HELSINKI UNIVERSITY OF TECHNOLOGY
Department of Computer Science and Engineering

INPUT SELECTION USING MUTUAL INFORMATION – APPLICATIONS TO TIME SERIES PREDICTION

Jin Hao

Laboratory of Computer and Information Science
Espoo 2005

HELSINKI UNIVERSITY OF TECHNOLOGY
Department of Computer Science and Engineering

ABSTRACT OF MASTER'S THESIS

Author:	Jin Hao	
Title of thesis:	Input Selection Using Mutual Information - Applications to Time Series Prediction	
Date:	September 1 2005	**Pages:** 12 + 65
Professorship:	Computer information science	**Code:** T-61
Supervisor:	Professor Olli Simula	
Instructor:	Doctor Amaury Lendasse	

Input selection aims at selecting the most relevant inputs set for a given task. This problem is complex and remains an important issue in many domains. The main goal of this thesis is to show how mutual information can be used for the input selection in time series prediction problem. Mutual information measures the relationship between input variables and output.

First, the problem of input selection for time series prediction is generally explained.

Then, various mutual information estimation methods are reviewed and compared. Here, we focus on two effective estimators in the case of high dimensional data space. The estimator based on the k-Nearest Neighbours statistics is proposed.

After that, different algorithms for implementing mutual information for input selection are explored. The aim is to select the best set of inputs which is the one that maximizes mutual information.

Finally, the proposed methodology is applied to several experiments and it is proved to be a useful input selection method in the problem of time series prediction.

Keywords: time series, input selection, mutual information
k-nearest neighbours, least squares support vector machines
Language: English

TEKNILLINEN KORKEAKOULU DIPLOMITYÖN TIIVISTELMÄ
Tietotekniikan osasto

Tekijä:	Jin Hao	
Työn nimi:	Input Selection Using Mutual Information - Applications to Time Series Prediction	
Päiväys:	10. maaliskuuta 2005	**Sivumäärä:** 12 + 65
Professuuri:	Informaatiotekniikan	**Koodi:** T-61
Työn valvoja:	Professor Olli Simula	
Työn ohjaaja:	Doctor Amaury Lendasse	

Syötteen valinnan tavoite on annettua tehtävää varten olennaisimpien syötteiden valinta. Tämä ongelma on kompleksi ja tärkeä monilla aloilla. Tämän diplomityön päätarkoitus on näyttää, kuinka keskinäisinformaatiota voidaan käyttää syötteen valinnassa aikasarjojen ennustus ongelmassa. Keskinäisinformaatio mittaa suhdetta syötemuuttujien ja ulostulon välillä.

Aluksi syötteen valinta aikasarjojen ennustuksessa selitetään yleisellä tasolla.

Tämän jälkeen erilaisia keskinäisinformaation estimointimenetelmiä esitetään ja verrataan keskenään. Keskitymme tässä korkeaulotteisiin data-avaruuksiin. K:n lähimmän naapurin statistiikkaan perustuvaa estimaattoria ehdotetaan.

Tämän jälkeen tarkastellaan erilaisia algoritmeja keskinäisinformaation implementoitiin syötteen valintaa varten. Tavoite on sen muuttujajoukon valitseminen, joka maksimoi keskinäisinformaation.

Lopuksi ehdotettua metodologiaa sovelletaan useisiin kokeisiin ja sen osoitetaan olevan käyttökelpoinen muuttujanvalintamenetelmä aikasarjojen ennustuksessa.

Avainsanat:	aikasarjat, syötteen valinta, keskinäisinformaatio k:n lähimmän naapurin menetelmä
Kieli:	Englanti

Acknowledgements

This thesis has been made in the Laboratory of Computer and Information Science in the Helsinki University of Technology.

I would like to start by thanking my supervisor, Professor Olli Simula, for his supervision and encouragement, as well as my instructor, Doctor Amaury Lendasse, for helping and guiding me through the long process of doing experiments and creating a thesis. I enjoyed many instructive and inspiring conversations that have given me deeper insight into the field of neural network and machine learning. It has been a great pleasure working with him.

The Laboratory of Computer and Information Science has provided an excellent environment for research, both in terms of facilities and inspiration.

I am also deeply indebted to all the members in the time series prediction group, for the invaluable support and cooperation, Antti Sorjamaa, Nima Reyhani, Yongnan Ji, Elia Liitiäinen, Tuomas Kärnä, and other past and present members. Hope we have chance to continue have fun in the future...

Most of all, I would like to dedicate this thesis to my dear parents. Special thanks to you for your support and love.

Espoo September 1 2005

Jin Hao

Abbreviations and Acronyms

AMIFS	Adaptive Mutual Information-based Feature Selection
ANNs	Artificial Neural Networks
HB	Histogram-Based
HDV	High Dimensional Volume
iid	independent identically distributed
KB	Kernel-Based
k-NN	k-Nearest Neighbours
LOO	Leave-One-Out
LS-SVMs	Least Square Support Vector Machines
LS-SVR	Least Square Support Vector Regression
MAE	Mean Absolute Error
MI	Mutual Information
MIFS	Mutual Information-based Feature Selection
MIFS-U	Mutual Information-based Feature Selection deduced from Uniform distribution
MSE	Mean Squre Error
NNE	Nonparametric Noise Estimator
pdf	probability density function
SVMs	Support Vector Machines

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Scope of the thesis

Input selection is one of the most important issues in machine learning, especially when the number of observations is relatively small compared with the number of inputs. It has been the subject in application domains like pattern recognition, process identification, time series modelling and econometrics. In this thesis, we focus on its application to the time series prediction problem, which is an important part of decision making and planning process in such as engineering, business and medicine.

In practice, when dealing with the problem of time series prediction, the necessary size of the data set increases dramatically with the number of observations (curse of dimensionality). To circumvent this, one should select the best features or inputs in the sense that they contain the necessary information. Then it would be possible to capture and reconstruct the underlying regularity or relationship between input and output data pairs.

With respect to this, several approaches have been proposed. Some of them deal with the input selection problem as a generalization error estimation problem. These approaches are very time consuming and may take prohibitive amount of time. However, there are other approaches [1, 2, 3] which select a priori inputs based only on the data set, so the computational cost would be less than the cost of the model dependent cases. Model independent approaches select a set of inputs by optimizing a criterion over different combinations of inputs. The criterion computes the dependencies between this combination of input variables and the output. Various alternatives of criterion exist.

In this thesis, the mutual information (MI) between the selected inputs and the output is used as the criterion. Basically, MI measures the amount of information contained in an input variable or a group of input variables, in order to predict the output. It has the advantage to be model-independent and nonlinear at the same time.

When one is to implement the MI based input selection approach, estimation of MI poses great challenge. Histogram based estimator is a simple and efficient estimator, but the accuracy of most histogram estimators is substantially degraded in high-dimensional data space. Compounded by this problem, continuous kernel based estimators are considered as a good alternative for estimating MI. However, the computational load of these methods increases rapidly with the number of data points, and the performances of them would be significantly degraded when there is highly redundant input variable in the high-dimensional data space. The above problem is addressed by two new parametrical methods introduced in this thesis. One is based on the idea of estimating local density in the joint space by computing the high-dimensional volume; the other is based on the k-Nearest Neighbours (k-NN) statistics. In this thesis, for the second estimator, a method is developed to find the optimal k value by l-Nearest Neighbours (l-NN[1]) approximator and Leave-One-Out (LOO) method.

With the estimation results of MI, one will need some input selection strategies to select the optimal inputs subset from a series of dedicate inputs. The optimal algorithm is to compute the MI for all the possible combinations of inputs, but it is extremely heavy from the computational point of view. There are various other input selection strategies, such as forward selection, backward elimination, forward-backward selection algorithm,...,etc. In this thesis, all of the input selection strategies will be compared from both theory and experiment aspect.

The aims in this thesis are to explore the problem of input selection, give a literature survey, and present one input selection method which is based on MI. Finally, the proposed MI based input selection method is applied to the long-term time series prediction problem.

1.2 Publications

The publications related to this work are:

Publication [4] discusses the use of l-NN approximator to select the main variable in MI based estimator, and the selection results are applied for the long-term time series prediction problem.

In Publication [5], a comparison of direct and recursive method for long-term time series prediction is presented. In the paper, MI is used to select the inputs and least squares support vector machines (LS-SVMs) are used as the

[1] l is used instead of k here to avoid confusion with the k appearing before.

prediction model.

In Publication [6], a comparison work of two input selection approaches is done. One is based on MI; the other is based on nonparametric noise estimator (NNE). Both of these methods are applied into the problem of function approximation and time series prediction.

1.3 Structure of the thesis

This thesis is organized as follows. In this chapter, the problem, goals and publications associated with this work have been presented.

In Chapter 2, an overview of the time series prediction and input selection problems is given.

In Chapter 3, the theory of MI is explored and the MI estimation methods are discussed and compared. Then, the one that performs best is promoted.

In Chapter 4, different input selection strategies are presented and the performances of them are compared.

Chapter 5 discusses the application of MI based input selection into long-term time series prediction problem. Two experiments are performed there. In both experiments, three different time series are used.

Finally, Chapter 6 gives a conclusion of the work and plan of future works.

Chapter 2

Time series prediction and input selection

2.1 Time series prediction

2.1.1 Introduction of time series prediction

Time series prediction plays an important role in many domains of science and engineering, such as finance [7], electricity [8], environment [9] and ecology [10]. Basically, time series prediction can be considered as a modelling problem: a model is built to establish a mapping between the input(s) and output(s). After such a mapping is set up, it can be used to predict the future values based on the previous and current values.

A time series is a sequence of observations made through time, in the form of vector or scalar. In general, a time series may exhibit non-linearity, non-stationarity, possibly periodic behaviour such as seasonality. Furthermore, observations may be contaminated by noise. Figure 2.1 shows a noisy, stationary and non-periodic time series. These four characteristics of a time series are described as follows:

- Linearity: A time series is linear if the future values can be expressed as a linear function of some or all of its previous values. In this thesis, we are interested in developing general models that can represent nonlinear time series, which include the linear case.
- Stationarity: A time series is stationary if its mean and variance are constant in time and the auto-covariance depends on the time lag only [11]. We are interested in stationary time series and a special class of non-stationary time series, which consists of several regimes in which each regime corresponds to a chaotic process, and the overall

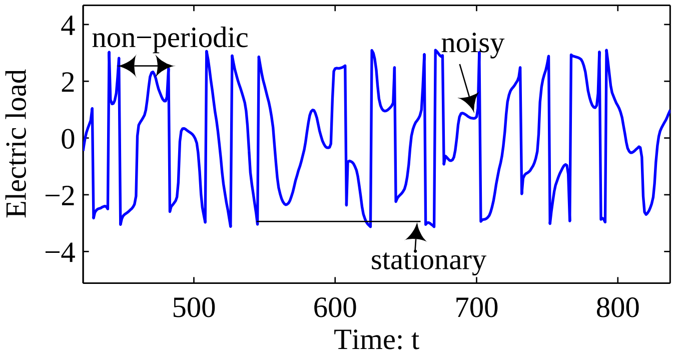

Figure 2.1: An example of a noisy, stationary and non-periodic time series

time series is a collection of multiple chaotic regimes, such as *Santa Fe* data set [12].

- Periodicity: A time series with periodic components are periodic. The periodic time series is easier to predict and in this thesis, we study the non-periodic one.

- Noise: Random noise can be present in the entire or some parts of a time series.

The time series prediction problem is the prediction of future values based on previous and current values:

$$\begin{aligned}\{\hat{y}(t+1), \hat{y}(t+2), \cdots, \hat{y}(t+h)\} &= F(y(t), y(t-1), \cdots, \\ &\quad y(t-p+1)),\end{aligned} \tag{2.1}$$

where h is the time steps parameter, representing the h step ahead prediction. F are the multi-output nonlinear prediction models as $F = \{f_1, f_2, \cdots, f_h\}$. Vectors $\{y(t), y(t-1), \cdots, y(t-p+1)\}$ are the regressor of size p.

Usually, one only needs to make a one-step ahead forecast ($h = 1$), which can be called short-term prediction. When $h > 1$, it is considered as the long-term prediction.

The long-term prediction is more difficult and time consuming, since it adds more uncertainty in the prediction of future values. Basically, two strategies can be used for the long-term prediction problem: direct and recursive forecasts. A comparison work of these two methods can be found in [5] and it will be detailed in Section 5.1. The direct forecast builds different models for each $\hat{y}(t+h)$ as:

$$\hat{y}(t+1) = f_1(y(t), y(t-1), \cdots, y(t-p+1)), \tag{2.2}$$

$$\hat{y}_2(t+2) \quad = \quad f_2(y(t), y(t-1), \cdots, y(t-p+1)),$$

The recursive forecast first makes one step ahead prediction as:

$$\hat{y}(t+1) \quad = \quad f_1(y(t), y(t-1), \cdots, y(t-p+1)), \tag{2.3}$$

and then predict the next value with the same model as:

$$\hat{y}(t+2) \quad = \quad f_1(\hat{y}(t+1), y(t), y(t-1), \cdots, y(t-p+2)), \tag{2.4}$$

2.1.2 Modelling methods

There exist many models for solving the time series prediction problem. The task is to select the best model according to some criteria such as the generalisation error. A large variety of time series models have been proposed and studied in the last four decades. Figure 2.2 lists a classification of the various prediction models. Next, we will introduce the methods following this classification, and then, give more explanation for the method we will use, as shown with thick arrows in Figure 2.2.

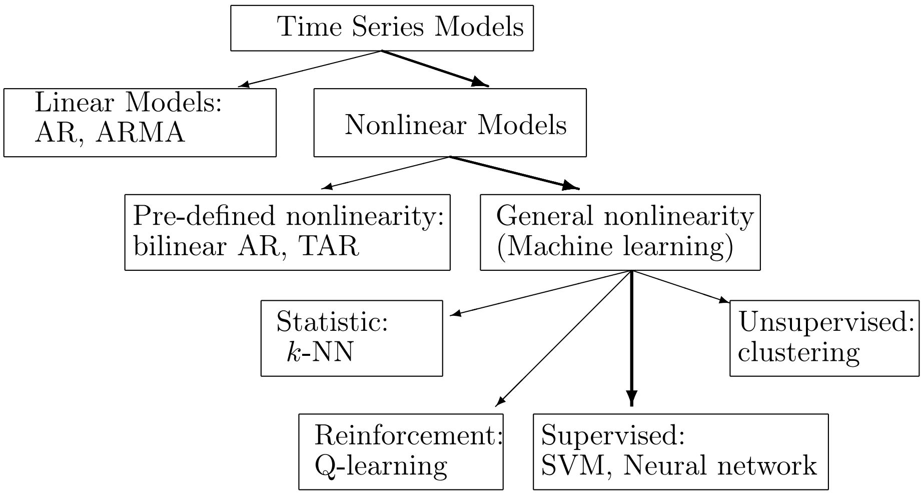
Time Series Models
Linear Models: AR, ARMA
Nonlinear Models
Pre-defined nonlinearity: bilinear AR, TAR
General nonlinearity (Machine learning)
Statistic: k-NN
Unsupervised: clustering
Reinforcement: Q-learning
Supervised: SVM, Neural network

Figure 2.2: A classification of time series models

Linear models

The simplest widely used model is the linear model, such as AR, ARMA, ARMAX,...,etc [13, 14]. This model is easy to use, and does not suffer too much from the choice of structural parameters. It performs really well in many cases, but will fail when the data is substantially nonlinear.

Nonlinear models

Nonlinear models can be classified into models with pre-defined non-linearity assumptions and general models, as shown in Figure 2.2. The first class includes bilinear autoregression [15], time-varying parameter models [16],...,etc. They are not effective for modelling time series with unknown nonlinear behaviour. The second class, which is also called machine learning, can handle nonlinear time series because it learns a model without non-linearity assumptions. Specific methods of machine learning include statistic learning (such as k-NN [17]), reinforcement learning (such as Q-learning [18]), unsupervised learning (such as clustering methods [19]), and supervised learning (such as support vector machines (SVMs) [20], LS-SVMs, and artificial neural networks (ANNs) [21]). In this thesis, we use the LS-SVMs because of their ability to escape from the local minima problem.

Least squares support vector machines Support vector machines, introduced by Vapnik, have been applied successfully to solve numerous problems in classification and regression. Trafalis and Santosa [22] used support vector regression (SVR) along with a feed-forward neural network and radial basis function networks to predict monthly flour prices in three cities. The results also showed that SVR outperformed the two other methods.

LS-SVMs [23] for function estimation were introduced by Saunders [24] as an interpretation of ridge regression in dual variables space. This approach is closely related to SVMs. Suyken [23, 25] then developed LS-SVMs and weighted LS-SVMs for function estimation. Compared to standard SVR, Least squares support vector regression (LS-SVR) is more effective in term of time complexity, since a linear system of equations is solved instead of a quadratic programming problem. The LS-SVMs are defined in its primal weight space by,

$$\hat{y} = \omega^T \varphi(\mathbf{x}) + b, \tag{2.5}$$

where $\varphi(\mathbf{x})$ is a function which maps the input space into a higher dimensional feature space, $\mathbf{x}$ is the N-dimensional vector of inputs $\mathbf{x}^i$. ω and b are the parameters of the model. In LS-SVMs for function estimation, the following optimization problem is formulated,

$$min_{\omega,b,e} J(\omega, e) = \frac{1}{2}\omega^T \omega + \gamma \frac{1}{2}\sum_{i=1}^{N}(e^i)^2, \tag{2.6}$$

subject to the equality constraints,

$$\begin{aligned} y^i &= \omega^T \varphi(\mathbf{x}^i) + b + e^i, \\ i &= 1, 2, \cdots, N, \end{aligned} \tag{2.7}$$

where the superscript i refers to the number of a sample. Solving this optimization problem in dual space leads to finding the α^i and b coefficients in the following solution,

$$h(\mathbf{x}) = \sum_{i=1}^{N} \alpha^i K(\mathbf{x}, \mathbf{x}^i) + b. \tag{2.8}$$

Function $K(\mathbf{x}, \mathbf{x}^i)$ is the kernel defined as the dot product between $\varphi(\mathbf{x}^T)$ and $\varphi(\mathbf{x})$ mappings. The meta-parameters of LS-SVMs models are the width of the Gaussian kernels (taken to be identical for all kernels) and the γ regularization factor. LS-SVMs can be viewed as a form of parametric ridge regression in the primal space. The training method for the estimation of the ω and b parameters can be found in [23].

Learning, validation and testing In order to learn the parameters of the nonlinear model, the traditional way is to divide the available data into three non-overlapping sets, respectively referred to as learning set (L), validation set (V), and testing set (T) [26]. Once the number of parameters of the model has been chosen, L will be used to learn the values of the parameters for the model. Normally, the performances of several models with different parameter values are compared, and the optimal parameters can be selected based on the performance evaluation. The performance evaluation must be done on an independent set of data, and the data set V is used for this task. Finally, set T is used for assessing the performance of the model selected after validation step.

However, the idea of splitting the data set into three independent sets has some problems. In many real situations, there is only a limited number of data available, but with the learning set L, only part of the original data is used for learning. To circumvent this problem, the re-sampling technique is used, such as d-fold cross-validation [2][1], LOO and bootstrap [1]. The main idea of these approaches is to repeat the learning and validation procedure with different divisions of the original data set. For example, in d-fold cross-validation [2], first, a part of data is set to be the testing set T. After that, the remaining data set are split into d sets with equal size. Then, d learning procedures are performed, each time, one of the d sets is taken as validation set V and the other sets as learning sets L. In this way, each sample except the ones in set T has been used both for learning and validation.

[1] d is used instead of k here to avoid confusion with the k appearing before in Section 1.1.

2.2 Input selection

Most of the nonlinear models perform rather poorly when faced with many irrelevant or redundant inputs. Usually, when the number of parameters is small, the model is not complex enough and the prediction will not be very accurate. On the contrary, if there are too many parameters, the parameters will try to capture also the noise contained in the data. This is the so-called overfitting phenomenon. The overfitting problem increases with model complexity, thus, it is more difficult to handle when there are many inputs. Therefore, some strategies are needed for choosing a set of most relevant inputs for building the model. This is the problem of input selection. The aim of input selection is to reduce the inputs as much as possible in order to improve the quality of the model built, and to improve the interpretability of the selected set of inputs.

Input selection is an essential pre-processing stage to guarantee high accuracy, efficiency, and scalability [27] in problems such as machine learning, especially when the number of observations is relatively small compared to the number of inputs. It has been the subject in application domains like pattern recognition, process identification, time series modelling and econometrics. Problems which can occur due to poor selection of input variables include:

- If the input dimensionality is too large, the 'curse of dimensionality' problem [28] may happen. Moreover, the computational complexity and memory requirement of the learning model will increase.
- Poor model may be built with additional unrelated inputs or not enough relevant inputs.
- Understanding complex models which contain too many inputs is more difficult than simple models with less inputs which can give comparable good performance.

Many input selection algorithms have been devised for this task. In this section, a general introduction is given, then, the promoted method will be explained elaborately in Chapters 3 and 4.

Usually, the input selection methods can be divided into two broad classes: *filter* method and *wrapper* method, see Figure 2.3.

2.2.1 Filter method

In case of the filter method, the best inputs subset is selected a priori based only on the data set. The input selection procedure in this case can be

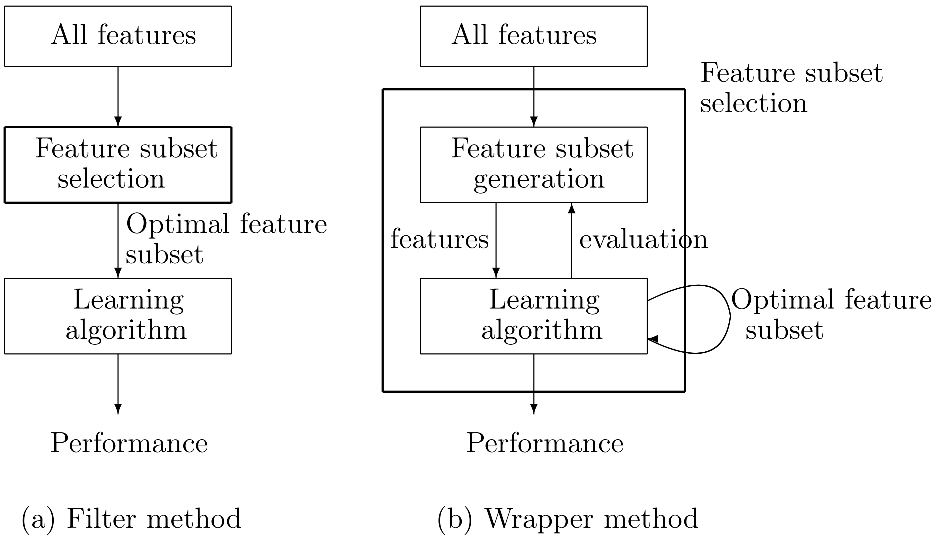

Figure 2.3: Two approaches of input variable subset selection

considered to be a pre-processing step, which is independent of the learning algorithm. The inputs subset is chosen by an evaluation criterion, which measures the relation of each subset of input variables with the output. The literature has plenty of filter measure methods with different natures [29]: distance metrics, dependence measures, scores based on the information theory,...,etc.

2.2.2 Wrapper method

In case of the wrapper method, the best inputs subset is selected according to the criterion which is directly defined from the learning algorithm. The wrapper method searches for a good subset of inputs using the learning model itself as a part of the evaluation function, which is the same algorithm that will be used to induce the final learning model. After selecting a model, the wrapper method uses an inputs subset and optimizing the parameters of the model by measuring some cost functions. Then, the inputs subset is changed and the same procedure repeated. Finally, the set of inputs that minimizes the generalization error can be selected using LOO, bootstrap or other re-sampling techniques.

2.2.3 Comparison

Comparing these two types of input selection strategies, the wrapper method tries to solve the real problem, hence the criterion can be really optimized for the specific problem. But it is potentially very time consuming, as the ultimate problem has to be included in the cost function, which may be evaluated thousands of times when searching for the best subset. For example, if LS-SVMs, which is introduced in Section 2.1.2, is used as the learning algorithm. Suppose we have 1000 data, then, for each subset of inputs, LS-SVMs need about 10 hours for evaluation. Thus, if we have 10 input variables and we use the forward selection strategy, which will be explained in Section 4.1.2, to select one more input at each step, we need to test $10(10-1)/2$ different subsets of inputs, so we will need 450 hours to evaluate all of the different subsets in this case, which is more than 2 weeks. In practice, we usually have more than 10 inputs in the time series prediction problem, and we need to use other selection strategies which need more operations than the forward selection method, the computational time using wrapper method will thus increase dramatically.

On the contrary, the filter method is much faster because the problem it solves is in general simpler. Due to the long computational time the wrapper method needs, it is unrealistic to compare the wrapper and filter method for input selection in time series prediction problem in this thesis.

2.2.4 Proposed method

In the following sections, we will focus on the filter method. As has been introduced in Section 2.2.1, the filter method selects a set of inputs by optimizing a criterion over different combinations of inputs. The criterion computes the dependencies between each combination of inputs and the output using predictability, correlation, MI or other statistics. So, there are two problems we need to solve for the input selection now:

1. Find an input evaluation criterion to compare the input variable subsets. In this thesis, MI is used as the evaluation criterion. MI based input selection has been developed more recently and is a general selection technique which is data hypothesis free and might be used for any system. It is based on a probabilistic dependence measure between two sets of variables. MI will be explained in detail in Chapter 3.

2. Find a search procedure, to explore a (sub)space of possible inputs subsets and select the most relevant inputs for the output with respect to the specific criterion. In Chapter 4, different search algorithms will be introduced and compared.

Chapter 3

Input selection method using mutual information

3.1 Definition of mutual information

Mutual information can be used for evaluating the dependencies between random variables. The MI of two variables, let say X and Y, is the amount of information obtained from X in the presence of Y. In time series prediction problem, if Y is the output and X is a subset of input variables, the MI between X and Y is particularly the criterion for measuring the dependence of Y on X. Thus, the inputs subset X giving maximum MI should be selected to predict the output Y.

The definition of MI begins from the entropy in the information theory. For continuous random variables (scalar or vector), let $p_{X,Y}, p_X$ and p_Y represent the joint probability density function (pdf) and the two marginal density functions of the variables. The entropy of X is defined by Shannon as [30]:

$$H(X) \quad = \quad -\int_{-\infty}^{\infty} p_X(x) \log p_X(x) dx. \tag{3.1}$$

where 'logt' means natural logarithm in the following of this thesis so that information is measured in natural units. When we know Y, but X is unknown, the remaining uncertainty of X is measured by the conditional entropy as:

$$\begin{aligned} H(X|Y) \quad = \quad -\int_{-\infty}^{\infty} p_Y(y) \int_{-\infty}^{\infty} p_X(x|Y=y) \\ \log p_X(x|Y=y) dxdy. \end{aligned} \tag{3.2}$$

The joint entropy is defined to be:

$$H(X,Y) \quad = \quad -\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} p_{X,Y}(x,y) \log p_{X,Y}(x,y) dxdy. \tag{3.3}$$

The MI between variables X and Y can be defined as [31]:

$$\begin{aligned} I(X,Y) &= H(Y) - H(Y|X) \\ &= H(X) + H(Y) - H(X,Y), \end{aligned} \tag{3.4}$$

It measures how much information one variable contains of another. From Eqs. 3.1, 3.3, 3.3 and 3.4, MI can be computed as:

$$I(X,Y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} p_{X,Y}(x,y)\log\frac{p_{X,Y}(x,y)}{p_X(x)p_Y(y)}dxdy. \tag{3.5}$$

From Eq 3.5, it can been seen that what we need for computing MI is only the estimations of the pdfs $p_{X,Y}, p_X$ and p_Y.

3.2 Estimating mutual information

As described in the previous section, the challenge of estimating MI lies in how to estimate the pdf values. There exist mainly three types of pdf estimation methods, namely, histogram based method, kernel based method and parametrical method. In the following, these three types of methods will be compared and the one suitable for input selection in our case will be proposed.

3.2.1 Histogram based estimator

For estimating pdf value, the most straightforward and widely used approach is the histogram based method (HB) [30, 32]. The basic idea of it is to divide the continuous input space into several discrete partitions. The entropy and MI can thus be estimated by substituting the integration operation by addition operation. Consider a collection of N variables x and y: $(x^i, y^i), i = 1, \cdots, N$, which are assumed to be independent and identically distributed (iid) realizations of a random variable $Z = (X, Y)$. With an origin a and a width h, the bins of the histogram for the variable x are defined through the intervals $[a + dh, a + (d + 1)h]$ with $d = 0, \cdots, D - 1$ [32], (Note that in [32], $d = 0, \cdots, D$ is incorrect). Hence, the data are partitioned into D bins $B^i, i = 1, \cdots, D$. Let g^i denotes the number of measurements lying inside the bin B^i, the pdf of x: p^i, can then be approximated by $p^i \approx g^i/N$. With the same idea, the joint density can be estimated by the number of measurements falling into the intersections of the bins of x and y coordinates. So, the true entropy in theory should be $H^{true} = -\sum_{i=1}^{D} p^i \log p^i$ and the observed entropy can be written as:

$$H^{observed} = -\sum_{i=1}^{D}\frac{g^i}{N}\log\frac{g^i}{N} \tag{3.6}$$

$$= -\sum_{i=1}^{D} q^i \log q^i,$$

where the notation $q^i = g^i/N$ has been used. It was also shown in [7] that:

$$\begin{aligned} E(g^i) &= Np^i, & (3.7)\\ E(q^i) &= p^i, & (3.8) \end{aligned}$$

where $E(.)$ represents expectation value.

It is known that the estimation of entropies from finite samples may be affected by systematic errors [33]. In [34, 35], the correction term has been used. Let us introduce a variable defined in [35],

$$\epsilon^i = \frac{q^i - p^i}{p^i}. \tag{3.9}$$

$H^{observed}$ in Eq. 3.7 can thus be written as:

$$\begin{aligned} H^{observed} &= -\sum_{i=1}^{D} p^i(1+\epsilon^i)\log(p^i(1+\epsilon^i)) \\ &= -\sum_{i=1}^{D} p^i(1+v^i)(\log p^i + \log(1+\epsilon^i))). & (3.10) \end{aligned}$$

When N is large enough, ϵ is small, Eq. 3.10 can be written in a Taylor series as:

$$\begin{aligned} H^{observed} &= -(\sum_{i=1}^{D} p^i \log p^i + \epsilon^i p^i(1+p^i) + \frac{(\epsilon^i)^2 p^i}{2} + O((\epsilon^i)^3)) \\ &= H^{true} - (\sum_{i=1}^{D} \epsilon^i p^i(1+p^i) + \frac{(\epsilon^i)^2 p^i}{2} + O((\epsilon^i)^3)). & (3.11) \end{aligned}$$

Since the expectation value of ϵ^i is zero, the expectation value of the observed entropy, to the second order in ϵ^i, will be:

$$E(H^{observed}) \approx H^{true} - (\sum_{i=1}^{D} \frac{E((\epsilon^i)^2)p^i}{2}. \tag{3.12}$$

From Eqs. 3.7, 3.8 and 3.9, it can be deduced that:

$$\begin{aligned} E((\epsilon^i)^2) &= \frac{(1-p^i)}{Np^i}, & (3.13)\\ p^i &\neq 0. \end{aligned}$$

Then, substituting Eq. 3.14 into Eq. 3.12 gives:

$$E(H^{observed}) \approx H^{true} - \frac{D-1}{2N}, \tag{3.14}$$

where D is the number of histogram bins with nonzero probability. From Eqs. 3.4 and 3.14, it can be found that:

$$\begin{aligned} I(X,Y)^{observed} &\approx I(X,Y)^{true} + \Delta I(X,Y), \\ \Delta I(X,Y) &= \frac{D_{xy} - D_x - D_y + 1}{2N}. \end{aligned} \tag{3.15}$$

Here, D_x, D_y, D_{xy} are the number of histogram bins with nonzero pdf.

3.2.2 Kernel based estimator

In [36], continuous kernel based pdf estimators (KB) are produced. Suppose a set of N M-dimensional training vectors $\{x^1, x^2, \cdots, x^N\}$, with a generalized kernel function $K(.)$, the pdf estimate is given by:

$$\hat{p}(x) = \frac{1}{N}\sum_{i=1}^{N} K(x - x^i, h), \tag{3.16}$$

where h is called the window width. Parzen showed that if the kernel function $K(.)$ and window width h are selected properly, the $\hat{p}(x)$ will converge to the true density function [36]. The Kernel function $K(.)$ is required to be a finite-valued non-negative density function where:

$$\int_{-\infty}^{\infty} K(z,h)dz = 1, \tag{3.17}$$

and the width parameter is required to be a function of n so that:

$$lim_{n\to\infty} h(n) = 0, \tag{3.18}$$

$$lim_{n\to\infty} nh^M(n) = \infty. \tag{3.19}$$

For kernel function, the rectangular and Gaussian functions are commonly used. In [37], Gaussian pdf estimators were chosen for estimating MI, where the Gaussian function is given as:

$$K(z,h) = \frac{1}{(2\pi)^{M/2} h^M |\sum|^{1/2}} \exp(-\frac{z^T \sum^{-1} z}{2h^2}), \tag{3.20}$$

with $\sum$ the covariance matrix of the M-dimensional vector of random variables z. The window width parameter h determines the influence field of the window, the smaller the h, the narrower the range of influence of window becomes. In this thesis, h is determined following the method in [38] as:

$$h = \frac{4}{(M+2)}^{\frac{1}{(M+4)}} N^{-\frac{1}{M+4}}, \tag{3.21}$$

where M is the dimension of variable z and N is the number of data points.

3.2.3 Parametrical estimator

In this thesis, two recent parametrical approaches for estimating MI are presented to deal with the problems in high-dimensional data space. One is based on the k-NN statistics [39]; the other uses the idea of high-dimensional volume [40].

k-NN based mutual information estimator

The novelty of this k-NN based MI estimator consists in its ability to estimate the MI between two variables of any dimensional space. Suppose one has a set of N input-output pairs $z^i = (x^i, y^i), i = 1, \cdots, N$, which are iid realizations of a random variable $Z = (X, Y)$, where x and y can be either scalars or vectors. Then, if z and z' are different variables from the data set, the maximum norm is,

$$\left\|z - z'\right\| \quad = \quad \max\{\left\|x - x'\right\|, \left\|y - y'\right\|\}, \tag{3.22}$$

The basic idea of [39] is to estimate $I(X, Y)$ from the average distances (estimated by the maximum norm) from z^i to its k nearest neighbours, averaged over all z^i. Let us denote $z^{k(i)} = (x^{k(i)}, y^{k(i)})$ as the k^{th} nearest neighbour of z^i, and $d^i = \left\|z^i - z^{k(i)}\right\|, d^i_X = \left\|x^i - x^{k(i)}\right\|, d^i_Y = \left\|y^i - y^{k(i)}\right\|$. Obviously, $d^i = \max(d^i_X, d^i_Y)$. Then, we count the number n^i_X of points x^j whose distance from x^i is strictly less than d^i, and similarly, n^i_Y is the number of points y^j whose distance from y^i is strictly less than d^i. Then, $I(X, Y)$ can be estimated as shown in [39]:

$$I(X, Y) \quad = \quad \psi(k) - \frac{1}{N}\sum_{i=1}^{N}[\psi(n^i_X + 1) + \psi(n^i_Y + 1)] + \psi(N), \tag{3.23}$$

where $\psi(.)$ is the digamma function:

$$\psi(t) \quad = \quad \frac{d}{dt}\log\Gamma(t), \tag{3.24}$$

where $\Gamma(.)$ is the gamma function:

$$\Gamma(t) \quad = \quad \int_0^\infty u^{t-1}e^{-u}du. \tag{3.25}$$

The digamma function satisfies the recursion $\psi(t + 1) = \psi(t) + 1/t$ and $\psi(1) = -c, c = 0.5772156\cdots$ is the Euler-Mascheroni constant. Software for calculating MI based on this method can be downloaded from [41].

For more variables such as $X_1, X_2, \cdots, X_M$, the MI estimate is defined as in [39]:

$$\begin{aligned} I(X_1, X_2, \cdots, X_M) &= \psi(k) - \frac{1}{N}\sum_{i=1}^{N}[\psi(n_{X_1}^i + 1) + \psi(n_{X_2}^i + 1) \\ &\quad + \cdots + \psi(n_{X_M}^i + 1)] + (M-1)\psi(N). \end{aligned} \tag{3.26}$$

From the grouping property of MI,

$$I(X, Y, Z) = I((X, Y), Z) + I(X, Y), \tag{3.27}$$

the MI between any set of random variables and any random variable can be computed by iterating Eq. 3.27. This is important in this thesis for input selection, as what we need to estimate is the MI between any inputs subset and the output.

The estimation of MI of this method depends on the pre-decided value k. It is explained in [39] that statistical errors increase when k decreases. In practice, it means that one should use $k > 1$ in order to reduce statistical errors. But on the other hand, too large values of k should be avoided since then the increase of systematic errors may outweigh the decrease of statistical errors. In [39], it is suggested to use a mid-range value k=6. But we found that when applied to time series prediction problem, it needs to be tuned for different data sets and different data dimensions to obtain better performance. In this thesis, to select the inputs based on this k-NN estimator when applied to the time series prediction problems, the optimal k value is obtained by l-NN and LOO methods. A general introduction of l-NN and LOO methods are given here, more details can be found in [4].

Leave-One-Out method Leave-One-Out method [3] is a special case of d-fold cross-validation re-sampling method. In d-fold cross-validation, training data is divided into d approximately equal sized sets. LOO procedure is the same as d-fold cross-validation with d equal to the size of the training set N. For each model to be tested, LOO procedure is used to calculate the generalization error estimated by removing each data point at a time from the training set, building a model with the rest of the training data and calculating the validation error with the one taken out. This procedure is done for every data point in the training set and the estimate of generalization error is calculated as a mean of all d, or N validation errors as shown in Eq. 3.28.

$$\hat{E}_{\text{gen}}(q) = \frac{\sum_{i=1}^{N}(h^q(x^i, (\theta^i)^*(q)) - y^i)^2}{N}, \tag{3.28}$$

where x^i is the i^{th} input vector from the training set, y^i is the corresponding output, h^q denotes the q^{th} tested model and $(\theta^i)^*(q)$ includes the model parameters without using (x^i, y^i) in training. Finally, as a result from the LOO procedure, we select the model that gives us the smallest generalization error estimate.

l-Nearest-Neighbours approximator l-Nearest Neighbours approximation method is a very simple, but powerful method. It has been used in many different applications and particularly in classification tasks [42]. The key idea behind l-NN is that similar input data vectors have similar output values. One has to look for a certain number of nearest neighbours, according to Euclidean distance [42], and their corresponding output values to get the output approximation. We can calculate the estimation of outputs by using average of the outputs of neighbours in the neighbourhood. If the pairs (x^i, y^i) represent the data with x^i as an M-dimensional input and y^i as a scalar output value, l-NN approximation is

$$\hat{y^i} = \frac{\sum_{j=1}^{l} y^{P(j)}}{l}, \tag{3.29}$$

where $\hat{y^i}$ represents the output estimation, $P(j)$ is the index number of the j^{th} nearest neighbour of input x^i and l is the number of neighbours used. We use the same neighbourhood size for every data point, so we use a global l, which can be determined by minimizing the error.

Selection of k for k-NN estimator using LOO and l-NN methods To select the parameter k for k-NN based MI estimator using the LOO and l-NN methods, three steps are followed:

1. First, the inputs are selected using the input selection process which will be introduced in Section 4. In the process, the MI is calculated by Eq. 3.26 with number of neighbours k varies from 2 to K, where K is the maximum nearest neighbours number that can be chosen by user.
2. Then, with each selected subset of inputs, l-NN and LOO methods are performed and the LOO error are calculated.
3. Finally, the selected inputs set which minimizes the LOO error are chosen and the corresponding k value is selected.

High-dimensional volume based mutual information estimator

This high-dimensional volume based MI estimator (HDV) is based on the idea of estimating local density in the joint space by computing the volume a single point x 'occupiest' in the sense of the maximal ball volume around x, which does not contain any other point [40]. Then for N sample points in the joint space, the average of N estimated local densities are used to compute the entropy.

Let $D = (x_1^i, x_2^i, \cdots, x_M^i), i = 1, \cdots, N$ be an ensemble of N data points in the M-dimensional joint space, with pdf value p_X. Suppose $x, x' \in D, x \neq x'$, take now

$$\begin{aligned} r_{x_m}^i &= \min(\|x_m^i - x_m'^i\|), \\ m &= 1, 2, \cdots, M+1, \end{aligned} \tag{3.30}$$

with $x_m^i, m = 1, 2, \cdots, M$ as the projection of point on the m^{th} coordinate of the joint space of D, and x_{M+1}^i represents the point in the joint space. So $r_{x_m}^i, m = 1, 2, \cdots, M$ are the minimum distances in each dimension of D and $r_{x_{M+1}}^i$ is the minimum distance in the joint space. Then,

$$V_x^i = B_m (r_{x_m}^i)^m, \tag{3.31}$$

where B_m is the volume of the m-dimensional unit ball. For the Euclidean norm which is used here:

$$B_m = \pi^{m/2}/\Gamma(1 + m/2) \tag{3.32}$$

Thus, V_x is the minimal ball volume around $x \in D$, which does not contain any other point in D.

Then, it is shown in [40] that the entropy can be accurately approximated as,

$$H(X) \approx \log N + \frac{1}{N}\sum_{i=1}^{N} \log V_x^i + c, \tag{3.33}$$

where c is the Euler-Mascheroni constant.

Consequently, the estimator leads to a very simple algorithm because the entropies for the marginal and joint distributions can both be computed by Eq. 3.33. The algorithm for estimating MI is as follows:

- Sample N points in $\mathbb{R}^M : D = (x_1^i, x_2^i, \cdots, x_M^i), i = 1, \cdots, N$.
- Compute in each dimension and in $\mathbb{R}^M$ the minimum distance as in Eq. 3.30, and the minimal volume as in Eq. 3.31, with $m = 1$ for each dimension of M and $m = M$ for the joint space in $\mathbb{R}^M$.
- Estimate the entropies $H(X)$ for each marginal of M dimensions and the joint space, as in Eq. 3.33.
- compute

$$\begin{aligned} I(X_1, X_2, \cdots, X_M) &= H(X_1) + H(X_2) + \cdots + H(X_M) \\ &\quad - H(X_1, X_2, \cdots, X_M). \end{aligned} \tag{3.34}$$

From the grouping property of MI, the $I((X,Y),Z)$ can then be estimated as,

$$\begin{aligned} I((X,Y),Z) &= I(X,Y,Z) - I(X,Y) \\ &= H(X) + H(Y) + +H(Z) - H(X,Y,Z) \\ &\quad -H(X) - H(Y) + H(X,Y) \\ &= H(Z) - H(X,Y,Z) + H(X,Y), \end{aligned} \tag{3.35}$$

and the MI between any set of random variables and any random variable can be computed by iterating it.

3.3 Comparison

Different MI estimation methods will be compared extensively in this section. They are histogram based estimator, kernel based estimator, k-NN based estimator and the high dimensional volume based estimator. These four approaches are first compared with respect to the algorithm theory. Then, they are applied to a typical data set with Normal distribution and the estimations of entropy and joint entropy are compared. Furthermore, the computational times are evaluated.

3.3.1 Algorithm theory

The first estimator introduced is the HB estimator. It can be seen from Eq. 3.16 that the finite-size corrections depend on the number of data points, and this method is feasible only when the number of data points is considerably larger than the number of the histogram bins. The accuracy of most HB estimators is dramatically degraded in high-dimensional space [21]. Hence, they can be applied to the problems with only one- or two-dimensional data space.

The KB estimator circumvented the problem of HB estimator for high-dimensional data space. However, the KB methods proposed in [37] still have some problems: the computational complexity of these methods increases rapidly with the size of data set; the performances of them will still be significantly degraded in high-dimensional data space; they cannot give respectable estimation when data set is relatively small.

In the two parametrical estimation methods, all of the above problems are addressed. These two approaches solve the problem of estimating the high-dimensional joint entropy in different ways based on the same starting point:

$$H(X) \approx -\psi(k) + \psi(N) + \frac{1}{N}\sum_{i=1}^{N} \log B_m(r^i)^m, \tag{3.36}$$

In k-NN estimator, the maximum norm is used. So, $B_m = 1$ and $2r^i = d^i$, Eq. 3.36 turns to be:

$$H(X) \approx -\psi(k) + \psi(N) + \log V_m + \frac{m}{N}\sum_{i=1}^{N} \log d^i, \tag{3.37}$$

In the HDV estimator, $k = 1$, which means the 1st nearest neighbour based on the Euclidean norm is used. With $\psi(N) \approx \log N$ when N is large, Eq. 3.36 becomes:

$$H(X) \approx \log N + \frac{1}{N}\sum_{i=1}^{N} \log B_m(r^i)^m - \psi(1), \tag{3.38}$$

So, the difference between these two methods is that the k-NN method estimates the joint entropy by using the idea of k nearest neighbours and while the HDV method estimates the joint entropy with the idea of computing the high-dimensional ball volume.

3.3.2 Implementation and results

It is noted from the definition of MI in Eq. 3.4 that to compute MI, what we need are the estimations of entropies in each dimension and the joint entropy. Thus, in this section, the estimations of entropy and joint entropy will be compared. To simpify the problem here, for the k-NN estimator, k is set to be 6.

Estimation of entropy

Let X be Gaussian with mean x^m and variance σ^2. So,

$$p_X(x) = \frac{1}{\sqrt{2\pi\sigma^2}} exp(-\frac{(x-x^m)^2}{2\sigma^2}), \tag{3.39}$$

The entropy should then be:

$$\begin{aligned} H(X) &= -\int_{-\infty}^{\infty} p_X(x)\log p_X(x)dx \\ &= -\int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi\sigma^2}} \exp(\frac{(x-x^m)^2}{-2\sigma^2}) \log(\frac{1}{\sqrt{2\pi\sigma^2}} \exp(\frac{(x-x^m)^2}{-2\sigma^2}))dx \\ &= \int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi\sigma^2}} \exp(-\frac{(x-x^m)^2}{2\sigma^2})(1/2)\log(2\pi\sigma^2) \\ &\quad + \int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi\sigma^2}} \exp(-\frac{(x-x^m)^2}{2\sigma^2})(1/2)(1/2)(-\frac{(x-x^m)^2}{\sigma^2})dx \end{aligned}$$

$$\begin{aligned} &= (1/2)\log(2\pi\sigma^2) + (1/2)E(\frac{(x-x^m)^2}{\sigma^2}) \\ &= (1/2)\log(2\pi\sigma^2) + 1/2 \\ &= (1/2)\log(2\pi\sigma^2 e), \end{aligned} \tag{3.40}$$

since

$$\int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi\sigma^2}} \exp(-\frac{(x-x^m)^2}{2\sigma^2})dx = 1. \tag{3.41}$$

$E(.)$ represents expectation value.

If X has zero mean and unit variance,

$$\begin{aligned} H(X) &\approx (1/2)\log(2\pi e) \\ &\approx 1.4189. \end{aligned} \tag{3.42}$$

So, the theoretical value of real entropy is already known. Now, let us compare the estimations of entropy from the four methods. A set of Gaussian X is generated with zero mean and unit variance. Here, the size of Gaussian X is set to be from 500 to 10000, with step size 500. For each size, 10 data sets were independently generated, the average results of these 10 trails are presented in Figure 3.1.

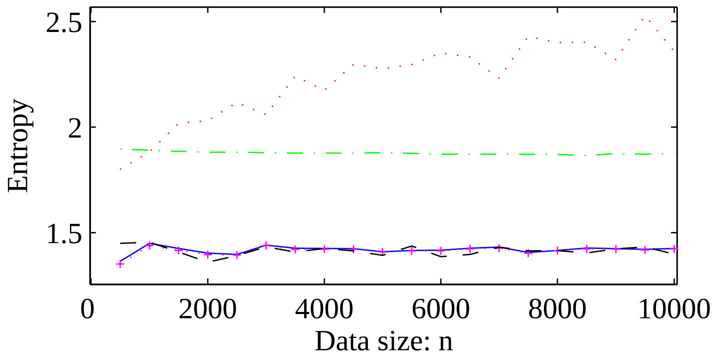

Figure 3.1: Comparison of entropy estimation: the solid line is the real value, dotted line with '+t' mark is from k-NN method, dashed line is from HDV method, dotted line is from HB method and dashed-dotted line from KB method

As is shown from Figure 3.1, both of the k-NN and HDV methods can give good estimation of entropy, and the k-NN method's results are more accurate. The KB method can give stable estimation with a constant offset from the real value. The HB method performs the worst.

Estimation of joint entropy

The estimations of joint entropy from different approaches will be compared in this section. Let X and Y be two Gaussians, with means x^m and y^m, variances σ_X^2 and σ_Y^2, and the covariance between them is r. In this case, $I(X,Y)$ is known exactly to be [43],

$$I(X,Y) = -\frac{1}{2}\log(1-r^2). \tag{3.43}$$

So, from Eqs. 3.4 and 3.40, the joint entropy can be computed as:

$$\begin{aligned} H(X,Y) &= H(X)+H(Y)-I(X,Y) \\ &= \frac{1}{2}\log(2\pi\sigma_X^2 e)+\frac{1}{2}\log(2\pi\sigma_Y^2 e)+\frac{1}{2}\log(1-r^2). \end{aligned} \tag{3.44}$$

Now, X and Y are generated to be two independent Gaussians. By computing the covariance r between X and Y, the real values of joint entropy can be estimated by Eq. 3.44. The size of data is set to be from 500 to 10000, with step size 500. For each size, 10 data sets were independently generated, and the average results of these 10 trails are shown in Figure 3.2:

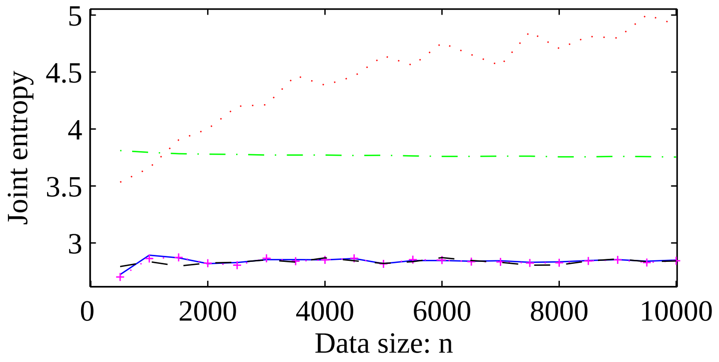

Figure 3.2: Comparison of joint entropy estimation: the solid line is the real value, dotted line with '+t' mark is from k-NN method, dashed line is from HDV method, dotted line is from HB method and dashed-dotted line from KB method

From Figure 3.2, it can be seen that for the estimation of joint entropy, both the HDV and k-NN methods' estimations are close to the real values. But the estimation results by HDV method is not as accurate as the k-NN method. The KB method's estimation is biased. The HB method gives the worst estimation.

Computational time

With respect to the computational time, as it mainly depends on the data size, in this test, we change the data size from 1000 to 5000, with step size 1000, and other parameters are fixed, the resulting computational times of four methods for different data sizes are shown in Table 3.1.

	1000 data	2000 data	3000 data	4000 data	5000 data
HB	0.0035	0.0064	0.0082	0.0101	0.0124
KB	23.19	46.03	73.01	102.39	133.73
HDV	0.50	1.91	4.96	9.61	22.64
k-NN	11.52	61.51	189.27	462.68	903.05

Table 3.1: Computational time of different estimation methods(in seconds)

It can be seen from Table 3.1 that the HB method consumes the least computational time, the HDV method is faster than the KB and k-NN based methods, and the k-NN method is the slowest when data size is large.

3.3.3 Conclusion of comparison

Based on the experimental results, the estimation accuracy and computational time of the four different methods are summarized in Table 3.2.

	Estimation of entropy	Estimation of joint entropy	Computational time
HB	- -	- -	+ +
KB	-	-	-
HDV	+	+	+
k-NN	+ +	+ +	- -

Table 3.2: Comparison of performance of different estimation methods: + + represents very good, + is good, - represents bad and - - means very bad

In the table, the k-NN method is shown to be most accurate in estimating entropy and joint entropy, and thus in estimating MI. As what we need in this thesis is to rank different inputs subset based on the MI between it and the output, the estimation of MI value needs to be as accurate as possible. With this respect, the estimator based on k-NN is proposed here.

Chapter 4

Mutual information based input selection process

4.1 Mutual information based input selection

The original input selection problem is to select the most k relevant input variables from a set of N inputs and Battiti named it as 'feature reductiont' problem in [30]:

> [FR$N - k$:] Given an initial set of N features, find the subset with $k < N$ features that is 'maximally informativet' about the class.

As shown in Chapter 3, the MI values can be used for selecting the relevant inputs, and the FR$N - k$ problem could then be reformulated as follows [30]:

> [FR$N - k$:] Given an initial set F with N features, find the subset $S \subset F$ with k features that i.e., maximizes the MI $I(C, S)$, where C represents the class.

In case of regression, we need to maximize the MI between a set of input variables and the output, but the selected input number k is unknown in our case. So, what we actually do is to test all the numbers of $k = 1, 2, \cdots, N$ and select the one giving maximum MI value. There are several strategies for solving both the problem of selecting the optimal number k and the best inputs subset. These approaches will be introduced in the following.

4.1.1 Exhaustive search

The optimal algorithm is to compute the MI for all the possible combinations of inputs, e.g. $2^M - 1$ input combinations are tested (M is the number of input variables). Then, the one that gives maximum MI is selected. However, it will be explained later that this procedure is too much time consuming.

4.1.2 Forward selection

In this method, starting from the empty set S of the selected input variables, the best available input is added to the set S one by one, until the size of S is M. Suppose we have a set of inputs $X_i, i = 1, 2, \cdots, M$ and the output Y, the algorithm is as follows:

1. (Initialization)
   Set F to be the initial set of original M inputs, and S to be the empty set which will contain the selected inputs.

2. (Selection of the first variable)
   Find:

   $$X_s = \arg\max_{X_i} I(X_i, Y), \qquad X_i \in F,$$

   where X_s represents the selected variable.
   Save $I(X_s, Y)$, and move X_s from F to S.

3. (Selection of the following variables)
   Find:

   $$X_s = \arg\max_{X_i} I(\{S, X_i\}, Y), \qquad X_i \in F.$$

   Save $I(\{S, X_s\}, Y)$ and move X_s from F to S.
   Continue with the same way, till the size of S is M.
   At the same time, save the MI value for each size of set S.

4. (Result)
   Compare the MI values for all the sizes of sets S, the selection result is set S with the corresponding size giving maximum MI.

4.1.3 Backward elimination or pruning

Backward elimination, also called pruning procedure, is the opposite of forward selection process. In this strategy, the selected inputs set S is initially set to contain all the input variables. Then, the worst input variable whose elimination gives the maximum MI between the rest of inputs and the output is removed from set S one by one, until the size of S is 1.

Suppose we have a set of inputs X_i, $i = 1, 2, \cdots, M$ and the output Y, the algorithm is like this:

1. (Initialization)
   Set S to be the set which contains all of the input variables.

2. (Elimination of the first variable)
   Find:
   $$X_r \quad = \quad \arg\min_{X_i} I(X_i, Y), \qquad X_i \in S.$$
   Save $I(S\backslash X_r, Y)$, where $S\backslash X_r$ means set S without X_r, and remove X_r from S.

3. (Elimination of the following variables)
   Find:
   $$X_r \quad = \quad \arg\max_{X_i} I(S\backslash X_i, Y), \qquad X_i \in S.$$
   Save $I(S\backslash X_r, Y)$, and remove X_r from S.
   Continue with the same way, till the size of S is 1.
   At the same time, save the MI value for each size of set S.

4. (Result)
   Compare the MI values for all the sizes of sets S, the selection result is set S with the corresponding size giving maximum MI.

4.1.4 Forward-backward selection

Both forward selection and backward elimination methods suffer from the incomplete search. Forward-backward selection algorithm combines both methods. It offers the flexibility to reconsider input variables previously discarded and vice versa, to discard input variables previously selected. It can start from any inputs set, even randomly initialized set.

Also suppose we have a set of inputs $X_i, i = 1, 2, \cdots, M$ and the output Y, the procedure of the forward-backward selection is:

1. (Initialization)
   Let set S to be the selected inputs set which can contain any input variables, and set F to be the unselected inputs set containing the rest inputs which are not in set S. Compute $I(S, Y)$.

2. (Forward-backward selection)
   Find:
   $$X_s \quad = \quad \arg\max_{X_{i,j}} \{I(\{S, X_j\}, Y)\} \cup \{I(S\backslash X_i, Y)\}, X_i \in S, X_j \in F.$$
   If the old $I(S, Y)$ is larger than the new MI, stop; otherwise, update set S and save the new MI, repeat step 2 till no further change can increase the MI value.

3. (Result)
   The selection result is in set S.

It is noted that the selection result depends on the initialization of the inputs set. Here, we consider two options. One is to begin from the empty set; the other is to begin from the full set $S = \{X_1, X_2, \cdots, X_M\}$.

4.1.5 MIFS, MIFS-U and AMIFS

The above four types of input selection algorithms can be used when MI between any combination of inputs and the output can be estimated. However, the estimators such as histogram based one can only estimate the MI between two variables. To avoid computing MI between high-dimensional vectors, so that we can use this kind of MI estimators, Battini [30] adopted a heuristic criterion which computes only $I(X_i, Y)$ and $I(X_i, X_s)$, instead of calculating $I(\{X_s, X_i\}, Y)$.

Three feature selection algorithms are developed based on this idea. They are MI based feature selection (MIFS) , modified MI based feature selection which is deduced from the uniform distribution data set (MIFS-U) and adaptive MI based feature selection (AMIFS). These three algorithms share the same idea with the forward selection algorithm. The only difference is in the third step.

Battiti's MIFS algorithm selects the input that maximizes the MI between the new input and the output, subtracted by a quantity proportional to the average MI between the new input and the already selected inputs. The third step in the forward selection algorithm is thus changed to:
Repeat until the size of S reaches M (M is the number of input variables):

$$X_{s_2} \quad = \quad \arg\max_{X_i}(I(X_i, Y) - \beta \sum_{s \in S} I(X_i, X_s)), \tag{4.1}$$

where β is a control parameter chosen by the user.

Kwak and Choi [44] made an enhancement of MIFS, called MIFS-U. It only changes the selection criterion in the MIFS and X_{s_2} is selected by:

$$X_{s_2} \quad = \quad \arg\max_{X_i}(I(X_i, Y) - \beta \sum_{s \in S} \frac{I(X_s, Y)}{H(X_s)} I(X_i, X_s)), \tag{4.2}$$

In both MIFS and MIFS-U algorithms, parameter β regulates the relative importance of MI between the candidate input and the already-selected inputs with respect to the MI with the output [30]. If parameter β is set to be zero, only the MI between the candidate input and the output is considered. As β increases, this measure is discounted by a quantity proportional to the total MI with respect to the already-selected inputs. However, the optimal value of parameter β is strongly dependent on the problem at hand [45]. In [30], a value for β between 0.5 and 1 is proposed, and $\beta = 1$ is used in [44].

In AMIFS [43], to avoid using the control parameter β which is hard to decide, the criterion is changed to be:

$$X_{s_2} \quad = \quad \arg\max_{X_i}(I(X_i, Y) - \sum_{s \in S} \frac{I(X_s, X_s)}{N_s \tilde{H}(X)}), \tag{4.3}$$

where $\tilde{H}(X) = \min(H(X_s), H(X_i))$ and N_s is the number of already selected inputs.

4.2 Comparison

Different input selection strategies introduced in the previous section are compared here. First, all of the approaches are compared from the algorithm theory point of view. Then, some of them are chosen for implementation based on the performance from theory viewpoint. A toy example of function approximation will be used. The MI estimator used here is the k-NN based estimator, which has been proposed in the end of Section 3.3.3. At the same time, the computational time will be compared.

4.2.1 Algorithm theory

Exhaustive search is the first choice when considering input selection. However, it requires a large number of computation. Indeed, for $M = 15$, where M is the number of input variables, the number of operation is 32767, so, it is not possible to test all the combinations of inputs when M exceeds such as 15. Therefore, in order to reduce the complexity when M is relatively large, some heuristic algorithms for selecting the inputs are need.

MIFS, MIFS-U and AMIFS are developed for finding the best inputs combination without considering the high-dimensional case. In all of these three approaches, the selection criterion is composed of two terms. The first one measures the relevance of the new input with the output; while the second one ensures that the redundant input which is quite similar with the already selected input variables will not be selected. However, MIFS, as in Eq. 4.1, is too simple that when there are many irrelevant and redundant inputs, its performance will degrade as it penalizes too much the redundancy. The MIFS-U algorithm, as in Eq. 4.2, can give better estimation than the original MIFS approach. But both MIFS and MIFS-U algorithms rely on the parameter β for balancing the redundancy penalization, whose optimal value is hard to choose and dependent on the specific problem, as pointed out in Section 4.1.5. AMIFS differs from the first two algorithms by replacing the fixed parameter β by an adaptive term. However, these three algorithms are developed to avoid estimating high-dimensional MI by doing it in an indirect way. This type of approaches might make it impossible to select truly informative input variables when the relationship between input and the output is strongly nonlinear. Consequently, as we have already the estimators such as k-NN based and HDV based methods which can solve the problem of high-dimensional data space, these three types of algorithm are not suitable for our case.

The forward-backward selection approach was generated as both the forward and backward selection methods suffer from the so-called *nesting effect* [46]. That means, once an input variable is discarded in the backward method, it is not possible to reconsider it anymore. The opposite is true for the forward selection: once an input variable is chosen, there is no way to discard it later on. It can be seen that although the forward-backward algorithm does not guarantee finding all of the best inputs, it results in substantially improvement compared with forward or backward processes.

4.2.2 Implementation and results

In this experiment, the MI is estimated by the k-NN method, whose k value is set to be 6, as it doesn't influence the results of comparison and can simplify the problem. Five input selection procedures will be compared by some toy examples of function approximation problem. The five methods are exhaustive search, forward selection, backward elimination, forward-backward selection from empty inputs set (forward-backward selection (a)) and forward-backward selection from full inputs set (forward-backward selection (b)). To test the robustness of these input selection methods working with different models, first, a linear model with changing parameters is tested. Then, a simple nonlinear model is used. After that, a nonlinear model with increasing noise is tested. Last, a nonlinear model with the size of data changed is used.

Linear model

Here, let X represents a uniform distributed 10-dimensional variable valued between 0 and 1. The number of observations of X is 1000 (the size of the data set). A linear model with four input variables with different coefficient values is built in order to see what is the robustness of the five methods to select the weakest input variable if some variables are more relevant to the output than the others.

- First, the following model is tested:

  $$Y_1 = aX_1 + 3X_2 + 3X_5 + 3X_{10}.$$

  a is the coefficient of the first variable. It is decreased from 3 to 0.1, with step size 0.1. It is found that all of the five methods can choose the four correct inputs (X_1, X_2, X_5 and X_{10}), until a is decreased to 1. After that, none of these five algorithms can select variable X_1 because the coefficient of it is too small comparing with other variables.

- Second, we increased the coefficients of X_5 and X_{10} :

  $$Y_1 = aX_1 + 3X_2 + 7X_5 + 9X_{10}.$$

  a is decreased in the same way. So Y becomes to be more dependent on X_5 and X_{10}. This time, after $a < 2.5$, no method can find the first input.

- Third, we increased the coefficients of X_2 and X_5:

  $$Y_1 = aX_1 + 9X_2 + 9X_5 + 9X_{10}.$$

  a is decreased from 5 to 0.1, with step size 0.1. With this model, after $a < 3.1$, the influence of variable X_1 is comparative too small that it can not be found out by any algorithm.

From this test, it is noted that all of these five methods can work well for the simple linear model, and they give similar performances in this test.

Nonlinear model

With the same X defined in the linear model, a nonlinear model with four input variables is built,

$$Y_2 = X_1X_2 + X_5 + \sin(X_{10}). \tag{4.4}$$

For this nonlinear model, all of the five different algorithms can select the correct inputs.

Nonlinear model with noise

It is noticed that the five approaches perform well on the simple nonlinear model. Now, we add a noise to the same nonlinear model, and investigate the robustness of different input selection approaches by increasing the noise. The model is built as:

$$Y = X_1X_2 + X_5 + \sin X_{10} + b\epsilon.$$

The noise ϵ is uniformly distributed in [-1, 1], X is the same as in the linear model, b is the weighting coefficient of noise increased from 10 to 200, with step size 10.

The performances of different methods is measured by good and bad inputs it selects. For example, in this example, X_1, X_2, X_5 and X_{10} are good inputs; while X_3, X_4, X_6, X_7, X_8 and X_9 are bad inputs. When the selected inputs by one algorithm is X_1, X_2, X_5 and X_9, then, Table 4.1 can be built. The criterion for the ability of selecting the correct inputs for this algorithm is defined to be: $(\frac{3}{3+1} + \frac{5}{5+1})/2$.

	Good Inputs	Bad Inputs
Selected Good Inputs	3	1
Non-Selected Bad Inputs	1	5

Table 4.1: Selected inputs classification

Based on the criterion defined above, the results of the five methods tested with different values of coefficient b is shown in Figure 4.1.

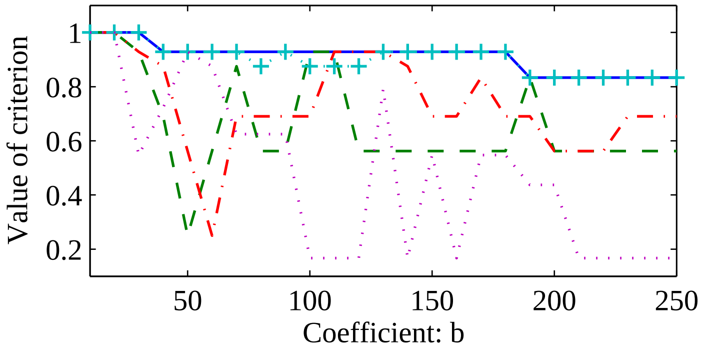

Figure 4.1: Comparison of input selection procedures with increasing noise: The solid line is from exhaustive search, the dashed line is from forward selection, and the dashed-dotted line from backward elimination, dotted line with '+t' mark is from forward-backward selection(a), and the dotted line is from forward-backward selection (b)

From Figure 4.1, it can be seen that when the noise increases, the performances of all of these five input selection strategies decrease. The exhaustive search can give the best result, and the forward-backward selection (a) performs better than the other algorithms. It is reasonable that forward-backward selection (a) is better than forward-backward seleciton (b), as the model includes 4 of the original 10 inputs, the algorithm initialized with the empty selected inputs set should work better than the one initialized from the full inputs set.

Change size of data set

Here, we change the size of data set to test the robustness of different methods with respect to data size. The size of data set is decreased from 1000 to 10, with step size 50. The model used is the same nonlinear model without noise, as in Eq. 4.4.

The criterion for evaluating the performances of different input selection algorithms are as defined in the previous test. The results based on this criterion are shown in Figure 4.2.

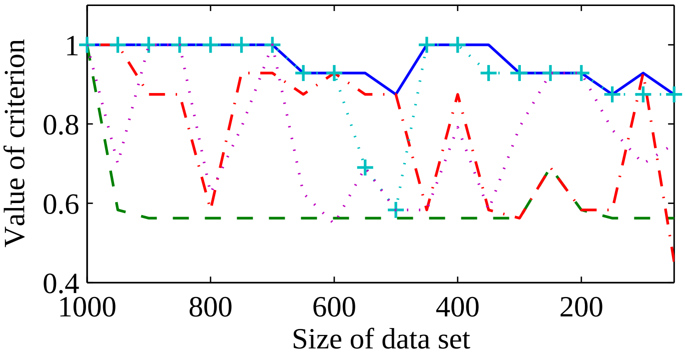

Figure 4.2: Comparison of input selection procedures with decreasing size of data set: The solid line is from exhaustive search, the dashed line is from forward selection, and the dashed-dotted line from backward elimination, dotted line with '+t' mark is from forward-backward selection(a), and the dotted line is from forward-backward selection (b)

It can be seen from Figure 4.2 that when there are less data, the performances of all these five input selection algorithms degrade. The exhaustive search still performs best in this case, and the forward-backward selection (a) performs better than the others.

Number of operation

Considering the number of operation, suppose M is the number of inputs, in the exhaustive search algorithm, $2^M - 1$ times of computation is needed. For both the forward selection and backward elimination algorithms, the number of operation is $\frac{M(M-1)}{2}$. For the forward-backward approaches, the number of operation varies. It depends on the initialization on the input set and the special problem. In theory, it needs around $M(M-1)$. However, with some prior knowledge to choose the initialization, the number of operation of forward-backward selection algorithms can decrease a lot.

Conclusion of comparison

Based on the experimental results, the input selection quality and the number of operation for the five different algorithms are summarized in Table 4.2.

	Input selection quality	Number of operations
Exhaustive search	+ +	-
Forward selection	-	+ +
Backward elimination	-	+ +
Forward-backward selection (a)	+	+
Forward-backward selection (b)	+	+

Table 4.2: Comparison of performance of the four input selection procedures: + + represents very good, + is good, and - represents bad

Comparing both the quality of input selection and the number of operation of the five algorithms, we can see that the exhaustive search performs best, but needs a large number of computations, hence, when the maximum number of inputs is less than for example 15, this algorithm is promoted. When the size of data set exceeds 15, the forward-backward selection process is better. In this thesis, to guarantee the performance, all of the four methods except the exhaustive search algorithm will be used to select the inputs for the time series prediction problem, and the best inputs subset giving maximum MI is selected, the process will be explained in Chapter 5.

Chapter 5

Application to time series prediction

The MI based input selection method introduced in chapters 3 and 4 is applied to the time series prediction problem in this chapter. Two experiments will be performed.

The first one is to compare the direct and recursive prediction strategies of the long-term time series using MI based input selection. LS-SVMs are used as nonlinear models to avoid local minima problems.

The second experiment is to compare the input selection method using MI with other two kinds of input selection methods: Nonparametric Noise Estimator and l-NN method. All of the three input selection methods will be applied to the problem of long-term time series prediction with direct prediction strategy and LS-SVMs learning models.

In both two experiments, three different time series will be used. The first one is the Santa Fe Laser data set [12], which has approximately 10000 points. It is a uni-variate time record of a single observed quantity, measured in a physics laboratory experiment. The second data set is the Poland Electricity data set, which has around 1500 points. It represents the daily electricity load of Poland in the 90s. The third time series is the Darwin sea level data set [47], which has 1400 data points. It is the monthly values of the Darwin sea level pressure series from year 1882 to 1998. In the experiments, for all of the three data sets, the first 1000 data set is used for training, and the remaining data for testing. The learning part of these three data sets are shown in Figure 5.1.

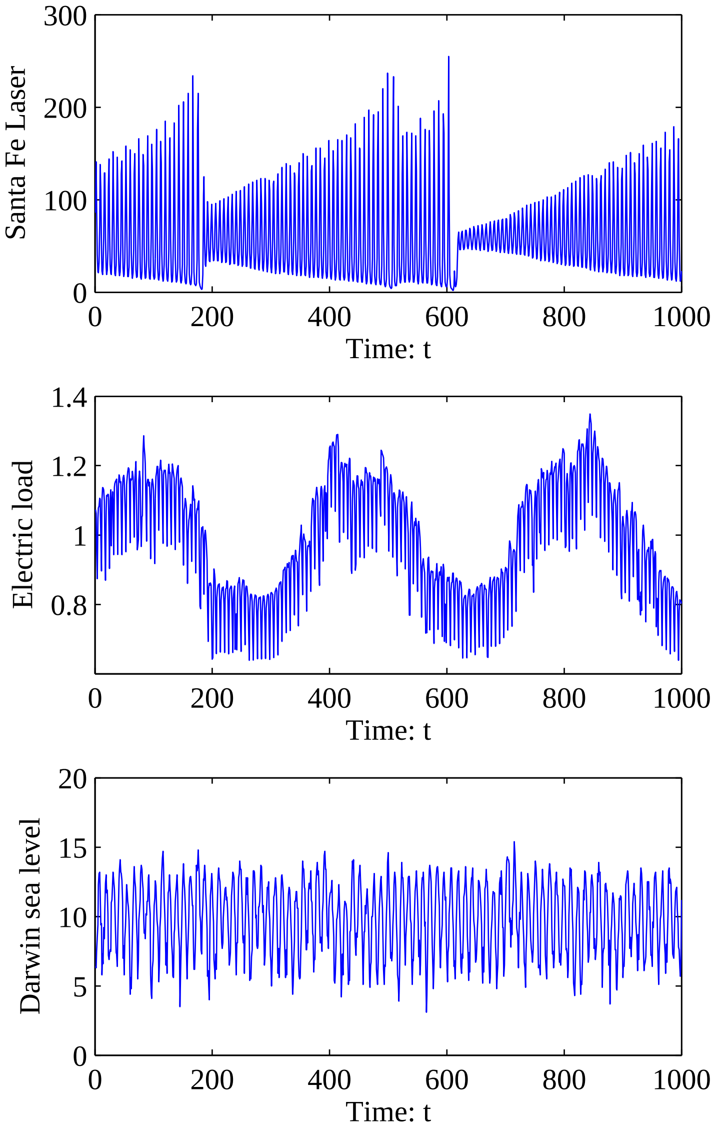

Figure 5.1: Learning part of three data sets: the first one is Santa Fe Laser data set, second one is Poland Electricity data set, and last one is Darwin sea level data set

5.1 Direct and recursive prediction of time series using mutual information based input selection

In this experiment, first, MI is used to select the inputs. The MI estimator used here is the k-NN estimator, and the input selection process is performed as follows:

- Estimate MI values based on k-NN method with different number of neighbours: $k = 2, \cdots, 10$.

  1. For each k value, four input selection processes: forward selection, backward elimination, forward-backward selection (a) and forward-backward selection (b), as introduced in Chapter 4, are performed to select the best inputs.
  2. From the four selected inputs sets, the one giving maximum MI between inputs and output is chosen.
  3. l-NN and LOO methods are used to calculate the LOO error for the chosen inputs subset.

- The selected k value is the one giving minimum LOO error and the corresponding inputs subset is finally selected.

In the following, for long-term time series prediciton, this whole process will be done for each time step to select the inputs subset.

After the selection of input variables, LS-SVMs are used as the nonlinear model, and the prediction model is:

$$\begin{aligned} \{\hat{y}(t+1), \hat{y}(t+2), \cdots, \hat{y}(t+h)\} &= F(y(t), y(t-1), \cdots, \\ &\quad y(t-p+1)), \end{aligned} \tag{5.1}$$

where F is the multi-output prediction model $F = \{f_1, f_2, \cdots, f_h\}$. As has been introduced in Section 2.1.1, the direct forecast uses different models f_h for different time steps; while the recursive forecast uses the same model f_1 for all the time steps. In all of the following experiments, we set the maximum time step $h = 15$.

5.1.1 Santa Fe Laser data set

Here, to apply the prediction model in Eq. 5.1, the maximum regressor size is set as $p = 15$. This choice is made according to previous experience on this time series [48]. First, MI is used to select the best inputs, l-NN and LOO methods are used to select the k value for the k-NN based MI estimateor. The resulting LOO errors with $k = 2, \cdots, 10$ for 15 time steps can be found in Appendix B.1.

Then, the first k values giving minimum LOO are selected for the 15 time steps as shown in Table 5.1:

The corresponding selected inputs for direct forecast are shown in Appendix B.1, and the recursive forecast only needs the one for the first time step.

Time step	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
k	2	5	3	2	2	2	2	2	2	10	3	4	5	5	7

Table 5.1: Selected k of MI estimator for Santa Fe Laser data set

For example, in Appendix B.1, the column with number 4 means that, $\hat{y}(t+4) = f_4(y(t), y(t-1), y(t-2), y(t-3), y(t-4), y(t-5))$.

Then, LS-SVMs are used to make the prediction. 10-fold cross-validation procedure for model selection purposes has been applied. The errors for the 10-fold cross-validation procedure of every pairs of γ and σ are listed. Then the area around the minima is zoomed and searched until the hyperparameters are found. For recursive prediction, only one function is used, so one pair of γ and σ is needed, which is $(1.25 \times 10^7, 620)$. For direct prediction, 15 pairs of function parameters are required. The selected ones are shown in Table 5.2.

Time step	1	2	3	4	5
γ	1.25×10^7	2.56×10^4	1.6	19	6.5
σ	620	65	545	39	40
Time step	6	7	8	9	10
γ	2.05×10^4	2.7	1.55	2×10^6	52
σ	48	70	40.2	37.9	13.3
Time step	11	12	13	14	15
γ	3.7	2.6×10^4	1.8×10^6	120	1.6×10^6
σ	28.2	27	25.6	27.6	27.9

Table 5.2: Slected parameters for LS-SVMs for Santa Fe Laser data

The mean square error (MSE) on the test set of data is used to compared the results. It is defined as:

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(\hat{y}(t+h) - y(t+h))^2, \tag{5.2}$$

where N is the number of data points, $\hat{y}(t+h)$ is the prediction result and $y(t+h)$ is the real value. The resulting MSE on the test set are listed in Table 5.3.

As illustration, the MSE values on the test set are presented also in Figure 5.2.

From the MSE values, it can be found that as time step increases, the performances of the direct predictions are better than that of the recursive ones.

Time step	1	2	3	4	5
Direct	19.668	116.96	155.35	128.58	175.47
Recursive ($\times10^4$)	0.00197	0.01295	3.1	47	51
Time step	6	7	8	9	10
Direct	200.25	217.92	189.74	141.72	286.25
Recursive ($\times10^6$)	0.59	1.7	1.3	1.8	1.6
Time step	11	12	13	14	15
Direct	263.04	252.94	258.7	245.1	247.39
Recursive ($\times10^6$)	1.4	1.7	1.6	1.9	1.8

Table 5.3: MSE values of direct and recursive predictions on test set of Santa Fe Laser data

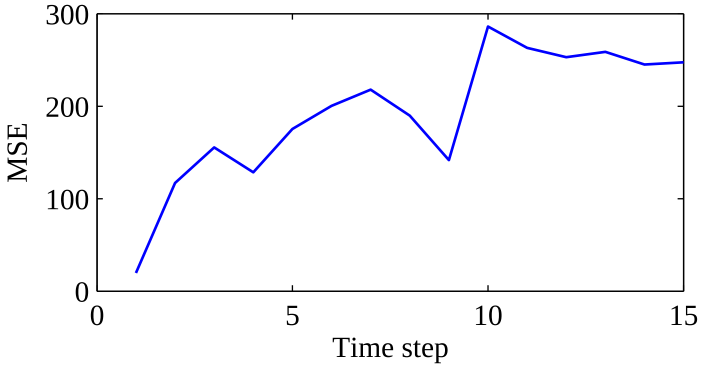

Figure 5.2: MSE values for 15 time steps of direct prediction on test set of Santa Fe Laser data

To illustrate the prediction results from direct forecast method, the predicted values of it are plotted against the real values in Appendix A.1.2. In the figure, the more the points are concentrated around a line, the better the predictions are. It can be seen that when the time step increases, the distribution of the points diverts from a line, because the prediction becomes more difficult.

Three different parts of the direct predictions on the test set of data are shown in Figure 5.3, in the order of the difficulty of prediction increasing. In each part of the figure, 15 time steps prediction results are plotted.

From Figure 5.3, it can be seen that the predicted values and the real values are very close. Hence, the models built using the corresponding inputs led to good prediction performance.

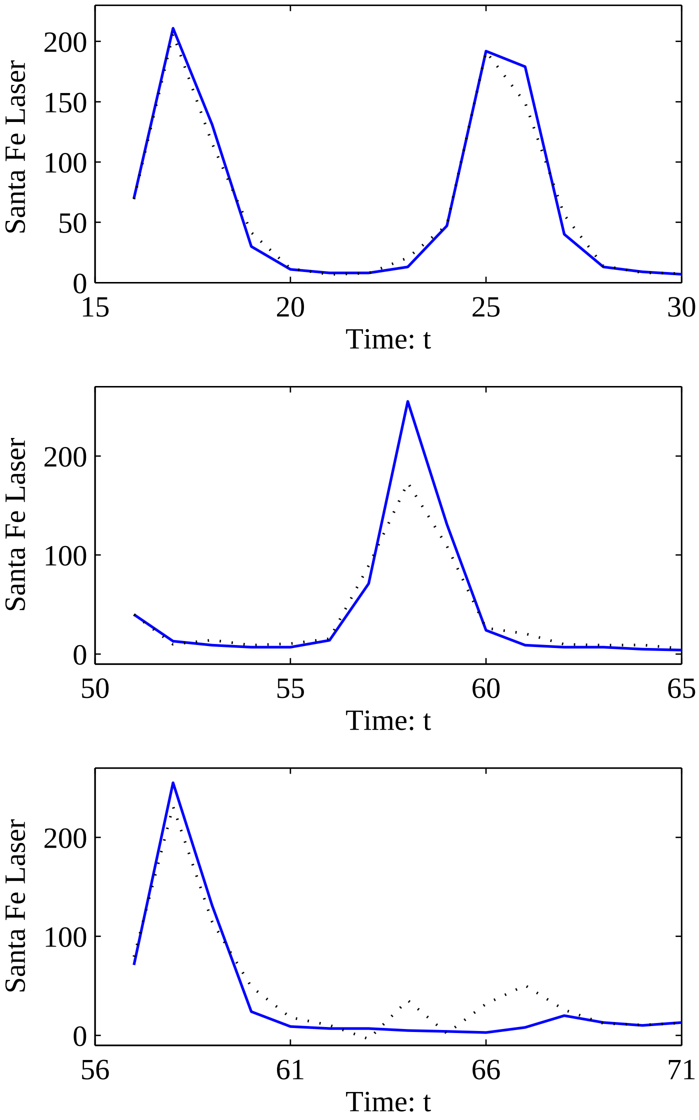

Figure 5.3: Three parts of 15 time steps predictions of test set of Santa Fe Laser data are represented in dotted line and the real values are represented in solid line

5.1.2 Poland Electricity data set

For this data set, the regressor size is set to 15. First, the k values in MI estimator are selected by the minimum LOO errors which is shown in Appendix A.2.1 for 15 time steps, the selected results are shown in Table 5.4:

Time step	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
k	6	2	2	2	4	4	6	2	2	2	5	5	3	4	3

Table 5.4: Selected k of MI estimator for Poland Electricity data set

The corresponding selected inputs for direct forecast can be found in Appendix B.2.

Then LS-SVMs are used to make the prediction. Also 15 pairs of parameter γ and σ are selected by 10-fold cross-validation procedure. The MSE values on the test set are listed in Table 5.5.

Time step	1	2	3	4	5
Direct ($\times 10^{-3}$)	2.16	2.67	2.80	2.88	3.03
Recursive ($\times 10^{-3}$)	2.16	2.86	3.73	6.80	1097.1
Time step	6	7	8	9	10
Direct ($\times 10^{-3}$)	3.04	3.16	3.71	4.45	4.98
Recursive	0.71	0.72	0.71	0.71	0.71
Time step	11	12	13	14	15
Direct ($\times 10^{-3}$)	4.93	5.06	4.97	4.63	5.11
Recursive	0.72	0.71	0.71	4.02	2.61

Table 5.5: MSE values of direct and recursive prediction on test set of Poland Electricity data set

As illustration, the MSE values on the test set are presented also in Figure 5.4.

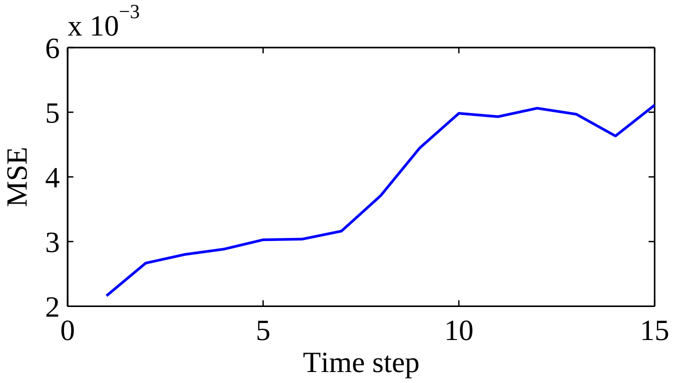

Figure 5.4: MSE values for 15 time steps of direct prediction on test set of Poland Electricity data

From the MSE values, it can be found that as time step increases, the performances of the direct prediction are better than the recursive prediction.

To illustrate the prediction results from direct forecast method, the prediction of it are plotted against the real values in Appendix A.2.2. It can be seen that when the time step is large, the distribution of the points diverts a lot.

Figure 5.5 shows 15 time steps direct predictions on the test set of Poland Electricity data set. It can be seen that the predicted values and the real values are very close in this case.

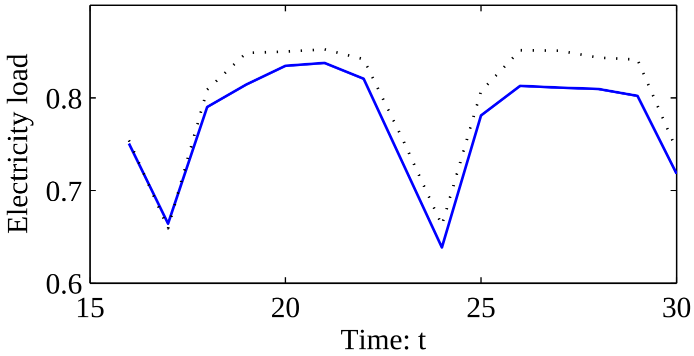

Figure 5.5: The 15 time steps prediction on the test set of Poland Electricity data set is represented in dotted line and the true value is represented in solid line

5.1.3 Darwin sea level data set

The regressor size is set to be 30 here. First, the k values in MI estimator are selected by the minimum LOO error which is shown in Appendix A.3.1 for 15 time steps. The selected k values are shown in Table 5.6.

Time step	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
k	2	5	3	3	7	5	5	4	4	4	3	3	6	9	3

Table 5.6: Selected k of MI estimator for Darwin sea level data set

The corresponding selected inputs for direct forecast can be found in Appendix B.3.

LS-SVMs are used to make the prediction. 15 pairs of parameter γ and σ are selected by 10-fold cross-validation procedure. The MSE values on the test set are listed in Table 5.7.

Time step	1	2	3	4	5
Direct	0.95	1.11	1.18	1.41	1.43
Recursive	0.95	1.14	1.30	1.42	1.54
Time step	6	7	8	9	10
Direct	1.46	1.46	1.50	1.55	1.58
Recursive	1.60	1.60	1.67	1.74	1.77
Time step	11	12	13	14	15
Direct	1.55	1.55	1.68	1.72	1.71
Recursive	1.78	1.78	1.79	1.82	1.84

Table 5.7: MSE values of direct and recursive prediction on the test set of Darwin sea level data

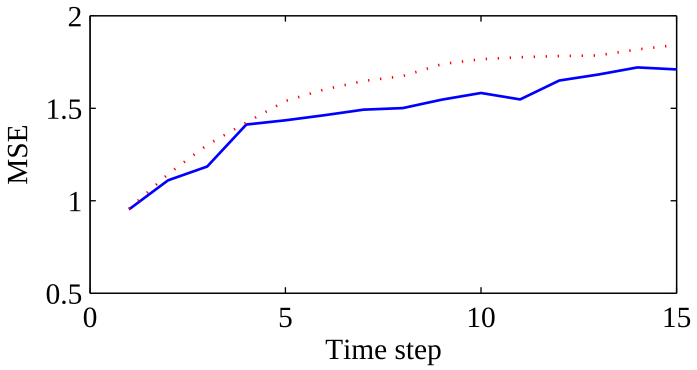

Figure 5.6: MSE values for 15 time steps of direct and recursive forecast on test set of Darwin sea level data: the solid line is from direct forecast, and dotted line corresponds to recursive forecast

As illustration, the MSE values on the test set are presented also in Figure 5.6. From the MSE values, it can be found that with this data set, as time step increases, the performances of the direct predictions are better than the recursive ones.

To illustrate the prediction results from direct forecast method, the predicted values are plotted against the real values in Appendix A.3.2.

15 time steps predictions from direct forecast method is given in Figure 5.7. Figure 5.7 shows that the predictions by the direct forecast method is good.

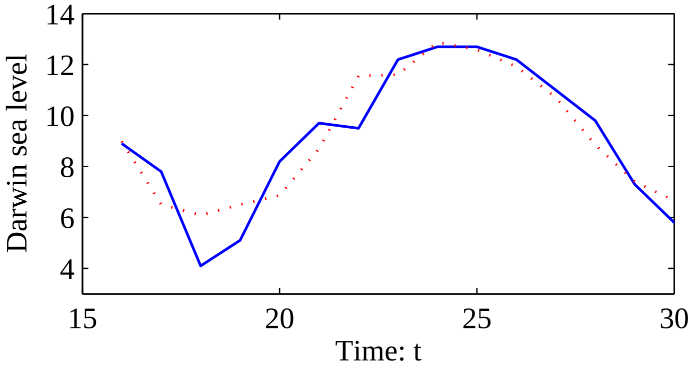

Figure 5.7: The 15 time steps prediction on the test set of Darwin sea level data set is represented in dotted line and the true value is represented in solid line

5.1.4 Conclusion

In this test, we compared two long-term time series prediction strategies: direct and recursive forecasts.

MI is used to perform the input selection for both strategies. Though for each time step, four input selection processes are performed and one parameter needs to be selected using l-NN and LOO methods, it is still fast compared to other input selection methods. The results show that this MI based method can provide a good input selection, and it has been illustrated with the experiments that the l-NN approximator and LOO method can be used to tune the main parameter of the MI estimator.

Comparing both long-term prediction strategies, direct long-term prediction is superior to recurrent prediction for all the time steps. But the former strategy requires multiple models. Nevertheless, due to the simplicity of MI based input selection method, direct prediction strategy can still be used in practice. Thus, direct prediction and MI based input selection can be considered as an efficient approach for a long-term time series prediction. The main advantage of this proposed approach is that it combines fast input selection with accurate but com- putationally demanding non-linear prediction.

5.2 Comparison of three input selection methods for long-term time series prediction

As discussed in Section 2.2.1, the filter input selection method selects a set of features by optimizing a criterion over different combinations of inputs.

The criterion computes the dependences between each combination of input variables and the corresponding output. Various alternatives of these criteria exist. This experiment is to compare three different criteria: mutual information, nonparametric noise estimator and l-nearest neighbours.

Nonparametric Noise Estimator (NNE) is a technique for estimating the variance of the noise, or the mean square error (MSE), that can be achieved without overfitting [49]. NNE is a generalization of the approach proposed in [49], which is basically based on the fact that the conditional expectation 5.3 approaches variance of the noise when the distance between the data points tends to zero.

$$\epsilon \left\langle 1/2(y' - y)^2 | \left| x' - x \right| \delta \right\rangle, \text{as}(\delta \to 0). \tag{5.3}$$

The best set of inputs is the one that minimizes the result of the Γ value calculated by NNE. In NNE, there is also a parameter q which represents the number of neighbours used in the calculation, and it can be also tuned for different problems using l- NN and LOO methods as described before for the MI based estimation, the q value will be tested from 2 to 15 here.

l-NN has been explained in Section 3.2.3, it can be also used in the input selection. The best inputs subset is the one that minimizes the error after l-NN approximation.

These three input selection approaches are used to select the best input variables (from a set of possible variables). The input selection processes are the same as introduced in Section 5.1, the only difference is that when using NNE and l-NN methods, the inputs are selected by minimizing Γ value and l-NN approximationerror, respectively, instead of maximizing MI value.

The experiments are performed on three different long-term time series in order to show the level of efficiency of these three methods for the problem of input selection. Again, the maximum time step is set to be 15. As it has been shown that the direct forecast approach is better than the recursive one in long-term time series prediction problem in the previous experiment, we will use only the direct forecast method in this test. Also, LS-SVMs are used to compare the performances in order to avoid the local minima problem.

5.2.1 Santa Fe Laser data set

Here, the regressor size is set to be 15. The selected k values in MI estimation have been shown in Table 5.1, the selected q values for NNE are listed in Table 5.8.

The corresponding selected inputs of the three methods are shown in Appendix B.1. It can be seen that the selected inputs from different methods are different, and the nearest regressors are selected by most of the methods.

Time step	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
q	12	10	11	5	12	14	5	11	12	13	9	13	5	6	6

Table 5.8: Selected q of NNE method for Santa Fe Laser data set

Time	Methods					
step	MI		NNE		l-NN	
h	MSE	MAE	MSE	MAE	MSE	MAE
1	19.67	1.59	33.85	1.90	29.52	2.15
2	116.96	2.82	130.59	3.91	67.82	3.07
3	155.35	3.84	131.20	3.60	211.74	3.07
4	128.58	3.18	148.33	3.33	203.41	3.99
5	175.47	3.89	154.23	3.75	102.84	2.89
6	200.25	3.58	154.40	4.00	146.31	3.72
7	217.92	5.64	139.81	3.91	144.66	3.66
8	189.74	5.13	215.41	5.25	134.51	3.31
9	141.72	4.38	139.42	4.05	145.72	4.32
10	286.25	7.19	213.78	6.20	230.43	6.96
11	263.04	5.94	252.64	5.36	338.55	8.36
12	252.94	5.50	249.22	5.90	234.09	4.88
13	258.70	5.70	285.41	5.85	245.64	5.12
14	245.10	5.23	305.69	6.03	247.48	5.54
15	247.39	5.52	305.05	6.31	251.50	5.23

Table 5.9: MSE and MAE of the three input selection methods for 15 time steps on test set of Santa Fe Laser data

Then, LS-SVMs are used for comparing the regressor selection performances. 10-fold cross-validation for model selection purposes has been applied. MSE and mean absolute error (MAE) on the test set of data are used to compare the performances. MAE is defined as:

$$\text{MAE} \quad = \quad \frac{1}{N}\sum_{i=1}^{N}|\hat{y}(t+h) - y(t+h)|\,, \tag{5.4}$$

where N is the number of data points, $\hat{y}(t+h)$ is the prediction result and $y(t+h)$ is the real value. The MSE and MAE values on the test set for the three methods are presented in Table 5.9.

As illustration, the MSE on the test set are also plotted in Figure 5.8

It can be seen from the error that the error increases with the time step, and the performances of the three input selection methods are quite similar.

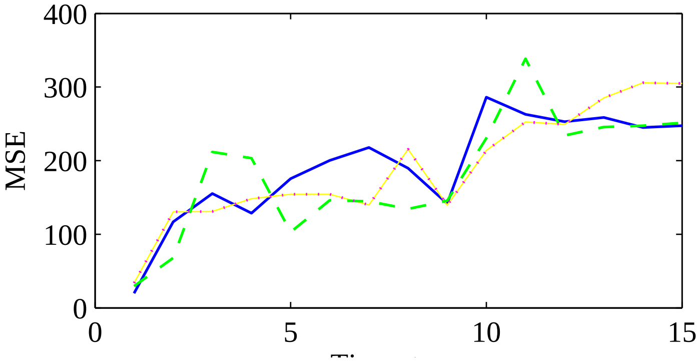

Figure 5.8: MSE values of 15 time steps on the test set of Santa Fe Laser data of three input selection methods: solid line is from MI, dotted line for NNE, dashed line is for l-NN

5.2.2 Poland Electricity data set

Here, the regressor size is also set to be 15. The selected k values in MI estimation have been shown in Table 5.4, and the selected q values for NNE are shown in Table 5.10.

Time step	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
q	7	13	14	12	10	14	8	11	7	15	11	15	13	9	7

Table 5.10: Selected q of NNE method for Poland Electricity data set

The corresponding selected inputs of the three methods are presented in Appendix B.2. It can be seen from Appendix B.2 that the selection results are quite similar for the three methods.

Then, LS-SVMs are used as the prediction models. 10-fold cross-validation approach has been applied for model selection purpose. The parameters γ and σ are found and the MSE and MAE on the test set for the three methods are presented in Table 5.11.

As illustration, the MSE values on the test set are also plotted in Figure 5.9

It can be seen from the error that the error increases with the time step, and the performances of these three input selection methods are also quite similar for this data set.

Time	Methods					
step	MI		NNE		$l-$NN	
h	MSE	MAE	MSE	MAE	MSE	MAE
1 ($\times 10^{-3}$)	2.16	26.19	1.59	23.42	1.73	24.01
2 ($\times 10^{-3}$)	2.16	30.98	2.31	30.86	2.28	30.61
3 ($\times 10^{-3}$)	2.80	32.44	2.88	32.76	2.62	31.54
4 ($\times 10^{-3}$)	2.88	33.82	2.78	33.03	2.89	34.47
5 ($\times 10^{-3}$)	3.03	35.04	2.91	33.96	3.01	35.28
6 ($\times 10^{-3}$)	3.04	35.05	3.03	35.46	3.00	34.80
7 ($\times 10^{-3}$)	3.16	35.51	2.87	34.66	3.34	36.95
8 ($\times 10^{-3}$)	3.71	40.73	3.46	40.11	3.76	40.33
9 ($\times 10^{-3}$)	4.45	43.96	4.37	44.71	4.31	44.81
10 ($\times 10^{-3}$)	4.98	47.15	4.84	49.06	4.55	47.09
11 ($\times 10^{-3}$)	4.93	47.60	4.69	47.39	4.69	48.76
12 ($\times 10^{-3}$)	5.06	47.62	4.54	46.14	4.71	47.89
13 ($\times 10^{-3}$)	4.97	46.51	5.31	47.81	4.73	48.15
14 ($\times 10^{-3}$)	4.63	45.22	4.60	44.99	4.60	44.93
15 ($\times 10^{-3}$)	5.11	50.58	5.90	52.46	5.40	52.56

Table 5.11: MSE and MAE of the three input selection methods for 15 time steps on test set of Poland Electricity data set

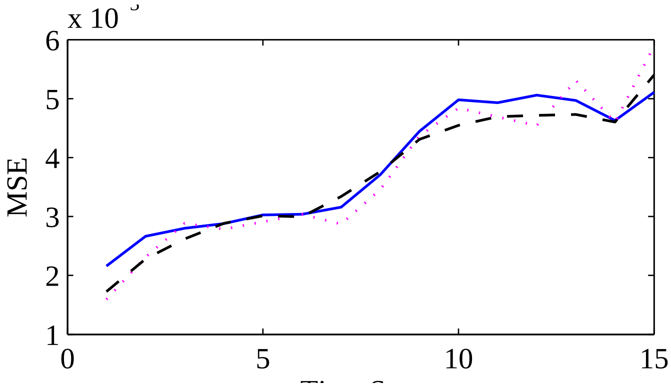

Figure 5.9: MSE values of 15 time steps on the test set of Poland Electricity data of three input selection methods: solid line is from MI, dotted line for NNE, dashed line is for l-NN

5.2.3 Darwin sea level data set

Here, the regressor size is set to be 30. The selected k values in MI estimation have been shown in Table 5.6, and the selected q values for NNE are shown in Table 5.12.

Time step	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
q	11	14	14	8	15	5	2	10	11	12	2	11	8	3	10

Table 5.12: Selected q of NNE method for Darwin sea level data set

The corresponding selected inputs of the three methods are shown in Appendix B.3. It can be seen from Appendix B.3 that the selection results are similar for the three methods.

Then, LS-SVMs are used for comparing the regressor selection performances. 10-fold cross-validation is used for model selection. The parameters γ and σ are found and the MSE and MAE values on the test set for the three methods are listed in Table 5.13.

The MSE on the test set are also plotted in Figure 5.10

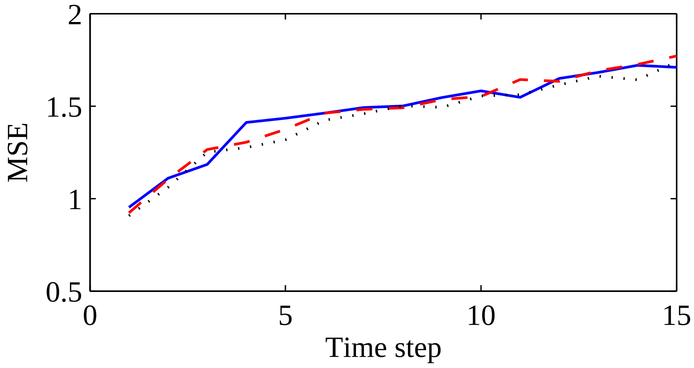

Figure 5.10: MSE values of 15 time steps on the test set of Darwin sea level data of three input selection methods: solid line is from MI, dotted line for NNE, dashed line is for l-NN

It can be seen that the error increases with the time step, and the results of the three methods are quite similar.

5.2.4 Computational time

The computational times for selecting the 15 time steps subsets of inputs for each data set using the three different input seleciton methods are shown in Table 5.14.

Comparing the times these three methods use to selecting 15 time steps inputs, it can be seen that with the same number of data points, the com-

Time	Methods					
step	MI		NNE		$l-$NN	
h	MSE	MAE	MSE	MAE	MSE	MAE
1	0.95	0.77	0.91	0.74	0.92	0.74
2	1.11	0.89	1.06	0.82	1.11	0.83
3	1.18	0.98	1.25	0.90	1.26	0.91
4	1.41	0.03	1.28	0.92	1.31	0.93
5	1.43	0.98	1.32	0.93	1.37	0.95
6	1.46	0.98	1.46	0.95	1.46	0.97
7	1.46	0.98	1.46	0.95	1.48	0.97
8	1.50	0.98	1.50	0.98	1.49	0.97
9	1.55	0.99	1.49	0.96	1.54	0.98
10	1.58	1.01	1.56	0.99	1.55	0.98
11	1.55	0.99	1.56	1.00	1.64	1.01
12	1.55	1.02	1.62	1.00	1.63	1.01
13	1.68	1.03	1.66	1.02	1.69	1.02
14	1.72	1.03	1.64	1.01	1.73	1.04
15	1.71	1.04	1.74	1.04	1.77	1.05

Table 5.13: MSE and MAE of the three input selection methods for 15 time steps on test set of Darwin sea level data set.

	Santa Fe Laser	Poland Electricity	Darwin sea level
MI	60.86	71.40	300.25
NNE	32.67	50.77	202.40
l-NN	1.30	2.01	7.16

Table 5.14: Computational time (in hours) of input selection process for three data sets using different input selecion methods

putational time increases with the regressor size. For each data set, l-NN method is the fastest one, NNE method requires around 25 times the computaional time of l-NN method, and MI method is the slowest one, which consumes about 40 times the computational time of l-NN method.

5.2.5 Conclusion

In this section, MI based input selection method is compared with other two approaches: one is based on NNE, and the other based on l-NN technology. Based on the experiments, the selected inputs from the three different methods are different, and the prediction results on the test set show that the performances of these three methods are quite similar. Hence, the one consuming least computational time is preferable. Based on the computational time comparison, the l-NN method is proposed as an effective and efficient input selection method in the case of this experiment.

Chapter 6

Conclusion

Input selection is an essential pre-processing stage in problems such as machine learning, especially when the number of observations is relatively small compared to the number of inputs. The aim of the input selection method is to reduce as much as possible the inputs in order to improve the quality of the model built, and to improve the interpretability of the selected set of input variables.

In this thesis, we propose a new mutual information based scheme to select the salient inputs that are relevant to the corresponding output and not redundant to the selected inputs. The MI has the property to be model-independent and able to measure nonlinear dependencies at the same time.

To apply this MI based input selection strategy, first, an accurate and efficient MI estimation method is needed. In this thesis, we have done a survey of the most classic estimators and introduced two new estimation techniques, which solve the problems appear in the high-dimensional data space case. Moreover, for one of these two new approaches: the k-NN based MI estimator, we developed a strategy to tune the main parameter of it for different data sets and special problems. Then, a thorough comparison of the estimation methods were performed and we decided to use the k-NN based estimator with respect to its superiority in the performance of comparative results. The main advantage of this method consists in its ability to estimate the MI between two variables in high-dimensional space.

After solving the problem of estimating MI, one must then consider how to perform the actual input selection process. The aim is to select the best set of inputs which is the one that maximizes the MI. Several ways of selecting the best inputs subset are investigated and compared in this thesis. The optimal technique is to do the exhaustive search, but the computational time of it will increase dramatically with the regressor size, thus, it is not possible to test all the combinations of inputs when regressor size is large. In this case, to ensure the performance, we propose to perform all of the other input selection strategies except exhaustive search, and select the best one which gives maximum MI value. This searching strategy is much faster than the exhaustive one while also accurate enough.

Finally, we applied our method into the long-term time series prediction problem. Two experimental results were demonstrated based on three different time series. The experiments verified the potential power of the proposed MI based input selection method. It has also been illustrated with the experiments that the strategy introduced in this thesis for choosing the main parameter of the k-NN based MI estimator is applicable and efficient. At the same time, it has been presented that the simplicity of the MI based input selection method can help to improve the performance of long-term time series prediction by applying the direct forecast. In addition, comparing with other two input selection approaches: NNE method and l-NN method, it has been shown that MI based method can give similar performance as the other two methods, and the l-NN method is proposed in the experiment as it consumes the least computational time.

In future, it would be interesting to see how this MI based input selection method extends to other types of data sets and how to speed up this method.

Bibliography

[1] B. Efron and R. Tibshirani, *An introduction to the bootstrap.* Chapman and Hall, 1993.

[2] M. Stone, "An asymptotic equivalence of choice of model by cross-validation and akaike's criterion," *J. Royal. Statist. Soc.*, vol. 39, pp. 44–7, 1977.

[3] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection," *Proc. of the* 14^{th} *Int. Joint Conf. on A.I*, vol. 2, pp. 1137–1145, 1995.

[4] A. Sorjamaa, J. Hao, and A. Lendasse, "Mutual information and k-nearest neighbors approximator for time series prediction," *Lecture Notes in Computer Science*, vol. 3697, pp. 553–558, 2005.

[5] Y. Ji, J. Hao, N. Reyhani, and A. Lendasse, "Direct and recursive prediction of time series using mutual information selection," in *International work-conference on artificial neural networks, (IWANN 2005)*, pp. 1010–1017, 2005.

[6] N. Reyhani, J. Hao, Y. Ji, and A. Lendasse, "Mutual information and gamma test for input selection," in *the* 15^{th} *International conference on machine learning*, pp. 515–521, 1998. Available from http://www.cis.hut.fi/projects/tsp/Publications/Publication24.pdf.

[7] J. Hamilton, "Analysis of time series subject to changes in regime," *Econometrics*, vol. 45, pp. 39–70, 1990.

[8] A. Lendasse, J. Lee, V. Wertz, and M. Verleysen, "Forecasting electricity consumption using nonlinear projection and self-organizing maps," *Neurocomputing*, vol. 48, pp. 299–311, 2002.

[9] C. D. Cin, L. Moens, P. Dierickx, G. Bastin, and Y. Zech, "An integrated approach for real-time flood-map forecasting on the belgian meuse river," *Natural Hazards.*

[10] M. Sulkava, J. Tikka, and J. Hollmén, “Sparse regression for analyzing the development of foliar nutrient concentrations in coniferous trees,” in *international workshop on environmental applications of machine learning, (EAML 2004)*, pp. 57–58, 2004.

[11] J. Fan and Q. Yao, *Nonlinear Time Series. Nonparametric and Parametric Methods.* Springer (Springer Series in Statistics), 2003. ISBN: 0387951709.

[12] Available from http://www-psych.stanford.edu/∼andreas/Time-Series/SantaFe.html.

[13] G. Box and G. Jenkins, *Time series analysis: Forecasting and control.* Cambridge University Press, Cambridge, 1976.

[14] L. Ljung, *System identification theory for User.* Prentice-Hall, Englewood CliPs, NJ, 1987.

[15] S. Gutjahr, M. Riedmiller, and J. Klingemann, “Daily prediction of the foreign ex-change rate between the us dollar and the german mark using neural networks,” *Proc. of SPICES*, pp. 492–498, 1997.

[16] D. Nicholls, A. Pagan, Hannan, P. Krishnaiah, and M. Rao, *Varying coefficient regression, Handbook of Statistics.* Amsterdam, North-Holland, 1985.

[17] R. Duda and P. Hart, *Pattern classification and scene analysis.* John Wiley and Sons, 1973.

[18] C. Watkins, “Models of delayed reinforcement learning,” Master’s thesis, Cambridge University, 1989. Ph.D. thesis.

[19] A. Jain, M. Murty, and P. Flynn, “Data clustering: A review,” *ACM Computing Surveys*, vol. 31, pp. 264–323, 1999.

[20] K. Müller, A. Smola, G. Rätsch, B.Schölkopf, J. Kohlmorgen, and V. Vapnik, “Predicting time series with support vector machines,” *Artificial neural networks, ICANN, Springer Lecture Notes in Computer Science*, vol. 1327, pp. 999–1004, 1997.

[21] S. Haykin, *Neural Networks: A Comprehensive Foundation.* Prentice Hall, NJ, 2 edition, 1999.

[22] T. Trafalis and B. Santosa, “Predicting monthly flour prices through neural networks,” *Intelligent Engineering Systems Through Artificial Neural Networks*, vol. 11, pp. 745–750, 2001.

[23] J. Suykens, K. V. Gestel, J. Brabanter, B. Moor, and J. Vandewalle, “Weighted least squares support vector machines: Robustness and sparse approximation,” vol. 48, pp. 85–105, 2002.

[24] C. Saunders, A. Gammerman, and V. Vovk, “Ridge regression learning algorithm in dual variables,” in *European Symposium on Artificial Neural Networks, (ESANN 2005), 27-29*, pp. 503–504, 2005.

[25] J. Suykens, T. Gestel, J. Brabanter, B. Moor, and J. Vandewalle, *Least Squares Support Vector Machines.* World Scientific publishing Co. Pte. Ltd, 2002.

[26] F. Rossi, A. Lendasse, D. François, V. Wertz, and M. Verleysen, “Mutual information for the selection of relevant variables in spectrometric nonlinear modeling,” *Chemometrics and Intelligent Laboratory Systems*, 2005.

[27] J. Han and M. Kamber, *Data mining: Concepts and Techniques.* Morgan Kaufmann Publishers, San Francisco, 2001.

[28] M. Verleysen and D. Francois, “The curse of dimensionality in data mining and time series prediction,” in *International work-conference on artificial neural networks, (IWANN 2005)*, pp. 758–770, 2005.

[29] M. Ben-Bassat, “Pattern recognition and reduction of dimensionality,” *Handbook of statistics II*, pp. 773–91, 1982.

[30] R. Battiti, “Using mutual information for selecting features in supervised neural net learning,” *IEEE Trans. Neural Network*, vol. 5, pp. 537–550, 1994.

[31] T. Cover and J. Thomas, *Elements of information theory.* Wiley, New York, 1991.

[32] R. Steuer, J. Kurths, C. Daub, J. Weise, and J. Selbig, “The mutual information: Detecting and evaluating dependencies between variables,” *Bioinformatics*, vol. 18, pp. 231–240, 2002.

[33] P. Grassberger, “Finite sample corrections to entropy and dimension estimates,” *Phys. Lett*, vol. 128, pp. 369–373, 1988.

[34] I. Grosse, *Estimating entropies from finite samples.* Dynamik-Evolution-Strukturen, Dr. Koster, Freund, JanA, 1996.

[35] M. Roulston, “Estimating the errors on measured entropy and mutual information,” *Phys.*, vol. 125, pp. 285–294, 1999.

[36] Y. Moon, B. Rajagopalan, and U. Lall, "Estimation of mutual information using kernel density estimators," *Phys. Rev*, vol. 52, pp. 2318–2321, 1995.

[37] N. Kwak and C.-H. Choi, "Input feature selection by mutual information based on parzen window," *IEEE Trans. PAMI*, vol. 24, pp. 1667–1671, 2002.

[38] B. Silverman, *Density estimation for statistics and data analysis.* Chapman and Hall, London, 1986.

[39] A. Kraskov, H. Stögbauer, and P. Grassberger, "Estimating mutual information," *Phys. Rev.*, vol. 69, p. 066138, 2004.

[40] J. Steil, "Markoffsche felder und wechselseitige information in der bildverarbeitung," Master's thesis, University of Bielefeld, 1993. Master's thesis.

[41] Available from http://tinyurl.com/bj73w.

[42] C. Bishop, *Neural Networks for Pattern Recognition.* Oxford University Press, 1995.

[43] G. Darbellay and I. Vajda, "Estimation of the information by adaptive partitioning of the observation space," *IEEE Trans. Inf. theory*, vol. 45, pp. 1315–1321, 1999.

[44] N. Kwak and C.-H. Choi, "Input feature selection for classification problems," *IEEE Trans. Neural Network*, vol. 13, pp. 143–159, 2002.

[45] M. Tesmer and P. Estévez, "AMIFS: Adaptive feature selection by using mutual information," in *proceedings of the International Joint Conference on Neural Networks and IEEE International Conference on Fuzzy Systems (IJCNN 2004)*, vol. 1, pp. 303–308, July.

[46] P. Pudil, J. Novovicova, and J. Kittler, "Floating search methods in feature selection," *Pattern Recognition Letters*, vol. 15, pp. 1119–1125, 1994.

[47] Available from http://www.cis.hut.fi/projects/tsp/Download/darwin.dat.

[48] A. Weigend and N. Gershenfeld, *Times Series Prediction: Forecasting the Future and Understanding the Past.* Addison-Wesley Publishing Company, 1994.

[49] A. J. Jones, "New tools in non-linear modeling and prediction," *Computational Management Science*, vol. 1, pp. 109–149, 2004.

Appendix A

Direct and recursive prediction

A.1 Santa Fe Laser data set

A.1.1 LOO errors versus different k values

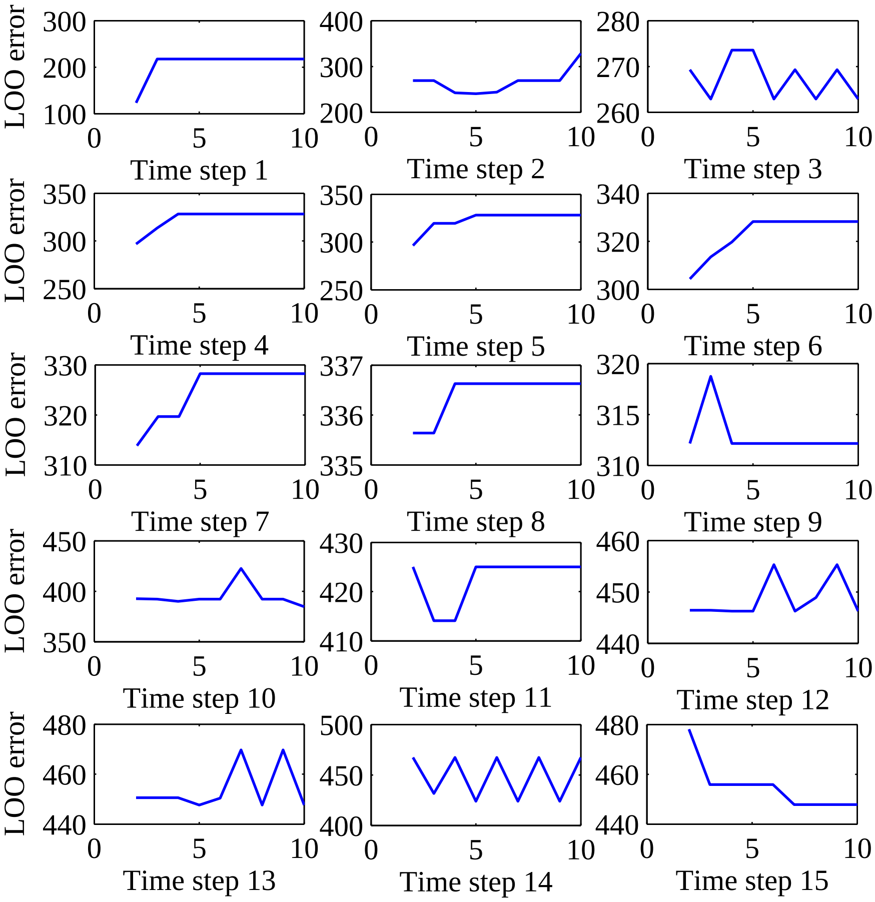

Figure A.1: Seleted k for MI estimator based on LOO error with 15 time steps on Santa Fe Laser data set

A.1.2 Predictions of direct forecast versus true values of Santa Fe Laser data set

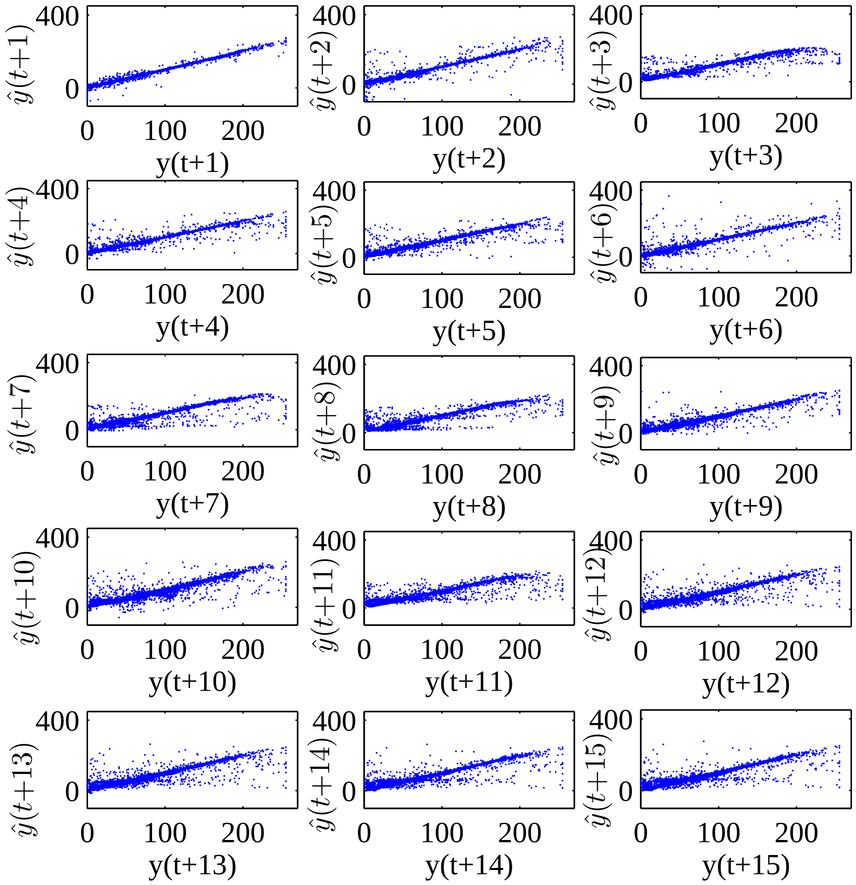

Figure A.2: Prediction results of direct forecast versus true values for each time step on test set of Santa Fe Laser data

A.2 Poland Electricity data set

A.2.1 LOO errors versus different k values

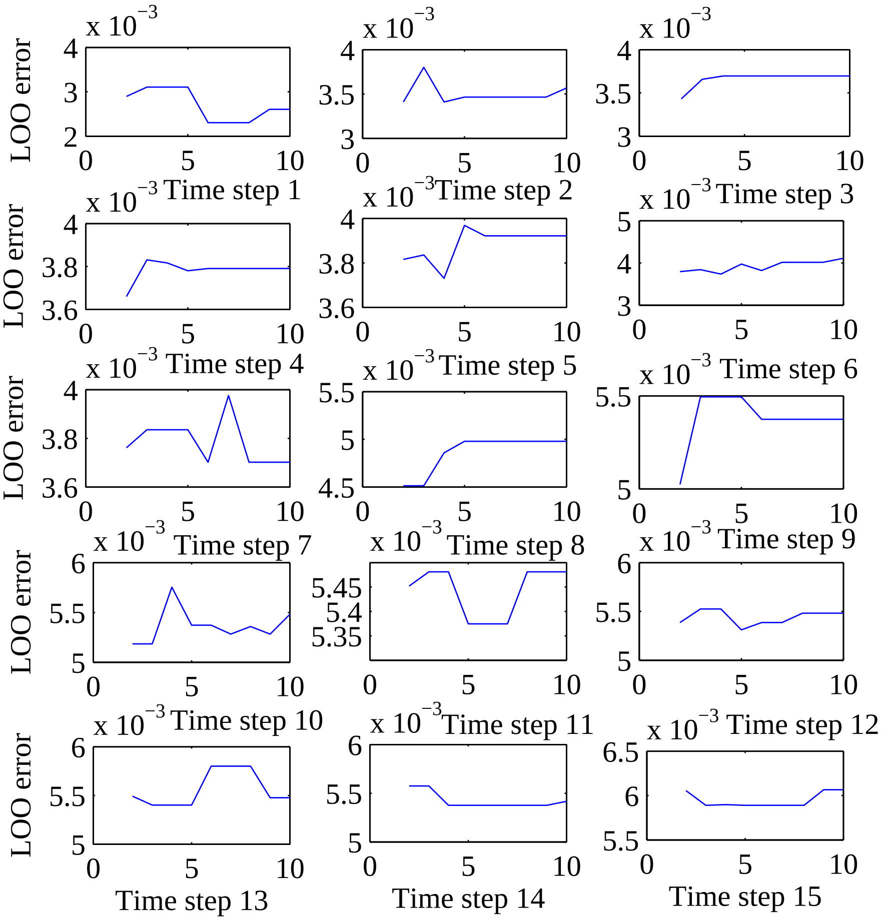

Figure A.3: Seleted k for MI estimator based on LOO error with 15 time steps on Poland Electricity data set

A.2.2 Predictions of direct forecast versus true values of Poland Electricity data set

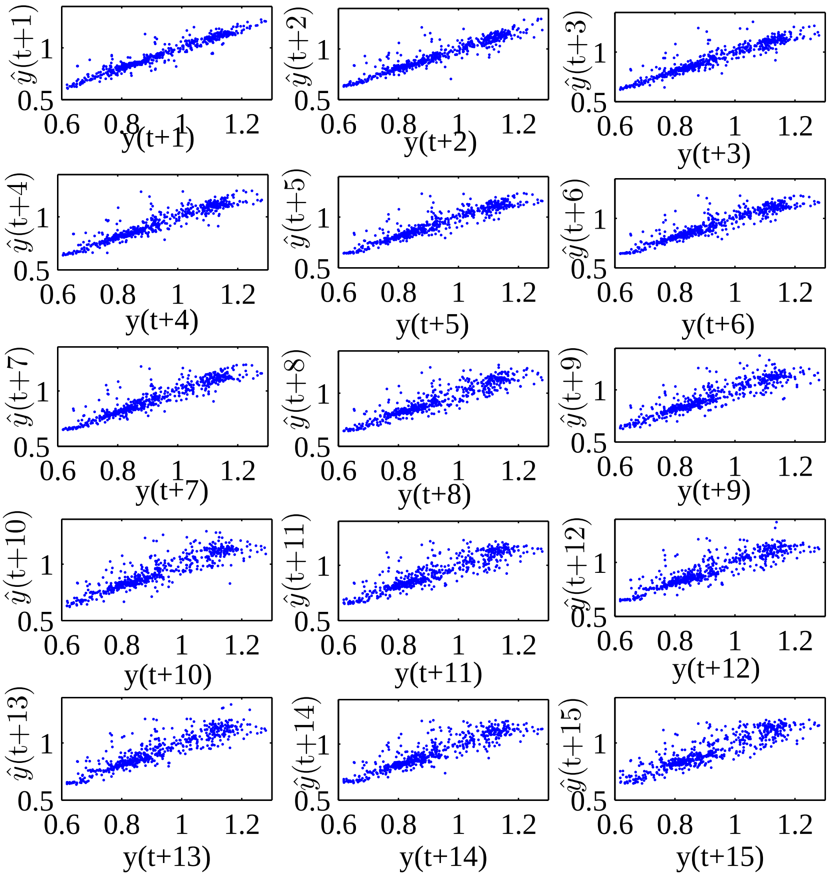

Figure A.4: Prediction results of direct forecast versus true values for each time step on test set of Poland Electricity data

A.3 Darwin sea level data set

A.3.1 LOO errors versus different k values

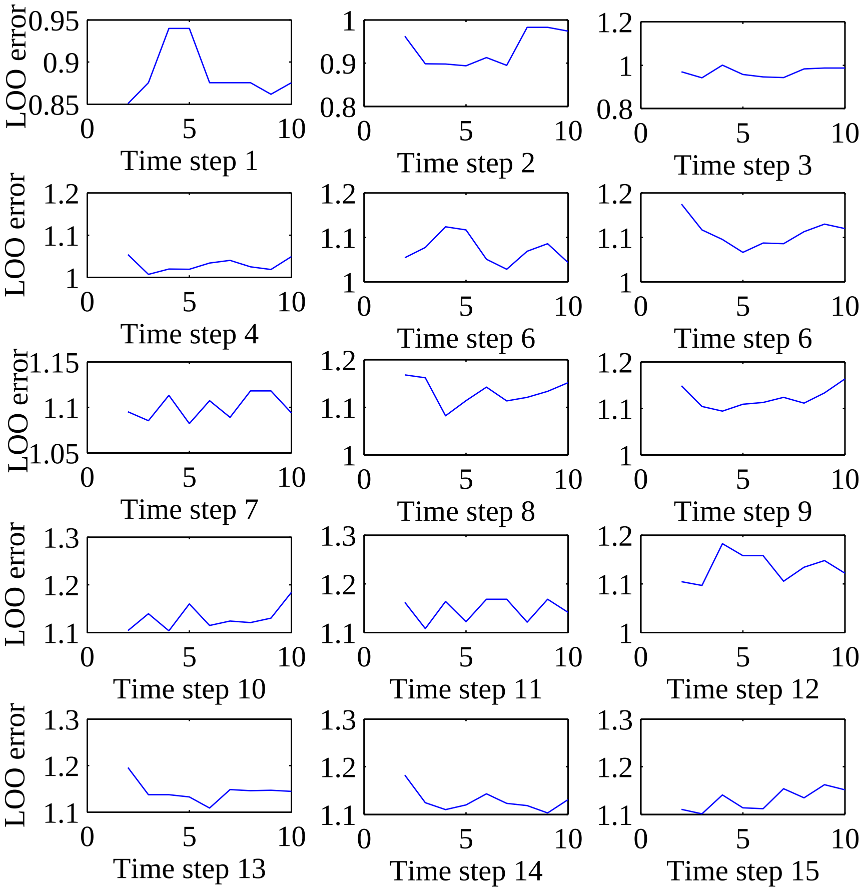

Figure A.5: Seleted k for MI estimator based on LOO error with 15 time steps on Darwin sea level data set

A.3.2 Predictions of direct forecast versus true values of Darwin Sea Level data set

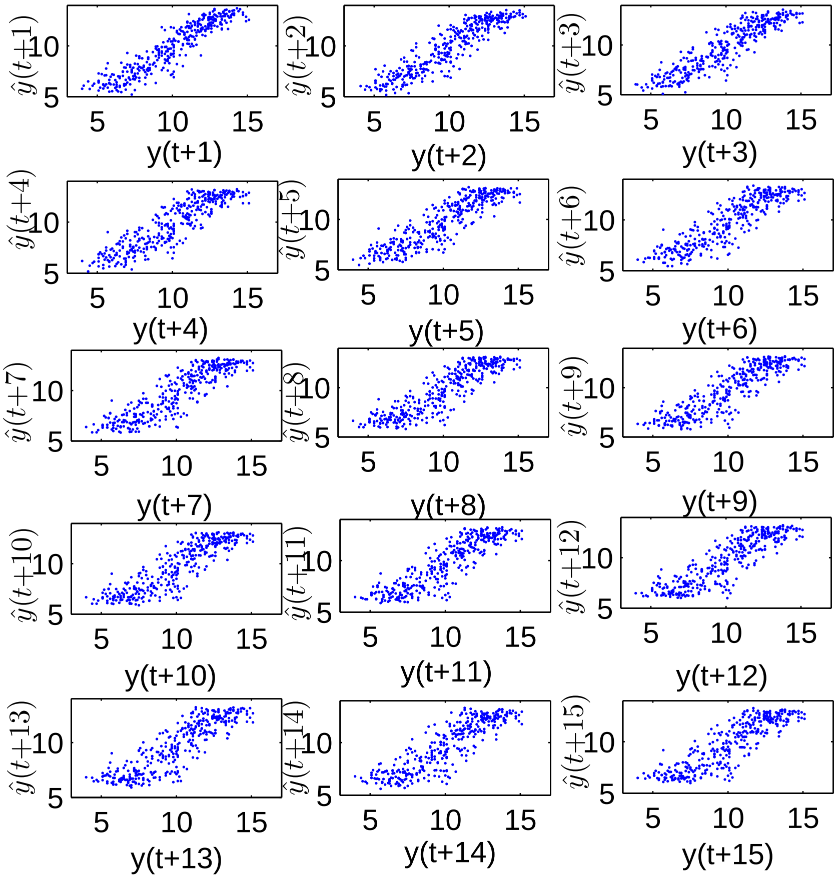

Figure A.6: Prediction results of direct forecast versus true values for each time step on test set of Darwin sea level data

Appendix B

Comparison of three input selection methods

B.1 Selected inputs for Sata Fe Laser data set.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
	○	○	○	○		○	○		○	○		○	○	○	
	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ	Δ
-1	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
	○		○	○	○	○		○		○	○	○	○		○
	Δ	Δ		Δ	Δ	Δ	Δ	Δ		Δ	Δ	Δ	Δ		Δ
-2	X	X	X	X	X	X			X	X	X	X	X	X	X
	○	○	○	○	○		○		○	○	○	○		○	
			Δ	Δ	Δ			Δ	Δ			Δ		Δ	
-3	X	X	X	X	X	X			X	X	X	X	X	X	X
		○	○	○		○					○		○		○
		Δ		Δ	Δ	Δ		Δ	Δ				Δ		Δ
-4	X	X	X	X		X			X	X	X	X	X	X	X
			○		○			○			○	○			
							Δ		Δ			Δ		Δ	
-5	X	X	X	X	X	X			X		X	X	X	X	X
				○			○	○			○			○	○
							Δ	Δ	Δ				Δ		
-6	X	X			X	X			X			X	X	X	X
			○			○		○				○	○	○	
					Δ	Δ	Δ		Δ			Δ	Δ	Δ	Δ
-7	X							X	X			X	X	X	X
						○	○	○			○			○	○
					Δ		Δ		Δ					Δ	
-8		X	X				X				X				X
					○	○	○	○		○			○	○	○
						Δ		Δ		Δ		Δ			
-9		X												X	
		○	○	○	○	○			○	○		○	○	○	
				Δ		Δ			Δ	Δ				Δ	Δ
-10											X				
		○	○	○	○				○	○	○	○	○		
		Δ	Δ	Δ			Δ						Δ	Δ	
-11															
	○	○	○	○			○		○		○				
	Δ				Δ				Δ			Δ	Δ	Δ	
-12															
	○					○		○	○		○				○
						Δ	Δ		Δ			Δ	Δ		
-13															
					○		○	○			○				○
					Δ		Δ					Δ		Δ	Δ
-14															
			○	○			○	○	○		○	○			○
				Δ										Δ	

Table B.1: Selected inputs for Santa Fe Laser data: The numbers in the first row and first column represent time steps and regressor index. Symbol X is for MI selected inputs, O represents NNE selection results, Δ is for l-NN selected results

B.2 Selected inputs for Poland Electricity data set

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	X ○ Δ	X Δ	X ○ Δ	X ○ Δ	○ Δ	X ○ Δ	X ○ Δ	X ○ Δ	X ○ Δ	X ○ Δ	○ Δ	○ Δ	X ○ Δ	X ○ Δ	X ○ Δ
-1		X ○	Δ	X ○	X ○ Δ	X ○ Δ		○		○ Δ	Δ	X ○ Δ	X ○ Δ		Δ
-2		X ○ Δ	X ○ Δ	X Δ	X ○ Δ		○		Δ	○	○ Δ	X ○ Δ	Δ		
-3		X	X Δ	X ○ Δ	Δ	Δ				○ Δ	X ○ Δ	○			Δ
-4		○ Δ	X ○ Δ	Δ	Δ				Δ	X ○ Δ	Δ	Δ			
-5	○ Δ	X ○ Δ	Δ	○			○		X ○ Δ	Δ	Δ		Δ		Δ
-6	X ○ Δ	○ Δ	Δ	○				X ○ Δ	Δ	○ Δ	○ Δ	Δ			X ○ Δ
-7	X Δ	○ Δ			○	X ○	○			○			○	○ Δ	○
-8					X ○	X ○ Δ		○		○		X ○	X ○ Δ	○	Δ
-9		○			X ○ Δ	○				○	Δ	X ○ Δ	Δ		
-10			○ Δ	X ○ Δ							X ○ Δ	○			
-11	○	○ Δ	X ○ Δ	Δ					○ Δ	X Δ	X				
-12	○	○ Δ		X				Δ	X ○ Δ						
-13	○ Δ		X ○	X				X ○ Δ						X	X ○ Δ
-14	○		X ○				X ○ Δ	X ○ Δ	X ○ Δ	X Δ	○		X ○	X ○ Δ	X Δ

Table B.2: Selected inputs for Poland Electricity data set: the numbers in the first row and first column represent time steps and regressor index. Symbol X is for MI selected inputs, O represents NNE selection results, Δ is for l-NN selected results

B.3 Selected inputs for Darwin sea level data set

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	X ○ Δ	X ○ Δ	X ○ Δ	X ○ Δ	X ○ Δ	X ○ Δ	X ○ Δ	X ○ Δ	X ○ Δ	X ○ Δ	X ○ Δ	○ Δ	X ○ Δ	X ○	○
-1	X ○ Δ	X ○ Δ	X ○ Δ	○ Δ	X ○ Δ	X ○ Δ	X ○	○ Δ	X ○	X ○ Δ	X ○	X ○		X ○	○ Δ
-2	○ Δ	X ○	X ○ Δ	X ○ Δ	X ○ Δ	X	Δ	X ○	Δ	Δ	X ○ Δ	○	○	○ Δ	X ○
-3	○	X ○ Δ	○ Δ	X ○ Δ	X Δ	X	X ○	X ○ Δ	○ Δ	X ○ Δ	X	X ○ Δ	Δ	X	X ○
-4	X	○ Δ	X ○ Δ	○ Δ		X ○ Δ	X ○ Δ	X	X ○ Δ	○	○ Δ	X ○	X ○ Δ	X ○ Δ	○ Δ
-5	○	○ Δ	X	○	X ○	X ○ Δ		X ○ Δ	○	X ○ Δ	X ○ Δ	X Δ	X ○ Δ	X ○ Δ	X ○ Δ
-6	X	○		X	X ○ Δ	X	X ○ Δ		X ○ Δ	X ○ Δ	X ○ Δ	X ○ Δ	○ Δ	○ Δ	X ○ Δ
-7	X	○ Δ	X ○	X ○ Δ	○	X ○ Δ	○	X Δ	X Δ	X ○ Δ	X Δ	X ○ Δ	X ○ Δ	○ Δ	X ○
-8	○	○	X ○ Δ		X ○ Δ	○	X ○	X ○	X	X Δ	X ○ Δ		○	○	○ Δ
-9		X ○		X ○ Δ	○	X Δ	X ○	X ○ Δ	X ○ Δ	X ○ Δ	○ Δ	○ Δ	○		X ○
-10	○ Δ	X ○ Δ	X ○ Δ	○		X Δ	X ○	X ○ Δ	X ○ Δ	○ Δ	X ○ Δ	X	Δ	○ Δ	
-11	○	○ Δ		○ Δ		X ○	○	X ○ Δ	○ Δ	○	X ○		○	X	
-12	X	○	X	X Δ	X ○ Δ	X Δ	X ○ Δ	○ Δ	○ Δ	X ○	○ Δ	X ○	X	○ Δ	
-13	○	X Δ	X Δ	○ Δ	○	X ○ Δ	○ Δ	○	X ○		X ○		X Δ	○	○
-14	○ Δ		X ○ Δ	○	X ○ Δ	X ○ Δ	○	X ○	○ Δ	○	X ○	Δ	Δ	○	○ Δ
-15	X	X ○ Δ	○ Δ	X ○	Δ	X Δ	X ○	Δ			Δ	X	○	X ○ Δ	X
-16	Δ	X ○	X	○ Δ	○	X ○ Δ	○		○	Δ	X		○ Δ	○ Δ	X
-17		○	X	○	○	X Δ	○ Δ	○	Δ	○	X ○	○ Δ	X Δ	○	X ○ Δ
-18	X		X	○	Δ			○ Δ		○	X ○	Δ	Δ	Δ	○
-19	X ○		○	○		○ Δ	○	○	○	Δ	X ○ Δ	X		○	X ○ Δ
-20	X ○	X ○	X	Δ	○	○	○	○	Δ	X ○ Δ	X ○ Δ		○	○ Δ	X ○ Δ
-21	○		Δ		Δ	○	○	Δ	○ Δ	X ○ Δ		X	○ Δ	X Δ	X ○ Δ
-22	○ Δ	○ Δ	X ○	○ Δ	○	○ Δ	○ Δ	Δ	X ○	X ○ Δ	X	○ Δ	X ○ Δ	○	○
-23	○	○	X ○ Δ	○	○ Δ	X ○	○ Δ	X ○	○ Δ	○ Δ	X ○ Δ	X ○ Δ		○ Δ	○
-24		X ○		○ Δ		X	X	○	○ Δ	○ Δ	○ Δ				○ Δ
-25	○ Δ	○	X Δ	○ Δ	X ○ Δ	X ○ Δ	○	○	○ Δ	X Δ	X ○	○	○	○ Δ	X ○
-26	○	○	X ○ Δ	○ Δ		X	○	○ Δ	X ○ Δ	○ Δ	X	○	Δ	○ Δ	X ○ Δ
-27	○	○ Δ		○		○	○ Δ	X ○ Δ	○ Δ	○		X ○ Δ	Δ	○ Δ	X ○
-28	X ○ Δ						X ○	○	○	○ Δ	Δ	○	X ○ Δ	○	X ○ Δ
-29					○		○		X ○	○ Δ	Δ	X ○ Δ	X ○	X Δ	X ○

Table B.3: Selected inputs for Darwin sea level data set: the numbers in the first row and first column represent time steps and regressor index. Symbol X is for MI selected inputs, O represents NNE selection results, Δ is for l-NN selected results