Data Analytics

Course Taught at IIFT

Lecture 14: Support Vector Machines

Tanujit Chakraborty

www.ctanujit.org

Today's Topics

- Introduction to SVM
- Concept of Maximum Margin Hyperplane (MMH)
- Computation of the margin of a MMH
- Formulation of the optimization problem for linear SVM
- The Lagrangian multiplier method
- Building model for linear SVM
- Multiclass classification
- Non-linear SVM
  - Concept of non-linear data
  - Φ-transformation
- Dual formation of optimization problem
- Kernel tricks and kernel functions

Introduction to Support Vector Machine

Support Vector Machine

Introduction

- A classification that has received considerable attention is support vector machine and popularly abbreviated as SVM.
- This technique has its roots in statistical learning theory *(Vlamidir Vapnik, 1992).*
- As a task of classification, it searches for the optimal hyperplane (also called decision boundary) separating the tuples of one class from another.

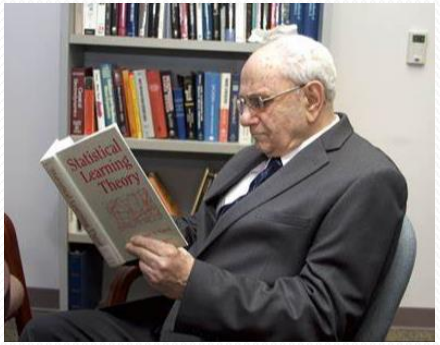


Fig. 0: Vapnik with his SLT Book

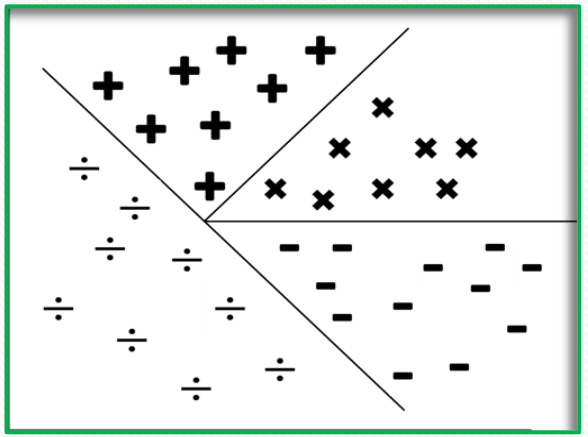


Fig. 1: Decision boundary in SVM

Support Vector Machine

Advantages of SVM

- SVM works well with higher dimensional data and thus avoids dimensionality problem.
- Although the SVM based classification (i.e., training time) is extremely high, the result, is however highly accurate. Further, testing an unknown data is very fast.
- SVM is less prone to overfitting than other methods.
- It also facilitates compact model for classification.

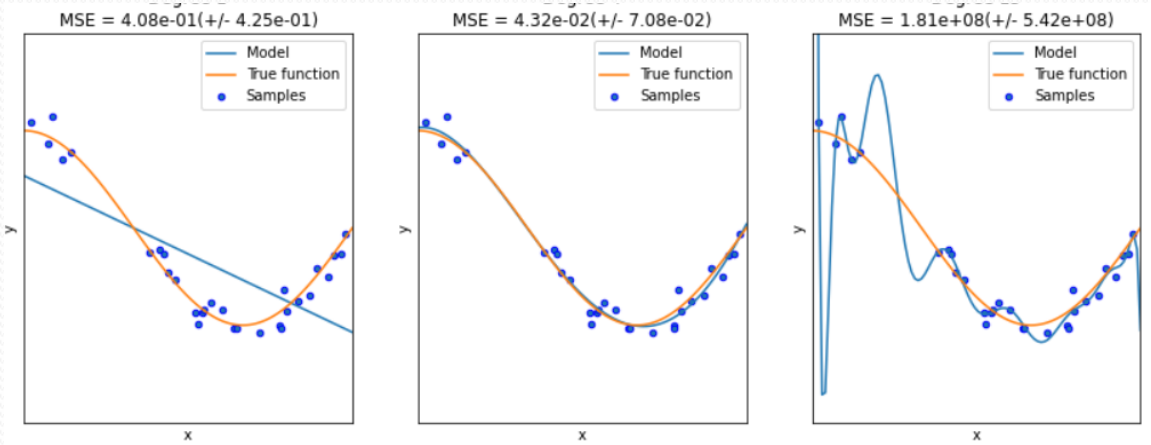
Overfitting and Underfitting Issues

Underfitting

- Refers to a model that can not fit many of the data in the training dataset. See Fig. 2(a)
- In this case, both training and testing errors are high

Overfitting

- Refers to a modeling that occurs when a model corresponds too closely to a dataset. See Fig. 2(c)
- In this case, the training error is low; however, testing error is high as the model may not work with unseen data.



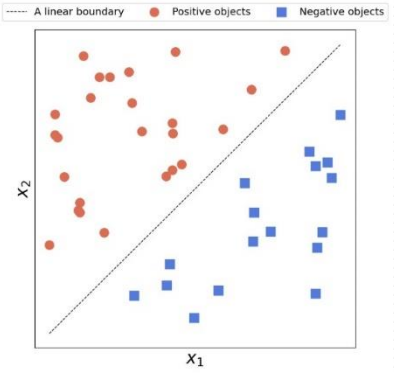
(a) Underfitting (b) Optimum (c) Overfitting

Fig. 2: Building models with training data set

Support Vector Machines

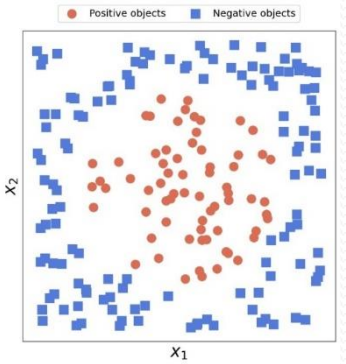
Types of SVMs

- **Linear SVM:** A classification technique when training data are linearly separable.
- **Non-linear SVM:** A classification technique when training data are linearly non-separable.



Linearly separable

Linear SVM



Linearly non-separable

Non-linear SVM

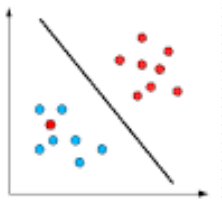
Fig. 3: Types of SVMs

Maximum Margin Hyperplane

Decision Boundaries

Decision boundary: A straight line

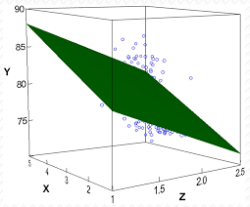
When the data are described in terms of two attributes (i.e., 2D data).



$ax + by + c = 0$

Decision boundary: A plane

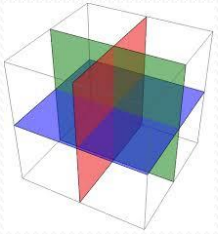
When the data are described in terms of three attributes (i.e., 3D data)



$ax + by + cz + d = 0$

Decision boundary: A hyperplane

When the data are described in terms of more than three attributes (e.g., 4D, 5D, data, etc.).



$a_1x_1 + a_2x_2 + a_3x_3 + \cdots + b = 0$

Classification Problem

Formulation of the classification problem

In our subsequent discussion, we shall assume a simplistic situation that given a **training data** $D = \{t_1, t_2, \ldots\ldots, t_n\}$ with a set of n tuples, which belong to two classes either $+$ or $-$ and each tuple is described by two attributes, say A_1, A_2.

Important Assumption

For this current example, we are considering that the data is **linearly separable.** So the current model is **linear SVM model**.

Classification with Hyperplane

Visualization of the dataset

Figure 4 shows a plot of data in 2D.

- The data is linearly separable.

⇒ we can find a hyperplane (in this case, it is a straight line) such that all +'s reside on one side, whereas all −'s reside on other side of the hyperplane

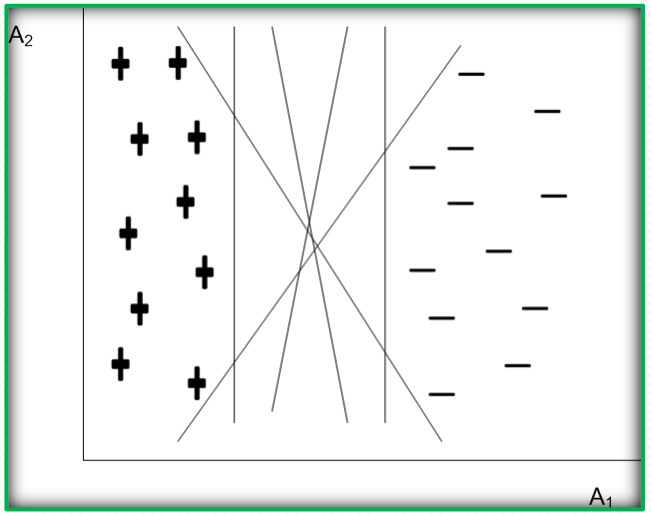


Fig. 4: Linear separation of a 2D data by hyperplanes

Classification with Hyperplane

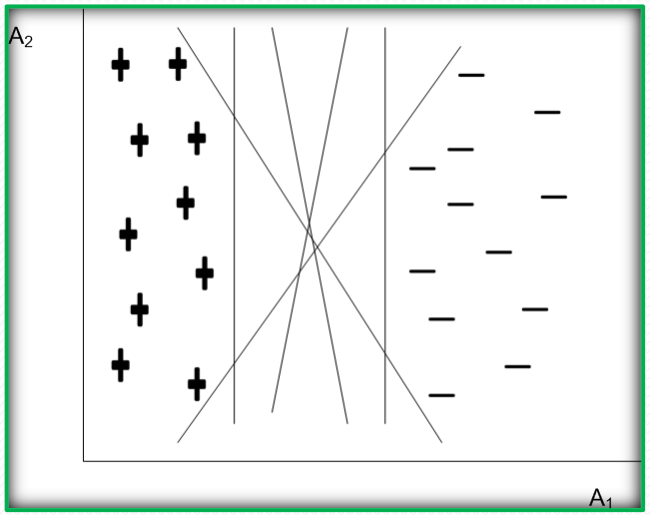


Fig. 4: Linear separation of a 2D data by hyperplanes

Possible Hyperplanes that separates the classes

- There are an infinite number of separating lines that can be drawn. Following two questions arise:

1. Whether all hyperplanes are equivalent so far the classification of data is concerned?
2. If not, which hyperplane is the best?

Classification with Hyperplane

Need for selecting the best hyperplane

- We may note that so far the classification error is concerned (with training data), all of them are with zero error (for training data).
- However, there is no guarantee that all hyperplanes perform equally well on unseen (i.e., test) data.
- Thus, for a good classifier, it must choose one of the infinite number of hyperplanes, so that it performs better not only on training data but as well as test data

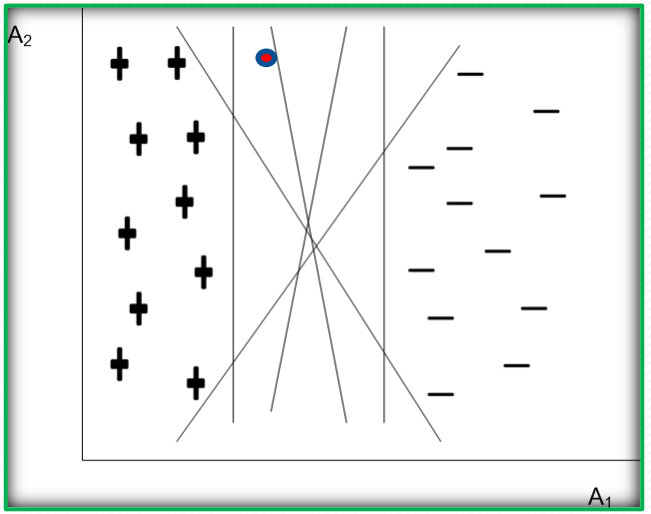


Fig. 4: Linear separation of a 2D data by hyperplanes

Classification with Hyperplane

Example: Comparing two possible hyperplanes

- To illustrate how the different choices of hyperplane influence the classification error, consider any arbitrary two hyperplanes H_1 and H_2 as shown in Fig. 5

- These two hyperplanes H_1 and H_2 have their own boundaries (denoted as b_{11} and b_{12} for H_1 and b_{21} and b_{22} for H_2).

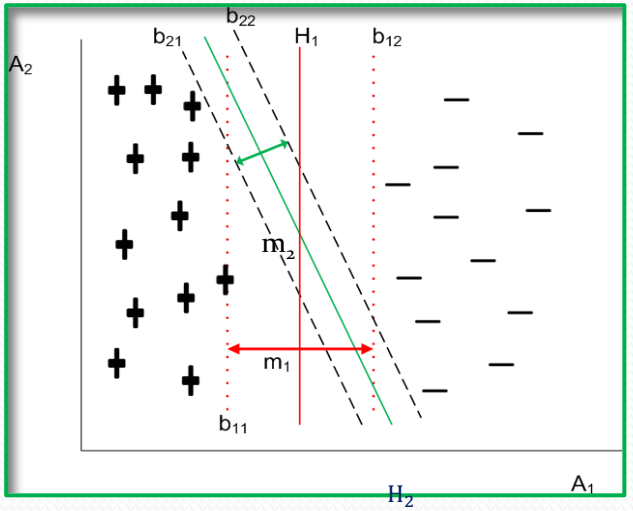


Fig. 5: Hyperplanes with decision boundaries and their margins.

Hyperplane and Margin

Definition: Decision boundary and margin

Decision boundary: A decision boundary is a boundary which is parallel to hyperplane and touches the closest class in one side of the hyperplane.

Margin: The distance between the two decision boundaries of a hyperplane is called the margin. So, if data is classified using hyperplane H_1, then it is with larger margin then using hyperplane H_2 .

The margin of hyperplane implies the error in classifier. In other words, the larger the margin, lower is the classification error.

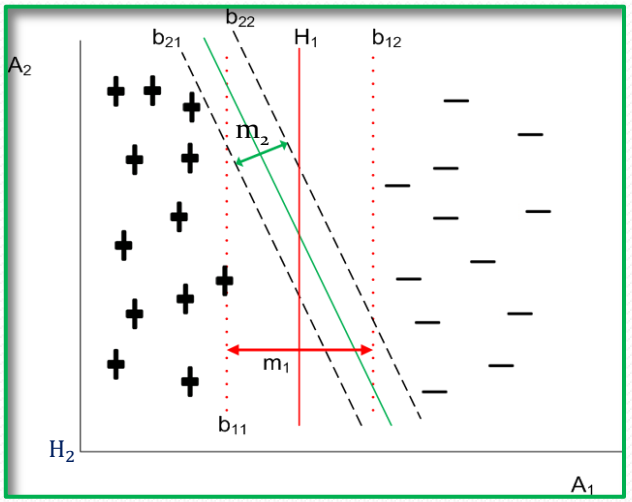


Fig. 5: Hyperplanes with decision boundaries and their margins.

Maximum Margin Hyperplane

Concept

- Intuitively, the classifier that contains hyperplane with a small margin is more susceptible to model overfitting and tend to classify with weak confidence on unseen data.
- Thus during the training or learning phase, the approach would be to **search for the hyperplane with maximum margin**
- Such a hyperplane is called **Maximum Margin Hyperplane (MMH)**.
- We may note the shortest distance from a hyperplane to one of its decision boundary is equal to the shortest distance from the hyperplane to the decision boundary at its other side.
- Alternatively, hyperplane is at the middle of its decision boundaries.

Linear SVM Model

Linear SVM

Introduction

- A SVM which is used to classify data which are linearly separable is called linear SVM.
- In other words, a linear SVM searches for a hyperplane with the maximum margin.
- This is why a linear SVM is often termed as a maximal margin classifier (MMC).

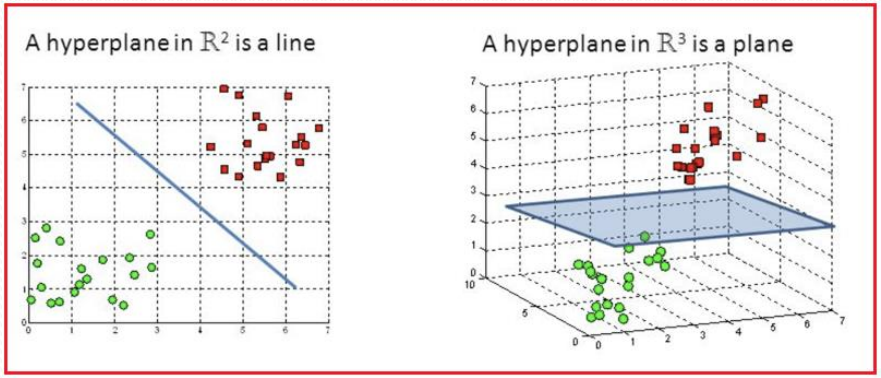


Fig. 6: Linear SVMs

Classification Problem and MMH

Notations

Finding the MMH for linear SVM for a given training data:

- Consider a binary classification problem consisting of n training data
- **Notation of the training data**: Each tuple is denoted by (X_i, Y_i) where $X_i = (x_{i1}, x_{i2}, \dots, x_{im})$ corresponds to the attribute set for the i^{th} tuple (data in m −dimensional space) and $Y_i \in [+, -]$ denotes its class label
- **Note:** The choice of which class should be labeled as $+$ or $-$ is arbitrary
- **Goal:** Given (X_i, Y_i), $i = 1,2, \dots, n$, we are to obtain a hyperplane which separates all into two sides of it (of course with maximum margin)

Representation of MMH

Equation of a hyperplane: The 2D case

Before, going to a general equation of a plane in n −dimension, let us consider first, a hyperplane in $2D$ plane.

- Let us consider a $2D$ training tuple with attributes A_1 and A_2 as $X = (x_1, x_2)$, where x_1 and x_2 are values of attributes A_1 and A_2, respectively for X

- Equation of a plane in $2D$ space can be written as

$$w_0 + w_1 x_1 + w_2 x_2 = 0 \qquad [e.g., ax + by + c = 0]$$

where, w_0, w_1, w_2 are the coefficients defining the slope and the intercept of the line

MMH and Classification

Classification with a hyperplane: The 2D case

- Equation of a hyperplane in $2D$ space
$$w_0 + w_1 x_1 + w_2 x_2 = 0$$
- Any point lying above such a hyperplane satisfies
$$w_0 + w_1 x_1 + w_2 x_2 > 0$$
- Similarly, any point lying below the hyperplane satisfies
$$w_0 + w_1 x_1 + w_2 x_2 < 0$$

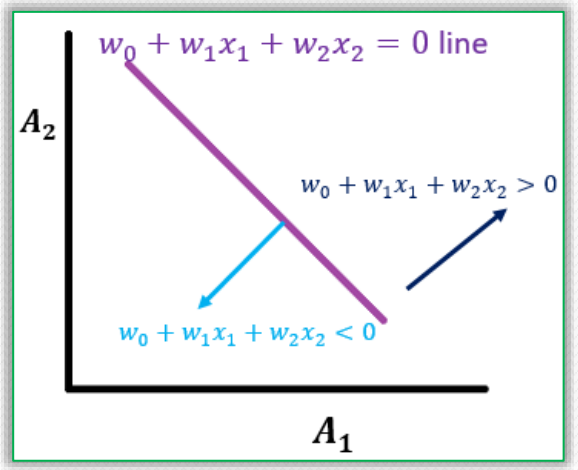


Fig. 7: Linear SVM to test

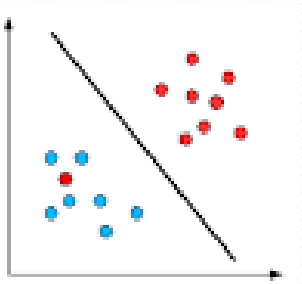
Representation of MMH

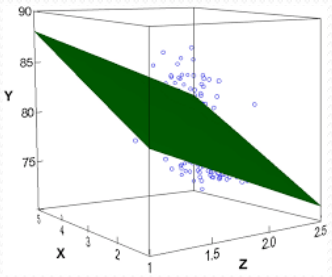
Generalization to higher dimensions

- An SVM hyperplane is an n −dimensional generalization of a straight line in $2D$.
- It can be visualized as a plane surface in $3D$, but it is not easy to visualize when dimensionality is greater than 3!
- In fact, Euclidean equation of a hyperplane in R_m is

$$w_1x_1 + w_2x_2 + \cdots + w_mx_m = b$$

where w_i's are the real numbers and b is a real constant (called the intercept, which can be positive or negative).





Representation of MMH

Equation of a hyperplane in matrix form

- In matrix form, a hyperplane can be represented as

$$W.X + b = 0$$

where $W = [w_1, w_2, \dots, w_m]$ and $X = [x_1, x_2, \dots, x_m]$ and b is a real constant.

- Here, W and b are parameters of the classifier model to be learned given a training data set D

$$[w_1, w_2, \dots, w_m] \begin{bmatrix} x_1 \\ x_2 \\ \cdot \\ \cdot \\ \cdot \\ x_m \end{bmatrix}$$

Margin of MMH

Finding the margin of a hyperplane

- Consider a 2D training set consisting of two classes + and − as shown in Fig. 8
- Suppose, b_1 and b_2 are two decision boundaries above and below a hyperplane, respectively
- Consider any two points X_+ and X_- as shown

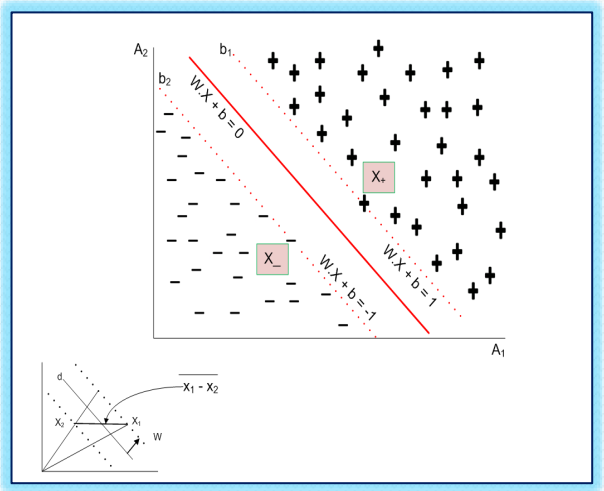


Fig. 8: Margin of the MMH

Margin of MMH

Finding the margin of a hyperplane

- For X_+ located above the decision boundary, the equation can be written as

$$W.X_+ + b = K\,; K > 0$$

- Similarly, for any point $X\_$ located below the decision boundary, the equation is

$$W.X\_ + b = K'; K' < 0$$

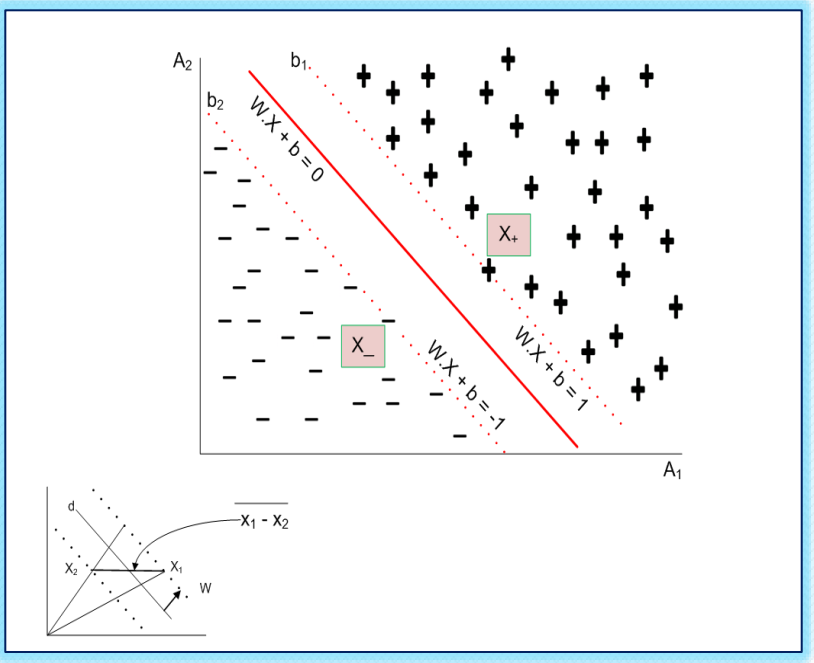


Fig. 8: Margin of the MMH

Margin of MMH

Finding the margin of a hyperplane

- Thus, if we label all +'s are as class label + and all −'s are class label −, then we can predict the class label Y for any test data X as

$$Y = \begin{cases} +, if\ W.X + b > 0 \\ -, if\ W.X + b < 0 \end{cases}$$

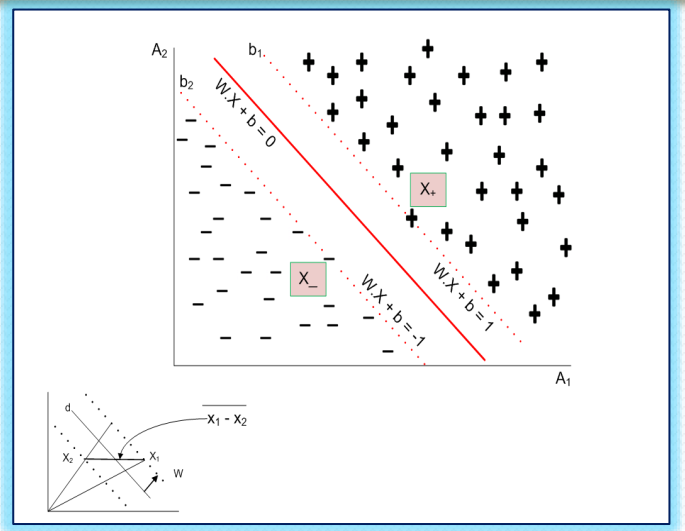


Fig. 8: Margin of the MMH

How to find the margin of the MMH : W.X+b = o?

Computing Margin of a MMH

MMH for Linear SVM

RECAP: Equation of the hyperplane in matrix form

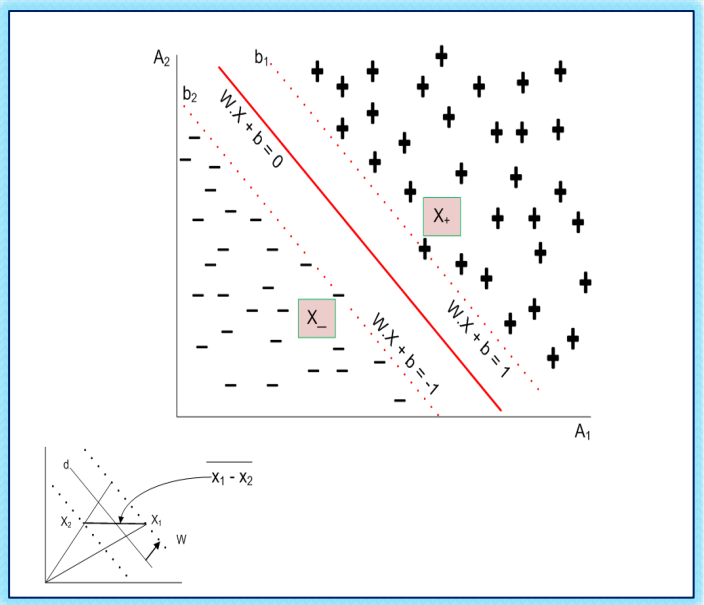
We learnt,
The equation of the hyperplane in matrix form is:

$$\boldsymbol{W.X + b = 0}$$

where $W = [w_1, w_2, \dots, w_m]$ and $X = [x_1, x_2, \dots, x_m]$ and b is a real constant.

- Thus, if we label all +'s are as class label + and all −'s are class label −, then we can predict the class label Y for any test data X as

$$\boldsymbol{Y} = \begin{cases} +, \boldsymbol{if\ W.X + b > 0} \\ -, \boldsymbol{if\ W.X + b < 0} \end{cases}$$



$$[w_1, w_2, \dots, w_m] \begin{bmatrix} x_1 \\ x_2 \\ . \\ . \\ . \\ x_m \end{bmatrix}$$

Fig. 8: Margin of the MMH

Computation of margin of MMH

Hyperplane and its margin

- In the equation $W.X + b = 0$, W represents orientation, and b represents the intercept of the hyperplane from the origin

  ⇒ If both W and b are scaled (up or down) by dividing a non zero constant, we get the same hyperplane

  ⇒ There can be infinite number of solutions using various scaling factors, all of them geometrically representing the same hyperplane

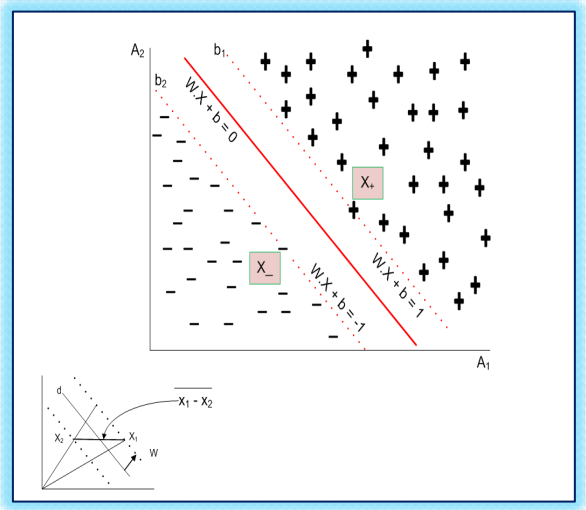


Fig. 8: Margin of the MMH

Margin of MMH

Hyperplane and its margin

- To avoid such a confusion, we can make W and b unique by adding a constraint that $W'.X + b' = \pm 1$ for data points on boundary of each class
- It may be noted that $W'.X + b' = \pm 1$ represents two hyperplanes parallel to each other.
  - For clarity in notation, we write this as $W.X + b = \pm 1$.

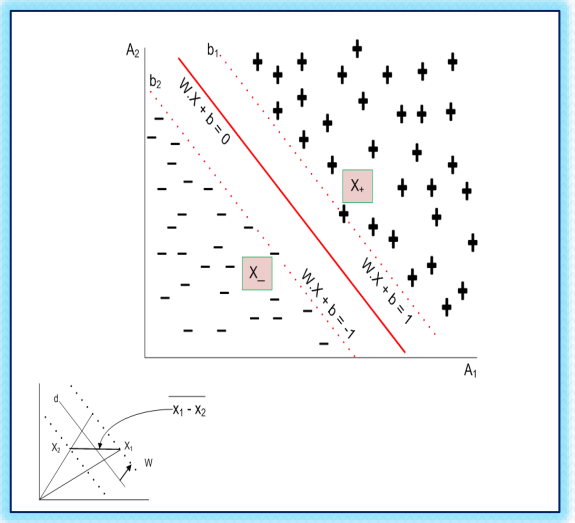


Fig. 8: Margin of the MMH

Having this understating, now we are in the position to calculate the margin of a hyperplane

Computation of Margin of a MMH

Computing the margin: Method-1

Suppose, X_1 and X_2 are two points on the decision boundaries b_1 and b_2, respectively. Thus,

$$W.X_1 + b = 1$$

$$W.X_2 + b = -1$$

or

$$W.(X_1 - X_2) = 2$$

This represents a dot (.) product of two vectors W and $(X_1 - X_2)$. Thus, taking magnitude of these vectors, the equation obtained is

$$\boldsymbol{d = \frac{2}{\|W\|}}$$

where, $\|W\| = \sqrt{w_1^2 + w_2^2 + \cdots + w_m^2}$, in a m-dimensional space.

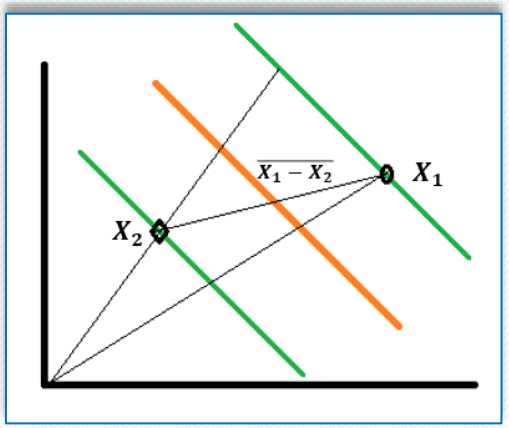


Fig. 9: Computation of the margin (Method 1)

Computation of Margin of a MMH

Computing the margin: Method-2

Consider two parallel hyperplanes H_1 and H_2. Let the equations of the hyperplanes be

$$H_1: w_1x_1 + w_2x_2 - b_1 = 0$$
$$H_2: w_1x_1 + w_2x_2 - b_2 = 0$$

To draw a perpendicular distance d between H_1 and H_2, we draw a right-angled triangle ABC as shown in Fig. 10.

⇒ Being parallel, the slope of H_1 (and H_2) is

$$\tan\theta = -\frac{w_1}{w_2}$$

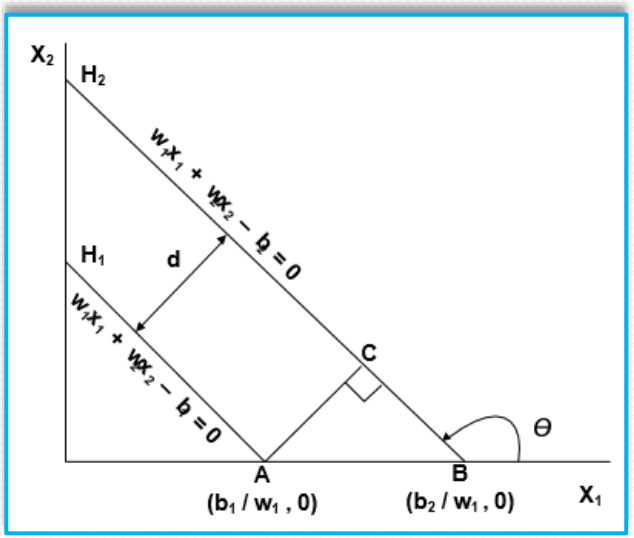


Fig. 10: Computation of the MMH (Method-2)

Computation of Margin of a MMH

Computing the margin: Method-2

Again in ΔABC, AB is the hypotenuse and AC is the perpendicular distance between H_1 and H_2

$\Rightarrow \sin(180 - \theta) = \frac{AC}{AB}$ or, $AC = AB.sin\theta$

Again, $AB = \frac{b_2}{w_1} - \frac{b_1}{w_1} = \frac{|b_2 - b_1|}{w_1}$;

and, $\tan\theta = -\frac{w_1}{w_2}$, $sin\theta = \frac{w_1}{\sqrt{w_1^2 + w_2^2}}$

Hence,

$$AC = \frac{|b_2 - b_1|}{\sqrt{w_1^2 + w_2^2}}$$

For n-dimensional space, this equation of margin, can be generalized as,

$$d = \frac{|b_2 - b_1|}{\sqrt{w_1^2 + w_2^2 + \cdots + w_n^2}} = \frac{|b_2 - b_1|}{\|W\|}$$

In SVM literature, this margin is famously written as $\mu(W, b)$

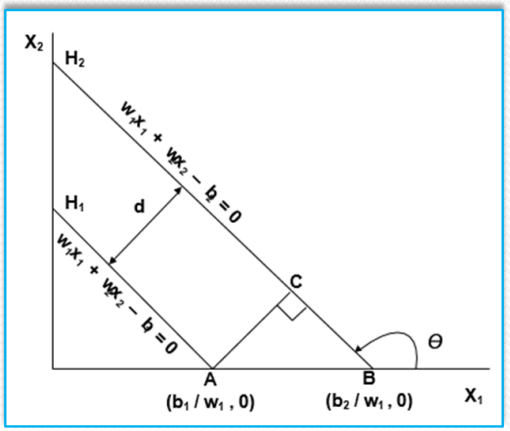


Fig. 10: Computation of the MMH (Method-2)

Finding MMH

Finding MMH of a SVM

Mathematical treatment for MMH

- The training phase of SVM involves estimating the parameters W and b for a hyperplane from a given training data.
- The parameters must be chosen in such a way that the following two inequalities are satisfied:

$$W.x_i + b \geq 1 \qquad if, y_i = 1$$
$$W.x_i + b \leq -1 \qquad if, y_i = -1$$

- These conditions impose the requirements that all training tuples from class $Y = +$ must be located on or above the hyperplane $W.x + b = 1$, while those instances from class $Y = -$ must be located on or below the hyperplane $W.x + b = -1$ (check Fig. 11)

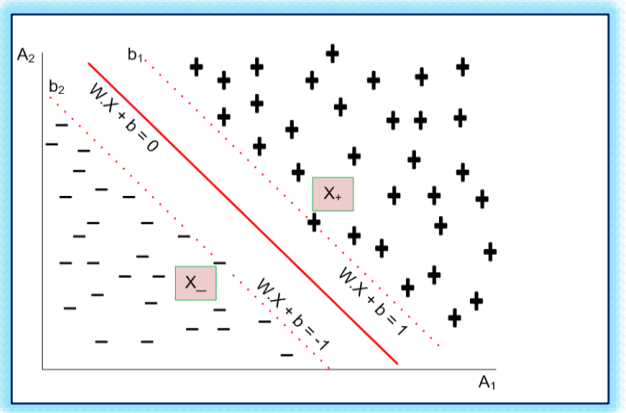


Fig. 11: Classification with MMH

Computing MMH of a SVM

Mathematical treatment for MMH

- The parameters must be chosen in such a way that the following two inequalities are satisfied:

$$W.x_i + b \geq 1 \qquad if, y_i = 1$$
$$W.x_i + b \leq -1 \qquad if, y_i = -1$$

- Both these inequalities can be summarized as,

$$y_i(W.x_i + b) \geq 1 \qquad \text{for all } i, i = 1, 2, \ldots, n$$

- Note that any tuples that lie on the hyperplanes H_1 and H_2 are called **support vectors**.
- Essentially, the support vectors are the most difficult tuples to classify and give the most information regarding classification.
- In the following, we discuss the approach of finding MMH and the support vectors.

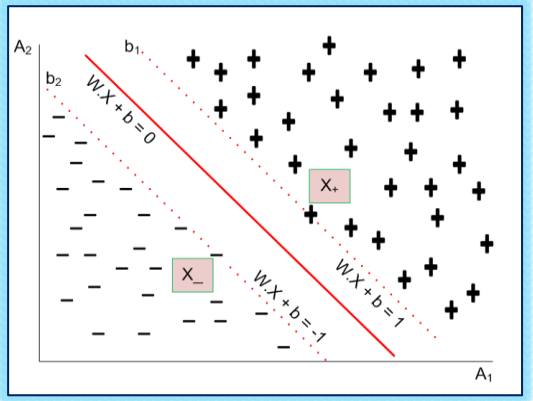


Fig. 12: Support vectors

Computing MMH of a SVM

Formulation of the problem

- We need to maximize the margin, i.e., $maximize\ \mu(W,b) = \frac{2}{\|W\|}$
- This is equivalent to minimizing the following objective function,

$$minimize\ \ \bar{\mu}(W,b) = \frac{\|W\|}{2}$$

- In nutshell, the learning task in SVM, can be formulated as the following constrained optimization problem:

$$\begin{aligned} & minimize\ \ \bar{\mu}(W,b) \\ subject\ to, \quad & y_i(W.x_i + b) \geq 1, \qquad i = 1, 2, \ldots, n \end{aligned}$$

Solving the Optimization Problem

Formulation of the problem

The learning task in SVM, can be formulated as the following constrained optimization problem:

$$minimize\ \mu`(W,b)$$

$$subject\ to, \quad y_i(W.x_i + b) \geq 1, \quad i = 1,2,\dots,n$$

- The above stated constrained optimization problem is popularly known as **convex optimization problem**, where objective function is quadratic and constraints are linear in the parameters W and b.
- The well known technique to solve a convex optimization problem is the standard **Lagrange multiplier method (LMM)**.
- First, let us learn the LMM, then we shall discuss solving of our own SVM problem.

Lagrange Multiplier Method

Lagrange Multiplier Method

The techniques

The Lagrange multiplier method follows two different steps depending on type of constraints.

1. **Equality Constraint Optimization Problem:** In this case, the problem is of the form

$$minimize\ f(x_1, x_2, \dots, x_d)$$
$$subject\ to, g_i(x) = 0, i = 1,2, \dots, p$$

2. **Inequality Constraint Optimization Problem:** In this case, the problem is of the form

$$minimize\ f(x_1, x_2, \dots, x_d)$$
$$subject\ to, h_i(x) \leq 0, i = 1,2, \dots, p$$

Lagrange Multiplier Method: Case 1

Case 1: Equality constraint optimization problem solving

1) Define the Lagrangian as follows:

$$L = (X, \lambda) = f(X) + \sum_{i=1}^{p} \lambda_i g_i(x)$$

where, λ_i are dummy variables called Lagrangian multipliers

2) Set the first order derivatives of the Lagrangian with respect to x and the Lagrangian multipliers λ_i to zero's. That is

$$\frac{\delta L}{\delta x_i} = 0, \qquad i = 1, 2, \dots, d$$

$$\frac{\delta L}{\delta \lambda_i} = 0, \qquad i = 1, 2, \dots, p$$

3) Solve $(d + p)$ equations to find optimal value of $X = [x_1, x_2, \dots, x_d]$ and λ_is

$$minimize\ f(x_1, x_2, \dots, x_d)$$

$$subject\ to$$

$$g_i(x) = 0, i = 1, 2, \dots, p$$

Lagrange Multiplier Method: Case 1

Example: Equality constraint optimization problem solving

Problem: Suppose, we need to $minimize\ f\ (x, y\ ) =\ x\ +\ 2y$

$$subject\ to\ \ x^2 + y^2 -\ 4\ =\ 0$$

Solution:

1) Lagrangian: $L(x, y, \lambda) = x + 2y + \lambda(x^2 + y^2 - 4)$

2) $\frac{\delta L}{\delta x} = 1 + 2\lambda x = 0;\ \frac{\delta L}{\delta y} = 2 + 2\lambda y = 0;\ \frac{\delta L}{\delta \lambda} = x^2 + y^2 - 4 = 0$

3) Solving the above three equations for x , y and λ, we get

$$x = \pm\frac{2}{\sqrt{5}} \quad y = \pm\frac{4}{\sqrt{5}} \quad \lambda = \pm\frac{\sqrt{5}}{4}$$

Lagrange Multiplier Method: Case 1

Example: Equality constraint optimization problem solving

Problem: Suppose, we need to $minimize\ f\ (x, y\ ) = x + 2y$

$subject\ to\ x^2 + y^2 - 4 = 0$

Solution: There are 2^3 choices:

When, $\lambda = +\frac{\sqrt{5}}{4}$, $x = -\frac{2}{\sqrt{5}}$ $y = -\frac{4}{\sqrt{5}}$

So, we get $f(x, y) = -\frac{10}{\sqrt{5}}$

… . . . . . .

When, $\lambda = -\frac{\sqrt{5}}{4}$, $x = \frac{2}{\sqrt{5}}$ $y = \frac{4}{\sqrt{5}}$

So, we get $f(x, y) = \frac{10}{\sqrt{5}}$

… . . . . . .

You can check that the function $f(x, y)$ has its minimum value at $x = -\frac{2}{\sqrt{5}}$ $y = -\frac{4}{\sqrt{5}}$

Lagrange Multiplier Method: Case 2

Case 2: Inequality constraint optimization problem solving

- The method for solving this problem is quite similar to the Lagrange multiplier method described previously.
- It starts with the Lagrangian

$$L = f(x) + \sum_{i=1}^{p} \lambda_i . h_i(x)$$

- In addition to this, it introduces additional constraints, called **Karush-Kuhn-Tucker (KKT)** constraints, which are stated in the next slide.

$$minimize\ f(x_1, x_2, \dots, x_d)$$

$$subject\ to$$

$$h_i(x) \leq 0, i = 1,2, \dots, p$$

Lagrange Multiplier Method: Case 2

Inequality constraint optimization problem solving: KKT constraints

$$\frac{\delta L}{\delta x_i} = 0, i = 1,2, \dots, d$$

$$\lambda_i \geq 0, i = 1,2, \dots, p$$

$$h_i(x) \leq 0, i = 1,2, \dots, p$$

$$\lambda_i h_i(x) = 0, i = 1,2, \dots, p$$

Solving the above equations, we can find the optimal value of $f(x)$.

Lagrange Multiplier Method: Case 2

Example: Inequality constraint optimization problem solving

Problem: Suppose, we need to $minimize\ f\ (x, y\ ) =\ (x-1)^2 + (y-3)^2$

$$subject\ to\ x + y \leq 2, y \geq x$$

Solution:

1) The Lagrangian for this problem is

$$L = (x-1)^2 + (y-3)^2 + \lambda_1(x + y - 2) + \lambda_2(x - y)$$

2) Subject to the KKT constraints:

$$\frac{\delta L}{\delta x} = 2(x-1) + \lambda_1 + \lambda_2 = 0$$

$$\frac{\delta L}{\delta y} = 2(y-3) + \lambda_1 - \lambda_2 = 0$$

$$\lambda_1(x + y - 2) = 0$$

$$\lambda_2(x - y) = 0$$

$$\lambda_1, \lambda_2 \geq 0$$

$$(x + y) \leq 2, \qquad y \geq x$$

Lagrange Multiplier Method: Case 2

Example: Inequality constraint optimization problem solving

Solution:

3) To solve KKT constraints, we have to check the following tests:

Case 1: $\lambda_1 = 0, \lambda_2 = 0$

$$2(x-1) = 0\,; 2(y-3) = 0 \quad \Rightarrow x = 1, y = 3$$

Since, $x + y = 4$, it violates $x + y \leq 2$; This is not a feasible solution.

Case 2: $\lambda_1 = 0, \lambda_2 \neq 0,\ (x - y) = 0$

$$2(x-1) + \lambda_2 = 0,\ 2(y-3) - \lambda_2 = 0$$

$$\Rightarrow x = 2, y = 2, \lambda_2 = -2$$

Since, $x + y \leq 2$, and $\lambda_2 < 0$; it violates This is not a feasible solution.

$$\frac{\delta L}{\delta x} = 2(x-1) + \lambda_1 + \lambda_2 = 0$$

$$\frac{\delta L}{\delta y} = 2(y-3) + \lambda_1 - \lambda_2 = 0$$

$$\lambda_1(x + y - 2) = 0$$

$$\lambda_2(x - y) = 0$$

$$\lambda_1, \lambda_2 \geq 0$$

$$(x + y) \leq 2, y \geq x$$

Lagrange Multiplier Method: Case 2

Example: Inequality Constraint Optimization Problem Solving

Solution:

3) To solve KKT constraints, we have to check the following tests:

Case 3: $\lambda_1 \neq 0, \lambda_2 = 0, (x + y) = 2$

$2(x-1) + \lambda_1 = 0, 2(y-3) + \lambda_1 = 0$

$\Rightarrow x = 0, y = 2, \lambda_1 = 2;$ This is a feasible solution.

Case 4:

$\lambda_1 \neq 0, \lambda_2 \neq 0, (x + y) = 2$

$(x - y) = 0; 2(x-1) + \lambda_1 + \lambda_2 = 0; 2(y-3) + \lambda_1 - \lambda_2 = 0$

$\Rightarrow x = 1, y = 1, and\ \lambda_1 = 2, \lambda_2 = -2;$ This is not a feasible solution.

$$\frac{\delta L}{\delta x} = 2(x-1) + \lambda_1 + \lambda_2 = 0$$
$$\frac{\delta L}{\delta y} = 2(y-3) + \lambda_1 - \lambda_2 = 0$$
$$\lambda_1(x + y - 2) = 0$$
$$\lambda_2(x - y) = 0$$
$$\lambda_1, \lambda_2 \geq 0$$
$$(x + y) \leq 2, y \geq x$$

Building Linear SVM

LMM to Solve Linear SVM

$$minimize\ \bar{\mu}(W,b) = \frac{\|W\|}{2}$$
$$subject\ to,$$
$$y_i(W.x_i + b) \geq 1,\ \ i = 1, 2, \dots, n$$

SVM: Inequality Constraint Optimization Problem

- The optimization problem for the linear SVM is inequality constraint optimization problem.
- The Lagrangian multiplier for this optimization problem can be written as

$$L = \frac{\|W\|^2}{2} - \sum_{i=1}^{n} \lambda_i \left(y_i(W.x_i + b) - 1\right)$$

where the parameters λ_i's are the Lagrangian multipliers, and

$W = [w_1, w_2 \dots \dots \dots w_m]$ and b are the model parameters.

Note:

We use $\frac{\|W\|^2}{2}$ instead of $\frac{\|W\|}{2}$ for the sake of simplification in calculations and it does not alter the problem's goal adversely.

LMM to Solve Linear SVM

$$L = \frac{\|W\|^2}{2} - \sum_{i=1}^{n} \lambda_i \left(y_i (W . x_i + b) - 1 \right)$$

The KKT constraints are:

$$\frac{\delta L}{\delta W} = 0 \Longrightarrow W = \sum_{i=1}^{n} \lambda_i . y_i . x_i$$

$$\frac{\delta L}{\delta b} = 0 \Longrightarrow \sum_{i=1}^{n} \lambda_i . y_i = 0,$$

$$\lambda_i \geq 0, i = 1, 2, \ldots \ldots n$$

$$\lambda_i [y_i (W . x_i + b) - 1] = 0, \qquad i = 1, 2, \ldots \ldots n$$

$$y_i \, (W . x_i + b) \geq 1, \qquad i = 1, 2, \ldots \ldots n$$

Solving KKT constraints are computationally intensive and can be solved using a typical linear/quadratic programming technique (or any other numerical technique).

LMM to Solve Linear SVM

- We first solve the above set of equations to find all the feasible solutions.
- Then, we can determine optimum value of $\mu(W, b)$.

Note:

- Lagrangian multiplier λ_i must be zero unless the training instance x_i satisfies the equation $y_i(W.xi + b) = 1$.
- Thus, the training tuples with $\lambda_i > 0$ lie on the hyperplane margins and hence are support vectors.
- The training instances that do not lie on the hyperplane margin have $\lambda_i = 0$.

Classifying an Unseen Data

- For a given training data, using SVM principle, we obtain MMH in the form of W, b and λ_i's. This is the machine (i.e., the SVM).
- Now, let us see how this MMH can be used to classify a test tuple, say X.
- This can be done as follows.

$$\delta(X) = W.X + b = \sum_{i=1}^{n}(\lambda_i.y_i.x_i.X + b)$$

Note that

$$W = \sum_{i=1}^{n}\lambda_i.y_i.x_i$$

- This is famously known as **Representor Theorem** which states that the solution W always be represented as a linear combination of training data.

Thus, $\delta(X) = W.X + b = \sum_{i=1}^{n}\lambda_i.y_i.x_i.X + b$

Classifying a Unseen Data

- The above involves a dot product of $x_i . X$, where x_i is a support vector (this is so because $(\lambda_i) = 0$ for all training tuples except the support vectors), we can check the sign of $\delta(X)$.
- If it is positive, then X falls on or above the MMH and so the SVM predicts that X belongs to class label +.
- On the other hand, if the sign is negative, then X falls on or below MMH and the class prediction is -.

Note:

- Once the SVM is trained with training data, the complexity of the classifier is characterized by the number of support vectors.
- Dimensionality of data is not an issue in SVM unlike in other classifier.

Illustration : Linear SVM

- Consider the case of a binary classification starting with a training data of 8 tuples as shown in **Table**.
- Using LMM, we can solve the KKT constraints to obtain the Lagrange multipliers λ_i for each training tuple, which is shown in **Table**.
- Note that only the first two tuples are support vectors in this case.

Table 1: Training Data

A_1	A_2	y	λ_i
0.38	0.47	+	65.52
0.49	0.61	-	65.52
0.92	0.41	-	0
0.74	0.89	-	0
0.18	0.58	+	0
0.41	0.35	+	0
0.93	0.81	-	0
0.21	0.10	+	0

Illustration : Linear SVM

- Let $W = (w_1, w_2)$ and b denote the parameter to be determined now.
- We can solve for w_1 and w_2 as follows:

$$w_1 = \sum_i \lambda_i . y_i . x_{i_1}$$
$$= 65.52 \times 1 \times 0.38 + 65.52 \times -1 \times 0.49$$
$$= \mathbf{-6.64}$$

$$w_2 = \sum_i \lambda_i . y_i . x_{i_2}$$
$$= 65.52 \times 1 \times 0.47 + 65.52 \times -1 \times 0.61$$
$$= \mathbf{-9.32}$$

Table 1: Training Data

A_1	A_2	y	λ_i
0.38	0.47	+	65.52
0.49	0.61	-	65.52
0.92	0.41	-	0
0.74	0.89	-	0
0.18	0.58	+	0
0.41	0.35	+	0
0.93	0.81	-	0
0.21	0.10	+	0

Illustration : Linear SVM

W = [-6.64 -9.32]
x_1 = [0.38 0.47]
x_2 = [0.49 0.61]

- The parameter b can be calculated for each support vector as follows

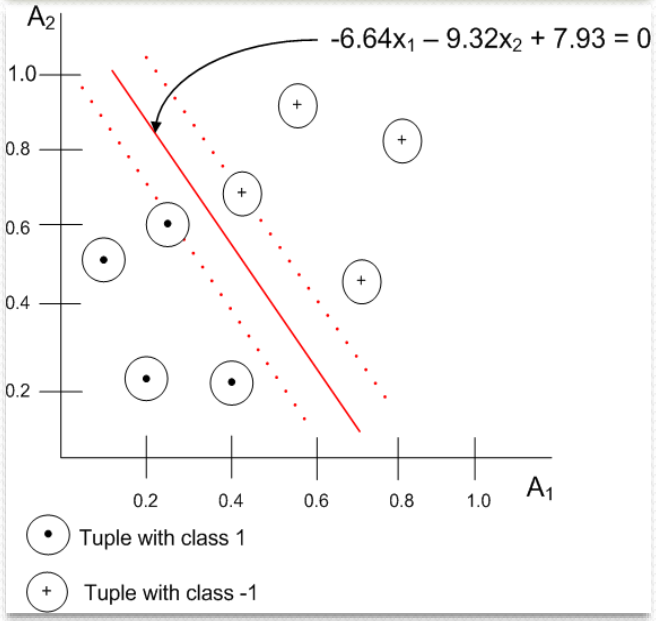
$$b_1 = 1 - W.x_1 \quad \text{// for support vector } x_1$$
$$= 1 - (-6.64) \times 0.38 - (-9.32) \times 0.47) \text{ //using dot product}$$
$$= \mathbf{7.93}$$

$$b_2 = 1 - W.x_2 \quad \text{// for support vector } x_2$$
$$= 1 - (-6.64) \times 0.49 - (-9.32) \times 0.61 \text{ //using dot product}$$
$$= \mathbf{7.93}$$

- Averaging these values of b_1 and b_2, we get $\boldsymbol{b = 7.93}$.

Illustration : Linear SVM

Fig. 13: Linear SVM example.



- Thus, the MMH is

$$-6.64x_1 - 9.32x_2 + 7.93 = 0$$

- Suppose, a test data is $X = (0.5, 0.5)$. Therefore,

$$\begin{aligned}\delta(X) &= W.X + b \\ &= -6.64 \times 0.5 - 9.32 \times 0.5 + 7.93 \\ &= -0.05 \\ &= -ve\end{aligned}$$

- This implies that the test data falls on or below the MMH and SVM classifies that X belongs to class label -.

Multi-class Classification

Classification of Multiple-class Data

- In the discussion of linear SVM, we have limited our discussion to binary classification (i.e., classification with two classes only).
- Note that the discussed linear SVM can handle any *n*-dimension, $n \geq 2$.
- Now, we are to discuss a more generalized linear SVM to classify *n*-dimensional data belong to two or more classes.
- **There are two possibilities:**
  - all classes are pairwise linearly separable, or
  - all classes are overlapping (not linearly separable.)

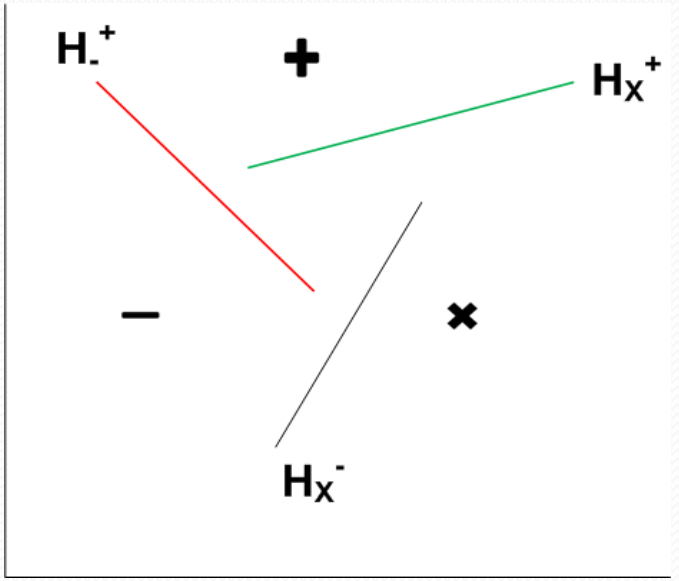
Classification of Multiple-class Data

- If the classes are pair-wise linearly separable, then we can extend the principle of linear SVM to each pair.
- **There are two strategies:**
  - One versus one (OVO) strategy
  - One versus all (OVA) strategy

Multi-Class Classification: OVO Strategy

- In OVO strategy, we are to find MMHs for each pair of classes.
- Thus, if there are n classes, then ${}^{n}c_{2}$ classifiers possible. For an example, see **Fig. 14** for 3 classes (namely $+$, $-$, and $\times$).
- Let, H_y^x denotes MMH between class labels x and y .
- As another examples, you can think, say for 4 classes (namely $+$, $-$, $\times$, and $\div$) and more.

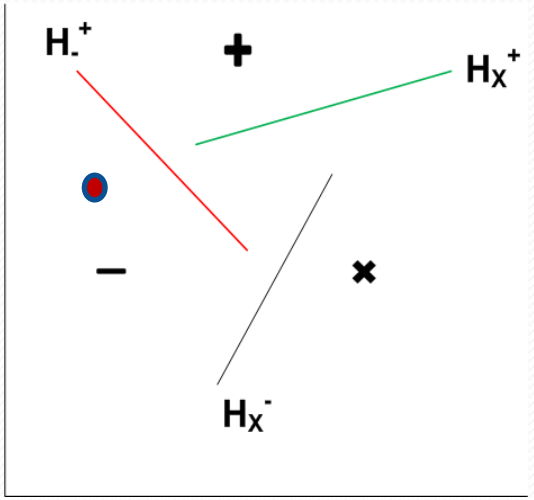
Fig. 14: 3-pairwise linearly separable classes.



Multi-Class Classification: OVO Strategy

- With OVO strategy, we test each of the classifier in turn and obtain $\delta_i^j(X)$, that is, the sign of the MMH between j^{th} and i^{th} classes for a test data X.

- If there is a class i, for which $\delta_i^j(X)$, for all j (and $j \neq i$), gives same sign, then unambiguously, we can say that X is in class i.



Multi-Class Classification: OVA Strategy

- OVO strategy is not useful for data with a large number of classes, as the computational complexity increases exponentially with the number of classes.
- As an alternative to OVO strategy, OVA(One Versus All) strategy has been proposed.
- In this approach, we choose any class say C_i and consider that all tuples of other classes belong to a single class.

Multi-Class Classification: OVA Strategy

- This is, therefore, transformed into a binary classification problem and using the linear SVM discussed above, we can find the hyperplane.
- Let the hyperplane between C_i and remaining classes be MMH_i.
- The process is repeated for each $C_i \in [C_1, C_2, \ldots . . C_k]$ and getting MMH_is
- In other words, with OVA strategies we get k classifiers.

Multi-Class Classification: OVA Strategy

- The unseen data X is then tested with each classifier so obtained.
- Let $\delta_j(X)$ be the test result with MMH_j, which has the maximum magnitude of the test values.
- That is, $\delta_j = max_{\forall i}\{\delta_i(X)\}$
- Thus, X is classified into class C_j.

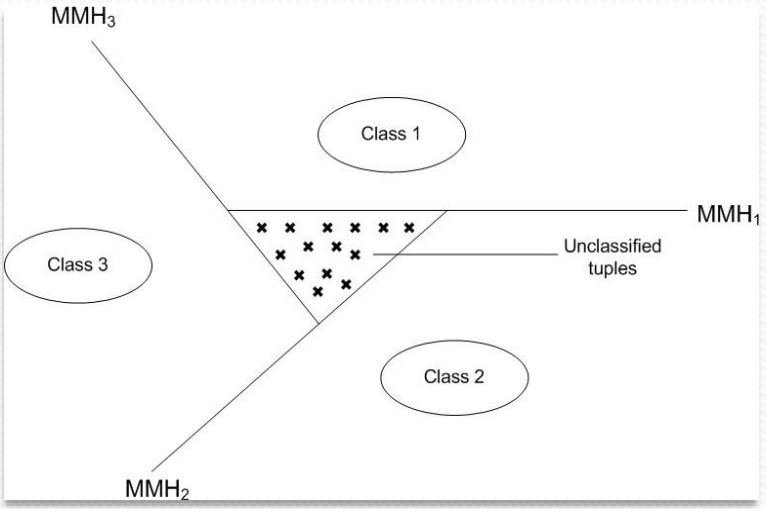
Note:

- The linear SVM that is used to classify multi-class data fails, if all classes are not linearly separable.
- If one class is linearly separable to remaining other classes and test data belongs to that particular class, then only it classifies accurately.

Multi-Class Classification: OVA Strategy

- Further, it is possible to have some tuples which cannot be classified to any of the linear SVMs.
- There are some tuples which cannot be classified unambiguously by neither of the hyperplanes.
- All these tuples may be due to noise, errors or data are not linearly separable.

Fig. 15: Unclassifiable region in OVA strategy.



Non-linear Support Vector Machine

Non-linear SVM

- **SVM classification for linearly non-separable data**
  - Figure shows a 2D views of data when they are linearly separable and not separable.
  - In general, if data are linearly separable, then there is a hyperplane otherwise no hyperplane.

Fig. 16: Two types of training data.

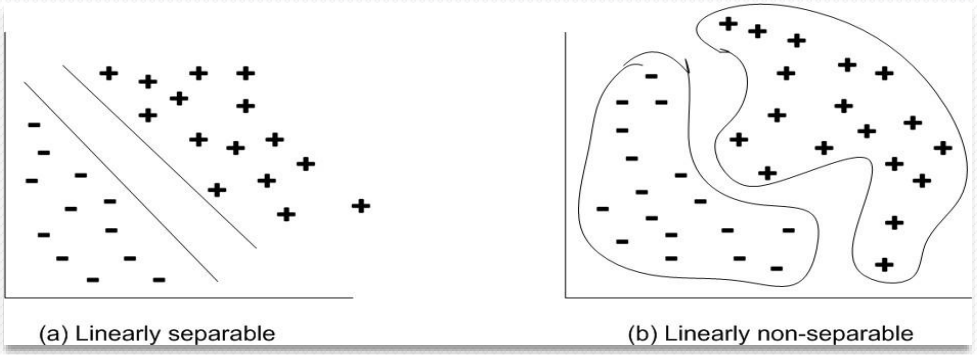


Fig. 16: Linear and Non-linear SVMs

Linear SVM for Non-separable Data

- Such a linearly not separable data can be classified using two approaches.
  - Linear SVM with soft margin
  - Non-linear SVM
- In the following, we discuss the extension of linear SVM to classify linearly non-separable data.
- We discuss non-linear SVM in detail later.

Linear SVM for Linearly Non-separable Data

- If the number of training data instances violating linear separability is less, then we can use linear SVM classifier to classify them.
- This can be done by reducing the margin
- The rational behind this approach can be better understood from **Fig. 17**.

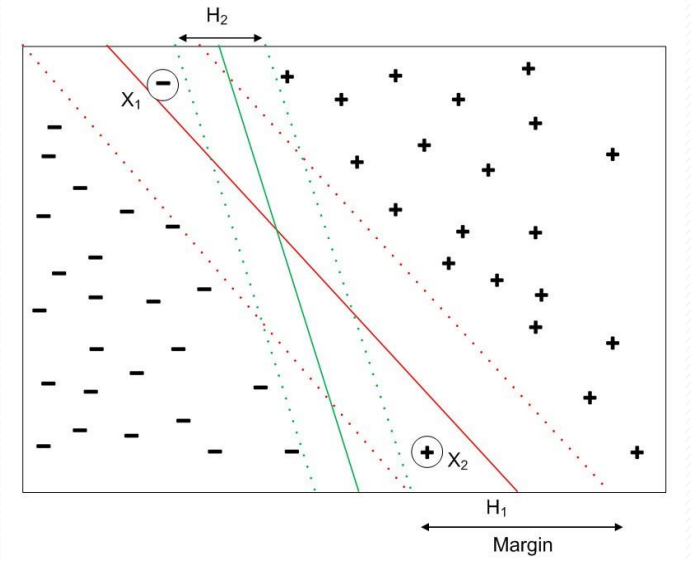


Fig. 17: Problem with linear SVM for non-linear data

Non-Linear SVM

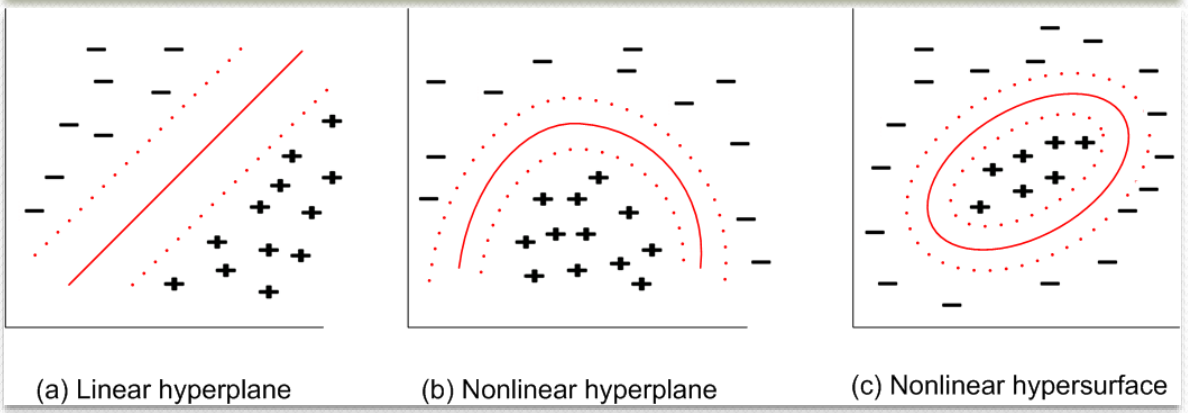
- Linear SVM undoubtedly better to classify data if it is trained by linearly separable data.
- Linear SVM also can be used for non-linearly separable data, provided that number of such instances is less.
- However, in real life applications, number of data overlapping is so high that the approach cannot cope to yield accurate classifier.
- As an alternative to this there is a need to compute a decision boundary, which is not linear (i.e., not a hyperplane rather hypersurface).

Non-Linear SVM

To understand these ideas, see below figure.

- Note that a linear hyperplane is expressed as a linear equation in terms of n-dimensional component, whereas a non-linear hypersurface is a non-linear expression.

Fig. 18: 2D view of few class separability.



Non-Linear SVM

- A hyperplane is expressed as

$$w_1 x_1 + w_2 x_2 + w_3 x_3 + b = 0$$

- Whereas a non-linear hypersurface is expressed as.

$$w_1 x_1^2 + w_2 x_2^2 + w_3 x_1 x_2 + w_4 x_3^2 + w_5 x_1 x_3 + b = 0$$

- The task therefore takes a turn to find a nonlinear decision boundaries, that is, a nonlinear hypersurface in input space comprising with linearly non separable data.

- This task indeed neither hard nor so complex, and fortunately can be accomplished extending the formulation of linear SVM, we have already learned.

Non-Linear SVM

This can be achieved in two major steps.

- Transform the original (non-linear) input data into a higher dimensional space (as a linear representation of data).
- Note that this is feasible because SVM's performance is decided by number of support vectors (*i.e.,* ≈ *training data*) not by the dimension of data.
- Search for the linear decision boundaries to separate the transformed higher dimensional data.
  - The above can be done in the same line as we have done for linear SVM.

Concept of Non-Linear Mapping

- In nutshell, to have a nonlinear SVM, the trick is to transform non-linear data into higher dimensional linear data.
- This transformation is popularly called non-linear mapping or attribute transformation or ϕ-transformation. The rest is same as the linear SVM.
- In order to understand the concept of non-linear transformation of original input data into a higher dimensional space, let us consider a non-linear second order polynomial in a 3-D input space.

$$X(x_1, x_2, x_3) = w_1x_1 + w_1x_1 + w_1x_1 + w_4x_1^2 + w_5x_1x_2 + w_6x_1x_3 + b$$

Concept of Non-Linear Mapping

$$X(x_1, x_2, x_3) = w_1x_1 + w_1x_1 + w_1x_1 + w_4x_1^2 + w_5x_1x_2 + w_6x_1x_3 + b$$

- The 3-D input vector $X(x_1, x_2, x_2)$ can be mapped into a 6-D space $Z(z_1, z_2, z_3, z_4, z_5, z_6)$ using the following mappings:

$$z_1 = \phi_1(x_1) = x_1$$

$$z_2 = \phi_2(x_2) = x_2$$

$$z_3 = \phi_3(x_3) = x_3$$

$$z_4 = \phi_4(x_1) = x_1^2$$

$$z_5 = \phi_5(x_1, x_2) = x_1.x_2$$

$$z_6 = \phi_6(x_1, x_3) = x_1.x_3$$

Concept of Non-Linear Mapping

$$X(x_1, x_2, x_3) = w_1x_1 + w_1x_1 + w_1x_1 + w_4x_1^2 + w_5x_1x_2 + w_6x_1x_3 + b$$

$$z_1 = \phi_1(x) = x_1$$
$$z_2 = \phi_2(x) = x_2$$
$$z_3 = \phi_3(x) = x_3$$
$$z_4 = \phi_4(x) = x_1^2$$
$$z_5 = \phi_5(x) = x_1.x_2$$
$$z_6 = \phi_6(x) = x_1.x_3$$

- The transformed form of linear data in 6-D space will look like.

$$Z: \; w_1z_1 + w_2z_2 + w_3z_3 + w_4z_4 + w_5z_5 + w_6z_6 + b$$

- Thus, if Z space has input data for its attributes x_1, x_2, x_3 (and hence Z's values), then we can classify them using linear decision boundaries.

Concept of Non-Linear Mapping

- **Example: Non-linear mapping to linear SVM**
  - Figure 19 shows an example of 2D data set consisting of class label + (as +) and class label – (as –).

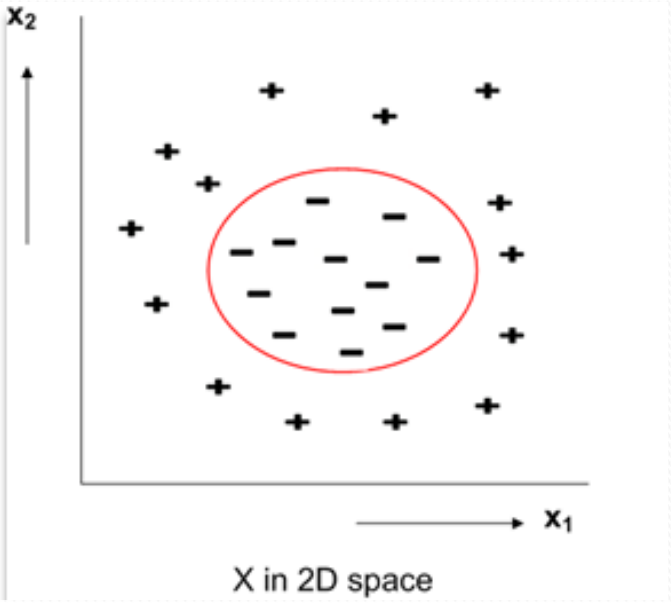
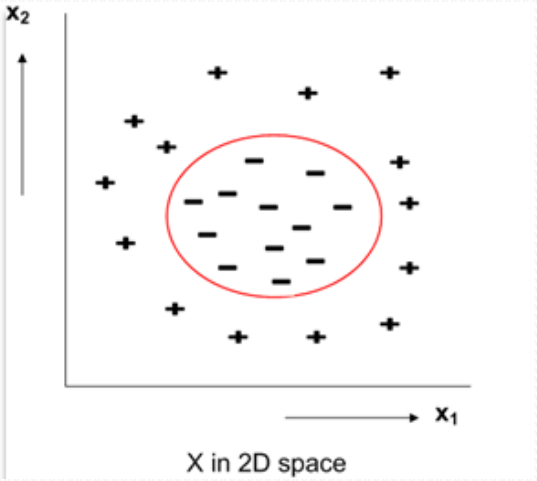


Fig. 19: Non-linear data separation

Concept of Non-Linear Mapping



- **Example: Non-linear mapping to linear SVM**
  - We see that all instances of class − can be separated from instances of class + by a circle. Without anonymity, say the following equation of the decision boundary can be thought of:

$$X(x_1, x_2) = + \quad if \quad \sqrt{(x_1 - 0.5)^2 + (x_2 - 0.5)^2} > 2$$

$$X(x_1, x_2) = - \quad otherwise$$

Concept of Non-Linear Mapping

- **Example: Non-linear mapping to linear SVM**
  - The decision boundary can be written as:

$$X = \sqrt{(x_1 - 0.5)^2 + (x_2 - 0.5)^2} = 2$$

$$or \quad x_1^2 - x_1 + x_2^2 - x_2 = 3.5$$

  - A non-linear transformation in 2-D space is proposed as follows:

$$Z(z_1, z_2): \ \phi_1(x_1) = x_1^2 - x_1, \qquad \phi_2(x_2) = x_2^2 - x_2$$

Concept of Non-Linear Mapping

- **Example: Non-linear mapping to linear SVM**
  - The Z space when plotted will take view as shown in figure given below, where data are separable with linear boundary, namely Z: $z_1 + z_2 = 3.5$

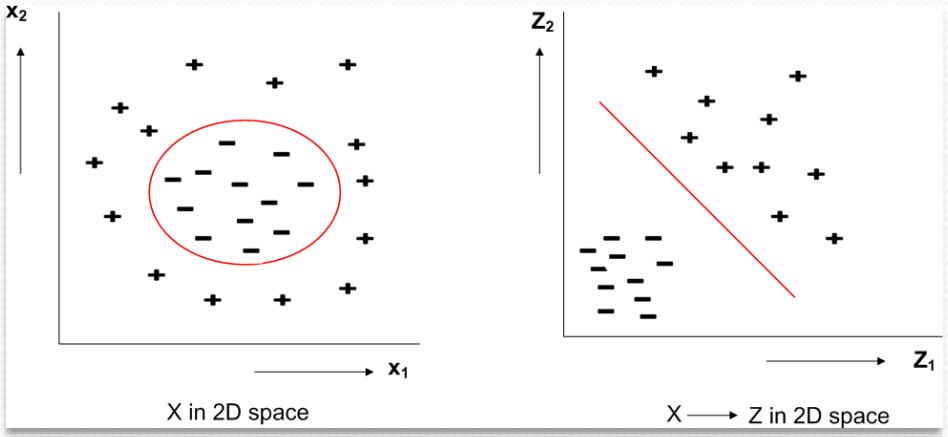


Fig. 20: Non-linear mapping to Linear SVM.

Non-Linear to Linear Transformation: Issues

- The non linear mapping and hence a linear decision boundary concept looks pretty simple. But there are many potential problems to do so.
  - Mapping: How to choose the non linear mapping to a higher dimensional space?
    - In fact, the ϕ-transformation works fine for small examples.
    - But, it fails for realistically sized problems.
    - For N-dimensional input instances there exist $N_H = \frac{(N+d-1)!}{d!(N-1)!}$ different monomials comprising a feature space of dimensionality N_H. Here, d is the maximum degree of monomial.
  - Dimensionality problem: It may suffer from the curse of dimensionality problem often associated with a high dimensional data.
    - More specifically, in the calculation of $W.X$ or $X_i.X$ (in $\delta(X)$), we need n multiplications and n additions (in their dot products) for each of the n −dimensional input instances and support vectors.
    - As the number of input instances as well as support vectors are enormously large, it will be computationally intensive.
  - Computational cost: Solving the inequality constrained optimization problem in the high dimensional feature space is a very computationally intensive task.

Non-Linear to Linear Transformation: Solution

- Fortunately, mathematicians have cleverly proposed an elegant solution to the above problems.
- Their solution consist of the following:
  - Dual formulation of optimization problem
  - Kernel trick
- In the next few lectures, we shall learn about the above-mentioned two concepts.

Non-Linear to Linear Transformation

Non-linear to linear transformation: Issues

As discussed in previous lecture, there are some issues with the non-linear to linear transformation:

1. Mapping
2. Dimensionality Problem
3. Computational Cost

Non-Linear to Linear Transformation

Non-linear to linear transformation: Solutions

- Fortunately, mathematicians have cleverly proposed an elegant solution to the above problems.
- Their solution consist of the following
  - Dual formulation of optimization problem
  - Kernel trick
- In the next few slides, we shall learn about the above-mentioned two concepts

Dual Formulation of Optimization Problem

Primal Form of Optimization Problem

The primal form

- We have already learned the Lagrangian formulation to find the maximum margin hyperplane for a linear SVM classifier.
- Such a formulation is called primal form of the constraint optimization problem.
- Primal form of the optimization problem is reproduced further

$$Minimize \ \frac{\|W\|^2}{2}$$

$$Subject\ to\ y_i(W.x_i + b) \geq 1, i = 1, 2, 3, \ldots, n$$

Primal Form of Optimization

The primal form of the optimization problem

- The primal form of the above mentioned inequality constraint optimization problem (according to Lagrange multiplier method) is given by

$$L_p = \frac{\|W\|^2}{2} - \sum_{i=1}^{n} \lambda_i (y_i (W . x_i + b) - 1)$$

where, λ_i is called the Lagrangian multipliers

- Here, L_p is called the primal form of the Lagrange optimization problem

Dual Formulation of Optimization Problem

$$L_p = \frac{\|W\|^2}{2} - \sum_{i=1}^{n} \lambda_i (y_i (W . x_i + b) - 1)$$

The Dual form from the primal form

- The dual form of the same problem can be derived (from L_p) as follows:
- Note that to minimize the optimization problem, we must take the derivative of L_p with respect to W, b and set them to zero.

$$\frac{\delta L_p}{\delta W} = 0 \Rightarrow W = \sum_{i=1}^{n} \lambda_i y_i x_i$$

$$\frac{\delta L_p}{\delta b} = 0 \Rightarrow \sum_{i=1}^{n} \lambda_i y_i = 0$$

Dual Formulation of Optimization Problem

$$L_p = \frac{\|W\|^2}{2} - \sum_{i=1}^{n} \lambda_i (y_i (W.x_i + b) - 1)$$

$$\frac{\delta L_p}{\delta W} = 0 \Rightarrow W = \sum_{i=1}^{n} \lambda_i y_i x_i \; ; \; \frac{\delta L_p}{\delta b} = 0 \Rightarrow \sum_{i=1}^{n} \lambda_i y_i = 0$$

The Dual form from the primal form

- From the above two equations, we get the Lagrangian L as

$$L = \sum_{i=1}^{n} \lambda_i + \frac{1}{2} \sum_{i,j} \lambda_i y_i \lambda_j y_j x_i . x_j - \sum_{i=1}^{n} \lambda_i y_i \sum_{j=1}^{n} (\lambda_j y_j x_j) x_i = \sum_{i=1}^{n} \lambda_i - \frac{1}{2} \sum_{i,j} \lambda_i y_i \lambda_j y_j . x_i . x_j$$

This form is called the **Dual form of Lagrangian** and distinguishably written as:

$$\boldsymbol{L_D = \sum_{i=1}^{n} \lambda_i - \frac{1}{2} \sum_{i,j} \lambda_i . \lambda_j \, y_i . y_j . x_i . x_j}$$

Dual Form versus Primal Form

Differences between the primal (L_P) and dual (L_D) forms

There are key differences between primal (L_P) and dual (L_D) forms of Lagrangian optimization problem as follows:

1. L_P involves a large number of parameters namely W, b and $\lambda_i's$. On the other hand, L_D involves only $\lambda_i's$, that is, Lagrangian multipliers.

2. L_P is the minimization problem as the quadratic term is positive. However, the quadratic term in L_D is negative sign. Hence, it turned out to be a maximization problem.

3. L_P involves the calculation of $W.x$, whereas L_D involves the calculation of $x_i.x_j$. This, in fact, advantageous, and we will realize it when we learn Kernel-based calculation.

4. The SVM classifier with primal form is $\delta_p(x) = W.x + b$, whereas the dual version of the classifier is

$$\delta_D(x) = \sum_{i=1}^{m} \lambda_i y_i (x_i.x) + b$$

where, x_i being the i^{th} support vector and assume that there are m support vectors.

Kernel Trick

Dealing with Non-linear Data

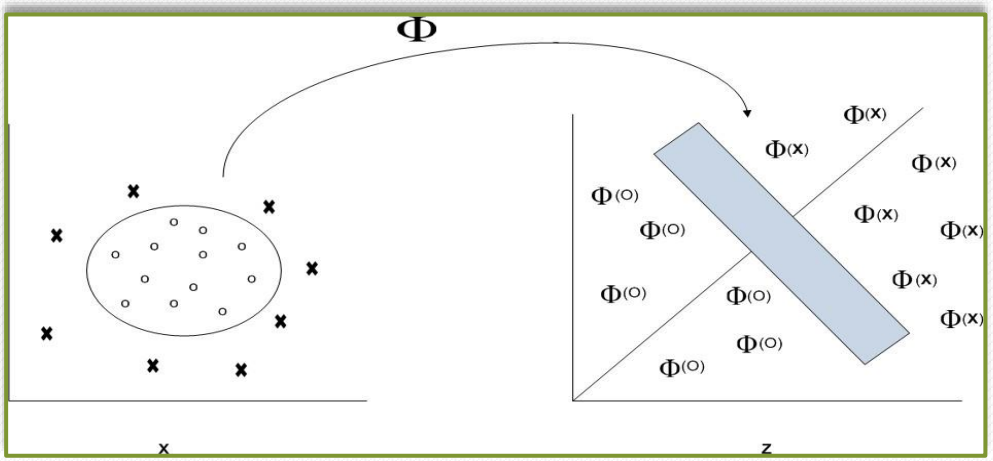


Fig. 21: SVM classifier in transformed feature space

- We have already learned an idea that training data which are not linearly separable, can be transformed into a higher dimensional feature space such that in higher dimensional transformed space a hyperplane can be decided to separate the transformed data and hence the original data.
- Clearly the data on the left in Fig. 21 is not linearly separable. Yet if we map it to a 3D space using φ-transformation, then with mapped data it is possible to have a linear decision boundary (i.e., a hyperplane in 3D space).

Dealing with Non-linear Data

Example: Working with non-linear data

- Suppose, there is a set of data in $\mathbb{R}^2$ (i.e., in 2D space).
- Let the hyperplane in $\mathbb{R}^2$ takes the form

$$w_1 x_1^2 + w_2 \sqrt{2} x_1 x_2 + w_3 x_2^2 = 0$$

which is the equation of an ellipse in 2D.

- Suppose, φ is the mapping for $X \in \mathbb{R}^2$ to $Z(z_1, z_2, z_3) \in \mathbb{R}^3$ in 3D space.

$$\mathbb{R}^2 \Rightarrow X(x_1, x_2) \qquad \mathbb{R}^3 \Rightarrow Z(z_1, z_2, z_3)$$

$$\varphi(x) \Rightarrow Z;$$

Consider, $z_1 = x_1^2, \quad z_2 = \sqrt{2} x_1 x_2, \quad z_3 = x_2^2$

- After the φ-transformation as mentioned above, we have the decision boundary of the form

$$w_1 z_1 + w_2 z_2 + w_3 z_3 = 0$$

- This is clearly a linear form in 3D space. In other words, the hyperplan in $\mathbb{R}^2$ has mapped onto $W.z + b' = 0$ in $\mathbb{R}^3$, which is in linear form

Dealing with Non-linear Data

Conclusion: Linear SVM classifier is possible for classifying non-linear data

- This means that data which are not linearly separable in 2D are separable in 3D. This implies that the non-linear data can be classified by a linear SVM classifier.
- The generalization of this formulation is provided next, which is key to kernel trick.

Kernel Trick

Kernel trick: A generalized formulation

Subject to, $\lambda_i \geq 0, \sum_i \lambda_i . y_i = 0$

- Learning:

$$Maximize \sum_{i=1}^{n} \lambda_i - \frac{1}{2} \sum_{i,j} \lambda_i . \lambda_j . y_i . y_j . x_i . x_j$$

$$Maximize \sum_{i=1}^{n} \lambda_i - \frac{1}{2} \sum_{i,j} \lambda_i . \lambda_j . y_i . y_j . \varphi(x_i) . \varphi(x_j)$$

- Classifier:

$$\delta(x) = \sum_{i=1}^{n} \lambda_i y_i x_i . x + b$$

$$\delta(z) = \sum_{i=1}^{n} \lambda_i y_i . \varphi(x_i) . \varphi(x) + b$$

Kernel Trick

How to choose the mapping function φ?

- Now, question here is how to choose φ, the mapping function $X \Rightarrow Z$, so that linear SVM can be directly applied to?
- A breakthrough solution to this problem comes in the form of a method as the kernel trick.
- The kernel trick is as follows.
- We know that (.) dot product is often regarded as a measure of similarity between two input vectors.
  - For example, if X and Y are two vectors, then $X.Y = |X||Y|\cos\theta$
  - Here, similarity between X and Y is measured as cosine similarity.
  - If $\theta = 0$ (i.e., $\cos\theta = 1$), then they are most similar, otherwise orthogonal, means dissimilar.

Kernel Trick

How to choose the mapping function φ?

- Analogously, if X_i and X_j are two tuples, then $X_i . X_j$ is regarded as a measure of similarity between X_i and X_j
- Again, $\varphi(X_i)$ and $\varphi(X_j)$ are the transformed features of X_i and X_j, respectively in the transformed space; thus, $\varphi(X_i).\varphi(X_j)$ is also should be regarded as the similarity measure between $\varphi(X_i)$ and $\varphi(X_j)$ in the transformed space.
- This is an important revelation and is the basic idea behind the kernel trick.
- Now, naturally question arises, if both measures the similarity, then what is the correlation between them (i.e., $X_i . X_j$ and $\varphi(X_i).\varphi(X_j)$).
- Let us try to find the answer to this question through an example.

Kernel Trick

Example: Correlation between X_i. X_j and $\varphi(X_i).\varphi(X_j)$

- Without the loss of generality, let us consider a situation stated below:

$$\varphi: \mathbb{R}^2 \Rightarrow \mathbb{R}^3 = x_1^2 \Rightarrow z_1, x_2^2 \Rightarrow z_2, \sqrt{2}x_1x_2 \Rightarrow z_3$$

- Suppose, $X_i = [x_{i1}, x_{i2}]$, and $X_j = [x_{j1}, x_{j2}]$ are any two vectors in $\mathbb{R}^2$
- Similarly,

$$\varphi(X_i) = [{x_{i1}}^2, \sqrt{2}x_{i1}x_{i2}, {x_{i2}}^2] \text{ and}$$
$$\varphi(X_j) = [{x_{j1}}^2, \sqrt{2}x_{j1}x_{j2}, {x_{j2}}^2]$$

are two transformed version of X_i and X_j in $\mathbb{R}^3$

Kernel Trick

Example: Correlation between $X_i . X_j$ and $\varphi(X_i).\varphi(X_j)$

Now,

$$\varphi(X_i).\varphi(X_j) = \begin{bmatrix} {x_{i1}}^2 & \sqrt{2}x_{i1}x_{i2} & {x_{i2}}^2 \end{bmatrix} \begin{bmatrix} {x_{j1}}^2 \\ \sqrt{2}x_{j1}x_{j2} \\ {x_{j2}}^2 \end{bmatrix}$$

$$= {x_{i1}}^2 . {x_{j1}}^2 + 2x_{i1}x_{i2}x_{j1}x_{j2} + {x_{i2}}^2 {x_{j2}}^2$$

$$= \left(x_{i1}.x_{j1} + x_{i2}.x_{j2}\right)^2$$

$$= \left\{ \begin{bmatrix} x_{i1} & x_{i2} \end{bmatrix} \begin{bmatrix} x_{j1} \\ x_{j2} \end{bmatrix} \right\}^2$$

$$= \left(X_i . X_j\right)^2$$

Kernel Trick

Example: Correlation between $X_i . X_j$ and $\varphi(X_i).\varphi(X_j)$

- With reference to the above example, we can conclude that $\varphi(X_i).\varphi(X_j)$ are correlated to $X_i . X_j$
- In fact, the same can be proved, in general, for any feature vectors and their transformed feature vectors. A formal proof is beyond the scope of this presentation.
- More specifically, there is a correlation between dot products of original data and dot products of transformed data.
- Based on the above discussion, we can write the following implications:

$$X_i . X_j \Rightarrow \varphi(X_i).\varphi(X_j) \Rightarrow K(X_i, X_j)$$

Here, $K(X_i, X_j)$ denotes a function more popularly called as **Kernel function**

Significance of Kernel Trick

The Implication of Kernel Trick

Significance of Kernel Trick

- This kernel function $K(X_i, X_j)$ physically implies the similarity in the transformed space (i.e., non-linear similarity measure) using the original attribute $X_i . X_j$
- In other words, K, the similarity function to compute a similarity of both whether data in original attribute space or in transformed attribute space.

Kernel Trick and SVM

Formulation of SVM with Kernel trick

Implicit transformation: The first and foremost significance is that we do not require any φ-transformation to the original input data at all! This is evident from the following re-writing of our SVM classification problem.

Subject to, $\lambda_i \geq 0, \sum_i \lambda_i . y_i = 0$

Learning:

$$Maximize \sum_{i=1}^{n} \lambda_i - \frac{1}{2} \sum_{i,j} \lambda_i . \lambda_j . y_i . y_j . K(X_i, X_j)$$

Classifier:

$$\delta(x) = \sum_{i=1}^{n} \lambda_i y_i K(X_i, X) + b$$

Kernel Trick and its Advantage

Application of Kernel trick

Computational efficiency:

- Kernel trick allows an easy and efficient computability.
- We know that in a SVM classifier, we need several and repeated round of computation of dot products both in learning phase as well as in classification phase.
- On other hand, using Kernel trick, we can do it once and with fewer dot products.
- This is explained next.

Kernel Trick and its Advantage

Design matrix

- We define a matrix called design matrix (X), which contains all data as follows:

$$X = \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ \vdots \\ X_n \end{bmatrix}_{n \times N}$$

- Note that X contains all input data in the original attribute space.

Kernel Trick and its Advantage

Gram matrix (with linear and non-linear data)

- Next, we define another matrix called Gram matrix (K), which contains all dot products as follows:

$$K = \begin{bmatrix} X_1^T.X_1 & X_1^T.X_2 & \cdots & \cdots & X_1^T.X_n \\ X_2^T.X_1 & X_2^T.X_2 & \cdots & \cdots & X_2^T.X_n \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ X_n^T.X_1 & X_n^T.X_2 & \cdots & \cdots & X_n^T.X_n \end{bmatrix}_{n \times n}$$

- Note that K contains all dot products among all training data and $X_i^T.X_j = X_j^T.X_i$. This means, we need to compute only half of the matrix.
- More significantly, all dot products are mere by means of matrix multiplication operation, and that is too one operation only
- The same when we use Kernel-based similarity estimation (non-linear data), the Gram matrix becomes:

$$K = \begin{bmatrix} K(X_1^T, X_1) & K(X_1^T, X_2) & \cdots & \cdots & K(X_1^T, X_n) \\ K(X_2^T, X_1) & K(X_2^T, X_2) & \cdots & \cdots & K(X_2^T, X_n) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ K(X_n^T, X_1) & K(X_n^T, X_2) & \cdots & \cdots & K(X_n^T, X_n) \end{bmatrix}_{n \times n}$$

SVM with Kernel Trick

Summary of Kernel trick

In nutshell, we have the following.

- Instead of mapping our data via φ and computing the dot products, we can accomplish everything in one operation.
- Classifier can be learnt and applied without explicitly computing $\varphi(X)$.
- Complexity of learning depends on n (typically it is $O(n^3)$), not on N, the dimensionality of data space.
- All that is required is the kernel $K(X_i, X_j)$.
- Next, we discuss the aspect of deciding kernel functions.

Kernel Functions

Kernel Functions

Formal definition of Kernel function

Before going to learn the popular kernel functions adopted in SVM classification, we give a precise and formal definition of kernel $K(X_i, X_j)$

A kernel function $K(X_i, X_j)$ is a real valued function defined on $\mathbb{R}$, such that there is another function $\varphi : X \rightarrow Z$ and $K(X_i, X_j) = \varphi(X_i).\varphi(X_j)$.

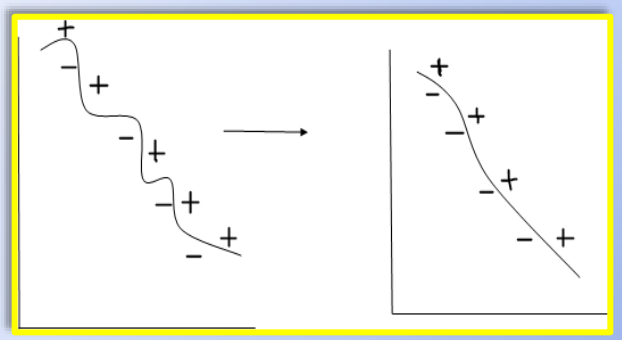
Symbolically, we write $\varphi : \mathbb{R}^m \rightarrow \mathbb{R}^n : K(X_i, X_j) : \varphi(X_i).\varphi(X_j)$ where $n > m$.

Well Known Kernel Functions

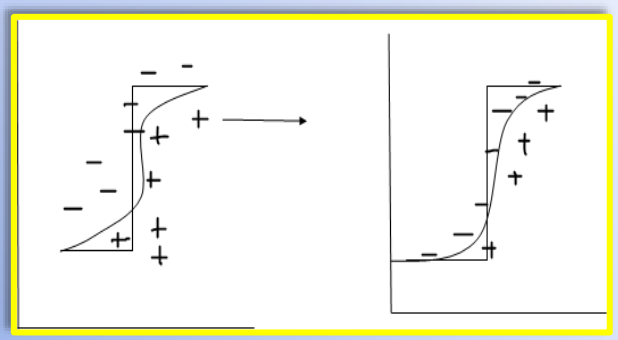
Some popular kernel functions

Kernel Name	Functional Form	Remark
Linear kernel	$K(X,Y) = X^T Y$	The simplest kernel used in linear SVM
Polynomial kernel of degree p	$K(X,Y) = (X^T Y + 1)^p$	It produces a large dot products. Power p is specified a priori by the user.
Gaussian (RBF) kernel	$K(X,Y) = e^{\frac{c\|X-Y\|^2}{2\sigma^2}}$	It is a non-linear kernel called Gaussian Radial Bias Function (RBF) kernel
Laplacian kernel	$K(X,Y) = e^{-\lambda\|X-Y\|}$	Follows Laplacian mapping
Sigmoid kernel	$K(X,Y) = \tanh(\beta_0 X^T Y + \beta_1)$	Followed when statistical test data is known
Mahalanobis kernel	$K(X,Y) = e^{-(X-Y)^T A(X-Y)}$	Followed when statistical test data is known

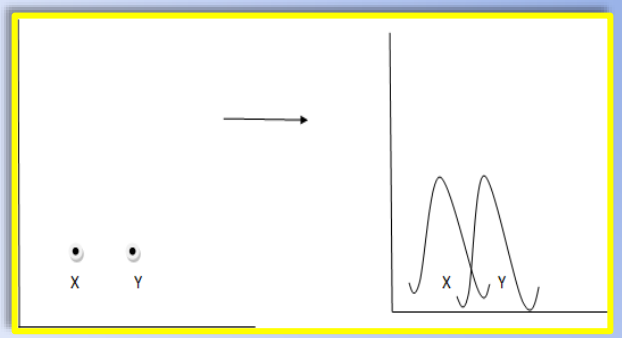
Kernel Functions: Example



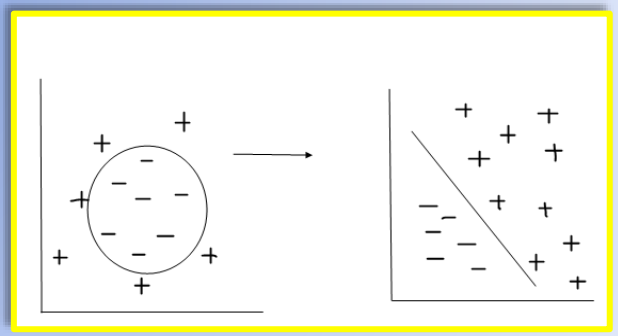
a) Polynomial Kernel



b) Sigmoid Kernel



c) Laplacian Kernel



d) Gaussian RBF Kernel

Fig. 22: Visual interpretation of a few kernel functions

Properties of Kernel Functions

Kernel Functions: Properties

Properties of kernel functions

- Other than the standard kernel functions, we can define of our own kernels as well as combining two or more kernels to another kernel.

- If K_1 and K_2 are two kernels then,
  - $K_1 + c$
  - aK_1
  - $aK_1 + bK_2$
  - $K_1 . K_2$

  are also valid Kernels.
- Here, $a, b, c \in \mathbb{R}^+$

Kernel Functions: Properties

Mercer's Theorem

- Another requirement for kernel function used in non-linear SVM is that there must exist a corresponding transformation such that the kernel function computed for a pair of vectors is equivalent to the dot product between the vectors in the transformed space.
- This requirement can be formally stated in the form of Mercer's theorem.

A kernel function K can be expressed as $K(X, Y) = \varphi(X).\varphi(Y)$, if and only if, for any function $g(x)$ such that $\int g(x)^2.dx$ is finite then

$$\int K(X, Y)g(x)g(y)\,dxdy \geq 0$$

- The kernels which satisfy the Mercer's theorem are called Mercer kernels.

Kernel functions: Properties

Additional properties of Kernel functions

- Mercer kernels should satisfy the following additional properties:

- **Symmetric:** $K(X,Y) = K(Y,X)$
- **Positive Definite:** $\alpha^T . K . \alpha \geq 0$ for all $\alpha \in \mathbb{R}^N$, where K is the $n \times n$ Gram Matrix.
  - It can be proved that all the kernels listed in Table 2 are satisfying the kernel properties, and Mercer's theorem and hence they are Mercer kernels.

Advantages of Support Vector Machine

Conclusions

Advantages

- 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. Other methods, namely rule based classifier, ANN classifier, etc. find only local optimum solutions.
- SVM is the best suitable to classify both linear as well as non-linear training data efficiently.
- SVM can be applied to categorical data also by introducing a suitable similarity measures.
- Computational complexity is influenced by the number of training data not the dimension of data.
  - In fact, learning is a bit computationally heavy and hence slow, but classification of test is extremely fast and accurate.

Any question?