CHAPTER 2

Principles and Theories of the Study

This chapter delineates the essential basic theories used in this study. Section 2.1 describes the definition of input-output clustering which is the core idea of the reduction method for regression problems. In Section 2.2, the definition of a prototype selection method was proposed to present the type of data reduction for classification problems. Section 2.3 to 2.5 states the necessary theories using for data reduction on regression problems. The last section explains the supervised learning for regression and classification problems such as Support vector machine, nearest neighbor rule, and artificial neural network.

2.1 Output-constrained clustering

The output-constrained clustering algorithm [23] consists of two stages; rough clustering and refined clustering. In the rough-clustering stage, the output space is partitioned into intervals. Each partition is called an output constraint. Then the training data was roughly grouped based on this output partition to obtain a set of clusters. These resulting partitions are called clusters. In the refined-clustering stage, data within each output constraint are further grouped based on the connectivity of the inputs. The resulting refined clusters are called subclusters. If the data within one output constraint are located in other input region, then only one subcluster is required to separate these data from the remaining data. If data within one output constraint are segmented into several nonconnective input regions by the data outside this output constraint, then the multiple subclusters (each of which represents one connective input region) are needed to separate these data from the remaining data. The number of subclusters is equal to the number of the nonconnective input regions within that output constraint.

2.2 Prototype selection method

The report papers of D.R. Wilson [21] and G. Salvador [22] summarized the properties of prototype selection methods. They consist of the direction of search, type of selection, evaluation of search and criteria for comparing the prototype selection methods performance as follows.

2.2.1 Direction of search

From the training data set ***T***, we can extract or maintain the important data and move this data to the subset ***S*** with the different direction of searching. The direction of search can be distinguished into five directions as follows.

1) Incremental search

An incremental search starts by initializing set ***S*** as empty set and adds each instance in ***T*** to ***S*** when the instance satisfies the criteria. The order of instances in ***T*** could be very important for some incremental algorithms. Then, the order of presentation of instances in ***T*** should be random. Recently, some incremental approaches are order independent because they add instances to ***S*** in a somewhat incremental fashion, but they examine all available instances to help selecting which instance to add next. This makes the algorithm not truly incremental approach.

The advantage of an incremental scheme is that it is faster and uses less storage during the learning phase than non-incremental algorithms. If instances were made available after training is complete, they can continue to be added to ***S*** according to the same criteria. This capability could be very helpful when dealing with data streams or online learning. The main disadvantage of incremental algorithms is it uses more sufficient information to decrease the errors.

2) Decremental approach

The decremental search starts by setting ***S*** = ***T***, and then using the criteria to search the instance in order to remove from ***S***. The order of

presentation is important but unlike the incremental algorithm, all of the training examples are available for examination at any time. The disadvantage of the decremental algorithm is a higher computation cost than incremental algorithm and the learning process must be done in an offline fashion because the decremental algorithms need all possible data in the data space.

3) Batch approach

All instances that agree with the removing criteria are removed at the same time. The batch process, like the decremental algorithms, consumes more computation time than incremental algorithms.

4) Mixed approach

The mixed search starts by selecting instance in the training set to initial subset ***S*** (randomly or manually selected by an incremental or decremental process) and uses iterative procedures to adding or remove instances according to the specific criterion. The main advantage of this algorithm is that it is easy to obtain a good accuracy but its drawback is similar to the decremental algorithms.

5) Fixed

A fixed search is a type of mixed search which the number of additions and removals remains the same. The final number of instances in ***S*** is determined at the beginning of the learning phase. This strategy of search is typical in prototype generation methods (PG).

2.2.2 Type of selection

The prototype selection algorithm has a different selection to retain or remove the border instances, central instances, or some other sets of data in ***T*** depending on the condition by the type of search. We can categorize the type of selection into condensation, edition and hybrid.

1) Condensation

This selection aims to retain the instances which near to the decision boundaries or the border instances. Because the internal instances do not much affect the decision boundaries as the border instances, they can be removed with causing a small effect on classification.

The reduction capability of condensation methods is rather high because the number of border instances is fewer than internal instances. This method preserves the accuracy over the training set but the generalization accuracy over the test set can be negatively affected.

2) Edition

This selection removes the near-border instances that are noisy or do not agree with their neighbors. Some algorithms smooth the decision boundaries and keep the internal instances that are not necessarily contributed to the decision boundaries. The generalization accuracy in test data is increase but the reduction rate is lower.

3) Hybrid

Hybrid methods aims to find the smallest subset ***S*** which maintains or even increases the generalization accuracy in test data. Above-mentioned strategies are applied to remove internal and border instances.

2.2.3 Evaluation of search

The kNN is a simple technique which can be used to direct the search of a prototype selection algorithm. The objective is to make a prediction on a non-definitive selection and to compare between selections. This characteristic influences the quality criterion and the evaluation of search can be divided into:

1) Filter

The kNN rule is used for partial data to determine the criteria of adding or removing. No leave-one-out validation scheme is used to obtain a good estimation of generalization accuracy. The reduction efficiency of these methods increases when the fact of using subsets of the training data in each decision, but the accuracy may not be enhanced.

2) Wrapper

The kNN rule is used for the complete training set with the leave-one-out validation scheme. The conjunction in the use of the two mentioned factors allows us to get better generalization accuracy over test data. However, the learning phase can be computationally expensive.

2.2.4 Criteria to compare prototype selection methods

The values of four criteria used to evaluate the prototype selection methods are storage reduction, noise tolerance, generalization accuracy and time requirements. The details of four criteria are described as follows.

1) Storage reduction

The main goal of the prototype selection methods is to reduce the number of instances for decreasing storage memory. The advantage of the smaller memory usage is increasing of speed of classification process.

2) Noise tolerance

The problems that occur when the training set is contain too much noise. The important instance will be removed because some prototype selection methods need to maintain the noisy decision boundaries and the generalization accuracy can suffer if noisy instances are retained instead of good instances.

3) Generalization accuracy

All of prototype selection methods have the same goals that are to maintain the significant data and reduce non-important data without reducing or losing the accuracy of classification. In some prototype selection methods, the reduced data can increase the classification accuracy than the original data.

4) Time requirement

Time requirement is not an important evaluation method because the learning process is done once on a training set. But some algorithms are not practical for real applications if the learning phase takes too long time.

2.3 Fuzzy c-means clustering algorithm

Fuzzy c-means clustering (FCM) is a popular clustering method used in pattern recognition work. The objective function of FCM is written in the form [24]

$$J_{FCM}^{m} = \sum_{i=1}^{n}\sum_{k=1}^{c} u_{ik}^{m} D_{ik}^{2}, \tag{2.1}$$

where n is the number of data points.

c is the number of clusters.

m is the fuzzifier and $m \in [1, \infty)$.

u_{ik}^{m} is the fuzzy membership value.

D_{ik}^{2} is the Euclidean distance between data point i and the center of cluster k.

The FCM procedure is described as follows:

1) Initialize parameters c, m and set $t = 1$.

2) Initialize fuzzy membership values in parameter $\mathbf{U}(t)$ with these properties

$$\mathbf{U} = [\vec{u}_1 \quad \vec{u}_2 \quad \dots \quad \vec{u}_n], \tag{2.2}$$

$$\vec{u}_i = [u_{i1} \quad u_{i2} \quad \dots \quad u_{ic}]^{\mathrm{T}}, \tag{2.3}$$

$$u_{ik} \in [0,1],\ k = 1,...,c, \tag{2.4}$$

$$\sum_{k=1}^{c} u_{ik} = 1,\ i = 1,...,n, \tag{2.5}$$

$$0 < \sum_{i=1}^{n} u_{ik} < n,\ k = 1,...,c, \tag{2.6}$$

3) Find the center of each cluster from Equation (2.7)

$$\vec{z}_k(t) = \frac{\sum_{i=1}^{N} u_{ik}^m (t-1)\vec{x}_i}{\sum_{i=1}^{N} u_{ik}^m (t-1)}, \tag{2.7}$$

where $\vec{x}_i$ is data point *i*.

4) Update the value in $\mathbf{U}(t)$ by Equation (2.8)

$$u_{ik}(t) = \frac{1}{\sum_{j=1}^{c} \left(\frac{D_{ik}^2}{D_{ij}^2} \right)^{\frac{1}{m-1}}}, \tag{2.8}$$

5) Increment *t*, $t \leftarrow t+1$

6) If

$$\sum_{k=1}^{c} \left\| \vec{z}_k(t) - \vec{z}_k(t-1) \right\| < \varepsilon, \tag{2.9}$$

or

$$\left(J_{FCM}^m(t) - J_{FCM}^m(t-1) \right)^2 < \varepsilon. \tag{2.10}$$

where $\vec{z}_k$ is the center of cluster *k*.

ε is an acceptable error.

Then stop process.

Else go to step 3)

2.4 Cluster validity indices

Cluster validity indices [25] are used for measuring the goodness of a clustering result. The indices consists of partition index, separation index, davies bouldin index, dunn's index and SD validity index.

2.4.1 Partition index (*SC*) is the ratio of the sum of compactness and separation of the clusters. It is a sum of individual cluster validity measures normalized through division by the fuzzy cardinality of each cluster as shown in Equation (2.11).

$$SC(c) = \sum_{i=1}^{c} \frac{\sum_{j=1}^{n} \left(\mu_{ij}\right)^{m} \left\| x_j - v_i \right\|^2}{n_i \sum_{k=1}^{c} \left\| v_k - v_i \right\|^2}. \tag{2.11}$$

where v_i is the center of cluster *i*.

SC is useful when the different partitions have equal number of clusters. A lower value of *SC* indicates a better partition.

2.4.2 Separation index (*S*), on the contrary to partition index (*SC*), uses a minimum-distance separation for partition validity as shown in Equation (2.12).

$$S(c) = \frac{\sum_{i=1}^{c} \sum_{j=1}^{n} \left(\mu_{ij}\right)^{2} \left\| x_j - v_i \right\|^2}{n \min_{i,k} \left\| v_k - v_i \right\|^2}. \tag{2.12}$$

2.4.3 Dunn's index (*DI*) is originally proposed to use as the identification of compact and well-separated clusters. Therefore, the result of the clustering has to be recalculated as it is a hard partition algorithm,

$$DI(c) = \min_{i \in c} \left\{ \min_{i \in c, i \neq j} \left\{ \frac{\min_{x \in c_i, y \in c_j} d(x, y)}{\max_{k \in c} \left\{ \max_{x, y \in c_k} d(x, y) \right\}} \right\} \right\}. \tag{2.13}$$

where $d(x, y)$ is the distance between data points *x* and *y*.

The main drawback of Dunn's index is consuming high computational cost since the calculation becomes very expensive as *c* and *n* increase.

2.4.4 Davies bouldin index (*DB*) is based on the similarity measure of cluster (R_{ij}) whose bases are the dispersion measure of a cluster (s_i) and the cluster dissimilarity measure ($d(v_i, v_j)$).

$$DB = \frac{1}{n_c}\sum_{i=1}^{n_c} \max_{j=1,..n_c, i\neq j} \left(\frac{\frac{1}{\|c_i\|}\sum_{x\in c_i} d(x, v_i) + \frac{1}{\|c_j\|}\sum_{x\in c_j} d(x, v_j)}{d(v_i, v_j)} \right). \tag{2.14}$$

2.4.5 SD validity index (*SD*) is the average scattering of clusters and total separation of clusters. The scattering is calculated by variance of the clusters and variance of the data set. The total separation of cluster is based on the distance of cluster centre points.

$$SD = \alpha \cdot \frac{1}{n_c}\sum_{i=1}^{n_c} \frac{\|\sigma(v_i)\|}{\|\sigma(x)\|} + \frac{\max\limits_{i,j=1...n_c}\left(\|v_j - v_i\|\right)}{\min\limits_{i,j=1...n_c}\left(\|v_j - v_i\|\right)} \sum_{k=1}^{n_c}\left(\sum_{j=1, i\neq j}^{n_c} \|v_j - v_i\| \right)^{-1}. \tag{2.15}$$

2.5 Technique for order preference by similarity to ideal solution

The Technique for order preference by similarity to ideal solution (TOPSIS) was developed based on the concept that the chosen alternative should have the shortest distance from the Positive ideal solution (PIS) and the furthest distance from the Negative ideal solution (NIS) [26]. Assume that a multi-attribute decision making (MADM) problem has m alternatives x_i $(i = 1, 2, ..., m)$, which are evaluated on n attributes a_j $(i = 1, 2, ..., n)$. Denote that $X = \{x_1, x_2, ..., x_m\}$ and $A = \{a_1, a_2, ..., a_n\}$. Let h_{ij} be the values of alternatives $x_i \in X$ on the attributes $a_j \in A$. All these values are concisely expressed in the matrix format as $H = (h_{ij})_{m\times n}$, which is referred to a decision matrix usually represented as an MADM problem.

Assume that weights of the attributes $a_j \in A$ are w_j, which satisfy the normalization conditions $w_j \in [0,1]$, and $\sum_{j=1}^{n} w_j = 1$. The process of the TOPSIS is summarized as follows.

1) Normalize the decision matrix $H = (h_{ij})_{m\times n}$ using

$$r_{ij} = \frac{h_{ij}}{\sqrt{\sum_{i=1}^{m} h_{ij}^2}} \quad (i = 1, 2, ...m;\ j = 1, 2, ..., n), \tag{2.14}$$

where r_{ij} is called the normalized value.

2) Calculate the weighted-normalized decision matrix $Y = \left(y_{ij}\right)_{m\times n}$ using

$$y_{ij} = w_j r_{ij} \ (i = 1, 2, ..., m;\ j = 1, 2, ..., n), \tag{2.15}$$

3) Determine the PIS and the NIS, whose weighted normalized-value vectors are defined as $Y^* = \left(y_1^*, y_2^*, ..., y_n^*\right)$ and $Y^- = \left(y_1^-, y_2^-, ..., y_n^-\right)$, respectively, where $y_j^* = \max\left\{y_{ij} \mid i = 1, 2, ..., m\right\}$ for $j \in \Omega_b$, or $y_j^* = \min\left\{y_{ij} \mid i = 1, 2, ..., m\right\}$ for $j \in \Omega_c$, $y_j^- = \min\left\{y_{ij} \mid i = 1, 2, ..., m\right\}$ for $j \in \Omega_b$, or $y_j^- = \max\left\{y_{ij} \mid i = 1, 2, ..., m\right\}$ for $j \in \Omega_c$. Ω_b and Ω_c are subscript sets of benefit attributes and cost attributes, respectively.

4) Calculate the Euclidean distances of alternatives from the PIS as well as the NIS as

$$D_i^* = \sqrt{\sum_{j=1}^{n}\left(y_{ij} - y_j^*\right)^2}, \tag{2.16}$$

and

$$D_i^- = \sqrt{\sum_{j=1}^{n}\left(y_{ij} - y_j^-\right)^2}, \tag{2.17}$$

5) Calculate the relative-closeness coefficients of alternatives to the PIS as

$$C_i^{hy} = \frac{D_i^-}{D_i^* + D_i^-} \ i = 1, 2, ..., m. \tag{2.18}$$

Obviously, $C_i^{hy} \in [0,1]$ where $i = 1, 2, ..., m$, and the larger C_i^{hy} is, the better x_i becomes.

6) Rank the alternatives according to the descending order of the relative-closeness coefficients to the PIS.

2.6 Supervised learning

2.6.1 Support vector machine

The fundamental idea of support vector machine (SVM) for regression is to map input data to a higher dimension using a kernel function and then construct a separating hyper plane that provides the maximum margin in the feature space [8-10]. The result of the SVM is the estimated function in the form

$$f(\mathrm{x}) = \langle \mathrm{w} \cdot \Phi(\mathrm{x}) \rangle + b \tag{2.24}$$

where w is a weight vector.

b is a bias.

$\langle \cdot \rangle$ is an inner product.

Φ is a kernel function for mapping data to a higher dimension.

The data set used to train support vector machine for regression is $\{\mathrm{x}_i, y_i\}_{i=1}^{\ell}$, $\mathrm{x}_i \in \Re^n$, $y_i \in \Re$ where x_i is the input data vector and y_i is the output data. The function includes the deviation value (ε) called the loss function as shown in Figure 2.1. All input data x_i that have the values within $\pm\varepsilon$ interval are called the "support vectors". If the values are outside the margin, then they will induce errors that should be minimized. The optimization equation in this problem is

$$\min_{\mathrm{w},\xi_i,\xi_i^*} \frac{1}{2}\|\mathrm{w}\|^2 + \mathrm{C}\sum_{i=1}^{\ell}\left(\xi_i + \xi_i^*\right) \tag{2.25}$$

subject to

$$\left.\begin{array}{c} y_i - \langle \mathrm{w} \cdot \Phi \mathrm{x}_i \rangle - b \le \varepsilon + \xi_i \\ \langle \mathrm{w} \cdot \Phi \mathrm{x}_i \rangle + b - y_i \le \varepsilon + \xi_i^* \\ \xi_i, \xi_i^* \ge 0 \end{array}\right\} \tag{2.26}$$

where C is the constant variable.

w is obtained by solving the optimization problem as

$$\mathrm{w} = \sum_{i=1}^{\ell} \left(\alpha_i - \alpha_i^* \right) \Phi \left(\mathrm{x}_i \right) \tag{2.27}$$

where α_i and α_i^* are Lagrange multipliers.

Substitute equation (2.27) into (2.24), the estimated function can be written in the form

$$\begin{aligned} f(\mathrm{x}) &= \sum_{i=1}^{\ell} \left(\alpha_i - \alpha_i^* \right) \left\langle \Phi \left(\mathrm{x}_i \right) \bullet \Phi \left(\mathrm{x} \right) \right\rangle + b \\ &= \sum_{i=1}^{\ell} \left(\alpha_i - \alpha_i^* \right) K \left(\mathrm{x}_i, \mathrm{x} \right) + b \end{aligned} \tag{2.28}$$

where $K \left(\mathrm{x}_i, \mathrm{x} \right)$ is a kernel mapping function between x and x_i.

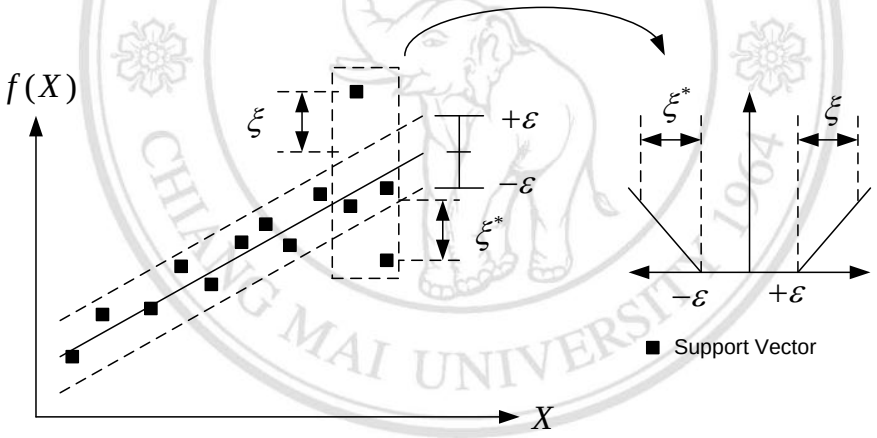

Figure 2.1 Soft margin ε –insensitive loss setting in a linear SVR.

After training support vector regression process, we obtain the solution of the optimization problem that is $\alpha_i^{(*)}$. The support vector regression model is shown in Figure 2.2

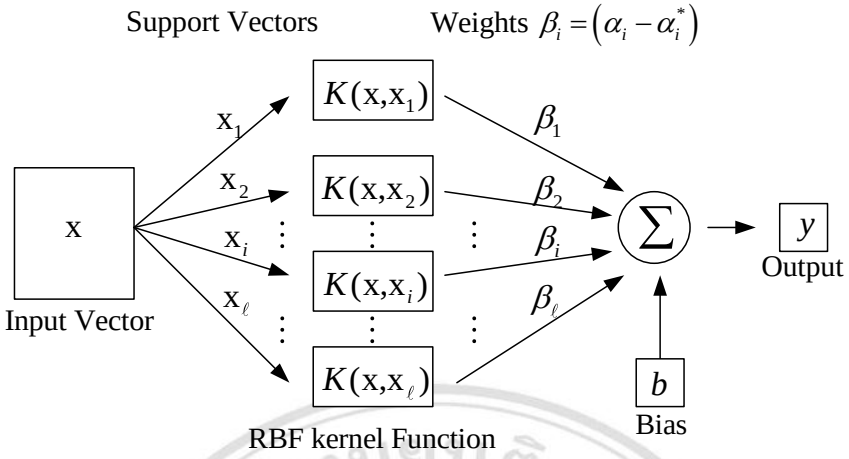

Figure 2.2 Support vector regression model.

2.6.2 Nearest neighbor rule

Nearest Neighbor (NN) [24] is the method for finding the closest or the most similar nearby point. The implementation of NN is described below

Given an unknown feature vector ***x*** and distance measure *a*, then:

- Out of the ***T*** training vectors, identify the *k* nearest neighbors, irrespective of class label. *k* is chosen to be odd for a two class problem, and in general not to be a multiple of the number of classes *C***.**

- Out of these *k* samples, identify the number of vectors. k_i, that belong to class ω_i, $i = 1, 2, \ldots, C$. Obviously, $\sum_i k_i = k$.

- Assign ***x*** to the class ω_i with the maximum number k_i of samples.

2.6.3 Artificial neural network

An artificial neural network (ANN) is a supervised learning method which forms a mathematical formulation by emulating the biological network. An ANN can be applied to both classification and regression problems so the areas of application are wide such as in engineering, medical, business, and military. Figure 2.3 demonstrates the typical structure of a feed forward neural network. The objective is to optimize the best set of weights (**W**) that the output $o_{j,n}$ are similar to the desired output as much as possible for an input $X_{i,n}$, i = 1,..., *N* and j = 1,...,*C*. *N* is the number of input features and *C* is the number of classes.

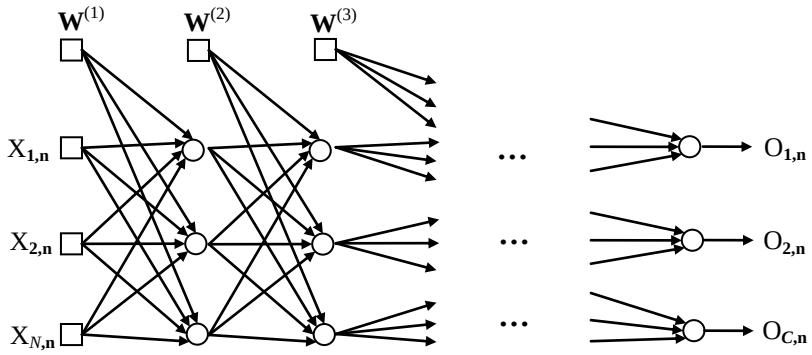

Figure 2.3 Feed-forward neural network model.