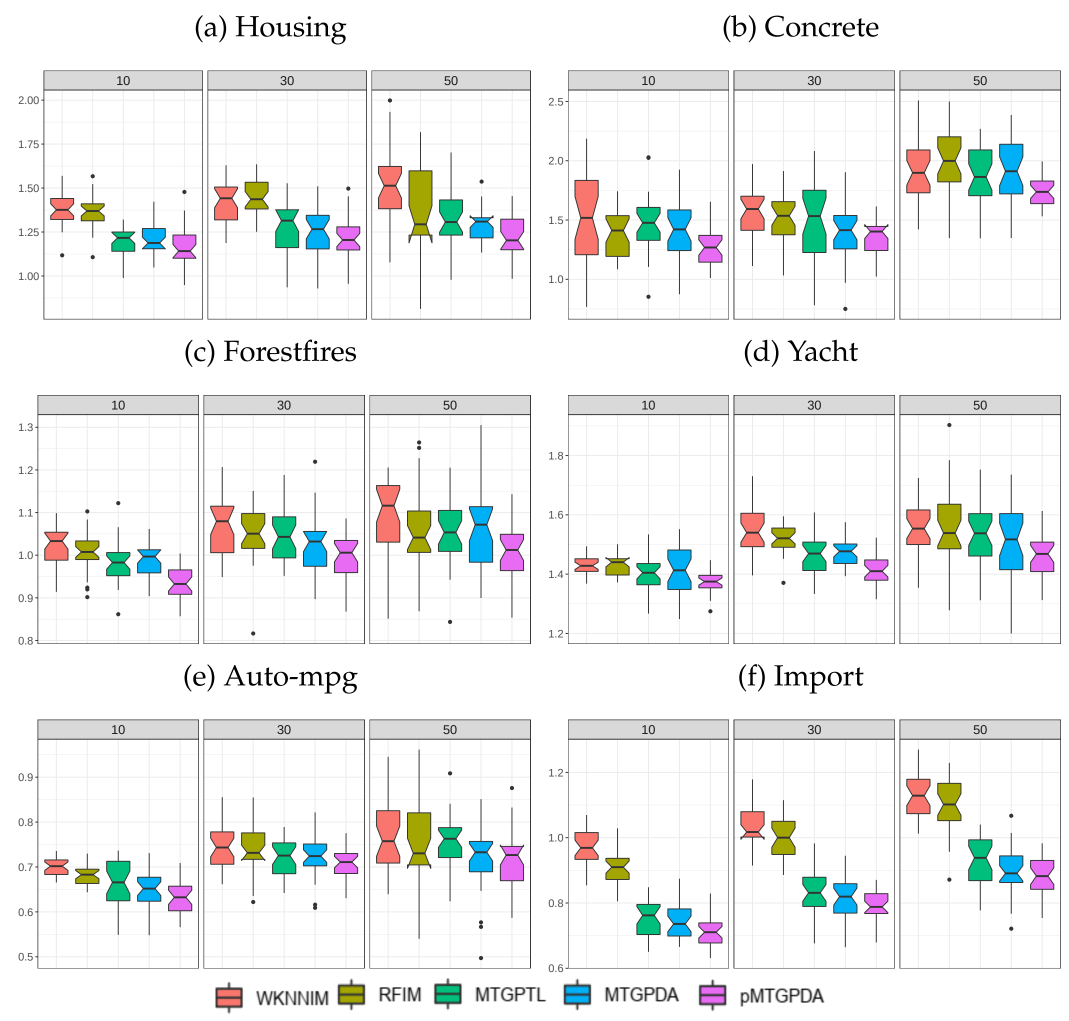


Figure 6.5: The SR test results (in RSE) on real-world data with synthetic incompleteness in the **heterogeneous** transfer learning scenario **(All-Half2)**.

ever, to compare the different methods on reality incompleteness, the learned models based on synthetic settings are applied to predict the hold out incomplete instances.

For each method, all possible models obtained from the homogeneous situation are used to predict the incomplete instances and their average

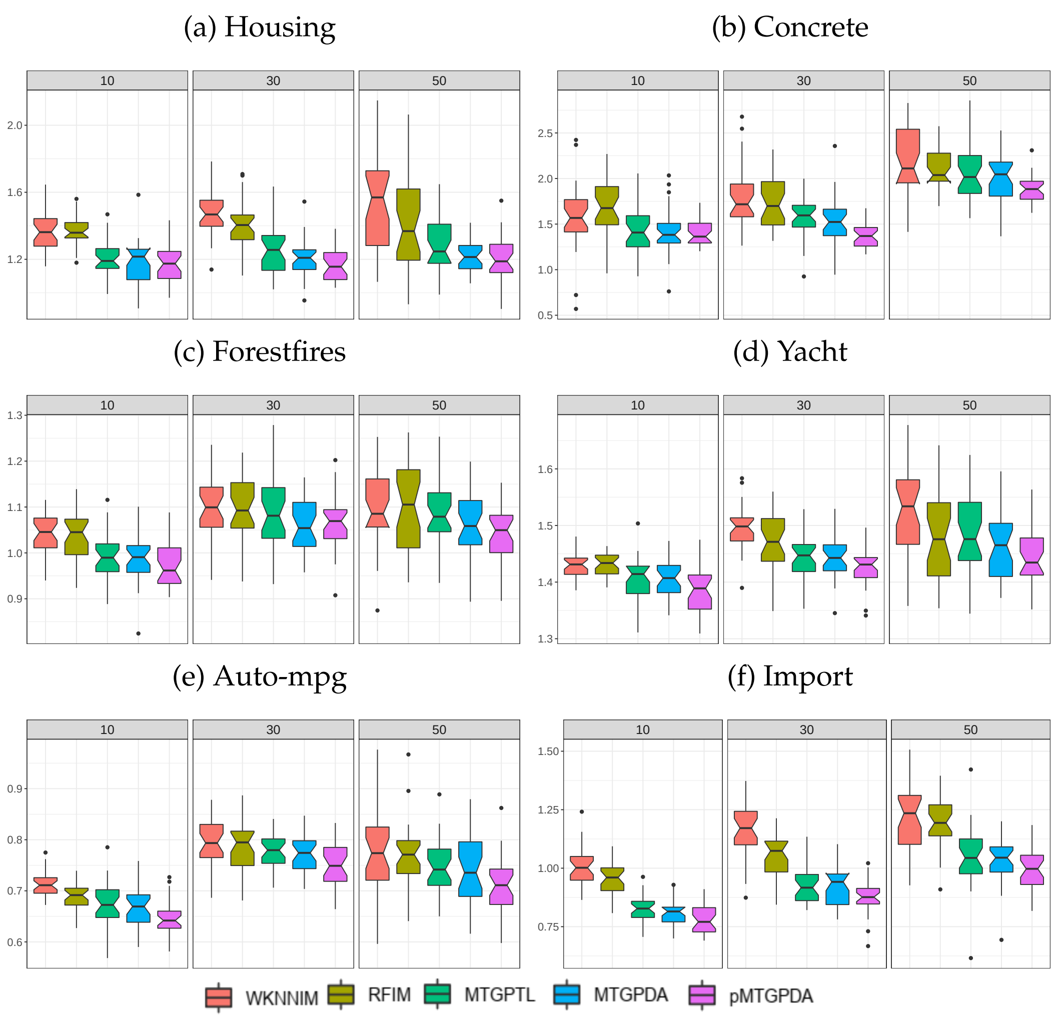


Figure 6.6: The SR test results (in RSE) on real-world data with synthetic incompleteness in the **heterogeneous** transfer learning scenario (**Half1-Half2**).

and standard deviation are calculated and shown in Table 6.4. Moreover, the significance in the difference between the results obtained using each method and the proposed method is denoted as "ST". For each method, "-" ("+") means that this method significantly outperforms (outperformed by) the pMTGPDA method while "=" appears when there is no significant difference.

Table 6.4: The symbolic regression results on real incomplete instances

Method	Data	Auto-mpg	Imports
WKNNIM	Mean	0.9523	0.7213
	Sdev	0.2347	0.0799
	ST	+	+
WKNNIMDC	Mean	1.0241	0.7681
	Sdev	0.3264	0.0654
	ST	+	+
RFIM	Mean	0.9326	0.7165
	Sdev	0.1973	0.028
	ST	+	+
RFIMDC	Mean	1.1954	0.8161
	Sdev	0.2671	0.0844
	ST	+	+
MTGPTL	Mean	0.9451	0.7201
	Sdev	0.1876	0.0435
	ST	+	+
MTGPDA	Mean	0.9346	0.7114
	Sdev	0.1651	0.0378
	ST	+	+
pMTGPDA	Mean	0.9187	0.7017
	Sdev	0.1599	0.0314

The results show that the pMTGPDA method significantly outperforms the other methods on the two data sets including the MTGP-based methods. These patterns are consistent with those obtained in the synthetic missingness case. As pMTGPDA has more stable models in the construction phase with artificial incompleteness, it leads to better predictions in the application case with real incompleteness. These results provide an important indicator that this approach might be effective regardless of the missingness operator.

6.4.4 Statistical Significance Test Results

In this subsection, we provide the results of the statistical significance tests for the results shown above to verify if the proposed method is significantly better than the benchmark methods. For this purpose, the Wilcoxon non-parametric statistical test with a significance level of 0.05 is applied to compare the results of the proposed method with the benchmark methods on each incomplete copy (data set) generated from each data set.

When comparing the proposed method, pMTGPDA, with each benchmark method on a specific data set, there are totally 90 comparisons ($90 = 30 \times 3$ per each missingness ratio). The number of times where pMTGPDA significantly outperforms the benchmark method is counted as a "+" comparison, while it is counted as a "$-$" comparison when pMTGPDA significantly under-performs, otherwise it is counted as a "$=$" comparison as there is no significant difference between the two methods.

Table 6.5 shows the statistical test for the SR results on real-world data with synthetic incompleteness in the homogeneous transfer learning scenario. The statistical test results for the heterogeneous transfer learning scenario considering the situations Half1-All, All-Half2, and Half1-Half2, are shown in Table 6.6, Table 6.7, and Table 6.8, respectively. Note that the methods WKNNIMDC and RFIMDC are not shown in the results of the heterogeneous scenarios as they are not applicable when the feature spaces of the source domain and the target domain are different.

These results verify the supremacy of the pMTGPDA method compared with any other method. It can be seen that the MTGPDA method is better than the other methods, whereas RFIMDC is the worst. This is expected as MTGPDA works by addressing the knowledge limitations in the incomplete target domain using domain adaptation from a richer complete source domain.

In addition to what MTGPDA does, pMTGPDA utilises transferring different aspects of learning such as feature importance and instance importance. On top of all of that, pMTGPDA involves new genetic opera-

Table 6.5: The statistical test for the SR results on real-world data with synthetic incompleteness in the **homogeneous** transfer learning scenario.

	WKNNIM			WKNNIMDC			RFIM			RFIMDC			MTGPTL			MTGPDA		
Data	+	−	=	+	−	=	+	−	=	+	−	=	+	−	=	+	−	=
Housing	84	0	6	90	0	0	80	1	9	90	0	0	75	2	13	71	3	16
Concrete	78	2	10	90	0	0	79	4	7	90	0	0	74	2	14	72	11	7
Forestfires	90	0	0	90	0	0	90	0	0	90	0	0	87	0	3	82	0	8
Yacht	82	3	5	87	0	3	81	2	7	90	0	0	71	4	15	68	5	17
Auto-mpg	85	0	5	90	0	0	84	0	6	90	0	0	74	2	14	71	3	16
Import	88	0	2	90	0	0	72	6	12	90	0	0	72	7	11	73	8	9
Total	**507**	**5**	**28**	**537**	**0**	**3**	**486**	**13**	**41**	**540**	**0**	**0**	**453**	**17**	**70**	**437**	**30**	**73**

Table 6.6: The statistical test for the SR results on real-world data with synthetic incompleteness in the **heterogeneous** transfer learning scenario **(Half1-All)**.

	WKNNIM			RFIM			MTGPTL			MTGPDA		
Data	+	−	=	+	−	=	+	−	=	+	−	=
Housing	81	0	9	81	0	9	73	0	17	70	0	20
Concrete	73	5	12	71	4	15	71	5	14	67	9	14
Forestfires	90	0	0	90	0	0	84	0	6	83	2	5
Yacht	78	6	6	75	7	8	73	3	14	61	12	17
Auto-mpg	83	0	7	80	1	9	69	5	16	66	9	15
Import	85	1	4	70	5	15	70	8	12	67	9	14
Total	**490**	**12**	**38**	**467**	**17**	**56**	**440**	**21**	**79**	**414**	**41**	**85**

tors that guide the evolutionary process of constructing the MTGP-based transformations. All of these extensions make pMTGPDA better than MT-GPDA and consequently better than the remaining methods.

Table 6.7: The statistical test for the SR results on real-world data with synthetic incompleteness in the **heterogeneous** transfer learning scenario **(All-Half2)**.

	WKNNIM			RFIM			MTGPTL			MTGPDA		
Data	+	−	=	+	−	=	+	−	=	+	−	=
Housing	86	0	4	84	0	6	75	0	15	72	0	18
Concrete	89	0	1	83	0	7	74	2	14	73	6	11
Forestfires	90	0	0	90	0	0	88	0	2	87	0	3
Yacht	83	2	5	82	1	7	79	0	11	76	4	10
Auto-mpg	86	0	4	84	0	6	81	1	8	72	5	13
Import	88	0	2	85	0	5	80	2	8	77	2	11
Total	**522**	**2**	**16**	**508**	**1**	**31**	**477**	**5**	**58**	**457**	**17**	**66**

Table 6.8: The statistical test for the SR results on real-world data with synthetic incompleteness in the **heterogeneous** transfer learning scenario **(Half1-Half2)**.

	WKNNIM			RFIM			MTGPTL			MTGPDA		
Data	+	−	=	+	−	=	+	−	=	+	−	=
Housing	82	0	8	84	0	6	74	0	16	72	0	18
Concrete	86	0	4	79	0	11	71	2	17	69	5	16
Forestfires	88	0	2	90	0	0	84	0	6	84	2	4
Yacht	81	1	8	80	0	10	73	0	17	71	7	12
Auto-mpg	82	0	8	82	0	8	78	1	11	69	5	16
Import	83	0	7	81	0	9	77	2	11	72	2	16
Total	**502**	**1**	**37**	**496**	**0**	**44**	**457**	**5**	**78**	**437**	**21**	**82**

6.4.5 Computational Cost

Here, we compare the proposed method, pMTGPDA, with other MTGP-based methods with respect to their learning computational time, namely MTGPTL and MTGPDA. The other methods do not include the transfer learning process and they perform only imputation and symbolic regression on the target domain. Table 6.9 shows the average computational time of constructing MTGP transformations (in rounded seconds$\times10$) for the GP-based transfer learning methods along with the number of features.

Table 6.9: Mean computational time for constructing MTGP transformations (in rounded seconds$\times10$).

Data	#Features	#Instances	MTGPTL	MTGPDA	pMTGPDA
Housing	13	506	967	1213	1322
Concrete	9	1030	902	1104	1189
Yacht	6	308	464	689	751
Forestfires	13	517	953	1263	1351
Auto-mpg	6	392	587	757	841
Imports	14	195	871	1025	1211

Generally, the MTGPDA method is more time consuming than MTGPTL, and pMTGPDA is the one with the highest computational time. This is because MTGPDA has the extra step of minimising the distribution mismatch compared with MTGPTL, while pMTGPDA has the feature/instance weighting steps on top of that. These methods are based on MTGP optimisation where each individual has a number of trees as many as the available features. Similarly, the data sets with higher number of instances require more learning time.

6.4.6 Further Analysis

6.4.6.1 Learned Transformations

As the main goal of the proposed method is to construct a feature-based transformation that maps the source domain to the target domain, an example of such a mapping is shown in Listing 1. This example is taken from the homogeneous transfer learning scenario on the Yacht data set when using pMTGPDA with probabilistic worst index mutation and probabilistic best index crossover (PWIM, PBIC). The constructed features X_i^c, $i = 1, \dots, 6$ are expressed in terms of the source features $X_j^s, j = 1, \dots, 6$. Each constructed feature, X_i^c, is intended to ensemble a corresponding target feature, X_i^t.

Listing 1. An MTGP transformation example from the Yacht source data to the target data

$$\begin{aligned}
X_1^c = {} & \%(*(e(e(e(X_1^s))), sine(+(X_4^s, e(e(-0.5208))))), \\
& e(\%(*(+(X_6^s, -0.5208), e(-0.3250)), sine(e(X_1^s))))) \\
X_2^c = {} & +(-(-(*(X_2^s, X_5^s), \%(X_1^s, X_5^s)), \%(X_1^s, X_5^s)), *(\%(e(X_4^s), +(X_5^s, X_2^s)), \\
& \%(e(X_4^s), sine(X_5^s)))) \\
X_3^c = {} & -(e(\%(+(sine(e(-(\%(X_5^s, X_1^s), +(X_4^s, X_4^s)))), \\
& +(-(sine(+(X_3^s, X_4^s)), sine(+(X_1^s, 0.8941))), sine(sine(e(X_2^s))))), \\
& *(+(sine(sine(+(X_4^s, X_1^s))), X_2^s), sine(X_5^s)))), \%(+(sine(sine(\\
& sine(e(+(-0.826, X_3^s))))), -(\%(e(-(-(X_3^s, -0.0548), \\
& sine(X_5^s))), e(\%(e(X_1^s), e(X_5^s)))), +(\%(*(e(X_3^s), +(X_4^s, X_3^s)), \\
& +(\%(X_2^s, X_1^s), -(X_2^s, X_2^s))), \%(*(sine(X_2^s), -(0.4106, X_6^s)), \\
& \%(sine(X_4^s), +(X_3^s, X_1^s)))))), -(+(-(+(+(+(X_4^s, 0.6661), e(X_6^s)), \\
& e(+(X_1^s, X_1^s))), e(-(sine(X_5^s), \%(X_1^s, X_3^s)))), *(+(-(e(X_1^s), \\
& -(X_4^s, X_1^s)), sine(-(X_2^s, X_1^s))), sine(-(e(X_6^s), +(X_5^s, X_5^s))))), \\
& sine(-(*(-(+(-0.2835, X_1^s), \%(-0.1978, X_2^s)), +(\%(X_6^s, X_1^s), \\
& \%(X_3^s, 0.8532))), e(-(sine(X_6^s), -(X_3^s, X_5^s)))))))) \\
X_4^c = {} & \%(\%(X_1^s, X_6^s), +(X_6^s, X_1^s)) \\
X_5^c = {} & -(\%(-(X_1^s, X_4^s), +(-(X_6^s, 0.7617), sine(X_4^s))), \\
& +(sine(+(X_5^s, X_1^s)), sine(e(X_6^s)))) \\
X_6^c = {} & +(+(sine(sine(X_6^s)), *(e(-0.37096), X_3^s)), -0.8782)
\end{aligned}$$

From this transformation, it can be noticed that the complexity of the constructed features differs from one to another. For example, the third feature has the most complex expression. In contrast, the fourth feature is represented by a simple transformation. However, this does not reflect the fitness quality of the constructed feature. When checking their distribution mismatch measures, the feature with less complex construction tree, X_4^c, has better mismatch performance than the complex one, X_3^c. Although this might not always true for all cases, it might indicate that the complicated transformations may not generalise well when mapping to new domains.

6.4.6.2 Genetic Operators

During our exploratory work, different genetic operators have been attempted before deciding the proposed probabilistic worst index mutation (PWIM) and the probabilistic best index crossover (PBIC). Here, we present these operators, then an empirical comparison between all operators is presented to show why the the mechanisms (PWIM, PBIC) are chosen.

a) Mutation:

1) Random index mutation (RIM): the mutation is performed on a randomly picked tree.

2) All index mutation (AIM): a standard subtree mutation is performed on every tree in the selected individual.

3) Best index mutation (BIM): the tree with the lowest mismatch value (i.e. the best distribution matching feature) is selected and a standard subtree mutation is performed. That is, the selected index idx satisfies the inequality $\Omega_{\mathcal{M}_{idx}} \leq \Omega_{\mathcal{M}_k}, \forall k \in \{1, ..., m^t\}$. If more than one tree has the best value, one of them is picked randomly. This operator encourages the change in good features.

4) Worst index mutation (WIM): the tree with the worst (i.e. highest) fitness value is selected and a standard subtree mutation is performed. In this case, the index idx is selected based on the constraint $\Omega_{\mathcal{M}_{idx}} \geq \Omega_{\mathcal{M}_k}, \forall k \in \{1, ..., m^t\}$. This approach encourages the change to happen for the poorly evolved trees hoping that the whole individual fits better when a wider range of target features are constructed well.

5) Probabilistic best index mutation (PBIM): the BIM is extended by encouraging the best trees to be considered more for mutation while allowing other trees to have a chance, but less, to be mutated as well. To do so, the probability of picking the idx^{th} tree for mutation is calculated as in Equation (6.17).

$$PBIM(\mathcal{M}, idx) = \frac{1 - \frac{\Omega_{\mathcal{M}_{idx}}}{\sum_{k=1}^{m^t} \Omega_{\mathcal{M}_k}}}{m^t - 1} = \frac{1 - \frac{\Omega_{\mathcal{M}_{idx}}}{m^t * \Omega_M}}{m^t - 1} \tag{6.17}$$

where $\mathcal{M}$ is an MTGP individual with m^t trees, $\mathcal{M}_k$ is the k^{th} tree in $\mathcal{M}$, and idx is the index of the tree to be selected.

b) Crossover:

1) Random same index tree crossover (RSIC): this operator is done by picking a random tree from one individual then applying standard subtree crossover with a tree of the same index in the other individual [139]. This approach increases the ability of each constructed feature to specialise while improving the performance of the whole set of constructed features when used together.

2) All-index crossover (AIC): the RSIC operator is extended by performing crossover between every pair of trees with the same index in both individuals. This implies more aggressive information exchange between individuals, which might provide more efficient training. However, it tends to produce offspring with different underlying trees regardless of the quality of the trees in the parents which may limit exploiting good trees during the evolutionary process.

3) Best index crossover (BIC): the pair of trees with the lowest average of mismatch (i.e. $\frac{\Omega_{\mathcal{M}_{idx}} + \Omega_{\mathcal{M}'_{idx}}}{2}$) is selected and a standard GP subtree crossover is performed between them. This operator encourages the exchange of information in good trees.

4) Worst index crossover (WIC): the pair of trees whose distribution similarity average is the worst are selected and a standard subtree crossover is performed between them. This operator encourages the exchange of information between poorly evolved trees.

5) Probabilistic worst same index crossover (PWIC): this operator is similar to WIC but it does not exclude the chance of good pairs to be mated. The probability of the idx^{th} pair to be considered for the crossover is calculated by Equation (6.18).

$$PWIC(\mathcal{M}, \mathcal{M}', idx) = \frac{1}{2}(\frac{\Omega_{\mathcal{M}_{idx}}}{m^t * \Omega_M} + \frac{\Omega_{\mathcal{M}'_{idx}}}{m^t * \Omega_{\mathcal{M}'}}), \qquad (6.18)$$

where $\mathcal{M}, \mathcal{M}'$ are MTGP individuals of m^t trees, $\mathcal{M}_k$ is the k^{th} tree in $\mathcal{M}$, and idx is the index of the tree to be selected.

As there are several operators for the genetic operators, an empirical comparison between them is carried out by comparing their effects on enhancing transfer learning on the data sets considering the homogeneous settings, i.e. 540 comparisons = 6 data sets $\times$ 30 incomplete copies $\times$ 3 missingness ratios. The number of positive transfer learning cases (significantly better than learning without transfer learning, i.e. WKNNIM) for each pair of GP operators settings is shown in Table 6.10.

Table 6.10: The number of positive transfer learning cases for each pair of GP operators settings.

	AIM	BIM	PBIM	PWIM	RIM	WIM	Sum
AIC	455	440	470	476	469	448	2758
BIC	425	414	440	453	450	423	2605
PBIC	483	472	489	506	504	476	2930
PWIC	373	362	376	402	386	365	2264
RIC	403	385	407	422	422	394	2433
WIC	341	331	350	374	364	341	2101
Sum	2480	2404	2532	2633	2595	2447	15091

Overall, although all operators tend to provide positive transfer learning, the best results are obtained by PBIC crossover and PWIM mutation. From the difference in the total numbers associated with each operator, the impact of changing the crossover operator is higher than of the mutation.

A potential reason is that the crossover rate is higher than the mutation rate.

For the operators that consider best trees, some features are constructed faster than others. This is because the best trees are the ones that will be affected by the GP operators. In contrast, for the operators on the worst trees, it is more likely for all features to change. However, the good ones do not show fast convergence along the whole process. This is because the unfit ones are more considered to apply the GP operators during the evolutionary process than the good trees. The probabilistic best operator is similar to the best case but there is a chance for the poorly constructed trees to be altered. Conversely, although the probabilistic worst mechanism prioritises the poor trees for GP operators, it still gives a chance for better trees to be considered.

6.5 Chapter Summary

In this chapter, a multi-tree GP-based method with new variation operators is developed for transfer learning in symbolic regression on incomplete data. This method constructs an asymmetric mapping from a complete source domain to an incomplete target domain while enhancing the estimation of the target missing values and reducing the distribution difference between the two domains. During the construction process, feature and instance weights as knowledge are extracted from the learned models in the source domain and then employed to benefit the learning in the target domain.

This chapter shows that GP-based transfer learning can be successfully utilised for symbolic regression with limited data with missing values. The evaluation of the proposed method is carried out by performing symbolic regression on incomplete data considering different scenarios. The experimental results not only show the ability of the proposed method to deliver positive transfer learning when compared with traditional methods, but also advances other transfer learning methods. This chapter provides empirical evidence of the ability of multi-tree GP in transferring useful knowledge between different (but related) domains. It also shows that the mechanism of the genetic operators can affect the whole evolutionary learning process.

One main finding of this chapter is that the probabilistic crossover and mutation operators have remarkable contributions to the convergence of the evolutionary process. In particular, the crossover operator prefers the best trees while giving some possibility for other trees to be mated tends to provide better results than other crossover operators, while the best results are obtained when the worst trees have more possibility to be mutated. This means that best trees are encouraged to participate in producing the new individuals more than others. On the other hands, the worst trees are given a high probability to be altered in hoping for a fitter ones. The detailed analysis shows that such operators have an impact on special kinds of exploration and exploitation that work locally at the level of trees in individuals.

However, the proposed method is only suitable for domains with a small number of features as the number of trees in each individual becomes large when dealing with a large number of features, which complicates the evolutionary process. Next chapter will provide an attempt to address this limitation.

Chapter 7

Multi-Tree GP with Feature-based Transfer Learning for Symbolic Regression on Incomplete Data

7.1 Introduction

In Chapter 6, we presented a transfer learning method based on multi-tree GP (MTGP) to transfer knowledge from complete source domains to improve symbolic regression in incomplete target domains. The transfer learning performance is improved by minimising the regression error and domain distribution mismatch when constructing the feature transformations. Although the results in the previous chapter are encouraging, a major limitation is that each MTGP individual contains as many trees as the number of target domain features. Consequently, for target domains with a large number of features, these individuals will be super complex. This chapter shows how this limitation can be addressed by integrating feature selection with MTGP-based transfer learning in symbolic regression with missing values.

7.1.1 Chapter Goals

The main goal of this chapter is to develop a novel MTGP-based transfer learning method that utilises feature selection in a multi-stage algorithm for symbolic regression on incomplete data. Specifically, the objectives of this chapter include:

1. Proposing a new feature selection mechanism that utilises knowledge about features extracted from the source domain (SD) to select a useful subset of features in the target domain (TD) based on the learning processes in the two domains. The weights of the SD features are estimated during the symbolic regression process on the SD, and this knowledge is used to calculate weights for the features transformed to the TD. The weights of the transformed features and the weights of the corresponding target features are combined to decide which features should be selected in the rest of the transfer learning process. This new feature selection mechanism helps utilising the feature knowledge from the two domains to improve the symbolic regression performance on TD.

2. Developing an adaptive MTGP-based transfer learning method that integrates the proposed feature selection mechanism to construct robust transformations from complete SDs to incomplete TDs. In this method, a multi-stage evolutionary process is performed with updated settings. Each stage consists of a set of generations starting from the last generation of the previous stage, where the individuals are modified by eliminating trees according to the selected features. This strategy enables each stage to benefit from the learning outcome of the previous stage, which accelerates the convergence of the evolutionary process while producing effective symbolic regression models.

3. Investigating the proposed adaptive MTGP-based transfer learning

method to symbolic regression with incomplete data considering different scenarios. The experimental work includes synthetic incompleteness on real-world data sets and synthetic data sets, and real-world incomplete data sets as well. These experiments simulate two transfer learning scenarios: homogeneous domains and heterogeneous domains. Furthermore, to analyse the impact of the proposed feature selection strategy, random irrelevant features are added to some data sets. The effectiveness and the efficiency of the proposed method have been shown by the improved symbolic regression error and the time for transfer learning.

7.1.2 Chapter Organisation

This chapter is organised as follows. The proposed method is presented in Section 7.2. The experiment design is setup in Section 7.3. After that, the experimental analysis is given in Section 7.4. Finally, Section 7.5 states the conclusions of this chapter.

7.2 The Proposed Method

7.2.1 Feature-based Multi-tree GP and TL

In Chapter 6, we have presented preliminary works on multi-tree GP (MTGP) to transfer knowledge from complete SDs to improve symbolic regression in incomplete TDs. Such transformations are constructed through an evolutionary process in which each individual consists of a set of trees each representing a constructed feature in the TD. Each tree constructs a TD feature by transforming the SD features. The constructed features are combined with the TD features to help imputing the missing values in TD. The imputed TD can then be used for symbolic regression in a normal way as it is now complete. However, this representation does not consider the importance of the TD features to be constructed, which may lead to redundant or irrelevant features being constructed. Consequently, in case there are many such features, the construction of their transformations might be an unnecessarily expensive process.

The main idea of the new approach is to integrate feature selection in order to adaptively reduce the number of trees in the MTGP-based feature transformations. The evolutionary process of MTGP-based feature transformations is performed in multiple stages. In each stage, a number of generations are evolved starting from the final generation of the previous stage. By the end of each stage, the TD symbolic regression process is used to estimate weights for the TD features, which, along with the constructed features, are used to select the features that should be considered in the next stage. The other features are ignored and their corresponding trees are eliminated from each MTGP individual in the population of the last generation of the current stage. This modified population is used as the initial (first) population in the next stage.

Figure. 7.1 shows the overall flowchart of the proposed method. In the complete SD, knowledge is extracted based on symbolic regression process. The knowledge consists of selected SD data and weights for the SD features based on the learned symbolic regression models (details in Subsection 7.2.2). The extracted SD knowledge is used to evolve MTGP-based feature transformations from SD to TD. The construction of the feature transformations involves different processes including: transforming SD data, imputing TD missingness, matching the distributions of the transformed SD and the imputed TD, performing regression in the imputed TD, and combining the TD regression performance metric with the SD-TD distribution mismatch measure in a fitness function to evaluate the evolved transformations (details in Subsection 7.2.3).

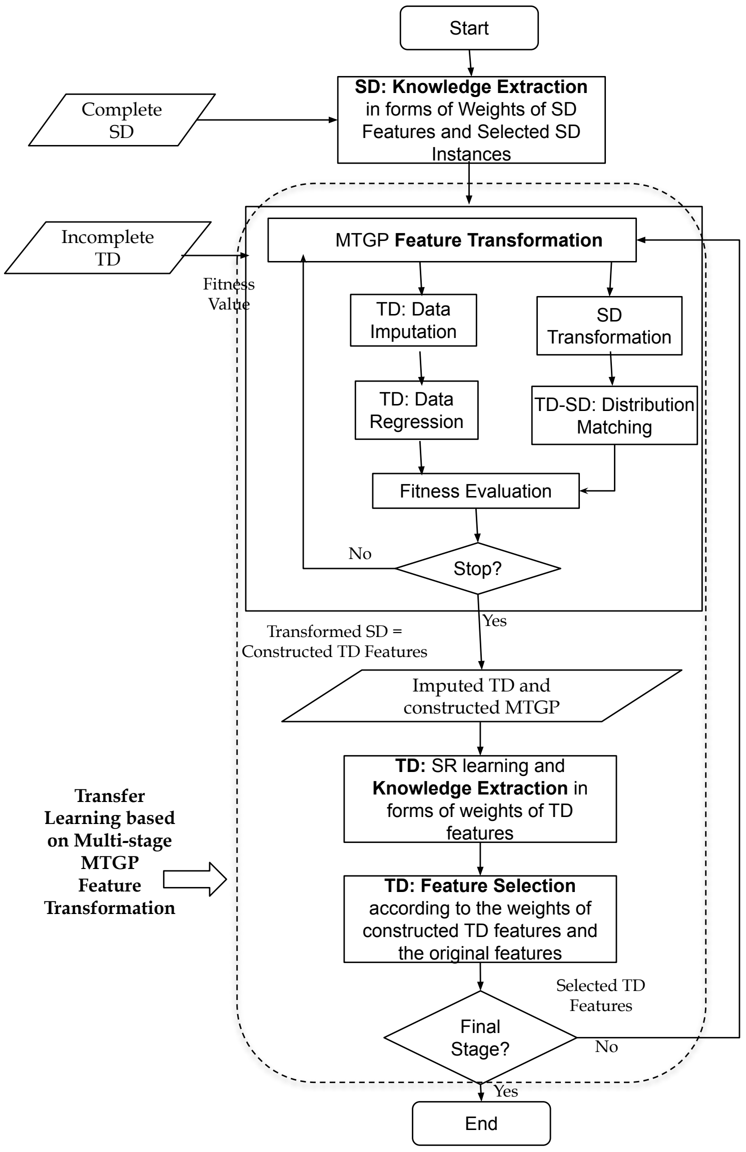

Start
Complete SD
SD: Knowledge Extraction in forms of Weights of SD Features and Selected SD Instances
Incomplete TD
MTGP Feature Transformation
Fitness Value
TD: Data Imputation
SD Transformation
TD: Data Regression
TD-SD: Distribution Matching
Fitness Evaluation
No
Stop?
Yes
Transformed SD = Constructed TD Features
Imputed TD and constructed MTGP
TD: SR learning and Knowledge Extraction in forms of weights of TD features
Transfer Learning based on Multi-stage MTGP Feature Transformation
TD: Feature Selection according to the weights of constructed TD features and the original features
Selected TD Features
Final Stage?
No
Yes
End

Figure 7.1: The main steps of the proposed multi-stage transfer learning method.

The constructed MTGP-based feature transformations aim at constructing a set of selected TD features in multiple stages. For the first stage, the set of selected TD features is initialised to be the whole set of the original TD features. In each of the other stages, after constructing feature transformations, the construction outcomes are used to extract knowledge from both the transformed SD and the imputed TD. For the transformed SD, weights for the constructed features are calculated (details in Subsection 7.2.4). The missing values in the TD are estimated during the feature transformation process, which produces an imputed complete TD. Symbolic regression is then performed in the imputed TD and weights for the selected TD features are extracted based on the learned symbolic regression (details in Subsection 7.2.5). The knowledge of the weights for the target and constructed features are then combined to filter the set of selected TD features, which are used to refine the multi-tree representation, i.e. redefine the number of trees according to the number of selected features (details in Subsection 7.2.6). If the maximum number of stages has not been reached, the refined transformations continue to be evolved to construct a new set of selected TD features by repeating the previous processing operations starting from building the feature transformations. Once the whole learning process stops, the constructed transformations and the learned symbolic regression models are then applied to unseen test data in the TD (details in Subsection 7.2.7).

In the following subsections, the details of the proposed methods are presented. For clarity, the notations used in this work are given in Table 7.1, where superscripts of $t, s,$ and c are used to refer to target, source, and constructed domains, respectively. For example, the symbols $\mathcal{D}^s$, $\mathcal{D}^t$, and $\mathcal{D}^c$ denote the source domain, the target domain, and the constructed domain, respectively.

7.2.2 Knowledge Extraction from Learning in the SD

To extract knowledge from the SD, R symbolic regression models are learned using R independent GP runs. These models, denoted as $\{f_r^s\}_{r=1}^R$, are used to estimate the weights of the SD space features. The features are weighted

Table 7.1: Summary of the used notations

Symbol	Meaning
$\mathcal{T}$	task
$\mathcal{D}$	domain
$\mathcal{X}$	feature space
$\mathcal{L}$	loss function
Ψ	MTGP individual
$\mathcal{P}$	MTGP population
Y	desired value
$\hat{Y}$	predicted output
m	number of features
n	number of instances
R	number of symbolic regression models
f	symbolic regression learned model
$\mathcal{W}_{X_j}$	weight of the j^{th} feature
$\mathcal{F}^t$	selected TD features

based on their contributions to the learned symbolic regression models as in Equation (7.1). The contribution of the feature is measured according to its participation in the symbolic regression models, i.e. the best models generated in GP runs for R times. The features that participate more in these symbolic regression models have more contributions. The large contribution that a feature has, the higher weight that it will get.

$$\mathcal{W}_{X_j^s} = \frac{1}{R}\sum_{r=1}^{R}\frac{frq(X_j^s, f_r^s)}{\sum_{p=1}^{m^s} frq(X_p^s, f_r^s)} \tag{7.1}$$

$\mathcal{W}_{X_j^s}$ is the weight of the j^{th} SD feature, X_j^s, according to the SD learning process, and $frq(X_j^s, f_r^s)$ is the number of appearances of the j^{th} feature in the r^{th} learned GP model, f_r^s, where m^s is the number of SD features. On the other hand, the SD instances in $\mathcal{D}^s$ are sorted based on their mean regression errors of $\{f_r^s\}_{r=1}^R$. Then the best two-thirds instances (i.e. the 66.7% least average errors) are selected to form the selected data $\overset{*}{\mathcal{D}}{}^s$. $\overset{*}{\mathcal{D}}{}^s$ and the weights, $\mathcal{W}_{X_j^s}$, of the SD features are used as the extracted TD knowledge in the proposed algorithm.

7.2.3 MTGP-based Feature Transformation

For the selected data from the SD, $\overset{*}{\mathcal{D}}^s$, with the feature space of m^s features, $\mathcal{X}^s = \{X_j^s\}_{j=1}^{m^s}$, and TD, $\mathcal{D}^t$, with the feature space of m^t features, $\mathcal{X}^t = \{X_j^t\}_{j=1}^{m^t}$, multi-tree GP (MTGP) is utilised to build a feature transformation from the SD to the TD as in shown in Algorithm 9.

Algorithm 9: The MTGP-based transformation.

```
Input  : Selected SD data, $\overset{*}{\mathcal{D}}^s$, Incomplete TD training data, $\mathcal{D}^t$, Initial
         population of individuals, $\mathcal{P}^{(in)}$, A set of selected TD features, $\mathcal{F}^t$,
         Maximum number of generations $genMax$.
Output: Imputed TD training data, $\hat{\mathcal{D}}^t$, Constructed feature transformation, $\Psi$,
         An evolved population, $\mathcal{P}^{(out)}$.
1  Transform the TD data, $\mathcal{D}^t$, to contain only the selected TD features, $\mathcal{F}^t$;
2  $gen = 0$;
3  $\mathcal{P}^{(gen)} = \mathcal{P}^{(in)}$;
4  while $gen < genMax$ do
5      foreach individual $\Psi$ in the population $\mathcal{P}^{(gen)}$ do
6          Apply $\Psi$ to $\overset{*}{\mathcal{D}}^s$ getting a constructed data $\mathcal{D}^c$;
7          Combine $\mathcal{D}^t$ and $\mathcal{D}^c$ to impute $\mathcal{D}^t$ using WKNN to produce imputed TD
             $\hat{\mathcal{D}}^t$;
8          Evaluate the fitness of $\Psi$ using Equation 7.2 (Subsection 7.2.3.3);
9      end
10     Select individuals from the current generation;
11     Produce the next generation, $\mathcal{P}^{(gen+1)}$, using genetic operators (Subsection
         7.2.3.4);
12     $gen = gen + 1$;
13 end
14 Get the population of the last generation $\mathcal{P}^{(out)} = \mathcal{P}^{(gen)}$;
15 Obtain the best individual in the last generation to be the constructed feature
     transformation, $\Psi$;
16 Return the imputed complete TD training data, $\hat{\mathcal{D}}^t$, the constructed feature
     transformation, $\Psi$, and the evolved population in the last generation, $\mathcal{P}^{(out)}$;
```

The algorithm starts with a given initial population of MTGP individuals, $\mathcal{P}^{(in)}$, that comes from the previous stage, except for the first stage where this population is initialised randomly with the given population size. The population is then evolved to construct new SD features that are similar to a selected set of TD features, $\mathcal{F}^t$. The selection of $\mathcal{F}^t$ is based on the outcomes of the transformation in the first stage (as will be shown in 7.2.6), whereas the set consists of all TD features in the first stage. For the given population of individuals, $\mathcal{P}^{(in)}$, each individual is applied to construct a transformed SD and then evaluated by a fitness function that measures the similarity between the transformed SD and the TD. A selection process is then performed and a new generation is generated using genetic operators (mutation, crossover, and elitism). This process is repeated $genMax$ times and the best individual of the last generation is then returned as the constructed feature transformation.

7.2.3.1 Representation

Each MTGP individual is represented by multiple trees, where the SD feature set is used as input for these trees. Each tree is meant to construct a feature similar to a feature selected from the TD. Each MTGP individual is called an MTGP-based feature transformation from the SD to the TD. The application of each MTGP-based feature transformation to the SD produces the transformed SD $\mathcal{D}^c$ with m^c features, where each tree represents a constructed feature (corresponding to one TD selected feature).

The set of features selected from the TD is denoted as $\mathcal{F}^t$. MTGP tries to use one tree to construct every feature in $\mathcal{F}^t$ based on the SD features. In the first stage of the multiple stages shown in Figure. 7.1, all the TD original features are constructed, i.e. no feature selection is applied, which means $\mathcal{F}^t = \mathcal{X}^t$ and $m^c = m^t$. However, later, when feature selection is applied, $\mathcal{F}^t \subset \mathcal{X}^t$ and $m^c < m^t$.

An illustration for the MTGP-based feature transformation is shown in Figure. 7.2. Although F^t consists of the selected TD features, it can refer to a similar set of constructed features as those features are build to construct the selected TD features set, F^t. As will be seen later, this selection process is based on a feedback from the symbolic regression process in the TD in addition to the learning outcomes of the symbolic regression process in the SD.

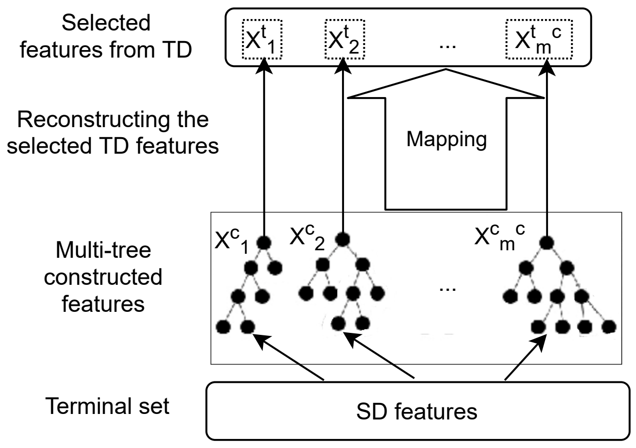

Selected features from TD
X^t_1
X^t_2
...
$X^t_{m^c}$
Reconstructing the selected TD features
Mapping
Multi-tree constructed features
X^c_1
X^c_2
$X^c_{m^c}$
...
Terminal set
SD features

Figure 7.2: MTGP-based feature transformation, where the SD features are used in the terminal set of multi-tree GP individuals that are evolved such that each tree constructs a feature similar (in distribution) to a feature selected from the TD.

7.2.3.2 Data Imputation

Imputation is performed on the TD by utilising the transformed/constructed SD, $\mathcal{D}^c$. For this purpose, the imputation method weighted k-nearest neighbour (WKNN), which was used in earlier chapters, is used. $\mathcal{D}^c$ is combined with the incomplete TD $\mathcal{D}^t$ then WKNN is used to estimate the missing values in $\mathcal{D}^t$, which results in an imputed TD, $\hat{\mathcal{D}}^t$.

WKNN works by finding k nearest instances for each incomplete instance to be imputed. These instances are distance-based neighbours and they are called imputation donors. The mean (mode) of the values of these neighbours are then used to replace the corresponding values in the numerical (categorical) features. WKNN is chosen as it can easily utilise the knowledge from the SD. It implicitly weights the instances when imputing the missing values. Its weighting is based on the distance between the incomplete instances and the complete instances.

7.2.3.3 Fitness Function of MTGP

Similar to Chapter 6, the quality of the MTGP-based feature transformations is measured based on a regularised fitness function that consists of two parts as in Equation (7.2).

$$fitness = \mathcal{L}(\mathcal{T}^t(\hat{\mathcal{D}}^t)) + \lambda * \Omega(\mathcal{D}^t, \mathcal{D}^c) \tag{7.2}$$

where the parameter λ is used to balance the learning loss, $\mathcal{L}$, and the distribution mismatch, Ω. The details of these two parts are given below.

This fitness function is designed to achieve two objectives at the same time. The first objective is to minimise the learning loss on the imputed TD data obtained by the feature transformation. The second objective is to minimise the distribution difference between the transformed SD and the TD. Such considerations are intended to guide the MTGP evolutionary process to produce individuals that can construct data from SD that can benefit the limited incomplete TD. The more similar the constructed SD to the TD, the more likely to compensate for the incompleteness in the TD.

The regularisation factor λ is used to balance the inter-dominance between the two measures $\mathcal{L}$ and Ω. λ is set empirically to 0.3, which gives more importance to the learning task part than the domain distribution part. Although the parameter λ could be more tuned, we used the same settings as the benchmark method in Chapter 6.

The first part of Equation (7.2), $\mathcal{L}(\mathcal{T}^t(\hat{\mathcal{D}}^t))$, measures the impact of the transformation on the symbolic regression learning process in the TD. It is achieved by a loss function, $\mathcal{L}$, of the TD task, $\mathcal{T}^t$. As the original TD is incomplete, the learning process is performed on the imputed TD, $\hat{\mathcal{D}}^t$, produced by WKNN.

The learning loss $\mathcal{L}$ is calculated based on the regression performance on the imputed TD $\hat{\mathcal{D}}^t$ measured using the relative squared error (RSE) shown in Equation (7.3).

$$RSE(Y, \hat{Y}) = \frac{\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2}{\sum_{i=1}^{n}(Y_i - \bar{Y})^2} \tag{7.3}$$

where n is the number of the instances, $\hat{Y}_i$ is the i^{th} predicted value, Y_i is the i^{th} desired value, and $\bar{Y}$ is the average of the desired values.

The second part of Equation (7.2) is the distribution mismatch, Ω, which measures the difference between the distribution of the constructed domain, $\mathcal{D}^c$, and the TD, $\mathcal{D}^t$. The distribution mismatch metric, Ω, is based on the Kolmogorov-Smirnov statistic, which measures the difference between the empirical cumulative distribution function of two samples [78]. Ω is calculated as in Equation (7.4).

$$\Omega(\mathcal{D}^c, \mathcal{D}^t) = \frac{\sum_{i=1}^{m^t} KS(X_i^c, X_i^t)}{m^t}, \tag{7.4}$$

where X_i^t and X_i^c, is the i^{th} feature of the domain $\mathcal{D}^t$ and $\mathcal{D}^c$ respectively, and Kolmogorov-Smirnov is calculated as in Equation (7.5).

$$KS(X_i^c, X_i^t) = \sup_x |F_{X_i^c}(x) - F_{X_i^t}(x)|, \tag{7.5}$$

where F is the cumulative distribution function and $\sup$ is the supremum function.

Kolmogorov-Smirnov has some important advantages [78]. For example, it is distribution free. This implies that it is applicable to samples from unknown distributions. Another advantage is its flexibility regarding the sample size, which makes it useful even on small samples. Moreover, Kolmogorov-Smirnov can be used for regression analysis as a measure of goodness of fit [153]. Kolmogorov-Smirnov provides the ability to measure feature-wise distribution dissimilarity which is of special importance in this work. In transfer learning, Kolmogorov-Smirnov has been used to examine the distribution similarity between the source and target domains in [47, 94]. In [165], Kolmogorov-Smirnov is used for heterogeneous defect prediction based on feature matching and it is used for integrating structured biological data in [32].

7.2.3.4 Genetic Operators

As each MTGP individual consists of multiple trees, the standard genetic operators are extended to multi-tree operators. Unlike Chapter 6, where probabilistic operators are used, in this chapter genetic operators are done by applying the standard genetic operations to single trees selected from MTGP individuals as follows:

- Mutation: In the standard GP single-tree mutation, a randomly picked subtree of the selected individual is changed. This mechanism is extended by applying it to a randomly selected tree in the MTGP individual.

- Crossover: Similar to the mutation operator, the standard GP crossover is extended to MTGP. In this work, we use the same-index crossover (SIC) strategy [138], where the crossover is performed on trees at the same index (randomly picked) in each individual. This mechanism encourages individual trees to specialise while interacting with different individuals through the crossover operation.

7.2.4 Constructed Feature Weighting

As the constructed features are GP trees representing symbolic expressions in terms of the SD features (terminal set), the weights of the constructed features can be calculated based on the weights of the SD features. Let us define the score of each constructed feature, X^c_j, using the weights of the SD features that are involved in its mathematical expression. That is, the score of the constructed feature X^c_j is the sum of all the weights of its contributing features (i.e. SD features included in the corresponding tree) as shown in Equation (7.6).

$$score(X_j^c) = \sum_{\forall X_p^s \text{ used to construct } X_j^c} \mathcal{W}_{X_p^s} \tag{7.6}$$

These scores are then normalised to get the weights of the constructed features as shown in Equation (7.7). The normalisation is done to make values between 0 and 1, which is the same range of the other weights extracted for features TD. This guarantees more fair combinations between weights for constructed features and weights for TD features later.

$$\mathcal{W}_{X_j^c} = \frac{score(X_j^c)}{\sum_{p=1}^{m^t} score(X_p^c)} \tag{7.7}$$

7.2.5 TD Features Weighting

Similar to the feature weighting process performed in the SD (Subsection 7.2.2), the TD features are weighted. Symbolic regression learning process is performed R independent times on the imputed TD training data, $\hat{\mathcal{D}}^t$, and the weights of the TD features are calculated using the same way as shown in Equation (7.1). The outcome of this step is $\mathcal{W}^t = \{\mathcal{W}_{X_j^t}, X_j^t \in \mathcal{F}^t\}$.

7.2.6 Population Refinement based on the Selected Features

Once one stage finishes (as shown in Figure. 7.1), a new stage starts from the refined final population of the previous stage. The main aim of this refinement is to remove the trees that are unlikely to be useful from each individual before starting the new stage. These excluded trees are determined based on the set of selected TD features, $\mathcal{F}^t$.

At the end of each stage, new scores for the selected TD features, $\mathcal{F}^t$, are calculated based on the weights of the TD features, $\mathcal{W}^t$, and the weights of the constructed features, $\mathcal{W}^c$. For each feature $X_j^t \in \mathcal{F}^t$, let $\mathcal{W}_{X_j^t}$ be its weight and $\mathcal{W}_{X_j^c}$ is the weight of the corresponding constructed feature, X_j^c, then the selection score of the j^{th} feature is given in Equation (7.8), .

$$\omega_j = \frac{(\mathcal{W}_{X_j^c} + \mathcal{W}_{X_j^t})}{2} \tag{7.8}$$

The set of selected TD features, $\mathcal{F}^t$, is then filtered based on their selection scores such that any feature whose score is less than 10% of the highest score is excluded. This threshold is chosen empirically.

After updating the selected TD feature set, $\mathcal{F}^t$, the constructed MTGP-based feature transformations in this stage are refined by filtering the trees that construct the un-selected features from each individual. Only trees for the selected TD features will survive in the next stage.

7.2.7 The Testing Phase

Algorithm 10: The Testing Process.

```
Input  : Transformed complete TD training data, $\hat{\mathcal{D}}^t$, set of selected TD
         features, $\mathcal{F}^t$, learned symbolic regression model, $f$, and (in)complete
         test instance, $x^t_{test}$.
Output : Prediction output for $x^t_{test}$.
1 Transform $x^t_{test}$ based on $\mathcal{F}^t$ to get an instance with selected features, $\bar{x}^t_{test}$;
2 if $\bar{x}^t_{test}$ is complete then
3     Apply the symbolic regression model $f$ to predict $\bar{x}^t_{test}$ ($\hat{y}^t_{test} = f(\bar{x}^t_{test})$);
4 else
5     Impute $\bar{x}^t_{test}$ to get a transformed complete instance $\hat{x}^t_{test}$ using WKNN with
        $\hat{\mathcal{D}}^t$;
6     Apply the symbolic regression model $f$ to predict $\hat{x}^t_{test}$ ($\hat{y}^t_{test} = f(\hat{x}^t_{test})$);
7 end
8 Return $\hat{y}^t_{test}$;
```

For a new (in)complete test instance, x_{test}, the outcomes of the aforementioned learning process are applied to it, i.e. the transformed complete TD training data, $\hat{\mathcal{D}}^t$, the selected TD feature set, $\mathcal{F}^t$, and the learned symbolic regression model, f, are employed to predict the regression output of the $\hat{y}_{test}$ as shown in Algorithm 10.

7.3 Design of Experiments

7.3.1 Data Sets

Three types of data sets are used for experimental evaluation: four real-world complete data sets, two real-world incomplete data sets, and six artificial data sets. Such a combination allows evaluating the applicability of the proposed method on both real and synthetic incompleteness. Moreover, these data sets have various characteristics such as different numbers of features and instances, which help examining the validity of the proposed method under different circumstances.

The real-world data sets are frequently used in the literature for investigating transfer learning regression [272, 175, 46]. More details on these data sets can be found in the UCI machine learning repository [62]. In this work, only the numerical features are involved as the impact of data types is beyond the scope of this study. For the two incomplete data sets Auto and Imports, the instances with real missing values are holdout during the process of constructing the feature transformations then they are used later for evaluating the applicability of the proposed method to real-world incompleteness. Note that, although the data considered are complete, synthetic incompleteness is imposed to all data sets before performing the feature transformation construction process. This means that, for the Auto and Imports data sets, the construction of the transformations is done on synthetic incompleteness on the training TD data and the real incomplete instances are kept for testing the learned models. This approach examines the applicability of the proposed method in the extreme situation of having unknown missingness in the unseen data.

Following previous research on regression [10], synthetic data sets based on Friedman function [85] are generated, which include both linear and non-linear relationships between the independent variables and the dependent variable. In addition to the five input features, additional independent randomly generated features are added into the data sets to measure the ability of the feature selection ability. The synthetic data sets are considered to get more controlled experiments, especially when varying the number of irrelevant features. A normalised noise, ϵ, is also added as shown in Equation (7.9). In addition to the five input features (Fri1), additional (5, 15, 35, 75, and 155) independently randomly generated features are added to form the data sets Fri2, Fri3, Fri4, Fri5, and Fri6, respectively. This strategy allows measuring the impact of non-relevant features. Unlike Chapter 5, where the data sets are obtained from the OpenMl repository [241], in this chapter the data sets are generated using the given function.

$$y = 10 * sin(\pi * x_1 * x_2) + 20 * (x_3 - 0.5)^2 + 10 * x_4 + 5 * x_5 + \epsilon \tag{7.9}$$

For simulating the transfer learning scenarios, the data sets are prepared following practices used in the related work [272, 175, 46, 47, 263, 264]. To form SD and TD with different distributions, each data set is sorted according to a certain continuous feature, then the sorted data set is split based on a specific value of this feature such that the first two-thirds of the data composite the SD while the TD consists of the last third of the data. The details of the formed data sets are shown in Table 7.2. m refers to the number of features in the original data set, n^s (n^t) is the number of instances in the SD (TD) after data split, and the split feature and value used are shown for each data set. For simulating the heterogeneous transfer learning scenario, the feature space in the SD is made different than the TD following [263, 264]. The SD contains only the first half of the features whereas all the available features are included in the TD.

To form heterogeneous source-target domains, a feature-wise split is used to ensure that the feature space of the TD is different from the SD feature space [263, 264]. Similar to [263, 264], the first half of the features

Table 7.2: Statistics of the used data sets

Data set	m	n^s	n^t	Split feature	Split value
Hou: Housing	13	369	137	TAX	600
Con: Concrete	9	687	343	fine	803.7
Yac: Yacht	6	198	110	Froude	0.35
For: Forestfires	13	347	170	DC	694.8
Aut: Auto-mpg	6	258	134	acceleration	16.5
Imp: Imports	14	125	70	length	176.6
Fri1: Friedman1	5	350	150	X.oz1	0.698464513
Fri2: Friedman2	10	350	150	X.oz1	0.698464513
Fri3: Friedman3	20	350	150	X.oz1	0.698464513
Fri4: Friedman4	40	350	150	X.oz1	0.698464513
Fri5: Friedman5	80	350	150	X.oz1	0.698464513
Fri6: Friedman6	160	350	150	X.oz1	0.698464513

form the SD (i.e. the first $\lceil n^t/2 \rceil$ most occurring features) while all features are used to form the TD.

Furthermore, for simulating data incompleteness, each TD data set is made incomplete synthetically by imposing 30% MAR missingness into 40% of the features. This step is repeated 30 times for each TD data set to get 30 incomplete data sets. The MAR mechanism is considered as it is suitable to evaluate the imputation methods [270].

7.3.2 Benchmark Methods and Settings

The benchmark methods in this work are the same as in Chapter 6. These methods are WKNN, WKNNIMDC, RFIM, RFIMDC, MTGPTL, and MT-GPDA. These methods are compared with the proposed method, **FMTG-PTL** with five stages, each has 10 generations, i.e. 50 generations in total. Note that, only the TD is considered in WKNNIM and RFIM, i.e. no transfer learning. On the other hand, although WKNNIMDC and RFIMDC combine both domains, these methods involve no transfer learning as the combination is done on the data level without transfer learning.

To compare the seven different methods, the symbolic regression performance on the imputed data sets is considered as an evaluation metric. It is measured using the RSE metric (Equation (7.3)). Since the considered methods are stochastic, 30 independent runs are executed for each method on each data set. After splitting the original data into 2:1 as the data for the SD and the TD, the TD is then randomly split into 3:7 training-test data sets. This split is common in transfer learning research to emphasise the shortage of data available for learning in the TD [46]. To examine the statistical significance in the difference between the results obtained by different methods, a Wilcoxon non-parametric statistical test with a significance level of 0.05 is used. For the GP-based methods, the DEAP python package [83] with the settings shown in Table 7.3 are used. The package missingpy is used for the imputation methods (WKNN and RFIM) with default settings and the Kolmogorov-Smirnov statistic is based on the SciPy package.

Table 7.3: MTGP parameters Settings.

Parameter	value
Generations	50
Population size	512
Crossover rate	0.8
Mutation rate	0.2
Elitism	5
Selection	Tournament-7
Maximum depth	8
Initialisation	Ramped-half-and-half
Function set	+, -, *, protected %, sine, exp
Terminal set	features and constants $\in (-1, 1)$

7.4 Results and Discussions

This section presents the experimental results along with analyse and discussions. It includes evaluating the SR performance on the data sets and the feature selection impact. For the SR results, the SR errors are first shown then statistical test results on these errors are presented. The SR results are given in Tables showing the average error of 30*30 runs for each method on each data set. Moreover, Figures are given to show the significance test results for each method on the 30 generated data sets corresponding to the 30 comparisons.

7.4.1 Symbolic Regression on Real-world Data with Synthetic Incompleteness in Homogeneous Domains

In homogeneous transfer learning, the feature spaces of both SD and TD are the same but their corresponding marginal distributions are different, i.e. $\mathcal{X}^s = \mathcal{X}^t$ but $P(\mathcal{X}^s) \neq P(\mathcal{X}^t)$. This subsection shows the results of the proposed method compared with other methods on the scenario of homogeneous transfer learning.

7.4.1.1 Symbolic Regression Errors on Homogeneous Domains

Table 7.4 shows the symbolic regression results when using the proposed FMTGPTL method and the benchmark methods.

Across all the different methods, the results obtained by the proposed method are associated with the lowest regression errors. The reduction of the regression error that FMTGPTL achieves differs based on the data set. To make the comparison clearer and easier, we use the relative reduction rate given as *percentage relative change* (PRC) in Equation (7.10) [112, 108]. This equation measures the percentage of regression error that is reduced by the proposed method over each benchmark method (rounded). The

Table 7.4: The symbolic regression results in terms of RSE in **homogeneous** domains with synthetic incompleteness.

Method		Hou	Con	Yac	For	Aut	Imp	Fri1	Fri2	Fri3	Fri4	Fri5	Fri6
WKNNIM	Mean	1.0306	1.2634	1.1526	1.0552	0.5417	0.6794	0.8458	0.8493	0.8689	0.8958	0.9187	0.9644
	Stdv	0.1704	0.3052	0.3345	0.0432	0.0753	0.1298	0.2111	0.3265	0.2833	0.1881	0.2067	0.2284
	PRC	15%	12%	13%	13%	5%	13%	4%	6%	6%	9%	10%	13%
WKNNIMDC	Mean	1.127	1.298	1.2178	1.0897	0.5775	0.7098	0.9147	0.9209	0.9397	0.9497	0.9555	0.9984
	Stdv	0.1699	0.2892	0.3345	0.0595	0.0919	0.1184	0.2927	0.297	0.3222	0.2851	0.2795	0.2157
	PRC	22%	15%	18%	15%	11%	16%	12%	13%	13%	14%	13%	16%
RFIM	Mean	0.9542	1.2439	1.1355	1.052	0.548	0.6461	0.8219	0.8462	0.8676	0.8779	0.887	0.9118
	Stdv	0.1378	0.2879	0.3116	0.0503	0.08	0.1162	0.2241	0.2846	0.2863	0.2458	0.2818	0.3296
	PRC	8%	11%	12%	12%	6%	8%	2%	5%	6%	7%	6%	8%
RFIMDC	Mean	1.2357	1.3293	1.2869	1.1099	0.5891	0.7377	0.9264	0.9652	0.9575	0.9866	1.0086	1.1065
	Stdv	0.2157	0.3016	0.3782	0.0692	0.083	0.1183	0.315	0.3284	0.3743	0.3597	0.3163	0.2317
	PRC	29%	17%	22%	17%	13%	20%	13%	17%	15%	17%	18%	24%
MTGPTL	Mean	0.9741	1.2463	1.1093	0.986	0.5404	0.6365	0.8218	0.8299	0.8491	0.8589	0.8807	0.9176
	Stdv	0.1133	0.2314	0.2639	0.034	0.076	0.1016	0.2694	0.216	0.2768	0.3055	0.3139	0.2948
	PRC	10%	11%	10%	7%	5%	7%	2%	3%	4%	5%	6%	9%
MTGPDA	Mean	0.9182	1.214	1.0425	0.9656	0.5261	0.6325	0.8092	0.8216	0.8307	0.8389	0.8655	0.8985
	Stdv	0.1002	0.2263	0.2645	0.0285	0.082	0.0991	0.2322	0.2193	0.278	0.2058	0.2167	0.3136
	PRC	4%	9%	4%	5%	2%	6%	0%	2%	2%	3%	4%	7%
FMTGPTL	Mean	0.8799	1.1072	0.9999	0.9219	0.5152	0.5938	0.8086	0.8015	0.8173	0.8155	0.8299	0.8387
	Stdv	0.2581	0.3195	0.2762	0.2656	0.2038	0.166	0.3161	0.1561	0.2864	0.2188	0.1726	0.2635

higher the PRC, the better improvement and getting a negative value indicates the proposed method is getting worse regression.

$$PRC(RSE_p, RSE_b) = Round(\frac{RSE_b - RSE_p}{RSE_b} * 100) \tag{7.10}$$

where RSE_p is the regression error achieved by the proposed method and RSE_b is the regression error achieved by a benchmark method.

The proposed FMTGPTL method achieves a positive relative change, PRC, on the test RSEs of all the used data sets, except for the Fri data sets against MTGPDA. The improvement varies along with the benchmark method and the data set. For example, the error of MTGPDA is barely reduced on the Fri1 data set, while the highest improvements achieved by FMTGPTL are observed when compared with the RFIMDC on all the data sets. This is because RFIMDC predicts the missing values of incomplete features using feature-based prediction on data from different domains without adaptation.

With the increase of the dimensionality of the synthetic data sets, a dramatic increase in the error is noticed. This change is the highest in the case of having 160 features and the difference between the mean of RSEs obtained when using FMTGPTL ranges from 7% to 24% compared with the old error in all cases on this data set. This is due to the important advantage that FMTGPTL brings, which is the inclusion of feature selection. As the underlying imputation method is WKNN, feature selection can be very beneficial. WKNN is based on a feature-wise similarity between instances, which makes it sensitive to irrelevant and noisy features. This is clear when having several random features. The more irrelevant features the more improvement gained. For example, the FMTGPTL method on Fri6 considerably advances the performance of WKNN.

7.4.1.2 The Significance Test Results on Homogeneous Domains

Figure. 7.3 shows the statistical significance test results on the differences between the symbolic regression errors of FMTGPTL and the benchmark methods. The comparison in which the proposed method significantly outperforms the benchmark method is considered as a "win" case whereas it is a "loss" if FMTGPTL is outperformed. In these figures, the numbers of win (loss) cases are shown, where the green (red) refers to the number of win/"+" (loss/"-") cases for FMTGPTL against each method.

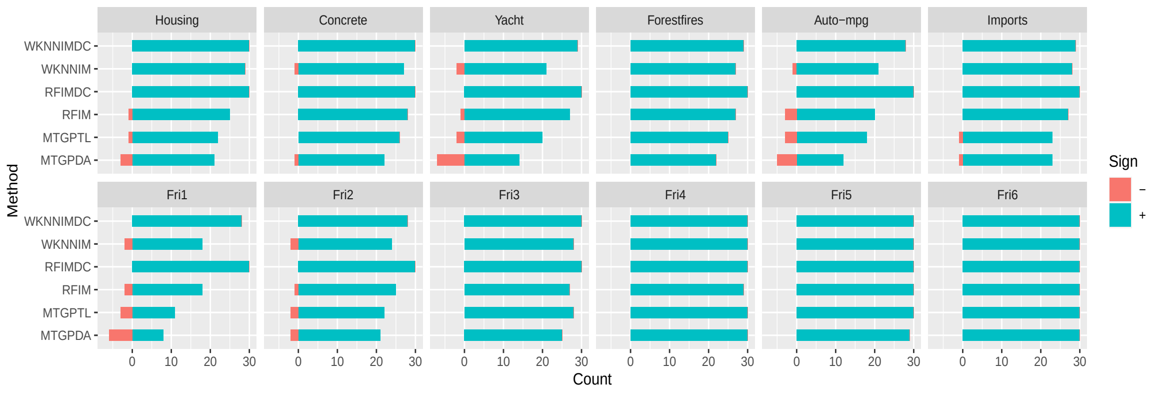


Figure 7.3: The statistical significance test results, where "+" ("-") refers to the number of win (loss) cases of FMTGPTL against each method on real-world data with synthetic incompleteness in homogeneous domains.

The significance test results show the notable improvement that can be gained when using the proposed method. From these results, the proposed method wins in most comparisons with each method on all the considered data sets. When comparing the significance test results in Figure. 7.3 and the results in Table 7.4, it is important to notice that even when the regression error difference between FMTGPTL and another method is not high, the results of FMTGPTL are more likely to be significantly better. For example, although the regression improvement of FMTGPTL over MTGPDA is about 4% on the Housing ("Hou") data set, FMTGPTL significantly outperforms MTGPDA in about 20 out of the 30 comparisons.

7.4.2 Symbolic Regression on Real-world Data with Synthetic Incompleteness in Heterogeneous Domains

7.4.2.1 Symbolic Regression Errors on Heterogeneous Domains

For the heterogeneous transfer learning scenario, the symbolic regression results in terms of RSE are shown in Table 7.5. In this scenario, the SD has only the first half of the available features. It can be noticed that the improvement achieved by the FMTGPTL method over the other methods is lower than that of the homogeneous scenario. This is expected as the SD has a smaller pool of features to be used for the transformation construction process, which limits the construction ability of the features. Moreover, the first half of the features might not contain the most useful features for building the transformations.

7.4.2.2 The Significance Test Results on Heterogeneous Domains

The results of the statistical significance tests between FMTGPTL and the benchmark methods on real-world data with synthetic incompleteness in heterogeneous domains are shown in Figure. 7.4. These results are consistent with those of the regression errors. The proposed method show less significant performance in the heterogeneous transfer learning scenario than the situation of homogeneous domains. However, the FMTGPTL method still has a considerable amount of significant wins against any other method.

7.4.3 SR with Real Incompleteness

The applicability of FMTGPTL to real-world incompleteness has been compared with the benchmark methods on Auto (Aut) and Imports (Imp) data sets. The results and the construction of the transformations in the previous sections on these data sets are based on using only the complete instances. Here, the learned models based on learning on synthetic in-

Table 7.5: The symbolic regression results in terms of RSE in **heterogeneous** domains with synthetic incompleteness.

Method		Hou	Con	Yac	For	Aut	Imp	Fri1	Fri2	Fri3	Fri4	Fri5	Fri6
WKNNIM	Mean	1.0306	1.2634	1.1526	1.0552	0.5417	0.6794	0.8458	0.8493	0.8689	0.8958	0.9187	0.9644
	Stdv	0.1704	0.3052	0.3345	0.0432	0.0753	0.1298	0.2111	0.3265	0.2833	0.1881	0.2067	0.2284
	PRC	12%	10%	10%	10%	4%	12%	4%	5%	5%	8%	10%	13%
RFIM	Mean	0.9542	1.2439	1.1355	1.052	0.548	0.6461	0.8219	0.8462	0.8676	0.8779	0.887	0.9118
	Stdv	0.1378	0.2879	0.3116	0.0503	0.08	0.1162	0.2241	0.2846	0.2863	0.2458	0.2818	0.3296
	PRC	5%	8%	9%	10%	5%	7%	2%	4%	5%	7%	6%	8%
MTGPTL	Mean	0.9741	1.2463	1.1093	0.986	0.5404	0.6365	0.8218	0.8299	0.8491	0.8589	0.8807	0.9176
	Stdv	0.1133	0.2314	0.2639	0.034	0.076	0.1016	0.2694	0.216	0.2768	0.3055	0.3139	0.2948
	PRC	7%	9%	7%	4%	4%	6%	2%	2%	3%	4%	6%	8%
MTGPDA	Mean	0.9182	1.214	1.0425	0.9656	0.5261	0.6325	0.8092	0.8216	0.8307	0.8389	0.8655	0.8985
	Stdv	0.1002	0.2263	0.2645	0.0285	0.082	0.0991	0.2322	0.2193	0.278	0.2058	0.2167	0.3136
	PRC	2%	6%	1%	2%	1%	5%	0%	1%	1%	2%	4%	6%
FMTGPTL	Mean	0.9026	1.1384	1.0334	0.9489	0.5201	0.6011	0.8084	0.8104	0.8213	0.8208	0.8311	0.8415
	Stdv	0.1841	0.1862	0.2549	0.2145	0.3031	0.2855	0.2836	0.1847	0.2811	0.2575	0.2346	0.2864

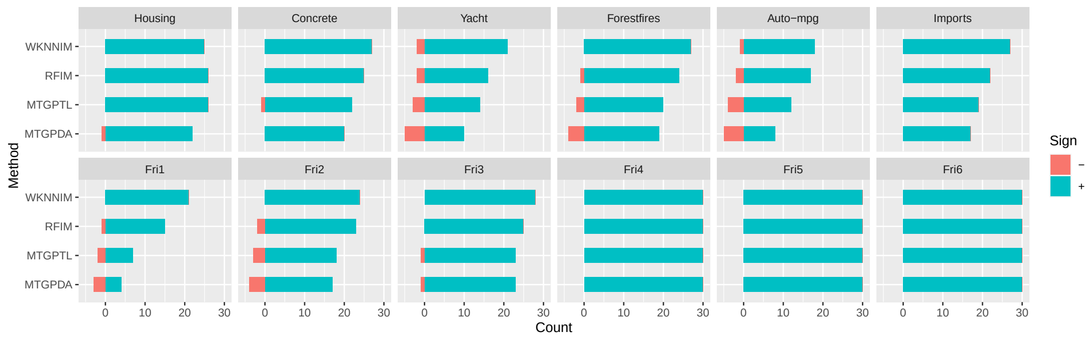


Figure 7.4: The symbolic regression comparison results between the proposed method, FMTGPTL, and the benchmark methods, where "+" ("-") refers to the number of win (loss) comparisons for FMTGPTL against each method on real-world data with synthetic incompleteness in heterogeneous domains.

complete data are applied to predict the regression output of the hold out incomplete instances.

For each method, the models obtained from the homogeneous scenario are used to predict the incomplete instances and their mean and standard deviation along with the significance test results are shown in Table 7.6. The significance on the difference between the results of each benchmark method and FMTGPTL is shown in the column "ST". For each method, "+" ("-") means that FMTGPTL significantly outperforms (outperformed by) the corresponding method while "=" means no significant difference.

According to the obtained results, FMTGPTL significantly outperforms all the benchmark methods on the two data sets. Such results confirm the superiority of FMTGPTL that has been shown in the synthetic incompleteness situation. The robust results of the FMTGPTL method during the transformation construction process with synthetic incompleteness lead to better performance when applied to real-world incompleteness. This is most likely because the learning process was able to extract useful knowledge that help the symbolic regression models generalise well. For example, the feature knowledge extracted considering synthetic incompleteness

Table 7.6: The symbolic regression results with regard to test RSEs and the statistical significance test on **real incomplete instances**

Method	Data	Auto	Imports
WKNNIM	Mean	0.9047	0.70063
	Sdev	0.3366	0.1065
	ST	+	+
WKNNIMDC	Mean	1.2557	0.7889
	Sdev	0.4567	0.1846
	ST	+	+
RFIM	Mean	0.8927	0.6896
	Sdev	0.1757	0.0684
	ST	+	+
RFIMDC	Mean	1.268	0.8644
	Sdev	0.3896	0.1131
	ST	+	+
MTGPTL	Mean	0.8892	0.6733
	Sdev	0.1347	0.0617
	ST	+	+
MTGPDA	Mean	0.8745	0.6679
	Sdev	0.1117	0.0312
	ST	+	+
FMTGPTL	Mean	0.8594	0.6588
	Sdev	0.1064	0.0458

remain useful for dealing with real incomplete instances that belong to the same data set.

7.4.4 Feature Selection and Transformation Construction Time

Here, we compare the proposed method with the other MFTGP-based transfer learning methods. This comparison aims at investigating whether the proposed method improve the MTGP-based transfer learning process. We did not compare with the other methods (WKNNIM, WKNNIMDS, RFIM, and RFIMDC) as they do not involve transfer learning. Table 7.7 shows the average computation time for constructing MTGP-based feature transformations in seconds (rounded to integers $\times 10$) for the GP-based transfer learning methods. The table also shows the average of the number of constructed features (trees) in the obtained MTGP-based feature transformation.

Table 7.7: The average number of constructed features and the average computation time of constructing MTGP-based feature transformations in seconds (rounded to integers $\times$ 10) for the GP-based transfer learning methods.

	MTGPTL		MTGPDA		FMTGPTL	
Data	#Trees	Time	#Trees	Time	#Trees	Time
Hou	13	908	13	1183	8	994
Con	9	893	9	1076	6	869
Yac	6	441	6	655	4	613
For	13	931	13	1118	10	1089
Aut	6	543	6	739	4	641
Imp	14	844	14	986	5	527
Fri1	5	622	5	783	5	819
Fri2	10	934	10	1123	6	903
Fri3	20	1121	20	1631	8	987
Fri4	40	1565	40	1952	11	1169
Fri5	80	2025	80	2578	11	1192
Fri6	160	2684	160	3544	14	1354

The results show that, in general, the proposed method constructs the MTGP-based feature transformations more efficiently compared with the benchmark MTGP-based transfer learning methods. This is because it employs a feature selection mechanism that reduces the number features (trees) to be constructed during the evolutionary process. Such feature reduction leads to a reduction in the computation time. Note that the more irrelevant features the data has, the more time reduction FMTGPTL achieves. For instance, on the Fri6 data set, which has the highest number of irrelevant/redundant features, FMTGPTL gained a time reduction of more than 60% less than the MTGPDA and more than 50% less than the MTGPTL computation time.

On the contrary, when there are no redundant features, such time reduction is not clear and, even, the proposed method might take more time. This is because feature selection might not be able to reduce the number of selected TD features. Instead, more time is consumed due to the extra step of checking the features in the TD side. For example, the FMTGPTL method spends more time than the other method on the Fri1 data set. In this data set, all features are relevant and they are selected by the proposed method. However, the increase of computational cost in such cases is not significant as the feature selection mechanism is not too costly.

To compare the three strategies regarding the generalisation ability, the significance test results of the symbolic regression on the test data are shown in Table 7.8.

Table 7.8: Summary of the comparison results between the suggested feature selection strategies when integrated with the MTGP-based transfer learning.

Strategy	Source	Target	Source+Target
Source	0	205	345
Target	89	0	302
Source+Target	0	27	0

Table 7.8 shows the number of wins of each strategy has in the column against the strategy in the row. There are 12 data sets, each has 30 different incomplete data copies, which sums up to 360 (360=12*30) comparisons. The strategy of combining feature weighting from the SD and the TD outperforms using only one of them individually by a big difference. Conversely, the use of feature selection based only on the weights from the SD shows the worst performance. This because these features represent a

domain whose distribution is different from that in the TD, on which the performance is evaluated. These results show significant improvement that is gained by combining the importance from the two domains.

7.5 Chapter Summary

This chapter proposes a new multi-stage GP-based transfer learning method for symbolic regression with missing values. It integrates a novel feature selection mechanism with a multi-tree GP evolutionary process that constructs multiple-feature transformations from complete source domains to incomplete target domains. Feature-based knowledge extracted from the source and the target domains is utilised to select features that enhance the transformations gradually. These transformations are built with the goal of handling the missing values in the target domain such that the symbolic regression learning performance is improved.

The proposed method has been successfully applied to transferring useful knowledge from complete domains to help performing symbolic regression in incomplete domains considering various situations. Two types of data sets are considered: real-world data sets and synthetic data sets. For the real-world data sets, two missingness kinds are addressed: synthetic incompleteness and real incompleteness. Moreover, two transfer learning scenarios are examined: homogeneous transfer learning and heterogeneous transfer learning.

The experimental results show the superiority of the proposed method over the benchmark methods in all considered cases. The proposed method is compared with four traditional learning methods and two MTGP-based transfer learning methods. While all MTGP-based transformation methods achieve positive knowledge transfer compared with the traditional methods, the proposed involvement of feature selection is a key factor for the proposed method to advance the benchmark MTGP-based transfer learning methods with respect to both the effectiveness and the efficiency.

One main finding in this work is that combining knowledge form both source and target domains is profitable for the evolutionary transfer learning process when handling the missingness issue. Although employing feature importance knowledge from the target domain alone is not reliable due to the shortage in the available data, it was advantageous when combined with feature importance knowledge from the source domain. Such a combination was beneficial in guiding the adaptation of the evolved MTGP individuals gradually through multiple stages by reducing the number of trees in the individuals according to the extracted feature importance knowledge.

The integrated feature selection mechanism enables the MTGP evolutionary transfer learning to be adapted to build transformations for important features instead of mapping all existing features. This ability addresses the limitation of MTGP feature-based transformations of being applicable to domains with a small number of features. The proposed approach not only increases the applicability of the MTGP-based transfer learning method to more symbolic regression tasks on incomplete data but also will open the door towards extending the method to high-dimensional domains.

Chapter 8

Conclusions

The overall goal of this thesis was to improve the performance of genetic programming based symbolic regression on incomplete data. This goal has been successfully achieved by developing a number of new methods based on GP for data imputation, instance selection, feature selection, and transfer learning. The proposed methods were evaluated and compared with state-of-the-art methods on a range of incomplete regression data sets considering various learning scenarios. The obtained results show that the proposed methods generally outperform the corresponding state-of-the-art methods on constructing symbolic regression models for incomplete data.

The rest of this chapter is organised as follows. Major conclusions from each contribution chapter of this thesis are firstly presented and we also highlight the findings that are discovered during the course of research. This chapter than provides potential research directions for future work.

8.1 Main Conclusions

This section provides the conclusions of the five contribution chapters (Chapter 3 to Chapter 7).

8.1.1 GP-based Imputation for Symbolic Regression on Incomplete Data

Chapter 3 presented a new method to improve imputation for symbolic regression on incomplete data by combining GP and WKNN.

8.1.1.1 Combining WKNN and GP for Data Imputation

This thesis found that the combination of GP and WKNN improves the imputation accuracy for missing values. In GP-based imputation, feature-wise regression models are built to estimate the missing values in incomplete features. On the other hand, the WKNN imputation is based on an instance-wise similarity mechanism, which estimates missing values in incomplete instances using existing values in similar instances. However, the pure GP-based approach does not utilise the instances similarity while the WKNN approach does not exploit the possible feature predicatability. This thesis showed that combining the two approaches enables gaining the benefits of each while mitigating their drawbacks.

It was found that integrating WKNN with GP can provide more accurate imputation than using WKNN and GP individually. Moreover, it provides better imputation performance than a number of popular imputation methods. Another important advantage of the proposed method is its efficient imputation in the application stage. Once the imputation models are trained, the application of these models to impute new test data is fast.

8.1.1.2 WKNN-GP Imputation for Symbolic Regression on Incomplete Data

This thesis showed how imputation based on the combination of GP and WKNN helped improve the symbolic regression results on incomplete data. Performing symbolic regression after WKNN-GP imputation achieved better performance than using several imputation methods such as support vector machine (SVM), Bayesian regression (BR), linear regression (LR), decision trees (DT), multilayer perceptron (MLP), k-nearest neighbour (KNN), and pure GP imputation. The proposed method was shown to be favourable towards producing satisfactory symbolic regression models.

It was found that the better the imputation performance, the better the symbolic regression performance. This is because better imputation provides better input data which in turn facilities learning better symbolic regression models. Note that, as the imputation comes with estimation errors, having a high portion of missing data causes worse performance.

8.1.2 Instance Selection Methods for Symbolic Regression on Incomplete Data

Chapter 4 utilised instance selection to improve imputation for symbolic regression on incomplete data.

8.1.2.1 GP with Hybrid Tree and Vector Representation (GPTV)

This thesis found that instance selection can be utilised to improve symbolic regression performance on incomplete data. Instances that participate positively in building GP models for symbolic regression during the training process are more likely to be beneficial in the testing process. The contribution of these instances is based on WKNN imputation while constructing GP-based symbolic regression models on incomplete data. That is, when selecting the instances, the symbolic regression performance is considered in evaluating the fitness function along with the reduction ratio of selected instances.

The results on both synthetic and real incompleteness situations showed that hybridising tree representation for symbolic modelling with vector representation for instance selection can achieve a good performance. This goodness is reflected by the enhanced symbolic regression accuracy on incomplete data. Moreover, the application of the symbolic regression models produced using this method is efficient to predict new test incomplete data.

8.1.2.2 GPTV for Instance Selection and Symbolic Regression on Incomplete Data

It was found that instance selection provides a smaller set of representative training data to perform imputation, which in turn reduces the imputation time of the test data. This is important as several imputation methods such as WKNN depend on the number of instances used for imputation. Therefore, using instance selection to estimate missing values for unseen incomplete instances can speed up the application process.

Although using instance selection to estimate missing values can potentially reduce the imputation accuracy as less data is used, this issue is mitigated by utilising the accuracy of symbolic regression to guide the selection process during the training process. This mechanism makes the imputation provide estimates that eventually leverage the symbolic regression performance. Selecting instance and evolving symbolic regression at the same time brings benefits to the two processes. The instances are selected based on their contributions in improving data imputation such that symbolic regression is enhanced.

8.1.3 GP-based Feature Selection for Symbolic Regression with High-dimensional Incomplete Data Sets

Chapter 5 provided a feature selection method to improve imputation for symbolic regression on incomplete data.

8.1.3.1 IGP for Feature Selection on Incomplete Data

The thesis showed that feature selection can be done effectively and efficiently by using IGP-based method for symbolic regression on incomplete data. GP has been widely used for feature selection with complete data, but it cannot directly work on data sets with missing values. To deal with this problem, Chapter 5 presented IGP which uses a set of interval functions to replace the normal function set in traditional GP. To use IGP for feature selection, first, each missing value for a feature is replaced by an interval associated with the feature. The interval functions then operate on the intervals as well as regular value to compute an output value for the feature. The features used by IGP are considered as the selected set of features. These features are then used in a learning process that includes imputation followed by symbolic regression.

8.1.3.2 IGP-based Feature Selection for High-dimensional Symbolic Regression on Incomplete Data

This thesis found that IGP can be utilised for feature selection to improve the efficiency and effectiveness of symbolic regression on incomplete high-dimensional data. By removing redundant and irrelevant features, feature selection produces a high quality feature set which then helps to build better symbolic regression models. More importantly, feature selection also reduces the number of incomplete features, so less imputation effort is required which then saves some time of the learning process. Moreover, removing redundant and irrelevant features can reduce the chance of noisy features, which improves the accuracy of the built models.

Using IGP is an alternative to using traditional GP to deal with the missing data. It was found that IGP can select useful features that could be complete or incomplete directly from incomplete data. The usefulness of the selection is maintained such that better symbolic regression models are generated. The key reason is that replacing a missing values with feature intervals can reflect the uncertainty associated with the missingness in these features. Therefore, IGP-based feature selection can be utilised to generate a set of predictive features. Moreover, the IGP-based feature selection followed by imputation then symbolic regression is more efficient than the approach of imputation followed by symbolic regression.

In the case of high-dimensional data, many features could be irrelevant/redundant. Imputing such features adds unnecessary cost to the learning process. Therefore, performing features selection was found to be gainful for the symbolic regression performance. The IGP-based approach involves estimating the intervals of the features and replacing the missing values with these intervals, which is not a time consuming process. This thesis found that, the approach of IGP-based feature selection followed by imputation then symbolic regression is better than the approach of just imputation followed by symbolic regression in terms of both effeciency and effectivness.

8.1.4 GP-based Transfer Learning Methods for Symbolic Regression on Incomplete Data

Chapters 6 and 7 presented transfer learning methods to improve imputation for symbolic regression on incomplete data in domains with small number of instances.

8.1.4.1 Multi-tree GP for Transfer Learning in Symbolic Regression on Incomplete Data

It was found that GP-based transfer learning can be successfully employed for symbolic regression on incomplete data, especially in the case of data shortage. Chapter 6 showed that multi-tree GP represents a powerful means to building maps between domains. The natural ability of GP to construct features was found to be a key factor in building feature-based transformations from a source domain to a target domain.

The thesis found that knowledge related to the importance of features and instances based on symbolic regression learning in certain domains is beneficial when transferred to improve learning in different related domains. Such knowledge extracted from source domains with complete data can help handling the missing values in target domains with incomplete data.

Another finding in this chapter is that probabilistic genetic operators lead to better results than the use of random operators. This chapter develops new MTGP crossover and mutation operators, where the selection of trees to be mated/muted is probabilistic based on their fitness values. Such a mechanism was shown to be advantageous in guiding the convergence of the evolutionary process in a profitable manner due to considering the goodness-of-fit to favour trees that participate in generating the new generations.

8.1.4.2 Integrating Feature Selection with MTGP-based Transfer Learning for Symbolic Regression on Incomplete Data

Chapter 7 found that feature selection can be effectively integrated with MTGP-based transfer learning for symbolic regression on incomplete data. The construction of feature-based transformations between domains is enhanced when feedback from a feature selection process is employed. It was found that feature selection based on both the learning in the source domain and the target domain is more useful than feature selection based only on one of the domains.

The inclusion of feature selection enables the MTGP-based transfer learning to work better on domains with higher dimensionalities. The combination of MTGP-based feature construction and single-tree GP feature selection helps improving the way of imputing the incomplete data in domains with a limited number of instances. This improvement is reflected in the performance of symbolic regression in the target such limited domains.

8.2 Limitations of Research

This section provides some limitations of the research conducted in this thesis.

One of the limitations of the proposed methods in this thesis is related to the parameter settings. The parameters are chosen empirically based on pilot experiments and no extensive parameter tuning has been conducted. Therefore, we can not claim that the used parameters are optimal. We think that the parameters can be tuned using different methods search as grid search or random search, which might enhance the results significantly.

Another limitation of this thesis is related to the practicality of the proposed methods. As a typical machine learning research, the proposed methods are not expected to work in all possible cases. There are some key assumptions for the proposed methods. Therefore these methods might have some limited applicability. Here are some examples of these assumptions. For the imputation methods, the incomplete features are imputed based on other features using the regression-based approach. This implies that the complete features are predictors for the incomplete features. This assumption might not hold true in all cases.

A key assumption in Chapter 6 and Chapter 7 is stated on page 135, i.e., "The more important features/instances in the source domain, the more likely they benefit the target domain, and therefore, the larger weight they get in the transformation construction process". This assumption works for our experiments because the task is the same in both the source domain and the target domain. Moreover, the input spaces are related. In case of having similar situations, this assumption may hold true in real-world applications as well. For example, if in the target domain we are addressing the COVID-19 task in a country with poor data (e.g. Afganstan), then the source domain can be a country with rich data about COVID-19 (e.g. Canada). The knowledge extracted from solving the prediction task (symbolic regression in our work) about COVID-19 in Canada can be reused to perform the same task in Afganistan. Technically, the learning task is the same, $\mathcal{T}^s = \mathcal{T}^t$, which is symbolic regression. Moreover, the target variable is the same, $y^t = y^s$ (although its distribution might be different in the two domains). However, the feature space or/and its distribution can be different in the two domains. Based on these conditions there is no need to measure the correlation between the domains. Another issue regarding the transfer learning work in Chapter 6 is that the distribution mismatch function (Equation 6.12) treats the distributions of the features independently. Although this assumption is considered in several machine learning approaches, such as Bayesian-based methods, it might not hold true in all real-world tasks.

8.3 Future Work

There are a number of potential future work directions suggested by the thesis.

8.3.1 Improve Handling Missing Values for Symbolic Regression on Incomplete Data

This thesis proposed several approaches to improve symbolic regression on incomplete data. However, there are some aspects that need to be investigated.

8.3.1.1 Improve GP-based imputation for Symbolic Regression on Incomplete Data

Regression-based imputation is a common approach to impute missing values. This approach is adopted in this thesis by introducing a GP-based imputation method for symbolic regression on incomplete data. GP has also been successfully applied to data imputation for classification on incomplete data [223, 225]. This success could be further exploited by developing more advanced GP-based imputation methods to provide more accurate estimates for the missing values. This includes methods that have been shown to improve symbolic regression such as semantic GP [237] and Zoetrope [31]. Moreover, more enhancement approaches can be integrated such as feature standardisation [171], coefficient optimisation [67], and bias-variance decomposition [172]. An effective investigation of using such methods for data imputation will be an interesting future work.

8.3.1.2 Improve Symbolic Regression on Incomplete Data with Mixed Data Types

A number of methods have been proposed to improve symbolic regression on incomplete data in this thesis. However, the issue of the data types was not adequately examined. The use of GP-based regression for imputation was shown to be beneficial in Chapter 3. Although regression can work for categorical/nominal data, it is more suitable for numerical/real data. Therefore, it is expected to gain more advantages if new methods are developed to handle different types of data effectively.

One possible approach to deal with mixed data types is by using a GP-based symbolic regressor-classifier modelling [121]. In this approach, GP can be used to build a regression imputer for numerical data and a classification imputer for categorical data. This might need to use strongly-typed GP. Another way to address this issue is to use encoding schemes for missing values that appear in categorical features. Such schemes can replace the missing values in these features with a new category and carry out the learning process without imputation.

8.3.2 GP for Direct Symbolic Regression on Incomplete Data

The methods in this thesis involve pr-processing steps such as data imputation to handle the missingness issue before performing symbolic regression. However, it would be interesting to investigate if it is possible to carry out effective symbolic regression directly on incomplete data.

8.3.2.1 Develop GP-based Methods for Direct Symbolic Regression on Incomplete Data

The ability to perform learning directly from incomplete data is an advantageous feature for the learning algorithm. This is one of the factors that increase the popularity of some methods such as C4.5 and CART [255]. Unfortunately, this ability is not available in classic GP for symbolic regression. In [224], interval GP is proposed to directly construct classifiers on incomplete data. This method is improved in [232] by utilising ensemble learning. These methods show more effective and efficient classification results than the imputation approach. However, these methods are designed for classification tasks and when we adapted them for direct symbolic regression on incomplete data, the results were not encouraging. Therefore, there is a need to come up with a new approach to perform symbolic regression on incomplete data without imputation.

One possible direction in this regard is proposing more advanced representations for GP. This could be by modified function sets that can tolerate the missing values safely. Another approach is to develop a new structure for GP. For example, a layered GP can be used where the first layer is responsible for handling the missing values implicitly, while the second layer performs symbolic regression on the complete outcomes of the first layer.

8.3.2.2 Investigate GP-based Direct Methods to Deliver Learning Tasks that Improve Symbolic Regression on Incomplete Data

Although the methods that perform symbolic regression on incomplete data directly might not be effective, they can be utilised for different learning tasks. In this thesis, IGP has been used for feature selection. However, it would be interesting to investigate IGP for feature construction in symbolic regression on incomplete data. On the other hand, as such methods are more direct and efficient than imputation-based methods, so it is worth to examine its applicability to instance selection and transfer learning tasks.

8.3.3 Considering More Data Challenges with Symbolic Regression On Incomplete Data

The challenges that drive to the presented contributions in this thesis are related to data quality problems. However, not all such problems are considered and there is more to address in future work.

8.3.3.1 Symbolic Regression on Imbalanced Incomplete Data

Although data imbalance is commonly connected to classification tasks, there are different kinds of imbalance that could be experienced in regression data sets. This imbalance could be seen from the input space side or from the target variable aspect. For the input space, some features could be unevenly distributed over their range. Similarly, the distribution of the data could be imbalanced with respect to the target variable. Some regions of the target variable/feature space might have more instances than others. Imbalanced data is more challenging in the presence of missing values.

This issue could be a subject of a future research. One way to approach this challenge is by integrating data imbalance mitigating techniques with data missingness handling methods. Fore example, some sampling methods that are successfully used to deal with imbalanced data can be utilised to deal with symbolic regression on incomplete data.

8.3.3.2 Symbolic Regression on Large-scale Incomplete Data

An interesting point to investigate is the impact of the incompleteness on learning from large-scale data. If the missingness ratio is low then it seems fine to just remove data with missing values. However, what if there is a large missing portion of the data. It might be desirable to recover some missing data even if the data set is large. If the impact of missingness in large data is significant, then some incompleteness mitigation measures can be taken.

Data imputation is a time-consuming process, which might make it unfavourable when dealing with large-scale data. Furthermore, some presented methods were not computationally suitable for large-scale data. One important question that could be addressed as a future work is the impact of missingness on symbolic regression in large-scale data. Therefore, it is desired to develop more scalable methods for symbolic regression on incomplete large-scale data. One potential future approach to handling this issue is more flexible instance selection methods. Another approach that could be utilised is parallelism. Developing methods that can handle the missing values in a parallel manner could improve the ability of dealing with large-scale incomplete data.

Bibliography

[1] missingpy: Missing data imputation for python. https://github.com/epsilon-machine/missingpy. Accessed: 2020-05-06.

[2] AHA, D. W., KIBLER, D., AND ALBERT, M. K. Instance-based learning algorithms. *Machine learning 6*, 1 (1991), 37–66.

[3] AKTAŞ, M. S., KAPLAN, S., ABACI, H., KALIPSIZ, O., KETENCI, U., AND TURGUT, U. O. Data imputation methods for missing values in the context of clustering. In *Big Data and Knowledge Sharing in Virtual Organizations*. IGI Global, 2019, pp. 240–274.

[4] AL-HELALI, B., CHEN, Q., XUE, B., AND ZHANG, M. A hybrid GP-KNN imputation for symbolic regression with missing values. In *Australasian Joint Conference on Artificial Intelligence* (2018), Springer, pp. 345–357.

[5] AL-SAHAF, H., SONG, A., AND ZHANG, M. Hybridisation of genetic programming and nearest neighbour for classification. In *Evolutionary Computation (CEC), 2013 IEEE Congress on* (2013), IEEE, pp. 2650–2657.

[6] AL-SAHAF, H., ZHANG, M., AND JOHNSTON, M. Binary image classification using genetic programming based on local binary patterns. In *Image and Vision Computing New Zealand (IVCNZ), 2013 28th International Conference of* (2013), IEEE, pp. 220–225.

[7] ALLA, S., AND ADARI, S. K. What is mlops? In *Beginning MLOps with MLFlow*. Springer, 2021, pp. 79–124.

[8] ALLISON, P. D. Multiple imputation for missing data: A cautionary tale. *Sociological methods & research 28*, 3 (2000), 301–309.

[9] ALPAYDIN, E. *Introduction to machine learning*. MIT press, 2009.

[10] AMASYALI, M. F., AND ERSOY, O. K. A study of meta learning for regression. *ECE Technical Reports* (2009), 386.

[11] ANINDITA, N., NUGROHO, H. A., AND ADJI, T. B. A combination of multiple imputation and principal component analysis to handle missing value with arbitrary pattern. In *2017 7th International Annual Engineering Seminar (InAES)* (2017), IEEE, pp. 1–5.

[12] ANJUM, A., SUN, F., WANG, L., AND ORCHARD, J. A novel continuous representation of genetic programmings using recurrent neural networks for symbolic regression. *arXiv preprint arXiv:1904.03368* (2019).

[13] ANNAS, S., KARTIKASARI, P., AND ARISANDI, R. Handling incomplete data with regression imputation. In *Journal of Physics: Conference Series* (2021), vol. 1752, IOP Publishing, p. 012049.

[14] ARDEH, M. A., MEI, Y., AND ZHANG, M. A novel genetic programming algorithm with knowledge transfer for uncertain capacitated arc routing problem. In *Pacific Rim International Conference on Artificial Intelligence* (2019), Springer, pp. 196–200.

[15] ARDEH, M. A., MEI, Y., AND ZHANG, M. Transfer learning in genetic programming hyper-heuristic for solving uncertain capacitated arc routing problem. In *2019 IEEE Congress on Evolutionary Computation (CEC)* (2019), IEEE, pp. 49–56.

[16] ARNALDO, I., KRAWIEC, K., AND O'REILLY, U.-M. Multiple regression genetic programming. In *Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation* (2014), pp. 879–886.

[17] ARSLAN, S., AND OZTURK, C. Multi hive artificial bee colony programming for high dimensional symbolic regression with feature selection. *Applied Soft Computing 78* (2019), 515–527.

[18] AUGUSTO, D. A., AND BARBOSA, H. J. Symbolic regression via genetic programming. In *Neural Networks, 2000. Proceedings. Sixth Brazilian Symposium on* (2000), IEEE, pp. 173–178.

[19] AUSSEM, A., AND DE MORAIS, S. R. A conservative feature subset selection algorithm with missing data. *Neurocomputing 73*, 4-6 (2010), 585–590.

[20] BÄCK, T., FOGEL, D. B., AND MICHALEWICZ, Z. Handbook of evolutionary computation. *Release 97*, 1 (1997), B1.

[21] BANZHAF, W., NORDIN, P., KELLER, R. E., AND FRANCONE, F. D. *Genetic programming: an introduction*, vol. 1. Morgan Kaufmann San Francisco, 1998.

[22] BARNARD, J., RUBIN, D. B., AND SCHENKER, N. Multiple imputation methods. *Encyclopedia of Biostatistics 5* (2005).

[23] BATISTA, G. E., MONARD, M. C., ET AL. A study of k-nearest neighbour as an imputation method. *HIS 87*, 251-260 (2002), 48.

[24] BEESLEY, L. J., BONDARENKO, I., ELLIOTT, M. R., KURIAN, A. W., KATZ, S. J., AND TAYLOR, J. M. Multiple imputation with missing data indicators. *arXiv preprint arXiv:2103.02033* (2021).

[25] BERETTA, L., AND SANTANIELLO, A. Nearest neighbor imputation algorithms: a critical evaluation. *BMC medical informatics and decision making 16*, 3 (2016), 74.

[26] BERTHOLD, M. R., AND HUBER, K.-P. Missing values and learning of fuzzy rules. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 6*, 02 (1998), 171–178.

[27] BEYER, H.-G., AND SCHWEFEL, H.-P. Evolution strategies–a comprehensive introduction. *Natural computing 1*, 1 (2002), 3–52.

[28] BHARDWAJ, H., SAKALLE, A., BHARDWAJ, A., TIWARI, A., AND VERMA, M. Breast cancer diagnosis using simultaneous feature selection and classification: A genetic programming approach. In *2018 IEEE Symposium Series on Computational Intelligence (SSCI)* (2018), IEEE, pp. 2186–2192.

[29] BIGGIO, L., BENDINELLI, T., LUCCHI, A., AND PARASCANDOLO, G. A seq2seq approach to symbolic regression. In *Learning Meets Combinatorial Algorithms at NeurIPS2020* (2020).

[30] BISONG, E. Introduction to scikit-learn. In *Building Machine Learning and Deep Learning Models on Google Cloud Platform*. Springer, 2019, pp. 215–229.

[31] BOISBUNON, A., FANARA, C., GRENET, I., DAEDEN, J., VIGHI, A., AND SCHOENAUER, M. Zoetrope genetic programming for regression. *arXiv preprint arXiv:2102.13388* (2021).

[32] BORGWARDT, K. M., GRETTON, A., RASCH, M. J., KRIEGEL, H.-P., SCHÖLKOPF, B., AND SMOLA, A. J. Integrating structured biological data by kernel maximum mean discrepancy. *Bioinformatics 22*, 14 (2006), e49–e57.

[33] BRAMEIER, M. F., AND BANZHAF, W. *Linear genetic programming*. Springer Science & Business Media, 2007.

[34] BRANDEJSKY, T. Model identification from incomplete data set describing state variable subset only–the problem of optimizing and

predicting heuristic incorporation into evolutionary system. In *Nostradamus 2013: Prediction, Modeling and Analysis of Complex Systems*. Springer, 2013, pp. 181–189.

[35] BRIGHTON, H., AND MELLISH, C. Advances in instance selection for instance-based learning algorithms. *Data mining and knowledge discovery 6*, 2 (2002), 153–172.

[36] BROLØS, K. R., MACHADO, M. V., CAVE, C., KASAK, J., STENTOFT-HANSEN, V., BATANERO, V. G., JELEN, T., AND WILSTRUP, C. An approach to symbolic regression using feyn. *arXiv preprint arXiv:2104.05417* (2021).

[37] BU, F., CHEN, Z., ZHANG, Q., AND WANG, X. Incomplete big data clustering algorithm using feature selection and partial distance. In *5th International Conference on Digital Home* (2014), IEEE, pp. 263–266.

[38] CANO, J. R., HERRERA, F., AND LOZANO, M. Using evolutionary algorithms as instance selection for data reduction in kdd: an experimental study. *IEEE transactions on evolutionary computation 7*, 6 (2003), 561–575.

[39] CHAKRABARTI, S., ESTER, M., FAYYAD, U., GEHRKE, J., HAN, J., MORISHITA, S., PIATETSKY-SHAPIRO, G., AND WANG, W. Data mining curriculum: A proposal (version 1.0). *Intensive Working Group of ACM SIGKDD Curriculum Committee 140* (2006).

[40] CHEN, B.-W., AND WANG, J.-C. Incomplete data analysis. *Pattern Recognition* (2020).

[41] CHEN, C., LUO, C., AND JIANG, Z. Elite bases regression: A real-time algorithm for symbolic regression. *arXiv preprint arXiv:1704.07313* (2017).

[42] CHEN, Q. *Improving the Generalisation of Genetic Programming for Symbolic Regression*. PhD thesis, Victoria University of Wellington, 2018.

[43] CHEN, Q., XUE, B., MEI, Y., AND ZHANG, M. Geometric semantic crossover with an angle-aware mating scheme in genetic programming for symbolic regression. In *European Conference on Genetic Programming* (2017), Springer, pp. 229–245.

[44] CHEN, Q., XUE, B., NIU, B., AND ZHANG, M. Improving generalisation of genetic programming for high-dimensional symbolic regression with feature selection. In *IEEE Congress on Evolutionary Computation (CEC)* (2016), IEEE, pp. 3793–3800.

[45] CHEN, Q., XUE, B., AND ZHANG, M. Differential evolution for instance based transfer learning in genetic programming for symbolic regression. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion* (2019), pp. 161–162.

[46] CHEN, Q., XUE, B., AND ZHANG, M. Instance based transfer learning for genetic programming for symbolic regression. In *2019 IEEE Congress on Evolutionary Computation (CEC)* (2019), IEEE, pp. 3006–3013.

[47] CHEN, Q., XUE, B., AND ZHANG, M. Genetic programming for instance transfer learning in symbolic regression. *IEEE Transactions on Cybernetics* (2020).

[48] CHEN, Q., ZHANG, M., AND XUE, B. Feature selection to improve generalization of genetic programming for high-dimensional symbolic regression. *IEEE Transactions on Evolutionary Computation 21*, 5 (2017), 792–806.

[49] CHEN, Q., ZHANG, M., AND XUE, B. Genetic programming with embedded feature construction for high-dimensional symbolic re-

gression. In *Intelligent and Evolutionary Systems*. Springer, 2017, pp. 87–102.

[50] CHOU, C.-H., KUO, B.-H., AND CHANG, F. The generalized condensed nearest neighbor rule as a data reduction method. In *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on* (2006), vol. 2, IEEE, pp. 556–559.

[51] CHOUDHURY, A., AND KOSOROK, M. R. Missing data imputation for classification problems. *arXiv preprint arXiv:2002.10709* (2020).

[52] CHRISTO, V. E., NEHEMIAH, H. K., BRIGHTY, J., AND KANNAN, A. Feature selection and instance selection from clinical datasets using co-operative co-evolution and classification using random forest. *IETE Journal of Research* (2020), 1–14.

[53] COVER, T., AND HART, P. Nearest neighbor pattern classification. *IEEE transactions on information theory 13*, 1 (1967), 21–27.

[54] CRANMER, M., SANCHEZ-GONZALEZ, A., BATTAGLIA, P., XU, R., CRANMER, K., SPERGEL, D., AND HO, S. Discovering symbolic models from deep learning with inductive biases. *arXiv preprint arXiv:2006.11287* (2020).

[55] DASH, M., AND LIU, H. Feature selection for classification. *Intelligent data analysis 1*, 3 (1997), 131–156.

[56] DAWOOD, H. *Theories of interval arithmetic: mathematical foundations and applications*. LAP Lambert Academic Publishing, 2011.

[57] DE ANDRADE SILVA, J., AND HRUSCHKA, E. R. Eacimpute: an evolutionary algorithm for clustering-based imputation. In *Intelligent Systems Design and Applications, 2009. ISDA'09. Ninth International Conference on* (2009), IEEE, pp. 1400–1406.

[58] DE FRANÇA, F. O., AND ALDEIA, G. S. I. Interaction-transformation evolutionary algorithm for symbolic regression. *Evolutionary Computation* (2020), 1–25.

[59] DE SOUTO, M. C., JASKOWIAK, P. A., AND COSTA, I. G. Impact of missing data imputation methods on gene expression clustering and classification. *BMC bioinformatics 16*, 1 (2015), 1–9.

[60] DE WAAL, T., PANNEKOEK, J., AND SCHOLTUS, S. *Handbook of statistical data editing and imputation*, vol. 563. John Wiley & Sons, 2011.

[61] DENG, W.-Y., LIU, D., AND DONG, Y.-Y. Feature selection and classification for high-dimensional incomplete multimodal data. *Mathematical Problems in Engineering* (2018), 1–9.

[62] DHEERU, D., AND KARRA TANISKIDOU, E. UCI machine learning repository, 2017.

[63] DICK, G. Bloat and generalisation in symbolic regression. In *Asia-Pacific Conference on Simulated Evolution and Learning* (2014), Springer, pp. 491–502.

[64] DICK, G. Interval arithmetic and interval-aware operators for genetic programming. *arXiv preprint arXiv:1704.04998* (2017).

[65] DICK, G. Revisiting interval arithmetic for regression problems in genetic programming. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion* (2017), pp. 129–130.

[66] DICK, G. Sensitivity-like analysis for feature selection in genetic programming. In *Proceedings of the Genetic and Evolutionary Computation Conference* (2017), pp. 401–408.

[67] DICK, G., OWEN, C. A., AND WHIGHAM, P. A. Feature standardisation and coefficient optimisation for effective symbolic regression.

In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference* (2020), pp. 306–314.

[68] DINH, T. T. H., CHU, T. H., AND NGUYEN, Q. U. Transfer learning in genetic programming. In *2015 IEEE Congress on Evolutionary Computation (CEC)* (May 2015), pp. 1145–1151.

[69] DINH, T. T. H., CHU, T. H., AND NGUYEN, Q. U. Transfer learning in genetic programming. In *2015 IEEE Congress on Evolutionary Computation (CEC)* (2015), IEEE, pp. 1145–1151.

[70] DO, C. B., AND NG, A. Y. Transfer learning for text classification. In *Advances in Neural Information Processing Systems* (2006), pp. 299–306.

[71] DOMINGOS, P. *The master algorithm: How the quest for the ultimate learning machine will remake our world*. Basic Books, 2015.

[72] DONDERS, A. R. T., VAN DER HEIJDEN, G. J., STIJNEN, T., AND MOONS, K. G. A gentle introduction to imputation of missing values. *Journal of clinical epidemiology 59*, 10 (2006), 1087–1091.

[73] DORIGO, M., AND BIRATTARI, M. Ant colony optimization. In *Encyclopedia of machine learning*. Springer, 2011, pp. 36–39.

[74] EGGERMONT, J., ET AL. *Data mining using genetic programming: Classification and symbolic regression*. Institute for Programming research and Algorithmics, Leiden Institute of Advanced Computer Science, Faculty of Mathematics & Natural Sciences, Leiden University, 2005.

[75] EIBEN, A. E., AND SCHIPPERS, C. A. On evolutionary exploration and exploitation. *Fundamenta Informaticae 35*, 1-4 (1998), 35–50.

[76] EIBEN, A. E., SMITH, J. E., ET AL. *Introduction to evolutionary computing*, vol. 53. Springer, 2003.

[77] ESPEJO, P. G., VENTURA, S., AND HERRERA, F. A survey on the application of genetic programming to classification. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) 40*, 2 (2010), 121–144.

[78] FABBRI, R., AND DE LEÓN, F. G. A statistical distance derived from the kolmogorov-smirnov test: specification, reference measures (benchmarks) and example uses. *arXiv preprint arXiv:1711.00761* (2017).

[79] FARHANGFAR, A., KURGAN, L., AND DY, J. Impact of imputation of missing values on classification error for discrete data. *Pattern Recognition 41*, 12 (2008), 3692–3705.

[80] FENG, X., WU, S., SRIVASTAVA, J., AND DESIKAN, P. Automatic instance selection via locality constrained sparse representation for missing value estimation. *Knowledge-Based Systems 85* (2015), 210–223.

[81] FIGUEROA GARCIA, J. C., KALENATIC, D., AND LOPEZ BELLO, C. A. An evolutionary approach for imputing missing data in time series. *Journal of Circuits, Systems, and Computers 19*, 01 (2010), 107–121.

[82] FOGEL, D. B. *Evolutionary computation: the fossil record*. Wiley-IEEE Press, 1998.

[83] FORTIN, F.-A., RAINVILLE, F.-M. D., GARDNER, M.-A., PARIZEAU, M., AND GAGNÉ, C. Deap: Evolutionary algorithms made easy. *Journal of Machine Learning Research 13*, Jul (2012), 2171–2175.

[84] FRIEDJUNGOVÁ, M., AND JIRINA, M. Asymmetric heterogeneous transfer learning: A survey. In *DATA* (2017), pp. 17–27.

[85] FRIEDMAN, J. H., GROSSE, E., AND STUETZLE, W. Multidimensional additive spline approximation. *SIAM Journal on Scientific and Statistical Computing 4*, 2 (1983), 291–301.

[86] FU, W., XUE, B., ZHANG, M., AND GAO, X. Transductive transfer learning in genetic programming for document classification. In *Asia-Pacific Conference on Simulated Evolution and Learning* (2017), Springer, pp. 556–568.

[87] GARCÍA, J. C. F., KALENATIC, D., AND BELLO, C. A. L. Missing data imputation in time series by evolutionary algorithms. In *International Conference on Intelligent Computing* (2008), Springer, pp. 275–283.

[88] GARCÍA, J. C. F., KALENATIC, D., AND BELLO, C. A. L. Missing data imputation in multivariate data by evolutionary algorithms. *Computers in Human Behavior 27*, 5 (2011), 1468–1474.

[89] GARCÍA, S., CANO, J. R., AND HERRERA, F. A memetic algorithm for evolutionary prototype selection: A scaling up approach. *Pattern Recognition 41*, 8 (2008), 2693–2709.

[90] GARCÍA-LAENCINA, P. J., SANCHO-GÓMEZ, J.-L., AND FIGUEIRAS-VIDAL, A. R. Pattern classification with missing data: a review. *Neural Computing and Applications 19*, 2 (2010), 263–282.

[91] GAUTAM, C., AND RAVI, V. Data imputation via evolutionary computation, clustering and a neural network. *Neurocomputing 156* (2015), 134–142.

[92] GENTLEMAN, R., AND CAREY, V. Unsupervised machine learning. In *Bioconductor Case Studies*. Springer, 2008, pp. 137–157.

[93] GONÇALVES, I., SILVA, S., AND FONSECA, C. M. On the generalization ability of geometric semantic genetic programming. In *European Conference on Genetic Programming* (2015), Springer, pp. 41–52.

[94] GUAN, Z., LI, A., AND ZHU, T. Local regression transfer learning with applications to users' psychological characteristics prediction. *Brain informatics 2*, 3 (2015), 145–153.

[95] GUSTAFSON, S., BURKE, E. K., AND KRASNOGOR, N. On improving genetic programming for symbolic regression. In *Evolutionary Computation, 2005. The 2005 IEEE Congress on* (2005), vol. 1, IEEE, pp. 912–919.

[96] HART, P. The condensed nearest neighbor rule (corresp.). *IEEE transactions on information theory 14*, 3 (1968), 515–516.

[97] HASLAM, E., XUE, B., AND ZHANG, M. Further investigation on genetic programming with transfer learning for symbolic regression. In *2016 IEEE Congress on Evolutionary Computation (CEC)* (July 2016), pp. 3598–3605.

[98] HASLAM, E., XUE, B., AND ZHANG, M. Further investigation on genetic programming with transfer learning for symbolic regression. In *2016 IEEE Congress on Evolutionary Computation (CEC)* (2016), IEEE, pp. 3598–3605.

[99] HAYES, T. *Using Classification and Regression Trees (CART) and Random Forests to Address Missing Data*. PhD thesis, University of Southern California, 2017.

[100] HOLLAND, J. H. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press, 1992.

[101] HUANG, M.-W., LIN, W.-C., CHEN, C.-W., KE, S.-W., TSAI, C.-F., AND EBERLE, W. Data preprocessing issues for incomplete medical datasets. *Expert Systems 33*, 5 (2016), 432–438.

[102] HUANG, M.-W., LIN, W.-C., AND TSAI, C.-F. Outlier removal in model-based missing value imputation for medical datasets. *Journal of healthcare engineering 2018* (2018).

[103] IQBAL, M., AL-SAHAF, H., XUE, B., AND ZHANG, M. Genetic programming with transfer learning for texture image classification. *Soft Computing* (2019), 1–13.

[104] IQBAL, M., XUE, B., AL-SAHAF, H., AND ZHANG, M. Cross-domain reuse of extracted knowledge in genetic programming for image classification. *IEEE Transactions on Evolutionary Computation 21*, 4 (Aug 2017), 569–587.

[105] IQBAL, M., XUE, B., AND ZHANG, M. Reusing extracted knowledge in genetic programming to solve complex texture image classification problems. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining* (2016), Springer, pp. 117–129.

[106] JIANG, K., CHEN, H., AND YUAN, S. Classification for incomplete data using classifier ensembles. In *Neural Networks and Brain, 2005. ICNN&B'05. International Conference on* (2005), vol. 1, IEEE, pp. 559–563.

[107] JOHNSON, C. G. Artificial immune system programming for symbolic regression. In *European Conference on Genetic Programming* (2003), Springer, pp. 345–353.

[108] KAISER, L. Adjusting for baseline: change or percentage change? *Statistics in medicine 8*, 10 (1989), 1183–1190.

[109] KALKAN, H. Online feature selection and classification with incomplete data. *Turkish Journal of Electrical Engineering & Computer Sciences 22*, 6 (2014), 1625–1636.

[110] KEIJZER, M. Improving symbolic regression with interval arithmetic and linear scaling. In *European Conference on Genetic Programming* (2003), Springer, pp. 70–82.

[111] KENNEDY, J. Swarm intelligence. In *Handbook of nature-inspired and innovative computing*. Springer, 2006, pp. 187–219.

[112] KHAN, M., AND OLIVIER, J. Regression to the mean for the bivariate binomial distribution. *Statistics in medicine 38*, 13 (2019), 2391–2412.

[113] KIM, J. T., KIM, S., AND PETERSEN, B. K. An interactive visualization platform for deep symbolic regression. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence* (2020).

[114] KIM, S., KIM, J., AND PETERSEN, B. A demonstration platform for deep symbolic regression. Tech. rep., Lawrence Livermore National Lab.(LLNL), Livermore, CA (United States), 2020.

[115] KIM, S., LU, P. Y., MUKHERJEE, S., GILBERT, M., JING, L., ČEPERIĆ, V., AND SOLJAČIĆ, M. Integration of neural network-based symbolic regression in deep learning for scientific discovery. *IEEE Transactions on Neural Networks and Learning Systems* (2020).

[116] KIRA, K., AND RENDELL, L. A. The feature selection problem: Traditional methods and a new algorithm. In *Aaai* (1992), vol. 2, pp. 129–134.

[117] KISHORE, J. K., PATNAIK, L. M., MANI, V., AND AGRAWAL, V. Application of genetic programming for multicategory pattern classification. *IEEE transactions on evolutionary computation 4*, 3 (2000), 242–258.

[118] KLEINKE, K., REINECKE, J., SALFRÁN, D., AND SPIESS, M. Missing data methods. In *Applied Multiple Imputation*. Springer, 2020, pp. 53–83.

[119] KOLLER, D., AND SAHAMI, M. Toward optimal feature selection. Tech. rep., Stanford InfoLab, 1996.

[120] KORDOS, M., AND ŁAPA, K. Multi-objective evolutionary instance selection for regression tasks. *Entropy 20*, 10 (2018), 746.

[121] KORNS, M. F., AND MAY, T. Strong typing, swarm enhancement, and deep learning feature selection in the pursuit of symbolic regression-classification. In *Genetic Programming Theory and Practice XVI*. Springer, 2019, pp. 59–84.

[122] KOTSIANTIS, S. B., ZAHARAKIS, I., AND PINTELAS, P. Supervised machine learning: A review of classification techniques. *Emerging artificial intelligence applications in computer engineering 160* (2007), 3–24.

[123] KOZA, J. R. *Genetic Programming II, Automatic Discovery of Reusable Subprograms*. MIT Press, Cambridge, MA, 1992.

[124] KOZA, J. R. Genetic programming as a means for programming computers by natural selection. *Statistics and computing 4*, 2 (1994), 87–112.

[125] KOZA, J. R. Survey of genetic algorithms and genetic programming. In *Wescon conference record* (1995), WESTERN PERIODICALS COMPANY, pp. 589–594.

[126] KOZA, J. R. Introduction to genetic programming. In *Proceedings of the 9th annual conference companion on Genetic and evolutionary computation* (2007), pp. 3323–3365.

[127] KRAUSE, S., AND POLIKAR, R. An ensemble of classifiers approach for the missing feature problem. In *Int. Joint Conf on Neural Networks* (2003), vol. 1, pp. 553–556.

[128] KRONBERGER, G., DE FRANÇA, F. O., BURLACU, B., HAIDER, C., AND KOMMENDA, M. Shape-constrained symbolic regression–improving extrapolation with prior knowledge. *Evolutionary Computation* (2021), 1–24.

[129] KUNCHEVA, L. I. Editing for the k-nearest neighbors rule by a genetic algorithm. *Pattern Recognition Letters 16*, 8 (1995), 809–814.

[130] KUNCHEVA, L. I. Fitness functions in editing k-nn reference set by genetic algorithms. *Pattern Recognition 30*, 6 (1997), 1041–1049.

[131] KUSHIDA, J.-I., HARA, A., TAKAHAMA, T., AND MIMURA, N. Cartesian ant programming introducing symbiotic relationship between ants and aphids. In *Computational Intelligence and Applications (IWCIA), 2017 IEEE 10th International Workshop on* (2017), IEEE, pp. 115–120.

[132] LA CAVA, W., SINGH, T. R., TAGGART, J., SURI, S., AND MOORE, J. H. Learning concise representations for regression by evolving networks of trees. *arXiv preprint arXiv:1807.00981* (2018).

[133] LAMPLE, G., AND CHARTON, F. Deep learning for symbolic mathematics. *arXiv preprint arXiv:1912.01412* (2019).

[134] LANGLEY, P., ET AL. Selection of relevant features in machine learning. In *Proceedings of the AAAI Fall symposium on relevance* (1994), vol. 184, pp. 245–271.

[135] LE, N., O'NEILL, M., FAGAN, D., AND BRABAZON, A. Social grammatical evolution with imitation learning for real-valued function

estimation. In *Evolutionary Computation (CEC), 2017 IEEE Congress on* (2017), IEEE, pp. 1572–1578.

[136] LEARDI, R. Genetic algorithms in feature selection. In *Genetic algorithms in molecular modeling*. Elsevier, 1996, pp. 67–86.

[137] LENSEN, A., AL-SAHAF, H., ZHANG, M., AND XUE, B. Genetic programming for region detection, feature extraction, feature construction and classification in image data. In *European Conference on Genetic Programming* (2016), Springer, pp. 51–67.

[138] LENSEN, A., XUE, B., AND ZHANG, M. Generating redundant features with unsupervised multi-tree genetic programming. In *European Conference on Genetic Programming* (2018), Springer, pp. 84–100.

[139] LENSEN, A., XUE, B., AND ZHANG, M. Genetic programming for evolving similarity functions for clustering: Representations and analysis. *Evolutionary computation* (2019), 1–29.

[140] LIANG, J., AND XUE, Y. Multi-objective memetic algorithms with tree-based genetic programming and local search for symbolic regression. *Neural Processing Letters* (2021), 1–23.

[141] LIN, W.-C., AND TSAI, C.-F. Missing value imputation: a review and analysis of the literature (2006–2017). *Artificial Intelligence Review 53*, 2 (2020), 1487–1509.

[142] LITTLE, R. J., AND RUBIN, D. B. *Statistical analysis with missing data*, vol. 333. John Wiley & Sons, 2014.

[143] LIU, C.-H., TSAI, C.-F., SUE, K.-L., AND HUANG, M.-W. The feature selection effect on missing value imputation of medical datasets. *Applied Sciences 10*, 7 (2020), 2344.

[144] LIU, H., AND MOTODA, H. On issues of instance selection. *Data Mining and Knowledge Discovery 6*, 2 (2002), 115–130.

[145] LIU, H., AND YU, L. Toward integrating feature selection algorithms for classification and clustering. *IEEE Transactions on knowledge and data engineering 17*, 4 (2005), 491–502.

[146] LIU, X., WANG, H., YE, W., AND XING, E. P. Sparse variable selection on high dimensional heterogeneous data with tree structured responses. *arXiv preprint arXiv:1711.08265* (2017).

[147] LOBATO, F., SALES, C., ARAUJO, I., TADAIESKY, V., DIAS, L., RAMOS, L., AND SANTANA, A. Multi-objective genetic algorithm for missing data imputation. *Pattern Recognition Letters 68* (2015), 126–131.

[148] LOBATO, F. M., TADAIESKY, V. W., ARAÚJO, I. M., AND DE SANTANA, Á. L. An evolutionary missing data imputation method for pattern classification. In *Proceedings of the Companion Publication of the 2015 Annual Conference on Genetic and Evolutionary Computation* (2015), ACM, pp. 1013–1019.

[149] LOU, Q., AND OBRADOVIC, Z. Margin-based feature selection in incomplete data. In *Twenty-Sixth AAAI Conference on Artificial Intelligence* (2012), pp. 1040–1046.

[150] LOVEARD, T., AND CIESIELSKI, V. Representing classification problems in genetic programming. In *Evolutionary Computation, 2001. Proceedings of the 2001 Congress on* (2001), vol. 2, IEEE, pp. 1070–1077.

[151] MALARVIZHI, M. R., AND THANAMANI, A. S. K-nearest neighbor in missing data imputation. *International Journal of Engineering Research and Development 5*, 1 (2012), 5–7.

[152] MARILL, T., AND GREEN, D. On the effectiveness of receptors in recognition systems. *IEEE transactions on Information Theory 9*, 1 (1963), 11–17.

[153] MASSEY JR, F. J. The kolmogorov-smirnov test for goodness of fit. *Journal of the American statistical Association 46*, 253 (1951), 68–78.

[154] MCCONAGHY, T. Latent variable symbolic regression for high-dimensional inputs. In *Genetic Programming Theory and Practice VII*. Springer, 2010, pp. 103–118.

[155] MCCONAGHY, T. Ffx: Fast, scalable, deterministic symbolic regression technology. In *Genetic Programming Theory and Practice IX*. Springer, 2011, pp. 235–260.

[156] MCKAY, R. I., HOAI, N. X., WHIGHAM, P. A., SHAN, Y., AND O'NEILL, M. Grammar-based genetic programming: a survey. *Genetic Programming and Evolvable Machines 11*, 3-4 (2010), 365–396.

[157] MCPHEE, N. F., POLI, R., AND LANGDON, W. B. Field guide to genetic programming.

[158] MICHALSKI, R. S., CARBONELL, J. G., AND MITCHELL, T. M. *Machine learning: An artificial intelligence approach*. Springer Science & Business Media, 2013.

[159] MOLLINEDA, R. A., FERRI, F. J., AND VIDAL, E. An efficient prototype merging strategy for the condensed 1-nn rule through class-conditional hierarchical clustering. *Pattern Recognition 35*, 12 (2002), 2771–2782.

[160] MONTANA, D. J. Strongly typed genetic programming. *Evolutionary computation 3*, 2 (1995), 199–230.

[161] MOORE, R. E. *Interval analysis*, vol. 4. Prentice-Hall Englewood Cliffs, 1966.

[162] MUNOZ, L., SILVA, S., AND TRUJILLO, L. M3gp–multiclass classification with gp. In *European Conference on Genetic Programming* (2015), Springer, pp. 78–91.

[163] MUÑOZ, L., TRUJILLO, L., AND SILVA, S. Transfer learning in constructive induction with genetic programming. *Genetic Programming and Evolvable Machines* (2019), 1–41.

[164] NAG, K., AND PAL, N. R. Genetic programming for classification and feature selection. In *Evolutionary and Swarm Intelligence Algorithms*. Springer, 2019, pp. 119–141.

[165] NAM, J., AND KIM, S. Heterogeneous defect prediction. In *Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering* (New York, NY, USA, 2015), ESEC/FSE 2015, Association for Computing Machinery, p. 508–519.

[166] NARENDRA, P. M., AND FUKUNAGA, K. A branch and bound algorithm for feature subset selection. *IEEE Transactions on computers*, 9 (1977), 917–922.

[167] NAUCK, D., AND KRUSE, R. Learning in neuro-fuzzy systems with symbolic attributes and missing values. In *Neural Information Processing, 1999. Proceedings. ICONIP'99. 6th International Conference on* (1999), vol. 1, IEEE, pp. 142–147.

[168] NESHATIAN, K., ZHANG, M., AND ANDREAE, P. A filter approach to multiple feature construction for symbolic learning classifiers using genetic programming. *IEEE Transactions on Evolutionary Computation 16*, 5 (2012), 645–661.

[169] OLVERA-LÓPEZ, J. A., CARRASCO-OCHOA, J. A., MARTÍNEZ-TRINIDAD, J. F., AND KITTLER, J. A review of instance selection methods. *Artificial Intelligence Review 34*, 2 (2010), 133–143.

[170] O'NEILL, D., AL-SAHAF, H., XUE, B., AND ZHANG, M. Common subtrees in related problems: A novel transfer learning approach for genetic programming. In *Evolutionary Computation (CEC), 2017 IEEE Congress on* (2017), IEEE, pp. 1287–1294.

[171] OWEN, C. A., DICK, G., AND WHIGHAM, P. A. Feature standardisation in symbolic regression. In *Australasian Joint Conference on Artificial Intelligence* (2018), Springer, pp. 565–576.

[172] OWEN, C. A., DICK, G., AND WHIGHAM, P. A. Characterizing genetic programming error through extended bias and variance decomposition. *IEEE Transactions on Evolutionary Computation 24*, 6 (2020), 1164–1176.

[173] O'SULLIVAN, J., AND RYAN, C. An investigation into the use of different search strategies with grammatical evolution. In *European Conference on Genetic Programming* (2002), Springer, pp. 268–277.

[174] PAN, S. J., YANG, Q., ET AL. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering 22*, 10 (2010), 1345–1359.

[175] PARDOE, D., AND STONE, P. Boosting for regression transfer. In *Proceedings of the 27th International Conference on International Conference on Machine Learning* (2010), Omnipress, pp. 863–870.

[176] PATIL, D. V., AND BICHKAR, R. Multiple imputation of missing data with genetic algorithm based techniques. *IJCA Special Issue on" Evolutionary Computation for Optimization Techniques* (2010), 74–78.

[177] PATRICIA, N., AND CAPUTO, B. Learning to learn, from transfer learning to domain adaptation: A unifying perspective. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2014), pp. 1442–1449.

[178] PENNACHIN, C., LOOKS, M., AND DE VASCONCELOS, J. Improved time series prediction and symbolic regression with affine arithmetic. In *Genetic Programming Theory and Practice IX*. Springer, 2011, pp. 97–112.

[179] PETKOVIC, M., PETKOVIĆ, M., PETKOVIC, M. S., AND PETKOVIC, L. D. *Complex interval arithmetic and its applications*, vol. 105. John Wiley & Sons, 1998.

[180] PHILLIPS, T., ZHANG, M., AND XUE, B. Genetic programming for evolving programs with recursive structures. In *Evolutionary Computation (CEC), 2016 IEEE Congress on* (2016), IEEE, pp. 5044–5051.

[181] PHILLIPS, T., ZHANG, M., AND XUE, B. Genetic programming for solving common and domain-independent generic recursive problems. In *Evolutionary Computation (CEC), 2017 IEEE Congress on* (2017), IEEE, pp. 1279–1286.

[182] PIRES, A., AND BRANCO, J. High dimensionality: The latest challenge to data analysis. *arXiv preprint arXiv:1902.04679* (2019).

[183] POLI, R., LANGDON, W. B., MCPHEE, N. F., AND KOZA, J. R. *A field guide to genetic programming*. Lulu. com, 2008.

[184] PORNPRASERTMANIT, S., MILLER, P., SCHOEMANN, A., QUICK, C., JORGENSEN, T., AND PORNPRASERTMANIT, M. S. Package 'simsem', 2016.

[185] PRIYA, R. D., AND KUPPUSWAMI, S. A genetic algorithm based approach for imputing missing discrete attribute values in databases. *WSEAS Transactions on Information Science and Applications 9*, 6 (2012), 169–178.

[186] PUDIL, P., NOVOVIČOVÁ, J., AND KITTLER, J. Floating search methods in feature selection. *Pattern recognition letters 15*, 11 (1994), 1119–1125.

[187] QIAN, W., AND SHU, W. Mutual information criterion for feature selection from incomplete data. *Neurocomputing 168* (2015), 210–220.

[188] QIU, J., WU, Q., DING, G., XU, Y., AND FENG, S. A survey of machine learning for big data processing. *EURASIP Journal on Advances in Signal Processing 2016*, 1 (2016), 67.

[189] QUADE, M., ABEL, M., SHAFI, K., NIVEN, R. K., AND NOACK, B. R. Prediction of dynamical systems by symbolic regression. *Physical Review E 94*, 1 (2016), 012214.

[190] QUATTONI, A., COLLINS, M., AND DARRELL, T. Transfer learning for image classification with sparse prototype representations. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on* (2008), IEEE, pp. 1–8.

[191] QUINLAN, J. R. *C4.5: programs for machine learning*. Elsevier, 2014.

[192] RAD, H. I., FENG, J., AND IBA, H. Gp-rvm: Genetic programing-based symbolic regression using relevance vector machine. *arXiv preprint arXiv:1806.02502* (2018).

[193] RAINA, R., BATTLE, A., LEE, H., PACKER, B., AND NG, A. Y. Self-taught learning: transfer learning from unlabeled data. In *Proceedings of the 24th international conference on Machine learning* (2007), ACM, pp. 759–766.

[194] RASHID, W., AND GUPTA, M. K. A perspective of missing value imputation approaches. In *Advances in Computational Intelligence and Communication Technology*. Springer, pp. 307–315.

[195] RENGGLI, C., RIMANIC, L., GÜREL, N. M., KARLAŠ, B., WU, W., AND ZHANG, C. A data quality-driven view of mlops. *arXiv preprint arXiv:2102.07750* (2021).

[196] RITTER, G., WOODRUFF, H., LOWRY, S., AND ISENHOUR, T. An algorithm for a selective nearest neighbor decision rule (corresp.). *IEEE Transactions on Information Theory 21*, 6 (1975), 665–669.

[197] ROBERT, C. Machine learning, a probabilistic perspective, 2014.

[198] RUBIN, D. B. Inference and missing data. *Biometrika 63*, 3 (1976), 581–592.

[199] RUBIN, D. B. *Multiple imputation for nonresponse in surveys*, vol. 81. John Wiley & Sons, 2004.

[200] RUSSELL, S., AND NORVIG, P. Artificial intelligence: a modern approach.

[201] RUSSELL, S. J., AND NORVIG, P. *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited,, 2016.

[202] SALLEH, M. N. M., AND SAMAT, N. A. An imputation for missing data features based on fuzzy swarm approach in heart disease classification. In *International Conference in Swarm Intelligence* (2017), Springer, pp. 285–292.

[203] SAMAT, N. A., AND SALLEH, M. N. M. A study of data imputation using fuzzy c-means with particle swarm optimization. In *International Conference on Soft Computing and Data Mining* (2016), Springer, pp. 91–100.

[204] SAMUEL, A. L. Some studies in machine learning using the game of checkers. *IBM Journal of research and development 3*, 3 (1959), 210–229.

[205] SÁNCHEZ, L. Interval-valued ga-p algorithms. *IEEE Transactions on Evolutionary Computation 4*, 1 (2000), 64–72.

[206] SCHAFER, J. L. *Analysis of incomplete multivariate data*. Chapman and Hall/CRC, 1997.

[207] SCHAFER, J. L. Multiple imputation: a primer. *Statistical methods in medical research 8*, 1 (1999), 3–15.

[208] SCHAPIRE, R. E. The boosting approach to machine learning: An overview. In *Nonlinear estimation and classification*. Springer, 2003, pp. 149–171.

[209] SEFIDIAN, A. M., AND DANESHPOUR, N. Missing value imputation using a novel grey based fuzzy c-means, mutual information based feature selection, and regression model. *Expert Systems with Applications 115* (2019), 68–94.

[210] SHERRY, D., VEERAMACHANENI, K., MCDERMOTT, J., AND OREILLY, U.-M. Flex-gp: Genetic programming on the cloud. In *European Conference on the Applications of Evolutionary Computation* (2012), Springer, pp. 477–486.

[211] SIEDLECKI, W., AND SKLANSKY, J. A note on genetic algorithms for large-scale feature selection. In *Handbook of Pattern Recognition and Computer Vision*. World Scientific, 1993, pp. 88–107.

[212] SIPPER, M., AND MOORE, J. H. Symbolic-regression boosting. *Genetic Programming and Evolvable Machines* (2021), 1–25.

[213] STEPHENS, T. gplearn: Genetic programming in python, with a scikit-learn inspired api. `https://gplearn.readthedocs.io/en/stable/`. Accessed: 2020-05-06.

[214] STIJVEN, S., MINNEBO, W., AND VLADISLAVLEVA, K. Separating the wheat from the chaff: on feature selection and feature importance in regression random forests and symbolic regression. In *Proceedings of the 13th annual conference companion on Genetic and evolutionary computation* (2011), ACM, pp. 623–630.

[215] SUTTON, R. S., BARTO, A. G., ET AL. *Reinforcement learning: An introduction*. MIT press, 1998.

[216] TAKAHASHI, M., AND ITO, T. Multiple imputation of turnover in edinet data: Toward the improvement of imputation for the economic census. *Work Session on Statistical Data Editing, UNECE* (2012), 24–26.

[217] TAMBURRI, D. A. Sustainable mlops: Trends and challenges. In *2020 22nd International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC)* (2020), IEEE, pp. 17–23.

[218] TAYLOR, M. E., AND STONE, P. Transfer learning for reinforcement learning domains: A survey. *Journal of Machine Learning Research 10*, Jul (2009), 1633–1685.

[219] THOMAS, T., AND RAJABI, E. A systematic review of machine learning-based missing value imputation techniques. *Data Technologies and Applications* (2021).

[220] TIBSHIRANI, R. Regression shrinkage and selection via the lasso: a retrospective. *Journal of the Royal Statistical Society: Series B (Statistical Methodology) 73*, 3 (2011), 273–282.

[221] TRAN, B. *Evolutionary Computation for Feature Manipulation in Classification on High-dimensional Data*. PhD thesis, Victoria University of Wellington, 2018.

[222] TRAN, C. T. *Evolutionary Machine Learning for Classification with Incomplete Data*. PhD thesis, Victoria University of Wellington, 2018.

[223] TRAN, C. T., ZHANG, M., AND ANDREAE, P. Multiple imputation for missing data using genetic programming. In *Proceedings of the 2015 annual conference on genetic and evolutionary computation* (2015), ACM, pp. 583–590.

[224] TRAN, C. T., ZHANG, M., AND ANDREAE, P. Directly evolving classifiers for missing data using genetic programming. In *IEEE*

Congress on Evolutionary Computation (CEC) (2016), IEEE, pp. 5278–5285.

[225] TRAN, C. T., ZHANG, M., AND ANDREAE, P. A genetic programming-based imputation method for classification with missing data. In *European Conference on Genetic Programming* (2016), Springer, pp. 149–163.

[226] TRAN, C. T., ZHANG, M., ANDREAE, P., AND XUE, B. Directly constructing multiple features for classification with missing data using genetic programming with interval functions. In *Proceedings on Genetic and Evolutionary Computation Conference Companion* (2016), pp. 69–70.

[227] TRAN, C. T., ZHANG, M., ANDREAE, P., AND XUE, B. Improving performance for classification with incomplete data using wrapper-based feature selection. *Evolutionary Intelligence 9*, 3 (2016), 81–94.

[228] TRAN, C. T., ZHANG, M., ANDREAE, P., AND XUE, B. A wrapper feature selection approach to classification with missing data. In *European Conference on the Applications of Evolutionary Computation* (2016), Springer, pp. 685–700.

[229] TRAN, C. T., ZHANG, M., ANDREAE, P., AND XUE, B. Bagging and feature selection for classification with incomplete data. In *European Conference on the Applications of Evolutionary Computation* (2017), Springer, pp. 471–486.

[230] TRAN, C. T., ZHANG, M., ANDREAE, P., AND XUE, B. Multiple imputation and genetic programming for classification with incomplete data. In *Proceedings of the Genetic and Evolutionary Computation Conference* (2017), ACM, pp. 521–528.

[231] TRAN, C. T., ZHANG, M., ANDREAE, P., XUE, B., AND BUI, L. T. Improving performance of classification on incomplete data using

feature selection and clustering. *Applied Soft Computing 73* (2018), 848–861.

[232] TRAN, C. T., ZHANG, M., XUE, B., AND ANDREAE, P. Genetic programming with interval functions and ensemble learning for classification with incomplete data. In *Australasian Joint Conference on Artificial Intelligence* (2018), Springer, pp. 577–589.

[233] TRUJILLO, L., MUÑOZ, L., LÓPEZ, U., AND HERNÁNDEZ, D. E. Untapped potential of genetic programming: Transfer learning and outlier removal. In *Genetic Programming Theory and Practice XVI*. Springer, 2019, pp. 193–207.

[234] TSAI, C.-F., AND CHANG, F.-Y. Combining instance selection for better missing value imputation. *Journal of Systems and Software 122* (2016), 63–71.

[235] UDRESCU, S.-M., AND TEGMARK, M. Ai feynman: A physics-inspired method for symbolic regression. *Science Advances 6*, 16 (2020), eaay2631.

[236] VAN DER LOO, M. simputation: Simple imputation. *R package version 0.2 2* (2017).

[237] VANNESCHI, L. An introduction to geometric semantic genetic programming. In *NEO 2015*. Springer, 2017, pp. 3–42.

[238] VANNESCHI, L., BAKUROV, I., AND CASTELLI, M. An initialization technique for geometric semantic gp based on demes evolution and despeciation. In *Evolutionary Computation (CEC), 2017 IEEE Congress on* (2017), IEEE, pp. 113–120.

[239] VANNESCHI, L., AND GALVAO, B. A parallel and distributed semantic genetic programming system. In *Evolutionary Computation (CEC), 2017 IEEE Congress on* (2017), IEEE, pp. 121–128.

[240] VANNESCHI, L., SILVA, S., CASTELLI, M., AND MANZONI, L. Geometric semantic genetic programming for real life applications. In *Genetic programming theory and practice xi*. Springer, 2014, pp. 191–209.

[241] VANSCHOREN, J., VAN RIJN, J. N., BISCHL, B., AND TORGO, L. Openml: networked science in machine learning. *ACM SIGKDD Explorations Newsletter 15*, 2 (2014), 49–60.

[242] VEENMAN, C. J., AND REINDERS, M. J. The nearest subclass classifier: A compromise between the nearest mean and nearest neighbor classifier. *IEEE Transactions on Pattern Analysis and Machine Intelligence 27*, 9 (2005), 1417–1429.

[243] VEENMAN, C. J., REINDERS, M. J. T., AND BACKER, E. A maximum variance cluster algorithm. *IEEE Transactions on pattern analysis and machine intelligence 24*, 9 (2002), 1273–1280.

[244] VIEGAS, F., ROCHA, L., GONÇALVES, M., MOURÃO, F., SÁ, G., SALLES, T., ANDRADE, G., AND SANDIN, I. A genetic programming approach for feature selection in highly dimensional skewed data. *Neurocomputing 273* (2018), 554–569.

[245] VIRGOLIN, M., ALDERLIESTEN, T., WITTEVEEN, C., AND BOSMAN, P. A. Model-based genetic programming with gomea for symbolic regression of small expressions. *arXiv preprint arXiv:1904.02050* (2019).

[246] VLADISLAVLEVA, E., SMITS, G., AND DEN HERTOG, D. On the importance of data balancing for symbolic regression. *IEEE Transactions on Evolutionary Computation 14*, 2 (2010), 252–277.

[247] VLADISLAVLEVA, E. J., SMITS, G. F., AND DEN HERTOG, D. Order of nonlinearity as a complexity measure for models generated by

symbolic regression via pareto genetic programming. *IEEE Transactions on Evolutionary Computation 13*, 2 (2008), 333–349.

[248] WANG, G., LU, J., CHOI, K.-S., AND ZHANG, G. A transfer-based additive ls-svm classifier for handling missing data. *IEEE transactions on cybernetics* (2018).

[249] WEBB, G. I. The problem of missing values in decision tree grafting. In *Australian Joint Conference on Artificial Intelligence* (1998), Springer, pp. 273–283.

[250] WEISS, K., KHOSHGOFTAAR, T. M., AND WANG, D. A survey of transfer learning. *Journal of Big data 3*, 1 (2016), 9.

[251] WHIGHAM, P. A. Grammatically-based genetic programming. In *Proceedings of the Workshop on Genetic Programming: From Theory to Real-World Applications* (Tahoe City, California, USA, 9 July 1995), J. P. Rosca, Ed., pp. 33–41.

[252] WHITNEY, A. W. A direct method of nonparametric measurement selection. *IEEE Transactions on Computers 100*, 9 (1971), 1100–1103.

[253] WILLIS, M.-J., HIDEN, H. G., MARENBACH, P., MCKAY, B., AND MONTAGUE, G. A. Genetic programming: An introduction and survey of applications. In *Genetic Algorithms in Engineering Systems: Innovations and Applications, 1997. GALESIA 97. Second International Conference On (Conf. Publ. No. 446)* (1997), IET, pp. 314–319.

[254] WILSON, D. R., AND MARTINEZ, T. R. Reduction techniques for instance-based learning algorithms. *Machine learning 38*, 3 (2000), 257–286.

[255] WU, X., KUMAR, V., QUINLAN, J. R., GHOSH, J., YANG, Q., MOTODA, H., MCLACHLAN, G. J., NG, A., LIU, B., PHILIP, S. Y.,

ET AL. Top 10 algorithms in data mining. *Knowledge and information systems 14*, 1 (2008), 1–37.

[256] XUE, B., AND ZHANG, M. Evolutionary computation for feature manipulation: Key challenges and future directions. In *Evolutionary Computation (CEC), 2016 IEEE Congress on* (2016), IEEE, pp. 3061–3067.

[257] XUE, B., AND ZHANG, M. Evolutionary feature manipulation in data mining/big data. *ACM SIGEVOlution 10*, 1 (2017), 4–11.

[258] XUE, B., ZHANG, M., AND BROWNE, W. N. Particle swarm optimisation for feature selection in classification: Novel initialisation and updating mechanisms. *Applied Soft Computing 18* (2014), 261–276.

[259] XUE, B., ZHANG, M., BROWNE, W. N., AND YAO, X. A survey on evolutionary computation approaches to feature selection. *IEEE Transactions on Evolutionary Computation 20*, 4 (2016), 606–626.

[260] ZELINKA, I., OPLATKOVA, Z., AND NOLLE, L. Analytic programming–symbolic regression by means of arbitrary evolutionary algorithms. *Int. J. of Simulation, Systems, Science and Technology 6*, 9 (2005), 44–56.

[261] ZHANG, J., ZHAN, Z.-H., LIN, Y., CHEN, N., GONG, Y.-J., ZHONG, J.-H., CHUNG, H. S., LI, Y., AND SHI, Y.-H. Evolutionary computation meets machine learning: A survey. *IEEE Computational Intelligence Magazine 6*, 4 (2011), 68–75.

[262] ZHANG, L. Transfer adaptation learning: A decade survey. *arXiv preprint arXiv:1903.04687* (2019).

[263] ZHAO, P., AND HOI, S. C. Otl: A framework of online transfer learning.

[264] Zhao, P., Hoi, S. C., Wang, J., and Li, B. Online transfer learning. *Artificial Intelligence 216* (2014), 76–102.

[265] Zhao, Y. *R and data mining: Examples and case studies.* Academic Press, 2012.

[266] Zheng, W., Zhu, X., Zhu, Y., and Zhang, S. Robust feature selection on incomplete data. In *IJCAI* (2018), pp. 3191–3197.

[267] Zhenzhong, W., Jiang, M., Xing, G., Liang, F., Weizhen, H., and Tan, K. C. Evolutionary dynamic multi-objective optimization via regression transfer learning. In *2019 IEEE Symposium Series on Computational Intelligence (SSCI)* (2019), IEEE, pp. 2375–2381.

[268] Zhong, J., Ong, Y.-S., and Cai, W. Self-learning gene expression programming. *IEEE Transactions on Evolutionary Computation 20*, 1 (2016), 65–80.

[269] Zhou, X., Chai, H., Zhao, H., Luo, C.-H., and Yang, Y. Imputing missing rna-seq data from dna methylation by using transfer learning based-deep neural network. *bioRxiv* (2019), 803692.

[270] Zhu, X., et al. Comparison of four methods for handling missing data in longitudinal data analysis through a simulation study. *Open Journal of Statistics 4*, 11 (2014), 933.

[271] Zhu, X., and Goldberg, A. B. Introduction to semi-supervised learning. *Synthesis lectures on artificial intelligence and machine learning 3*, 1 (2009), 1–130.

[272] Zuo, H., Zhang, G., Pedrycz, W., Behbood, V., and Lu, J. Fuzzy regression transfer learning in takagi–sugeno fuzzy models. *IEEE Transactions on Fuzzy Systems 25*, 6 (2016), 1795–1807.