Support Vector Machines

(Non-Linear Models)

CSED, TIET

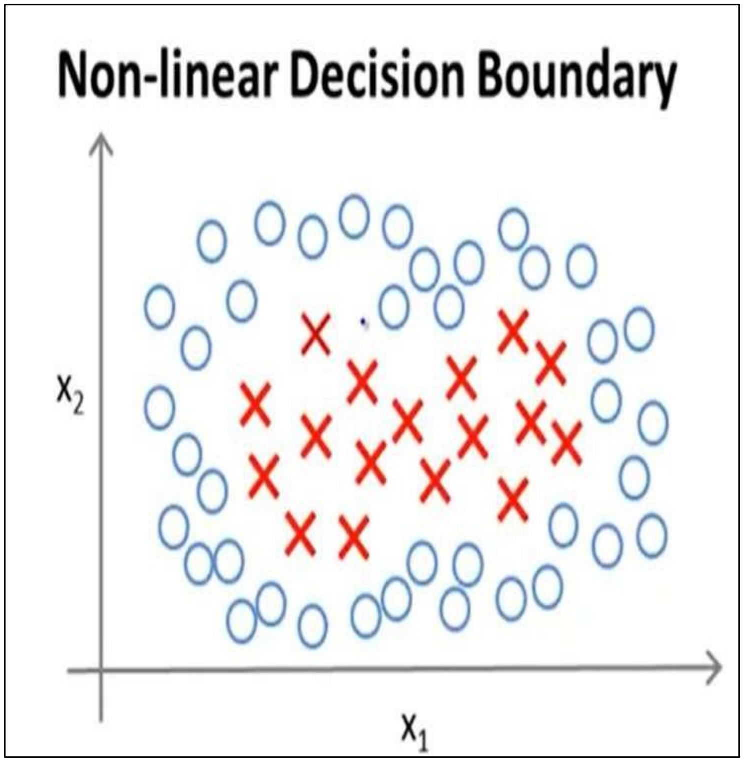
Introduction

- One way of dealing with non linear decision boundary is to fit a higher order polynomial hypothesis function (as shown below):

$$f(x) = \beta_0 + \beta_0 x_1 + \beta_0 x_2 + \beta_0 x_1 x_2 + \beta_0 x_1^2 + \beta_0 x_2^2 + \cdots \ldots$$

- But this method has few limitations:

1. We can not estimate the order of the polynomial function from the multi- dimensional training data.
2. The higher order polynomials are computationally quite expensive especially for images (where the features are pixel values) or text (where the features are frequency of words).



Kernel Functions for Non-Linear Boundaries

- In order to fit a non linear decision boundary, SVM uses Kernel-based approach or Kernel functions.
- The Kernel based-approach is described as follows:

1. Choose points from the training data. These points are called landmarks or pivot points.
2. Compute similarity/proximity of the n-training points from these landmarks.
3. Each of the n-dimensional similarity vector containing similarity of n training points from each landmark is a new *feature for the non-linear model*.

These similarity functions to compute similarity between a data point and a landmark are called *kernel functions*.

Gaussian Kernel Function

- One of the most popular kernel function used to compute similarity between the data points and landmark point is the Gaussian Kernel Function (or Gaussian Radial Bias Function (RBF)).

- A Gaussian Kernel Function is given by:

$$K(x_i, l) = e^{-\left(\frac{|x_i - l|^2}{2\sigma^2}\right)}$$

where $|x_i - l|^2$ is the length of the difference vector between the data point x_i and landmark l and σ^2 is a constant parameter that controls the behavior of the kernel function

Gaussian Kernel Function (Contd.....)

- Now, if a data point lies close to the landmark, then $|x_i - l|^2 \approx 0$ and hence

$$K(x_i, l) = e^{-\left(\frac{|x_i - l|^2}{2\sigma^2}\right)} \approx e^{-\left(\frac{0}{2\sigma^2}\right)} \approx 1$$

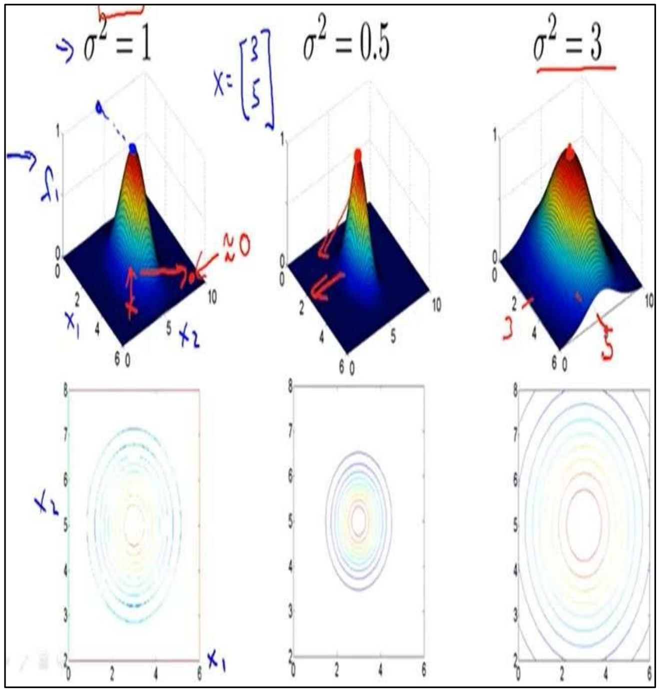
- If a data point lies far away from the landmark, then $|x_i - l|^2 \approx large\ value$ and hence

$$K(x_i, l) = e^{-\left(\frac{|x_i - l|^2}{2\sigma^2}\right)} \approx e^{-\left(\frac{large\ value}{2\sigma^2}\right)} \approx 0$$

Gaussian Kernel Function (Contd…..)

Effect of σ^2 in Gaussian Kernel

- In order to see the effect of σ^2, consider a pivot point (3,5) and the value of a feature computed from the pivot point with three different values of σ^2 *(i.e. σ^2 =1,0.5 and 3) (as shown in figure).*
- If σ^2 *is large (i.e., σ^2 =3)*, the feature f vary very smoothly. So large value of σ^2 leads to high bias, lower variance.
- *If σ^2 is small (i.e., σ^2 =0.5)*, the feature vary less smoothly. So, small value of σ^2 leads to low bias and high variance.



Kernel Method- Intuition

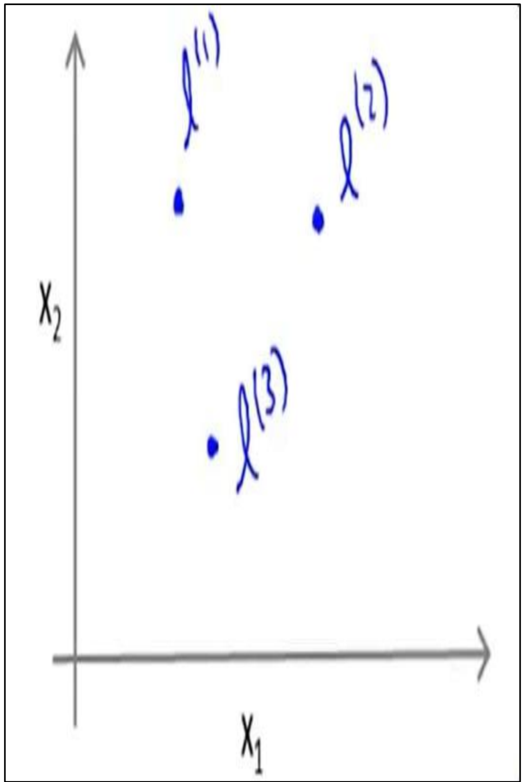
- Consider, we have training data with n examples and k features.
- Let us consider, we have chosen three land marks points (as shown in figure) of which landmark $l^{(1)}$ and $l^{(2)}$ belong to positive class and $l^{(3)}$ belongs to negative class .
- So, we will have three features f_1, f_2 and f_3 which are computed with Gaussian (or any other kernel) as follows:

$$f_1 = e^{-\left(\frac{\sum_{j=1}^{k} |x_{ij}-l_j^{(1)}|^2}{2\sigma^2}\right)}$$

$$f_2 = e^{-\left(\frac{\sum_{j=1}^{k} |x_{ij}-l_j^{(2)}|^2}{2\sigma^2}\right)}$$

$$f_3 = e^{-\left(\frac{\sum_{j=1}^{k} |x_{ij}-l_j^{(3)}|^2}{2\sigma^2}\right)}$$

Each of which is a n-dimensional column vector. So, we have transformed from $n \times k$ space to $n \times 3$ space (if we have used 3 landmarks).



Kernel Method- Intuition (Contd….)

- The hypothesis function (according to new features) is thus given by

$$f(x) = \beta_0 + \beta_1 f_1 + \beta_2 f_2 + \beta_3 f_3$$

- Let's say (according to some optimization method like stochastic gradient descent), you obtain following values of coefficients.

$$\beta_0 = -0.5, \beta_1 = 1, \beta_2 = 1, \beta_3 = 0$$

- So, if a datapoint lies close to landmark $l^{(1)}$, then $f_1 \approx 1, f_2 \approx 0, f_3 \approx 0$ and hence,

$$f(x) \approx -0.5 + 1 \times 1 + 1 \times 0 + 0 \times 0 = 0.5$$

So, it is assigned a positive class

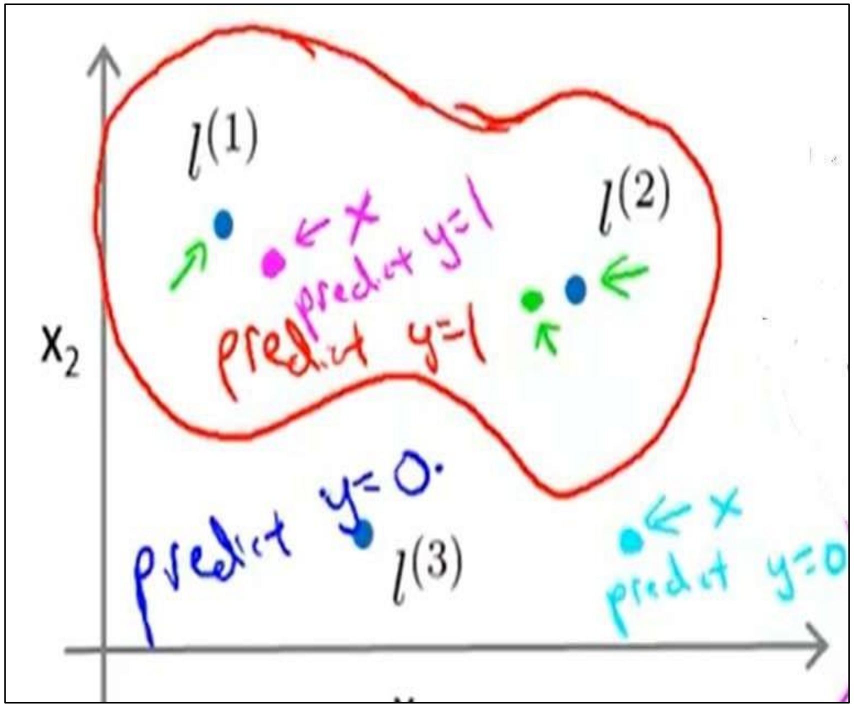
Kernel Method- Intuition (Contd….)

- If a datapoint lies close to landmark $l^{(3)}$, then $f_1 \approx 0, f_2 \approx 0, f_3 \approx 1$ and hence,

$$f(x) \approx -0.5 + 1 \times 0 + 1 \times 0 + 0 \times 1$$
$$= -0.5$$

So, it is assigned a negative class.

Hence, all points will be assigned according to the values of $\beta_0, \beta_1, \beta_2, \beta_3$ which are learnt through feature values f_1, f_2, f_3.



How to choose landmark points?

- The general tendency to choose landmark is to choose each training point as a landmark (though we may exclude the noise points from the training data).
- So, a $n \times k$ feature matrix (X) will be transformed to $n \times n$ feature matrix (X') as shown below:

$$X = \begin{bmatrix} x_{11} & \cdots & \cdots\cdots & \cdots & x_{1k} \\ x_{21} & \vdots & \vdots\ \vdots & \vdots & x_{2k} \\ \vdots & \vdots & \vdots\ \vdots & \vdots & \vdots \\ x_{n1} & \cdots & \cdots\cdots & \cdots & x_{nk} \end{bmatrix} \text{ to } X' = \begin{bmatrix} f_{11} & \cdots & \cdots\cdots & \cdots & f_{1n} \\ f_{21} & \vdots & \vdots\ \vdots & \vdots & f_{2n} \\ \vdots & \vdots & \vdots\ \vdots & \vdots & \vdots \\ f_{n1} & \cdots & \cdots\cdots & \cdots & f_{nn} \end{bmatrix}$$

where f_{ij} is the similarity between i^{th} training example and j^{th} landmark (according to some Kernel function) $1 \leq i, j \leq n$. For instance, f_{21} is the similarity between 2^{nd} training example and 1^{st} landmark point (which is the first training example).

- We also add a column of 1 in order to find the intercept term of the hypothesis function for the transformed feature matrix (using Stochastic Gradient Descent method)

Cost Function

- The cost function for non linear kernel-based method is given by:

$$J(\beta) = \frac{1}{2}\sum_{j=0}^{n} \beta_j^2 + C\sum_{i=1}^{n} \max(0, 1 - y_i f(x_i))$$

- It is different from the linear Soft SVM in two ways:

1. There are n+1 coefficients (β) as the $n \times k$ feature matrix is transformed to Kernel-based $n \times (n+1)$ matrix.
2. The hypothesis function for non-linear kernel function is given by:

$$f(x_i) = \beta_0 + \beta_1 f_{i1} + \beta_2 f_{i2} + \cdots \ldots \ldots \ldots .. \beta_n f_{in}$$

Cost Function-Optimization

- The optimization of the cost function of Kernel-based non linear SVM is done using Stochastic Gradient Descent algorithm (as in case of Soft Linear SVM model) to find the optimal value of (β) matrix

- Thus, β values are updated as:

$$\beta_j = \beta_j - learning\ rate \times \frac{\partial J(\beta)}{\partial \beta_j}$$

$$where\ \frac{\partial J(\beta)}{\partial \beta_j} = \begin{cases} \beta_j & if\ y_i f(x_i) \geq 1 \\ \beta_j - C \sum_{i=1}^{n} y_i f_{ij} & if\ y_i f(x_i) < 1 \end{cases}$$

where j=0,1,2,3…………n

Commonly used Kernel Functions

Beside, Gaussian Kernel Function, following Kernel Functions are commonly used in non-linear SVM:

$$Linear\ Kernel = K_{\mathrm{ij}} = x_{\mathrm{i}} . l_{\mathrm{j}}$$

$$Polynomial\ Kernel = K_{\mathrm{ij}} = (x_{\mathrm{i}} . l_{\mathrm{j}} + constant)^{\mathrm{degree}}$$

$$Sigmoid\ Kernel = K_{\mathrm{ij}} = \tanh(a(x_{\mathrm{i}} . l_{\mathrm{j}}) + \mathrm{b})\ \text{for constants a, b}$$

$$Log\ Kernel = K_{\mathrm{ij}} = -\log(|x_{\mathrm{i}} - l_{\mathrm{j}}|^{\mathrm{degree}}) + 1$$

Where x_{i} is any n-vector training point and l_{j} is any n-vector landmark.