Faculty of Natural and
Mathematical Sciences
Department of Informatics

7CCSMPRJ

Individual Project Submission 2021/22

Name:	Shilin Zhang
Student Number:	20057905
Degree Programme:	MSc Computational Finance
Project Title:	Optimal Scoring-Based Lag Selection for Time Series
Supervisor:	Professor Carmine Ventre
Word Count:	9416

Plagiarism Statement

All work submitted as part of the requirements for any examination or assessment must be expressed in your own words and incorporates your own ideas and judgements. Plagiarism is the taking and using of another person's thoughts, words, judgements, ideas, etc., as your own without any indication that they are those of another person.

Plagiarism is a serious examination offence. An allegation of plagiarism can result in action being taken under the *B3 Misconduct Regulations*.

I acknowledge that I have read and understood the above information and that the work I am submitting is my own.

Signature: **Date:** August 8, 2022

Shilin Zhang Aug 8th 2022

Department of Informatics
King's College London
WC2R 2LS London
United Kingdom

Optimal Scoring-Based Lag Selection for Time Series

Shilin Zhang
Student Number: 20057905
Course: MSc Computational Finance

Supervisor: Professor Carmine Ventre

Thesis submitted as part of the requirements for the award of the MSc in Computational Finance.
7CCSMPRJ - MSc Individual Project - 2022

Abstract

In time series prediction problem, it is essential to determine the proper number of past observations (lags) included as features. The optimal lags could be selected by applying greedy methods, heuristic methods, statistical methods or other methods. In this study, a robust and fast quadratic programming lag selection (QPLS) method is proposed. The main idea of the method is to minimize the redundancy within features and maximize the relevance between features and target. In QPLS, the optimization problem is converted into the quadratic programming format and solved by a quadratic programming solver. The proposed approach assigns scores to all features, including the lagged ones instead of dichotomously deciding to include or exclude features from the feature set. Based on these scores, the features are selected and trained with various prediction models. The root mean square error is used to evaluate the predictive performance of models. In the experiments conducted, several criteria are used to find the one that most suits the proposed method. The usefulness and efficiency of the new lag-selection technique are shown by comparing lag-selection methods via genetic algorithm and partial correlation function.

Contents

List of Figures

List of Tables

Acknowledgements

I would like to thank my supervisor Prof Carmine Ventre for providing guidance and feedback throughout this project.I would also like to thank Lingbo, Vitali Avagyan, Vardan Voskanyan and Buhong Liu. They continuously provided encouragement and was always willing and enthusiastic to assist in any way they could throughout the research project. Many thanks to all participants that took part in the study and enabled this research to be possible.

1 Introduction

Time series data is a series of data points listed in time order and it is easy to access in our daily life. And this kind of series comes up quite often in our daily life. The modelling of this kind of data is a very essential part of making better use of this kind of data source. In time series analysis, the primary goal is to use past values to predict future values. While we want to make input data as informative as possible, in the meantime we do also aim to choose the most representative set of features in the time-series regression training process. Ideally, we want a proper number of past observations of the target variable. This number should be as small as possible while capturing unique features within the data. Too many selected lags will affect the learning or prediction capability of the models and slow down the speed of the model training process. Similarly, too few selected lags will only contain limited features of the search space which is not enough to get a good prediction model. Therefore, determining the lags used in the model building process and finding the minimum number of essential lags is one of the necessary works influencing the quality or the predictive performance of the models.

1.1 Aims and Objectives

Given the importance of the lag selection, this study aims to find a scoring-based method of determining the optimal lags for different kinds of time series data from different industries. The method is used on different kinds of regression models such as linear regression models (LR), random forest regression models (RF) and support vector regression models (SVR). By applying the proposed lagged feature selection method, the features which contribute the most to the quality of the model will be preserved. As a result, the model training process will be faster and the trained model will be of higher quality. To achieve the aims, the following objectives introduced in depth later are listed below and this study will carry out the following steps:

- Build a quadratic programming lagged feature selection method to determine the optimal lag.
- Build a heuristic algorithm lagged feature selection method to determine the optimal lag.
- Build a statistics-based method for lag selection.
- Compare all these models on different datasets and regression models to select the optimal lags.

1.2 Literature Review

Many lag selection approaches have been proposed in the literature to improve the performance of the model using the selected lags as well as the efficiency of the selection process. It has been shown in the Econometrics textbook [1] that taking into account 30 lags is enough to capture the pattern of the time series data. Because of the mentioned reasons, this study makes a practical decision to use lags of up to thirty.
Traditional lag selection methods are mainly the method based on the statistical test such as the likelihood ratio (LR), Lagrange Multiplier (LM), and Wald test [2]. Other traditional methods are based on the information criterion (IC) including Schwarz Information Criterion (SIC), the Hannan-Quinn Criterion (HQC), Akaike Information (AIC), Bayesian Information Criterion (BIC) and Autocorrelation Criterion (AC) [3]. A drawback of this kind of test-based method is that the method is only suitable for nested models. Under the assumption that the errors of related equations are normally distributed, the statistics of the tests are roughly regarded as the chi-square distribution with the degrees of freedom as the total number of restrictions. For a good criterion, there must be a balance between the model complexity (which is related to the systemic errors) and the model variability (which is associated with the noise of the data set according to [4]). There is a lot of research related to finding the optimal lags for vector autoregressive (VAR) models. In 2003, Hatemi proposed a method combining BIC and HQC criteria in both stable and unstable VAR models. And the experiment result shows that the method could perform well in a small dataset with a sample size of no more than 40. In 2008, Scott and Abdulnasser introduced corrected Akaike (AICC) and Schwarz Bayesian (SBC) criteria to this problem. They found that the accuracy and forecast performance regarding SBC-based lag selection is the best.[5] For the circumstances that different criteria give the different lags, the likelihood ratio test could help us pick the optimal lag for the SVR model according to Hatemi and Hacker in 2009 [6].

There are also methods based on the information theory which could be applied to the inputs of the model without considering the model outputs. In 2006, Geoffroy proposed a lag selection method related to high-dimensional mutual information in the regression model. By using this kind of estimator, MI with more than two variables could be calculated [7]. In 2009, Oswaldo, Urbano and Rui applied the cross entropy function (XEF) to select lags because that XEF is able to describe the non-linear relationship between the input and output which is more appropriate than cross correlation function (XCF) on non-linear relationships. They also combined the joint conditional entropy and indirect genetic algorithm[8]. In 2011, Ribeire, Neto, Cavalcanti and Tsang developed

a method based on Frankenstein's Particle Swarm Optimization (FPSO) which works reasonably well regardless of the size of the dataset and the type of relationships (linear or non-linear) among variables. By using this method, a set of optimal lags can be selected and produced the best result on the SVR model compared with the original Particle Swarm Optimization feature selection[9]. According to the research by Oliveira, they design a hybrid evolutionary system based on a global optimization method for selecting both lags and the parameters on autoregressive integrated moving average (ARIMA) models and a support vector machine (SVM) models [10]. In 2016, Widodo et al proposed a method based on multiple kernel learning (MKL) method which could select the optimal lags or size of sliding windows for forecasting models automatically. They showed that one can automatically select either the optimal lags or the size of the sliding windows in forecasting models Besides, the method could be used to predict the future values of time series data without extra computation of the seasonality [11].

The best solution to the lag selection problem depends on a machine-learning prediction model and domain-specific dataset. For example, Hossein and Mohammad found that the optimal lag for the Ljung-Box test depends on the length of the time series as well as the level of the test [12]. Therefore, we could regard the lag selection problem as the lagged feature selection problem and solve the accordingly by using the method of solving the feature selection problem. Furthermore, optimal lags could be selected via selecting the lagged features.

Generally, all the feature selection methods could be divided into three kinds: filter, wrapper, and embedded methods. Li et al surveyed feature selection algorithms in 2016, which gives a systematic analysis of filter, wrapper, and embedded methods. The existing feature selection methods includes heuristics method [13] [14], greedy search method [15] and regularization method [16]. A drawback of these methods is that the user must consider the speciality of the data set. Besides, the mutual information-based method is also widely used in data mining. In 2012, Brown et al listed 17 feature selection methods related to mutual information in their survey [17]. This study considers feature selection methods based on scoring functions that estimate the quality of a feature subset, such as least angle regression (LARS) [18], Ridge [19], and the Elastic Net [20], Lasso [21], and some sequential search methods, such as forward selection [22] and the genetic algorithm [23]. The weighted sum of the l_2 norm of the residuals and the l_1 norm of the parameter vector is the Lasso scoring function. This scoring algorithm penalises big components in the parameter vector while providing a good approximation to the target vector. Additionally, the generated parameter vector is made sparser by the parameter vector's l_1 norm, which conducts feature selection. Ridge employs the l_2 norm rather

than the l_1 norm for scoring, which is similar to Lasso. Although it doesn't provide a sparse parameter vector and chooses features less aggressively than Lasso, this method increases the stability of the result. A linear combination of the l_1 and l_2 norms of the parameter vector is used by the Elastic Net [20] as a penalty for the residual norm. This penalty allows us to combine the advantages of both Lasso and Ridge. Tuning the weights corresponding to the penalty terms and taking into consideration the structure of a data set are two prevalent issues with these feature selection approaches. Aha and Bankert [24] researched on sequential search-based feature selection techniques. One of such selection techniques is the forward-selection algorithm which begins with an empty feature set and sequentially adds a single feature on each iteration based on the importance determined by an F-test, whereas the genetic algorithm uses a random search that maximises the objective function of the classifier accuracy and adds or removes some features on each iteration.

2 Background Theories

2.1 Architecture of the Prediction System

The prediction system 2 consists of the lag selection part and prediction model part. The first part of the system aims to select the best-lagged feature for the regression model. To achieve this goal, we apply the minimum redundancy maximum relevance method as well as quadratic programming or genetic algorithm. For the second part, we construct a linear regression model, random forest regression model, elastic net model or support vector regression model to evaluate the performance of those models with selected features. The performance of the whole pipeline is calculated by using Root Mean Squared Error (RMSE). The input of the system is the shifted data set in which all features are shifted by a time step to avoid data leakage. Dataset is then combined with all possible lags to construct the search space containing all candidate features. The following step includes the optimization method to optimize the objective function to be described in upcoming sections.

2.2 Feature Selection

Feature selection is a process of selecting a subset of relevant features(sometimes referred to as variables or predictors in this study) for use in model construction. This could help us simplify the models and make them more interpretable [1], shorten the time cost of the model training process, avoid the curse of dimensionality [2] and improve the compatibility of data with a learning model class. For the search space constructed by all the lagged feature $X = [X_1, X_2, \cdots, X_n] \in \mathbb{R}^{m \times n}$, where X_i is the ith feature in the matrix, the feature selection problem could be regarded as finding the subset of the search space that makes the objective function reach the optimal or near optimal status. To find the subset $X^* \in \mathbb{R}^n$ such that

$$X^* = \underset{X \in \mathbb{R}^n}{argmin}\ obj(X) \tag{2.1}$$

where obj is the objective function, which could be used to measure the feature performance or feature importance score with given the selected feature subset and the search space.

There are different feature selection methods proposed by academia but there is no generalized way of solving all feature selection problems. Therefore, the best feature

[1] A machine learning model is interpretable if we can fundamentally understand how it arrived at a specific decision

[2] In data mining, the curse of dimensionality refers to a data set with too many features.

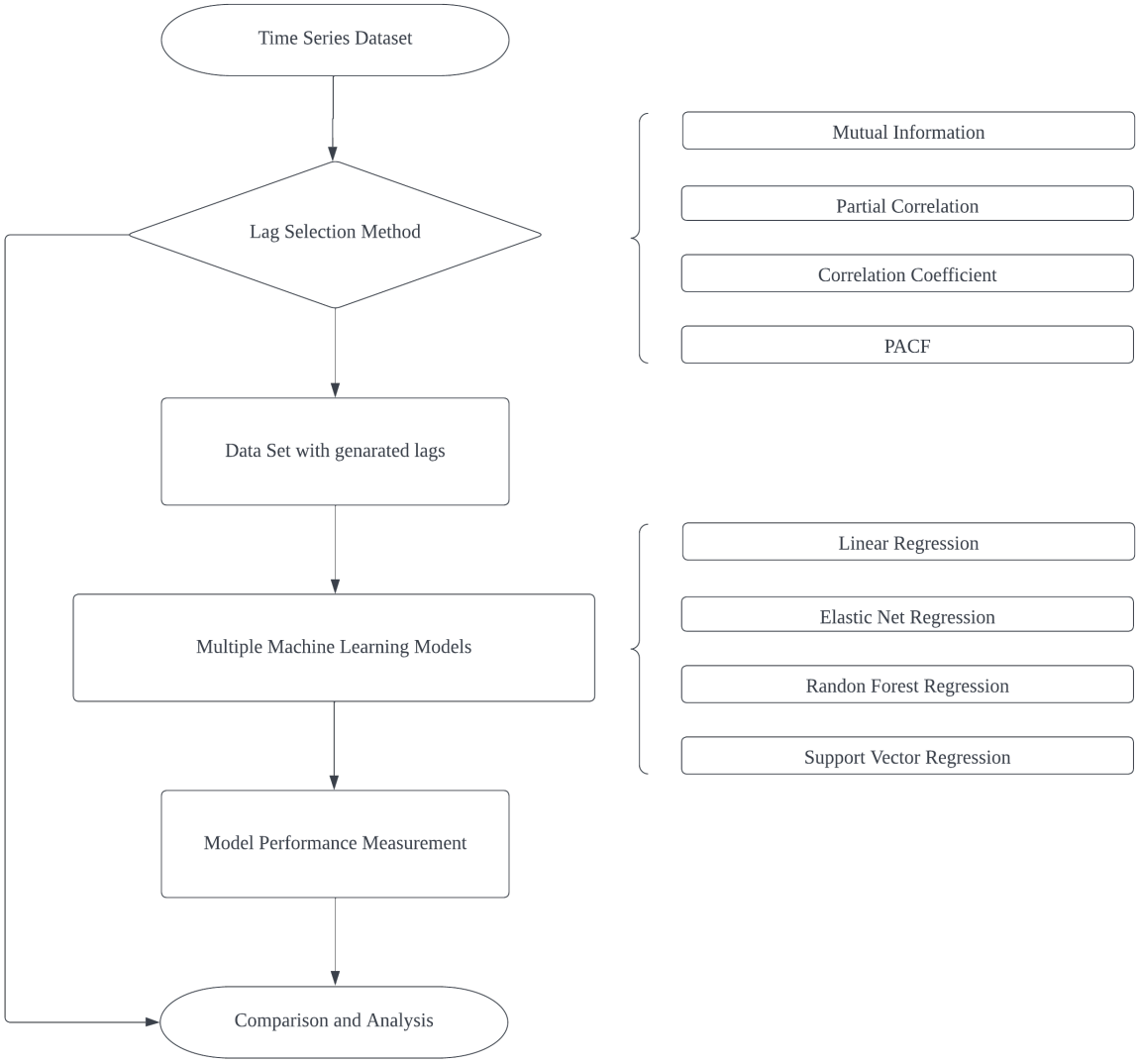


Figure 1: Architecture of the Prediction System

selection method varies from situation to situation.
Generally, there are wrapper, filter and embedded methods for feature selection [25]. To score and select feature subsets, the wrapper approaches employ the predictive models. Each new subset is used to train a model, which is tested on a holdout set. In the hand-out method, the training set is used to train the model while the unseen test set is used to evaluate how well the model works. The score for that subset can be calculated by counting the number of errors made on that hold-out set (the model's error rate). Wrapper approaches are very computationally costly because they train a new model for each subset, but they typically produce the highest-performing feature set for that specific model or problem.
The error rate is not used in filter methods, instead, a proxy measure is used to score a feature subset. This measurement was chosen since it is quick to compute and captures the value of the feature collection. Common metrics include inter-class distance, intra-class distance, relief-based techniques, mutual information, pointwise mutual information, Pearson product-moment correlation coefficient, and scores from significance tests for each class/feature combination [26]. Filters are usually less computationally intensive than wrappers, but they produce a feature set which is not tuned to a specific type of predictive model [27]. The lack of tuning means a feature set from a filter is more general than the set from a wrapper, usually giving lower prediction performance than a wrapper. Although filters often use less computing power than wrappers, they create a feature set that is not tuned to a particular kind of prediction model. Due to the lack of tuning, a filter's feature set is generally more generic than a wrapper's, which results in worse prediction performance. The feature set is more beneficial for revealing the links between the characteristics because it does not include the underlying assumptions of a prediction model. Instead of providing a clear best feature subset, many filters offer a feature ranking, with cross-validation used to determine the cutoff point. To enable the employment of wrapper methods in more complex situations, filter techniques have also been applied as a preprocessing step. One more widely used method is the Recursive Feature Elimination algorithm, which is frequently combined with Support Vector Machines to iteratively build a model and eliminate features with low weights. The AutoViML team developed a new feature selection algorithm SULOV A.4 (Searching for Uncorrelated List of Variables). The algorithm is based on the Minimum-Redundancy-Maximum-Relevance (MRMR) and recursive XGBoost method.
Embedded methods refer to a broad class of methods that execute feature selection throughout the model construction process. The LASSO method for building a linear model, which penalises the regression coefficients with an l_1 penalty and shrinks many of

them to zero, is an example of this strategy. The LASSO algorithm selects any features that have non-zero regression coefficients. Elastic net regularisation, which combines the l_1 penalty of LASSO with the l_2 penalty of Ridge regression. Bolasso (which bootstraps samples) and FeaLect (which scores all the features based on combinatorial analysis of regression coefficients) are used to extend LASSO [28]. With autoencoders, AEFS [29] expands LASSO to nonlinear scenarios. In terms of computing complexity, these methods often fall between filters and wrappers. A feature selection method proposed by Peng [30] could be used to make the feature subset have the minimum redundancy within itself as well as the maximum relevance with the target. Hence the objective function for the problem could be expressed as:

$$\max_x [Rel(x) - Red(x)] \tag{2.2}$$

where the *Red* and *Rel* are the redundancy and relevance evaluation function. While the mRMR is an incremental greedy search method which consists of two separate stages, it cannot find the best feature subset in one stage. Therefore, the feature selected in the first selection stage could not be deleted in the second selection stage. A quadratic programming feature selection which is designed to solve the global optimization problem could help to solve the problem.

2.3 Quadratic Programming

Quadratic programming (QP) is the process of solving certain mathematical optimization problems involving quadratic functions. Specifically, one seeks to optimize (minimize or maximize) a multivariate quadratic function subject to linear constraints on the variables. A standard strictly convex problem involving quadratic functions could be written as below:

$$\min_x \quad f(x) = \frac{1}{2} x^T G x - a^T x \tag{2.3}$$

$$\text{s.t.} \quad C^T x \geq b \tag{2.4}$$

where x and a are vectors with length n, G is an $n \times n$ symmetric positive definite matrix, and superscript T denotes the transpose.
For different problems, the weight of the quadratic part and linear part could be tweaked to achieve a good result. Therefore, the weighted quadratic objective function could be denoted as below:

$$\min_x \{\frac{1}{2}(1-\alpha) x^T G x - \alpha a^T x\} \tag{2.5}$$

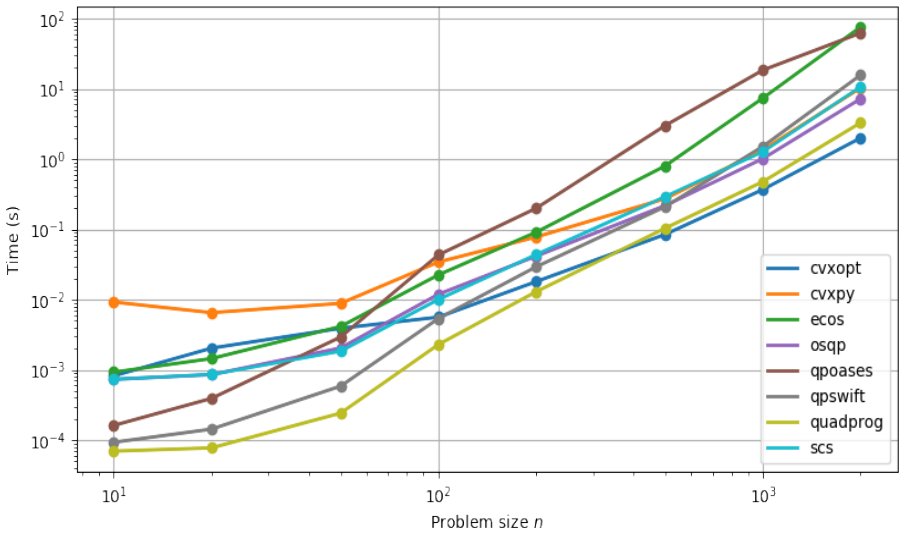


Figure 2: Comparison of Quadratic Programming Solvers

where x, G, a are the same as defined before and $\alpha \in [0, 1]$. For the circumstance that $\alpha = 0$, the objective function only considers the quadratic part. If $\alpha = 1$, the objective function only includes the linear term. The numerically dual algorithm proposed by Goldfarb and Idnani [31] could be used to solve the unconstrained problem effectively compared with primal algorithms according to the test on randomly generated test problems. Based on the algorithm proposed above, the package quadprog has been developed in Python by Stéphane Caron A.1. Furthermore, the speed of solving the problem by quadprog is faster than by other python packages according to the experiment result on random dense problems which is shown in the picture below (each data point corresponds to an average over 10 runs) A.1:

2.4 Scoring Criteria

Different criteria could give us different results even for the same feature selection process. Most of the methods can use mutual information, correlation, or distance/similarity scores to select features.

2.4.1 Correlation Coefficient

The linear relationship between the feature X and Y could be described by using the Pearson correlation coefficient. The Pearson correlation between feature X_i and X_j could be defined as

$$\rho_{XY} = \frac{Cov(X,Y)}{Std(X)Std(Y)} \tag{2.6}$$

where $Cov(X,Y)$ is the covariance between feature X and Y, and $Std(\cdot)$ is the standard deviation of the features. The correlation coefficient of the sample could be calculated by using the formula

$$\hat{\rho}_{xy} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})}} \tag{2.7}$$

2.4.2 Mutual Information

The concept of mutual information is from information theory, thus the basic concept of the subject needs to be introduced first. Entropy is a quantitative measurement of uncertainty which means the lower the entropy, the less uncertainty the system. or is about the variable according to Shannon. For a random variable X, it could be defined as:

$$H(X) = -\sum_{x} P_X(x) log P_X(x) \tag{2.8}$$

where $P_X(x)$ is the probability distribution of variable X.
When the concept of entropy expands to two variables, there is conditional entropy which marks the uncertainty of the system under the condition of being influenced by other variables. The conditional entropy represents the average uncertainty of variable X after observing another variable Y, which could be shown as:

$$H(X|Y) = \sum_{y} P_Y(y)[-\sum_{x} P_{X|Y}(x|y) log(P_{X|Y}(x|y))] \tag{2.9}$$

Mutual information is one of the criteria which measures how much the reduction in uncertainty related to variable X. For two features X and Y whose joint probability distribution are $P_{XY}(x,y)$, the mutual information between them could be denoted as:

$$I(X;Y) = H(X) - H(X|Y) = \sum_{x,y} P_{XY}(x,y) \log \frac{P_{XY}(x,y)}{P_X(x)P_Y(y)} \tag{2.10}$$

where $P_X(x)$ and $P_Y(y)$ are the marginals.

$$P_X(x) = \sum_y P_{XY}(x, y) \tag{2.11}$$

where the values of features X and Y are discrete. In consequence, the higher mutual information means the more elimination of the uncertainty concerning variable X after observing Y.

2.4.3 Partial Correlation

In statistics, the partial correlation function (PACF) is used to measure the association relationship between two variables without the effect of other random variables. In multivariate time series analysis, the partial correlation between one feature and the target could be more accurate compared with the correlation coefficient because the PACF could avoid misleading information from other features. The PACF between feature X and feature Y with one confounding factor Z could be denoted as:

$$\rho_{XY.Z} = \frac{\rho_{XY} - \rho_{XZ}\rho_{YZ}}{\sqrt{(1 - \rho_{XZ}^2)(1 - \rho_{YZ}^2)}} \tag{2.12}$$

where the ρ_{XY} is the correlation coefficient between X and Y. For multiple controlling variables $Z = \{Z_1, Z_2, \cdots, Z_n\}$, the correlation is calculated between the residuals resulting from the linear regression of X with Z and of Y with Z.

2.5 Benchmark Methods

To compare the result generated from the proposed method and other methods of selecting lags, we use the genetic algorithm and partial autocorrelation function as the benchmark in this study.

2.5.1 Genetic Algorithm

The genetic Algorithm is one kind of evolutionary algorithm which simulates the process of natural gene selection. The method consists of biological operators such as mutation operator, crossover operator and selection operator. GA is commonly used in generating solutions for optimization problems. In this algorithm, the solutions are always called individuals or chromosomes and each element of the solution is called a gene. In the traditional encoding method, the gene is a binary value and the chromosome is a binary

string of 0s and 1s. Generally, the algorithm has two necessary core parts: a genetic representation of the solution and a fitness function designed for the solution evaluation. First, the evolution process starts from a randomly generated group of chromosomes which is called the population. The population which put into the iterations later often covers the entire search space of the solutions. Each iteration represents the generation of the population. Second, the subset of the population will be selected via an objective function to generate new generations. In this step, the fitness of every chromosome will be calculated via the user-designed objective function. Third, a portion of the best chromosomes according to the objective function will be selected in each iteration. Meanwhile, a small portion of fewer fit chromosomes also will be selected to keep the diversity of genes in the population. Fourth, to avoid the locally optimal result, there are genetic operations such as mutation and crossover operation on the selected group [23]. In this process, a pair of parent chromosomes are chosen to generate one child chromosome until the population for the next generation is generated. As the parents are selected by previous evaluation, the average score of the next generation will be increased. Last, the algorithm repeats the process from the second step to the fourth step until it meets the termination condition. The normal termination condition includes reaching the maximum number of iterations, finding the solution satisfying the minimum criteria and finding the highest ranking solution.

2.5.2 Partial Autocorrelation Function

For a stationary time series, the partial autocorrelation function (PACF) is a useful tool to determine the lag values of the variables since the plot of PACF with lags shows the variable's high partial auto-correlation characteristic on some lags given the confidence interval. By using this method, the best lag order p of the variable could be found easily, especially for AR model and ARIMA model. For stationary time series z_t, the partial autocorrelation could be denoted as follow:

$$\begin{aligned} &\phi_{1,1} = corr(z_{t+1,z_t}), \text{for } k = 1, \\ &\phi_{k,k} = corr(z_{t+k} - \hat{z}_{t+k}, z_t - \hat{z}_t), \text{for } k \geq 2, \end{aligned} \tag{2.13}$$

where k is the lag of the partial autocorrelation, $\phi_{k,k}$ is the autocorrelation between z_t and z_{t+k} with the linear dependence of z_t on z_{t+1} through and z_{t+k-1} removed [32].
For specific models, the PACF shows the specific patterns. For instance, the partial autocorrelation is 0 with no relation to lag values for the data fitting the white noise model. For the data fitted with an autoregressive model with order p, the partial autocorrelation

is not equal to zero when lag is less than or equal to p and 0 for lags are greater than p. For the data fitted with the moving average model, the partial autocorrelation could either oscillates to 0 or geometrically decays to 0. For the data fitted with ARMA(p, q) model, the partial autocorrelation geometrically decays to 0 when the lags greater than p.

In this study, we use the PACF to measure the autocorrelation of every variable separately. First, we make the stationarity check for every original feature. In this study, we use the Augmented Dickey–Fuller (ADF) test to find out if the series is stationary. If the series is stationary, we move to the next step. If the series is not stationary, we apply the differencing operation on the series until the differenced series becomes stationary. Second, we calculate the PACF over all the lags less than or equal to thirty for every single series. Third, the lag with the partial correlation of larger than the significant threshold is selected. In this study, the threshold is set to 0.05. Finally, the selected lagged features would be used for model training and testing and the time series prediction process.

2.6 Regression Model

In the prediction model selection, we have considered three types of models: the tree-based models, linear models and kernel-based models. For each type of models, we select one representative model in this study. They are the random forest regression model, elastic net regression model and support vector regression model.

2.6.1 Random Forest Regression

The Random Forest model is a kind of bagging algorithm that connects both ensemble learning methods and the decision tree framework to create multiple decision trees. The model combines the results to generate a new result to reach strong predictions/classifications. The boosting algorithm and bagging algorithm are the two main algorithms for ensemble learning algorithms. The main purpose of ensemble learning is to train multiple models over the same data and reach a powerful average performance of the models. And the error of each model is independent and different from each other. In practice, the models could be decision trees, support vector machines and some other models. In a random forest algorithm, the series of models are decision trees. The decision tree is a method used in both regression and classification problems. In a regression problems, the tree starts from the root and generates follows the branches of judgement until reaching a leaf node. To achieve randomness in the random forest, the algorithm

designer has introduced bootstrapping. Bootstrapping is a random sampling method over a dataset. Given the number of iterations and number of features, the method will generate multiple subsets for ensemble learning.

2.6.2 Elastic Net Regression

Elastic net linear regression combines both the least absolute shrinkage and selection operator(LASSO) and Ridge regression methods by setting the weight of regularization to balance the two kinds of models. The model applies the penalties from both the lasso and Ridge regression to regularize regression models. For the lasso regression method, the limitation is that when the input data is a highly correlated group, the method tends to select too few or one variable from the group and ignore the other variables. The situation is not what we want especially for the high dimensional input. To solve the limitation problem, the elastic net regression includes a quadratic part in the penalty parameter λ. The criterion of the model could be expressed as follow:

$$L(\lambda_1, \lambda_2, \beta) = |y - X\beta|^2 + \lambda_2|\beta|^2 + \lambda_1|\beta|_1 \tag{2.14}$$

where the β are the coefficients of the features as known as the feature importance score and

$$\begin{aligned} |\beta|^2 &= \sum_{j=1}^{p} |\beta|_j{}^2 \\ |\beta|_1 &= \sum_{j=1}^{p} |\beta|_j \end{aligned} \tag{2.15}$$

There are two special cases for the elastic net regression model, when $\lambda_1 = \lambda, \lambda_2 = 0$, the model is LASSO regression model. Similarly, when $\lambda_2 = \lambda, \lambda_1 = 0$, the model is Ridge regression model. Hence, by setting proper λ_1 and λ_2, the model could keep a balance of both penalties to lead to a better performance than LASSO and Ridge regression.

2.6.3 Support Vector Regression

The support vector regression (SVR) is a model based on the logic of the support vector machine (SVM). The model is proposed by Vapnik in 1999 [33] based on the target of minimizing the upper bound of the generalized error. For the SVR model trained over the training set $\{x_i, y_i\}_{i=1}^{l}$, where $x_i \in R^d$ is the ith input vector, d is the dimension of the input data set, l is the length of the training samples. The possible regression

function could be denoted as follow:

$$\{f|f(x) = w^T x + b, \omega \in R^d, b \in R\} \tag{2.16}$$

where b is the bias of the function, ω is the weight of the vector estimated via minimizing following equation which is also known as regularized risk function

$$\frac{1}{2}||w||^2 + C\sum_{i=1}^{l} L(y_i, f(x_i)) \tag{2.17}$$

To find the best function, the loss function L introduce the tolerance parameter ϵ, then the loss function become:

$$L(y_i, f(x_i)) = \begin{cases} 0, & |f(x) - y| < \epsilon \\ |f(x) - y| - \epsilon, & \text{otherwise} \end{cases} \tag{2.18}$$

Then the problem is transferred into minimizing

$$\frac{1}{2}||w||^2 + C\sum_{i=1}^{l}(\xi_i - \xi_i^*) \tag{2.19}$$

, subject to

$$\begin{cases} w^T x_i + b - y_i \leq \epsilon + \xi_i^* \\ y_i - W^T x_i - b \leq \epsilon + \xi_i^* \\ \xi_i, \xi_i^* \geq 0, i = 0, \cdots, l \end{cases} \tag{2.20}$$

where ξ_i, ξ_i^* are slack variables measuring if the error is over the predefined ϵ range.
Another essential part of the SVR model is the selection of the kernel function which could make the features projected to higher dimensions. Then the regression function could be shown as $f(x) = \sum_{i=1}^{l}(\alpha_i - \alpha_i^*)k(x_i, z) + b$ where α_i and α_i^* are Lagrange multipliers and $k(x_i, x)$ is the kernel function. The most common kernel is the gaussian kernel function, $k(x_i, x_j) = exp(\frac{-||x_i - x_j||^2}{2\gamma^2})$, where γ is a parameter determining if the model is overfitting or underfitting.

3 Quadratic Programming Lagged Feature Selection

3.1 Overview

The whole process of quadratic programming lag selection (QPLS) is shown below in the flowchart: The input of the QPLS system is the original data set. First, the original features combine with all the possible lags to construct search space. All the following calculations will be based on the lagged feature search space. Second, the redundancy matrix which marks the similarity level within the features and the relevance vector which marks the relationship between each feature and the target will be calculated simultaneously. Third, the matrix and the vector will be preprocessed before putting into the optimization solver to make them fit the format that the solver requires. Fourth, the quadratic solver will generate the result of the optimization problem. The result also represents the feature importance of each feature. Last, the rank of the features is combined with the predefined significance threshold/ number of features to select the top n lagged features.

3.2 Search Space Construction

The original data set $Z = \{f_1, f_2, \cdots, f_n\}$ where $f_i, i \in [1, n)$ is the ith feature in the original data set and f_n is the prediction target of the regression model. Since the data set is multivariate time series, the shift operation needs to be conducted on the dataset Z to avoid data leakage. Then, the shifted data set Z' will be combined with the time lags from 0 to 29 to generate the lagged feature search space X. After that, the lagged features including shifted target y are generated from lag 1 to lag 30.

3.3 Redundancy Matrix Construction

In the proposed method, there are two ways of measuring the redundancy within the features in the subset selected. One is the correlation coefficient and the other is the mutual information. Naturally, we could use a matrix Q to express the level of redundancy according to the criterion we select.

$$Q = \begin{bmatrix} Q_{11} & Q_{12} & \cdots & Q_{1n} \\ Q_{21} & Q_{22} & \cdots & Q_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ Q_{n1} & Q_{n2} & \cdots & Q_{nn} \end{bmatrix} \tag{3.1}$$

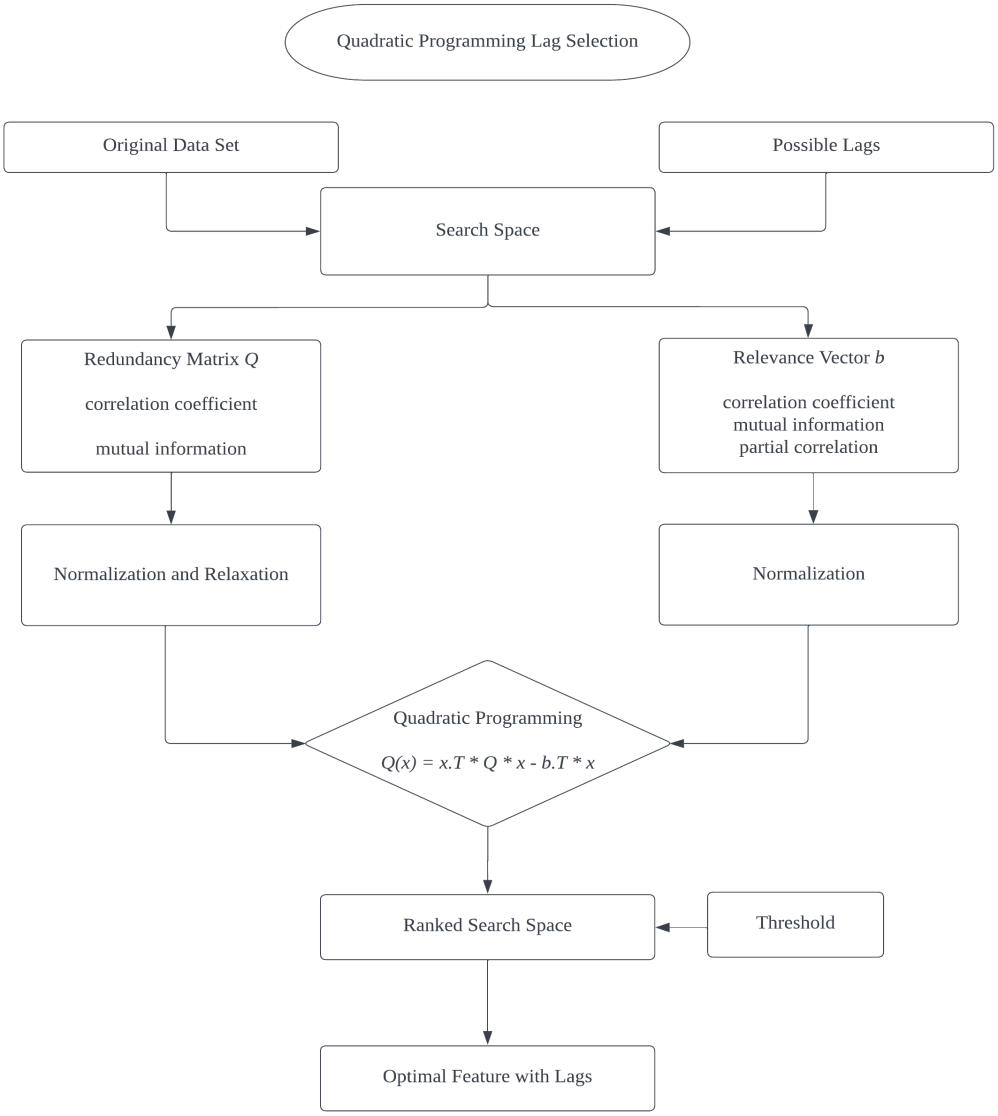


Figure 3: Overview of the QPLS Method

For the calculation of the redundancy score between lagged features from the same feature, there is a significant characteristic that the redundancy score of two lagged features is almost the same as that between another two lagged features generated from the same feature when the difference of two lags is the same. The reason is that the lagged feature is the original feature shifted with several time steps and the influence of these shifts on the redundancy score is small. For example, for one original feature in the data set, the correlation coefficient value between the feature lagged one and lagged three is almost the same as that between feature lagged three and lagged five which only depends on the difference of the lags. In this example, the difference is 3 - 1 = 5 - 3 = 2. Hence, this characteristic could be used to simplify the calculation cost in computing the redundancy matrix for the mass dataset.

For instance, for the feature $f1$, the original correlation coefficient matrix between its own lagged features Q_{11} could be expressed in the format below:

$$Q_{11} = \begin{bmatrix} X_{1,1} & X_{1,2} & X_{1,3} & X_{1,4} & \cdots & X_{1,30} \\ X_{2,1} & X_{2,2} & X_{2,3} & X_{2,4} & \cdots & X_{2,30} \\ X_{3,1} & X_{3,2} & X_{3,3} & X_{3,4} & \cdots & X_{3,30} \\ X_{4,1} & X_{4,2} & X_{4,3} & X_{4,4} & \cdots & X_{4,30} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ X_{30,1} & X_{30,2} & X_{30,3} & X_{30,4} & \cdots & X_{30,30} \end{bmatrix} \tag{3.2}$$

In the matrix above, $X_{i,j}$ represents the the correlation coefficient between feature $f1$ with lag i and that with lag j. The matrix is symmetric since the commutative property in both mutual information calculation and the correlation coefficient calculation holds. Hence, half of the matrix could be assigned as the value in its symmetric position. Besides, the values on the diagonal represent the relationship between the feature and itself is not what we are interested in, so to simplify the calculation, we could assign them as constant. Therefore, the matrix above could be expressed as

$$Q_{11} = \begin{bmatrix} c & X_{1,2} & X_{1,3} & X_{1,4} & \cdots & X_{1,30} \\ X_{1,2} & c & X_{1,2} & X_{1,3} & \cdots & X_{1,29} \\ X_{1,3} & X_{1,2} & c & X_{1,2} & \cdots & X_{1,28} \\ X_{1,4} & X_{1,3} & X_{1,2} & c & \cdots & X_{1,27} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ X_{1,30} & X_{1,29} & X_{1,28} & X_{1,27} & \cdots & c \end{bmatrix} \tag{3.3}$$

And the matrix Q could be expressed as

$$Q = \begin{bmatrix} Q_{11} & Q_{12} & \cdots & Q_{1n} \\ Q_{12} & Q_{22} & \cdots & Q_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ Q_{1n} & Q_{2n} & \cdots & Q_{nn} \end{bmatrix} \tag{3.4}$$

where the diagonal matrix $Q_{11}, Q_{22}, \cdots, Q_{nn}$ are calculated by using the method demonstrated earlier.

3.4 Relevance Vector Construction

In the proposed method, there are three ways of measuring the relationship between the subset selected and the target vector. The methods are correlation coefficient, mutual information and partial correlation. For each criterion, the relevance between a specific feature and target could be calculated and expressed as a real number. Therefore, the relevance vector b could be denoted as

$$b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_l \end{bmatrix} \tag{3.5}$$

where b_i is the criterion value between the ith candidate feature in lagged feature search

space and the target y, l is the number of the features in the search space.

3.5 Quadratic Programming Format Conversion

As the aim of the selection process is to find the subset with the minimum redundancy as well as the maximum relation, the objective function could be shown as:

$$\min_x\{Red(x) - Rel(x)\}$$

where $Red(\dot{)}, Rel(\dot{)}$ are redundancy and relevance part in the objective function. For the situation applying mutual information criteria for both redundancy and relevance, the function is similar to the mRMR method of feature selection. The difference is that mRMR is a greedy selection method and treats the redundancy and relevance parts separately, while quadratic programming considers both factors simultaneously and provides a global optimized result.
According to the previous research related to quadratic programming feature selection [34], the objective function of the original quadratic programming optimization algorithm could be denoted as

$$\min_x \quad f(x) = \frac{1}{2}x^T Gx - a^T x \tag{3.6}$$

$$\text{s.t.} \quad C^T x \geq b \tag{3.7}$$

For the lag selection problem, in order to minimize the redundancy and maximize the relevance of the selected subset at the same time, the quadratic program could be expressed as

$$\begin{aligned} \min_x \quad & Q(x) = x^T Qx - b^T x \\ \text{s.t.} \quad & x \in \{0,1\}^l \end{aligned} \tag{3.8}$$

where x is a l-dimensional vector, $Q \in R^{l\times l}$ is a symmetric matrix, b is a vector $\in R^l$.
After considering the weights of the quadratic part and the linear part, the objective function could be denoted as

$$\min_x\{(1-\alpha)x^T Qx - \alpha b^T x\} \tag{3.9}$$

where the x, Q, b is the same as the one defined above and $\alpha \in [0,1]$ is the weight of the linear term in the objective function. If $\alpha = 0$, there will only be a quadratic part

in the function which means that the objective is to minimize the redundancy of the selected subset. The selected features are the ones that are the most independent from one another. On the contrary, if $\alpha = 1$, only the linear part is preserved in the objective function which means that the objective is to maximize the relevance of the selected subset. The selected features are the ones most relevant to the target. The best α is different for different datasets and regression problems. In this study, we want to keep a balance between two factors. Hence, we set the α as $\frac{1}{2}$.

3.6 Normalization and Relaxation

In the objective function, what we are concerned about is the degree of the relationship between one feature and another feature or one feature and the target. Both positive and negative values of the criterion are meaningful for us. A positive value means they have a positive correlation which means there is more likely they will change in the same direction. The situation is similar for the negative value which means the negative correlation or moving in the opposite direction. Therefore, we take the absolute value of matrix Q and vector b.

Criterion such as correlation coefficient, mutual information and partial correlation could give us the calculated relevance or redundancy value but these methods do not show us the degree of the relevance directly. Especially for the situation where we use the combination of different criteria. Therefore, we need to apply the normalization to make the matrix Q and vector b comparable. For each value in each column, we calculate the proportion of the value in the column and use it to represent the normalized feature significance. The value in the normalized redundancy matrix and relevance vector could be denoted as:

$$\hat{X_{ij}} = \frac{b_i}{\sum_{i=1}^{l} X_{ij}} \tag{3.10}$$

$$\hat{b_i} = \frac{b_i}{\sum_{i=1}^{l} b_i} \tag{3.11}$$

where $X_{ij}\ i \in R^{l \times l}$ is the value of ith row and jth column in the matrx Q, $b_i\ i \in R^l$ is the ith value in the vector b, $\hat{X_{ij}}$ and $\hat{b_i}$ are the normalized value.

In the process of solving quadratic programming, there is a requirement that the hessian matrix Q is a positive definite matrix. If the normalized matrix Q is not a positive semidefinite matrix, the optimization problem is not convex. Therefore, we need to apply convex relaxation on the matrix Q. According to the method proposed by Naghibi et al [35], semi-definite programming relaxation (SDP) is a better method to solve the

problem compared with the linear programming relaxation method. Here, we use a simple example to demonstrate the subset selection process, the initial optimization problem could be denoted as:

$$\max_{x} \; x^T Q x \quad \sum_{i=1}^{N} x_i = P \quad i \in \{0, \cdots, N\} \tag{3.12}$$

In this example, N is the number of the features and $x_i \in \{\ 0, 1\}$ is the presence or absence of the ith feature in the feature set. Therefore, the size of the search space is 2^N. P is the number of features in the subset which is assumed as given. In the proposed method, optimal P could be found by evaluating the performance of the subsequent prediction models. Q is a symmetric redundancy matrix. Then we transfer the (0,1) - quadratic programming problem to the (-1,1) - quadratic programming problem by using $y = 2x - e$ transformation.

$$\begin{aligned} &\max_{y} \quad \frac{1}{4} y^T Q y + \frac{1}{2} y^T Q e + \frac{1}{4} e^T Q e \\ &\sum_{i=1}^{N} y_i = 2P - N \\ &y_i \in \{-1, 1\} \quad \text{for} \quad i = 1, \cdots, N \end{aligned} \tag{3.13}$$

where e is a matrix whose all elements are ones, $\frac{1}{4} e^T Q e$ is a constant which is not influenced by y. And the constant $\frac{1}{4}$ could also be ignored. By introducing the matrix Q':

$$Q' = \begin{pmatrix} 0 & e^T Q \\ Q^T e & Q \end{pmatrix} \tag{3.14}$$

Then the objective function could be denoted as the following homogeneous form:

$$\begin{aligned} &\max_{Y} \quad y^T Q' Y \\ &\sum_{i=1}^{N} y_i y_0 = 2P - N \\ &y_i \in \{-1, 1\} \quad \text{for} \quad i = 1, \cdots, N \end{aligned} \tag{3.15}$$

where $y_0 = \pm 1$ Furthermore, we apply the semidefinite programming relaxation method [36] on the problem above. The SDP format of the problem could be shown as:

$$\begin{aligned} & S_{SDP} = \max_{Y} \ tr\{Q'Y\} \\ & \sum_{i,j=1}^{N} Y_{ij} = (2P - N)^2 \\ & \sum_{i=1}^{N} Y_{i0} = (2P - N) \\ & diag(Y) = e \\ & Y \geq 0, \end{aligned} \tag{3.16}$$

where Y is a positive semi-definite matrix, $tr\{Y\}$ means the trace of matrix Y. Besides, the relationship between Y and y is that $Y = yy^T$.
After the previous steps, we get the homogeneous SDP format of the original optimization problem. The SDP format problem could be solved by using convex based relaxation approximation (COBRA) algorithm. Finally, the vector calculated via this method means the feature importance level. According to the vector, the top k largest value of Y will be assigned as ones. Other values of Y will be assigned as zeros. The ones and zeros correspond to the presence and absence of the features in the subset.
In the proposed method, there are small differences between the example above. For example, the problem we want to solve is a minimization problem, so we use the numerically stable dual method from the research of Goldfarb and Idnani to solve the SDP format problem instead of the method from their essay. The majority of the SDP part is similar, through the whole process, we transfer the problem whose solution is in the binary domain to a problem whose solution is in the continuous domain. As a result, we get feature scores as an indication of the importance of all features. By using the importance value, we could easily rank the candidate features. The top k features in the rank will be selected. To achieve this goal, the rank is combined with the predefined significance threshold (number of features in the selected subset) to generate the input subset for the prediction model.

3.7 Hyperparameter Tuning

For the method of solving the lagged feature selection problem proposed by us, tuning the hyperparameter k is performed to find the optimal number of features in the selected subset. According to the research of Vinh, Chan, Romano and Bailey [37], the number

could be selected in the range of 1 to the minimum of the total number of lagged features l and 100. If the number of candidate features is less than 100, the end of the range will be set as l. If the number of candidate features is larger than 100, the end of the range will be set as 100. Furthermore, the scoring result will be combined with the range of k to generate 100 subsets if the end of the range is 100. Besides, 100 subsets will be put into the prediction model training process to construct 100 prediction models. Then, we apply the test set whose lag is the same as the training set on the prediction model to get the curve of model performance as a function of the number of features in the subset k. In this study, we use the Root Mean Square Error (RMSE) as the performance measurement of the prediction model. By observing the curve, the optimal hyperparameter for a specific data set on a specific kind of prediction model based on a specific combination of the lag selection criterion could be found.

4 Experiment and Result

The chapter includes the design and implementation of the experiment of quadratic programming lag selection and the regression model based on the selected lags. The chapter contains the choice of important factors in the design phase according to the background and theories chapter. What is more, the chapter introduces the structure of the project and some details of implementation.

4.1 Experiment Design

4.1.1 Data

To test the proposed optimal lag selection method based on the different combinations of criteria, we use five test data sets from yahoo finance A.3 and UCI machine learning repository A.3. The data sets are from different industries. The summary of the data sets is shown below:
The first data set is the SPDR S&P 500 ETF dataset from yahoo finance. The data set has 2558 records and 6 features including "open", "High", "Low", "Close", "Adj Close" and "Volume". The target of the regression model is "Close" which means the daily close price of the S&P 500 ETF.
The second data set is the individual household electric power consumption dataset [3] from UCI machine learning repository. The data set has 2075259 records and 9 features including "date", "time", "global_active_power", "global_reactive_power", "voltage", "global_intensity", "sub_metering_1", "sub_metering_2" and "sub_metering_3". The target of the regression model is "global_active_power" which means the household global minute-averaged active power (in kilowatt).
The third data set is the air quality dataset [4] from UCI machine learning repository. The data set has 9358 records and 14 features including "Date", "Time", "CO(GT)", "PT08.S1(CO)", "NMHC(GT)", "C6H6(GT)", "PT08.S2(NMHC)", "NOx(GT)", "PT08.S3(NOx)", "NO2(GT)", "PT08.S4(NO2)", "PT08.S5(O3)", "T", "RH" and "AH". The target of the regression is "C6H6(GT)" which means the true hourly averaged Benzene concentration in microg/m^3 (reference analyzer).

[3] This dataset is made available under the "Creative Commons Attribution 4.0 International (CC BY 4.0)" license

[4] S. De Vito, E. Massera, M. Piga, L. Martinotto, G. Di Francia, On field calibration of an electronic nose for benzene estimation in an urban pollution monitoring scenario, Sensors and Actuators B: Chemical, Volume 129, Issue 2, 22 February 2008, Pages 750-757, ISSN 0925-4005

The fourth data set is the concrete compressive strength dataset [5] from UCI machine learning repository. The data set has 1030 records and 9 features including "cement", "blast_furnace_slag", "fly_ash", "water", "superplasticizer", "coarse_aggregate", "fine_aggregate", "age" and "concrete_compressive_strength". The target of the regression is "concrete_compressive_strength" which means the daily close price of the ETF. The last data set is the Productivity Prediction of Garment Employees dataset [6] from UCI machine learning repository. The data set has 1197 records and 15 features including "date", "day", "quarter", "department", "team_no", "no_of_workers", " no_of_style_change ", "targeted_productivity", "smv", "wip", "over_time", " incentive", "idle_time", "idle_men" and " actual_productivity ". The target of the regression is " actual_productivity " which means the actual percentage of productivity that was delivered by the workers. It ranges from 0 to 1.

4.1.2 Technology Stack

The related code is conducted and executed on the Windows system. The system environment is Microsoft Windows [Version 10.0.22000.795]. And the datasets are stored in the local place of the personal computer. In this study, we choose Python as the programming language. Python is an advanced programming language which is widely used in the data science field. It has advanced data structure and the features of object-oriented programming. Compared with traditional programming languages, is more readable and interpreted. To acquire the visualization result directly, we also use the Jupyter Notebook. Jupyter Notebook is an open-source web application maintained by the members of project Jupyter. In this project, the version of Python is 3.10.4 (tags/v3.10.4:9d38120, Mar 23 2022, 23:13:41) [MSC v.1929 64 bit (AMD64)] on win32. Besides, the Python libraries designed for specific machine learning or optimization task are used in this project. For instance, pandas, NumPy, matplotlib, quadprog, sklearn, pingouin and warnings.

4.1.3 Code Structure

The Jupyter Notebooks are designed for five datasets and three methods (QPLS, GA, PACF) separately. The code is composed of three parts: data prepossessing part,

[5] I-Cheng Yeh, "Modeling of strength of high performance concrete using artificial neural networks," Cement and Concrete Research, Vol. 28, No. 12, pp. 1797-1808 (1998)

[6] A. A. Imran, M. N. Amin, M. R. Islam Rifat and S. Mehreen, "Deep Neural Network Approach for Predicting the Productivity of Garment Employees," 2019 6th International Conference on Control, Decision and Information Technologies (CoDIT), 2019, pp. 1402-1407, doi: 10.1109/CoDIT.2019.8820486.

quadratic programming part and experiment part. The first is the data prepossessing part, the input of the system is the original data set. Then the data set will be checked if there are missing values. If there are missing values, those values will be solved with test statistic by using the function $fill\_by\_mean$. Then, the data set will be sampled at the day level to make the training data at thousands level which is to speed up the following model training process. After that, the data set will be separated into training and testing sets by using the function $train\_test\_split$. The raw data is shifted by a time step to avoid data leakage in the function $data\_shift$. The raw data set as the benchmark for comparing could be generated in function $benchmark\_raw\_generator$.
The second part is the quadratic programming part, the input of this part is the preprocessed data generated from the previous part. The data will be combined with the possible lags to construct the lagged features search space by using the function $feature\_space\_generator$. The next step is to construct the objection function based on the search space according to the different combinations of the criterion. The core factor of the objective function construction is the calculation of redundancy matrix Q and relevance vector b. These main components of the objective function could be generated by using the function $opt\_problem\_generator$. The normalization operation and definite positive relaxation operation of the matrix Q and vector b could be achieved by using the function $normalization$ and $definite\_positive\_relaxation$. Furthermore, the prepared quadratic programming optimization problem will be solved by the function $quadprog\_solver$. To apply different criteria to the same dataset, we design the function $multi\_lag\_selection$. And the output of the function is the feature importance score lists. Next, the feature importance score list could be transferred into a selected subset by using the function "score_ 2_ set". After this step, the selected subsets for training and testing of the prediction model are generated.
The third part is the experiment part which consists of prediction model training and hyperparameter tuning. In this part, four regression models including linear regression, random forest regression, elastic net regression and support vector regression are applied to predict the target. Therefore, the function $linear\_regression\_result$, $random\_forest\_result$, $elastic\_net\_result$ and svr_result are designed to calculate the minimum RMSE for the different subsets as well as plot the relationship between the RMSE and number of the features in the subset. The plot could be used to determine the optimal number of features k. Finally, the function $QPLS$ are used to integrate all the functions above. Hence, the system could be regarded as a black box whose input is the data set and the target and the output is the performance of different regressions models based on optimal lagged feature selection according to different criteria.

4.1.4 Implementation

As the structure mentioned above, the implementation of the project is composed of three parts. In the first part, the local data sets are imported into the system by using *pandas.read_csv* function. For the missing values processing, in the $fill\_by\_mean$ function, missing values check is conducted by using *notnull* by columns. If there is NA in the column, it will be filled by using *mean* function. For the large data set we want to predict at the day level, the data sampling is conducted by using the function *resample*. Then the data set is split by the index of rows in function *train_test_split*, and the train and test ratio in this study is set as 0.7. The last step in prepossession is the $data\_shift$ function where function $shift$ is applied to achieve the goal.
In the second part, to generate the search space, the processed data set is bound with all the possible lags of which the maximum possible value is set as 30 according to the previous sections. In the $feature\_space\_generator$ function, the columns of the data are shifted according to the lags by using the $shift$ function and concatenated by using *concatenate*. Besides, the *dropna* function is used to trim the data set with NA values after being shifted. To calculate the objective function of the selected subset based on various criteria. The function "opt_ problem_ generator" includes parameters *sim* and *rel* related to redundancy and relevance criteria. The "correl", "mi" are for redundancy matrix and "correl", "mi", "pcor" are for relevance vector. The string input of the function "correl", "mi", "pcor" are short for correlation coefficients, mutual information and partial correlation. These criterion values are calculated by using the function $corrcoef, mutual\_info\_score$, and *pcorr*. The relaxation and normalization are also included in the function "opt_ problem_ generator", these functions are designed according to the theories in section three. What is more, the matrix Q and vector b could be transferred into the format that matches the quadratic programming solver. Then the function *solve_qp* from Python package *quadprog* could be used to solve the problem accurately and efficiently. After the calculation, the feature importance score list are generated in the sequence of "correl-correl","correl-mi","correl-pcor","mi-correl","mi-mi" and "mi-pcor". In the combination "a-b", the first criterion is redundancy measurement and the second is relevance measurement. Furthermore, the score lists could be used for features sorting by the function *sort_values*. In *score_2_set* function, each sorted feature set will be used to generate the subset of which the number is the minimum of 100 and the number of lagged features.
In the third part, there are up to six hundred subsets for all the criterion combinations. The prediction models are built on these data sets. In this project, the

LinearRegression, *RandomForestRegressor*, *ElasticNet* and *SVR* are all from *sklearn* package. Based on the score lists from the second part, we not only rearrange and generate the training set but also the test set which means the training set and the related testing set have the same selected feature and different instances. The training data set and testing data set are split into data set $train\_x, train\_y, test\_x$ and $test\_y$ according to the column. And the raw data shifted with a time step will be split into $train\_raw\_x$, $train\_raw\_x$, $test\_raw\_x$ and $test\_raw\_y$. For each model, the model is trained based on $train\_x$ and $train\_y$. Then, the trained model will be used to predict $pred\_y$ on the $test\_x$. Furthermore, the root mean square error will be calculated by using the function *math.sqrt* and *mean_squared_error* on $pred\_y$ and $test\_y$. The min-max scaling and back transformation are conducted by using $TransformedTargetRegressor$ and $MinMaxScaler$. For the linear regression model, random forest model and support vector regression model, we use the default parameter. For the elastic net model, we use the function $GridSearchCV$ to conduct the grid search parameter tuning. The range of parameter *alpha* is [0.0, 1.0, 10.0, 100.0]. And the range of $l1\_ratio$ is [0.1, 0.5, 0.7, 0.9, 0.95, 0.99, 1]. By using this way, the optimal parameter for the elastic net model could be found. After all the experiments, the curve of RMSE as a function of the number of features in the subset could be plotted by using the function *plot*. And the optimal number of selected lagged features could be found.

4.2 Experiment Result

In this subsection, the performances of the prediction models applying different lag selection methods based on various criteria will be plotted. Besides, the corresponding methods will be evaluated and compared with the benchmark. The subsection will cover the following materials:

1. Performance measurement
2. Results of QPLS method, GALS method, PACF method and no selection method
3. The analysis of the experiment result.

In this study, the root square mean error is used as the measurement of the prediction model [**?**]. The effect of each error on the RMSE is proportional to the magnitude of the squared error. Therefore, the larger errors have a very large effect on the RMSE [**?**]. Hence, the selected criterion according to RMSE could be more robust and not easily affected by the extreme errors generated from the outliers of the data set. The formula

Table 1: The Result of Linear Regression Model

Index of Data Set	1	2	3	4	5
Model	LR	LR	LR	LR	LR
Redundancy Criterion	cor	cor	cor	cor	cor
Relevance Criterion	pcor	cor	cor	cor	cor
QPLS	4.81241	**356.6821**	**14.38637**	**13.15699**	**0.118661**
# of features (QPLS)	8	17	11	8	2
Raw Data	5.056045512	381.7826451	14.95143996	18.74812074	0.430906607
GALS	4.811976164	366.5183273	14.83010418	16.76568298	0.283426869
# of features (GA)	31	46	114	72	110
PACF	**4.802422**	372.5474	14.68717	17.04485	0.383681
# of features (PACF)	27	87	134	136	178

of the measurement can be as follows:

$$RMSE = [\frac{1}{n}\sum_{i=1}^{n}(\hat{y_i} - y_i)^2] \tag{4.1}$$

where n is the number of observations, y is the actual values and $\hat{y_i}$ is the prediction values.

The result of the experiments are shown in table 1, table 2, table 3 and table 4. In the tables, the index of datasets is the same as that in the previous dataset section. "LR", "RF", "EN" and "SVR" are short for the prediction models introduced above. The redundancy and relevance criteria are the best criterion of the QPLS method. The raw data here is the raw data sampled with the same level as other methods and then the data is shifted with a time step. The result of the GALS is the best result generated from six combinations of the criterion.

First, the result tables show the RMSE of different models based on different methods. For a specific model, the tables show that QPLS method is always the best compared with the GALS method, PACF method and no selection benchmark on the data set from 2 to 5. For the S&P ETF data set, the PACF method could perform better than the proposed method on some of the models. This may be the result of the highly correlated relationship of the data set.

Second, the experiments are designed to select the feature subset from the lagged features search space. The number of lagged features selected by the QPLS is always less than other methods with better or similar performance to the prediction model. The method could reduce the complexity of the model without losing the prediction accuracy.

Third, the experiments are also designed to find the best criterion combination for

Table 2: The Result of Elastic Net Model

Index of Data Set	1	2	3	4	5
Model	EN	EN	EN	EN	EN
Redundancy Criterion	mi	cor	cor	cor	cor
Relevance Criterion	pcor	cor	cor	cor	cor
QPLS	**4.799017**	**353.9058**	**14.39562**	**13.12184**	**0.118661**
# of features (QPLS)	85	79	7	8	2
Raw Data	4.805495649	372.2380218	14.95292938	15.02669731	0.534463665
GALS	4.804665354	366.4977952	14.84882961	16.20489001	0.445201127
# of features (GA)	36	46	114	68	139
PACF	4.974613	365.9038	14.68604	14.20764	0.732166
# of features (PACF)	27	87	134	136	178

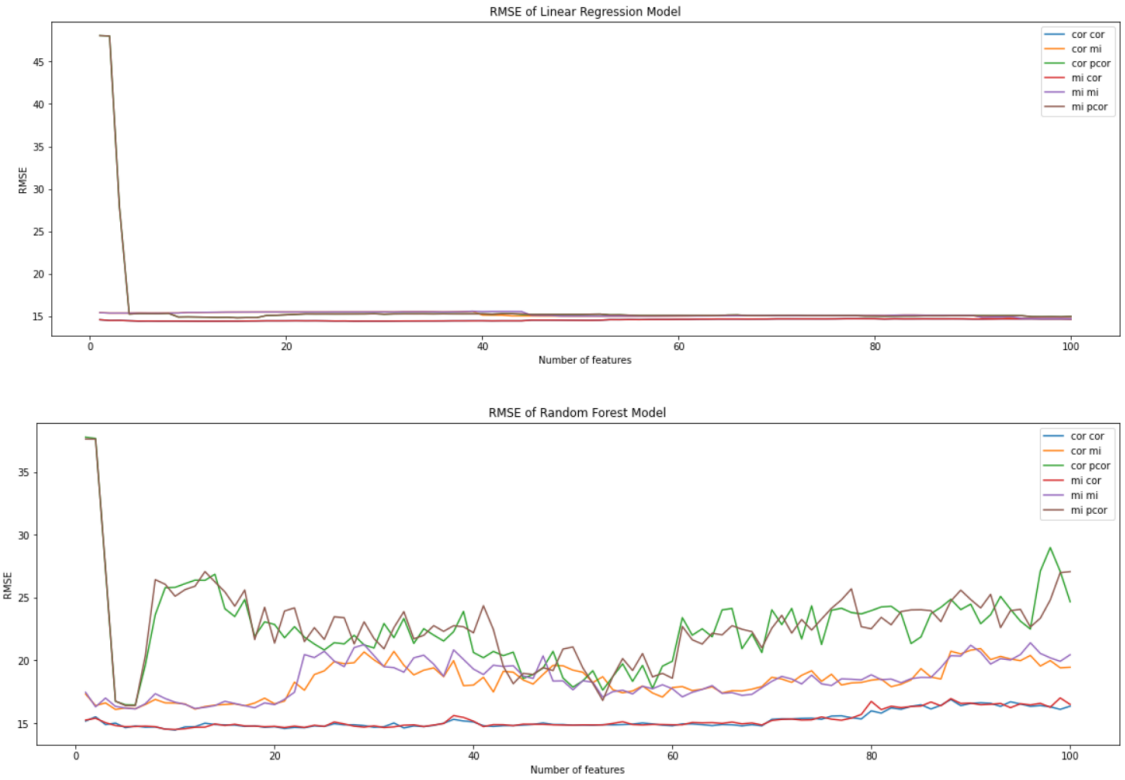
Table 3: The Result of Random Forest Regression Model

Index of Data Set	1	2	3	4	5
Model	RF	RF	RF	RF	RF
Redundancy Criterion	mi	cor	cor	mi	mi
Relevance Criterion	pcor	cor	cor	mi	pcor
QPLS	98.99794	**340.8156**	**14.46136**	**13.2065**	0.104295
# of features (QPLS)	2	17	10	83	48
Raw Data	100.3012543	352.5281158	22.97635737	13.58507532	0.097485423
GALS	100.36633	354.9942771	24.30973124	14.20644274	0.095480067
# of features (GA)	39	46	122	72	110
PACF	**98.58506**	342.3979	31.06855	13.23143	**0.094522**
# of features (PACF)	27	87	134	136	178

Table 4: The Result of Support Vector Regression Model

Index of Data Set	1	2	3	4	5
Model	SVR	SVR	SVR	SVR	SVR
Redundancy Criterion	cor	cor	cor	mi	mi
Relevance Criterion	mi	cor	cor	cor	cor
QPLS	161.6713	**352.6413**	**15.66956**	**13.23746**	**0.100163**
# of features (QPLS)	36	16	6	68	2
Raw Data	157.3879508	410.5072509	19.52533875	14.57933756	0.200674884
GALS	146.4223086	412.8632948	19.50327573	14.2831009	0.180563283
# of features (GA)	39	46	114	72	147
PACF	**151.611**	382.3686	21.79395	13.81251	0.191864
# of features (PACF)	27	87	134	136	178

Dataset: Air Quality

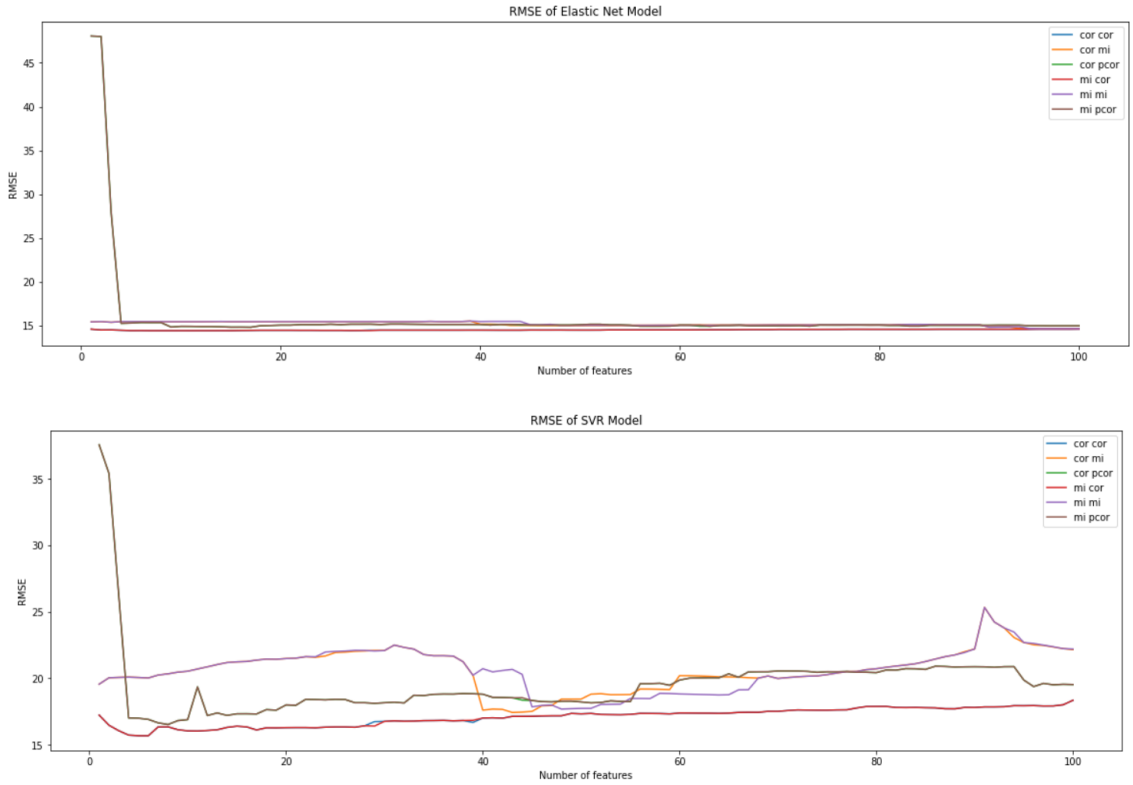


the lag selection algorithm. According to the results of all the data sets, the top two criterion combinations are correlation redundancy with correlation relevance and mutual information redundancy with correlation relevance. In different models over different data sets, these two combinations always lead to a good performance of the prediction model.

Fourth, the proposed method is much fast and more robust compared with the GALS. Based on the same objective function, the lag selection process is faster by using the proposed method.

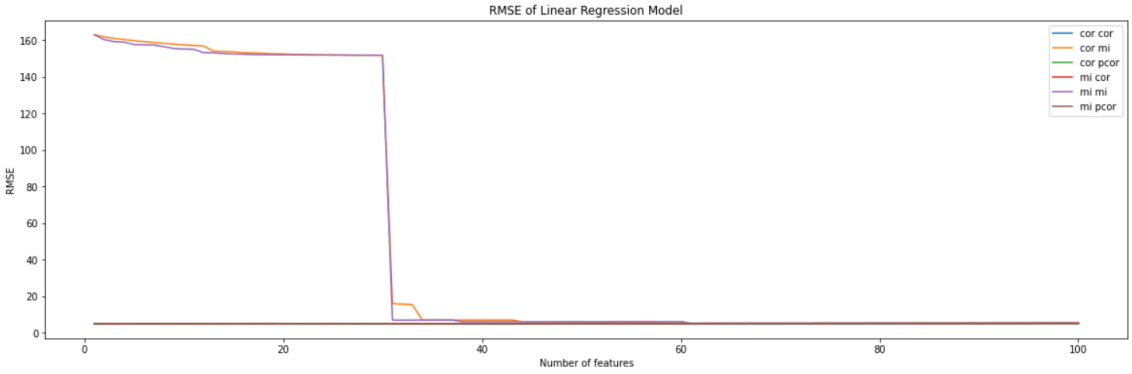
The figures of the QPLS hyperparameter tuning for air quality data set are shown below:

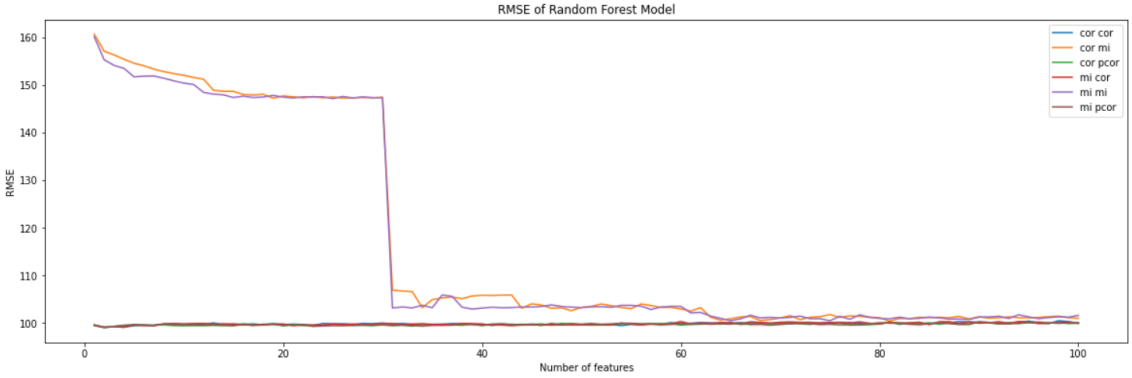
For the RMSE curve generated from the QPLS method, there is one specific original data set, one specific model and six criterion combinations in one figure. The picture shows the RMSE curve as the function of the number of features in the feature subset. In most cases, the RMSE curve first rises and then falls as the number of features increases. This is because the variables added at the beginning increase the amount of useful information for training the model. When the variables continue to be added until a certain level, the added variables cannot provide more effective information, and the

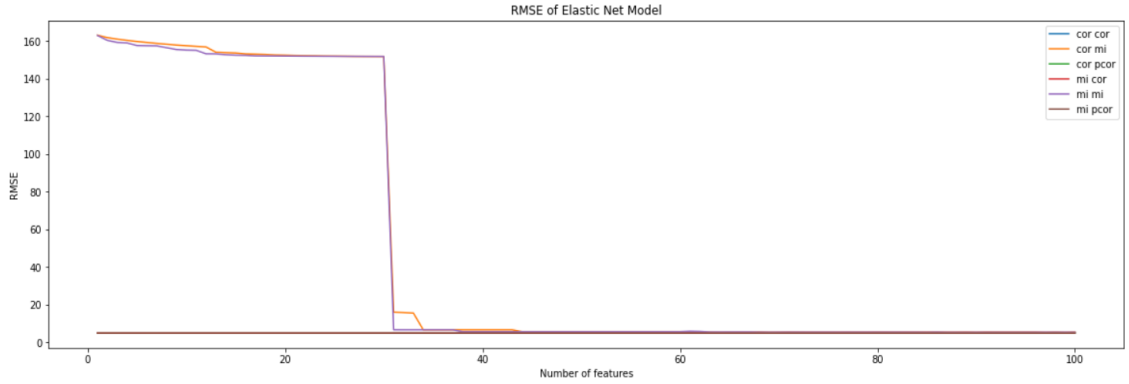


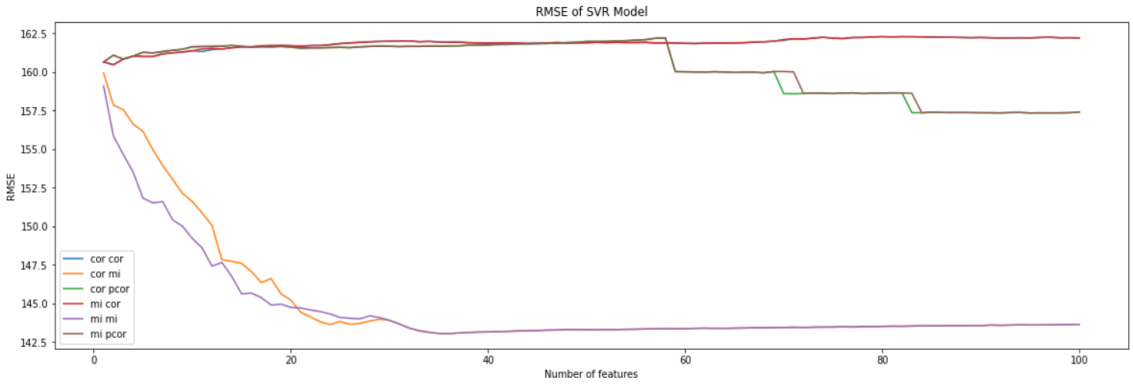
noise of the variables affects the performance of the model. Therefore, the curve goes down. Besides, for the independent variables with high redundancy, the curve may also show a downward trend as few variables could satisfy the needs of the model training. The results of other data sets are shown below:

Dataset: S&P ETF

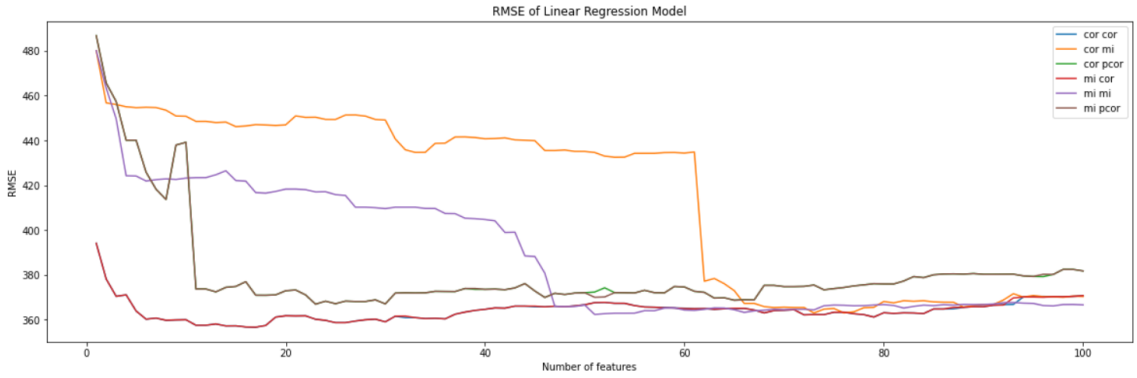


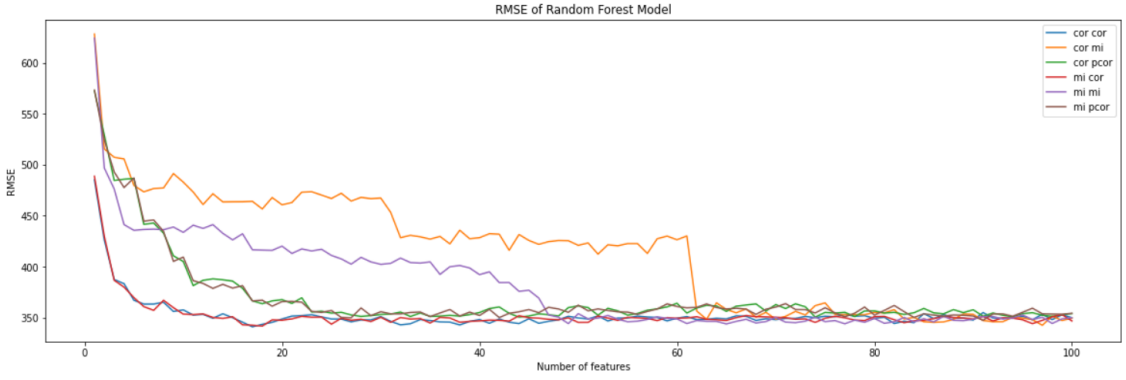


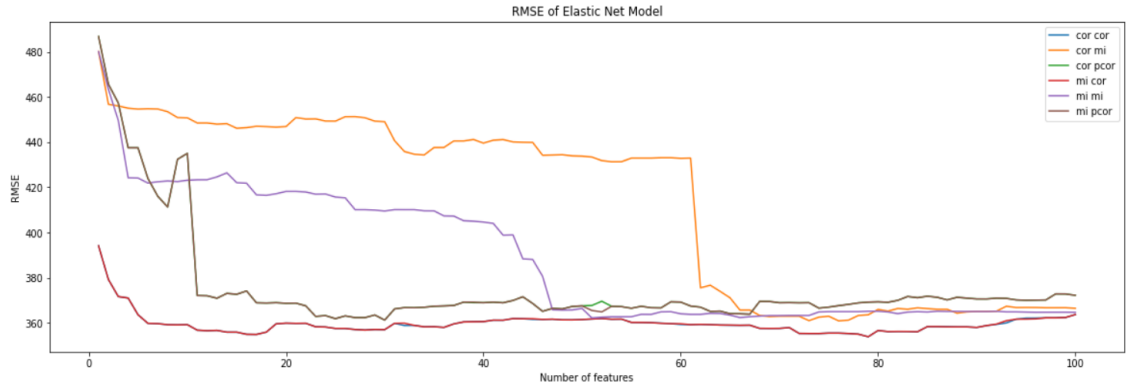


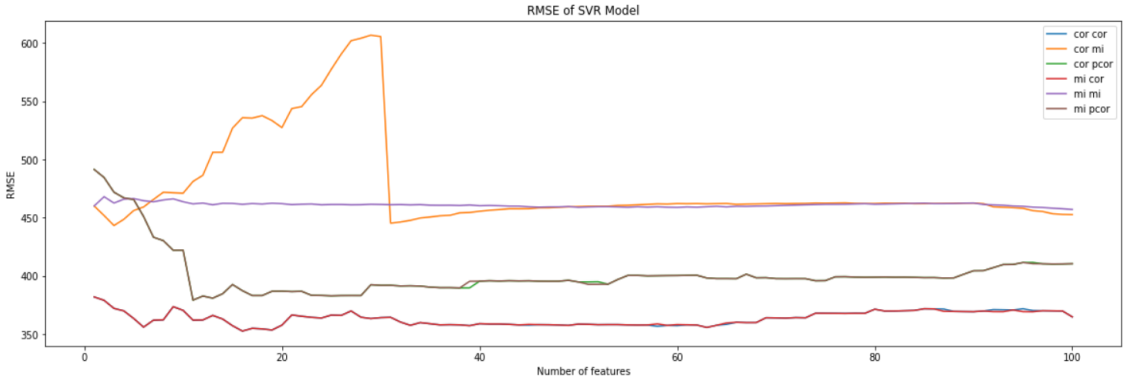


Dataset: Electricity

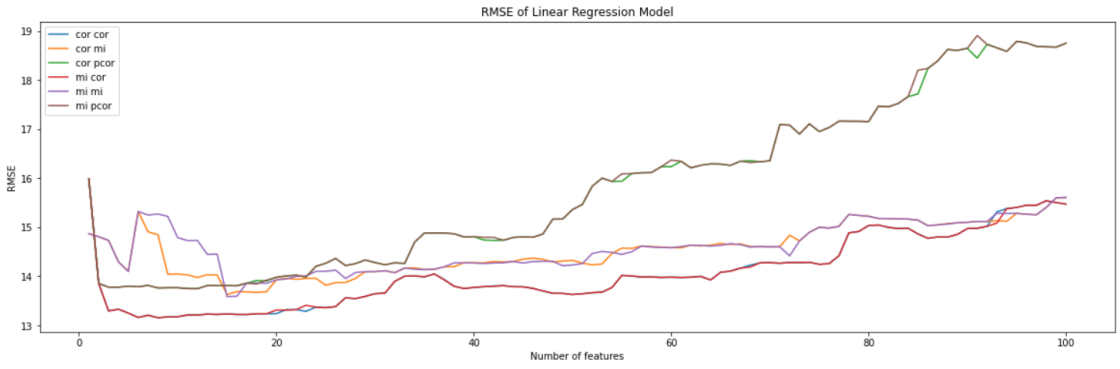


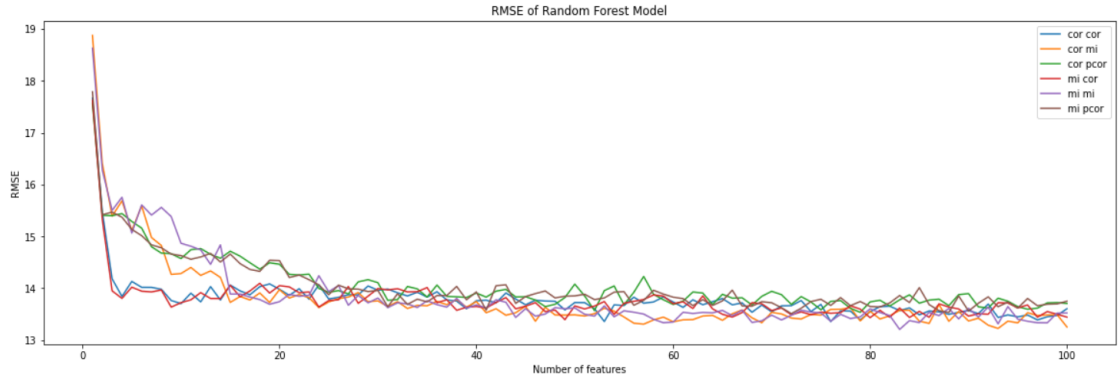


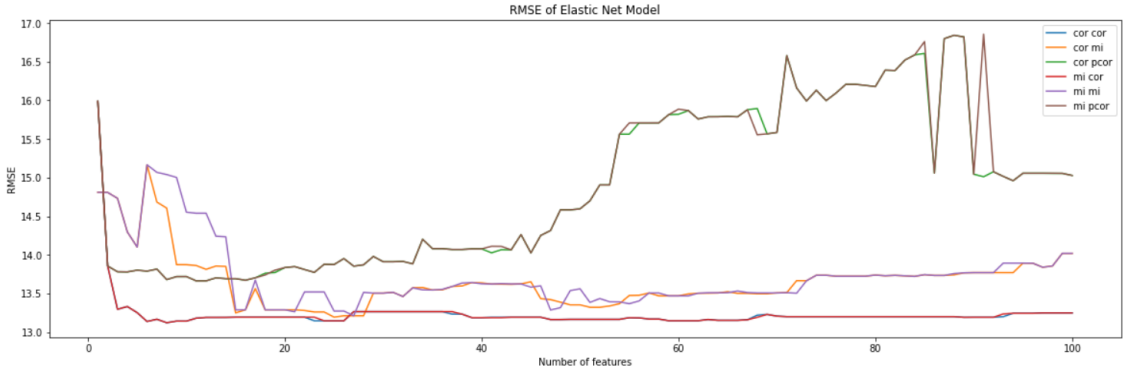


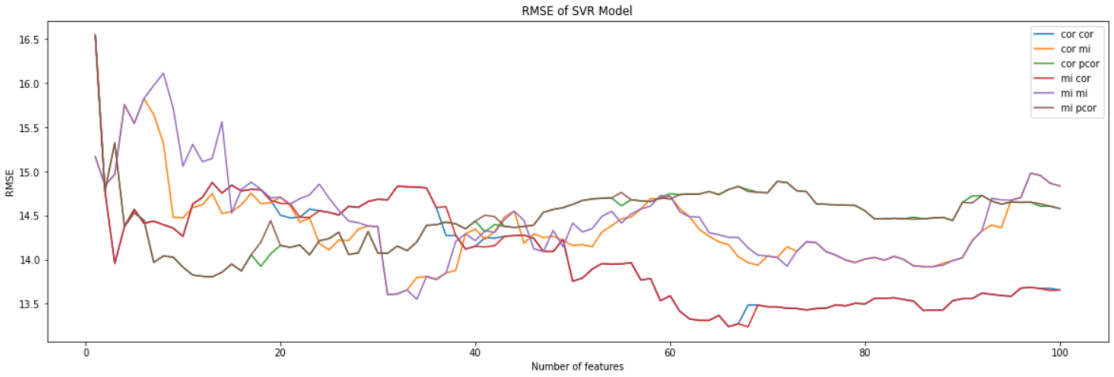


Dataset: Strength

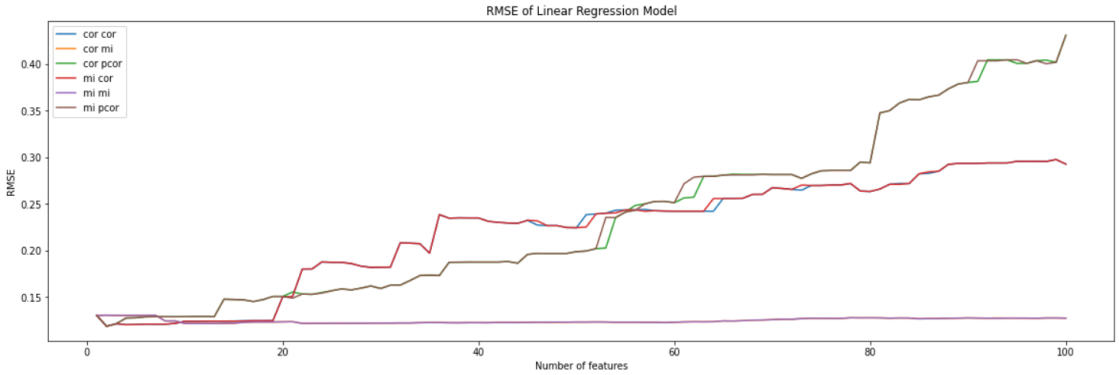


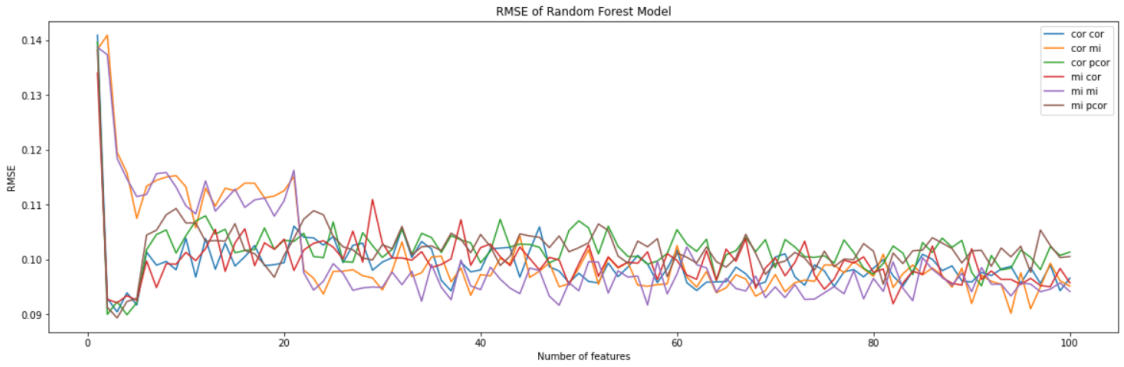

RMSE of Random Forest Model
RMSE
Number of features
cor cor
cor mi
cor pcor
mi cor
mi mi
mi pcor

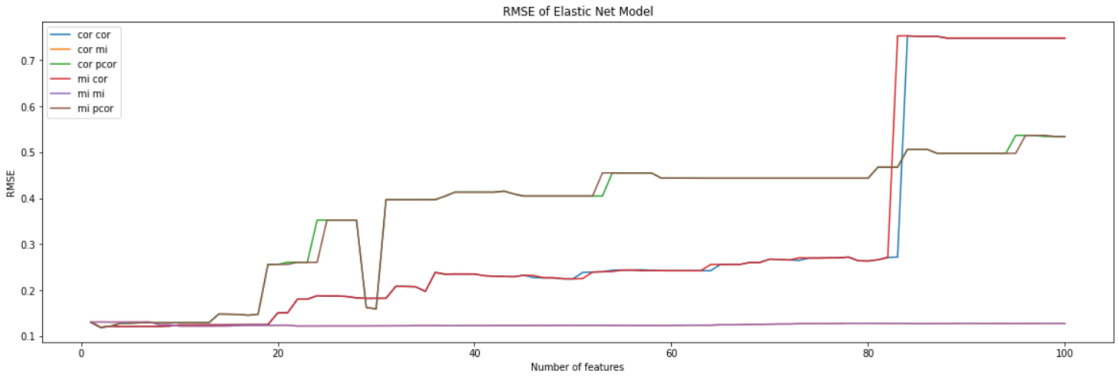

RMSE of Elastic Net Model
RMSE
Number of features
cor cor
cor mi
cor pcor
mi cor
mi mi
mi pcor

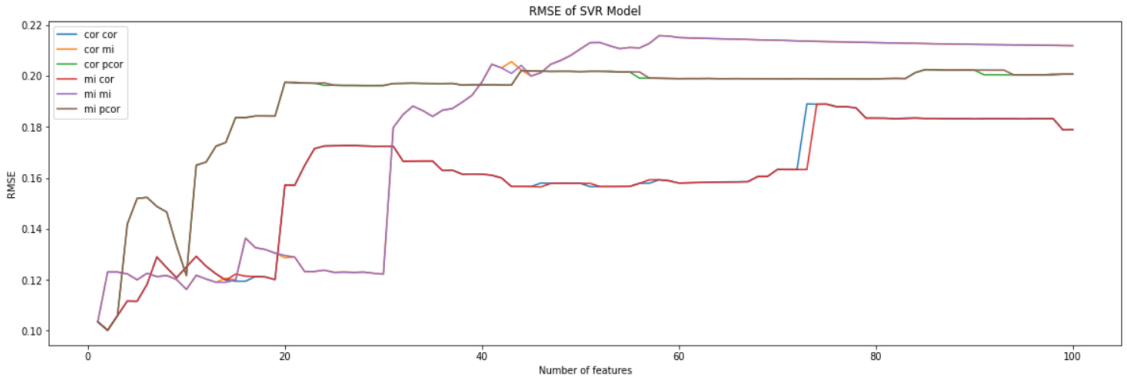

RMSE of SVR Model
RMSE
Number of features
cor cor
cor mi
cor pcor
mi cor
mi mi
mi pcor

Dataset: Productivity








RMSE of SVR Model
cor cor
cor mi
cor pcor
mi cor
mi mi
mi pcor
RMSE
0.22
0.20
0.18
0.16
0.14
0.12
0.10
0
20
40
60
80
100
Number of features

5 Conclusion and Future Work

5.1 Conclusion

In this project, a robust method for lagged feature selection problems on regression models has been proposed. To solve the problem and test the methods, the lag selection problem has been transferred into feature selection problems on time series data. The proposed method based on quadratic programming is compared with the genetic algorithm selection method, PACF method and no selection benchmark. First, the proposed method achieved the best results in 4 out of 5 data sets in regarding the RMSE performance measures. Second, in all models, the criteria composed of correlation and correlation and the criteria composed of mutual information and correlation could lead to good prediction performance. Finally, the method could help us reduce the complexity of the regression models by selecting fewer features than other methods without decreasing the predictive performance of the model.

5.2 Future Work

In this study, the objective function of quadratic programming optimization has the quadratic term and the linear term. To keep the balance between the redundancy factor and the relevance factor, the weight is set as 1:1. But in the practical lag selection process, there are different intrinsic information within the different data sets. Therefore, the next step of this study is to find a method to set the weight of the redundancy factor and the relevance factor dynamically based on the characteristic of different data set.
In this project, the hyperparameter k which represents the number of features in the subset is determined by observing the performance of the prediction model. If we could determine the hyperparameter before the model training phase, it could help us save a lot of time. To set the value of the parameter automatically, we could refer to the idea of the LASSO regression by adding the penalty. In the result of ranking, a specific number of the feature importance scores could be set as zero through the calculation. And the penalty factor λ is used to make a trade-off between the model prediction performance and the model complexity. Based on this idea, the next step is to find a method that can determine the number of selected features by adjusting the level of feature selection penalty λ.

References

[1] D. N. Gujarati, *Econometrics by example*, vol. 1. Palgrave Macmillan New York, 2011.

[2] R. F. Engle, "Wald, likelihood ratio, and lagrange multiplier tests in econometrics," *Handbook of econometrics*, vol. 2, pp. 775–826, 1984.

[3] P. Stoica and Y. Selen, "Model-order selection: a review of information criterion rules," *IEEE Signal Processing Magazine*, vol. 21, no. 4, pp. 36–47, 2004.

[4] E. Polyzos and C. Siriopoulos, "Autoregressive random forests: Machine learning and lag selection for financial research," *Available at SSRN 4118546*, 2022.

[5] R. Scott Hacker and A. Hatemi-J, "Optimal lag-length choice in stable and unstable var models under situations of homoscedasticity and arch," *Journal of Applied Statistics*, vol. 35, no. 6, pp. 601–615, 2008.

[6] A. Hatemi-J and R. S. Hacker, "Can the lr test be helpful in choosing the optimal lag order in the var model when information criteria suggest different lag orders?," *Applied Economics*, vol. 41, no. 9, pp. 1121–1125, 2009.

[7] G. Simon and M. Verleysen, "Lag selection for regression models using high-dimensional mutual information.," in *ESANN*, pp. 395–400, Citeseer, 2006.

[8] O. Ludwig Jr, U. Nunes, R. Araújo, L. Schnitman, and H. A. Lepikson, "Applications of information theory, genetic algorithms, and neural models to predict oil flow," *Communications in Nonlinear Science and Numerical Simulation*, vol. 14, no. 7, pp. 2870–2885, 2009.

[9] G. H. Ribeiro, P. S. d. M. Neto, G. D. Cavalcanti, and R. Tsang, "Lag selection for time series forecasting using particle swarm optimization," in *The 2011 International Joint Conference on Neural Networks*, pp. 2437–2444, IEEE, 2011.

[10] J. F. L. de Oliveira and T. B. Ludermir, "A hybrid evolutionary system for parameter optimization and lag selection in time series forecasting," in *2014 Brazilian Conference on Intelligent Systems*, pp. 73–78, IEEE, 2014.

[11] A. Widodo, I. Budi, and B. Widjaja, "Automatic lag selection in time series forecasting using multiple kernel learning," *International Journal of Machine Learning and Cybernetics*, vol. 7, no. 1, pp. 95–110, 2016.

[12] H. Hassani and M. R. Yeganegi, "Selecting optimal lag order in ljung–box test," *Physica A: Statistical Mechanics and its Applications*, vol. 541, p. 123700, 2020.

[13] N. Zhong, J. Dong, and S. Ohsuga, "Using rough sets with heuristics for feature selection," *Journal of intelligent information systems*, vol. 16, no. 3, pp. 199–214, 2001.

[14] M. Sharma and P. Kaur, "A comprehensive analysis of nature-inspired meta-heuristic techniques for feature selection problem," *Archives of Computational Methods in Engineering*, vol. 28, no. 3, pp. 1103–1127, 2021.

[15] Z. Cai and W. Zhu, "Feature selection for multi-label classification using neighborhood preservation," *IEEE/CAA Journal of Automatica Sinica*, vol. 5, no. 1, pp. 320–330, 2017.

[16] A. Y. Ng, "Feature selection, l 1 vs. l 2 regularization, and rotational invariance," in *Proceedings of the twenty-first international conference on Machine learning*, p. 78, 2004.

[17] G. Brown, A. Pocock, M.-J. Zhao, and M. Luján, "Conditional likelihood maximisation: a unifying framework for information theoretic feature selection," *The journal of machine learning research*, vol. 13, pp. 27–66, 2012.

[18] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani, "Least angle regression," *The Annals of statistics*, vol. 32, no. 2, pp. 407–499, 2004.

[19] M. El-Dereny and N. Rashwan, "Solving multicollinearity problem using ridge regression models," *International Journal of Contemporary Mathematical Sciences*, vol. 6, no. 12, pp. 585–600, 2011.

[20] H. Zou and T. Hastie, "Regularization and variable selection via the elastic net," *Journal of the royal statistical society: series B (statistical methodology)*, vol. 67, no. 2, pp. 301–320, 2005.

[21] R. Tibshirani and L. Wasserman, "Some aspects of the reparametrization of statistical models," *Canadian Journal of Statistics*, vol. 22, no. 1, pp. 163–173, 1994.

[22] E. W. Steyerberg, M. J. Eijkemans, F. E. Harrell Jr, and J. D. F. Habbema, "Prognostic modeling with logistic regression analysis: in search of a sensible strategy in small data sets," *Medical Decision Making*, vol. 21, no. 1, pp. 45–56, 2001.

[23] P. Ghamisi and J. A. Benediktsson, "Feature selection based on hybridization of genetic algorithm and particle swarm optimization," *IEEE Geoscience and remote sensing letters*, vol. 12, no. 2, pp. 309–313, 2014.

[24] R. L. Bankert and D. W. Aha, "Improvement to a neural network cloud classifier," *Journal of Applied Meteorology (1988-2005)*, pp. 2036–2039, 1996.

[25] V. F. Rodriguez-Galiano, J. A. Luque-Espinar, M. Chica-Olmo, and M. P. Mendes, "Feature selection approaches for predictive modelling of groundwater nitrate pollution: An evaluation of filters, embedded and wrapper methods," *Science of the total environment*, vol. 624, pp. 661–672, 2018.

[26] Y. Yang and J. O. Pedersen, "A comparative study on feature selection in text categorization," in *Icml*, vol. 97, p. 35, Citeseer, 1997.

[27] Y. Zhang, S. Li, T. Wang, and Z. Zhang, "Divergence-based feature selection for separate classes," *Neurocomputing*, vol. 101, pp. 32–42, 2013.

[28] H. Zare, G. Haffari, A. Gupta, and R. R. Brinkman, "Scoring relevancy of features based on combinatorial analysis of lasso with application to lymphoma diagnosis," in *BMC genomics*, vol. 14, pp. 1–9, Springer, 2013.

[29] K. Han, Y. Wang, C. Zhang, C. Li, and C. Xu, "Autoencoder inspired unsupervised feature selection," in *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pp. 2941–2945, IEEE, 2018.

[30] 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.

[31] D. Goldfarb and A. Idnani, "A numerically stable dual method for solving strictly convex quadratic programs," *Mathematical programming*, vol. 27, no. 1, pp. 1–33, 1983.

[32] D. C. Montgomery, C. L. Jennings, and M. Kulahci, *Introduction to time series analysis and forecasting*. John Wiley & Sons, 2015.

[33] V. Vapnik, *The nature of statistical learning theory*. Springer science & business media, 1999.

[34] I. Rodriguez-Lujan, C. Elkan, C. Santa Cruz, R. Huerta, *et al.*, "Quadratic programming feature selection," *Journal of Machine Learning Research*, 2010.

[35] T. Naghibi, S. Hoffmann, and B. Pfister, "A semidefinite programming based search strategy for feature selection with mutual information measure," *IEEE transactions on pattern analysis and machine intelligence*, vol. 37, no. 8, pp. 1529–1541, 2014.

[36] M. X. Goemans and D. P. Williamson, "Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming," *Journal of the ACM (JACM)*, vol. 42, no. 6, pp. 1115–1145, 1995.

[37] X. V. Nguyen, J. Chan, S. Romano, and J. Bailey, "Effective global approaches for mutual information based feature selection," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 512–521, 2014.

Declaration

I declare that this thesis is the solely effort of the author. I did not use any other sources and references than the listed ones. I have marked all contained direct or indirect statements from other sources as such.

Neither this work nor significant parts of it were part of another review process. I did not publish this work partially or completely yet. The electronic copy is consistent with all submitted copies.

Signature and date: Shilin Zhang Aug 8th 2022

A Appendix

A.1 Python Quaprog Library

https://github.com/quadprog/quadprog
https://github.com/stephane-caron/qpsolvers

A.2 Project Code

Shilin Zhang_ K20057905_ Carmine Ventre_ SupplementalFile_ 2021-22.zip

A.3 Date Source

Yahoo finance
https://finance.yahoo.com/

UCI machine learning repository
https://archive.ics.uci.edu/ml/index.php

A.4 Featurewiz

https://github.com/AutoViML/featurewiz