# Performance Evaluation of Kernel SVM on Sparse Datasets with Large Attributes

Chamila Walgampaya[1], Modestus Lorence[2]
[1]Department of Engineering Mathematics, Faculty of Engineering, University of Peradeniya, Peradeniya, Sri Lanka
[2]Network and Communication Services Unit, Faculty of Engineering, University of Peradeniya, Peradeniya, Sri Lanka
[1]ckw@pdn.ac.lk, [2]lorence@eng.pdn.ac.lk

***Abstract*—Support Vector Machines (SVM) is a Machine Learning Algorithm which is used for Classification and Regression in many applications. The vital characteristic of SVM is that the classification decision function is formulated using very few points in the training dataset. We have provided the less publicized mathematical formulation of Hard Margin SVM Classifier, Soft Margin SVM Classifier and the Kernel Trick. In this paper we have used two sparse data sets, we found that Kernel SVM shows significantly better generalization and prediction accuracy for sparse datasets. We have compared the classification performance with other Machine Learning algorithms such as Logistic Regression, Neural Networks, Bayesian Network, KNN, Bagging and Random Forest.**

***Index Terms*—Support Vector Machines, Kernels, sparse data, large attributes, Machine Learning**

## I. INTRODUCTION

Support Vector Machines (SVM) was introduced in 1992 by Vapnik et al. [1]. SVM is a learning system which uses a hypothesis space of linear functions in a high dimensional feature space. It is one of the very successful algorithms due to some key factors such as 'maximum margin' which leads to better generalization, 'dual form' makes it easier to optimize and the 'kernel trick' which is used to perform the classification in a higher dimensional space without the expense of the high computational cost. Due to this SVM has been used in many areas such as text classification [3], image classification [4], speech recognition [5], face detection [6]. Most of the current developments in SVM are focused on Kernel improvements [7], changing the learning methods [8], parameter optimization [9] and tuning [10].

In SVM classification, the learning problem transformed to a Quadratic Programming (QP) problem subject to the constraints. The basic linear framework is easily extended to a non-linearly separable data points in SVM. The fundamental idea behind this extension is to transform the input space where the dataset is not linearly separable into a higher dimensional feature space where the data could be linearly separable. The function associated with this transformation is called the "Kernel Function", and the process of using this function to move from a linear to a non-linear SVM is called the 'Kernel Trick'.

## II. FORMULATION

### A. *Hard Margin SVM Classifier*

The Hard Margin SVM Classifier is also known as Maximum Margin Classifier was introduced in 1992 [1]. In Hard Margin SVM Classifier for binary classification, it is based on searching for a separating plane (or hyperplane in higher dimensional space) that is equidistant to the class boundaries where the two classes are closest to each other. It also maximizes the distance to these class boundaries. The separating hyperplane is placed bisecting the shortest line connecting the class boundaries.

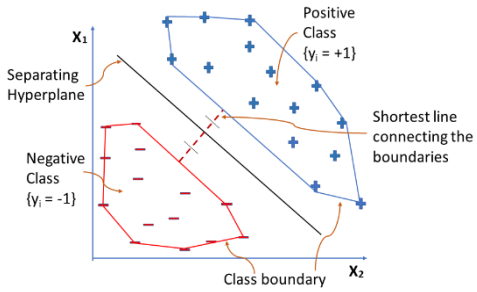


Fig. 1. Maximum Margin Classifier

Consider that we have two arrays of linearly separable data as shown in Fig1. Let the input $X$ has two features $\{X_1, X_2\} = \{(x_{i1}, x_{i2})\}$ where $i = 1,2,\dots n$ The data has two classes; Positive Class (Class 1) when $y_i = +1$, Negative Class (Class 2) when $y_i = -1$. We have to find the optimal separating hyperplane which has the largest margin.

The Equation of a Plane is derived as follows:

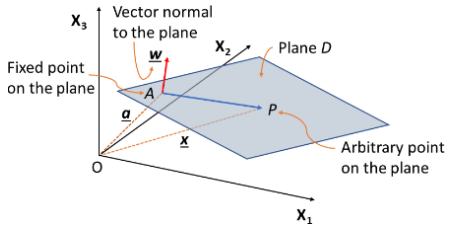


Fig. 2. Diagram to derive the equation of a plane

The above Fig 2 depicts a plane in 3-dimensional space and the vectors related to the plane. $A$ is a fixed point on the

Plane $D$, $\vec{OA} = \boldsymbol{a}$, $P$ is an arbitrary point on the Plane $D$, $\vec{OP} = \boldsymbol{x}$.

$$\begin{aligned} & \vec{AP} = \vec{OP} - \vec{OA} \\ & \vec{AP} = \boldsymbol{x} - \boldsymbol{a} \\ & \boldsymbol{w} \perp \vec{AP} \quad (\because \boldsymbol{w} \perp \text{Plane}D) \\ \therefore \quad & \boldsymbol{w} \cdot \vec{AP} = 0 \\ & \boldsymbol{w} \cdot (\boldsymbol{x} - \boldsymbol{a}) = 0 \\ & \boldsymbol{w} \cdot \boldsymbol{x} - w \cdot a = 0 \end{aligned} \tag{1}$$

Equation of a plane: $\boxed{\boldsymbol{w} \cdot \boldsymbol{x} + b = 0}$ $(b = -\boldsymbol{w} \cdot \boldsymbol{a})$

Where the vector $\boldsymbol{w}$ is normal to the Plane $D$. The above derived equation could be generalized to a hyperplane in $d$ dimension. If vector $\boldsymbol{x} \in \mathbb{R}^d$, then $\boldsymbol{w} \in \mathbb{R}^d$ and $b$ is a scalar. In the binary classification case (as in Fig1), the equation of the separating hyperplane could be written as follows:

$$D(x) = \boldsymbol{w} \cdot \boldsymbol{x} + b = 0 \tag{2}$$

The decision function $D(x)$ of the optimal separating hyperplane is selected to have equal distance from the positive margin and the negative margin. The separating plane and the margins are illustrated in Fig 3.

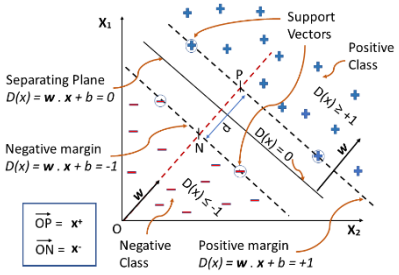


Fig. 3. Hard Margin Classification

The equations for decision function and the margins can be written as follows.

Decision Function: $D(x) = \boldsymbol{w} \cdot \boldsymbol{x} + b$

Separating Plane: $\boldsymbol{w} \cdot \boldsymbol{x} + b = 0$

Positive margin: $\boldsymbol{w} \cdot \boldsymbol{x} + b = +1$

Negative margin: $\boldsymbol{w} \cdot \boldsymbol{x} + b = -1$

From Fig 3, there are two classes of points; Positive class $(D(x) > 0)$ and Negative class $(D(x) < 0)$. The points lie on either the positive margin or the negative margin are called Support Vectors [2]. The dotted line is perpendicular to the margins, this line intersects the margins at $N$ and $P$. The distance between these two points $(NP = d)$ will be the closest distance between the boundaries of the two classes. Basically we have to find the Decision function which maximize the distance $d$. Point $N$ lies on the Negative margin, and it satisfies,

$$\boldsymbol{w} \cdot \boldsymbol{x}^- + b = -1 \tag{3}$$

Similarly, point $P$ satisfies the following equation.

$$\boldsymbol{w} \cdot \boldsymbol{x}^+ + b = +1 \tag{4}$$

(4)-(3) $\Rightarrow$ $\mathbf{w} \cdot (\mathbf{x}^+ - \mathbf{x}^-) = 2$

$\|\mathbf{w}\| \times \|(\mathbf{x}^+ - \mathbf{x}^-)\| = 2 \quad (\because \mathbf{w} \parallel (\mathbf{x}^+ - \mathbf{x}^-))$

$\|\mathbf{w}\| \times d = 2$

$d = 2 / \|\mathbf{w}\|$

To obtain the optimal separating plane we have to maximize $d = 2/\| \boldsymbol{w} \|$. Next step is to derive the constrains.

For Positive class Points: $\boldsymbol{w} \cdot \boldsymbol{x} + b \geq +1, \ \{y_i = +1\}$ (5)

For Negative class Points: $\boldsymbol{w} \cdot \boldsymbol{x} + b \leq -1, \ \{y_i = -1\}$ (6)

If we multiply either inequalities (5) or (6) with the corresponding $y_i$ we get the same outcome as follows:

$$y_i(\boldsymbol{w} \cdot \boldsymbol{x_i} + b) \geq +1, \quad i = 1,2,\ldots.n \tag{7}$$

In order to find $\boldsymbol{w}$ and $b$ of the Decision function $D(x)$, we have to maximize $d = 2/\| \boldsymbol{w} \|$ such that the constraint

$y_i(\boldsymbol{w} \cdot \boldsymbol{x} + b) \geq 1$ is satisfied. Maximizing $2/\| \boldsymbol{w} \|$ would yeild the same results if we minimize $\frac{1}{2} \| \boldsymbol{w} \|^2$. The function $\frac{1}{2} \| \boldsymbol{w} \|^2$ is a quadratic, continuous and a differentiable function and we could reach a global minimum when we optimize it.

The Optimization Problem could be written as follows:

$$\begin{aligned} & \min_{w,b} \ \frac{1}{2} \| \boldsymbol{w} \|^2 \\ & \text{s.t.} \ \ y_i(\boldsymbol{w} \cdot \boldsymbol{x_i} + b) \geq +1, \quad i = 1,2,\ldots n \end{aligned} \tag{8}$$

We have to understand the $(Karush - Kuhn - Tucker)$ $(KKT)$ conditions in order to resolve the above optimization problem.

***Karush-Kuhn-Tucker (KKT) Conditions:***

KKT Conditions are used for Optimization Problems with Inequality Constraints [11]. Suppose we have to resolve the following optimization problem.

$$\begin{aligned} & \min_{\boldsymbol{w}} \ \ f(\boldsymbol{w}) \\ & \text{s.t.} \ \ g_i(\boldsymbol{w}) \leq 0, \quad i = 1,2,\ldots n \end{aligned} \tag{9}$$

Where the function $g_i : \mathbb{R}^d \longmapsto \mathbb{R}$, $i = 1,2,\ldots n$ are the Inequality Constraints. Let's use $S$ to denote the domain of this problem. $\boldsymbol{w}^*$ is the solution of the above optimization problem, if $f(w^*) < f(\boldsymbol{w})$ for all $\boldsymbol{w}$ satisfying $g_i(\boldsymbol{w}) \leq 0$, where $i = 1,2,\ldots n$

The Lagrangian function $L$ streamlined with the optimization problem is given as

$$L(\boldsymbol{w}, \boldsymbol{\alpha}) = f(\boldsymbol{w}) + \sum_{i=1}^{n} \alpha_i g_i(\boldsymbol{w}) \tag{10}$$

where $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \ldots. \alpha_n]^T \in \mathbb{R}^n$ is called as the Lagrange multiplier.

$q(\boldsymbol{\alpha})$ is the Lagrange Dual function, it is defined as the infimum of the Lagrange function $L(\boldsymbol{w}, \boldsymbol{\alpha})$ in respect of $\boldsymbol{\alpha}$.

$$q(\boldsymbol{\alpha}) = \inf_{w \in \S} L(\boldsymbol{w}, \boldsymbol{\alpha}) \tag{11}$$

Indicate $f^*$ as the optimal value of the primal problem.

$q(\boldsymbol{\alpha}) \leq f^*$. This is known as the weak duality.

We have to optimize the following Lagrange dual optimization problem in order to get the lower bound of $f^*$.

$$\begin{aligned} &\max_{\boldsymbol{\alpha}} \quad q(\boldsymbol{\alpha}) \\ &\text{s.t.} \quad \alpha_i \geq 0, \quad i = 1,2,\dots n \end{aligned} \tag{12}$$

When the primal problem and the dual problem optimal values are same, it is called Strong Duality. KKT conditions are broadly used with problems where strong duality exists.

KKT conditions for the optimization problem in (9) are

$$\begin{aligned} &(1) \quad \nabla L(\boldsymbol{w}^*, \boldsymbol{\alpha}) = \nabla f(\boldsymbol{w}^*) + \textstyle\sum_{i=1}^{n} \alpha_i \nabla g_i(\boldsymbol{w}^*) = 0 \\ &(2) \quad g_i(\boldsymbol{w}^*) \leq 0, \quad i = 1,2,\dots n \\ &(3) \quad \alpha_i^* g_i(\boldsymbol{w}^*) = 0, \quad i = 1,2,\dots n \\ &(4) \quad \alpha_i^* \geq 0, \quad i = 1,2,\dots n \end{aligned} \tag{13}$$

$\boldsymbol{w}^*$ is an optimal solution when there exist a $\boldsymbol{\alpha}^* \in \mathbb{R}^d$ that makes the constraint qualifications in (12) holds.

Next, we will apply the KKT conditions to the SVM Optimization problem in (8). $f$ and $g_i$ could be written as follows for the SVM Optimization Problem in (8),

$$\begin{aligned} &f(\boldsymbol{w}) = \tfrac{1}{2} \| \boldsymbol{w} \|^2 = \tfrac{1}{2} \boldsymbol{w}^T \boldsymbol{w} \\ &g_i(\boldsymbol{w}, b) = -[y_i(\boldsymbol{w} \cdot \boldsymbol{x_i} + b) - 1] \leq 0, \quad i = 1,2,\dots n \end{aligned} \tag{14}$$

As per (10), the Lagrangian function for the SVM Optimization Problem,

$$L(\boldsymbol{w}, b, \boldsymbol{\alpha}) = \frac{1}{2} \boldsymbol{w}^T \boldsymbol{w} - \sum_{i=1}^{n} \alpha_i [y_i(\boldsymbol{w} \cdot \boldsymbol{x_i} + b) - 1] \tag{15}$$

$\boldsymbol{\alpha} \in \mathbb{R}^n$ is the associated Lagrange multiplier.

$\alpha_i \geq 0$ for $i = 1,2,..n$. By applying the $KKT$ conditions to the Optimization problem in (8) we get:

$$\begin{aligned} &(1) \quad \nabla_w L(\boldsymbol{w}^*, b, \boldsymbol{\alpha}) = 0 \\ &\quad \boldsymbol{w}^* - \textstyle\sum_{i=1}^{n} \alpha_i y_i x_i = 0, \quad i = 1,2,\dots n \\ &\quad \therefore \quad \boldsymbol{w}^* = \textstyle\sum_{i=1}^{n} \alpha_i y_i x_i, \quad i = 1,2,\dots n \\ &\quad \tfrac{\partial L(\boldsymbol{w}, b, \boldsymbol{\alpha})}{\partial b} = 0 \\ &\quad \textstyle\sum_{i=1}^{n} \alpha_i y_i = 0, \quad i = 1,2,\dots n \\ &(2) \; g_i(\boldsymbol{w}^*) \leq 0, \quad i = 1,2,\dots n \\ &\quad g_i(\boldsymbol{w}) = -[y_i(\boldsymbol{w} \cdot \boldsymbol{x_i} + b) - 1] \leq 0, \quad i = 1,2,\dots n \\ &\quad y_i(\boldsymbol{w} \cdot \boldsymbol{x_i} + b) \geq 1, \quad i = 1,2,\dots n \\ &(3) \; \alpha_i^* g_i(x^*) = 0, \quad i = 1,2,\dots n \\ &\quad \alpha_i^* [-(y_i(\boldsymbol{w}^* \cdot \boldsymbol{x_i} + b) - 1)] = 0, \quad i = 1,2,\dots n \\ &(4) \; \alpha_i^* \geq 0, \quad i = 1,2,\dots n \end{aligned} \tag{16}$$

From the above findings in (16),

$$\alpha_i^* [y_i(\boldsymbol{w}^* \cdot \boldsymbol{x_i} + b) - 1] = 0, \tag{17}$$

where $\alpha_i^* \geq 0, \quad i = 1,2,\dots n$

$\Rightarrow$

$(a)$ When $\alpha_i = 0$, then $[y_i(\boldsymbol{w}^* \cdot \boldsymbol{x_i} + b) - 1] \neq 0$,

$(b)$ When $\alpha_i > 0$, *then* $[y_i(\boldsymbol{w}^* \cdot \boldsymbol{x_i} + b) - 1] = 0$,

As per $(a)$ and $(b)$, when $\alpha_i > 0$ then $y_i(\boldsymbol{w}^* \cdot \boldsymbol{x_i} + b) = 1$. These are the points lie on both the margins. $\alpha_i = 0$ for all other points. As we know from (16), $\boldsymbol{w}^* = \sum_{i=1}^{n} \alpha_i y_i x_i$.

So $\boldsymbol{w}^*$ will be calculated based on the points which lie on the margin, thus the decision function also depends on these points. In Hard Margin Classification the points lying on the Positive or Negative margins are called ***Support Vectors*** [2]. These are the only points needed to classify the unseen data and that is why it is called Support Vector Machines (SVM)

The Lagrange dual function $q(\boldsymbol{\alpha})$ for the Lagrange primal function $L(\boldsymbol{w}, b, \boldsymbol{\alpha})$ in (15) is written as,

$$q(\boldsymbol{\alpha}) = \inf_{\boldsymbol{w}, b} L(\boldsymbol{w}, \boldsymbol{\alpha}, b)$$

$$q(\boldsymbol{\alpha}) = L(\boldsymbol{w}^*, \boldsymbol{\alpha}, b^*)$$

By substituting $\boldsymbol{w}^* = \sum_{i=1}^{n} \alpha_i y_i x_i$, we get,

$$q(\boldsymbol{\alpha}) = \frac{1}{2} \Big(\sum_{i=1}^{n} \alpha_i y_i x_i\Big)^T \left(\sum_{j=1}^{n} \alpha_j y_j x_j\right)$$

$$- \sum_{j=1}^{n} \alpha_j [y_j ((\sum_{i=1}^{n} \alpha_i y_i x_i)^T x_j + b^*) - 1]$$

$$q(\boldsymbol{\alpha}) = \frac{1}{2} \Big(\sum_{i=1}^{n} \alpha_i y_i x_i\Big)^T \Big(\sum_{j=1}^{n} \alpha_j y_j x_j\Big) + \sum_{j=1}^{n} \alpha_j$$

$$- \Big(\sum_{i=1}^{n} \alpha_i y_i x_i\Big)^T \Big(\sum_{j=1}^{n} \alpha_j y_j x_j\Big) + b^* \underbrace{\sum_{j=1}^{n} \alpha_j y_j}_{\text{equaltozero}}$$

$$\boxed{q(\boldsymbol{\alpha}) = \sum_{j=1}^{n} \alpha_j - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} y_i y_j (x_i \cdot x_j) \alpha_i \alpha_j} \tag{18}$$

As shown in (12) we have to optimize the $q(\boldsymbol{\alpha})$ in (18).

$$\begin{aligned} &\max_{\alpha} \sum_{j=1}^{n} \alpha_j - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} y_i y_j (x_i \cdot x_j) \alpha_i \alpha_j \\ &\text{s.t.} \sum_{j=1}^{n} \alpha_j y_j = 0, \quad \text{and } \alpha_i \geq 0, \quad i = 1,2,\dots n \end{aligned} \tag{19}$$

The above dual problem is a maximization problem. In optimization the maximization problem is often replaced by a minimization problem [12].

The following convex quadratic programming problem has to be solved to find the optimum $\boldsymbol{\alpha}^*$.

$$\begin{aligned} &\min_{\alpha} \ \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} y_i y_j (x_i \cdot x_j)\alpha_i\alpha_j - \sum_{j=1}^{n}\alpha_j \\ &\text{s.t.} \ \sum_{j=1}^{n}\alpha_j y_j = 0 \\ &\text{and} \ \ \alpha_i \geq 0, \quad i = 1,2,\dots n \end{aligned} \tag{20}$$

After we solve the optimization problem in (20), the solution $\boldsymbol{\alpha}^*$ will be used to find the Decision Function $D(x)$ of the separating hyperplane. The optimal separating hyperplane,

$$\begin{aligned} &D(x) = \boldsymbol{w}^* \cdot \boldsymbol{x} + b^* \\ &\text{where} \ \ \boldsymbol{w}^* = \textstyle\sum_{j=1}^{l} \boldsymbol{\alpha_j}^* y_j x_j \end{aligned} \tag{21}$$

$l$ - number of Support Vectors ($\boldsymbol{\alpha_j}^* > 0$).

To obtain $b^*$ choose any positive component of $\boldsymbol{\alpha}^*$. From (17), for $\alpha > 0$, $[1 - y_i(\boldsymbol{w}^* \cdot \boldsymbol{x_i} + b^*)] = 0$.

$$\begin{aligned} y_i(\boldsymbol{w}^* \cdot \boldsymbol{x_i} + b^*) - 1 &= 0 \\ \boldsymbol{w}^* \cdot \boldsymbol{x_i} + b^* &= \tfrac{1}{y_i} \\ \boldsymbol{w}^* \cdot \boldsymbol{x_i} + b^* &= y_i \quad (\because \ y_i = \pm 1) \\ b^* &= y_i - \boldsymbol{w}^* \cdot \boldsymbol{x_i} \\ b^* &= y_i - \textstyle\sum_{j=1}^{l} \boldsymbol{\alpha_j}^* y_j (x_j \cdot x_i) \end{aligned} \tag{22}$$

$\{x_i, y_i\}$ belongs to one of the Support Vector point.

Decision Function $D(x)$,

$$D(x) = \boldsymbol{w}^* \cdot \boldsymbol{x} + b^* \tag{23}$$

$$\boxed{D(x) = \sum_{j=1}^{l} \boldsymbol{\alpha_j}^* y_j (x_j \cdot x) + [y_i - \sum_{j=1}^{l} \boldsymbol{\alpha_j}^* y_j (x_j \cdot x_i)]}$$

Where $\{x_i, y_i\}$ is a support vector and $j = 1,2,\dots l$ are the indices of all the support vectors.

We can see from the Decision function $D(x)$, that only the training points corresponding to the support vectors have contributed and all other points have not contributed at all. What If we have some noisy data and the classes are not linearly separable in feature space? In the next section we are going to look into this problem.

### *B. Soft Margin SVM Classifier*

If the data is noisy, in general there will be no linear separation in feature space. In situations where linear separation is not possible due to noisy data, we could construct a linear classifier by introducing a penalty function ($\xi_i$) for the classification errors. Then optimize the function to minimize the classification errors ($\xi_i$) as much as possible.

As depicted in Fig 4, the classes are not linearly separable due to noisy data. $\xi_i$ is the distance of the points which are on the wrong side of the margins. It is also called the Slack variable, and where $\xi_i \geq 0$ for all points.

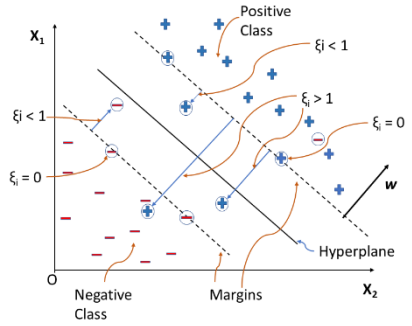


Fig. 4. Soft Margin Classification

For linearly classifiable data, the following inequality holds,

$$y_i((\boldsymbol{w} \cdot \boldsymbol{x_i}) + b) \geq 1, i = 1,2,\dots n$$

For the points on the wrong side, the above inequality will not be satisfied, in this case those points would satisfy the following inequality.

$$y_i((\boldsymbol{w} \cdot \boldsymbol{x_i}) + b) < 1, i = 1,2,\dots n$$

We could generalize the inequality condition for all the points which are on the correct side of the margins and wrong side of the margins as follows.

$$y_i((\boldsymbol{w} \cdot \boldsymbol{x_i}) + b) \geq 1 - \xi_i, \quad i = 1,2,\dots n$$

where

$\xi_i = 0$ for the points on the correct side of the margin

$\xi_i \geq 0$ for the points on the wrong side of the margin

Therefore, we could formulate the optimization problem for the soft margin SVM as follows,

$$\begin{aligned} &\min_{w,b} \quad \frac{1}{2} \parallel \boldsymbol{w} \parallel^2 + C\sum_{i=1}^{n}\xi_i \\ &\text{s.t.} \quad y_i\big((\boldsymbol{w} \cdot \boldsymbol{x}) + b\big) \geq 1 - \xi_i \quad , i = 1,2, \dots n \\ &\Rightarrow \quad -\big[y_i\big((\boldsymbol{w} \cdot \boldsymbol{x}) + b\big) - 1 + \xi_i\big] \leq 0 \quad , i = 1,2, \dots n \\ &\text{where} \quad \xi_i \geq 0 \quad (-\xi_i \leq 0) \end{aligned} \tag{24}$$

$C$ is the penalty parameter. Lagrangian function for the optimization problem in (18). This is the primal form of the Lagrange function for Soft Margin Classification.

$$\begin{aligned} L(\boldsymbol{w}, b, \xi_i, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \ &\frac{1}{2}\boldsymbol{w^T w} + C\sum_{i=1}^{n}\xi_i - \sum_{i=1}^{n}\beta_i\xi_i \\ &- \sum_{i=1}^{n}\alpha_i[y_i(\boldsymbol{w} \cdot \boldsymbol{x_i} + b) - 1 + \xi_i] \end{aligned} \tag{25}$$

$\boldsymbol{\alpha} \in \mathbb{R}^n$ and $\boldsymbol{\beta} \in \mathbb{R}^n$ are the associated Lagrange multipliers,

where $\alpha_i \geq 0$ and $\beta_i \geq 0$ for $i = 1,2,..n$

Apply the $KKT$ conditions to the Optimization problem in (24):

$$
\begin{aligned}
&(1)\nabla_w L(\boldsymbol{w}^*, b, \xi_i, \boldsymbol{\alpha}, \boldsymbol{\beta}) = 0 \\
&\Rightarrow \quad \boldsymbol{w}^* = \sum_{i=1}^{n} \alpha_i y_i x_i, \quad i = 1,2,\dots n \\
&\frac{\partial L(\boldsymbol{w}^*, b, \xi_i, \boldsymbol{\alpha}, \boldsymbol{\beta})}{\partial b} = 0 \\
&\Rightarrow \quad \sum_{i=1}^{n} \alpha_i y_i = 0, \quad i = 1,2,\dots n \\
&\frac{\partial L(\boldsymbol{w}^*, b, \xi_i, \boldsymbol{\alpha}, \boldsymbol{\beta})}{\partial \xi_i} = 0 \qquad (26) \\
&\Rightarrow \quad C - \beta_i - \alpha_i = 0, \quad i = 1,2,\dots n \\
&(2) \quad -[y_i(\boldsymbol{w} \cdot \boldsymbol{x_i} + b) - 1 + \xi_i] \le 0, \quad -\xi_i \le 0 \\
&\Rightarrow \quad y_i(\boldsymbol{w} \cdot \boldsymbol{x_i} + b) \ge 1 - \xi_i, \quad i = 1,2,\dots n \\
&\Rightarrow \quad \xi_i \ge 0, \quad i = 1,2,\dots n \\
&(3) \quad \alpha_i^*[(y_i(\boldsymbol{w}^* \cdot \boldsymbol{x_i} + b) - 1 + \xi_i)] = 0, \quad i = 1,2,\dots n \\
&\qquad \beta_i^* \xi_i^* = 0, \quad i = 1,2,\dots n \\
&(4) \quad \alpha_i^* \ge 0, \quad \beta_i^* \ge 0, \quad i = 1,2,\dots n
\end{aligned}
$$

Findings from above $KKT$ Conditions (26) are:

• $C - \beta_i^* - \alpha_i^* = 0$ so $\beta_i^* = C - \alpha_i^*$

$\beta_i^* \ge 0$, thus $C - \alpha_i^* \ge 0$ so $\alpha_i^* \le C$

• $\alpha_i^* \ge 0 \quad \Rightarrow \quad 0 \le \alpha_i^* \le C$

• $\beta_i^* \xi_i^* = 0$;

$(C - \alpha_i^*)\xi_i = 0$; $\because \beta_i^* = (C - \alpha_i^*)$

$\Rightarrow \quad \alpha_i^* = C$, when $\xi_i^* > 0$

• $\alpha_i^*[(y_i(\boldsymbol{w}^* \cdot \boldsymbol{x_i} + b) - 1 + \xi_i)] = 0, \quad i = 1,2,\dots n$

if $\alpha_i^* > 0$, then $y_i(\boldsymbol{w}^* \cdot \boldsymbol{x_i} + b) = 1 - \xi_i$

- when $\xi_i = 0$, the points lie on the margins
- when $\xi_i < 1$, the points are on the wrong side of the margin, but correctly classified
- when $\xi_i > 1$, the points are misclassified
- $\xi_i = 1 - y_i(\boldsymbol{w}^* \cdot \boldsymbol{x_i} + b)$

Since $\alpha_i^* > 0$ when $\xi_i \ge 0$ all the points which lie on the margin and which are on the wrong side of the margins contribute to the Decision function $D(x)$.

As shown in (10) the Lagrange dual function $q(\boldsymbol{\alpha})$ for the Lagrange primal function $L(\boldsymbol{w}, b, \boldsymbol{\alpha})$ is written as,

$$q(\boldsymbol{\alpha}) = \inf_{w,b,\xi_i} L(\boldsymbol{w}^*, b, \xi_i, \boldsymbol{\alpha}, \boldsymbol{\beta})$$

$$q(\boldsymbol{\alpha}) = L(\boldsymbol{w}^*, b^*, \xi_i^*, \boldsymbol{\alpha}, \boldsymbol{\beta})$$

Apply the outcome of (26) to $q(\boldsymbol{\alpha})$,

$$
\begin{aligned}
q(\boldsymbol{\alpha}) &= L(\boldsymbol{w}^*, b^*, \xi_i^*, \boldsymbol{\alpha}, \boldsymbol{\beta}) \\
q(\boldsymbol{\alpha}) &= \frac{1}{2}\boldsymbol{w^T w} + C\sum_{i=1}^{n} \xi_i - \sum_{i=1}^{n} \beta_i \xi_i \\
&\quad - \sum_{i=1}^{n} \alpha_i[y_i(\boldsymbol{w} \cdot \boldsymbol{x_i} + b) - 1 + \xi_i] \\
q(\boldsymbol{\alpha}) &= \frac{1}{2}\boldsymbol{w^T w} - \boldsymbol{w} \cdot \underbrace{(\sum_{i=1}^{n} \alpha_i y_i x_i)}_{w} + b \underbrace{\sum_{i=1}^{n} \alpha_i y_i}_{\text{equaltozero}} \\
&\quad + \sum_{i=1}^{n} \underbrace{(C - \alpha_i - \beta_i)}_{\text{equaltozero}} \xi_i + \sum_{i=1}^{n} \alpha_i \\
q(\boldsymbol{\alpha}) &= \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\boldsymbol{w^T w} \\
q(\boldsymbol{\alpha}) &= \sum_{i=1}^{n} \alpha_i - \frac{1}{2}(\sum_{i=1}^{n} \alpha_i y_i x_i)^T \left(\sum_{j=1}^{n} \alpha_j y_j x_j\right)
\end{aligned}
$$

$$\boxed{q(\boldsymbol{\alpha}) = \sum_{i=1}^{n} \alpha_j - \frac{1}{2}\sum_{j=1}^{n}\sum_{j=1}^{n} y_i y_j (x_i \cdot x_j)\alpha_i \alpha_j}$$

Next optimize $q(\boldsymbol{\alpha})$,

$$
\begin{aligned}
&\max_{\boldsymbol{\alpha}} \sum_{i=1}^{n} \alpha_j - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} y_i y_j (x_i \cdot x_j)\alpha_i \alpha_j \\
&\text{s.t.} \sum_{j=1}^{n} \alpha_j y_j = 0 \qquad (27) \\
&\text{and} \quad 0 \le \alpha_i \le C, \quad i = 1,2,\dots n
\end{aligned}
$$

We clearly notice that the Lagrange dual function for Hard Margin SVM and Soft Margin SVM are exactly the same except the constraints.

As shown above, Soft Margin SVM optimization problem (27) is same as the Hard Margin Optimization Problem (20) with an additional constraint that all the $\alpha_i$ are upper bounded by $C$. We also notice that $\xi_i$ does not exist in the final form of $q(\boldsymbol{\alpha})$. After the optimized $\alpha_i^*$ is obtained the decision function can be defined.

Decision Function $\boldsymbol{D}(\boldsymbol{x})$,

$D(x)$ for Soft Margin Classification is derived similar to the Hard Margin Classification by substituting for $w^*$ and $b^*$.

$$D(x) = \boldsymbol{w}^* \cdot \boldsymbol{x} + b^*$$

$$\boxed{D(x) = \sum_{j=1}^{l} \boldsymbol{\alpha_j}^* y_j (x_j \cdot x) + [y_i - \sum_{j=1}^{l} \boldsymbol{\alpha_j}^* y_j (x_j \cdot x_i)]}$$

Where $\{x_i, y_i\}$ is one of the support vector point which lies on either margin lines $(0 < \alpha_i^* < C)$, and $j = 1,2,\ldots l$ are the indices of all the support vectors $(0 < \alpha_i^* \leq C)$.

Using the Decision function the unknown $x$ could be classified as follows:

$$x \in \begin{cases} \text{Positive Class (Class 1)} & \text{if } D(x) > 0 \\ \text{Negative Class (Class 2)} & \text{if } D(x) < 0 \\ \text{Unclassified} & \text{if } D(x) = 0 \end{cases}$$

After we have trained the SVM with the training dataset we only need store the support vectors $(\alpha_i > 0)$ to classify any new pattern $x$.

### C. *Kernel SVM Formulation*

In Kernel SVM the training dataset is mapped into a higher dimensional feature space using $\phi$, and a separating hyperplane is constructed in the transformed feature space. Through Kernel functions, it is feasible to create a separating hyperplane without explicitly executing the map in the feature space.

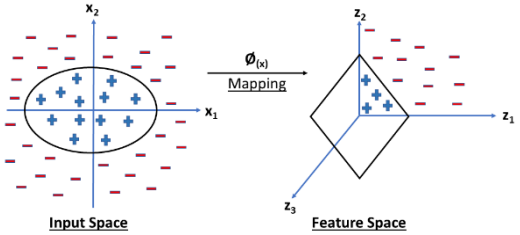


Fig. 5. Mapping from input space (2-D) to a feature space (3-D)

Have a look at the dual form of the optimization problem in (27), the key component is the dot product of $x_i$ and $x_j$, which could be seen as a similarity function. If we consider the transform feature space is a $Z$ space, then the dual form could be written as follows.

$$\max_{\alpha} \quad -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} y_i y_j (z_i \cdot z_j)\alpha_i \alpha_j + \sum_{j=1}^{n} \alpha_j$$
$$\text{s.t.} \quad \sum_{j=1}^{n} \alpha_j y_j = 0 \qquad (28)$$
$$\text{and} \quad 0 \leq \alpha_i \leq C \quad , i = 1,2,\ldots n$$

Let $\phi$ be the mapping function, so $z_i = \phi(x_i)$. Suppose we have a function K: $\mathbb{R}^d \times \mathbb{R}^d \longrightarrow \mathbb{R}$ such that $K(x_i, x_j) = (\phi(x_i) \cdot \phi(x_j))$ called Kernel function, and the computation of $K(x_i, x_j)$ is about as expensive as $(x_i \cdot x_j)$. If we have to compute $(\phi(x_i) \cdot \phi(x_j))$ it would be very expensive depending on the dimension of $\phi(x_i)$. Replacing $(z_i \cdot z_j)$ by $K(x_i, x_j)$ we could re-write the dual form as follows.

$$\max_{\alpha} \quad -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} y_i y_j K(x_i, x_j)\alpha_i \alpha_j + \sum_{j=1}^{n} \alpha_j$$
$$\text{s.t.} \quad \sum_{j=1}^{n} \alpha_j y_j = 0 \qquad (29)$$
$$\text{and} \quad 0 \leq \alpha_i \leq C \quad , i = 1,2,\ldots n$$

The main advantage of using kernels is that, it is not required to perform any tedious calculations in high-dimensional feature spaces. Common Kernels used in SVM are,

- Linear Kernel: $K(x_i, x_j) = (x_i \cdot x_j)$
- Polynomial Kernel: $K(x_i, x_j) = [(x_i \cdot x_j) + 1]^d$
- RBF Kernel: $K(x_i, x_j) = \exp[-\gamma \parallel x_i - x_j \parallel^2]$

## III. RESULTS

In this paper we have used the sparse datasets with large attributes. The experiment is to compare the classification performance using Machine Learning Algorithms such as Kernel SVM, Logistic Regression, Neural Networks, Bayesian Network, K-Nearest Neighbors (KNN), Bagging and Random Forest. We have used two datasets; Eplilepsy Dataset [13] and Basic Motion Dataset [14].

**Dataset1 [Epilepsy]**: This dataset contains $four$ classes and 207 attributes, the number of attributes are larger (151% of the total training instances) compared to the training instances of 137. The test instances are also 137, to have a more robust test.

TABLE I. DATASET: EPILEPSY

| Test Inst. | Attr. | classes | classifier | CCP |
|---|---|---|---|---|
| 137 | 207 | 4 | Kernel SVM | 89.85% |
| 137 | 207 | 4 | Logistic Regression | 32.61% |
| 137 | 207 | 4 | Neural Networks | 60.87% |
| 137 | 207 | 4 | Bayes Network | 56.52% |
| 137 | 207 | 4 | KNN | 68.84% |
| 137 | 207 | 4 | Bagging | 54.35% |
| 137 | 207 | 4 | Random Forest | 73.19% |

CPP - Correctly classified Points in percentage.

As shown in Table 1, the Kernel SVM has performed significantly better (CPP - 89.85%) than the other tested Algorithms.

**Dataset2 [Basic Motion]**: This dataset contains $four$ classes and 100 attributes, the number of attributes are much larger (250% of the total training instances) compared to the training instances of 40. The test instances are also kept at 40, to have a more robust test.

TABLE II. DATASET: BASICMOTION

| Test Inst. | Attr. | classes | classifier | CCP |
|---|---|---|---|---|
| 40 | 100 | 4 | Kernel SVM | 97.50% |
| 40 | 100 | 4 | Logistic Regression | 60.00% |
| 40 | 100 | 4 | Neural Networks | 60.00% |
| 40 | 100 | 4 | Bayes Network | 85.00% |
| 40 | 100 | 4 | KNN | 62.50% |
| 40 | 100 | 4 | Bagging | 67.50% |
| 40 | 100 | 4 | Random Forest | 80.00% |

CPP - Correctly classified Points in percentage.

As shown in Table 2, the Kernel SVM has performed significantly better (CPP - 97.50%) than the other tested Algorithms.

## IV. CONCLUSION

In this paper we have used the data from two sparse datasets and we have compared the performance with several Machine Learning Algorithms. Experimental results indicate that Kernel based Support Vector Machines performed significantly better on Sparse Datasets with Large Attributes.

If we could identify the support vectors, we can ignore the rest of the data points, this is one of the key characteristics of SVM, by incorporating the Kernel functions, classification is performed much efficiently in a higher dimensional feature space, thus SVM can efficiently handle sparse datasets and outperforms rest of the classification methods.

## REFERENCES

[1] Boser, B. E., Guyon, I., and Vapnik, V. (1992). A training algorithm for optimal margin classifiers. In Proceedings of the Fifth Annual ACM Workshop on Computational Learning Theory, pages 144-152. ACM, Madison, WI.

[2] C. Cortes and V. Vapnik. Support vector networks. Machine Learning, 20:273-297, 1995.

[3] Joachims T. (1998) Text categorization with Support Vector Machines: Learning with many relevant features. In: Nédellec C., Rouveirol C. (eds) Machine Learning: ECML-98. ECML 1998. Lecture Notes in Computer Science (Lecture Notes in Artificial Intelligence), vol 1398. Springer, Berlin, Heidelberg

[4] O. Chapelle, P. Haffner and V. N. Vapnik, "Support vector machines for histogram-based image classification," in IEEE Transactions on Neural Networks, vol. 10, no. 5, pp. 1055-1064, Sept. 1999.

[5] A. Ganapathiraju, J. E. Hamaker and J. Picone, "Applications of support vector machines to speech recognition," in IEEE Transactions on Signal Processing, vol. 52, no. 8, pp. 2348-2355, Aug. 2004.

[6] E. Osuna, R. Freund and F. Girosit, "Training support vector machines: an application to face detection," Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, San Juan, Puerto Rico, USA, 1997, pp. 130-136.

[7] A. Vedaldi and A. Zisserman, "Efficient additive kernels via explicit feature maps," 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, San Francisco, CA, 2010, pp. 3539-3546.

[8] C. P. Diehl and G. Cauwenberghs, "SVM incremental learning, adaptation and optimization," Proceedings of the International Joint Conference on Neural Networks, 2003., Portland, OR, 2003, pp. 2685-2690 vol.4.

[9] Mingzhi Li, Yong Liu and Junhua Wang, "A new parameter optimization algorithm of SVM," 2011 International Conference on Advanced Intelligence and Awareness Internet (AIAI 2011), Shenzhen, 2011, pp. 174-178.

[10] M. A. Aiman Ngadilan, N. Ismail, M. H. F. Rahiman, M. N. Taib, N. A. Mohd Ali and S. N. Tajuddin, "Radial Basis Function (RBF) tuned Kernel Parameter of Agarwood Oil Compound for Quality Classification using Support Vector Machine (SVM)," 2018 9th IEEE Control and System Graduate Research Colloquium (ICSGRC), Shah Alam, Malaysia, 2018, pp. 64-68.

[11] Selected applications of convex optimization. Heidelberg: Springer, 2015.

[12] Deng, Naiyang, Yingjie Tian, and Chunhua Zhang. Support vector machines : optimization based theory, algorithms, and extensions. Boca Raton: CRC Press, Taylor and Francis Group, 2013.

[13] Dataset1 Link: www.timeseriesclassification.com/description.php?Dataset=Epilepsy

[14] Dataset2 Link: www.timeseriesclassification.com/description.php?Dataset=BasicMotions