CHAPTER 5

ANALYSIS OF FEATURE SELECTION AND CLASSIFICATION METHODS FOR NEW DSS DESIGN

5.1 INTRODUCTION

Data mining can enable healthcare organizations to predict the flow in patient conditions and their behaviors. Medical science is another field where large amount of data is generated using different clinical reports and other patient symptoms. Healthcare data mining provides countless possibilities for hidden pattern investigation from the data sets. These patterns can be used by physicians to determine diagnosis, predictions and treatments for patients in healthcare organizations as stated by Boris Milovic & Milan Milovic (2012).

There are varieties of data mining methods available today; some of them are Support Vector Machines (SVM), Decision Tree, Principle Component Analysis (PCA), Neural Networks (NN), Naïve Bayesian Classifier, Genetic Algorithm and K-Nearest Neighbour (KNN).

Among numerous approaches, K-Nearest Neighbour algorithm and Artificial Neural Networks are considered as the most common and effective methods in classification problems. From the literature review it is concluded that the NN and KNN classifiers only produced more accurate result for heart disease prediction. Based on the feature attributes the classifiers predict the disease with more accuracy.
5.2 DECISION SUPPORT SYSTEM

In Healthcare industry generally clinical diagnosis is done mostly by doctor’s capability and experience. The computer aided decision support system helps the physician as a tool for disease diagnosis.

Manuel Filipe et al (2012) suggested that, “A Decision Support System for health care help doctors to decide about the best care to provide their patients. Such system is expected to reduce medical errors, fasten clinical intervention and overall to provide better care by having the required information at the right place, at the right time”.

5.2.1 Hybrid System for Decision Support

A combination of two or more methodologies within a design of single system results into a hybrid system. Hybrid systems extract the best from all methodologies and provide an optimal solution for clinical decision support systems. For example, to identify the clinically relevant aspects automatically, the combination of knowledge-based and statically techniques can be good approach.

5.2.2 Working Principle of Hybrid System

- Data Pre-processing

The selected data was checked for noise, inconsistency and missing values using distribution frequency. Noises and inconsistencies identified in the dataset were corrected manually, while missing values were replaced with the most probable value determined with regression and outliers were replaced with the mean value of the attributes.
At this stage after consulting with the domain expert a few transformations were implemented on the dataset to make the data more suitable for the data mining algorithms.

- **Feature Selection**

  In machine learning and statistics, feature selection is also known as variable selection, feature reduction, attribute selection or variable subset selection. It is the technique of selecting a subset, removing most irrelevant and redundant features from the data. Feature selection helps to improve the performance of learning models.

- **Classification**

  Classification is a data mining function that assigns items in a collection to target categories or classes. The goal of classification is to accurately predict the target class for each case in the data. A learning classifier is able to learn based on a sample.

### 5.2.3 Algorithms selected for this Research

In this research we have used three Algorithms for attribute selection. They are Back Propagation Algorithm., Singular Value Decomposition and Genetic Algorithm. Each algorithm works separately and provides the reduced feature attributes.

In the next step we have selected appropriate classifier for developing a predictive model. The algorithms Neural Network and K-Nearest Neighbour Classifier were selected for this research.
5.3. ALGORITHMS USED FOR FEATURE SELECTION

- Back propagation Algorithm
- Singular Value Decomposition
- Genetic Algorithm.

5.3.1 Back Propagation Algorithm

BP algorithm, which has been proposed by Rumelart in 1986. BP neural network has a great application because of the simple structure, multiple adjustable parameters and well manipulation. According to statistics, 80% to 90% neural networks adopt BP network or the transformation of BP. BP neural network is similar to multilayer perceptron. It is a multilayer feed-forward neural network. Its name derives from adjusting training algorithm which is error counter-propagation algorithm in the training network.
The typical back-propagation network has an input layer, an output layer, and at least one hidden layer. There is no theoretical limit on the number of hidden layers but typically there are just one or two. Each layer is fully connected to the succeeding layer. Training inputs are applied to the input layer of the network, and desired outputs are compared at the output layer. Back propagation passes error signals backwards through the network during training to update the weights of the network prediction and the actual target value and these modifications are applied in the backward direction.

Figure 5.2 illustrates the process of Back Propagation.

**Figure 5.2 Back propagation process**

5.3.1.1 Back Propagation Algorithm Steps

Step 1. Provide training data to network.

Step 2. Compare the actual and desired output.
Step3. Calculate the error in each neuron.

Step4. Calculate what output should be for each neuron and how much lower or higher output must be adjusted for desired output.

Step5. Then adjust the weights.

5.3.1.2 Applications of Back Propagation Algorithm

- speech recognition
- voice recognition
- image pattern recognition
- medical diagnosis
- automatic controls

5.3.2 Singular Value Decomposition

Singular Value Decomposition (SVD) is one of the most important matrix decompositions used in computer vision. SVD has the added benefit that in the process of dimensionality reduction. The representation of items that share substructure become more similar to each other and items that were dissimilar to begin with may become more dissimilar as well. This reduction removes unnecessary data that are linearly dependent in the point of view of Linear Algebra.

5.3.2.1 Principal Component Analysis

Principal Component Analysis (PCA) is a specific case of SVD.

PCA is one of the factor analysis techniques that identify a meaningful interrelationship among the attributes in the data. It provides a way of obtaining data with a reduced dimension without any loss. Singular
value decomposition and principal component analysis are two eigenvalue methods used to reduce a high-dimensional dataset into fewer dimensions while retaining important information.

Principal component analysis is a standard tool in modern data analysis. It is a simple non-parametric method for extracting relevant information from confusing data sets. PCA method is used for achieving the simplification and generates a new set of variables, called principal components. Each principal component is a linear combination of the original variables. All the principal components are orthogonal to each other, so there is no redundant information. The principal components as a whole form an orthogonal basis for the space of the data. SVD, PCA and "total least-squares" are the same thing. PCA needs SVD or eigenvalue decomposition to calculate principal components.

5.3.2.2 Procedure of PCA

1. Derive PCs which are axes of PCA using SVD and select n eigenvectors corresponding to the n largest eigen values.

2. Project some samples to these PCs (to denoise, dimensionality reduction, feature extraction)

Step 1: Obtain the input matrix
Step 2: Subtract the mean from the data set in all dimensions
Step 3: Calculate covariance matrix of this mean subtracted data set.
Step 4: Calculate the Eigen values and Eigen Vector from covariance matrix
Step 5: Form a feature vector
Step 6: Derive the new data set.
Applications of SVD

- face recognition
- pattern recognition
- image compression
- signal processing
- linear inverse problems
- analysis of regularization methods
- natural language text processing
- computational models of subsurface oil and gas reservoirs
- numerical weather prediction
- Recommender systems

5.3.3 Genetic Algorithms

Genetic Algorithms (GAs) are stochastic search methods, which have been inspired by the process of biological evolution. The advantages of GAs become more obvious when the search space of a task is large. The GA was developed by John H. Holland in the 1970s to allow computers to evolve solutions to difficult search and combinatorial problems, such as function optimization and machine learning. Genetic Algorithms have become a highly effective tool for solving hard optimization problems.

In feature selection GA includes an optimization process, in which many combinations of features and their interactions are considered. Because GA searches for solutions efficiently in high dimensional and difficult response surfaces, it can be utilized for feature selection in a variety of problems and multivariate calibration in particular.
5.3.3.1 Life cycle of Genetic Algorithm

In GA, the parameters of the search space are encoded in the form of strings, called chromosomes. A collection of such strings is called a population. Initially, a random population is created, which represents different points in the search space. An objective and fitness function is associated with each string that represents the degree of goodness of the string. Based on the principle of survival of the fittest, a few of the strings are selected and each is assigned a number of copies that go into the mating pool. Biologically inspired operators like cross-over and mutation are applied on these strings to yield a new generation of strings. The process of selection, crossover and mutation continues for a fixed number of generations or till a termination condition is satisfied.

Figure 5.3 Genetic Algorithm Life Cycle
5.3.3.2 Procedure of Genetic Search

Step 1: Represent the problem variable domain as a chromosome of a fixed length; choose the size of a chromosome population $N$, the crossover probability $P_c$, and the mutation probability $P_m$.

Step 2: Define a fitness function to measure the performance, or fitness of an individual chromosome in the problem domain. The fitness function establishes the basis for selecting chromosomes that will be mated during reproduction.

Step 3: Randomly generate an initial population of chromosomes of size $N$: $x_1, x_2, \ldots, x_N$.

Step 4: Calculate the fitness of each individual chromosome: $F(x_1), f(x_2), \ldots, f(x_N)$.

Step 5: Select a pair of chromosomes for mating from the current population. Parent’ chromosomes are selected with a probability related to their fitness.

Step 6: Create a pair of offspring chromosomes by applying the genetic operators-cross over and mutation.

Step 7: Place the created offspring chromosomes in the new population.

Step 8: Repeat step 5 until the size of the new chromosome population becomes equal to the size of the initial population $N$.

Step 9: Replace the initial (parent) chromosome population with the new (offspring) population.

Step 10: Go to step 4, and repeat the process until the termination criterion is satisfied.
Genetic Algorithms Applications

- VLSI design
- aircraft Design
- communication Network
- keyboard configuration
- image processing
- graph colouring and partitioning
- job shop scheduling
- audio watermark insertion/detection
- power electronics design
- gas pipeline control

5.4 CLASSIFIERS USED FOR DATA PREDICTION

- Artificial Neural Network
- K-Nearest Neighbour

5.4.1 Artificial Neural Network

Artificial Neural Networks (ANN) are those systems modeled based on the human brain working. As the human brain consists of millions of neurons that are interconnected by synapses, a neural network is a set of connected input/output units in which each connection has a weight associated with it. The network learns in the learning phase by adjusting the weights so as to be able to predict the correct class label of the input. It required long training time and poor interpretability. Support for highly noisy data and classify the untrained data. It is well suitable for classifying the continuous valued data. The feature of neural networks is an iterative learning
process in which data cases are presented to the network one at a time and the weights associated with the input values are adjusted each time. After all cases are presented, the process often starts over again.

ANN is an artificial intelligence technique that is used for generating training data set and testing the applied input data. A Multilayer Perceptron is used for the proposed method. Normally, a back propagation neural network has an input layer, an output layer, with one or more hidden layers in between the input and output layer. The input layer consists of the following inputs i.e. $A_1, A_2, A_3, A_4, A_5, A_6, A_7, A_8, A_9, A_{10}, A_{11}, A_{12}, A_{13}, A_{14}$. The ANN functions as follows: each node $H_i$ in the input layer has a signal $A_i$ as network’s input, multiplied by a weight value between the input layer and the hidden layer.

![Figure 5.4 NN Design for Heart Disease prediction](image_url)

**Figure 5.4 NN Design for Heart Disease prediction**
The training phase is classified into following four major steps. The training steps involved in neural network are as follows,

**Step 1:**

- Initialize the input, output and weight of each neuron. Here, \( A_1, A_2, A_3, A_4, A_5, A_6, A_7, A_8, A_9, A_{10}, A_{11}, A_{12}, A_{13}, A_{14} \) are technical indicators, i.e. input of the network and \( (P_k)_{output} \) is the predict value, i.e. output of the network.

**Step 2:**

- Each node \( I_k \) the hidden layer receives the value \( \ln(I_k) \) according to:
  \[
  \ln(I_k) = \sum_{i,k=1}^{N} A_i I_k
  \]  
  \( (5.1) \)

- Then passed through the bipolar function:
  \[
  f(A) = \frac{2}{1 + \exp(-A)} - 1
  \]  
  \( (5.2) \)

- The output of the activation function \( f(\ln(I_k)) \) is then broadcast all of the neurons to the output layer:
  \[
  (P_k)_{output} = \sum_{k,n=1}^{N} H_k P_k(n)
  \]  
  \( (5.3) \)

**Step 3:**

- The inputs of training dataset are \( A_1, A_2, A_3, A_4, A_5, A_6, A_7, A_8, A_9, A_{10}, A_{11}, A_{12}, A_{13}, A_{14} \) to classifier and determine the error function as follows:
  \[
  E_v = (P_k)_{target} - (P_k)_{output}
  \]  
  \( (5.4) \)
In Equation (5.4) $(P_k)_{target}$ is the target output and $(P_k)_{output}$ is the network output

**Step 4:**

- Adjust the weights $w$ of all neurons as $w = w + \Delta w$, where, $\Delta w$ is the change in weight which can be determined as follows,

$$\Delta w = \beta . P_k . E_v$$  \hspace{1cm} (5.5)

In Equation (5.5), $\beta$ is the learning rate, usually it ranges from 0 to 1.

**Step 5:**

- Repeat the process from step 2, until error gets minimized to a least value i.e

$$E_v < 0.1$$  \hspace{1cm} (5.6)

**Advantages of Neural Networks for classification**

- Neural Networks are more robust because of the weights and is more robust in noise environment

- The Neural Networks improves its performance by learning.

- The use of Neural Networks can be parallelized as specified above for better performance.

- There is a low error rate and thus a high degree of accuracy once the appropriate training has been performed.
5.4.2 K-Nearest Neighbour Classifier

KNN algorithms have been used since 1970 in many applications like statistical estimation and pattern recognition etc. One of the very simple machine learning and the most usable classification algorithm which is used largely in different applications. Due to good results although with a simplicity to solve many classification problems, this algorithm is still extremely popular in many research fields. No training is given in this method and all of the training data are memorized. Therefore, it is known as an instance-based and lazy method. K nearest neighbour (KNN) is an algorithm which stores all cases and classifies new cases based on similarity measure.

The Simple K-NN Algorithm

K-NN algorithm as follows:

- Determine the K i.e., the number of nearest neighbours
- Using the distance measure, calculate the distance between the query instance and all the training samples.
- The distance of all the training samples are sorted and nearest neighbour based on the K minimum distance is determined.
- Since the KNN is supervised learning, get all the categories of the training data for the sorted value which fall under K.
- The prediction value is measured by using the majority of nearest neighbours.

This algorithm operation is based on comparing a given new record with training records and finding training records that are similar to it. Each record with n attributes represents a point in an n-dimensional space.
Therefore, all of the training records are stored in an n-dimensional space. When given a new record, KNN algorithm searches the space for the K training records that are nearest to the new record as the new record neighbours and then predict the class label of new record with use of the class label of these neighbours. In this algorithm nearest is defined in terms of a distance metric such as Euclidean distance.

5.4.2.1 Problems in K-Nearest Neighbour

- **Distance Function**

  KNN is based on a distance function that measures the difference or similarity between two instances. In KNN, the standard Euclidean distance is used that will be dominated by the large number of irrelevant attributes. If data instances have many attributes then the algorithm may tend to over fit the data by picking up on false relationship. This results in misleading of classification process and decreasing the accuracy of classification algorithm. KNN is especially sensitive to this problem.

- **The Curse of Dimensionality**

  If data instances have many features, then algorithms may tend to over the data by picking up on spurious correlations. This is known as the Curse of Dimensionality (too many dimensions are bad).

- **Feature Selection**

  The author (Jiang 2006) stated a strong approach to overcoming this problem is to weight each attribute differently and completely eliminate the least weighed attributes from the attribute space and selects the weighted attributes when calculating the distance between two instances. Use a tuning
set on different combinations of features to find out what works best to eliminate the least weighted attributes. The ordinary tuning set take more amount of time to try all possibilities for f features. It takes 2f trials (way too much time). Instead, we can use hill-climbing heuristics such as forward feature selection and backward feature selection.

5.5 IMPROVED K-NEAREST NEIGHBOUR CLASSIFIER

A novel hybrid decision support system named Improved K-Nearest Neighbour classifier is designed to classify biomedical bench mark data. It is a combination of K-nearest Neighbour classifier technique and back propagation. KNN is one of the memory-based classification methods but it still has many defects, such as great calculation complexity. To avoid this, the back propagation algorithm is used. Here the multilayer perceptron train with back-propagation algorithm used for reduction of all unwanted attributes, after the reduction of irrelevant attributes used KNN algorithm for better classification.

5.5.1 Distance Calculation

It is one of a memory-based Classification technique with continuous attributes. The difference between attributes is calculated by using the Euclidian distance formula as

\[ d(X_i, X_j) = \sqrt{\sum_{m=1}^{n} (X_{im} - X_{jm})^2} \]  \hspace{1cm} (5.7)

Where \( X_i, X_j \) are the training and testing attributes, \( d \) is the function used to find the distance between them and \( m \) is number of attributes \( m=1,2,\ldots,n \).
5.5.2 Normalization

The main complexity with Euclidean distance is the larger values rate swamps the smaller values. So the Normalization technique has to be applied on the continuous attributes. From that they have the same influence on the distance measure between instances. KNN usually deals with continuous attributes however it can also deal with discrete attributes. When dealing with discrete attributes if the attribute values for the two instances are different so the difference between them is equal to one otherwise it is equal to zero.

From Equation 5.7 replace \( X_{im} \) by \( Z_{im} \).

\[
Z_{im} = \frac{X_{im} - \overline{X_m}}{\sigma_m}
\]  

(5.8)

So that we can make them as zero mean \((Z_{im})\) and unit variance. where in Equation (5.8)

\[
\overline{X_m} = \frac{1}{N} \sum_{i=1}^{N} x_{im}
\]

is the empirical mean of \( m^{th} \) feature.

\[
\sigma_m = \frac{1}{N} \sum_{i=1}^{N} \left( X_{im} - \overline{X_m} \right)^2
\]

is the empirical variance of \( m^{th} \) feature.
5.5.3 Assign Weight by Back Propagation

Assigning weight to the features can improve the accuracy and minimize the error rate using back propagation method. Back Propagation [BP] is the technique which is used to find the net (weighted) value. The name given is back propagation because, it calculates the difference between the actual and predicted values and propagated from output nodes, backwards to nodes in previous layer.

The BP algorithm is capable of adjusting the network weights and biasing values to reduce the square sum of the difference between the given output \(X_{im}\) and an output values computed by the net \(X_{jm}\) with the aid of gradient decent method. Figure 5.3 shows the weight assignment for the attributes.

![Figure 5.5 Weight Assignment for Attributes](image)

- Calculate the weighted value

\[
net = \sum_{i=0}^{n} W_i X_i
\]
Where net is a weighed mean value, \( W_i \rightarrow \) weight assignment for attributes \( (W_1, W_2, ..., W_n), \) \( X_i \rightarrow \) attribute \( (X_1, X_2, ..., X_n) \) for each attribute we assign the weight value based on that we calculate the net value.

- **Calculate the output**

\[
O = \sigma(\text{net}) = \frac{1}{1 + e^{-\text{net}}}
\]

Where \( O \rightarrow \) is Output, Use the sigmoid activation function to calculate the output of each neuron.

Now, by combining \( k \) neighbours to provide a classification decision for the new record, we need a combination function. Generally, two types of combination functions exist: unweighted voting and weighted voting. In the unweighted voting combination function, the class label which has the majority between neighbours of new record is selected as the class label of the new record without considering the preference of each neighbour. But, in the weighted voting more weight is given to some neighbours, the ones which are more similar to new record. Thus, more weighted records have more effects on determining the class label. In this proposed algorithm we apply weighted voting called the net value calculated by BP algorithm into the standard Euclidean distance function to improve the performance of KNN classifier.

This is done to improve weights during processing. Now the standard Euclidean distance function is defined as

\[
d(X_i, X_j) = \sum_{m=1}^{n} \text{net} \cdot d(X_{im}, X_{jm})
\]

(5.9)

This Improved distance function which enhances the effectiveness of classification.
Advantages and Disadvantages of KNN

• Advantages

KNN has several main advantages: simplicity, effectiveness, intuitiveness and competitive classification performance in many domains. It is Robust to noisy training data and is effective if the training data is large.

• Disadvantages

  - KNN have poor run-time performance when the training set is large. It is very sensitive to irrelevant or redundant features because all features contribute to the similarity and thus to the classification. By careful feature selection or feature weighting, this can be avoided.
  
  - Computation cost is quite high.

5.6 STUDY ON REINFORCEMENT LEARNING

Machine learning is generally classified into three categories:

1. Supervised Learning (SL). These are learning in which we know the input as well as the output.

2. Unsupervised Learning (USL). These are learning in which we know the input but not the output.

3. Reinforcement Learning (RL). This kind of learning falls between the first two categories, in which we have the input, but not output, instead we have "reviewer" whether the classifier's output is correct or wrong.
RL is often the issue of an agent acting in an environment that tries to achieve the best performance over time by trial and error.

The Figure 5.6 illustrates a standard model of Reinforcement Learning.

![Figure 5.6 Reinforcement Learning](image)

RL algorithms can also be classified into

- Model-free methods
- Model-based methods

**Model-free methods** include Monte Carlo methods and temporal difference methods.

**Model-based methods** include dynamic programming, certainty equivalent methods, Prioritized sweeping, RTDP (Real-Time Dynamic Programming), the Plexus planning system etc.
5.6.1 Q-Learning

One important RL algorithm is the Q-learning algorithm.

The Q-learning algorithm, introduced by Watkins in 1989, is rooted in dynamic programming, and is a special case of TD (lambda) when lambda = 0. Q-learning is a form of model-free reinforcement learning. It can also be viewed as a method of asynchronous dynamic programming. It provides agents with the capability of learning to act optimally in Markovian domains by experiencing the consequences of actions, without requiring them to build maps of the domains.

It holds discounted infinite-horizon MDP (Markov Decision Process), is easy to implement, is exploration insensitive and is so far one of the most popular and seems to be the most effective model-free algorithm for learning from delayed reinforcement. However, it does not address scaling problem and may converge quite slowly. The Q-learning rule is

\[ Q(s, a) = Q(s, a) + \alpha [r + \gamma \max_{a'} Q(s', a') - Q(s, a)] \]

\[ \Pi^*(s) = \arg\max Q(s, a) \]  \hspace{1cm} (5.10)

Where: s - current state, s' - next state, a - action, a' - action of the next state, r - immediate reward, \( \alpha \) - learning rate, \( \gamma \) - discount factor, \( Q(s, a) \) - expected discounted reinforcement of taking action a in state s. \( <s, a, r, s'> \) is an experience tuple.
Q-learning algorithm

For each s, a, initialize table entry Q(s,a) <- 0
Observe current state s
Do forever:
    Select an action a and execute it
    Receive immediate reward r
    Observe the new state s'
    Update the table entry for Q(s, a) as follows:
    Q (s, a) = Q(s, a) + \alpha [ r + \gamma \max Q (s', a') - Q (s, a)]
    s <- s'

The converge criteria are:

1. The system is a deterministic MDP.
2. The immediate reward values are bounded by some constant.
3. The agent visits every possible state-action pair infinitely often.

Applications of Reinforcement Learning

- game playing
- robotics
- elevator control
- network routing and finance