AIML - CS 337

Lecture 12: Kernel Methods

26/09/2022

Lecturer: Abir De *Scribe: Group 23, Group 25, Group 26*

In this Lecture we discuss the smooth transition from Linear to Non-Linear space of functions. We discuss treatments where f(x) is non-linear in x that is, $f(x) \neq w^T x$, but we can construct it using our prior knowledge. We transform the data into a higher dimensional space and finds out that data is separable when transformed into higher dimensions. We introduce "kernel method", a method of applying SVMs to problems with non linear classification boundaries. kernels are basically measure of similarity between two vectors.

1 Introduction

Remember we had a question in midsem (question 2.d) where the labels are +1 for all x such that |x| > 2 and -1 otherwise. We were asked to provide a 1D transformation $\phi : \mathbb{R} \to \mathbb{R}$ so that SVM applied to the given dataset.

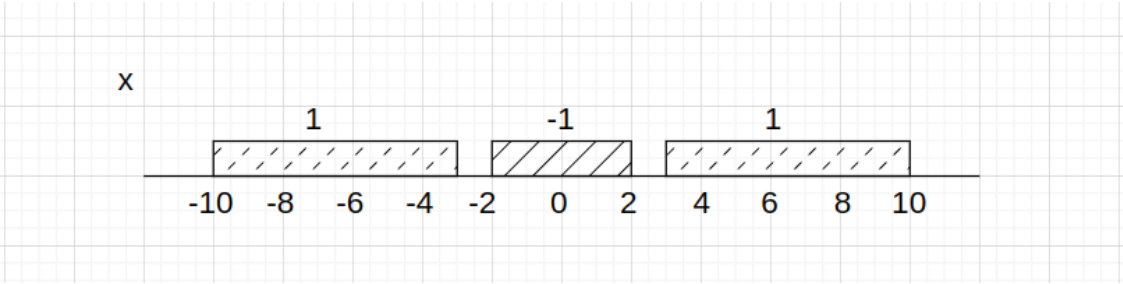

Figure 1: Initial input space : x

As it can be seen that SVM can not be applied to the initial input space.
Now let us first define a mapping $\phi : R \hookrightarrow R^2$ as follows:

$$\phi(x) = (x, x^2)$$

Now our prediction will be $h(\phi(x))$ instead of $h(x)$, where $\phi(x) = x^2$ and h is the learnt classifier, it is possible to apply SVM in the transformed space as seen in the Figure 2.

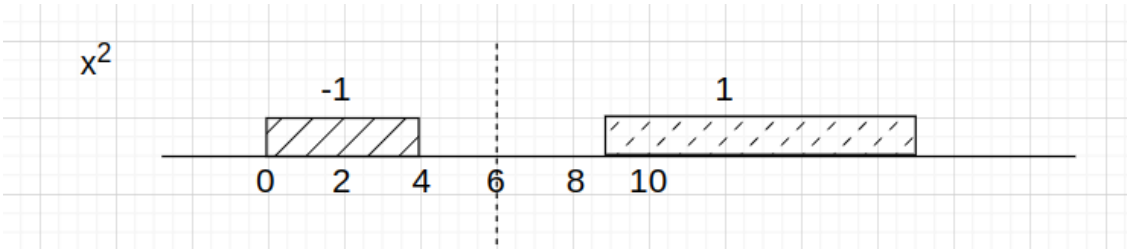

Figure 2: Transformed input space : x^2

The Main Concept behind Kernel Methods is that there is a possibility that, data-points that are not linearly-separable in lower dimensions are linearly-separable in higher dimensions.

2 Kernel Methods

The Basic Algorithm will be:-

- Given some Data Set $S = (\{\boldsymbol{x_i}\}_{i=0}^n, \{y_i\}_{i=0}^n)$, where $\boldsymbol{x_i} \in \mathbb{R}^d$
- Consider a function $\boldsymbol{\phi}$ such that $\boldsymbol{\phi(x)} \in \mathbb{R}^{d'}$ where $d' > d$ (and can even be infinity)
- Create a new Data Set $\hat{S} = (\{\boldsymbol{\phi(x_i)}\}_{i=0}^n, \{y_i\}_{i=0}^n)$
- Train a linear Predictor h over $\hat{S}$
- And then the prediction of any point $\boldsymbol{x_{test}}$ in the test dataset is given by $h(\boldsymbol{\phi(x_{test})})$

Thus the prediction is given by $\boldsymbol{w^T \phi(x)}$ where both $\boldsymbol{w}, \boldsymbol{\phi(x)} \in \mathbb{R}^{d'}$

As d' can reach infinity, this is theoretically possible for non linear functions but calculating and storing $\boldsymbol{w}$ and $\boldsymbol{\phi(x)}$ becomes practically impossible, but the dot product $\boldsymbol{w^T \phi(x)}$ is a scalar.

Here we are enriching the expressive power of halfspaces by first mapping the data into a high dimensional feature space, and then learning a linear predictor in that space. While this approach greatly extends the expressiveness of halfspace predictors, it raises both sample complexity and computational complexity challenges. We tackle this using the method of *kernels*.

A popular choice for the mapping ϕ is a polynomial mapping: $x \to (1, x, x^2, \ldots)$

The Setup:
Given a mapping ϕ we are left to solve the optimization problem:

$$\min_w f(\{y_i\}_i^n, \{\boldsymbol{w^T \phi(x_i)}\}_i^n) + \lambda R(\boldsymbol{w})$$

Where f is a loss function and R is a monotonic Regularization function.

2.1 Transformation

- In a finite case if we sample d dimension vector N times such that $d << N$. .It is with high probability that there will be d vectors that are linearly independent of each other. But as the value of d itself tends to infinity it becomes less and less likely
- In the given case the vectors $\{\boldsymbol{\phi(x_i)}\}_i^n$ are of infinite dimension. So if the optimum solution is $\boldsymbol{w^*}$ then there exists $\{\alpha_i\}_i^n$ such that $\boldsymbol{w^*}$ can be represented as

$$\boldsymbol{w^*} = \sum_{i=0}^{n} \alpha_i * \boldsymbol{\phi(x_i)} \ + \boldsymbol{v}$$

  where $\boldsymbol{v^T \phi(x_i)} = 0$ for all i, that is, **v** is orthogonal to the span of the vectors mapped by the function ϕ.
- Let us define $\boldsymbol{w} = \boldsymbol{w^*} - \boldsymbol{v}$

$$\begin{aligned}
&||\boldsymbol{w}||^2 = ||\boldsymbol{w^*} - \boldsymbol{v}||^2 \\
&||\boldsymbol{w}||^2 = ||\boldsymbol{w^*}||^2 - 2\boldsymbol{w^*}.\boldsymbol{v} + ||\boldsymbol{v}||^2 \qquad \text{As the value } \boldsymbol{w^*}.\boldsymbol{v} = \boldsymbol{v}.\boldsymbol{v} = ||\boldsymbol{v}||^2 \\
&||\boldsymbol{w}||^2 = ||\boldsymbol{w^*}||^2 - 2||\boldsymbol{v}||^2 + ||\boldsymbol{v}||^2 \\
&||\boldsymbol{w}||^2 = ||\boldsymbol{w^*}||^2 - ||\boldsymbol{v}||^2
\end{aligned}$$

- So as norm is a positive function we have $||\boldsymbol{w}|| \leq ||\boldsymbol{w^*}||$
- And since R is non-decreasing, we obtain $R(\boldsymbol{w}) \leq R(\boldsymbol{w^*})$
  (R is monotonic regularization function as defined above)

- Also as $\boldsymbol{w^T\phi(x)} = (\boldsymbol{w^*} - \boldsymbol{v})^T\boldsymbol{\phi(x)} = \boldsymbol{w^{*^T}\phi(x)}$
  (since v is orthogonal to $\boldsymbol{\phi(x)}$, we have $\boldsymbol{v^T\phi(x)}$=0)
- And so,

$$f(\{y_i\}_i^n, \{\boldsymbol{w^T\phi(x_i)}\}_i^n) = f(\{y_i\}_i^n, \{\boldsymbol{w^{*^T}\phi(x_i)}\}_i^n)$$

- We have shown that the loss function is same for both $\boldsymbol{w^*}$ and $\boldsymbol{w}$ and the regularization function is less for $\boldsymbol{w}$ than $\boldsymbol{w^*}$,
- Hence the objective function for $\boldsymbol{w}$ is less than that for $\boldsymbol{w^*}$, but as $\boldsymbol{w^*}$ is the optimum solution, we must have that $\boldsymbol{w}$ is also an optimum solution .
- Hence we have proved that the value

$$\boldsymbol{w} = \sum_{i=0}^{n} \alpha_i * \boldsymbol{\phi(x_i)}$$

  is an optimum solution for the objective function.[1]

Theorem 2.1 (**Representer Theorem**). *Given a mapping from $\mathbb{R}^d$ to $\mathbb{R}^{d'}$, there exists a vector $\boldsymbol{\alpha} \in \mathbb{R}^{d'}$ such that $\boldsymbol{w} = \sum_{i=1}^{d'} \alpha_i\boldsymbol{\phi(x_i)}$ is an optimal solution.*

$$\min_w f(\{y_i\}_i^n, \{\boldsymbol{w^T\phi(x_i)}\}_i^n) + \lambda R(\boldsymbol{w})$$

2.2 Kernel

Substituting the fact that $\boldsymbol{w} = \sum_{i=1}^{d'} \alpha_i\boldsymbol{\phi(x_i)}$ is an optimum solution into the value $\boldsymbol{w^T\phi(x)}$

$$\begin{aligned}
\boldsymbol{w^T\phi(x)} &= \left(\sum_{i=1}^{d'} \alpha_i\boldsymbol{\phi(x_i)^T}\right)\boldsymbol{\phi(x)} \\
&= \sum_{i=1}^{d'} \alpha_i\boldsymbol{\phi(x_i)^T\phi(x)} \qquad \text{consider a function } K(\boldsymbol{x_i}, \boldsymbol{x_j}) = \boldsymbol{\phi(x_i)^T\phi(x_j)} \\
&= \sum_{i=1}^{d'} \alpha_i K(\boldsymbol{x_i}, \boldsymbol{x})
\end{aligned}$$

[1] Also remember that this was the result that we got by applying the Lagrange multiplier on the objective functions

Here K is called a *Kernel* Function and it denotes *Similarity* between the data points $\boldsymbol{x_i}$ and $\boldsymbol{x_j}$.

What does this expression physically mean? We want to make a prediction at a new test point x, then $w^{*T}\phi(x)\ y(x)$ gives the **weighted mean** of labels $y(x_i)$ with weights as similarity

$$S(x, x_i) \ = \ \phi(x_i)^T\phi(x)$$

thus giving more weight to similar neighbours. This essentially represents the idea of K-Means clustering.
We can note that this similarity function can be any form, not necessarily dot product, The term "kernels" is used in this context to describe inner product in the feature space

Implications of this result

- This representation takes us beyond SVM
- w is linear combination of $\phi(x_i)$ thus $w^T\phi(x)$ can be easily calculated.

Some Popular Kernels are:

- **SVM** $K(\boldsymbol{x_i}, \boldsymbol{x_j}) = \boldsymbol{x_i^T x_j}$
- **Gaussian** $K(\boldsymbol{x_i}, \boldsymbol{x_j}) = e^{\frac{-||\boldsymbol{x_i}-\boldsymbol{x_j}||^2}{\sigma^2}}$
- **k degree polynomial** $K(\boldsymbol{x_i}, \boldsymbol{x_j}) = (1 + \langle \boldsymbol{x_i}, \boldsymbol{x_j} \rangle)^k$

where $\langle ., . \rangle$ denotes the inner product.

Exercise

Que- Show that RBF can be written as inner product

It is easy to see that RBF(or Gaussian) Kernel satisfies the positive semidefinite condition, we can infact state and prove a possible inner product representation

Consider

$$\phi(x) \ = \ \sum_{n=0}^{\infty} \frac{1}{\sqrt{n!}} e^{-\frac{x^2}{2}} x^n$$

Observe that

$$< \phi(x), \phi(x^{'}) > = \sum_{n=0}^{\infty} \left(\frac{1}{\sqrt{n!}} e^{-\frac{x^2}{2}} x^n \right) \left(\frac{1}{\sqrt{n!}} e^{-\frac{x^{'2}}{2}} x^{'n} \right)$$

$$< \phi(x), \phi(x^{'}) > = e^{-\frac{x^2 + x^{'2}}{2}} \sum_{n=0}^{\infty} \left(\frac{(xx^{'})^n}{n!} \right)$$

$$< \phi(x), \phi(x^{'}) > = e^{-\frac{x^2 + x^{'2}}{2} + xx^{'}}$$

$$< \phi(x), \phi(x^{'}) > = e^{-\frac{(x - x^{'})^2}{2}}$$

Que- Do the same for polynomial kernel
Left for reader
Hint: Solution can be checked in book Understanding machine learning chapter Kernel Methods

3 SVM Kernel

Now going back to the dual formulation of Hard-SVM:

$$\max_{\boldsymbol{\alpha} \in R^D : \boldsymbol{\alpha} \geq \mathbf{0}} \left(\sum_{i=1}^{D} \alpha_i - \frac{1}{2} \sum_{i=1}^{D} \sum_{j=1}^{D} \alpha_i \alpha_j y_i y_j \boldsymbol{x_i^T x_j} \right)$$

$$= \max_{\boldsymbol{\alpha} \in R^D : \boldsymbol{\alpha} \geq \mathbf{0}} \left(\sum_{i=1}^{D} \alpha_i - \frac{1}{2} \sum_{i=1}^{D} \sum_{j=1}^{D} \alpha_i \alpha_j y_i y_j K(\boldsymbol{x_i}, \boldsymbol{x_j}) \right)$$

$$= \max_{\boldsymbol{\alpha} \in R^D : \boldsymbol{\alpha} \geq \mathbf{0}} \left(\sum_{i=1}^{D} \alpha_i - \frac{1}{2} (\boldsymbol{\alpha} * \boldsymbol{Y})^{\boldsymbol{T}} \boldsymbol{G} (\boldsymbol{Y} * \boldsymbol{\alpha}) \right)$$

$*$ denotes element-wise multiplication

$\boldsymbol{G} = \{K(\boldsymbol{x_i}, \boldsymbol{x_j})\}_{i,j=1}^{D \times D}$ Known as the **Gram matrix**

Hence, we are optimising for $\boldsymbol{\alpha}$ instead of $\boldsymbol{w}$.

The advantage of working with kernels rather than directly optimizing w in the feature space is that in some situations the dimension of the feature space is extremely large while implementing the kernel function is very simple.

Theorem 3.1 (Mercers's Theorem)**.** $K : [a, b] \times [a, b] \to \mathbb{R}$ *is a kernel if and only if :*

1. *Symmetric :* $K(x, y) = K(y, x)$ *for all* $x, y \in [a, b]$*, and*
2. *Positive Semi-definite :* $\sum_{j=1}^{n} \sum_{j=1}^{n} K(x_i, x_j)c_i c_j \geq 0$ *for all finite sequences of points* $\mathbf{x}_1, ..., \mathbf{x}_n$ *of* $[a, b]$ *and all choices of real numbers* $c_1, ..., c_n$

References

[1] S. B.-D. Shai Shalev-Shwartz. *Understanding Machine Learning, Chapter 16*. Cambridge University Press, 2014.