S21CSE471: Statistical Methods in AI **Spring 2021**

Support Vector Machines

Prepared by: Abhishek Sharma, Akansha Srivastava, Aayushi Nigam

1 Introduction

Support Vector Machine (SVM) is a machine learning model that takes a multidimensional vector and the class they belong to and creates a boundary between the different classes and their input, so that future data can be easily classified by inspecting the boundary which it falls in [1]. In mathematical terms, one or more hyper-planes are constructed, which acts as a boundary allowing classification to be performed. The objective is to find the optimal hyperplane (Figure 1) that is uniquely determined by a set of the vectors at equal distance from the hyperplane – the support vectors. The support vectors are selected from the training data so that the distance between the two hyperplanes passing through the support vectors x^+ and x^- is at a maximum. The optimal hyperplane maximizes the empty space (or margin) between all training data points.

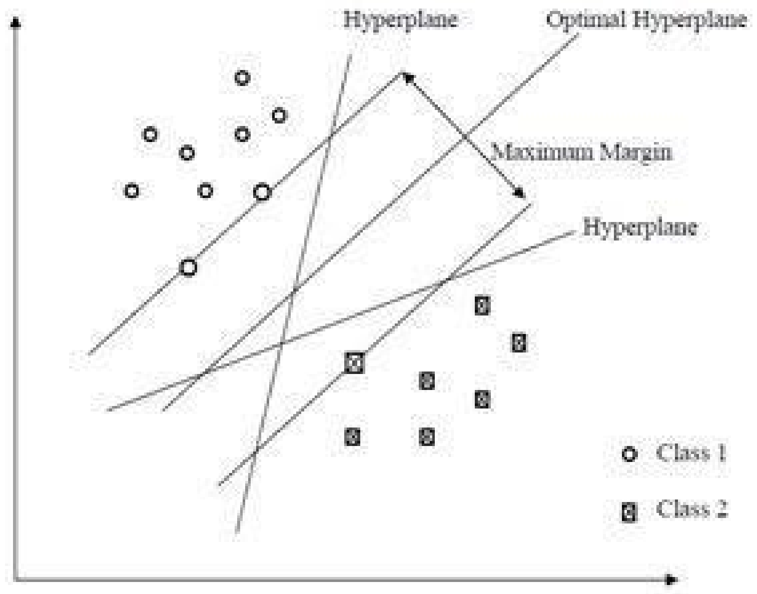


Figure 1: An optimal hyper-plane separating data points with maximum margin.

For the linearly separable case there will not be any training data between the two hyperplanes and the decision plane, which is midway between the two hyperplanes. Moreover, this optimal decision plane should minimize classification error (Figure 2). This has been proved by Vapnik [2].

From Figure 2, the decision hyperplane can be expressed as follows:

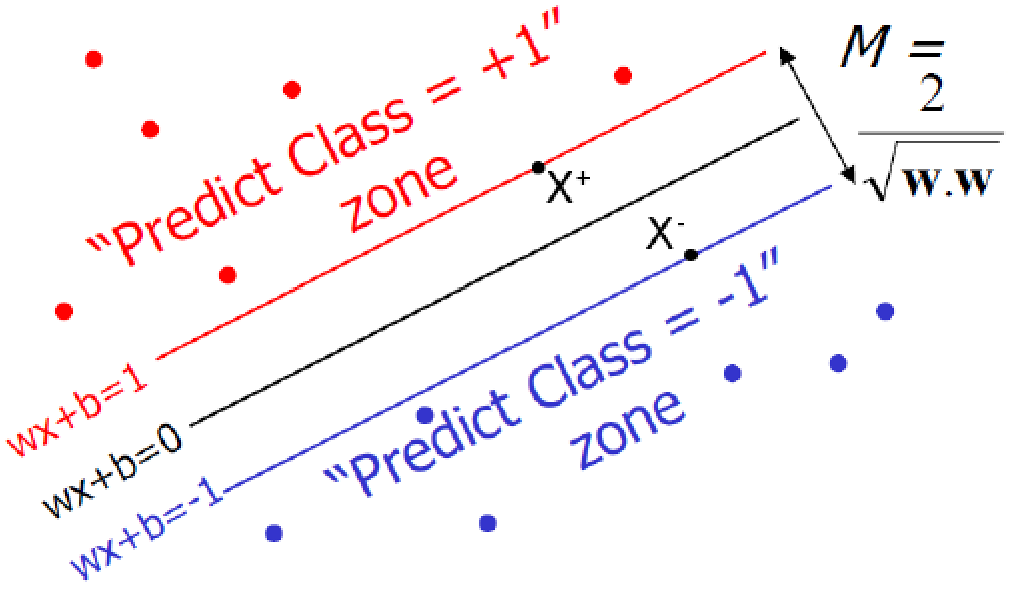


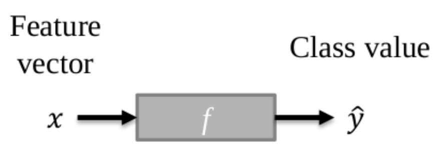
Figure 2: A sample linear SVM classifier and its margin[2].

$$x.w + b = 0 \tag{1}$$

All the n training data must satisfy the following constraint :

$$y_i(x_i w_i + b) \geq 1 \text{for i = 1, ..., n} \tag{2}$$

So, Classification Rule is $sign(x_i w_i + b)$. Positive value of $sign(x_i w_i + b)$ represents class 1, while, negative value represents class 2.



$$f(x, w, b) = sign(w \circ x + b)$$

$$= sign\ (\textstyle\sum w[i]x[i] + \text{b})$$

The SVM approach aims at maximizing the margin of separation which can be calculated as follows

$$M = \frac{w.(x^+ - x^-)}{||w||} = \frac{2}{||w||} \tag{3}$$

The decision boundary can be found by solving the following constrained optimization problem.

Minimise

$$\frac{1}{2}(||w||)^2$$

subject to :

$$y_i(w^T x_i + b) \geq 1 \; \forall i$$

This is a constrained optimization problem. Solving it requires to use Lagrange multipliers which converts above optimization problem into the dual form:

$$max \, W(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1,j=1}^{n} \alpha_i \alpha_j y_i y_j {x_i}^T x_j$$

subject to

$$\alpha_i \geq 0, \; \sum_{i=1}^{n} \alpha_i y_i = 0$$

This is a quadratic programming (QP) problem which can solved for the value of α_i.
Optimal w is

$$w = \sum_{i=1}^{n} \alpha_i y_i x_i$$

Many of the α_i are zero. x_i with non-zero α_i are called support vectors (SV). Therefore, the decision boundary is determined only by the SV.

2 Soft - Margin SVM

In real world problems, data is noisy i.e, data points are not completely separable by a linear classifier. For cases where the training data points are not clearly separable the margin can be relaxed to facilitate a more robust decision (refer Figure 3). In this case, the constraint is posed as follows:

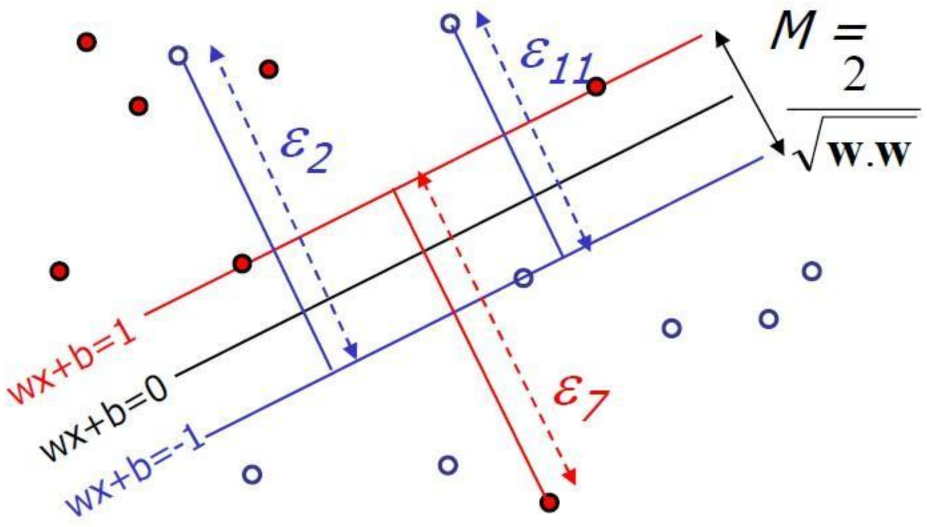


Figure 3: A sample linear SVM classifier with a soft margin.

$$y_i(x_i.w_i + b) \geq 1 - \varepsilon_i \;\; \text{for i = 1,...,n} \qquad (4)$$

where ε_i are slack variables which measure the deviation of data points from the marginal hyperplane and allow some data points to violate the initial constraint.
Therefore, the objective is to **minimise** :

$$\frac{1}{2}\,||w||^2 + C\sum \epsilon_i \tag{5}$$

subject to :

$$y_i(w^T x_i + b) \geq 1 - \xi_i, \;\; \xi_i \geq 0$$

where C is a parameter used to penalize violations of the margin. The value of C is defined constant or chosen by validation.
A typical and good approach to solving the above optimization problem is by introducing **Lagrange multipliers** $\alpha_i \geq 0$ and reformulating the optimization equation as a Lagrangian.

$$L = \frac{1}{2}w^T w + C\sum \xi_i + \sum_{i=1}^{n} \alpha_i(1 - \xi_i - y_i(w^T x_i + b)) - \sum_{i=1}^{n} \mu_i \xi_i$$

With positive α and μ Lagrange multipliers
At Optimum,

$$\frac{\partial L}{\partial w_j} = w_j - \sum_{i=1}^{n} \alpha_i y_i x_{ij} = 0 \implies \vec{w} = \sum_{i=1}^{n} \alpha_i y_i \vec{x_i} = 0$$

$$\frac{\partial L}{\partial \xi_j} = C - \alpha_j - \mu_j = 0 \implies C = \alpha_j + \mu_j$$

$$\frac{\partial L}{\partial b} = \sum_{i=1}^{n} \alpha_i y_i = 0$$

Using above equations, we get the dual form with slack variable

$$max\, W(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1,j=1}^{n} \alpha_i \alpha_j y_i y_j {x_i}^T x_j$$

subject to the following constraints :

$$\sum_{i=1}^{n} \alpha_i y_i = 0 \text{ where } 0 \leq \alpha_i \leq C \tag{6}$$

This is very similar to the optimization problem in the linear separable case, except that there is an upper bound C on α_i now.
Solve the dual form using QP solver for the value of α_i to obtain the value of w

$$w = \sum_{j=1}^{s} \alpha_{t_j} y_{t_j} X_{t_j} = 0$$

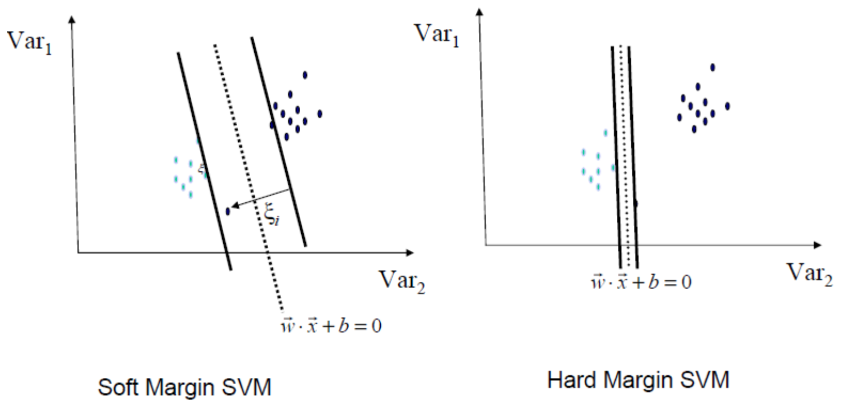
This formulation helps in two ways. First, the constraints on Lagrange multipliers are much simpler to handle. Secondly, the training data now only appears in the form of dot products of vectors.

The algorithm try to keep ξ null, maximising the margin. Here, The algorithm does not minimise the number of errors. Instead, it minimises the sum of distances from the hyperplane.

When C increases the number of errors tend to lower. At the limit of C tending to infinite, the solution tend to that given by the hard margin formulation, with 0 errors.

3 Non-linear SVM (Kernel SVM)

We considered a large-margin classifier with a linear decision boundary to handle the noises in the data. But, we can't apply linear svm to classify the non-linear data. For the cases where the data is not linearly separable, the key idea is to transform the data x_i to a higher dimensional space (figure 4). The transformation is done for two reasons:



- Non-linear operation in input space (the space and the point x_i are located) is equivalent to Linear operation in the feature space (the space of $f(x_i)$ after transformation).
- Classification can become easier with a proper transformation (figure 4).

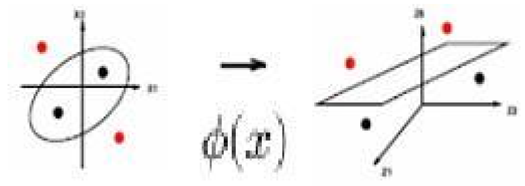


Figure 4: Mapping of data to a higher dimensional feature space.

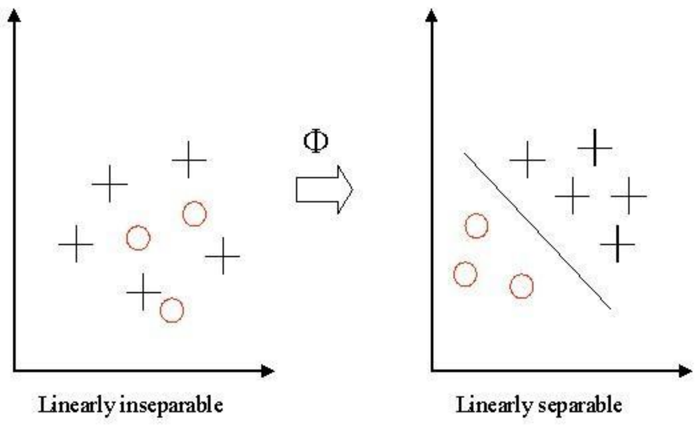


Figure 5: Use of Kernel

In the SVM optimization equation (Dual form), the data points only appear as dot products. If there is a kernel function, we do not need to compute the dot product explicitly. This introduces the concept of **kernel tricks** (Figure 5). A kernel is defined as follows (refer Eq. 7):

$$K(x, x^0) = \varphi(x).\varphi(x^0) \tag{7}$$

where $\varphi(x)$ is a function mapping x to a higher dimensional space (Figure 6). Since transforming data with $\varphi(x)$ can be expensive with high dimensional spaces. Instead, an SVM employs a kernel function k which gives the dot product of the two examples in the higher dimensional space without transforming them into that space. This notion, dubbed the kernel trick, allows us to perform the $\varphi(x)$ transformation for purposes of classification to large dimensional spaces. It is for this reason that SVM can handle large dimensional feature vectors without significant effect on performance. The kernel function, being an

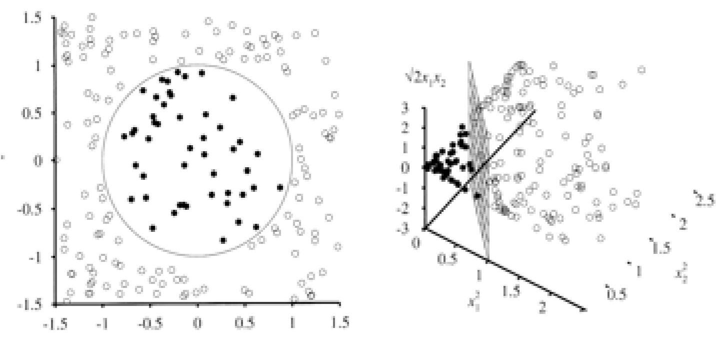


Figure 6: Feature Space Representation

inner product, can be considered as a similarity measure between the objects.
Dual Form with Kernel Function

$$max\, W(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1,j=1}^{n} \alpha_i \alpha_j y_i y_j K(x_i x_j)$$

subject to :

$$C \geq \alpha_i \geq 0 \, , \sum_{i=1}^{n} \alpha_i y_i = 0$$

For a test object z, the discriminant function essentially is a weighted sum of the similarity between z and a pre-selected set of objects (the support vectors)

$$f(z) = \sum_{x_i \in S} \alpha_i y_i K(z, x_i) + b$$

where, S : the set of support vectors
A function can be considered as kernel function when it must satisfy the Mercer's condition :

$$\forall g(x) \text{ such that } \int g^2(x)\, \mathrm{d}x \geq 0 \implies \int K(x,y)\, g(x)\, g(y)\, \mathrm{d}x\, dy \geq 0$$

In Support Vector typically one of the following Kernel functions are used:

Polynomial:

$$K(x_i, x_j) = (x_i.x_j)^d \tag{8}$$

Sigmoid:

$$K(x_i, x_j) = tanh(\gamma(x_i, x_j) + r) = (x_i.x_j)^d \tag{9}$$

where d,γ,r $\in$ R.

Gaussian:

$$K(x_i, x_j) = e^{-y*(||x^i - x^j||)^2} \tag{10}$$

4 Important Parameters

- **Kernel:** Type of kernel function to be used. Choosing the right kernel function is important and the most tricky part of SVM because it creates the kernel matrix, which summarizes all the data. In practice, a low degree polynomial kernel or RBF kernel with a reasonable width is a good initial try.
- **Kernel parameters :** For example, gamma (γ) - RBF kernel width in rbf kernel

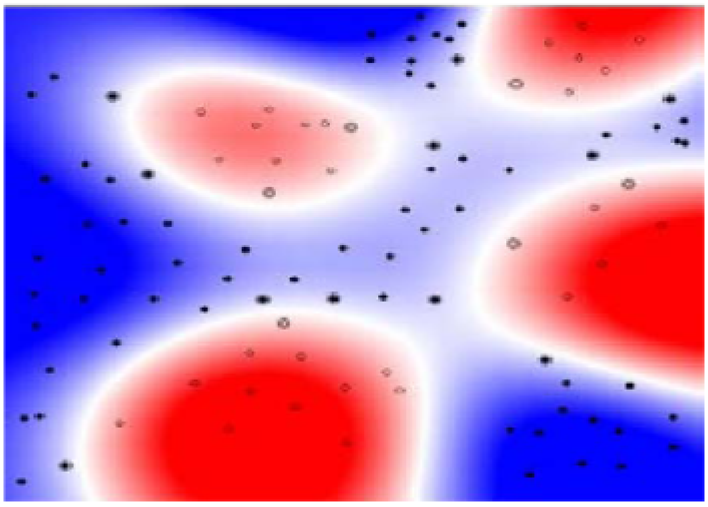


Figure 7: Non-linear Rbf kernel

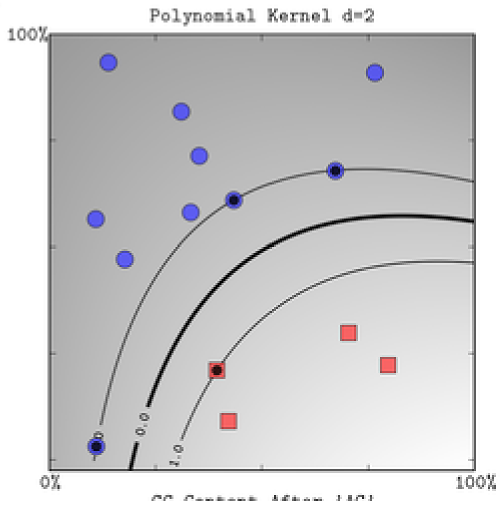


Figure 8: Polynomial Kernel

- **C (regularization parameter):** Very small value of C allows the svm classifier to do more misclassification and may result in the underfitting, while a very large value of C gives a strict classifier which classifies each individual point correctly and may overfit the training data. C is a tradeoff parameter between error and margin.Typically C and gamma are tuned at the same time.

5 Characteristics of SVM :

- The SVM learning problem can be formulated as a convex optimization problem, in which efficient algorithms are available to find the global minimum of the objective function.
- SVM is the best suitable to classify both linear as well as non-linear training data efficiently
- SVM can be applied to categorical data by introducing a suitable similarity measures.
- Computational complexity is influenced by number of training data not the dimension of data.

6 Strengths and Weaknesses of SVM

Strength :

- Can perform well on a range of datasets.
- Training is relatively easy.
- Versatile: different kernel functions can be defined for specific data types.
- Tradeoff between classifier complexity and error can be controlled explicitly.
- It works well for both low-and high-dimensional data.

Weakness :

- Need to choose a "good" kernel function.
- Interpretation is difficult.
- Needs careful normalization of input data and parameter tuning.

References

[1] Johan A.K. Suykens and Joos Vandewalle. Least squares support vector machine classifiers. Neural processing letters, 9(3):293–300, 1999.

[2] V. Vapnik, Estimation of Dependences Based on Empirical Data. New York, USA: Springer Verlag, 1982.