A STUDY ON SUPERVISED CLASSIFICATION APPROACH USING SUPPORT VECTOR MACHINES

A Dissertation submitted in Accordance with the Requirements
for the award of the Degree of

Master of Science
in
Mathematics and Computing

By
Anshu Chopra
301703003

Under the supervision of
Dr. Vikas Sharma and Dr. Rajesh Mehta

JULY 2019

School of Mathematics
Thapar Institute of Engineering and Technology
Patiala-147004 (Punjab)
India

Dedicated

to God, My Family

and

Supervisors

CERTIFICATE

This is to certify that the work incorporated in this dissertation entitled **A STUDY ON SUPERVISED CLASSIFICATION APPROACH USING SUPPORT VECTOR MACHINES** in accordance with the requirements for the award of the degree of **MASTER OF SCIENCE IN MATHEMATICS AND COMPUTING** from Thapar Institute of Engineering and Technology during the year **2017-2019** is a study carried out by **ANSHU CHOPRA** under the guidance of **DR. VIKAS SHARMA AND DR. RAJESH MEHTA.**

The matter embodied in this dissertation is the review of the previous works and has not been submitted by this or any other university in partial or full form of award of such a degree.

Anshu Chopra
Reg. no. 301703003

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

SUPERVISORS:

Dr. Vikas Sharma
Assistant Professor
School of Mathematics,
Thapar Institute of Engineering and Technology,
Patiala

Dr. Rajesh Mehta
Assistant Professor
CSED,
Thapar Institute of Engineering and Technology,
Patiala

ACKNOWLEDGEMENT

Guidance plays an important role in achieving appropriate goals. Although hard work, strong will and dedication are vital elements, yet good assistance and support is required for better tomorrow. This dissertation was not possible without the support, guidance and patience of the following members of our institution.

First and foremost, I would like to thank God Almighty who has given me the courage and inner support to work hard for this dissertation. I was lucky enough to have supervision of two intellectual Assistant Professors of TIET, **Dr. Vikas Sharma and Dr. Rajesh Mehta**, who have always been there to encourage and guide me with their knowledge and patience. They have been a source of motivation and a constant support system throughout this dissertation. I appreciate both for being an inspiration to achieve the desired goal.

A sincere gratitude to **Dr. Satish Kumar Sharma**, Head SOM, TIET, who has provided us with all the facilities needed for this work. He has always helped and encouraged to work hard.

Our faculty and staff members are happy to help and are so co-operative. They have given solutions to every problem I faced.

A token of thanks and love to my family and friends who have given me a comfortable environment and always been there in the hours of need. Without their support this work would not have been possible.

My special gratitude to each and everyone who has supported me throughout this dissertation. Their love, support and encouragement has helped me to complete this work with dedication and courage.

ABSTRACT

Support Vector Machine (SVM) is the key for all the classification problems. It is a powerful tool that finds a classifier to separate two different classes that maximizes the margin between them. It was developed for binary classification and further it was used for solving multi-class problems. In this dissertation we have reviewed the study of support vector machines for binary and multi-class classification problem. This work is divided into the following four chapters:

CHAPTER 1: This chapter consists of the introduction to Machine learning algorithm. We have also included the history, importance and some applications of machine learning. Some important definitions that are required throughout the dissertation and some models that were used earlier are also written in chapter 1.

CHAPTER 2: The basics of Support Vector Machines (SVM) is introduced in this chapter. This involves some derivations for linear as well as non-linear classification. Further, algorithms for solving such problems are also presented in this chapter.

CHAPTER 3: In this chapter we have reviewed the work of O.L. Mangasarian and David R. Musicant, Lagrangian Support Vector Machines (LSVM). Their work consists of iterative method for binary classification unlike Support Vector Machines (SVM) which uses quadratic programming for finding support vectors. Algorithms for both linear and non-linear classification is explained in this chapter. Further, comparison between SVM and LSVM on various data sets is also included.

CHAPTER 4: This chapter explains Lagrangian Support Vector Machines for multi-class problems by J. P. Hwang, B. Choi, I. W. Hong, and E. Kim. Firstly, SVM for multi-class problems is described. Further, a method to solve multi-class problems (LSVMMPAC) considering all the classes at once is presented in this chapter. Later, LSVMMPAC for non-linear problems using kernel methods is explained in this chapter.

Contents

Chapter 1

Introduction

1.1 Machine Learning

Machine learning is the science of training computers in order to make them act, think and work like humans. It is a branch of artificial intelligence that finds solutions independently. It is such a tool that enables machines to work from experience. It is based on the idea that systems can learn from data, identify the patterns and make decisions with minimal human intervention. Machine learning contains various algorithms that train computers to work, like humans do every single day. Firstly, user feeds large amount of data to the computer algorithm and then further computer itself analyzes and makes decisions based on the data input by the user.

If we want a computer to recognize different objects for example distinguishing between fruits, first of all a user trains computer to learn the difference between their colors, features etc. After learning from data input by user, computer can itself identify the object given for testing. Machine learning also helps in calculating probabilities, making predictions, preventing fraudulent insurance claims etc.

The main aim of machine learning is understanding the structure of data and organizing this data into model for the understanding and utilization of people. Every technology now a days uses machine learning algorithm for instance, face recognition for tagging friends on social media, optical character recognition (OCR), recommendation engines that suggest users movies or shows of their choice, self driving cars etc. The focus of machine learning techniques is to make computers learn automatically from the data and predict from the given information.

1.1.1 Machine Learning - HISTORY

Machine learning today is quite different from the earlier learning techniques and is widely used in business and research work. In the early 1940s a model was created by Warren McCulloch and Walter Pitts using electric circuit which gave rise to neural network. Further, Turing Test was developed in 1950 by Alan Turing. It was a test for computers to convince humans that they are humans not computers. The most famous first computer game developed by Arthur Samuel at IBM for playing checkers helped world champions to improve their skills.

Later on Frank Rosenblatt created the first artificial neural network called Perceptron in 1958. This was used for pattern as well as shape recognization. In 1959 two models were developed named Adeline and Madeline at Stanford University by Bernard Widrow and Marcian Hoff. These models were used to find binary pattern and delete echo on phone lines respectively. These real world applications are still used today. In 1970s, a new architecture named Von Neumann came into existence. This is much easier than the neural network as both data as well as instructions are stored in the same memory. Further researchers started working on neural networks again in 1980s. Later the combination of computer science and statistics was a good idea for probabilistic approaches.

In 1997 a new computer known as Deep Blue was created which defeated the world champion of chess at that time, Garry Kasparov. Furthermore, many researches were made on digit recognition. In 2006 a new term Deep Learning came forward. Then many projects were launched for example GoogleBrain, AlexNet, DeepFace, DeepMind, OpenAI, ResNet, U-Net. Today cloud computing is in trend. This is making machine learning much more easier for further researches.

1.2 Importance of Machine learning

Machine Learning has become the base of business and a subject of great research. Companies are using large amount of data which requires large computational work. Machine Learning helps in every field and makes work much easier. Now a days humans are working less and machines are performing every task including decision making, predictions, automation etc. Machine Learning is growing day by day from the last few year. Large amount of data is available which we need to train the machines for making predictions this will help machines to learn better and grow. Machines can work more accurately than humans, therefore, efforts are being made and researchers are working to make machines work more effectively.

Earlier machine learning was not that popular as it is today. This is because of the developments in the field of artificial learning which is reducing human tasks and has become the building block of many industries. Now there is lot of competition among businesses, for example a company extracting values from data effectively can increase their sale by knowing the taste of the customers through data analysis. Company using data science effectively and making use of good strategies can rule the market.

Machine learning has made the work easier for humans. Machines can work much faster and with ease than humans can. It is a difficult task for a human to identify the category of email he or she receives, than computers. Therefore, machines are trained well to categorize the type of email received. Similarly, many applications like finding a path to travel has become much easier using map applications, even weather forecasting has helped in a lot of ways. If we need to search about something, we are just a second away from google search and can easily get the information.

Now we are safe enough from frauds. We can easily receive messages if someone tries to steal our money. Machine learning has helped in image recognition which made possible to see planets, night sky, starts etc. It has also helped to diagnose diseases from the previous data available. We can see now, machine learning is showing its impact in every field of this entire universe. People are working hard to get enough work for machines to make our work easier and effective.

1.3 Applications of Machine Learning

Machine Learning now a days plays an important role in industries, health care services, transportation, retail, government etc. There is no such field in this world where machine learning is not used. Many industries have recognized the value of machine learning as they use large amount of data which is difficult to work with, for a human being. Machine learning is such a tool that provides techniques for creative and efficient work.

Let us discuss some of its applications:

1. **Industries:** Machine learning has helped many industries especially finance companies to get rid of fraud. Companies can analyze data and can easily know when to invest and where. This helps in increasing business and making money. Marketing is improving day by day with the help of machine learning technologies. Now companies are attracting their customers by evaluating their likes and dislikes. Products are designed according to the customer's requirement. These innovations are increasing customer's loyalty and leading the companies in the market.

2. **Health Department:** There is a tremendous increase in the heath care industry in the past few years. Many gadgets with sensors are created that can scan and detect patient's health. Diseases are diagnosed so fast and easily now a days, so that they can

be easily cured. Predictions are made from the previously saved data of old patients and machines predict the chance of disease from the trained data.

3. **Transport system:** Machine learning is being used by many traveling organizations to identify routes. The GPS system in our devices helps in managing traffic. It can detect the area where traffic is high. The online cab booking system has also been using machine learning techniques by estimating the price of the ride.

4. **Social Networking:** Has anyone thought how do we get the news feed of our choice? This is machine learning technique only. Machines can easily detect from the type of data we see and show us the same kind of data every single time we open our news feed. It keeps check on the site we watch, shows we see and suggests us the data of our choice. Even suggestions of friends are also made on the basis of same likings. Machine learning uses face recognition technique by detecting the pictures of our friends. Searches made on google are also made simpler with the help of machine learning.

5. **Virtual Assistants:** Voice detector applications are the perfect example of virtual personal assistants for example siri, alexa, google etc. These applications run on the basis of commands given through voice. This is also a machine learning technology. These have made our work simpler, we can easily know anything just by talking to such applications. Such technology is so smart that they can help us remind of certain sort of things told by us to them.

We can clearly see the impact, machine learning has made in every field. No such department in this world is left without using machine learning techniques. In the fast growing world of data, machine learning will work wonders.

1.4 Categories of Machine Learning Algorithms

Machine learning algorithms can be classified into :

1. **Supervised Learning:** The word supervised clearly explains that the machines need to get trained earlier in order to test unknown data. In this type of learning algorithm computers learn from input data. Already some data is given with correct labels, which is called training data and the rest of the data is labeled by analyzing those trained data points. Supervised learning algorithm trains machines about the data labels, learning from this data, machines now classify the unknown data of same type easily. The purpose of this type of learning algorithm is to train itself from its actual output and compare this with taught output to calculate error percentage. For example Classification and Regression.

2. **Unsupervised Learning:** In this type of learning algorithm no prior labeling is done in the data. There is no training or guidance for the machines. The machine learn

from the information and feature of the data itself and sort the data according to their characteristics. The main aim is to find hidden patterns from the given data set as well as feature learning which trains machines to find representations required for the classification of raw data. For example Clustering and Association.

3. **Semi-Supervised Learning:** This type of learning is the combination of supervised as well as unsupervised learning. It includes small amount of labeled and large amount of unlabeled data. It is an improved learning technique with much accuracy as compared to the others. It helps to generate patterns and create relation between the target variable and the labeled data.

4. **Active Learning:** It is type of learning that questions user whenever it finds data unlabeled. It is type of supervised learning that chooses unlabeled data itself with high relevance. This feature of algorithm in which it learns from the data selected itself gives better accuracy.

1.5 Some Definitions

Definition 1.5.1. Shattering: Let S be any set of data points, suppose it is divided into two classes positive and negative. Let H be any function, S is said to be shattered by H if it divides the data points into the same positive and negative class as separated previously.

Definition 1.5.2. VC-dimension: The VC dimension generally known as Vapnik-Chervonenkis dimension denoted by VC(H). The VC dimension of hypothesis space H of the instance space S is defined as the size of the largest finite subset of S shattered by H.

Definition 1.5.3. Linearly separable sets: Let X and Y are two sets in $\mathcal{R}^n$. These sets are said to be linearly separable if there exists real numbers $k_1, k_2, ..., k_n, b$ then every $x \in X$ satisfies $\sum_{i=1}^{n} k_i x_i > b$ and $y \in Y$ satisfies $\sum_{i=1}^{n} k_i y_i < b$.

Definition 1.5.4. Hyperplane: Let there exists scalars $w_1, w_2, ..., w_n$ then vectors $(y_1, y_2, ..., y_n)$ in $\mathcal{R}^n$ such that

$$w_1 y_1 + w_2 y_2 + ... + w_n y_n = b$$

b is some constant. This is called hyperplane.

Definition 1.5.5. Bounding Hyperplanes: The parallel hyperplanes drawn within their respective class of data. These are also said to be supporting hyperplanes that help in finding optimal hyperplane. The two bounding hyperplanes are given as $w^T y - b \geq +1$ and $w^T y - b \leq -1$.

Definition 1.5.6. Margin: A margin in support vector machine is said to be the distance between two parallel hyperplanes of two different classes.

Definition 1.5.7. Support Vectors: The vectors that lie on the bounding hyperplanes or within the margin are called support vectors. They play an important role in finding the optimal hyperplane. The circles marked red in the figure 1.1 are support vectors.

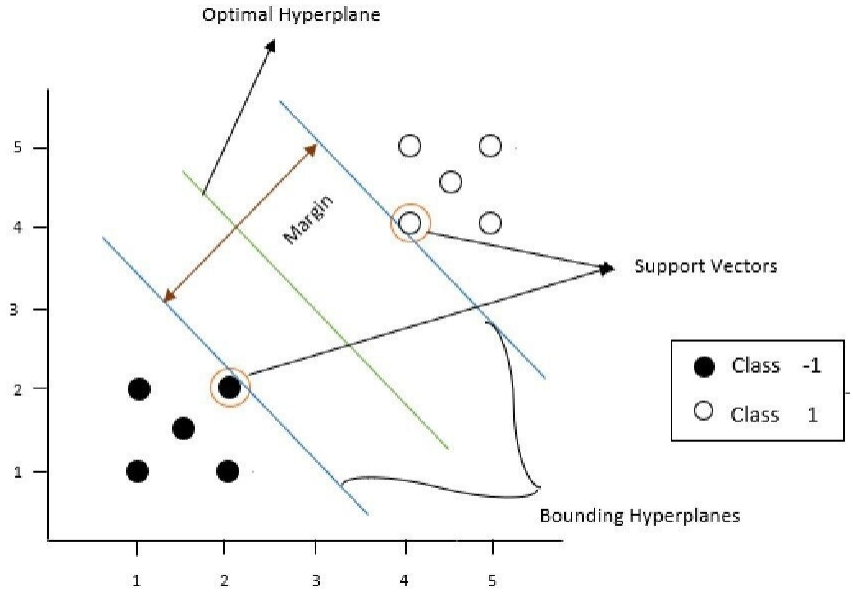

Figure 1.1

Definition 1.5.8. Sherman-Morrison-Woodbury identity: Let X be any square matrix which is invertible. Consider any two column vectors m and n $\in \mathcal{R}^n$. Then $X + mn^t$ is invertible if and only if $1 + n^T X^{-1} m \neq 0$. Then Sherman-Morrison-Woodbury identity is given as:

$$(X + mn^T)^{-1} = X^{-1} - \frac{X^{-1} m n^T X^{-1}}{1 + n^T X^{-1} m}$$

Definition 1.5.9. Input Space: The actual space where the data points lie is called the input space.

Definition 1.5.10. Feature Space: Feature space is the higher dimension space where the data is separated by non-linear classifier as it was not linearly separable in input space.

1.6 Machine Learning Models

1. **Decision Tree Based Model:** In this type of model, a tree is constructed from the data points from top to bottom. There are two components of a decision tree leaves and decision nodes. Nodes are created when the data splits and result is made on the basis of leaves. The decision is in the form of yes or no depends on the leaves of the particular query.

2. **K-Nearest Neighbor:** This is a classification model which learns from the data already labeled and then works on unlabeled data. It works on the theory of nearest neighbor point. To label an unlabel data point, it finds the closest labeled points to that unlabeled point. The decision is made on the basis of the label of the more close points to that unlabeled point. For example if more positive points are there then the unlabeled point belongs to positive class otherwise negative class.

3. **Perceptron Model:** Perceptron model came into existence by IBM. It is a binary classification technique and the classifier in this is known as **threshold function**. It is a function which maps its input value y to the output value g(y) which is given by :

$$g(y) = \begin{cases} 1 & \text{if } w^T y_j + b > 0 \\ 0 & \text{otherwise} \end{cases}$$

   Where w is the weight vector, b is the bias term and y_j is the data point where j = 1,2,...,n.
   The function g(y) generates two values 0 or 1 as per the positive and negative class. This algorithm does not terminate on non-linearly separable data but will give the results without classifying the non-linearly separable data points.

4. **Naive-Bayesian Network:** This type of model was developed to tackle large data sets. The name Naive suggests that it does not care about other features available if it has chosen particular feature already. It is basically used to calculate probabilities. The formula is given as:

$$P(M|N) = \frac{P(N|M)P(M)}{P(N)}$$

   Where P(M|N) is the probability of class M when predictor N is given, P(M) is the probability of class M separately, P(N|M) is the probability of predictor when class is given and P(N) is the probability of predictor.

   It is used in text classification and also used for multi-class problems. It is not considered as a good estimator as if the categories of some data is not given during training time it will mark the probability as zero and will not make decision. Also predictors are not available each time so it becomes difficult to assume predictors.

Chapter 2

Support Vector Machines (SVM)

2.1 Introduction

Support Vector Machines by Cortes and Vapnik in the year 1995[1] is considered as one of the best tools in machine learning for classification problems. In this algorithm a classifier is obtained to separate data from one class to another class. The classifier or the separating surface obtained can be linear or non-linear. There can be many classifiers for separating data points but we require the one which is optimal. The objective of this algorithm is to find an optimal hyperplane that maximizes the margin between two classes and to minimize the empirical classification error.

The binary classification problem i.e a two class problem is considered in this case and data points of each class is separated. The two bounding parallel hyperplanes are constructed for each class and then the hyperplane that maximizes the margin between these two bounding hyperplanes is the optimal hyperplane. The nature of statistical learning theory Vapnik [2] proposed that larger the margin between the two bounding hyperplanes, better the generalization error of the classifier.

2.2 Linear SVM

In this type of SVM, the data points are linearly separable that is we can easily classify these points with the help of a linear classifier. It is further divided into cases, separable and non-separable.

2.2.1 Separable Case (Hard-Margin Classifier)

Let $X = \{(y_1, p_1), (y_2, p_2).....(y_m, p_m)\}$ be the data set, $y_i \in \mathcal{R}^n$ and $p_i \in$ {-1,+1}
where y_i's are the data points that are separable and p_i's are the class labels. As per the theory proposed by Vapnik, a hyperplane of the form $w^T y - b = 0$ is found to separate these data points which will even classify the test data points. For this consider two bounding hyperplanes of the form:

$$w^T y_i - b \geq +1 \quad where \ p_i = +1 \tag{2.1}$$

$$w^T y_i - b \leq -1 \quad where \ p_i = -1 \tag{2.2}$$

where w is the coefficient vector of the hyperplane and b is called the bias term.

Equations (2.1) and (2.2) can be combined as follows:

$$p_i(w^T y_i - b) \geq 1 \quad where \quad i = 1, 2, ..., m \tag{2.3}$$

The perpendicular distance of $w^T y - b = +1$ from origin is $\frac{|\ -b-1\ |}{||w||}$ and the perpendicular distance of $w^T y - b = -1$ from origin is $\frac{|\ -b+1\ |}{||w||}$. The margin or distance between these two bounding hyperplanes is $\frac{2}{||w||}$.

To maximize margin i.e $\frac{2}{||w||}$ is same as minimizing margin i.e $\frac{1}{2}||w||^2$.
Now the new minimization problem is:

$$\min_{w,b} \quad \frac{1}{2}||w||^2$$

Subject to

$$p_i(w^T y_i - b) \geq 1 \quad where \quad y = 1, 2, ..., m \tag{2.4}$$

This optimization problem can be easily solved for the values of w and b with the help of **Lagrangian Dual** of equation (2.4) and solving:

$$= \max_{v} \big(\min\ L(w, v, b)\big)$$

The lagrangian is given as:

$$\min_{w,b} \; L(w,b,v) = \frac{1}{2}||w||^2 - \sum_{i=1}^{m} v_i[p_i(w^T y_i - b) - 1] \tag{2.5}$$

where $v_i's$ are Lagrange multipliers. For a given v, w and b can be easily found, that minimize L(w,b,v) and substitute to equation (2.5).

The **Karush-Kuhn-Tucker conditions(KKT)** [3,4] are applied that are necessary and sufficient to be satisfied by the optimal solution of the above optimization problem. The primal problem should satisfy the following conditions:

$$\begin{aligned}
\frac{\partial}{\partial w} L(w,b,v) &= 0 \\
\frac{\partial}{\partial b} L(w,b,v) &= 0 \\
p_i(w^T y_i - b) - 1 &\geq 0 \quad \forall i \\
v_i &\geq 0 \quad \forall i \\
v_i(p_i(w^T y_i - b) - 1) &= 0 \quad \forall i
\end{aligned}$$

Differentiating the Lagrangian with respect to primal variables and finding minimum value, it gives:

$$\frac{\partial}{\partial w} L(w,b,v) = 0 \Rightarrow w = \sum_{i=1}^{m} v_i p_i y_i \tag{2.6}$$

$$\frac{\partial}{\partial b} L(w,b,v) = 0 \Rightarrow \sum_{i=1}^{m} v_i p_i = 0 \tag{2.7}$$

Substituting value of $w = \sum_{i=1}^{m} v_i p_i y_i$ in equation (2.5)

$$\begin{aligned}
L(w,b,v) &= \frac{1}{2}||w||^2 - \sum_{i=1}^{m} v_i[p_i(w^T y_i - b) - 1] \\
&= \frac{1}{2}(w^T w) - \sum_{i=1}^{m} v_i[p_i(w^T y_i - b) - 1] \\
&= \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} v_i v_j p_i p_j (y_i^T y_j) - \sum_{i=1}^{m} v_i p_i (w^T y_i) + b\sum_{i=1}^{m} v_i p_i + \sum_{i=1}^{m} v_i
\end{aligned}$$

Substitute $\sum_{i=1}^{m} v_i p_i = 0$ from equation (2.7) in the above equation

$$\begin{aligned} L(w,b,v) &= \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} v_i v_j p_i p_j (y_i^T y_j) - \sum_{i=1}^{m}\sum_{j=1}^{m} v_i v_j p_j p_i (y_j^T y_i) + \sum_{i=1}^{m} v_i \\ &= \sum_{i=1}^{m} v_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} v_i v_j p_i p_j (y_i^T y_j) \end{aligned}$$

As w and b are vanished and the function is left with 'v' alone, let the function is:

$$\underset{v}{\text{Max}}\ L_D(v) = \sum_{i=1}^{m} v_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} v_i v_j p_i p_j (y_i^T y_j)$$

Subject to

$$\sum_{i=1}^{m} v_i p_i = 0,\ \ v_i \geq 0 \quad where \quad i = 1, 2, ..., m \tag{2.8}$$

The standard form of the above Quadratic Programming in equation (2.8) is given by:

$$\underset{v}{\text{Min}}\ L_D(v) = \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} v_i v_j p_i p_j (y_i^T y_j) - \sum_{i=1}^{m} v_i$$

Subject to

$$\sum_{i=1}^{m} v_i p_i = 0,\ \ v_i \geq 0 \quad where \quad i = 1, 2, ..., m \tag{2.9}$$

This Quadratic Programming gives values of $v_i's$ and a decision function is obtained as

$$f(y) = sign(w^T y - b) = sign\left\{ \sum_{i=1}^{m} v_i p_i y_i^T y - b \right\}$$

For each data point y_i we have a dual variable v_i. Those y_i that corresponds to non-zero $v_i's$ are called **support vectors**. w and b can be calculated for which value of $v \neq 0$.

This SVM formulation can search for the optimal hyperplane only for those data points which are linearly separable. What if the data points are not separable? Does this algorithm work for such data points? To deal with such data points they developed a new algorithm. We will discuss this in next section.

2.2.2 Algorithm

Algorithm 1 Matlab code for linearly separable data points

```
X = input('Enter the data:')
[m,n]= size(X)
d=input('Enter the class labels of data:')
D=diag(d)
H=(X*X').*(d*d')
y=-ones(m,1)
Aeq=d'
aeq=0
lb=zeros(m,1)
v=quadprog(H,y,[ ],[ ],Aeq,aeq,lb)
svin=find(v>target)
u=v(svin)
sv=X(svin,:)
```

2.3 Example

Let Y={(1,1),(2,1),(1,2),(2,2),(1.5,1.5),(4,4),(4,5),(5,4),(5,5),(4.5,4.5)} be the set of data points.
Let p={-1, -1, -1, -1, -1, 1, 1, 1, 1, 1} be the labels of the above data points.

P = diag(p)

$$P = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

The positive semi-definite matrix H for linear SVM can be calculated as:

$H = (Y * Y^T) .* (p * p^T)$, where (.*) represents the element wise multiplication.

$$H = PYY^TP = \begin{bmatrix} 2 & 3 & 3 & 4 & 3 & -8 & -9 & -9 & -10 & -9 \\ 3 & 5 & 4 & 6 & 4.5 & -12 & -13 & -14 & -15 & -13.5 \\ 3 & 4 & 5 & 6 & 4.5 & -12 & -14 & -13 & -15 & -13.5 \\ 4 & 6 & 6 & 8 & 6 & -16 & -18 & -18 & -20 & -18 \\ 3 & 4.5 & 4.5 & 6 & 4.5 & -12 & -13.5 & -13.5 & -15 & -13.5 \\ -8 & -9 & -9 & -10 & -9 & 32 & 36 & 36 & 40 & 36 \\ -12 & -13 & -14 & -15 & -13.5 & 36 & 41 & 40 & 45 & 40.5 \\ -12 & -14 & -13 & -15 & -13.5 & 36 & 40 & 41 & 45 & 40.5 \\ -16 & -18 & -18 & -20 & -18 & 40 & 45 & 45 & 50 & 45 \\ -12 & -13.5 & -13.5 & -15 & -13.5 & 36 & 40.5 & 40.5 & 45 & 40.5 \end{bmatrix}$$

Using the quadprog function in matlab we can easily calculate the values of lagrange multiplier 'v', which comes out to be:

$$v^T = \begin{bmatrix} 0 & 0 & 0 & 0.25 & 0 & 0.25 & 0 & 0 & 0 & 0 \end{bmatrix}$$

We can select the data points for the positive values of 'v'. These data points are support vectors, with this we can calculate 'w' and 'b'. From the above values of lagrange multiplier 'v', data points (2,2) and (4,4) are support vectors as 'v' is positive corresponding to these data points.
Now w can be calculated as:

$$w = \sum p_j v_j y_j \quad = -1 \times 0.25 \begin{bmatrix} 2 \\ 2 \end{bmatrix} + 1 \times 0.25 \begin{bmatrix} 4 \\ 4 \end{bmatrix} = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}$$

Similarly, b can be calculated as:

$$w_j y_j - b = 1 \quad \text{or}$$
$$w_j y_j - b = -1$$

as per the class label of the support vector. As (2,2) has class label as -1

$$b = 0.5 \times 2 + 0.5 \times 2 + 1 = 3$$

The required optimal hyperplane is :

$$0.5y_1 + 0.5y_2 - 3 = 0$$

2.3.1 Non-separable Case (Soft-Margin Classifier)

In the previous section the data points were separable, hyperplane found, classified data points easily without any misclassification. Now consider the case where the data points are not separable. In such case it is impossible to find a hyperplane that separates the data

points of one class from another class completely.

The possible solution is to find a hyperplane with maximum margin and less number of data points that lie in that wrong class. The abnormality of such data points from its respective bounding plane is called error. This is quite a contradicting task to find maximally separating hyperplane and minimum number of data points contributing to errors at same time. As the margin will be maximum there will be more chances for the data points with errors. A control parameter, **V** is introduced to control the weights given for these contradicting goals. The method formulated is termed as Soft-margin classifier [1].

L1 norm SVM

Consider a data set $X = \{(y_1, p_1), (y_2, p_2), ..., (y_m, p_m)\}$ with 'm' number of data points, where $y_i \in \mathcal{R}^n$ and $p_i \in \{$-1,+1$\}$. If all the points are not linearly separable, they allowed points to lie between bounding hyperplanes and beyond.

When a point from class label +1 lie either between bounding hyperplanes or into the region of class label -1, a positive quantity ξ_i generally termed as slack variable is added to the left hand side of the inequality to satisfy the constraint $w^T y_i - b \geq +1$, similarly when a point from class label -1 lie either between bounding hyperplanes or into the region of class label +1, a positive quantity ξ_i is subtracted from the left hand side of the inequality to satisfy the constraint $w^T y_i - b \leq -1$. Thus equations become $w^T y_i - b + \xi_i \geq +1$ and $w^T y_i - b - \xi_i \leq -1$. For rest of the points ξ_i is considered to be zero.

The formulation becomes:

$$\min_{w,b,\xi} \quad \frac{1}{2}||w||^2 + V \sum_{i=1}^{m} \xi_i$$

Subject to

$$\begin{aligned} p_i(w^T y_i - b) + \xi_i - 1 \geq 0, \quad & 1 \leq i \leq m \\ \xi_i \geq 0, \quad & 1 \leq i \leq m \end{aligned} \tag{2.10}$$

Where V is the control parameter whose value is given by the user while training and $\sum_{i=1}^{m} \xi_i$ is the sum of non-negative errors. The value of V is very important for good generalization power of the classifier. Mostly V is obtained by cross validation. It's value can be given according to the data points to be tested. A set of values is provided to the parameter V and accordingly the one that satisfies both the above mentioned contradicting conditions is chosen for the particular data set.

Minimization of $\frac{1}{2}||w||^2 + V\sum_{i=1}^{m}\xi_i$ will maximize the margin between bounding parallel planes and minimize the number of data points with errors.

The above primal problem is solved with the help of Lagrangian Dual.

$$= \max_{v}\big(\min\ L(w, v, \xi, b))$$

where lagrangian is defined as:

$$\min_{w,b,\xi}\ L(w, b, \xi, v, \alpha) = \frac{1}{2}||w||^2 + V\sum_{i=1}^{m}\xi_i - \sum_{i=1}^{m} v_i\left[p_i(w^T y_i - b) + \xi_i - 1\right] - \sum_{i=1}^{m}\alpha_i\xi_i$$

$$= \frac{1}{2}w^T w + \sum_{i=1}^{m}(V - v_i - \alpha_i)\xi_i - \left\{\sum_{i=1}^{m} v_i p_i y_i^T\right\}w - \left\{\sum_{i=1}^{m} v_i p_i\right\}b + \sum_{i=1}^{m} v_i \tag{2.11}$$

where v, α are the lagrange multipliers.

Differentiating the lagrangian and finding minimum value:

$$\frac{\partial}{\partial w}L(w, b, \xi, v, \alpha) = 0 \Rightarrow w = \sum_{i=1}^{m} v_i p_i y_i \tag{2.12}$$

$$\frac{\partial}{\partial b}L(w, b, \xi, v, \alpha) = 0 \Rightarrow \sum_{i=1}^{m} v_i p_i = 0 \tag{2.13}$$

$$\frac{\partial}{\partial \xi_i}L(w, b, \xi, v, \alpha) = 0 \Rightarrow V - v_i - \alpha_i = 0, \quad 1 \le i \le m \tag{2.14}$$

As $v > 0$ and $m > 0$ it leads to:

$$0 \le v_i \le V \tag{2.15}$$

Substituting the values of equations (2.12), (2.13) and (2.14) in equation (2.11):

$$\begin{aligned} L(w, b, \xi, v, \alpha) &= \frac{1}{2}w^T w + \sum_{i=1}^{m} 0 \times \xi_i - w^T w - 0 \times b + \sum_{i=1}^{m} v_i \\ &= -\frac{1}{2}w^T w + \sum_{i=1}^{m} v_i \\ &= \sum_{i=1}^{m} v_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} p_i p_j y_i^T y_j v_i v_j \end{aligned}$$

The dual of the problem is:

$$\underset{v}{\text{Max}}\ L_D(v) = \sum_{i=1}^{m} v_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} p_i p_j y_i^T y_j v_i v_j$$

Subject to

$$\sum_{i=1}^{m} v_i p_i = 0, \quad 0 \leq v_i \leq V \quad 1 \leq i \leq m \tag{2.16}$$

In quadratic programming the above maximization problem is converted to minimization as standard form:

$$\underset{v}{\text{Min}}\ L_D(v) = \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} p_i p_j y_i^T y_j v_i v_j - \sum_{i=1}^{m} v_i$$

Subject to

$$\sum_{i=1}^{m} v_i p_i = 0, \quad 0 \leq v_i \leq V \quad 1 \leq i \leq m \tag{2.17}$$

In Soft-margin classifier the lagrange multipliers $v_i's$ are bounded equation (2.17). The following cases arise for $v_i's$:

1. When $v_i = 0$ then $\xi_i = 0$ which means $y_i's$ corresponding to such $v_i's$ is correctly classified.

2. When $0 < v_i < V$ then $p_i(w^T y_i - b) + \xi_i = 1$ and $\xi_i = 0$, such data points are unbounded support vectors. These data points lie on the bounding hyperplane of their class.

3. When $v_i = V$ then $p_i(w^T y_i - b) + \xi_i = 1$ and $\xi_i > 0$, such data points are bounded support vectors. These data points lie beyond their respective hyperplane.

4. When $0 \leq \xi_i < 1$ then $y_i's$ are correctly classified, if $\xi_i \geq 1$ then $y_i's$ are misclassified.

2.3.2 Algorithm

Algorithm 2 Matlab code for non-separable data points **(L1-SVM)**

```
X=input('Enter the data:')
[m,n]=size(X)
d=input('Enter the class labels of data:')
D=diag(d)
V=2
target=0.0001
H=(X*X').*(d*d')
y=-ones(m,1)
Aeq= d'
aeq= 0
lb=zeros(m,1)
ub=V*ones(m,1)
v=quadprog(H,y,[ ],[ ],Aeq,aeq,lb,ub)
svin=find(v>target)
svmin=find((v>target)&(v<(V-target)))
u=v(svmin,:)
sv=[X(svmin,:)]
```

2.4 Example

Let Y={(1,0.8),(3,2.5),(2.5,1),(1,1.8),(3,4.5),(2.5,2.8)} be the set of data points.
Let p={1, 1, 1, -1, -1, -1} be the labels of the above data points.

V is the control parameter which is taken as 2. Upper bound in non-separable case depends upon the value of V which comes out to be vector with each value 2 as given below:

$$UB = \begin{bmatrix} 2 \\ 2 \\ 2 \\ 2 \\ 2 \\ 2 \end{bmatrix}$$

P = diag(p)

$$P = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix}$$

The positive semi-definite matrix H can be calculated as:

$H = (Y * Y^T) .* (p * p^T)$, where (.*) represents the element wise multiplication.

$$H = PYY^TP = \begin{bmatrix} 1.64 & 5 & 3.3 & -2.44 & -6.6 & -4.74 \\ 5 & 15.25 & 10 & -7.5 & -20.25 & -14.5 \\ 3.3 & 10 & 7.25 & -4.3 & -12 & -9.05 \\ -2.44 & -7.5 & -4.3 & 4.24 & 11.1 & 7.54 \\ -6.6 & -20.25 & -12 & 11.1 & 29.25 & 20.10 \\ -4.74 & -14.45 & -9.05 & 7.54 & 20.10 & 14.09 \end{bmatrix}$$

Using the quadprog function in matlab we can easily calculate the values of lagrange multiplier 'v', which comes out to be:

$$v^T = \begin{bmatrix} 1.6222 & 2 & 7.1531 & 1.8444 & 4.2731 & 1.7778 \end{bmatrix}$$

0 < v < V, corresponds to the support vectors falling on the bounding plane which are (1,0.8), (1,1.8) and (2.5,2.8). We can find 'w' and 'b' with the help of these support vectors. w comes out to be:

$$w = \begin{bmatrix} 1.3333 \\ -2 \end{bmatrix}$$

Similarly, b can be calculated as:

$$w_j y_j - b = 1 \quad \text{or}$$
$$w_j y_j - b = -1$$

as per the class label of the support vector. As (1,0.8) has class label as 1

$$b = 1.3333 \times 1 - 2 \times 0.8 - 1 = -1.2667$$

The required optimal hyperplane is :

$$1.333y_1 - 2y_2 + 1.2667 = 0$$

L2-norm SVM

In L2 norm SVM, the sum of squares of slack variables are minimized along with the margin term. The formulation becomes:

$$\min_{w,b,\xi} \quad \frac{1}{2}||w||^2 + \frac{V}{2}\sum_{i=1}^{m}\xi_i^2$$

Subject to

$$\begin{aligned} p_i(w^T y_i - b) + \xi_i - 1 \geq 0, \quad & 1 \leq i \leq m \\ \xi_i \geq 0, \quad & 1 \leq i \leq m \end{aligned} \tag{2.18}$$

This optimization problem is solved with the help of Lagrange multipliers and applying KKT conditions it gives:

$$\frac{\partial}{\partial w}L(w,b,\xi,v,\alpha) = 0 \Rightarrow w = \sum_{i=1}^{m} v_i p_i y_i \tag{2.19}$$

$$\frac{\partial}{\partial b}L(w,b,\xi,v,\alpha) = 0 \Rightarrow \sum_{i=1}^{m} v_i p_i = 0 \tag{2.20}$$

$$\frac{\partial}{\partial \xi_i}L(w,b,\xi,v,\alpha) = 0 \Rightarrow V\xi_i - \alpha_i = 0, \quad 1 \leq i \leq m \tag{2.21}$$

Substituting the values the final formulation of quadratic programming is formed as:

$$\min_{v} \; L_D(v) = \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} v_i v_j p_i p_j (y_i^T y_j + \frac{\mu_{ij}}{V}) - \sum_{i=1}^{m} v_i$$

Subject to

$$\begin{aligned} &\sum_{i=1}^{m} v_i p_i = 0 \\ &v_i \geq 0 \quad 1 \leq i \leq m \end{aligned} \tag{2.22}$$

Thus from here values of v_i can be calculated. For $v_i > 0$, y_i is a support vector and from here b can be easily calculated by substituting values. Therefore b comes out to be:

$$b = \sum_{j=1}^{m} v_i p_j \left(y_j^T y_i + \frac{\mu_{ij}}{V}) \right) - p_i \tag{2.23}$$

2.4.1 Algorithm

Algorithm 3 Matlab code for non-separable data points **(L2-SVM)**

```
X=input('Enter the data:')
[m,n]=size(X)
d=input('Enter the class labels of data:')
d=diag(d)
V=2
target=0.0001
I=diag(ones(m,1)*(1/V))
H=(X*X'+I).*(d*d')
y=-ones(10,1)
Aeq= d'
aeq= 0
lb=zeros(m,1)
ub=V*ones(m,1)
u1=zeros(m,1) v=quadprog(H,y,[ ],[ ],Aeq,aeq,lb,ub,u1)
svin=find(v>target)
u=v(svin,:)
sv=[X(svin,:)]
```

2.5 Non-linear SVM (Kernel Methods)

Until now, we have discussed the case where the data was linear. So to make the SVM formulation more general, data points are mapped from original space (input space) into higher dimensional space (feature space) and to find maximally separating hyperplane in that space. In linear SVM formulation using Lagrangian transformation the training data appears as the dot product $(y_i^T y_j)$ (see equation 2.10). This proves that all the information required for training can be obtained from dot product of training vectors.

A mapping $\phi(.)$ can be used to map each data point y_i to the higher dimensional space. So the SVM formulation for non-linear kernels require the dot product of $\phi(y_i)^T \phi(y_j)$ instead of dot product of $y_i^T y_j$ for all i and j. When the dimension of data is high or data points are large

in number, huge computation is required to map dot product in the higher dimensional space. If a **Kernel-function** k is used to define the dot product i.e. $k(y_i, y_j) = \phi(y_i)^T \phi(y_j)$ [5,6] in higher dimension then only the value of k is required to be computed without explicitly mapping y_i and y_j into higher dimensional space.

Replacing the dot product in the input space with the kernel function and the formulation becomes:

$$\underset{v}{\text{Min}} \; L_D(u) = \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} p_i p_j k(y_i, y_j) v_i v_j - \sum_{i=1}^{m} v_i$$

Subject to

$$\sum_{i=1}^{m} v_i p_i = 0, \quad 0 \leq v_i \leq V \quad 1 \leq i \leq m \tag{2.24}$$

2.5.1 Kernel Functions

The kernel function $k(y_i, y_j)$ is a function in the input space. Using kernel function the mapping $\phi(.)$ is avoided and only value of kernel k is computed to find the dot product in feature space. Therefore, one can formulate SVM and find maximally separating non-linear classifier by using the kernel function $k(y_i, y_j)$. Any kernel defined should satisfy the **Mercer's Conditions** [7] which states that the kernel matrix $k = (k(y_i, y_j))_{i,j=1}^{m}$ must be positive semi definite i.e. it has all non-negative eigen values. Kernels can be of the following types:

1. Linear kernel: $k(y_i, y_j) = (y_i)^T (y_j)$
2. Polynomial Kernel: $k(y_i, y_j) = (y_i^T y_j + 1)^p$
3. Radial basis Kernel: $k(a, b) = exp(-\sigma(||a - b||^2)$
4. Gaussian Kernel: $k(a, b) = exp(-\frac{||a - b||^2}{2\sigma^2})$

2.6 Drawbacks of SVM

1. The training of data points involves solving of dual problem. The order of complexity of the worst case for m data points is $O(m^3)$. Thus when there is large data set, large memory and more time is consumed.
2. The control parameter and kernel functions play an important role in the generalization of SVM. The value of control parameter is obtained by cross validation through training and testing repeatedly. This consumes time.

Chapter 3

Lagrangian Support Vector Machines(LSVM)

3.1 Introduction

Classification can be achieved through linear as well as non-linear separating surfaces by one of the powerful algorithms, Support vector machines [1,8,9,10]. In this chapter we have reviewed the work of Mangasarian and Musicant[11]. Further, they developed a very fast simple algorithm based on implicit Lagrangian formulation [12] and reformed the standard quadratic program for linear support vector machines into an unconstrained differentiable convex function. The SVM we studied earlier was based on quadratic programming that was both memory and time consuming. The algorithm Mangasarian and Musicant[11] proposed is fast and requires inverse of a positive definite matrix by Sherman-Morrison-Woodbury (SMW) identity [13] with good generalization.

This chapter is further divided into five sections. The first section includes the formulation of Lagrangian Support Vector Machine (LSVM) from the standard SVM. The objective of SVM was to maximize the margin between the bounding hyperplanes with respect to the orientation (w). Mangasarian and Musicant [14] in Active Set Support Vector Machine (ASVM) made changes to standard linear SVM by changing the objective function. Further, Mangasarian and Musicant [14] maximized the margin between the bounding hyperplanes with respect to both orientation (w) as well as location (b) relative to the origin. The convex optimization problem in LSVM thus takes into account both w and b. This lowers the dimension of the kernel matrix $(n+1)\times(n+1)$ that was $m\times m$ in SVM and further linearly iterative scheme is discussed that needs no optimization package.

The second section involves the algorithm for solving the unconstrained minimization problem. This is an iterative method and the best part of this algorithm is, the convergence is obtained from any starting point. In the third section LSVM is further extended for solving non-linear classification problem using kernel method. Fourth section includes the algorithm for non-linear kernel LSVM. Comparisons between SVM and LSVM on the basis of training, testing, time and memory is made in the fifth and final section.

3.2 Linear LSVM

A data set of 'm' data points in $\mathcal{R}^n$ is taken into consideration which is represented by matrix X of dimension $m \times n$. The class labels X+ and X- are given by the diagonal matrix P. For the formulation of LSVM, the standard support vector machine [1] quadratic program with parameter C > 0 as discussed in chapter 2 is given by:

$$\min_{(w,b,\xi)\in\mathcal{R}^{n+1+m}} \frac{1}{2}w^T w + Ce^T\xi$$

Subject to:

$$\begin{aligned} P(Xw - eb) + \xi &\geq e \\ \xi &\geq 0 \end{aligned} \tag{3.1}$$

where $\dfrac{2}{||w||}$ is the margin between two bounding hyperplanes and w is the normal to bounding hyperplanes given by:

$$\begin{aligned} X_i w - b + \xi &\geq +1 \quad for \quad P_{ii} = +1 \\ X_i w - b - \xi &\leq -1 \quad for \quad P_{ii} = -1 \end{aligned} \tag{3.2}$$

The equation (3.1) is solved by using Lagrangian Dual:

$$\min_v \; L = \frac{1}{2}w^T w + Ce^T\xi - v^T\Big[P(Xw - eb) + \xi - e\Big] - \mu^T\xi$$

Where v and μ are lagrange multipliers.

Differentiating L with respect to primal variables, it gives:

$$\begin{aligned} &\frac{\partial}{\partial w}L = 0 \Rightarrow w - X^T Pv = 0 \\ &\frac{\partial}{\partial b}L = 0 \Rightarrow e^T Pv = 0 \\ &\frac{\partial}{\partial \xi}L = 0 \Rightarrow Ce^T - v^T - \mu^T = 0 \end{aligned}$$

Substituting values of the above partial derivatives in L:

$$\begin{aligned}\max_{v}\ L &= \frac{1}{2}(X^TPv)^T(X^TPv) + (Ce^T - v^T - \mu^T)\xi - v^T\Big[P\Big(XX^TPv - eb - e\Big)\Big] \\ &= \frac{1}{2}(v^TPXX^TPv) - v^TPXX^TPv + bv^TPe + v^Te \\ &= -\frac{1}{2}(v^TPXX^TPv) + v^Te\end{aligned}$$

The optimal hyperplane can be obtained by the solution of the dual problem:

$$\min_{v\in\mathcal{R}^m}\frac{1}{2}v^TPXX^TPv - e^Tv$$

Subject to:

$$\begin{aligned}&e^TPv = 0 \\ &0 \le v \le Ce\end{aligned} \tag{3.3}$$

Further, Mangasarian and Musicant noticed that the matrix in the dual problem in equation (3.3) given by PXX^TD is not positive definite in general as m $>>$ n. The equality constraints with the bounded constraints are difficult to deal with when there is large data set. To overcome such difficulties and instead of inverting a very large matrix of order $m\times m$, Mangasarian and Musicant [11] proposed the following modifications and then changed L1-norm to L2-norm squared which turns $\xi \ge 0$ redundant. They maximized the margin between the bounding hyperplanes by optimizing with respect to both w and b.

We get the following optimization problem :

$$\min_{(w,b,\xi)\in\mathcal{R}^{n+1+m}}\frac{1}{2}(w^Tw + b^2) + C\frac{\xi^T\xi}{2}$$

Subject to:

$$P(Xw - eb) + \xi \ge e \tag{3.4}$$

The above problem is solved with the help of Lagrange multipliers and using the KKT conditions:

$$\min_{w,b,\xi}\ L = \frac{1}{2}(w^Tw + b^2) + C\frac{\xi^T\xi}{2} - v^T\Bigg[P(Xw - eb) + \xi - e\Bigg] \tag{3.5}$$

where v is the lagrange multiplier and now applying KKT conditions it gives:

$$\frac{\partial}{\partial w}L = 0 \Rightarrow w - X^TPv = 0 \tag{3.6}$$

$$\frac{\partial}{\partial b}L = 0 \Rightarrow b + e^T Pv = 0 \tag{3.7}$$

$$\frac{\partial}{\partial \xi}L = 0 \Rightarrow C\xi - v = 0 \tag{3.8}$$

Substituting the values in the equation (3.5):

$$\begin{aligned}
L &= \frac{1}{2}\Big[(X^T Pv)^T (A^T Pv) + (-e^T Pv)^2\Big] + \frac{C}{2}\Big(\frac{v}{C}\Big)^T \frac{v}{C} - v^T\Big[P\Big(XX^T Pv - e(-e^T Pv)\Big) + \frac{v}{C} - e\Big] \\
&= \frac{1}{2}v^T PXX^T Pv + \frac{1}{2}(e^T Pv)^2 + \frac{v^T v}{2C} - v^T PXX^T Pv - v^T ee^T PPv - \frac{v^T v}{C} + v^T e \\
&= -\frac{1}{2}v^T PXX^T Pv + \frac{1}{2}v^T ee^T PPv + \frac{v^T v}{2C} - v^T ee^T PPv - \frac{v^T v}{C} + v^T e \\
&= -\frac{1}{2}v^T DAA^T Pv - \frac{1}{2}v^T ee^T PPv - \frac{v^T v}{2C} + v^T e
\end{aligned}$$

The dual optimization thus becomes:

$$\max_{0 \le v \in \mathcal{R}^m} \; L_P = -\frac{1}{2}v^T PXX^T Pv - \frac{1}{2}v^T ee^T PPv - \frac{v^T v}{2C} + v^T e \tag{3.9}$$

The standard form of the above dual problem is:

$$\min_{0 \le v \in \mathcal{R}^m} \; \frac{1}{2}v^T\Big[\frac{I}{C} + P\Big(XX^T + ee^T\Big)P\Big]v - e^T v \tag{3.10}$$

The matrix in the equation (3.10) is positive definite with no equality constraint. The non-negative constraint $v \geq 0$ is present in the above problem. This helped to develop simple iterative Lagrangian support vector machine which uses the inverse of positive definite $(n+1) \times (n+1)$ matrix.

In order to construct the iterative method for LSVM the two matrices were defined for simplification:

$$S = P\begin{bmatrix} X & -e \end{bmatrix}, \quad R = \frac{I}{C} + SS^T \tag{3.11}$$

The equation (3.10) thus becomes:

$$\min_{0 \le v \in \mathcal{R}^m} \; \frac{1}{2}vTRv - e^T v \tag{3.12}$$

R^{-1} is computed using the SMW identity, therefore only $(n+1) \times (n+1)$ matrix is inverted. LSVM relies on Karush-Kuhn-Tucker optimality conditions O.L. Mangasarian [15] for the

dual problem (3.12):

$$0 \leq v \perp Rv - e \geq 0 \tag{3.13}$$

The optimality condition can be further transformed using the identity between any two real numbers as:

$$\begin{aligned} < x, y - \nu y > &= < x, x > - < x, \nu y > \\ &= 1 - \nu < x, y > \\ &= 1 - 0 \\ &= 1 \end{aligned}$$

The inner product came out to be **1** which implies:

$$0 \leq x \perp y \geq 0 \Longleftrightarrow x = (x - \nu y)_+, \quad \nu \geq 0 \tag{3.14}$$

Therefore the equation (3.13) becomes

$$Rv - e = ((Rv - e) - \nu v)_+ \tag{3.15}$$

Simplifying the above equation in order to construct iterative scheme:

$$\begin{aligned} Rv &= e + ((Rv - e) - \nu v)_+ \\ v &= R^{-1}(e + ((Rv - e) - \nu v)_+) \end{aligned}$$

Thus the iterative method of LSVM is thus given by:

$$v^{i+1} = R^{-1}(e + ((Rv^i - e) - \nu v^i)_+), \quad i = 0, 1, ... \tag{3.16}$$

which should satisfy the following condition:

$$0 < \nu < \frac{2}{C} \tag{3.17}$$

The necessary and sufficient condition in the equation (3.15) satisfied by the implicit Lagrangian Mangasarian and Solodov [12] given as

$$\min_v \ L(v, \nu) = \min_{v \geq 0} \ \frac{1}{2} v^T Rv - e^T v + \frac{1}{2\nu} \Big(||\big(- \nu v + Rv - e \big)_+||^2 - ||Rv - e||^2 \Big) \tag{3.18}$$

associated with the dual equation (3.12).

Differentiating the above Lagrangian to zero and setting gradient with respect to v:

$$(Rv - e) + \frac{1}{\nu}(R - \nu I)((R - \nu I)v - e)_+ - \frac{1}{\nu} R(Rv - e) = 0 \tag{3.19}$$

Further solving the above equation:

$$\begin{aligned}(Rv-e)\Big[I-\frac{R}{\nu}\Big]+\frac{1}{\nu}(R-\nu I)((R-\nu I)v-e)_+=0\\ \frac{1}{\nu}(Ru-e)(\nu I-R)+\frac{1}{\nu}(R-\nu I)((R-\nu I)v-e)_+=0\\ \frac{1}{\nu}(\nu I-R)((Rv-e)-((R-\nu I)v-e)_+)=0\end{aligned}$$

Therefore equation (3.19) is equivalent to:

$$(\nu I-R)((Rv-e)-((R-\nu I)v-e)_+)=0 \tag{3.20}$$

The above equation is equivalent to the optimality condition (3.15) with the assumption that ν is positive and it is not the eigen value of R.

3.3 Algorithm for linear LSVM and It's global Convergence

Define a symmetric positive definite matrix $R \in \mathcal{R}^{m\times m}$ see equation (3.11) and suppose equation (3.17) holds. Consider any arbitrary starting point v^0 $in\mathcal{R}^m$, let $\bar{v}$ be the unique solution of equation (3.12), therefore, it satisfies the optimality condition (see equation (3.15) for any $\nu > 0$.

Now subtract $R\bar{v}$ from iterative equation (3.16) and take norm it gives:

$$||Rv^{i+1}-R\bar{v}|| = ||(Rv^i-e-\nu v^i)_+-(R\bar{v}-e-\nu\bar{v})_+|| \tag{3.21}$$

With the help of Projection Theorem [16] which states that the distance between any two points in $\mathcal{R}^m$, the above equation becomes:

$$\begin{aligned}||Rv^{i+1}-R\bar{v}|| &\le ||(R-\nu I)(v^i-\bar{v})||\\ &\le ||I-\nu R^{-1}||.||R(v^i-\bar{v})||\end{aligned} \tag{3.22}$$

This shows that the iterates v^i equation (3.16) converges to a unique solution $\bar{v}$ of equation (3.12) at the linear rate.

Further to prove $||I-\nu IR^{-1}|| \le 1$. Starting from equation (3.17) and definition of matrix R. Let a_i = 1,2,.....,m are non-negative eigen values of SS^T this follows:

$$\begin{aligned}-1 < 1-\nu\Big(\frac{1}{C}+a_i\Big)^{-1} < 1\\ 2 > \nu\Big(\frac{1}{C}+a_i\Big)^{-1} > 0\end{aligned} \tag{3.23}$$

The above equation is satisfied under equation (3.17).

In the next section LSVM matlab code for linear data set is proposed which can solve problems with millions of data points. imax, which refers to the maximum number of iterations, tl, which refers to the tolerance error, matrix X, D and parameter C is given as input. To avoid the violation of optimality condition equation (3.15) $||v^{i+1} - v^i||$ bounds from above:

$$||R||^{-1}.||Rv^i - e - ((Rv^i - e) - \nu v_+^i|| \tag{3.24}$$

Mangasarian and Ren [17] stated that $||v^{i+1} - v^i||$ bounds $||v^i - \bar{v}||$ and from equation (3.6) and (3.7) it also bounds $||w^i - \bar{w}||$ as well as $||b^i - \bar{b}||$, where $\bar{w}, \bar{b}, \bar{\xi}$ is the unique solution of primal problem (3.5).

3.3.1 Algorithm

Algorithm 4 Matlab code for Linear LSVM

```
function[i,p,w,b]=LSVM1(X,P,C,imax,tl)
[m,n ]=size(X)
ν=1.9/C
g=ones(m,1)
S=P*[X -e]
i=0
R=S*inv((speye(n+1)/C+S'*S))
v=C*(1-R*(S'*g))
v1=v+1
while i<imax & norm(v1 - v)>tl do
r=(1+q(((v/C+S*(S'*v)) - ν*v) - 1))
v1=v
v=C*(r-R*(S'*r))
i=i+1
end while
p=norm(v-v1)
w=X'*P*v
b=-g'*P*v
function q=q(y)
q=(abs(y)+y)/2
```

3.4 Non-Linear LSVM using Kernel Tricks

This section consists of LSVM for non-linear kernels. The LSVM for linear classification is extended to construct the formulation for non-linear kernels. The only modification made is finding inverse using Sherman-Morrison-Woodbury (SMW) if the dot product of the kernel is known, which are not known in general. Therefore, they developed simple algorithm which does not make use of SMW identity.

Consider matrix $X \in \mathcal{R}^{m\times n}$ and $Y \in \mathcal{R}^{n\times l}$ and let kernel k (X,Y) maps into $\mathcal{R}^{m\times n} \times \mathcal{R}^{n\times l}$ into $\mathcal{R}^{m\times l}$. Let x be the column vector in $\mathcal{R}^n$, related to this row vector is $k(x^T, X^T)$. The linear separating surface in linear LSVM is replaced by non-linear surface:

$$k(\begin{bmatrix} x^T & -1 \end{bmatrix}, \begin{bmatrix} X^T \\ -e^T \end{bmatrix})Pv = 0 \tag{3.25}$$

where v is the solution of equation (3.12) and R is defined as:

$$M = \begin{bmatrix} X & -e \end{bmatrix}, \quad R = \frac{I}{C} + Pk(M, M^T)P \tag{3.26}$$

The non-linear classifier(3.25) can be converted into linear simply by letting kernel $k(M, M^T) = MM^T$.

Rewriting the equation (3.10) the dual optimization problem for linear kernel is:

$$\min_{0\leq v\in\mathcal{R}^m} \ \frac{1}{2}v^T\Big[\frac{I}{C} + PMM^TP\Big]v - e^Tv \tag{3.27}$$

Replacing the linear kernel by non-linear kernel $k(M, M^T)$ the above equation becomes:

$$\min_{0\leq v\in\mathcal{R}^m} \ \frac{1}{2}v^T\Big[\frac{I}{C} + Pk(M, M^T)P\Big]v - e^Tv \tag{3.28}$$

The KKT conditions for the given problem is:

$$0 \leq v \perp \Big(\frac{I}{C} + Pk(\begin{bmatrix} X & -e \end{bmatrix}, \begin{bmatrix} X^T \\ -e^T \end{bmatrix})P\Big)v - e \geq 0 \tag{3.29}$$

The solution of equations (3.28) and (3.29) exists if the non-linear kernel $k(M, M^T)$ is positive semidefinite.

The algorithm of LSVM for non-linear kernel is given in the next section. This algorithm involves iterative method for solving non-linear classification but it does not use Sherman-Morrison-Woodbury identity.

3.4.1 Algorithm

Algorithm 5 Matlab code for Non-Linear LSVM using Kernel

function[i,p,v]LSVMK(C,imax,=tl,P,K) [m,n]=size(K,1)
ν=1.9/C
g=ones(m,1)
I=speye(m)
i=0
R=I/C+P*K*P
N=inv(P)
v=P*e
v1=v+1
while i<imax & norm(v1 - v)>tl **do**
v1=v v=N*(1+q(R*v-1-b*v))
i=i+1
end while
p=norm(v-v1); [i p]
function q=q(y)
q=(abs(y)+y)/2

3.5 Comparison between SVM and LSVM

Comparison between LSVM and SVM				
Datasets	**Size of datasets**	**Algorithm**	**Testing Accuracy**	**Time**
Soy Beans	208 x 61	PlainSVM	96.0784	0.0456
		L1-SVM	96.0784	0.0625
		L2-SVM	96.0784	0.0469
		LSVM	94.7712	0.0314
Horse	368 x 24	PlainSVM	61.4130	0.6406
		L1-SVM	61.4130	0.0369
		L2-SVM	61.4130	0.2513
		LSVM	61.4130	0.4688
Heart	270 x 14	PlainSVM	67.4074	0.6994
		L1-SVM	80	0.0469
		L2-SVM	79.2593	0.3438
		LSVM	82.222	0.5000
Pima	768 x 9	PlainSVM	65.3646	0.6031
		L1-SVM	65.3646	0.1094
		L2-SVM	67.0104	0.1406
		LSVM	68.4896	0.6563
Sonar Cells	208 x 61	PlainSVM	74.0784	0.0563
		L1-SVM	75	0.0625
		L2-SVM	75.9615	0.5938
		LSVM	77.8462	0.7188
Parkinson	195 x 23	PlainSVM	74.2268	0.5156
		L1-SVM	77.3196	0.001
		L2-SVM	86.5979	0.0156
		LSVM	86.5979	0.0156
Liver Disorder	345 x 7	PlainSVM	69.5637	0.6575
		L1-SVM	69.1860	0.0469
		L2-SVM	69.1860	0.3750
		LSVM	74.4419	0.6094
Mushroom	8124 x 23	PlainSVM	86.8784	25.3125
		L1-SVM	95.9626	16.6250
		L2-SVM	96.4549	14.9375
		LSVM	94.7809	6.5781

Chapter 4

Multi-class Lagrangian Support Vector Machines

4.1 Introduction

Support Vector Machines is one of the best tools used to classify binary classification problem. It has been designed by Cortes and Vapnik [2] and its objective is to maximize the distance between two bounding hyperplanes of two different classes. SVM algorithm uses quadratic programming for finding the optimal hyperplane which we discussed in Chapter 2. Although it is an easy approach but it consumes large memory and takes more processing time, therefore, a new approach called LSVM, Lagrangian Support Vector Machines [11] came into existence.

LSVM is an iterative scheme which is more effective and does not use entire training data. In this algorithm inverse of positive definite kernel matrix using SMW Sherman-Morrison-Woodbury identity is calculated. Further, some changes are applied to the objective function which lowers the dimension of the kernel matrix. Therefore, LSVM is more effective and consumes less memory.

Till now we have discussed about the classification of two-classes i.e. binary classification problem. SVM also helped in solving multi-class classification problem directly or indirectly. Two approaches has been used to deal with multi-class problems that are one-against-one (OAO) and one-against-all (OAA) using LSVM which was proposed by Duan et al. [18]. To find the optimal hyperplane this approach considered one class at a time. There was no such method available which considers all the classes simultaneously until J. P. Hwang, B.

Choi, I. W. Hong, and E. Kim, [19] proposed a new multi-class LSVM approach known as LSVMMPAC. In such an approach, LSVM is formulated as single optimization problem and support vectors are chosen simultaneously. To solve the optimization problem, multi-class output representation matrix (MORM) is defined.

This chapter is further organized into three sections. Section one represents Support Vector Machine for multi-class classification problems. In this section two method for solving multi-class problem that are one-against-one (OAO) and one-against-all (OAA)[20]. In section three new LSVM method for solving multi-class problem is derived. In this method, all the classes are considered at once. Further the matrix representation of the optimization problem. Section four includes the feature and comparison of LSVMMPAC with other methods is made. In the last section this method is tested on some data sets from UPI repository.

4.2 Support Vector Machines for Multi-class

In this section of the paper firstly SVM and LSVM for binary classification is discussed for revision purpose. For the data set $X = \{(y_1, p_1), (y_2, p_2).....(y_N, p_N)\}$ is considered where $y_j's$ are the data points and $p_j's$ are the class labels belongs to $\{-1, 1\}$. Therefore, the plain SVM formulation is given by:

$$\begin{aligned} &\min_{w,b,\nu} \frac{1}{2} w^T w + V 1^T \nu \\ &\text{Subject to } P(Yw + b\mathbf{1}) \geq 1 - \nu \\ &\nu \geq 0 \quad \text{j} = \{1, ..., N\} \end{aligned} \tag{4.1}$$

where ν is the slack variable

Similarly, LSVM formulation is designed as:

$$\begin{aligned} &\min_{w,b,\nu} \frac{1}{2} w^T w + \frac{1}{2} b^2 + V \nu^T \nu \\ &\text{Subject to } P(Yw + b\mathbf{1}) \geq 1 - \nu \\ &\nu \geq 0 \quad \text{j} = \{1, ..., N\} \end{aligned} \tag{4.2}$$

These formulations are for the binary classification problems. Further, these formulations are extended to solve multi-class problems by making certain changes and introducing new terms.

A data set is considered $X = \{(y_1, p_1), (y_2, p_2).....(y_N, p_N)\}$ where $y_j's$ are data points and $p_j's$ are the class labels of the data points which are {1,2,3,....,k}. The first method for solving multi-class problem is (OAO) i.e. One-Against-One [20]. In this method pairs of two classes are formed and then the classifier is found. This results in the formulation of k(k-1)/2 classifiers. The votes are counted to know the class of a data point, one with maximum number

of votes is chosen as the class for that data point. One classifier votes for one class of the two classes paired.

The classifier between the two classes is defined as:

$$g_{j,i}(y) = \begin{cases} j & \text{if } w_{j,i}^T y + b_{j,i} < 0 \\ i & \text{otherwise} \end{cases} \tag{4.3}$$

The function to calculate votes is defined as:

$$f(y, j, i, K) = \begin{cases} 1 & \text{if } g_{j,i}^T y = K \\ 0 & \text{otherwise} \end{cases} \tag{4.4}$$

The decision function to determine the class of the data point is given by:

$$S_{OAO} = \arg\max_{K=1...k} \sum_{j=1}^{k-1} \sum_{i=j+1}^{k} f(y, j, i, K) \tag{4.5}$$

The another method for the multi-class SVM is (OAA) i.e. One-Against-All. In this method k different classifiers are designed which considers one class as positive and the rest classes as negative. The decision function of this method which determines the class of the data point is given as:

$$S_{OAA} = \arg\max_{K=1...k} \; w_K^T y + b_K \tag{4.6}$$

4.3 Multi-class Lagrange Support Vector Machine

This section contains the newly proposed LSVM method for solving multi-class problems. Many researches were made before to deal with multi-class problems using one-against-one and one-against-all techniques [18]. All these methods consider classes one at a time but the new method developed in this paper that is lagrangian for multi-class problems (LSVMM-PAC) [19] trains computer to consider the classes together. To derive the formulation of this new method they considered a set of data points as $X = \{(y_1, p_1), (y_2, p_2).....(y_N, p_N)\}$ where $y_j's \in \mathcal{R}^n$ and $p_j's$ are the class labels of the data points which are {1,2,3,....,k}.

From the previous section we have come to know that SVMs for the multi-class problems are trained as:

$$w_{p_j}^T y_j + b_{p_j} \geq w_m^T y_j + b_m \quad where\, m \neq p_j \tag{4.7}$$

For linearly separable data points the above equation has infinitely many solutions for w_m. For the unique values some changes are made to the constraints in binary SVM, the new constraint thus becomes $w_{p_j}^T y_j + b_{p_j} = w_m^T y_j + b_m + 2$. Simply '1' is replaced by '2' and now when the data points are separable linearly the constraint is $w_{p_j}^T y_j + b_{p_j} \geq w_m^T y_j + b_m + 2$

and for not linearly separable data points constraint is $w_{p_j}^T y_j + b_{p_j} \geq w_m^T y_i + b_m + 2 - \nu_{m,j}$ and $\nu_{m,j} \geq 0$ where $m \neq p_j$. Therefore, the optimization problem becomes:

$$\min_{w,b,\nu} \ \frac{1}{2}\sum_{m=1}^{k} w_m^T w_m + \frac{1}{2}\sum_{m=1}^{k} b_m^2 + \frac{V}{2}\sum_{j=1}^{N}\sum_{\substack{m=1 \\ m\neq p_j}}^{k} \nu_{m,j}^2$$

Subject to

$$\begin{aligned} w_{p_j}^T y_j + b_{p_j} &\geq w_m^T y_i + b_m + 2 - \nu_{m,j} \\ \nu_{m,j} &\geq 0 \quad where \ \ m \neq p_j \end{aligned} \tag{4.8}$$

Solving the above optimization problem by Lagrange multipliers the equation of Lagrangian becomes:

$$\begin{aligned} L(w,b,\nu,u) = &\frac{1}{2}\sum_{m=1}^{k} w_m^T w_m + \frac{1}{2}\sum_{m=1}^{k} b_m^2 + \frac{V}{2}\sum_{j=1}^{N}\sum_{\substack{m=1 \\ m\neq p_j}}^{k} \nu_{m,j}^2 \\ &- \sum_{j=1}^{N}\sum_{\substack{m=1 \\ m\neq p_j}}^{k} u_{m,j}(w_{p_j}^T y_j + b_{p_j} - w_m^T y_i - b_m - 2 + \nu_{m,j}) \end{aligned} \tag{4.9}$$

where u is the lagrange multiplier and the matrices and vectors in the above equation is given as follows:

$$\begin{aligned} w &= [w_1 w_2 .... w_k] \in \mathcal{R}^{n\times k} \\ b &= [b_1 b_2 .... b_k]^T \in \mathcal{R}^{k\times 1} \\ \nu &= [\nu_1^T \nu_2^T .... \nu_N^T]^T \in \mathcal{R}^{N(k-1)} \\ \nu_j &= [\nu_{1,j}\nu_{2,j}...\nu_{p_i-1,j}\nu_{p_i+1,j}....\nu_{k-1,j}\nu_{k,j}]^T \in \mathcal{R}^{(k-1)\times 1} \\ u &= [u_1^T u_2^T .... u_N^T]^T \in \mathcal{R}^{N(k-1)} \\ u_j &= [u_{1,j}u_{2,j}...u_{p_i-1,j}u_{p_i+1,j}....u_{k-1,j}u_{k,j}]^T \in \mathcal{R}^{(k-1)\times 1} \end{aligned} \tag{4.10}$$

The dual of the above Lagrangian problem is: $f(u) = \min_{w,b,\nu} L(w,b,\nu,u)$

$$\begin{aligned} =\min_{w,b,\nu}\Bigg\{ &\frac{1}{2}\sum_{m=1}^{k} w_m^T w_m + \frac{1}{2}\sum_{m=1}^{k} b_m^2 + \frac{V}{2}\sum_{j=1}^{N}\sum_{\substack{m=1 \\ m\neq p_j}}^{k} \nu_{m,j}^2 \\ &- \sum_{j=1}^{N}\sum_{\substack{m=1 \\ m\neq p_j}}^{k} u_{m,j}(w_{p_j}^T y_j + b_{p_j} - w_m^T y_i - b_m - 2 + \nu_{m,j}) \Bigg\} \end{aligned} \tag{4.11}$$

Differentiating the Lagrangian by primal variables to minimize:

$$\begin{aligned}
\frac{\partial L}{\partial w_h} &= w_h - \sum_{\substack{j=1 \\ p_j=h}}^{N} \sum_{\substack{m=1 \\ m\neq h}}^{k} u_{m,j} y_j + \sum_{\substack{j=1 \\ p_j\neq h}}^{N} u_{h,j} y_j \\
\frac{\partial L}{\partial b_h} &= b_h - \sum_{\substack{j=1 \\ p_j=h}}^{N} \sum_{\substack{m=1 \\ m\neq h}}^{k} u_{m,j} + \sum_{\substack{j=1 \\ p_j\neq h}}^{N} u_{h,j} \\
\frac{\partial L}{\partial \nu_{h,j}} &= V\nu_{h,j} - u_{h,j}
\end{aligned} \tag{4.12}$$

where h = 1,2,...,k and j = 1,2,...,N

Calculating the values of the primal variables it gives:

$$\begin{aligned}
w_h &= \sum_{\substack{j=1 \\ p_j=h}}^{N} \sum_{\substack{m=1 \\ m\neq h}}^{k} u_{m,j} y_j - \sum_{\substack{j=1 \\ p_j\neq h}}^{N} u_{h,j} y_j \\
b_h &= \sum_{\substack{j=1 \\ p_j=h}}^{N} \sum_{\substack{m=1 \\ m\neq h}}^{k} u_{m,j} - \sum_{\substack{j=1 \\ p_j\neq h}}^{N} u_{h,j}
\end{aligned} \tag{4.13}$$

$$\nu_{h,j} = \frac{u_{h,j}}{V}$$

The above values are substituted in the equation (4.11) and it becomes:

$$\begin{aligned}
f(u) =& \frac{1}{2} \sum_{m=1}^{k} w_m^T w_m + \frac{1}{2} \sum_{m=1}^{k} b_m^2 + \frac{V}{2} \sum_{j=1}^{N} \sum_{\substack{m=1 \\ m\neq p_j}}^{k} \nu_{m,j}^2 \\
&+ \sum_{j=1}^{N} \sum_{\substack{m=1 \\ m\neq p_j}}^{k} 2u_{m,j} - \sum_{j=1}^{N} \sum_{\substack{m=1 \\ m\neq p_j}}^{k} u_{m,j}\nu_{m,j} \\
&- \sum_{j=1}^{N} \sum_{\substack{m=1 \\ m\neq p_j}}^{k} u_{m,j}\big(w_{p_j} - w_m\big) y_j - \sum_{j=1}^{N} \sum_{\substack{m=1 \\ m\neq p_j}}^{k} u_{m,j}\big(b_{p_j} - b_k\big)
\end{aligned} \tag{4.14}$$

To find the value of weight vector w and bias term b, the terms in the above equation are further simplified separately for easier calculations:

$$\sum_{j=1}^{N}\sum_{\substack{m=1\\m\neq p_j}}^{k} u_{m,j}\big(w_{p_j}-w_m\big)^T y_j$$

$$=\sum_{j=1}^{N}\sum_{\substack{m=1\\m\neq p_j}}^{k} u_{m,j}w_{p_j}^T y_j-\sum_{j=1}^{N}\sum_{\substack{m=1\\m\neq p_j}}^{k} u_{m,j}w_m^T y_j$$

$$=\sum_{h=1}^{k}\sum_{\substack{j=1\\p_j=h}}^{N}\sum_{\substack{m=1\\m\neq p_j}}^{k} u_{m,j}w_{p_j}^T y_j-\sum_{j=1}^{N}\sum_{\substack{m=1\\m\neq p_j}}^{k} u_{m,j}w_m^T y_j$$

$$=\sum_{h=1}^{k}\sum_{\substack{j=1\\p_j=h}}^{N}\sum_{\substack{m=1\\m\neq h}}^{k} u_{m,j}w_h^T y_j-\sum_{m=1}^{k}\sum_{\substack{j=1\\p_j\neq m}}^{N} u_{m,j}w_m^T y_j$$

$$=\sum_{h=1}^{k} w_h^T\left(\sum_{\substack{j=1\\p_j=h}}^{N}\sum_{\substack{m=1\\m\neq h}}^{k} u_{m,j}y_j-\sum_{\substack{j=1\\p_j\neq h}}^{N} u_{m,j}y_j\right)=\sum_{h=1}^{k} w_h^T w_h \tag{4.15}$$

Similarly solving the another term for simplification:

$$\sum_{j=1}^{N}\sum_{\substack{m=1\\m\neq p_j}}^{k} u_{m,j}\big(b_{p_j}-b_k\big)$$

$$=\sum_{j=1}^{N}\sum_{\substack{m=1\\m\neq p_j}}^{k} u_{m,j}b_{p_j}-\sum_{j=1}^{N}\sum_{\substack{m=1\\m\neq p_j}}^{k} u_{m,j}b_m$$

$$=\sum_{h=1}^{k}\sum_{\substack{j=1\\p_j=h}}^{N}\sum_{\substack{m=1\\m\neq p_j}}^{k} u_{m,j}b_{p_j}-\sum_{m=1}^{k}\sum_{\substack{j=1\\p_i\neq m}}^{N} u_{m,j}b_m$$

$$=\sum_{h=1}^{k}\sum_{\substack{j=1\\p_j=h}}^{N}\sum_{\substack{m=1\\m\neq h}}^{k} u_{m,j}b_h-\sum_{h=1}^{k}\sum_{\substack{j=1\\p_j\neq h}}^{N} u_{h,j}b_h$$

$$=\sum_{h=1}^{k} b_h\left(\sum_{\substack{j=1\\p_j=h}}^{N}\sum_{\substack{m=1\\m\neq h}}^{k} u_{m,j}-\sum_{\substack{j=1\\p_j\neq h}}^{N} u_{h,j}\right)=\sum_{h=1}^{k} b_h^2 \tag{4.16}$$

The value from the above equation (4.15), equation (4.16) and equation (4.13) is used in the equation (4.14) to find the solution of dual problem, the equation becomes:

$$\begin{aligned} f(u) =& \frac{1}{2}\sum_{m=1}^{k} w_m^T w_m + \frac{1}{2}\sum_{m=1}^{k} b_m^2 + \frac{V}{2}\sum_{j=1}^{N}\sum_{\substack{m=1\\ m\neq p_j}}^{k} \frac{u_{m,j}^2}{V^2} \\ &+ 2\sum_{j=1}^{N}\sum_{\substack{m=1\\ m\neq p_j}}^{k} u_{m,j} - \sum_{j=1}^{N}\sum_{\substack{m=1\\ m\neq p_j}}^{k} \frac{u_{m,j}^2}{V} - \sum_{m=1}^{k} w_m^T w_m - \sum_{m=1}^{k} b_m^2 \\ &= 2\sum_{m=1}^{k}\sum_{\substack{j=1\\ p_j\neq m}}^{N} u_{m,j} - \frac{1}{2}\sum_{h=1}^{k}\left(\sum_{\substack{j=1\\ p_j=h}}^{N}\sum_{\substack{m=1\\ m\neq h}}^{k} u_{m,j}y_j - \sum_{\substack{j=1\\ p_j\neq h}}^{N} u_{h,j}y_j\right)^T \\ &\times \left(\sum_{\substack{j=1\\ p_j=h}}^{N}\sum_{\substack{m=1\\ m\neq h}}^{k} u_{m,j}y_j - \sum_{\substack{j=1\\ p_j\neq h}}^{N} u_{h,j}y_j\right) \\ &- \frac{1}{2}\sum_{h=1}^{k}\left(\sum_{\substack{j=1\\ p_j=h}}^{N}\sum_{\substack{m=1\\ m\neq h}}^{k} u_{m,j} - \sum_{\substack{j=1\\ p_j\neq h}}^{N} u_{h,j}\right)^2 - \frac{1}{2V}\sum_{m=1}^{k}\sum_{\substack{j=1\\ p_j\neq m}}^{N} u_{m,j}^2 \end{aligned} \tag{4.17}$$

Further

$$\begin{aligned} f(u) =& 2\sum_{m=1}^{k}\sum_{\substack{j=1\\ p_j\neq m}}^{N} u_{m,j} - \frac{1}{2}\sum_{m=1}^{k}\left(\sum_{j=1}^{N}\sum_{\substack{m=1\\ m\neq p_j}}^{k}(J_{h,m,j}u_{m,j}y_j\right)^T \\ &\times \left(\sum_{j=1}^{N}\sum_{\substack{m=1\\ m\neq p_j}}^{k}(J_{h,m,j}u_{m,j}y_j\right) \\ &- \frac{1}{2}\sum_{h=1}^{k}\left(\sum_{j=1}^{N}\sum_{\substack{m=1\\ m\neq p_j}}^{k}(J_{h,m,j}u_{m,j})\right)^2 - \frac{1}{2V}\sum_{m=1}^{k}\sum_{\substack{j=1\\ p_j\neq m}}^{N} u_{m,j}^2 \end{aligned} \tag{4.18}$$

Where $J_{h,m,j}$ is defined as new parameter for solving dual easily

$$J_{h,m,j} = \begin{cases} 1 & \text{if } p_j = h, m \neq h, p_j \neq m \\ -1 & \text{if } p_j \neq h, m = h \\ 0 & \text{otherwise} \end{cases} \tag{4.19}$$

The optimization problem can be reformulated as:

$$\begin{aligned}\max_{u} f(u) = 2\sum_{m=1}^{k}\sum_{\substack{j=1\\p_j\neq m}}^{N} u_{m,j} \\ -\frac{1}{2}\sum_{m=1}^{k}\left(\sum_{j=1}^{N}\sum_{\substack{m=1\\m\neq p_j}}^{k}(J_{h,m,j}u_{m,j}y_j\right)^T\left(\sum_{j=1}^{N}\sum_{\substack{m=1\\m\neq p_j}}^{k}(J_{h,m,j}u_{m,j}y_j\right) \\ -\frac{1}{2}\sum_{h=1}^{k}\left(\sum_{j=1}^{N}\sum_{\substack{m=1\\m\neq p_j}}^{k}(J_{h,m,j}u_{m,j})\right)^2 - \frac{1}{2V}\sum_{m=1}^{k}\sum_{\substack{j=1\\p_j\neq m}}^{N} u_{m,j}^2\end{aligned}$$

Subject to $u \geq 0$

$$\text{Where } J_{h,m,j} = \begin{cases} 1 & \text{if } p_j = h, m \neq h, p_j \neq m \\ -1 & \text{if } p_j \neq h, m = h \\ 0 & \text{otherwise} \end{cases} \tag{4.20}$$

4.3.1 Matrix Representation

The multi-class output representation matrix (MORM) can be defined as:

$$J_h = \begin{bmatrix} J_{h1} & 0_{1\times(k-1)} & \cdots & 0_{1\times(k-1)} \\ 0_{1\times(k-1)} & J_{h2} & \cdots & 0_{1\times(k-1)} \\ \vdots & \vdots & \ddots & \vdots \\ 0_{1\times(k-1)} & 0_{1\times(k-1)} & \cdots & J_{hN} \end{bmatrix} \tag{4.21}$$

Where

$$J_{hj} = \begin{bmatrix} J_{h,1,j} & \cdots & J_{h,p_j-1,j} & J_{h,p_j+1,j} & \cdots & J_{h,k,j} \in \mathcal{R}^{1\times(k-1)} \end{bmatrix} \tag{4.22}$$

For the binary classification problem the multi-class output representation matrix is reduced to output diagonal matrix which is used in the LSVM.

Now with the help of J_h which is multi-class output representation matrix (MORM) the terms in the optimization problem are written in the vector and matrix form as follows:

$$\sum_{j=1}^{N}\sum_{\substack{m=1\\m\neq p_j}}^{k}(J_{h,m,j}u_{m,j}y_j) = [y_1, ..., y_N] \times \begin{bmatrix} J_{h1} & 0_{1\times(k-1)} & \cdots & 0_{1\times(k-1)} \\ 0_{1\times(k-1)} & J_{h2} & \cdots & 0_{1\times(k-1)} \\ \vdots & \vdots & \ddots & \vdots \\ 0_{1\times(k-1)} & 0_{1\times(k-1)} & \cdots & J_{hN} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_N \end{bmatrix} = Y^T J_h u \tag{4.23}$$

Where $Y = [y_1, ..., y_N]^T \in \mathcal{R}^{N \times n}$

Similarly,

$$\sum_{j=1}^{N} \sum_{\substack{m=1 \\ m \neq p_j}}^{k} (J_{h,m,j} u_{m,j}) = \begin{bmatrix} 1 & \cdots & 1 \end{bmatrix} \times \begin{bmatrix} J_{h1} & 0_{1\times(k-1)} & \cdots & 0_{1\times(k-1)} \\ 0_{1\times(k-1)} & J_{h2} & \cdots & 0_{1\times(k-1)} \\ \vdots & \vdots & \ddots & \vdots \\ 0_{1\times(k-1)} & 0_{1\times(k-1)} & \cdots & J_{hN} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_N \end{bmatrix} = 1^T J_h u \tag{4.24}$$

Where $1 = [1 \ ... \ 1]^T \in \mathcal{R}^{N \times 1}$

Using the equations (4.23) and equation (4.24) in equation (4.20) it gives:

$$f(u) = 2.1^T u - \frac{1}{2} u^T \left(\sum_{h=1}^{k} J_h^T \left(Y Y^T + 11^T \right) J_h + \frac{1}{V} I \right) u \tag{4.25}$$

$I = \begin{pmatrix} 1 & 1 & \cdots & 1 \end{pmatrix} \in \mathcal{R}^{N \times n}$

The dual formulation of LSVMMPAC in the matrix form is given as:

$$\min_u \ \frac{1}{2} u^T \left(\sum_{h=1}^{k} J_h^T \left(Y Y^T + 11^T \right) J_h + \frac{1}{V} I \right) u - 2.1^T u$$

$$\text{Subject to } u \geq 0 \tag{4.26}$$

SVM for multi-class classification problem is given as:

$$S = \arg \max_{h=1,...,k} (w_h^T + b_h) \tag{4.27}$$

Where weight and bias term are defined as:

$$\begin{aligned} w_h &= \sum_{\substack{j=1 \\ p_j = h}}^{N} \sum_{\substack{m=1 \\ m \neq h}}^{k} u_{m,j} y_j - \sum_{\substack{j=1 \\ p_j \neq h}}^{N} u_{h,j} y_j = \sum_{j=1}^{N} \sum_{\substack{m=1 \\ m \neq p_j}}^{k} (J_{h,m,j} u_{m,j} y_j) = Y^T J_h u \\ b_h &= \sum_{\substack{j=1 \\ p_j = h}}^{N} \sum_{\substack{m=1 \\ m \neq h}}^{k} u_{m,j} - \sum_{\substack{j=1 \\ p_j \neq h}}^{N} u_{h,j} = \sum_{j=1}^{N} \sum_{\substack{m=1 \\ m \neq p_j}}^{k} (J_{h,m,j} u_{m,j}) = 1^T J_h u \end{aligned} \tag{4.28}$$

To facilitate the training of the dual problem a theorem is constructed by the writers which is given below:

Theorem 1. *Let an optimization problem*

$$\min_{u} \quad \frac{1}{2}u^T Hu - g^T u$$

$$\text{Subject to } u \geq 0 \tag{4.29}$$

is taken into consideration, where $u \in \mathcal{R}^n, g \in \mathcal{R}^n$ *and* $H \in \mathcal{S}^n_{++}$.

The value of α *is chosen as*

$$0 < \alpha < 2\mu_{min}(H) \tag{4.30}$$

updated value of u can be calculated as:

$$u^{i+1} = H^{-1}\{g + [(Hu^i - g) - \alpha u^i]_+\}, i = 0, 1, 2, ... \tag{4.31}$$

then the lagrange multiplier u converges to the optimization point, $\mu_{min}(.)$ *refers to the minimum eigen value of the matrix and* $(y)_+$ *is defined as*

$$((y_+)_m = \begin{cases} 0 & \text{if } y_m < 0 \\ y_m & \text{if } y_m \geq 0 \end{cases} \tag{4.32}$$

where m is the mth component of y.

Proof. With the help of [21] the proof goes like this, define

$$H_{MPAC} = \left(\sum_{h=1}^{k} J_h^T (YY^T + 11^T) J_h + \frac{1}{V} I \right) \tag{4.33}$$

The optimal point is sure for the variable u as per the equation below:

$$u^{i+1} = H_{MPAC}^{-1}\Big(2.1 + ((H_{MPAC}u^i - 2.1) - \alpha u^i)_+\Big), \text{i} = 0,1,2,.... \tag{4.34}$$

Whichever kernel is selected, H_{MPAC} is positive definite always. It can be easily seen from the above equation. □

4.3.2 Non-linear LSVMMPAC with kernel trick

For non-linear classifier using kernel methods [2,11,21], The formulation of LSVM for non-linear kernel trick is:

$$\min_{w,b,\nu} \frac{1}{2}\sum_{m=1}^{k} w_m^T w_m + \frac{1}{2}\sum_{m=1}^{k} b_m^2 + \frac{V}{2}\sum_{j=1}^{N}\sum_{\substack{m=1 \\ m\neq p_j}}^{k} \nu_{m,j}^2$$

Subject to

$$w_{p_j}^T \phi(y_j) + b_{p_j} \geq w_m^T \phi(y_j) + b_m + 2 - \nu_{m,j}(m \neq p_j), \quad \nu_{m,j} \geq 0 \tag{4.35}$$

The dual of the above problem is given as:

$$\min_{u} \quad \frac{1}{2}u^T \left(\sum_{h=1}^{k} J_h^T \Big(K(Y,Y) + 11^T \Big) J_h + \frac{1}{V} I \right) u - 2.1^T u$$

$$\text{Subject to } u \geq 0 \tag{4.36}$$

To train the non-linear algorithm the iterative scheme is:

$$u^{i+1} = H_{MPACNL}^{-1} \Big(2.1 + ((H_{MPACNL} u^i - 2.1) - \alpha u^i)_+ \Big), \text{i} = 0,1,2,.... \tag{4.37}$$

and

$$H_{MPACNL} = \left(\sum_{h=1}^{k} J_h^T (K(Y,Y) + 11^T) J_h + \frac{1}{V} I \right) \tag{4.38}$$

The decision function for non-linear kernel trick is defined as follows:

$$S_{MPACNL} = \arg\max_{h=1,...,k} (K(y,Y) J_h u + b_h) \quad \text{and } b_h = 1^T J_h u \tag{4.39}$$

4.4 Features of LSVMMPAC

The optimization problem in the equation (4.20) is a quadratic program but solution space is quite large and QP will take much time for solving the above problem. LSVMMPAC is much more effective than SVMs. Some of the features of LSVMMPAC are given below:

1. Multi-class SVMs use two method OAO and OAA to find the classifiers considering one class at a time whereas LSVMMPAC considers all the classes together.

2. This new method just have a simple non-negative inequality constraint and it does not have any equality constraint.

3. The method in [22] has many unknown variables which complicates the optimization whereas LSVMMPAC has only one unknown variable.

4. The LSVM is a simple method therefore it can be generalized for multi-class problem.

Conclusion

This dissertation is the study of algorithms for solving binary and multi-class classification problems. Support Vector Machines, the base of many algorithms is considered and further, two research papers are reviewed explaining the Lagrangian Support Vector Machines for binary classification and multi-classification. Algorithms are applied on some data sets from UPI repository to check the accuracy of these methods. Machine Learning is a vast topic and I have tried to grab some knowledge about some of its methods like Support Vector Machines, Lagrangian Support Vector Machines and kernel methods. Mathematical derivations of the optimization problems using lagrange multipliers, working on matlab has developed my skills in programming.

Bibliography

[1] C. Cortes and V. Vapnik, "Support-Vector Networks", *Machine Learning*, 20(3), pp. 273–297, 1995.

[2] V. Vapnik, *The Nature of Statistical Learning Theory*, Springer, New York, 1995.

[3] B. Wallace Constrained optimization: *Kuhn-Tucker conditions*, 2004.

[4] H.C. Wu The Karush–Kuhn–Tucker optimality conditions in an optimization problem with interval-valued objective function *European Journal of Operational Research*, 176, pp. 46-59, 2007.

[5] VV. I. Paulsen, M. Raghupathi *An Introduction to the Theory of Reproducing Kernel Hilbert Spaces*, Cambridge, pp. 65-81, 2016.

[6] M. Aizerman, E. Braverman, and L. Rozonoer, *Theoretical foundations of the potential function method in pattern recognition learning, Automation and Remote Control*, 25, pp. 821–837, 1964.

[7] N. Aronszajn, "Theory of Reproducing Kernels", *Transactions of the American Mathematical Society*, 68, pp. 337-404, 1950.

[8] P. S. Bradley and O. L. Mangasarian, Massive data discrimination via linear support vector machines. *Optimization Methods and Software*, 13, pp. 1–10, 2000.

[9] O. L. Mangasarian, Generalized support vector machines, A. Smola, P. Bartlett, B. Scholkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*, pp. 135–146, Cambridge, MA, 2000.

[10] Yuh-Jye Lee and O. L. Mangasarian, A smooth support vector machine. *Computational Optimization and Applications*, 2000.

[11] O. Mangasarian and D. Musicant, Lagrangian support vector machines. *Journal of Machine Learning Research (JMLR)*, 1, pp. 161–177, 2001.

[12] O. L. Mangasarian and M. V. Solodov, linear complementarity as unconstrained and constrained minimization. *Mathematical Programming, Series B*, 62, pp. 277–297, 1993.

[13] G. H. Golub and C. F. Van Loan, *Matrix Computations*. The John Hopkins University Press, Baltimore, Maryland, 3rd edition, 1996.

[14] O. L. Mangasarian and D. R. Musicant, Active support vector machine classification. *Advances in Neural Information Processing Systems*, 2000.

[15] O. L. Mangasarian, *Nonlinear Programming*. SIAM, Philadelphia, PA, 1994.

[16] D. P. Bertsekas, *Nonlinear Programming*. Athena Scientific, Belmont, MA, second edition, 1999.

[17] O. L. Mangasarian and J. Ren, New improved error bounds for the linear complementarity problem. *Mathematical Programming*, 66, pp. 241–255, 1994.

[18] H. Duan, Q. Liu, G. He, Two multi-class lagrangian support vector machine algorithms. *Lecture Notes in Computer Science, ICIC* , pp. 891–899, 2007.

[19] J. P. Hwang, B. Choi, I. W. Hong, and E. Kim, Multiclass lagrangian support vector machine, *Neural Computing and Applications*, 22(3-4), pp. 703– 710, 2013.

[20] C. Demirkesen, H. Cherifi, A comparison of multiclass SVM methods for real world natural scenes, *Advanced Concepts for Intelligent Vision Systems*, pp. 752-763, 2008

[21] J. P. Hwang, S. Park, E. Kim, Dual margin approach on a Lagrangian support vector machine. *Int. J. Comput. Math.* 88(4), pp. 695–708, 2011.

[22] C. W. Hsu, C.J. Lin, A comparison of methods for multiclass support vector machines. *IEEE Transactions on Neural Networks* 13(2), pp. 415–425, 2002.