# Feature Selection to Improve Generalisation of Genetic Programming for High-Dimensional Symbolic Regression

Qi Chen, *Student Member, IEEE,* Mengjie Zhang, *Senior Member, IEEE,* and Bing Xue, *Member, IEEE*

***Abstract*—When learning from high-dimensional data for symbolic regression, genetic programming typically could not generalise well. Feature selection, as a data preprocessing method, can potentially contribute not only to improving the efficiency of learning algorithms but also to enhancing the generalisation ability. However, in genetic programming for high-dimensional symbolic regression, feature selection before learning is seldom considered. In this work, we propose a new feature selection method based on permutation to select features for high-dimensional symbolic regression using genetic programming. A set of experiments has been conducted to investigate the performance of the proposed method on the generalisation of genetic programming for high-dimensional symbolic regression. The regression results confirm the superior performance of the proposed method over the other examined feature selection methods. Further analysis indicates that the models evolved by the proposed method are more likely to contain only the truly relevant features and have better interpretability.**

***Index Terms*—Genetic Programming, Feature Selection, Generalisation, Symbolic Regression**

## I. Introduction

OVER the past decade, with the development of data collection techniques, the dimensionality of data (i.e. the number of features) has become increasingly higher in real-world machine learning tasks. The high dimensionality of the data often decreases the ability of learning algorithms to extract useful information from original data with all features, since it is prone to learn from irrelevant features as well as noise. Furthermore, there are some other problems when learning from high-dimensional data, such as the curse of dimensionality [1], higher risk of overfitting, and more expensive computational cost.

Feature selection is a process of identifying a subset of relevant features that are necessary to describe the output variables. When learning from high-dimensional data, feature selection is desired. Much work has already been devoted to feature selection [2], [3], [4]. However, most of them are proposed for classification tasks. Genetic Programming (GP) [5] for symbolic regression (SR) seldom considers feature selection, and it is even rare when GP tackles high-dimensional symbolic regression tasks. The underlying reason is that GP has the built-in feature selection ability when exploring the feature space to create a SR model. However, the built-in feature selection ability of GP is typically not strong enough for high-dimensional regression tasks.

Generalisation is a kind of ability with which the learnt model can have good prediction performance on unseen data. Generally, it is measured by the prediction error of the model over a set of unseen test samples. While generalisation has been considered as an important aspect in many fields in machine learning for a long time [6], [7], it has not received much attention in GP for symbolic regression. Since the publication of Kushchu's work on the generalisation of GP in 2002 [8], growing attention has been devoted to promoting the generalisation of GP for symbolic regression [9], [10], [11]. However, contributions to the generalisation in GP are far behind when compared with the fast development of GP for symbolic regression. Thus, generalisation remains an open issue on GP [12].

When learning an unknown model for symbolic regression in a high-dimension feature space, the generalisation capability of GP would decrease. Feature selection, as a data preprocessing method, can remove noise and irrelevant features. It has the potential to reduce the risk of overfitting, thus promote the generalisation ability of GP. However, not much work on feature selection to improve the generalisation of GP has been proposed for high-dimensional regression to date.

*Goals:* The goal of this work is to propose a new feature selection method to improve the generalisation ability of GP for symbolic regression, specifically when learning from high-dimensional data. This work will cover three research questions as follows:

- how feature selection can influence the learning ability of GP regarding the training performance,
- whether feature selection can enhance the generalisation ability of GP for high-dimensional regression tasks, and
- whether the new feature selection method can select the truly relevant features for high-dimensional symbolic regression.

## II. Background

### A. Genetic Programming for Symbolic Regression

Symbolic regression, as a kind of regression analysis, is a model identification process [5]. The task is to identify the relationship between the input variables and the response variables in a dataset, and express their relationship in symbolic models. The main strength of symbolic regression is that it neither requires any pre-specified form and size of the model nor needs to assume any distribution of the data. Symbolic regression is named to emphasise that the identification process can find a symbolic description. This is the sharp difference from classical regression methods aiming to find a set of coefficients of a pre-defined model.

The symbolic nature of solutions and requiring no prior knowledge make GP very suitable for symbolic regression. Many different GP variants have been developed to improve symbolic regression in its efficiency and effectiveness in previous work [13], [14], [15], [16], [17], [18], [19]. Despite the many success stories in GP for symbolic regression, improving its generalisation ability is still an open issue as it tends to overfit.

## *B. Feature Selection*

Feature selection removes irrelevant and redundant features. Thus it brings a dimensional reduction in the dataset. Feature selection can not only make the learning process more efficient but also enhance the learning performance [20], [21]. Techniques for feature selection can be divided into three groups: filter methods, wrapper methods and embedded methods [20], [22]. Filter methods [23], [24] select a subset of features based on kinds of criteria such as mutual information. Yu et al. [24] proposed a filter feature selection method based on a measure named predominant correlation. In their method, the predominant features are identified and selected using the symmetrical uncertainty (SU) measure. Wrapper methods [25], [26], [27] use a learning algorithm as a black-box and select subsets of features based on the performance of the learning algorithm. Embedded methods [28], [29], [30] incorporate the feature selection process within the learning process.

*1) Genetic Programming for Feature Selection:* As the features appearing in the evolved individual can be treated as a set of selected features, thus GP is considered to have the built-in feature selection ability. The built-in ability of GP in detecting important features by exploring the feature space has made it a valuable method for feature selection. A number of different GP-based feature selection methods have been proposed in the literature [3], [31], [32], [33], [34].

Neshatian et al. [3] developed a Pareto GP for feature selection in classification tasks. They designed a function to measure the relevance of subsets of features. A Pareto front archive was maintained, which consists of non-dominated subsets of features having a low cardinality (i.e. the number of features the subset contains) and high relevance. They also adopted methods to avoid bloat and overfitting to allow GP to explore large subsets of features. The experimental results show that the feature selection method can improve the classification accuracy while decreasing the complexity of the evolved classifiers. However, the proposed method might have some limitations when the cardinality of desired best subset of features is high. Muni et al. [31] proposed a GP-based feature selection method to address the skewed/unbalanced high-dimensional classification tasks, which combined multiple most commonly used feature selection metrics. The results indicate that the method can bring dimension reduction as well as increase the classification accuracy. Moore et al. [35] introduced a nonlinear gene-gene (feature-feature) interactions measure based on information entropy into their Pareto GP system for genetic analysis of diseases. The many-objectives GP system uses three objectives (classification accuracy, model size and the interaction measure) to guide the search towards models including features that are risk factors for the disease. The GP system is claimed to be able to find accurate models despite the size and complexity of the feature space.

We recently proposed a method namely *genetic programming with feature selection* (GPWFS) in [36]. GPWFS is a two-stage feature selection method for high-dimensional symbolic regression. It splits the evolutionary process of GP into two phases by a parameter $G_f$. The major task of the first phase is feature selection. On each generation of this phase, all the distinct features appearing in the top $\beta$ percent individuals are collected, since these features are considered to be candidates of important features. At the end of the first phase, a set of potentially important features $F_c$ is formed. The second phase is the standard evolutionary process on a population of reinitialised individuals. On the first generation of the second phase, GPWFS reinitialises the population by keeping the top $\beta$ percent individuals while replacing the rest. The replacement will take the form of an equal number of randomly generated individuals using a new terminal set formed by the set of selected features $F_c$. The effectiveness of GPWFS on enhancing the generalisation of GP was investigated and confirmed in [36]. However, GPWFS needs to tune two key parameters $G_f$ and $\beta$, which are problem dependent and sensitive to the parameter settings of GP, such as the population size and the total number of generations. More details can be seen in [36].

*2) Random Forests and Decision Tree-based Methods for Feature Selection:* Random forests (RF) [37] is an ensemble learning algorithm, which constructs a forest of decision trees and provides solutions for classification and regression tasks. RF uses a concept of permutation variable importance over the out-of-bag (OOB) examples. The OOB examples are observations that are not exposed to the decision tree. For each variable in the decision tree, the values of the variable in the OOB examples are permuted (randomly rearranged within the OOB examples). Then each permuted example is passed down the tree. The total increase of the regression error is defined to be the variable importance. The bigger value means the variable is more important.

The family of decision trees like ID3 [38], C4.5 [39], C5.0 [40] can also be applied to feature selection. In these decision trees, the variable importance is computed in two ways. The first one is calculating the percentage of training examples falling into all the terminal nodes after the split of the variable. Generally, it is biased to features in the early split nodes. The other one tries to avoid this bias by taking the percentage of splits that a variable is used into consideration as well when assigning importance score.

## *C. Generalisation in GP for Symbolic Regression*

Generalisation is a kind of ability with which a learnt model can obtain good prediction results on unseen test data. For supervised learning problems, generalisation is one of the most important performance measures for learning algorithms, since it proves the existence of effective learning. Overfitting is the contrary concept of generalisation. Overfitting occurs when the learnt model has a low training error but a high test error. In contrast, models with good generalisation capability can perform well not only on the training data but more importantly on unseen data. Learning unknown models from a high-dimensional feature space runs the risk of overfitting and leads to poor generalisation.

Generalisation has been treated as an important issue in many fields of machine learning for a long time [6]. In GP for classification, the generalisation ability of GP programs has been investigated for a long time. Much work has already been devoted to improving the generalisation in GP for classification [41], [42], [43], but not much done for generalisation in GP for symbolic regression in the past. For a long time, symbolic regression has been treated as an optimisation task and used all the available data for evolution. In recent years, growing attention has been devoted to increasing generalisation in GP for symbolic regression [9], [17], [18], [44]. Uy et al. [9] developed new semantic-aware operators to maintain locality thus yielding a better generalisation of GP. Haeri et al. [17] proposed variance-based layered learning GP, which decompose the evolutionary process into several hierarchical layers. From lower to higher, these layers were trained using different training sets, from less to more complex. The complexity of the training sets is measured by the variance of the output values. This measure is also applied to evaluate the complexity of candidate models. The experimental results show that the layered GP can enhance the generalisation ability of GP while reducing the model complexity. Mousavi et al. [18] presented a multiobjective GP (MOGP) method to enhance the generalisation by controlling the first order derivative of GP models. In MOGP, in addition to purchasing the lower training error, the first order derivative of the candidate model, which measures the model complexity, is considered to be the other objective. The experimental results show that MOGP has better generalisation gain than standard GP.

In summary, much work has been devoted to GP-based feature selection to improve the classification performance. However, for regression tasks, especially for high-dimensional regression, more attention is deserved. There is no existing work in this field and feature selection is desired to improve the regression performance and generalisation of GP when the dimensionality is high.

## III. The Proposed Feature Selection Method

This work proposes a new feature selection method, which is named *genetic programming with permutation importance* (GPPI). It is based on our previous research, i.e. GPWFS [36]. The two methods share the same assumption that GP can explore the search space to detect important features automatically. We assume that relevant features appear in highly fit GP individuals, even though not all these features are relevant. Thus, they can form a candidate set of important features for feature selection. However, GPPI has a significantly different way to determine the importance of the features from GPWFS. Moreover, while GPWFS is a GP method for regression with an embedded feature selection phase, GPPI is a feature selection method for preprocessing the data for GP for symbolic regression (GPSR).

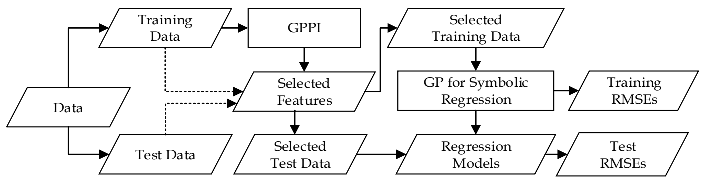


Fig. 1. Data Flow Diagram for GP-GPPI.

### A. The Overall Structure

Fig. 1 shows the data flow diagram for the new GPSR system namely — GP-GPPI, which includes a feature selection component GPPI. The training process of the GP system consists of two sequential phases. Firstly, GPPI is applied to the training data to select a subset of important features. Then standard GP evolves regression models on the selected training data with only the selected features. The key component of the new GPSR system is GPPI. GPPI improves GPWFS in two aspects. The first is the definition of the *good* individuals, which are the source of important features. GPWFS treats certain top individuals with the highest fitness values as the *good* individuals. Instead, GPPI collects the best-of-run individuals from a number of GP runs. Compared to their counterpart in GPWFS, the *good* individuals in GPPI are sufficiently evolved and contain important features by utilising the natural feature selection ability of GP. The second aspect lies in the determination of important features. GPWFS collects all the distinct features present in the good individuals as the relatively important features. GPPI computes a quantitative importance value of these distinct features. Feature selection produces in a metric of importance values. The details of GPPI will be presented in Section III.C and III.D.

### B. Fitness Function

In this work, the fitness function in both GPPI and GPSR is *Normalised Root Mean Square Error* (NRMSE), which evaluates the performance of individuals for feature selection and regression. The definition of $NRMSE$ is given in Equation (1).

$$NRMSE = \frac{RMSE}{Y_{max} - Y_{min}} \tag{1}$$

where the term $(Y_{max} - Y_{min})$ is the range of the target variable and $RMSE$ is the root mean square error. The definition of $RMSE$ is shown in Equation (2).

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(f(X_i) - Y_i)^2} \tag{2}$$

where $N$ is the number of instances, $f(X_i)$ is the output of the model and $Y_i$ is the target output.

### C. Permutation Feature Importance

As mentioned above, not all the features appearing in the highly fit individuals are important. Thus, the crucial component of the feature selection method is a quantitative measure of feature importance, which can tell the difference between the appearing features. Feature importance can be defined as the correlation between the feature and the target

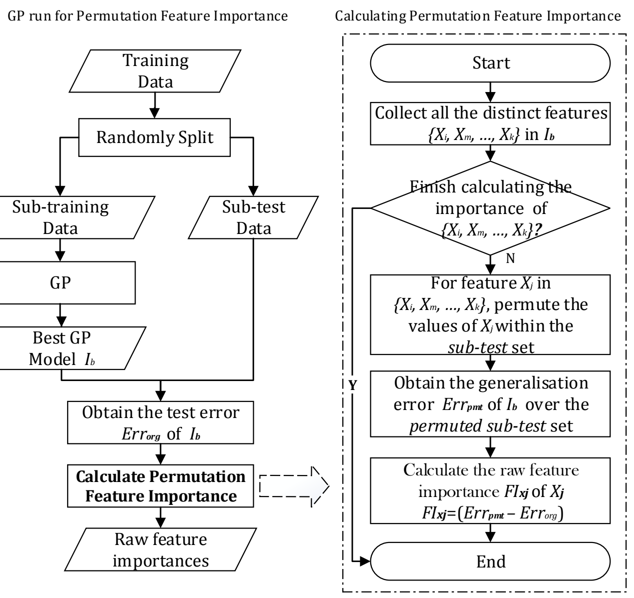

GP run for Permutation Feature Importance
Calculating Permutation Feature Importance
Training Data
Randomly Split
Sub-training Data
Sub-test Data
GP
Best GP Model $I_b$
Obtain the test error $Err_{org}$ of $I_b$
Calculate Permutation Feature Importance
Raw feature importances
Start
Collect all the distinct features $\{X_i, X_m, ..., X_k\}$ in $I_b$
Finish calculating the importance of $\{X_i, X_m, ..., X_k\}$?
N
Y
For feature $X_j$ in $\{X_i, X_m, ..., X_k\}$, permute the values of $X_j$ within the *sub-test* set
Obtain the generalisation error $Err_{pmt}$ of $I_b$ over the *permuted sub-test* set
Calculate the raw feature importance $FI_{xj}$ of $X_j$
$FI_{xj}=(Err_{pmt} - Err_{org})$
End

Fig. 2. GP for Permutation Importance (GPPI).

feature, or the extent to which the feature can contribute to reducing the error between the output and the target values.

Permutation variable/feature importance in RF is a widely used score to measure the importance of a given feature [45]. Permuting the values of the feature refers to rearranging the values of the feature randomly within the dataset. For example, the values of a feature are denoted as {4,7,9}, the permutation of the feature can take a random form among {4,9,7}, {7,4,9}, {7,9,4}, {9,7,4}, {9,4,7}. The underlying mechanism for permutation importance is that important features should have a higher influence on the performance of models, i.e. for regression problems, permuting a more important feature will lead to a higher regression error. Based on this hypothesis, we measure the feature importance in GP for symbolic regression based on permutation.

Fig.2 presents the main process of GPPI. While the left part of the figure shows the process of calculating permutation feature importance in one GP run, the right part describes the process to obtain the permutation importance of one feature. The whole process is defined as:

1) Randomly split the training data into a sub-training set and a sub-test set.
2) Carry out a standard GP run and get the best-of-run individual $I_b$, which has the lowest training error over the sub-training set.
3) Compute the generalisation error of $I_b$ over the sub-test set, which is referred to $Err_{org}(I_b)$.
4) For each feature $X_j$ in $I_b$, permute its values within the sub-test set, and get the test error of $I_b$ on the permuted sub-test set, shown as $Err_{pmt}(I_b)$.
5) Calculate the distance between $Err_{org}(I_b)$ and $Err_{pmt}(I_b)$, and use it to measure the raw feature importance of the feature $FI_{raw}(X_j)$ according to:

$$FI_{raw}(X_j)=Err_{pmt}(I_b)-Err_{org}(I_b) \tag{3}$$

**Algorithm 1:** Permutation Importance of Features

```
Initialise the number of GP runs $n$, the total number of features $m$, all
  feature importances $FIs[n][m]$;
**for** $g := 1$ *to* $n$ **do**  *GP run loop*
    Randomly split the training set into a sub-training set (70%) $D_{ta}$
      and a sub-teset set (30%) $D_{te}$;
    Run GP on $D_{ta}$;
    Select the individual $I_b$ with $min(Err_{ta})$ on $D_{ta}$;
    Calculate the $Err_{org}$ of $I_b$ over $D_{te}$;
    Collect all the distinct features $\{X_i,...,X_k\}$ in $I_b$;
    **for** $j := 1$ *to the length of* $I_b$ **do**  *Raw Importance loop*
        Shuffle/permute the values of $X_j$ within $D_{te}$, to form the
          permuted sub-test set $D_{pte}$;
        Compute the $Err_{pmt}$ of $I_b$ over $D_{pte}$;
        Calculate the raw importance value
          $FI_{raw}(X_j)=Err_{pmt}(I_b)-Err_{org}(I_b)$ ;
        Put $FI_{raw}(X_j)$ into $FIs$, i.e. $FIs[g][j]=FI_{raw}(X_j)$;
    **end**
**end**
**for** $f := 1$ *to* $m$ **do**  *Scaled Importance loop*
    Obtain the mean raw importance of $X_f$ by
      $\overline{FI_{raw}(X_f)}=\sum_{i=1}^{n} FIs[i][f]/n$;
    Obtain the standard deviation of $X_f$ by
      $\delta(X_f)=\sqrt{\sum_{i=1}^{n}\left(FIs[i][f]-\overline{FI_{raw}(X_f)}\right)^2/n}$;
    Calculate the scaled importance value of $X_f$ by
      $FI_{sca}(X_f)=\overline{FI_{raw}(X_f)}/\left(\frac{\delta}{\sqrt{n}}\right)$;
**end**
```

Steps 4 and 5 need to be performed for each distinct feature in the best-of-run individual $I_b$. It is important to note that the importance values of features absent from $I_b$ are defined to be 0 in that GP run. The whole process repeats $n$ ($n \geq 30$) independent GP runs on the given training data. The condition $n \geq 30$ is to reduce the bias of random seed used in GP runs on the importance values. The pseudo-code of this procedure is shown in Algorithm 1.

In order to make the feature selection according the importance values to be fexible and problem-independent, the final importance of a feature is defined as the scaled importance, which is the average raw feature importance normalised by the standard error of raw feature importance. It is given as:

$$FI_{sca}(X_j)=\frac{\overline{FI_{raw}(X_j)}}{\frac{\delta}{\sqrt{n}}} \tag{4}$$

where $n$ is the number of GP runs, $\overline{FI_{raw}(X_j)}$ is the average value of the raw feature importances in $n$ GP runs, $\delta$ is the standard deviation, and $\frac{\delta}{\sqrt{n}}$ is the standard error.

## *D. Feature Selection according to Permutation Feature Importance*

Instead of employing the whole set of features, GP-GPPI will use a subset of features selected by GPPI. In GPPI, features with positive $FI_{sca}(X_j)$ are selected. The positive value indicates that these features have a positive effect in reducing the regression error, and they are potentially more important than their counterparts with negative importance values. The main advantage of this selection metric is problem-independent, which is especially suitable for regression problems without domain knowledge. It does not depend on the problem domain and the total number of features. The set of positive features is expected to remove noise and irrelevant features. Thus, it can reduce the risk of overfitting and potentially improve the generalisation performance of GP for high-dimensional symbolic regression.

## IV. Experimental Design

To demonstrate the feature selection ability of GPPI and investigate its effectiveness on promoting the generalisation ability of GPSR, sets of experiments have been conducted. A comparison has been made between our new GPSR system GP-GPPI and GPSR with several various feature selection methods.

### A. Benchmark Methods

A comparison between GP-GPPI and five benchmark methods, which employ feature selection for GPSR, has been conducted in this work. The first two benchmark methods are *GP-Random Forests* (GP-RF) and *GP-C5.0 decision trees* (GP-C5.0). The other three methods are our previous method GPWFS and two variants of GPWFS. The details of the five benchmark methods are as follows:

- *GP-Random Forests (GP-RF)* is a GPSR method using features selected by random forest (RF). The permutation feature/variable importance values in RF are obtained from 30 runs using 30 different seed numbers. This setting can reduce the influence of the random seed on the importance of features. Bootstrap samples are exposed to construct the trees, and the out-of-bag samples are used to calculate the permutation importance.
- *GP-C5.0 decision trees (GP-C5.0)* is a GPSR method which employs C5.0 for feature selection. The feature importances in C5.0 are also obtained from 30 runs using the same sub-training sets as GPPI. When calculating the importance of a feature, the metric, which considers the percentage of splits the feature makes, is employed.
- *GPWFS* is a GPSR method for simultaneous feature selection and regression. A brief description of GPWFS has been given in Section II-B1. For more details, readers are referred to [36].
- *GPWFS1* is GPWFS using a different setting. GPWFS1 differs from GPWFS in the number of generations for the two stages. The first stage of GPWFS1 has the same number of generations that GPPI uses for feature selection. The number of generations in the second stage is the same as that GP-GPPI used for symbolic regression. GPWFS1 is examined to make the comparison between GP-GPPI and GPWFS close to fair.
- *GPWFS2* is a variant of GPWFS. It splits GP for feature selection and GP for symbolic regression into two separate stages. At the end of the feature selection stage, GPWFS2 collects all the distinct features appearing in the best-of-run individuals of 30 GP runs. Then the regression will perform on these selected features. In fact, GPWFS2 differs from GP-GPPI only in lacking the permutation method to determine the important of features.

In addition to these feature selection methods, standard GP, which uses the whole set of features as input and performs built-in feature selection, is also used for comparison. However, it is used as a baseline for comparison. The main focus of this work is on the comparison among GP with various feature selection methods. In each GP for regression method, 100 independent GP runs have been conducted. All the GP methods are implemented under the ECJ GP framework [46]. RF and C5.0 for feature selection are implemented under the R packages, which are "randomForest" [47] for RF and "C50" [48] for C5.0.

For a more comprehensive comparison, we also compare GP-GPPI with two non-symbolic regression methods, least absolute shrinkage and selection operator (LASSO) [49] and RF for regression. LASSO performs feature selection by employing an $\ell_1$ penalty to shrink some coefficients in the regression model to be 0. In this way, LASSO can effectively enhance the prediction performance of the regression model. These two methods are both implemented under R packages "glmnet" [50] and "randomForest" [47] with default settings. Furthermore, we investigate the influence of the computation load on GP-GPPI, and conduct a comparison between GP and GP-GPPI under the same computation time.

### B. Parameters

The parameters for all GP runs are summarised in Table I. The generations for GPWFS1 is 100 (50 generations for the first stage and the other 50 generations for the second stage). For GPWFS, the two key parameters, which are the number of generations $G_f$ to decide the splitting point of the two phases and $\beta$ to define the percentage of top individuals, are tuned using three different values, respectively. Since the total number of generations is 50, $G_f$ is properly set to $25, 30, 35$. $\beta$ takes the values of $5\%, 10\%, 15\%$. Thus, 9 $(3 * 3)$ different settings of GPWFS have been conducted. For GPWFS1, the value of $G_f$ is fixed ($G_f$=50). The value of $\beta$ in GPWFS1 is tuned among $5\%, 10\%, 15\%$. Thus, 3 different settings of GPWFS1 have been conducted.

In each run of RF for feature selection, a forest of 500 trees is built. The size of randomly chosen candidate features for each node is defined to be $m/3$, where $m$ is the total number of features on the dataset. The values are recommended values for regression [37]. The same settings are used in RF for regression. In C5.0, the parameter "metric" is set to "splits", which means the percentage of splits associated with each feature will take a part in calculating the feature importance. Other parameters use the default values. The feature selection criterion in C5.0 and RF is the same as GPPI, which is selecting features with positive importance values. As mentioned earlier, we assume these features have the potential to reduce the regression error. Thus, they are more important and should be selected. The selection criterion has a benefit of being problem-independent. Compared with only selecting the top features, it can reduce the risk of missing some important features (particularly when the top features are redundant).

We have compared GP and GP-GPPI under the same computation time and investigated the influence of the higher computation load on the results of GP-GPPI at the same time. In this set of experiments, the population size of GP-GPPI is increased to 1024, while other parameters are the same as shown in Table I. Standard GP has a population of 2048 and

TABLE I
PARAMETER SETTINGS

| Parameters | Values |
|---|---|
| Population Size | 512 |
| Generations | 50 (100 for GPWFS1) |
| Crossover Rate | 0.9 |
| Mutation Rate | 0.1 |
| Elitism Rate | 0.01 |
| Maximum Tree Depth | 10 |
| Initialisation | Ramped-Half&Half |
| Minimum Initialisation Depth | 2 |
| Maximum Initialisation Depth | 6 |
| Function Set | $+$, $0-$, $*$, $Inv(\frac{1}{x})$, sqrt |
| Terminal Set | Features (Selected Features), Random Constant $\in [-1.0, 1.0)$ |
| Fitness Function | NRMSE |
| GPWFS (GPWFS1) | |
| Generation for Feature Selection — $G_f$ | 25, 30, 35 (50 for GPWFS1) |
| Percentage of Top Individuals — $\beta$ | 5%, 10%, 15% |

TABLE II
TWO SYNTHETIC FUNCTIONS

| Functions | Training Samples | Test Sample | Noise |
|---|---|---|---|
| $F_1 = -g\frac{X_1 X_2}{X_3^2}$ | 70 points $X_1, X_2 = rnd[0,1]$ $X_3 = rnd[1,2]$ | 30 points $X_1, X_2 = rnd[0,1]$ $X_3 = rnd[1,2]$ | 50 input variables $= rnd[0, 1]$ |
| $F_2 = \frac{30 X_1 X_3}{(X-10) X_2^2}$ | 1000 points $X_1, X_3 = rnd(-1,1)$ $X_2 = rnd(1,2)$ | 10000 points $X_1, X_3 = rnd(-1,1)$ $X_2 = rnd(1,2)$ | 50 input variables $= rnd[0, 1]$ |

will be terminated when it uses the same computation time as GP-GPPI (the time of feature selection and regression).

## C. Datasets

In this work, the experiments are conducted on six high-dimensional regression datasets. While two of the datasets are synthetic data, the other four are real-world high-dimensional data. The relevant features in the two synthetic datasets with noise are known, which makes these datasets particularly suitable to examine the ability of a feature selection method [51]. The functions of the two synthetic datasets are shown in Table II. $F_1$ is the famous Newton's Law of gravitation, $g$ is the gravitational constant with the value of $6.67408E-11$. $F_2$ was taken from [52]. The sampling strategies for the training data and the test data are also shown in Table II. The noise, which were added to each dataset, consists of 50 input variables with random values in the range $[0, 1]$. The purpose of adding noise is to check whether the feature selection methods can eliminate noise and select the truly relevant features.

The four real-world regression datasets are taken from UCI [53] and previous literature on the generalisation of GP for symbolic regression [54], [55]. They are high-dimensional regression datasets having hundreds to thousands of features. Feature selection is more desired for these datasets than their counterparts with a smaller number of features. The first dataset LD50 is about the pharmacokinetics, the task of which is to predict the value of a pharmacokinetics parameter — the median lethal dose (represented as LD50). It has been used in much recent work on the generalisation of GP [54], [55], [56]. The second dataset is the Diffuse Large-B-Cell Lymphoma (represented as DLBCL), which was collected from Rosenwald et al. [57]. The task is to predict the survival time of patients who have diffuse large-B-cell lymphoma and received chemotherapy. The remaining two datasets are taken from UCI [53]. They are about communities and crimes within the United States, the Communities and Crime unnormalised dataset (CCUN) and the Communities and Crime normalised dataset (CCN). Both of them are to predict the per capita crimes. We discarded the instances which have missing values, so the number of instances in CCUN and CCN used in this work is smaller than the original data.

TABLE III
BENCHMARK PROBLEMS

| Name | # Features | #Total Instances | #Training Instances | #Test Instances |
|---|---|---|---|---|
| $F_1$ | 53 | 100 | 70 | 30 |
| $F_2$ | 53 | 11000 | 1000 | 10000 |
| LD50 | 626 | 234 | 163 | 71 |
| DLBCL | 7399 | 240 | 180 | 60 |
| CCUN | 124 | 1994 | 1395 | 599 |
| CCN | 122 | 1994 | 1395 | 599 |

## D. The Training Sets and the Test Sets

In this work, each dataset is split into a training set and a test set to investigate the generalisation performance of the evolved model in GP on unseen data. The number of features, training instances and test instances of the six datasets (including two synthetic datasets) are shown in Table III. Four of the six datasets (expect DLBCL and $F_2$) are split with 70% of instances randomly selected from the datasets for training and the other 30% instances forms the test set. This is a widely accepted way of splitting the dataset in machine learning [11], [55]. The training set and test set are provided in DLBCL. The number of training data points and test data points are given in $F_2$ [52].

During the feature selection process, the data available for all the feature selection methods is only the training sets. The test set of each task is kept to be unseen during the feature selection and model training process, so that a *fair* comparison on the effect of the feature selection methods on the generalisation of GP can be conducted. In each GP for feature selection run, the training set is further split, where 70% randomly selected instances forms the sub-training set and the other 30% forms the sub-test set to obtain the feature importance. The same sub-training sets are used in $C5.0$. RF uses the whole training set by bootstrapping a number of samples (i.e. sub-training set) for constructing the trees and obtaining the permutation importance of features on the out-of-the-bag samples (i.e. sub-test set).

## V. COMPARING GP-GPPI WITH GP-C5.0 AND GP-RF

A comparison between GP-C5.0, GP-RF and GP-GPPI will be presented in this section. Standard GP is used as a baseline for comparison. The comparison on the influence of the three feature selection methods (C5.0, RF and GPPI) to the learning ability and generalisation of GP for symbolic regression is the main focus.

Fig. 3 shows the distribution of training NRMSEs of the 100 best individuals from 100 GP for regression runs on the training data of the six datasets, while Fig. 4 gives the distribution of their corresponding test NRMSEs. Each boxplot

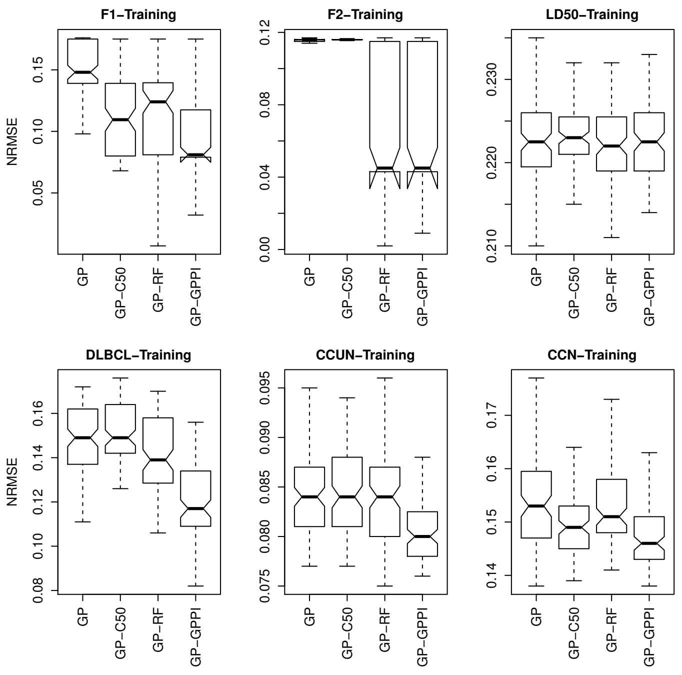


Fig. 3. Distribution of Training NRMSEs of the 100 best-of-run individuals.

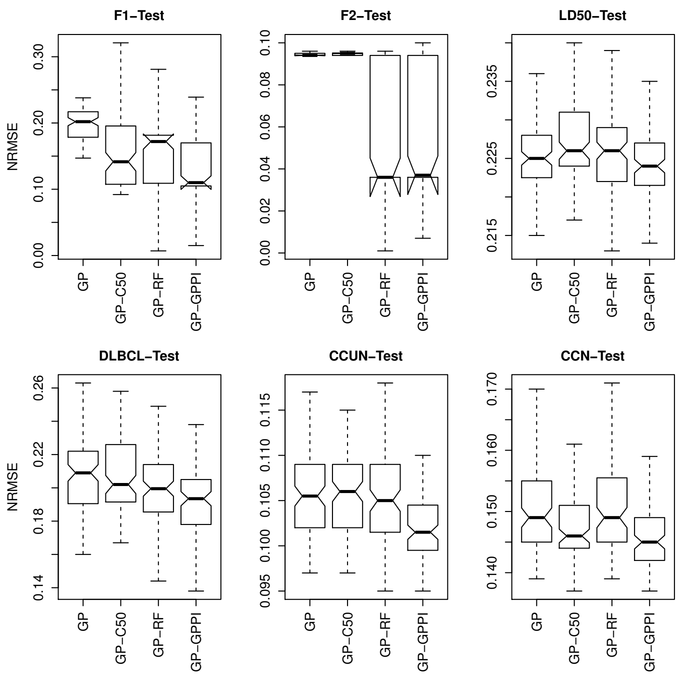


Fig. 4. Distribution of the Corresponding Test NRMSEs.

consists of four whiskered boxes for the four GP methods, respectively (On $F_2$, the notched boxplots in GP-RF and GP-GPPI look different from others, this is because the first quartile is too close to the median of the results).

Fig. 5 and Fig. 6 show the evolution plots for the datasets. On every generation, the lowest NRMSEs on the training set are recorded, and the corresponding test NRMSEs of the individual are also obtained (the test errors serve to examine the evolution of generalisation, but are never take into account during the evolutionary process). For the 100 GP runs, 100 lowest training NRMSEs and the test NRMSEs are collected on every generation. Since the median value is suggested to be more robust to outliers[11], it is preferred over the mean value in this work. The evolution plots are drawn using the median values of the 100 NRMSEs. The non-parametric statistical significance test — Wilcoxon test is conducted to compare the 100 training NRMSEs and test NRMSEs of the 100 best-of-run models. The Wilcoxon test has been conducted on comparisons between GP-GGPI and the other three methods, and also between GP and GP-RF (GP with GP-C5.0). The significance level is 0.05.

## *A. Results on the Training Sets — Learning Ability*

To investigate and demonstrate the effect of the feature selection methods on the learning ability of GP for high-dimensional regression tasks, the regression performance regarding the training NRMSEs are reported here.

As shown in the training boxplots in Fig. 3, the difference between the median of GP-GPPI and the other three methods are large on four of the six datasets, i.e. $F_1$, DLBCL, CCUN and CCN. For these four datasets, the boxes of GP-GPPI and other three methods overlap but not the median values. It indicates GP-GPPI has much better training performance than the other three methods on these datasets. On $F_2$, the boxes of GP-GPPI and GP-RF overlap with medians. However, there is no overlap between these two methods and the other two methods, i.e. GP and C5.0. This means GP-GPPI has comparable performance as GP-RF, which is much better than the other two methods. On LD50, there is no obvious difference between the four methods. According to the statistical significance tests, on $F_1$, DLBCL, CCUN and CCN, GP-GPPI has the best training performance among the four methods. On $F_2$, GP-GPPI has no significant difference from GP-RF on the training set. However, both are significantly better than the other two methods (GP and GP-C5.0). On LD50, there is no significant difference between the training performance of all the methods.

Fig. 5 shows more details of the evolutionary training process of the four GP methods. GP-GPPI generally achieves better training performance than the other three methods over generations on four out of the six datasets except for on $F_2$ and LD50. On these four training sets, GP-GPPI outperforms the other three methods on the first several generations. The difference on the NRMSEs between GP-GPPI and the other methods becomes bigger and bigger over generations. On $F_2$, GP-GPPI and GP-RF are significantly superior to the other two methods.

It is clear that in most of the datasets, feature selection can promote the learning ability of GP in most cases. An intuitive reason is that the reduction of feature space shrinks the search space of GP. Thus the evolutionary process is more likely to be guided towards the better models. Better feature selection methods can shrink the search space of GP to be much smaller but more effective since they can discard more irrelevant features while keeping important features. Thus less effort is needed for GP to converge to optimal models. It also explains

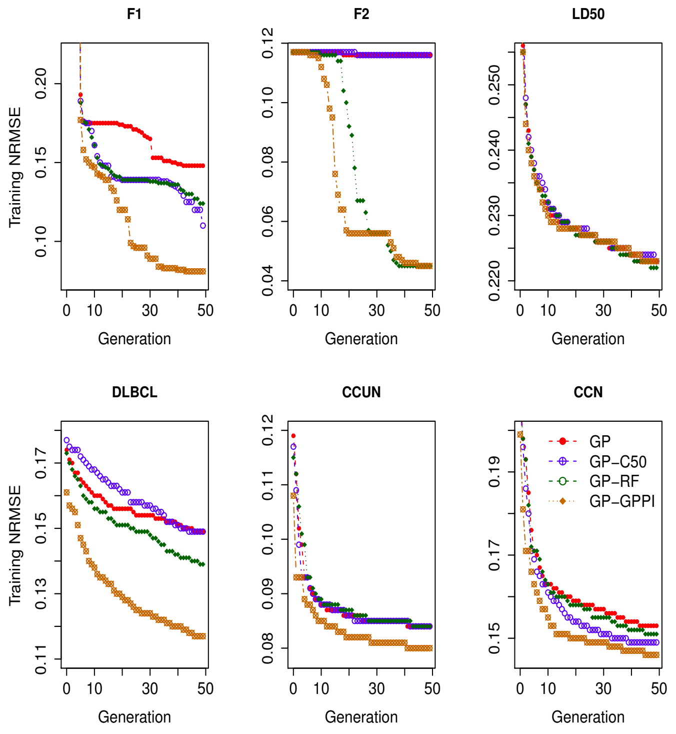


Fig. 5. The Training Error Evolution Plots.

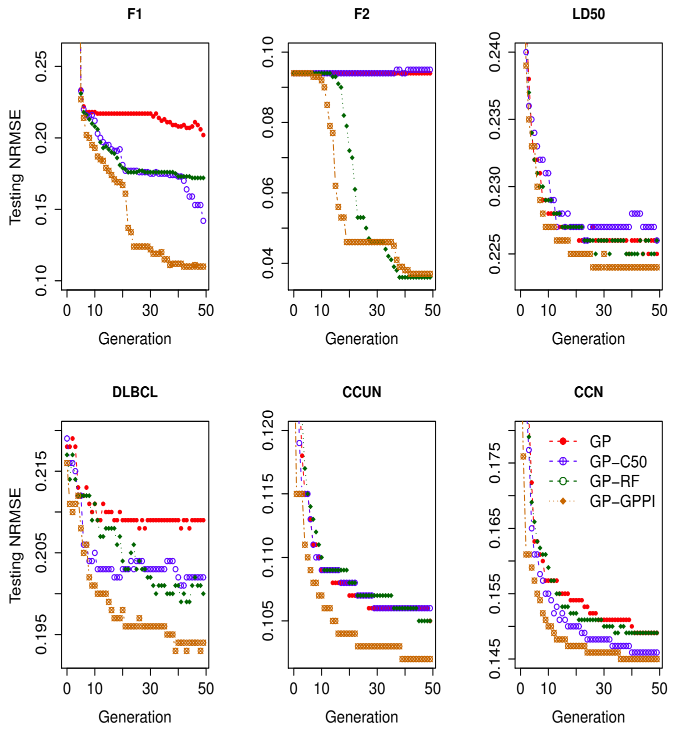


Fig. 6. The Testing Error Evolution Plots.

the pattern that the difference on NRMSEs between GP-GPPI and the other three methods is increasing over generations, and why GP-GPPI has a distinguished advantage over the other methods.

## *B. Results on the Test Sets — Generalisation Ability*

Fig. 4 shows the distribution of generalisation errors of the 100 best-of-run individuals. The overall trend is similar to the training set, i.e. GP-GPPI outperforms the other methods. Particularly on LD50, GP-GPPI has much lower median NRMSEs than the other methods on the test set. The test evolution plots in Fig.6 clearly show that GP-GPPI generally has the best generalisation performance among all the methods i.e. the lowest test NRMSE over generations. Based on the results of the statistical significance tests, GP-GPPI achieves significantly better generalisation gain than the other three methods on five of the six datasets, except for $F_2$. On $F_2$, GP-GPPI has slightly higher test error than GP-RF, but not significantly. They both have a significantly lower NRMSE than C5.0 and standard GP on the test set of $F_2$.

On the two synthetic datasets $F_1$ and $F_2$, which contain the same number of relevant features and noisy features, the generalisation ability of the four methods show different patterns. The intuitive reason might be that the target function of $F_1$($F_1 = -g\frac{x_1 x_2}{x_3^2}$) is simpler than $F_2$ ($F_2 = \frac{30x_1x_3}{(x_1-10)x_2^2}$). All the three feature selection methods can have generalisation gain for GP on $F_1$. On $F_2$, which has a more complex target function, the test errors of GP and GP-C5.0 do not reduce over generations and even increases slightly over the final several generations, while GP-RF and GP-GPPI can generalise well. On $F_2$, applying C5.0 for feature selection does not enhance but rather decrease the generalisation of GP. GP-C5.0 has significantly larger NRMSEs than GP on the test set. One of the possible reasons is that it did not keep all the relevant features (in fact, it excluded the $2nd$ feature), although it discarded a number of irrelevant features.

From the generalisation performance on the two synthetic datasets, it can be observed that in GP-RF and GP-GPPI, where the feature selection methods can reduce the noise that was deliberately added while keeping the relevant features, the evolved models have a higher probability to include the truly relevant features. They are more accurate in expressing the true relationship between the input variables and the target variables, thus can definitely have better generalisation performance.

On the four real-world datasets, GP-GPPI achieves the best generalisation performance among the four methods, which is confirmed by the Wilcoxon test. While on DLBCL, GP-GPPI has notable generalisation gain over the other three methods, on the other three tasks, GP-GPPI still outperforms the other methods. On LD50 and DLBCL, while GP-RF has slightly but not significantly better generalisation performance than GP, GP-C5.0 has significantly higher test NRMSEs than GP on LD50 and slightly better generalisation gain than GP on DLBCL. On CCUN and CCN, GP-RF can not improve the generalisation performance of GP to a significant level. GP-C5.0 achieves a significant generalisation gain on CCN.

In summary, GPPI can enhance the generalisation of GP more effectively because it can discard more noisy/irrelevant features than other feature selection methods (feature selection results will be presented in more detail in Section VII), so that GP is more likely to construct models using the

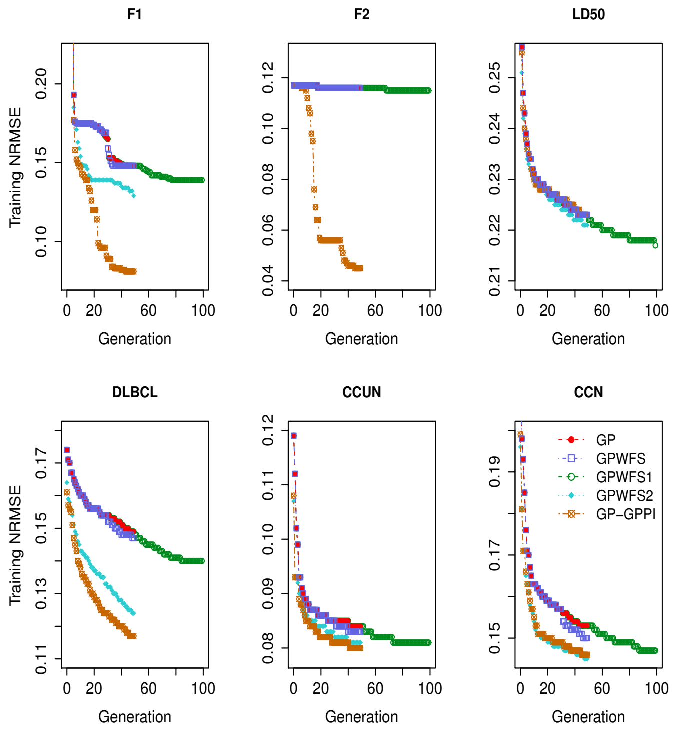


Fig. 7. GP-GPPI and Variants of GPWFS — The Training Error Evolution Plots (GPWFS1 is deliberately set to run 100 generations).

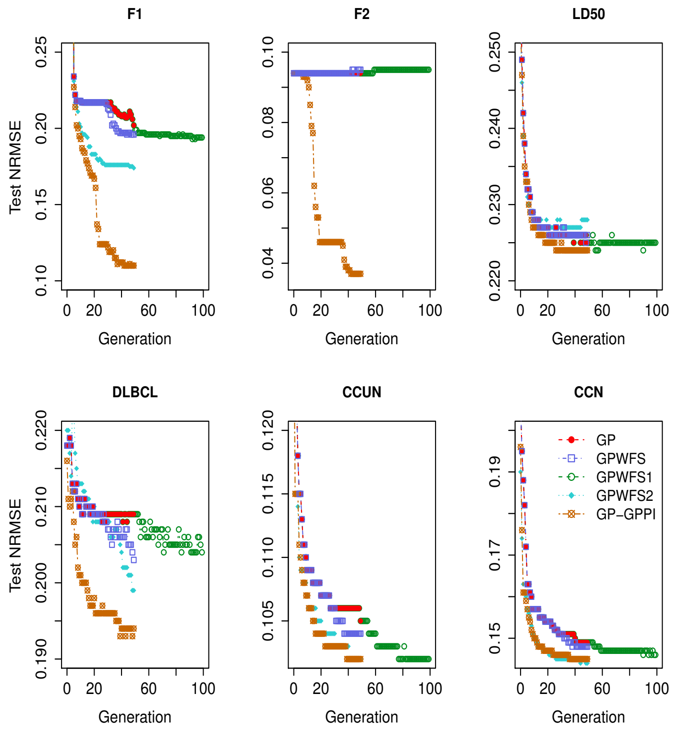


Fig. 8. GPPI and Variants of GPWFS — The Corresponding Testing Error Evolution Plots (GPWFS1 is deliberately set to run 100 generations).

relevant features. Moreover, the GP models with a continuous property learn more than the stepwise function of the decision trees, thus leading to better performance of GPPI in detecting important features. This could be a major reason that GP-GPPI is superior to the other GPSR methods on generalisation performance.

## VI. Comparisons between GP-GPPI and Variants of GPWFS

The major difference between feature selection in GP-GPPI and GPWFS is the method to decide the importance of features. GPPI has an additional component, which is the permutation feature importance. To investigate the effect of the permutation method on identifying the truly important features from the potentially relevant features, the comparison between GP-GPPI and variants of GPWFS is necessary.

Fig. 7 and Fig. 8 show the evolution plots of the median training and test NRMSEs of the 100 best-of-generation individuals obtained on the training data in GP, GPWFS, GPWFS1, GPWFS2 and GP-GPPI. Here, the best settings among the 9 settings of GPWFS and 3 settings of GPWFS1, which lead to the best generalisation performance in these two methods, are chosen to report and compare.

As shown in Fig. 7, compared with the three variants of GPWFS, GP-GPPI has better training performance on five of the six datasets except for LD50. On the two synthetic datasets $F_1$ and $F_2$, GP-GPPI has a much better learning performance than all the three variants of GPWFS, shown as the much lower training errors over generations. The advantage of GP-GPPI over the variants of GPWFS are all significant on $F_1$ and $F_2$. On DLBCL, GP-GPPI achieves significantly smaller training errors than GPWFS and GPWFS1, and slightly better training performance than GPWFS2. On CCUN and CCN, GP-GPPI has slightly but not significantly better learning performance than GPWFS1 and GPWFS2. However, it still outperforms GPWFS significantly on these two training sets. On LD50, GP-GPPI achieves a comparable training performance with GPWFS and GPWFS2, which are significantly worse than GPWFS1 (the plot in green colour in Fig.7).

From Fig. 8, it can be observed that the overall pattern in the test sets is similar to the training sets, i.e. GP-GPPI outperforms the three variants of GPWFS in achieving better generalisation performance on all the six datasets. On $F_1$, $F_2$ and DLBCL, GP-GPPI has achieved a dramatic generalisation gain than the three variants of GPWFS. On LD50, GP-GPPI also obtains significantly better generalisation performance than the three variants of GPWFS. On CCUN and CCN, GP-GPPI has significantly smaller test errors than GPWFS. Compared with the generalisation performance of GPWFS1, GP-GPPI obtained comparable results on CCUN and significantly better results on CCN. The comparison between GP-GPPI and GPWFS2 is different from GPWFS1. While GP-GPPI has significantly smaller test errors than GPWFS2 on CCUN, their generalisation performance is similar on CCN.

In summary, GP-GPPI is superior to the variants of GPWFS both in the learning performance and the generalisation ability on most of the datasets. On the two synthetic datasets, which contain a large number of noisy features, the advantage of GP-GPPI over the variants of GPWFS is more obvious than the other four real-world datasets. GP-GPPI outperforms GPWFS, which might be because GPPI collects features from a number of best-of-run GP individuals in different GP runs, which are more sufficiently evolved than those in GPWFS. Features appearing in these individuals are intuitively more reliable than those selected from the best-of-generation individuals in GPWFS. However, the major contribution is owed to the effectiveness of permutation importance, which helps to identify the real important features in GPPI. The comparison between GP-GPPI and GPWFS2 confirms the contribution of the permutation importance method.

## VII. Further Analysis

This section presents a further analysis of the feature selection results and the regression models evolved by the GP methods. It is expected to provide a good way to understand how GPPI advances the other feature selection methods in promoting the generalisation of GPSR.

### A. Feature Selection Results

To further analyse the feature selection ability of GPPI, it is necessary to compare the selected features from different methods. Here, we focus mainly on the comparison between RF and GPPI, since they generally have the best performance on benchmark problems and share the same mechanism(i.e. permutation) to evaluate the importance of features. As the main difference between the two methods lies in searching the features and building the trees, the comparison is to demonstrate whether the feature selection ability of GP is superior to the search ability of RF.

Regarding the importance values of features, a positive importance value is formed by the increased regression error when the feature is shuffled. Since a new sub-test error is obtained from a completely random feature, it should be higher than the initial error on the sub-test sets. A positive value indicates that the feature can contribute to reducing the regression error. The bigger the value is, the more important the feature is to the response variable. In highly correlated datasets, the importance value of a feature in a single tree (in GP or RF) can be small since another feature might have duplicate information. However, obtaining the feature importance over a group of trees can reduce the limitation. A negative importance value is obtained by the reduced new regression error on the permuted sub-test set. It indicates that the completely random feature works even better than the original feature. Thus, the feature is probably not predictive enough to the response value, i.e. it is very likely not important. The zero value means the feature does not appear in the GP trees or the decision trees in most cases.

*1) Feature Selection Results on the Synthetic Datasets:* The feature importance values in GPPI and RF on $F_1$ and $F_2$ are shown in Fig. 9. It can be seen that for $F_1$, both methods have a relatively small number of positive features, which indicates that both can dramatically reduce the number of features. Moreover, in the two methods, the top three important features are consistent with all the truly relevant features. In RF, the top three features all have much higher feature importance from the other features, although it has a

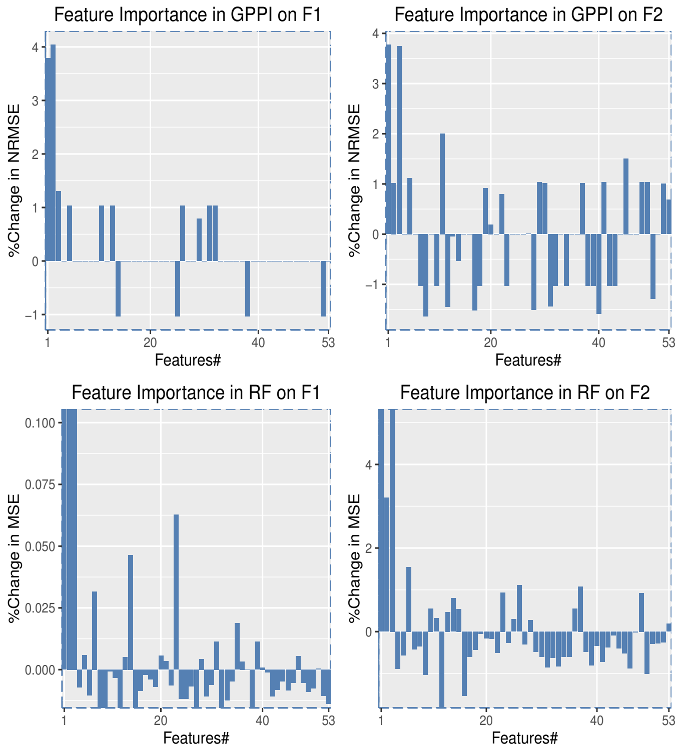


Fig. 9. Feature Selection Results on the Two Synthetic Datasets.

larger number of positive features. On the right part which is for $F_2$, the most obvious pattern is the same as $F_1$. Both GGPFI and RF can find the truly relevant features, and most of these features can have much higher importance values than other features. Compared with $F_1$, both methods have a much higher number of negative features on $F_2$. The existence of these features indicates permutation importance can identify the noise which is not predictive to the response variable. As mentioned earlier, $F_2$ is harder than $F_1$. This might cause a decrease in the search performance of GP and RF for the important features. Permutation importance contributes more to identify the irrelevant features in this case. It is clear that for the two tasks, both GPPI and RF can have good feature selection ability on discarding a large number of noisy features while keeping the most important features. The discarding of noise takes the forms of not being used/included by the *good* individuals and assigning a negative importance value to them.

*2) Feature Selection Results on the Real-World Datasets:* For the real-world datasets, which have higher feature dimensionality than the synthetic datasets, the overall pattern is different. The detailed importance values of LD50 and DLBCL are presented since they have extremely higher numbers of features and are (much) harder than the other two datasets (CCUN and CCN).

Fig. 10 shows the importance values of features in LD50. In GPPI, the number of features included in the evolved models is much lower, and the permutation of features helps to identify more negative features than RF. Thus the percentage of features with negative importance is much higher in GPPI (around 30%) than in RF, and a much smaller number of positive features in GPPI than RF. Fig. 11 shows the feature importance results on DLBCL. Compared with the total number of features (which is 7399), both GPPI and RF can discard a large number of noisy/irrelevant features. With regards to the number of features included in models, the two methods are quite different. GPPI selects a much smaller number of features than RF. Different from LD50, where GPPI detects a large number of negative features, on DLBCL most of the features included in the *good* individuals of GPPI are positive. However, many of the features appearing in the trees in RF have negative importance values. Considering the number of positive features, GPPI is still much smaller than RF. The overall pattern on the other two datasets (CCUN and CCN) is similar with LD50 and DLBCL, that is, RF selects much more features than GPPI. This is partially due to the larger number of trees in RF, but the major reason is that due to the randomly restricted feature selection scheme in RF, more redundant features can appear to be important in RF, which would be considered redundant and eliminated by GP. This is confirmed by the much larger number of positive features in RF than GPPI on LD50, where many pairwise features are highly correlated regarding Pearson correlation (the correlation plots are shown in online Supplementary Material). Overall, GPPI is better in selecting fewer important features when tackling high-dimensional real-world datasets.

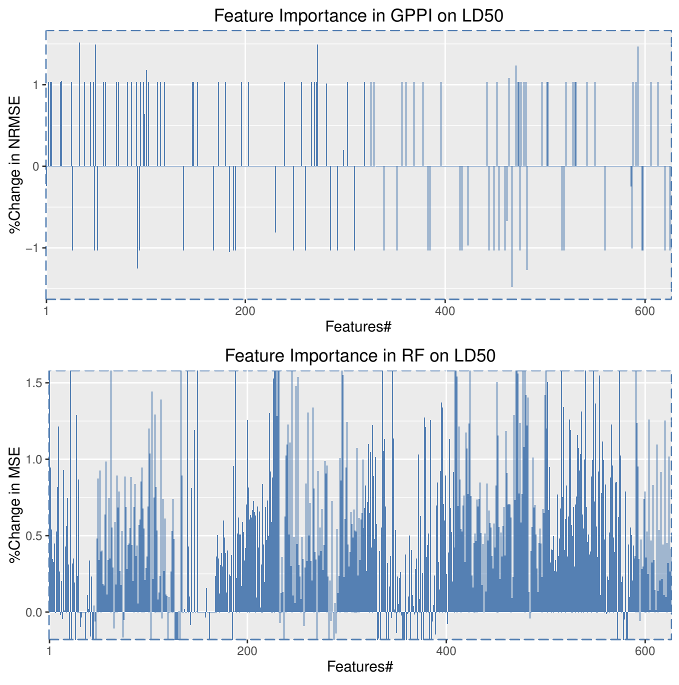


Fig. 10. Feature Selection Results on LD50.

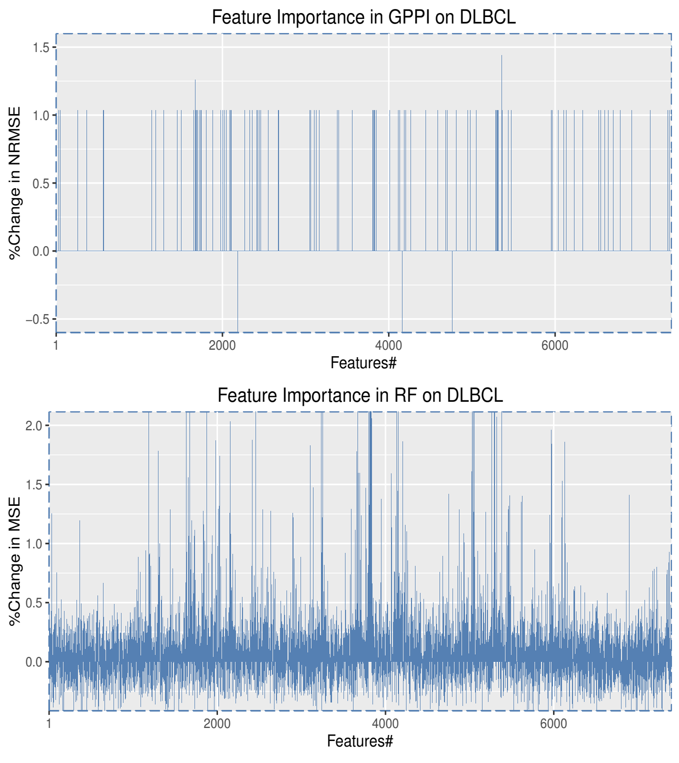


Fig. 11. Feature Selection Results on DLBCL.

TABLE IV
Top 10 Important Features on Real-World Datasets.

| Dataset | Method | Top 10 Features |
|---|---|---|
| LD50 | RFs | $228, 335, 227, 469, 244, 0, 132, 408, 345, 547$ |
| | GPPI | $33, 271, 49, 592, 470, 100, 463, 15, 2, 146$ |
| DLBCL | RFs | $4130, 5289, 2458, 1629, 1187, 5378, 5292, 3798, 3251, 5291$ |
| | GPPI | $5357, 1674, 2441, 3400, 6134, 48, 262, 566, 573, 1197$ |
| CCUN | RFs | $43, 49, 2, 42, 67, 48, 14, 39, 123, 1$ |
| | GPPI | $49, 37, 40, 26, 70, 39, 4, 16, 48, 67$ |
| CCN | RFs | $50, 41, 68, 2, 32, 40, 27, 71, 30, 69$ |
| | GPPI | $44, 50, 3, 43, 15, 68, 2, 40, 49, 41, 71$ |

Since the number of selected features is large on all of the four datasets in both methods, we zoom in the results by listing the most important features. The top 10 important features selected by the two methods on the four real-world datasets are shown in Table IV in a descending order of importance. It can be observed that the top features selected by the two methods are totally inconsistent on LD50 and DLBCL. The small ratio of the instances over the number of features is a major reason. Another reason is the existence of redundant features. In contrast, the two methods selected many consistent features among the top ten important ones on the other two datasets. On CCN, six out of the ten are consistent. On CCUN, the number is four. Compared with LD50 and DLBCL, the ratios of the instances over the number of features are much higher on CCN and CCUN. In summary, for the tasks that have a high dimensionality and a large number of training samples, such as CCUN and CCN, GPPI and RF can select many consistent features. While for tasks which have a much higher number of features than the number of instances, like LD50 and DLBCL, feature selection process becomes much harder and many features have similar importance to the response variable. Thus it is difficult to select the same features between the two methods.

## B. *Analysis of Evolved Models*

A further analysis was taken on examining the models evolved by GP using features selected by the different methods. We randomly picked three evolved regression models from the 100 GP runs on the synthetic dataset $F_1$, where the target function is known, and all the feature selection methods performed well. The evolved models are shown in Table V. The mathematically simplified forms of the models are also

TABLE V
EXAMPLES OF GP INDIVIDUALS ON $F_1 = -g\frac{X_1X_2}{X_3^2}$ $(g = 6.67408E-11)$

| # Run | Method | Example of Best-of-Run Individual | Simplified Individual |
|---|---|---|---|
| 23rd | GP | $(*(*-0.33(*(*(*-0.600.099)(0-X_2))(InvX_3)))(*-0.052(*(sqrt(sqrtX_{14}))$ $(*(*(0-(*0.10(0--0.0054)))(*0.047(0--0.0054)))(*-0.29(0-(Inv\ X_3)))))))$ | $\frac{4.83E-11X_2\sqrt{\sqrt{X_{14}}}}{X_3^2}$ |
|  | GP-C50 | $(*0.083(*(*0.10(*(0-(*-0.015X_2))(*(*(*0.10(0-(*-0.015X_2)))(*X_1-0.039))-0.97)))-0.0054))$ | $-4.449E-11X_1X_2$ |
|  | GP-RF | $(*(*(0-0.36)(*(*(0-0.36)(*0.083(*(*(*-0.15-0.018)(+0.83(*X_{23}X_{31})))$ $(*-0.034-0.015))))-0.15))(*-0.15-0.018))$ | $-1.049E-11(0.83+X_{23}X_{31})$ |
|  | GPWFS | $(*(*(0-X_2)(*(*(*(*-0.60\ 0.099)(0-X_2))((InvX_3))(InvX_4)))(*-0.052(*(sqrtX_{14})$ $(*(*(0-(*0.11(0-\ -0.0054)))(*0.047(0--0.0054)))(*-0.29-0.60)))))$ | $\frac{7.366E-11X_2^2\sqrt{X_{14}}}{X_3^2}$ |
|  | GP-GPPI | $(*(*0.90(*(*-0.052(*(*(0-0.032)(Inv(+X_3\ 0.74)))(Inv(+(+X_30.74)0.74))))(*-0.29-0.60)))$ $(*(*(*(*-0.015(*(*(*(*-0.29\ 0.099)0.054)(Inv(+X_30.74)))X_1))(Inv-0.78))(0-(0-0.055)))X_2))$ | $\frac{7.9104E-9X_1X_2}{(X_3+0.74)^2(X_3+1.48)}$ |
| 55th | GP | $(*(*-0.17\ 0.073)(*(*(*(*-0.0970.56)(+(*(*X_{50}-0.44)(*0.56-8.30E-4))-6.86E-4))$ $(+(*0.0065\ (0-X_1))-6.86E-4))(*\ 0.0065(0-X_1))))$ | $4.381E-6\ X_1(2.045E-4\ X_{50}$ $-6.86E-4)\ (0.0065\ X_1+6.86E-4)$ |
|  | GP-C50 | $(*(*(0-X_2)(0-\ 0.092))(*(*(0-(+(*(sqrt0.39)(sqrt0.39))\ (0-(+0.45X_1))))-8.30E-4)$ $(*(Inv(0-(Inv(*(*0.01-0.82)\ 0.002))))(*(sqrt0.39)(*-0.033\ -0.82)))))$ | $2.116E-11\ X_2\ (X_1+0.06))$ |
|  | GP-RF | $(*7.16E-4(0-(*(*(*X_1(*(*X_2-0.76)-8.30E-4))(*0.010\ -0.76))(*0.010-0.76))))$ | $-2.608E-11\ X_1\ X_2$ |
|  | GPWFS | $(*(0-(*-0.17\ 0.073))(*(*(*(*-0.097\ 0.56)(0-\ (+(*0.0065(0-X_1))-6.86E-4)))(0-X_1))(*0.0065$ $(+(*(*X_{50}-0.44)(+(*(*X_{50}-0.44)(*0.56-8.31E-4))-6.86E-4))\ -6.86E-4))))$ | $4.381E-6(0.0065X_1+6.86E-4)$ $(2.12E-4X_{50}^2-6.86E-4)$ |
|  | GP-GPPI | $(*(*(*(+-0.23\ 0.86)(*X_1\ 0.0022))\ (Inv(+X_3(*(Inv(Inv(+X_3X_3)))(*0.074-0.046)))))$ $(*(*(sqrt(*(Inv(+X_3\ (*-0.046(+-0.230.86))))(Inv(+X_3\ X_3))))\ (*(*(0-0.0065)-0.44)$ $(*0.074(*-0.046\ 0.0065))))(Inv(Inv(Inv(Inv\ X_2))))))$ | $\frac{8.711E-11X_1X_2}{X_3\sqrt{2X_3^2-0.05796X_3}}$ |
| 69th | GP | $(*(*(*(*(0--0.086)(*-0.018(0-(0--0.086))))(*(0-\ -0.086)(*(*(*(0-0.0051)(0-\ -0.086))$ $(0-(+-0.46(0-\ -0.018))))(0-\ 0.54))))0.0051)(Inv-0.90))$ | $6.527E-12$ |
|  | GP-C50 | $(*(sqrt(+(sqrt(sqrt\ X_2))\ X_2))(*(*(*(*(0-\ 0.0051)(0-\ 0.0051))(*(sqrtX_2)\ 0.0046))(*$ $-0.080(0-(*(sqrtX_2)0.0046))))\ -0.28))$ | $-1.03E-10X_2\sqrt{X_2+\sqrt{\sqrt{X_2}}}$ |
|  | GP-RF | $(*(*(*(*(*0.19\ 0.0039)(*(+(+(*0.98\ 0.34)X_1)(*0.900.28))(*0.98\ 0.0039)))$ $(0-0.0051))(*(0-(*(+(+(*0.980.0039)X_1)X_47)(*0.98\ 0.0039)))X_2))\ (0-(+(*(*$ $(+X_2X_5)\ (*(*-0.014(*0.19\ 0.0039))X_2))(0-(+X_2X_5)))(+\ -0.014(*0.98\ 0.28)))))$ | $5.413E-11X_2(X_1+0.5852)$ $(X_1+X_{47}+0.0038)$ |
|  | GPWFS | $(*(*(*(*(sqrt(0-\ -0.24))(*(*(*(0-\ 0.0051)(0-\ -0.086)\ )\ (0-\ (+-0.46\ 0.11)))(0-$ $0.54)))\ (0-(0-0.0051)))0.0051)\ (*(+-0.46\ 0.11)(0-\ -0.018)))$ | $7.04E-12$ |
|  | GP-GPPI | $(*(Inv(0-(*X_3X_3)))(*(*(0-\ -0.012)(*(0-\ -0.012)\ (*-0.023(0-(*0.032\ 0.95)))))$ $(*(*(+(*-0.014\ X_1)\ (*-0.014X_1))\ (*0.057\ X_2))(+(+(Inv(0-\ (Inv0.33)))$ $(*-0.014(*-0.023(0-(Inv0.33)))))(*(*-0.014X_1)\ (*-0.023\ (Inv(*-0.014X_1))))))))$ | $\frac{5.669E-10X_1X_2}{X_3^2}$ |

TABLE VI
COMPARISON BETWEEN LASSO, RANDOM FOREST (RF), GP, AND GP-GPPI

| Benchmark | Method | Training NRMSE (Medain±MAD) | Test NRMSE (Medain±MAD) | Significance Test (with GP-GPPI) (training, test) |
|---|---|---|---|---|
| $F_1$ | LASSO | 0.17 | 0.22 | (−, −) |
|  | RF | 0.055±0.0013 | 0.16±0.0017 | (−, −) |
|  | GP | 0.012±0.016 | 0.095±0.03 | (+, −) |
|  | GP-GPPI | 0.037±0.043 | 0.049±0.064 |  |
| $F_2$ | LASSO | 0.11 | 0.09 | (−, −) |
|  | RF | 0.040±4.20E-4 | 0.078±5.61E-4 | (−, −) |
|  | GP | 0.002±2.97E-3 | 0.005±4.45E-3 | (=, =) |
|  | GP-GPPI | 0.005±4.45E-3 | 0.004±2.97E-3 |  |
| LD50 | LASSO | 0.04 | 0.68 | (+, −) |
|  | RF | 0.097±7.61E-4 | 0.23±0.0013 | (+, −) |
|  | GP | 0.19±0.009 | 0.25±0.026 | (+, −) |
|  | GP-GPPI | 0.21±4.45E-3 | 0.21±4.45E-3 |  |
| DLBCL | LASSO | 0.18 | 0.22 | (−, −) |
|  | RF | 0.058±7.77E-4 | 0.13±0.0014 | (+, −) |
|  | GP | 0.088±0.012 | 0.182±0.032 | (−, −) |
|  | GP-GPPI | 0.081±0.012 | 0.11±0.019 |  |
| CCUN | LASSO | 0.13 | 0.15 | (−, −) |
|  | RF | 0.030±1.18E-4 | 0.098±2.25E-4 | (+, =) |
|  | GP | 0.073±1.48E-3 | 0.099±2.22E-3 | (+, =) |
|  | GP-GPPI | 0.076±1.48E-3 | 0.097±2.97E-3 |  |
| CCN | LASSO | 0.21 | 0.23 | (−, −) |
|  | RF | 0.054±1.77E-4 | 0.141±3.44E-4 | (+, −) |
|  | GP | 0.133±2.97E-3 | 0.143±2.97E-3 | (+, −) |
|  | GP-GPPI | 0.139±2.22E-3 | 0.139±2.97E-3 |  |

presented in Table V for an easier analysis of the models. In these three runs, regarding the relevant features in the target function $F_1 = -g\frac{X_1X_2}{X_3^2}$ $(g = 6.67408E-11)$, it can be observed that GP-GPPI can include all the important features, and uses only the relevant features to construct the models. On the other hand, the other three GPSR methods and standard GP either include some noisy features or can not include all the relevant features. Concerning the shape of the models, it is also easy to find that models in GP-GPPI have a much closer/similar shape with the target model than those in the other four methods (in the 69th run, GPPI almost found the "true" model).

## *C. Further Comparison*

TABLE VI shows the results of GP and GP-GPPI under the same computation time. The results of LASSO and RF for regression are also shown in the TABLE. For methods with stochastic results (i.e. RF, GP and GP-GPPI), the median value and the mean absolute deviation (MAD) of the 100 lowest training errors and their corresponding test errors are presented. Median and MAD are claimed to be more robust to the outliers [58]. For LASSO, only one unique result is reported. The Wilcoxon test has been conducted on the paired training errors and the test errors of the 100 best-of-run individuals in GP-GPPI and the two methods (i.e., GP-GPPI vs. GP, GP-GPPI vs. RF). The statistical significance test, Z-test, is used to test the significant difference between GP-GPPI (with a group of 100 results) and LASSO (one result). While "−" means GP-GPPI performs significantly better than the compared method, "+" indicates GP-GPPI is significantly worse, and "=" stands for no significant difference.

It is clear that GP-GPPI achieves much smaller NRMSEs than LASSO on both the training sets and the test sets on most of the benchmark problems except for LD50. On LD50, GP-GPPI has a significantly worse training performance than LASSO. However, it achieves a much better generalisation performance on LD50, which is also significant. On these synthetic datasets generated by the nonlinear target functions with much noise and real-world high-dimensional regression tasks, LASSO does not generalise well. Compared with RF for regression, GP-GPPI obtains much smaller training errors and test errors on the two synthetic datasets, both of which are significant. On the four real-world datasets, GP-GPPI has significantly larger training errors than RF. However, the generalisation performance of GP-GPPI is better than RF on all the four test sets. While on CCUN, the generalisation error of GP-GPPI is slightly better than RF, on the other three test sets, GP-GGPI outperforms RF in a significant way. In general, GP-GPPI has a better generalisation ability than RF on the benchmark problems.

When comparing GP-GPPI with GP (GP doubles the population size of GP-GPPI) under the same computation time, the results show that on most of the benchmark problems, the training errors of GP-GPPI are larger than RF, but it has much smaller test errors than GP. On DLBCL which has a large number of features (7399) and a small number of instances, GP-GPPI outperforms GP on both the training set and the test set, which indicates that the generalisation improvement brought by GPPI to GP is not because of more computation effort. Moreover, the regression performance of GP-GPPI in this set of experiments is generally better than the original setting, which means that better regression performance can be expected when increasing the computation load properly. However, it does not indicate that more computation load can always achieve better generalisation, because of the risk of overfitting.

## VIII. Conclusions and Future Work

In this paper, a new feature selection method GPPI for GP for high-dimensional symbolic regression is proposed. GPPI collects features appearing in a number of best-of-run GP individuals, and obtains the importance of features using a permutation measure. Feature selection is based on the importance values. The feature selection results show that compared with RF, which is effective in finding important features along with the presence of redundant features, GPPI is more effective in identifying the truly relevant features. The regression results of GP employing various feature selection methods show that GPPI not only outperforms RF and C5.0 in improving the learning performance of GP, but also gains much more benefits on the generalisation of GP. GP-GPPI also outperforms GPWFS in enhancing both the learning performance and generalisation ability of GP. Further analysis of the models evolved indicates that GP-GPPI is superior to the other methods on evolving models including only the truly relevant features. Generally, these models are more comprehensible.

Several interesting directions can be further investigated in the near future. The ability of GPPI to identify the truly relevant features is only confirmed on the two synthetic datasets in this work. We would like to find more real-world datasets, on which the truly relevant features are known in advance, to further investigate and have more confidence in the feature selection ability of GPPI. Meanwhile, since the focus of this work is on feature selection as a way to improve the generalisation of GP, the major comparison has been conducted between different feature selection methods. The comparison between GP-GPPI with some other state-of-the-art methods to promote GP's generalisation performance, such as Pareto GP taking the model complexity as the second objective of GP [13], [18], and improving generalisation of GP by operating the training samples [59], [60], will also be investigated in the near future. Furthermore, we intend to investigate whether GPPI can promote the performance of RF for regression/classification. It is also worth exploring the effect of feature selection based on permutation importance and Gini importance in RF to enhance the prediction ability of regression/classification trees.

## Acknowledgment

This work was supported in part by the Marsden Fund of New Zealand Government under Contracts VUW1209, VUW1509 and 16-VUW-111, Huawei Industry Fund E2880/3663, and the University Research Fund at Victoria University of Wellington 209861/3580, 209862/3580, and 213150/3662.

## References

[1] R. E. Bellman, *Dynamic Programming*. Dover Publications, Incorporated, 2003.

[2] D. P. Muni, N. R. Pal, and J. Das, "Genetic programming for simultaneous feature selection and classifier design," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 36, no. 1, pp. 106–117, 2006.

[3] K. Neshatian and M. Zhang, "Pareto front feature selection: using genetic programming to explore feature space," in *Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation Conference (GECCO)*, 2009, pp. 1027–1034.

[4] K. Nag and N. R. Pal, "A multiobjective genetic programming-based ensemble for simultaneous feature selection and classification," *IEEE Transactions on Cybernetics*, vol. 46, no. 2, pp. 499–510, 2016.

[5] J. R. Koza, *Genetic programming: on the programming of computers by means of natural selection*. The MIT Press, 1992.

[6] S. Geman, E. Bienenstock, and R. Doursat, "Neural networks and the bias/variance dilemma," *Neural computation*, vol. 4, no. 1, pp. 1–58, 1992.

[7] S. Y. Chong, P. Tino, D. C. Ku, and X. Yao, "Improving generalization performance in co-evolutionary learning," *IEEE Transactions on Evolutionary Computation*, vol. 16, no. 1, pp. 70–85, 2012.

[8] I. Kushchu, "Genetic programming and evolutionary generalization," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 5, pp. 431–442, 2002.

[9] N. Q. Uy, N. T. Hien, N. X. Hoai, and M. O'Neill, "Improving the generalisation ability of genetic programming with semantic similarity based crossover," in *Proceedings of the 13th European conference on Genetic Programming (EuroGP)*, 2010, pp. 184–195.

[10] M. Castelli, L. Trujillo, L. Vanneschi, S. Silva *et al.*, "Geometric semantic genetic programming with local search," in *Proceedings of the 17th Annual Conference on Genetic and Evolutionary Computation Conference (GECCO)*, 2015, pp. 999–1006.

[11] I. Gonçalves, S. Silva, and C. M. Fonseca, "On the generalization ability of geometric semantic genetic programming," in *Proceedings of the 18th European conference on Genetic Programming (EuroGP)*, 2015, pp. 41–52.

[12] M. ONeill, L. Vanneschi, S. Gustafson, and W. Banzhaf, "Open issues in genetic programming," *Genetic Programming and Evolvable Machines*, vol. 11, no. 3-4, pp. 339–363, 2010.

[13] E. J. Vladislavleva, G. F. Smits, and D. Den Hertog, "Order of nonlinearity as a complexity measure for models generated by symbolic regression via pareto genetic programming," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 2, pp. 333–349, 2009.

[14] J. Ni, R. H. Drieberg, and P. I. Rockett, "The use of an analytic quotient operator in genetic programming," *IEEE Transactions on Evolutionary Computation*, vol. 17, no. 1, pp. 146–152, 2013.

[15] J. Ni and P. Rockett, "Tikhonov regularization as a complexity measure in multiobjective genetic programming," *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 2, pp. 157–166, 2015.

[16] S. Gustafson, E. K. Burke, and N. Krasnogor, "On improving genetic programming for symbolic regression," in *IEEE Congress on Evolutionary Computation (CEC)*, 2005, pp. 912–919.

[17] M. A. Haeri, M. M. Ebadzadeh, and G. Folino, "Improving GP generalization: a variance-based layered learning approach," *Genetic Programming and Evolvable Machines*, vol. 16, no. 1, pp. 27–55, 2015.

[18] S. S. Mousavi Astarabadi and M. M. Ebadzadeh, "Avoiding overfitting in symbolic regression using the first order derivative of GP trees," in *Proceedings of the 17th Annual Conference on Genetic and Evolutionary Computation Conference (GECCO)*, 2015, pp. 1441–1442.

[19] T. Kowaliw and R. Doursat, "Bias-variance decomposition in genetic programming," *Open Mathematics*, vol. 14, no. 1, pp. 62–80, 2016.

[20] B. Xue, M. Zhang, W. N. Browne, and X. Yao, "A survey on evolutionary computation approaches to feature selection," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 4, pp. 606–626, 2016.

[21] Y. Zhai, Y. S. Ong, and I. W. Tsang, "The emerging "big dimensionality"," *IEEE Computational Intelligence Magazine*, vol. 9, no. 3, pp. 14–26, 2014.

[22] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *The Journal of Machine Learning Research*, vol. 3, pp. 1157–1182, 2003.

[23] H. Peng, F. Long, and C. Ding, "Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 8, pp. 1226–1238, 2005.

[24] L. Yu and H. Liu, "Feature selection for high-dimensional data: A fast correlation-based filter solution," in *ICML*, vol. 3, 2003, pp. 856–863.

[25] R. Kohavi and G. H. John, "Wrappers for feature subset selection," *Artificial intelligence*, vol. 97, no. 1, pp. 273–324, 1997.

[26] F. Ahmad, N. A. M. Isa, Z. Hussain, M. K. Osman, and S. N. Sulaiman, "A ga-based feature selection and parameter optimization of an ann in diagnosing breast cancer," *Pattern Analysis and Applications*, vol. 18, no. 4, pp. 861–870, 2015.

[27] S. Gu, R. Cheng, and Y. Jin, "Feature selection for high-dimensional classification using a competitive swarm optimizer," *Soft Computing*, pp. 1–12, 2016.

[28] B. Xue, M. Zhang, and W. N. Browne, "Particle swarm optimization for feature selection in classification: A multi-objective approach," *IEEE Transactions on Cybernetics*, vol. 43, no. 6, pp. 1656–1671, 2013.

[29] F. Nie, H. Huang, X. Cai, and C. H. Ding, "Efficient and robust feature selection via joint 2, 1-norms minimization," in *Advances in neural information processing systems*, 2010, pp. 1813–1821.

[30] G. Guo and C. R. Dyer, "Simultaneous feature selection and classifier training via linear programming: A case study for face expression recognition," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition(CVPR)*, 2003, pp. 346–352.

[31] I. Sandin, G. Andrade, F. Viegas, D. Madeira, L. Rocha, T. Salles, and M. Gonçalves, "Aggressive and effective feature selection using genetic programming," in *IEEE Congress on Evolutionary Computation (CEC)*, 2012, pp. 1–8.

[32] S. Ahmed, M. Zhang, and L. Peng, "Enhanced feature selection for biomarker discovery in lc-ms data using GP," in *IEEE Congress on Evolutionary Computation (CEC)*, 2013, pp. 584–591.

[33] R. Suarez, J. Valencia-Ramirez, and M. Graff, "Genetic programming as a feature selection algorithm," in *the 2014 IEEE International Autumn Meeting on Power, Electronics and Computing (ROPEC)*, Nov, pp. 1–5.

[34] D. Y. Harvey and M. D. Todd, "Automated feature design for numeric sequence classification by genetic programming," *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 4, pp. 474–489, 2015.

[35] J. H. Moore, D. P. Hill, A. Saykin, and L. Shen, "Exploring interestingness in a computational evolution system for the genome-wide genetic analysis of alzheimers disease," in *Genetic Programming Theory and Practice XI*. Springer, 2014, pp. 31–45.

[36] Q. Chen, B. Xue, B. Niu, and M. Zhang, "Improving generalisation of genetic programming for high-dimensional symbolic regression with feature selection," in *IEEE Congress on Evolutionary Computation (CEC)*, 2016, pp. 3793–3800.

[37] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[38] J. R. Quinlan, "Induction of decision trees," *Machine learning*, vol. 1, no. 1, pp. 81–106, 1986.

[39] R. J. Quinlan, "C4.5: Programs for machine learning," 1992.

[40] R. Quinlan, "Data mining tools See5 and C5.0," 2004.

[41] W. Banzhaf, F. D. Francone, and P. Nordin, "The effect of extensive use of the mutation operator on generalization in genetic programming using sparse data sets," in *Parallel Problem Solving from NaturePPSN IV*. Springer, 1996, pp. 300–309.

[42] M. J. Cavaretta and K. Chellapilla, "Data mining using genetic programming: the implications of parsimony on generalization error," in *IEEE Congress on Evolutionary Computation (CEC)*, 1999, pp. 1330–1337.

[43] J. Fitzgerald and C. Ryan, "On size, complexity and generalisation error in gp," in *Proceedings of the 16th Annual Conference on Genetic and Evolutionary Computation Conference (GECCO)*, 2014, pp. 903–910.

[44] Q. Chen, B. Xue, L. Shang, and M. Zhang, "Improving generalisation of genetic programming for symbolic regression with structural risk minimisation," in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, 2016, pp. 709–716.

[45] L. Chen, C. Chu, T. Huang, X. Kong, and Y.-D. Cai, "Prediction and analysis of cell-penetrating peptides using pseudo-amino acid composition and random forest models," *Amino acids*, vol. 47, no. 7, pp. 1485–1493, 2015.

[46] S. Luke, L. Panait, G. Balan, S. Paus, Z. Skolicki, E. Popovici, K. Sullivan, J. Harrison, J. Bassett, R. Hubley *et al.*, "A java-based evolutionary computation research system," *Online (March 2004) http://cs. gmu. edu/~ eclab/projects/ecj*, 2004.

[47] A. Liaw and M. Wiener, "Classification and regression by randomforest," *R news*, vol. 2, no. 3, pp. 18–22, 2002.

[48] M. Kuhn, S. Weston, N. Coulter, and R. Quinlan, "C50: C5. 0 decision trees and rule-based models," *R package version 0.1. 0-21, URL http://CRAN. R-project. org/package C*, vol. 50, 2014.

[49] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 267–288, 1996.

[50] J. Friedman, T. Hastie, and R. Tibshirani, "Regularization paths for generalized linear models via coordinate descent," *Journal of statistical software*, vol. 33, no. 1, p. 1, 2010.

[51] S. Stijven, W. Minnebo, and K. Vladislavleva, "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 on Genetic and Evolutionary Computation Conference (GECCO)*, 2011, pp. 623–630.

[52] M. Keijzer, "Improving symbolic regression with interval arithmetic and linear scaling," in *Proceedings of the 6th European conference on Genetic Programming (EuroGP)*, 2003, pp. 70–82.

[53] M. Lichman, "UCI machine learning repository," 2013. [Online]. Available: http://archive.ics.uci.edu/ml

[54] L. Vanneschi, M. Castelli, and S. Silva, "Measuring bloat, overfitting and functional complexity in genetic programming," in *Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation Conference (GECCO)*, 2010, pp. 877–884.

[55] L. Vanneschi, S. Silva, M. Castelli, and L. Manzoni, "Geometric semantic genetic programming for real life applications," *Genetic Programming Theory and Practice XI*, p. 191, 2014.

[56] L. Vanneschi, M. Castelli, L. Manzoni, and S. Silva, "A new implementation of geometric semantic GP and its application to problems in pharmacokinetics," in *Proceedings of the 16th European conference on Genetic Programming (EuroGP)*, 2013, pp. 205–216.

[57] A. Rosenwald, G. Wright, W. C. Chan, J. M. Connors, E. Campo, R. I. Fisher, R. D. Gascoyne, H. K. Muller-Hermelink, E. B. Smeland, J. M. Giltnane *et al.*, "The use of molecular profiling to predict survival after chemotherapy for diffuse large-b-cell lymphoma," *New England Journal of Medicine*, vol. 346, no. 25, pp. 1937–1947, 2002.

[58] C. Leys, C. Ley, O. Klein, P. Bernard, and L. Licata, "Detecting outliers: Do not use standard deviation around the mean, use absolute deviation around the median," *Journal of Experimental Social Psychology*, vol. 49, no. 4, pp. 764–766, 2013.

[59] I. Gonçalves and S. Silva, "Balancing learning and overfitting in genetic programming with interleaved sampling of training data," in *Proceedings of the 16th European conference on Genetic Programming (EuroGP)*, 2013, pp. 73–84.

[60] J. Fitzgerald, R. Azad, and C. Ryan, "A bootstrapping approach to reduce over-fitting in genetic programming," in *Proceedings of the 15th Annual Conference on Genetic and Evolutionary Computation Conference (GECCO)*, 2013, pp. 1113–1120.

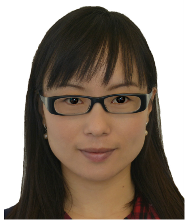
**Qi Chen** received the B.E. degree in Automation from the University of South China, Hunan, China in 2005 and the M.E. degree in Software Engineering from Beijing Institute of Technology, Beijing, China in 2007. Since 2014, she has joined the Evolutionary Computation Research Group(ECRG) at Victoria University of Wellington (VUW). Currently, she is a Ph.D. candidate in School of Engineering and Computer Science at VUW.

Qi's current research mainly focus on genetic programming for symbolic regression. Her research interests including machine learning, evolutionary computation, feature selection, feature construction, transfer learning, domain adaptation and statistical learning theory. She serves as a reviewer of international conferences, including IEEE Congress on Evolutionary Computation, and international journals, including IEEE Transactions on Evolutionary Computation and Journal of Heuristics.

**Mengjie Zhang** Mengjie Zhang (M04-SM10) received the B.E. and M.E. degrees from Artificial Intelligence Research Center, Agricultural University of Hebei, Hebei, China, and the Ph.D. degree in computer science from RMIT University, Melbourne, VIC, Australia, in 1989, 1992, and 2000, respectively.

Since 2000, he has been with the Victoria University of Wellington, Wellington, New Zealand, where he is currently Professor of Computer Science, Head of the Evolutionary Computation Research Group, and the Associate Dean (Research and Innovation) in the Faculty of Engineering. His current research interests include evolutionary computation, particularly genetic programming, particle swarm optimization, and learning classifier systems with application areas of image analysis, multiobjective optimization, feature selection and reduction, job shop scheduling, and transfer learning. He has published over 350 research papers in refereed international journals and conferences.

Prof. Zhang has been serving as an Associated Editor or Editorial Board Member for ten international journals (including IEEE Transactions on Evolutionary Computation, IEEE Transactions on Cybernetics, Evolutionary Computation Journal, and IEEE Transactions on Emergent Topics in CI) and as a Reviewer of over 20 international journals. He has been serving as a Steering Committee Member and a Program Committee Member for over 100 international conferences. He has supervised over 50 postgraduate research students. He is the Chair of the IEEE CIS Intelligent Systems and Applications Technical Committee, a member of the IEEE CIS Evolutionary Computation Technical Committee, a Vice-Chair of the IEEE CIS Task Force on Evolutionary Computer Vision and Image Processing, a Vice-Chair of the IEEE CIS Task Force on Evolutionary Computation for Feature Selection and Construction, a member of IEEE CIS Task Force of Hyper-heuristics, and the Founding Chair for IEEE Computational Intelligence Chapter in New Zealand.



**Bing Xue** Bing Xue is currently a Senior Lecturer in School of Engineering and Computer Science at Victoria University of Wellington. Her research focuses mainly on evolutionary computation, feature selection, feature construction, multi-objective optimisation, data mining and machine learning.

Dr Xue is an Associate Editor/member of Editorial Board for five international journals including IEEE Computational Intelligence Magazine, Applied Soft Computing, International Journal of Swarm Intelligence, and International Journal of Computer Information Systems and Industrial Management Applications. She is a Guest Editor for the Special Issue on Evolutionary Feature Reduction and Machine Learning for the Springer Journal of Soft Computing. She is also a Guest Editor for Evolutionary Image Analysis and Pattern Recognition in Journal of Applied Soft Computing.She is serving as a reviewer of over 15 international journals including IEEE Transactions on Evolutionary Computation and IEEE Transaction on Cybernetics.

She has been a chair for a number of international conferences including the Lead Chair of IEEE Symposium on Computational Intelligence in Feature Analysis, Selection, and Learning in Image and Pattern Recognition (FASLIP) at SSCI 2016 and 2017, a Program Co-Chair of the 7th International Conference on Soft Computing and Pattern Recognition (SoCPaR2015), Special Session Chair for The 20th Asia-Pacific Symposium on Intelligent and Evolutionary Systems (IES2016), a Tutorial Chair for the 30th Australian Joint Conference on Artificial Intelligence, publicity chair for the international conference on Simulated Evolution And Learning (SEAL) 2017. She is the organiser of the special session on Evolutionary Feature Selection and Construction in IEEE Congress on Evolutionary Computation (CEC) 2015, 2016 and 2017 and SEAL 2014.

Dr Xue is currently the Chair of the IEEE Task Force on Evolutionary Feature Selection and Construction. She is chairing the IEEE CIS Graduate Student Research Grants Committee. She is also serving as the Director of Women in Engineering for the IEEE New Zealand Central Section and the Secretary of the IEEE Chapter on Computational Intelligence in that Section.