Chapter 1

Lecture 2022/03/22

ELL888 Indian Institute of Technology Delhi
Lecture Kernel methods, SVM regression: An overview
Scribed by: *Hari Krishnan CK*
Instructors: Prof.Sandeep Kumar and Prof.Jayadeva

Contents

1.1 Recap of previous lectures and basics

In the previous lectures we have gone through the definition of Hilbert space and RKHS. We had also gone through the formulation of SVM and MCM for the linear case. We will go over them briefly below as recap.

1.1.1 Hilbert space

A vector space equipped with an inner product is defined as an inner product space.An inner product induces the notion of norm which is defined as $\|x\| = \langle x, x\rangle^{\frac{1}{2}}$. Norm introduces the notion of distance between points $d(x, y) = \langle x, y\rangle^{\frac{1}{2}}$.
A sequence in a metric space is termed as Cauchy is there exists an N for all ζ¿0 such that

$$d\left(x_n, x_m\right) < \zeta \quad \forall m, n \geq N$$

.The space where all Cauchy sequences are convergent is called a complete space.A complete inner product space is called as a Hilbert space.So in short an inner product space where all Cauchy sequences converge is known as a Hilbert space.

1.1.2 Kernels

Given an abstract space $\mathcal{X}$ a function which maps $\kappa : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}$ is called a kernel function. Kernel functions are used to quantify the similarity between two points $\mathbf{x}$ and $\mathbf{x}'$ in $\mathcal{X}$. A kernel function typically satisfies the following two properties (but this is not required for all kernel methods).

$$\text{(symmetric) } \forall \mathbf{x}, \mathbf{x}' \in \mathcal{X}, \kappa\left(\mathbf{x}, \mathbf{x}'\right) = \kappa\left(\mathbf{x}', \mathbf{x}\right)$$
$$\text{(non-negative) } \forall \mathbf{x}, \mathbf{x}' \in \mathcal{X}, \kappa\left(\mathbf{x}, \mathbf{x}'\right) \geq 0$$

1.1.3 SVM for linearly separable case

We will briefly go over the optimisation problem for SVM in linear case.The main idea is to maximise the margin between the two classes which is given by $\frac{2}{\|w\|}$.Instead of maximising the margin we solve a minimisation problem which tries to maximise margin and also tries to correctly classify samples.

$$\begin{aligned} \min \quad & \frac{1}{2} w^T w \\ \text{subject to} \quad & y_i \left[w^T x_i + b\right] \geq 1, i = 1, \ldots, l. \end{aligned}$$

The corresponding dual is obtained by writing the Lagrangian and KKT conditions. The dual optimisation problem is given by

$$\max \quad q(\mu) = \sum_{i=1}^{l} \mu_i - \frac{1}{2} \sum_{i,j=1}^{l} \mu_i \mu_j y_i y_j x_i^T x_j$$

$$\text{subject to} \quad \mu_i \geq 0, \forall i$$

$$\sum_{i=1}^{l} \mu_i y_i = 0$$

For a convex optimisation problem both primal and dual will give the optimum solution.

1.2 Introduction to Kernels

1.2.1 Kernel function and Kernel matrices

Given an abstract space $\mathcal{X}$ a function which maps $\kappa : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}$ is called a kernel function. Kernel functions are used to quantify the similarity between two points $\mathbf{x}$ and $\mathbf{x}'$ in $\mathcal{X}$. A kernel function typically satisfies the following two properties (but this is not required for all kernel methods).

$$\text{(symmetric) } \forall \mathbf{x}, \mathbf{x}' \in \mathcal{X}, \kappa(\mathbf{x}, \mathbf{x}') = \kappa(\mathbf{x}', \mathbf{x})$$
$$\text{(non-negative) } \forall \mathbf{x}, \mathbf{x}' \in \mathcal{X}, \kappa(\mathbf{x}, \mathbf{x}') \geq 0$$

Suppose we are given a kernel function κ, and elements $x_1, x_2, \ldots, x_m \in \chi$, $m \times m$ matrix K such that

$$K_{ij} = \kappa(x_i, x_j) = K_{ji}$$

then the matrix K is called as gram matrix of the kernel function κ w.r.t the m elements of the domain set χ. A Positive Definite Kernel is the kernel function which satisfies the equation given below:

$$\sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j k(x_i, x_j) \geq 0$$

Where, $\alpha_t \in \mathbb{R}$.We will be discussing more on Positive Definite Kernels while discussing Mercer's theorem.

1.2.2 Cover's Theorem

Cover's theorem is a fundamental theorem on which the basis of kernel based methods lie. The theorem states that if we have a data which is not linearly separable then we should be able to transform the data to a sufficiently high dimension via a non-linear transform where it is linearly separable.
An example of the data which is not linearly separable in 2D but is separable in 3D is given below. Note that we don't need to transform the data to very high dimensions always to get linearly separability, it could be possible with just increasing one dimension via a non-linear map. The idea is illustrated in the image given below

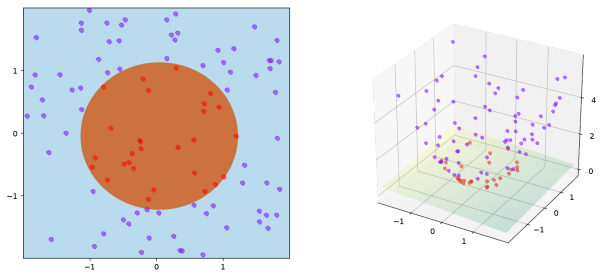

Figure 1.1: Data linearly non-separable in 2D becomes linearly separable in 3D

1.2.3 Mercer's theorem

Assume χ be an arbitrary non empty set of features. A function $k : \chi \times \chi \to \mathbb{R}$ is called a Mercer kernel function if there exists a Hilbert space $\mathcal{H}$ and a mapping ϕ which is defined as

$$\phi : \chi \to \mathcal{H} \quad \text{s.t. } \forall x_1, x_2 \in \chi$$
$$k\left(x_1, x_2\right) = \left\langle \phi\left(x_1\right), \phi\left(x_2\right) \right\rangle_{\mathcal{H}}$$

It can also be defined as a kernel function which satisfies the Mercer's condition.There exists a mapping Φ and an expansion

$$K(\mathbf{x}, \mathbf{y}) = \sum_i \Phi(\mathbf{x})_i \Phi(\mathbf{y})_i$$

for a kernel if and only if, for all real valued functions $g(\mathbf{x})$ such that $\int g(\mathbf{x})^2 d\mathbf{x}$ is finite then

$$\int K(\mathbf{x}, \mathbf{y}) g(\mathbf{x}) g(\mathbf{y}) d\mathbf{x} d\mathbf{y} \geq 0.$$

The kernel functions should satisfy the above equation which is the Mercer's condition to be called a Mercer's kernel. If the kernel function satisfies Mercer's theorem then the dot product between two points in the Hilbert space can be replaced with the kernel function value between the same two points.

1.2.4 Popular kernels

Linear kernels

Linear kernels are obtained by the dot product between the two vectors.

$$\kappa\left(\mathbf{x}, \mathbf{x}'\right) = \mathbf{x}^T \mathbf{x}'$$

and the corresponding mapping is $\phi(x) = x$.

Polynomial kernels

Polynomial kernel function is defined as $\mathrm{K}\left(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}\right) = \left(\mathbf{x}^{(i)^T} \mathbf{x}^{(j)} + 1\right)^d$ where d is the dimensionality of x. Quadratic kernels are used mostly in NLP problems where high dimensionality of RBF kernels might be a problem.

Gaussian kernels

A Gaussian kernel is defined as

$$\kappa\left(\mathbf{x}, \mathbf{x}'\right) = \exp\left(-\frac{1}{2} \sum_{j=1}^{p} \frac{1}{\sigma_j^2} \cdot \left(x_j - x_j'\right)^2\right)$$

The dimensionality of the corresponding mapping Φ obtained is infinity.

1.3 Primal vs Dual SVM

The primal formulation of SVM in non-linear case is

$$\min \frac{1}{2} w^T w + C \sum_{i=1}^{l} \xi_i$$

subject to

$$y_i \left[w^T x_i + b\right] \geq 1 - \xi_i, i = 1, \ldots, l$$
$$\xi_i \geq 0, i = 1, \ldots, l.$$

Here ξ_i's are the slack variables, (x_i,y_i) are training data provided. The KKT conditions for the above optimisation problem are
- K1. $w - \sum_{i=1}^{l} \mu_i y_i x_i = 0$.
- K2. $\sum_{i=1}^{l} \mu_i y_i = 0$.
- K3. $C - \mu_i - \lambda_i = 0$.
- K4.1 $- \xi_i - y_i \left(w^T x_i + b\right) \leq 0$.
- K5. $\xi_i \geq 0, \mu_i \geq 0, \lambda_i \geq 0, \forall i$.
- K6. $\mu_i \left[1 - \xi_i - y_i \left(w^T x_i + b\right)\right] = 0, \forall i$.
- K7. $\lambda_i \xi_i = 0, \forall i$

The corresponding dual formulation of the problem is given by :

$$\max \quad q(\mu) = \sum_{i=1}^{l} \mu_i - \frac{1}{2} \sum_{i,j=1}^{l} \mu_i \mu_j y_i y_j x_i^T x_j$$

subject to

$$0 \leq \mu_i \leq C, \forall i$$
$$\sum_{i=1}^{l} \mu_i y_i = 0$$

The solution of the primal is an upper bound on the solution of dual and solution of dual is lower bound on the solution of primal.For a convex optimisation problem both the solutions become same which is the optimal solution.

1.4 Kernel SVM

For non-linear case we may not get good results with the non-linear optimisation problem in the above section. Another approach is to use nonlinear SVM's which uses kernel idea to project the data to a high dimensional space without actually knowing the mapping or the space. The optimisation problem for the nonlinear SVM case is given by

$$\max q(\mu) = \sum_{i=1}^{l} \mu_i - \frac{1}{2} \sum_{i,j=1}^{l} \mu_i \mu_j y_i y_j K\left(x_i, x_j\right) \text{ subject to } 0 \leq \mu_i \leq C, \quad i = 1, \ldots, l \sum_{i=1}^{l} \mu_i y_i = 0$$

The value of b and W is obtained by

$$f(x) = \sum_{i \in S} \mu_i y_i K\left(x_i, x\right) + b.$$

Here, $S = \{i : \mu_i > 0\}$ and b is given by

$$b = y_j - \sum_{i \in S} \mu_i y_i K\left(x_i, x_j\right)$$

The above optimisation problem is Quadratic Optimisation problem (QP) with one equality and one inequality constraints.

1.5 PCA vs Kernel PCA

1.5.1 PCA

We assume that the data we have is mean centered i.e. given observations $\mathbf{x}_k, k = 1, \ldots, M, \mathbf{x}_k \in \mathbf{R}^N, \sum_{k=1}^{M} \mathbf{x}_k = 0$. PCA tries to diagonalise the covariance matrix given by $C = \frac{1}{M} \sum_{j=1}^{M} \mathbf{x}_j \mathbf{x}_j^\top$ by solving the eigen value problem $\lambda \mathbf{v} = C\mathbf{v}$. Then we get the projections of the data on these eigen vectors to get the new samples.The main disadvantage of PCA is that we may loose some cluster properties and neighbourhood infformation since PCA is basically a linear transform. To deal with this drawback one can make use of kernel PCA which is discussed below.

1.5.2 Kernel PCA

In kernel PCA, we first project the data to a very high dimension so that it becomes linear in the high dimension and then we transform the data to a lower dimension using a linear transform. /newline Emperical co-variance operator is given by.

$$\hat{\Sigma} = \frac{1}{n} \sum_{i=1}^{n} \phi\left(x^i\right) \otimes \phi\left(x^i\right)$$

We want to get direction where variance is maximised in the new space. i.e like in PCA we want to solve eigen value problem in the high dimensional space.

$$(\hat{\Sigma})(\hat{\phi}) = \lambda(\hat{\phi})$$

We can assume

$$\hat{\phi} = \sum_{i=1}^{n} \alpha_i \phi\left(x^i\right)$$

where $\alpha_i \in R$. This is valid because consider $\hat{\phi} = \hat{\phi}^{\perp} + \hat{\phi}^{in}$ where $\hat{\phi}^{in}$= component of $\hat{\phi}$ is in span of $\phi\left(x^i\right)$ and $\hat{\phi}^{\perp}$= component of $\hat{\phi}$ perpendicular to the span of $\phi(x')$. $\hat{\phi}^{\perp} \cdot \phi\left(x^i\right) = 0 \quad \forall i = 1.n$. Hence it is a valid assumption. Multiplying by $\hat{\phi}$

$$\hat{\Sigma}(\hat{\phi}) = \frac{1}{n} \sum_{i=1}^{n} \left\langle \phi\left(x^i\right), \hat{\phi} \right\rangle \phi\left(x^i\right).$$

The eigen vector equation becomes

$$\hat{\Sigma} \left(\sum_{i=1}^{n} \alpha_i \phi\left(x^i\right) \right) = \lambda \sum_{i=1}^{n} \alpha_i \phi\left(x^i\right)$$

From above we have,

$$\text{LHS } = \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \left\langle \phi\left(x^i\right), \phi\left(x^j\right) \right\rangle \phi\left(x^j\right)$$

But $k\left(x^i, x^j\right) = \left\langle \phi\left(x^i\right), \phi\left(x^j\right) \right\rangle$ so the equation becomes

$$\frac{1}{n} \sum_{i,j=1}^{n} \alpha_i k\left(x^i, x^J\right) \phi\left(x^{\mathrm{j}}\right) = \lambda \sum_{i=1}^{n} \alpha_i \phi\left(x^i\right)$$

taking inner product with $\phi\left(x^l\right) \quad l = 1, 2 \ldots n$

$$\frac{1}{n} \sum_{i,j=1}^{n} \alpha_i k\left(x^i, x^J\right) k\left(x^j, x^\ell\right) = \lambda \sum_{i=1}^{n} \alpha_i k\left(x^i, x^\ell\right)$$

In matrix form

$$K^2\alpha = \lambda_n K\alpha \quad K \in R^{n\times n}.$$

so

$$K\alpha = \lambda n\alpha$$

. which is similar to the eigen value problem in normal PCA but here instead of the covariance matrix use the Kernal matrix K.

Examples showing the effectiveness of Kernel PCA

The code for the examples and figures are available at:
`https://github.com/harikuttan7136/Scribing-Kernel-PCA-Kernel-SVM-SVM-Regression/tree/main/Scribing/Code%20and%20Notebooks`

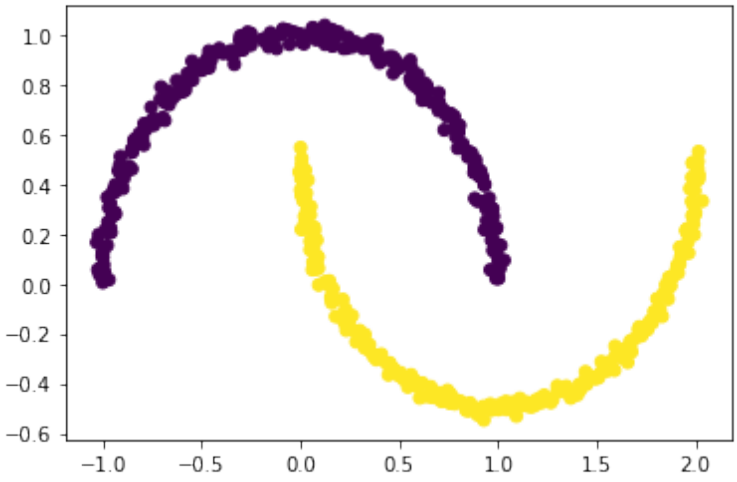

Figure 1.2: Original 2D data

1.6 Kernel optimisation

Till now in all the algorithms the kernels were fixed i.e. we chose kernels depending on our intuitions. This section deals with learning the data-dependent kernel along with the optimisation problem.Let us see one of the methods of optimising kernel.For this we define the following class separability measure

$$J = \frac{\operatorname{tr} S_b}{\operatorname{tr} S_w}$$

where

$$S_b = \frac{1}{m}\sum_{i=1}^{2} m_i \left(\bar{y}_i - \bar{y}\right)\left(\bar{y}_i - \bar{y}\right)^T$$

$$S_w = \frac{1}{m}\sum_{i=1}^{2}\sum_{j=1}^{m_i} \left(y_j^i - \bar{y}_i\right)\left(y_j^i - \bar{y}_i\right)^T$$

S_b is called the Between-class cluster and S_w is called within class cluster.We can derive the kernel formulation of class separability measure and solve the optimisation problem. Since the algorithm or technique was covered in detail in another lecture we are going to skip the details in this notes.

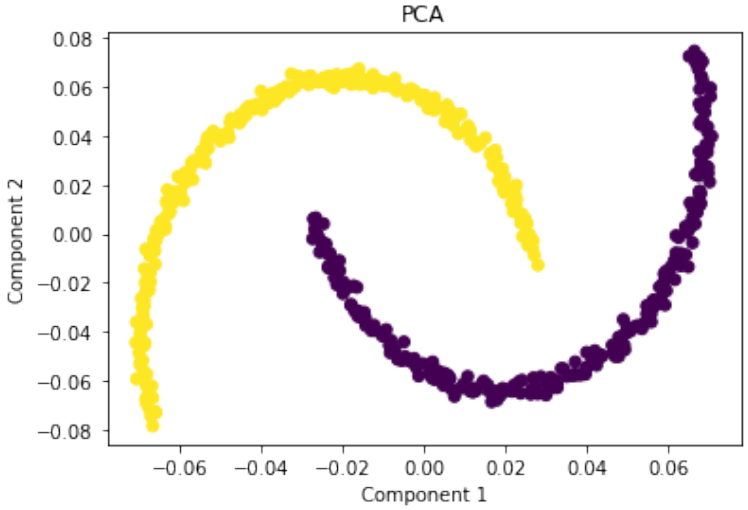

Figure 1.3: Data after doing normal PCA in 2D

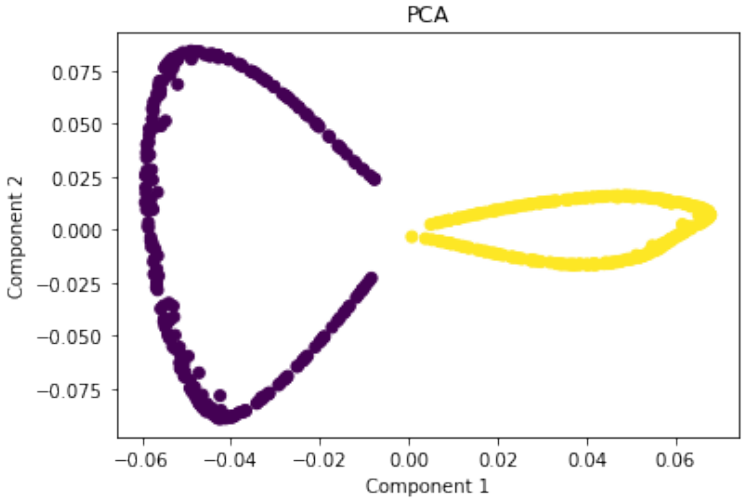

Figure 1.4: Data after doing Kernel PCA in 2D

1.7 SVM regression non-linear case

We will now discuss how to use the SVM for solving the regression problem. Here we will discuss ϵ tube regression idea where the loss function is defined as

$$L(y, g(x, w, b)) = 0 \quad \text{if } |y - g(x, w, b)| < \epsilon$$
$$L(y, g(x, w, b)) = |y - g(x, w, b)| - \epsilon \quad \text{otherwise.}$$

The primal optimisation problem for the above loss function is given by

$$\min \quad \frac{1}{2} w^T w + C \sum_{i=1}^{l} (\xi_i + \xi_i')$$
$$\text{subject to } y_i - w^T \phi(x_i) - b \leq \epsilon + \xi_i, i = 1, \ldots, l$$
$$w^T \phi(x_i) + b - y_i \leq \epsilon + \xi_i', i = 1, \ldots, l$$
$$\xi_i \geq 0, i = 1, \ldots, l$$
$$\xi_i' \geq 0, i = 1, \ldots, l$$

The linear case is simpler and the optimisation problem is exactly equal to the above optimisation problem without the slack variables (ξ_i). The corresponding dual of the problem can be solved and the dual optimisation problem is given by

$$\max \sum_{i=1}^{l} y_i (\mu_i - \mu_i') - \epsilon \sum_{i=1}^{l} (\mu_i + \mu_i') - \frac{1}{2} \sum_{i,j=1}^{l} (\mu_i - \mu_i') (\mu_j - \mu_j') \phi(x_i)^T \phi(x_j)$$
$$\text{subject to} \quad 0 \leq \mu_i, \mu_i' \leq C, \forall i$$
$$\sum_{i=1}^{l} (\mu_i - \mu_i') = 0$$

Although we get good results for the regression using the above optimisation technique there is a big problem at the heart of the primal equation. There is no physical significance for W in the above optimisation problem while in classification formulation minimising W leads to higher margin. Here there is no physical meaning for minimising W other than the notion of considering it as a regularisation term. We will discuss more about the significance of W in the section Regression to Classification.

1.7.1 Non-linear SVM regression using Kernels

We can extend the above idea of regression to use kernel methods which will ensure more linearity at higher dimensions according to the Cover's theorem.The optimisation problem is very closely related to the dual formulation for regression with the exception of dot product being replaced by a positive semi-definite kernel matrix K.

$$\max \sum_{i=1}^{l} y_i (\mu_i - \mu_i') - \varepsilon \sum_{i=1}^{l} (\mu_i + \mu_i') - \frac{1}{2} \sum_{i,j=1}^{\ell} (\mu_i - \mu_i') (\mu_j - \mu_j') K(x^i, xj)$$
$$\text{subject to} \quad 0 \leqslant \mu_i, \mu_i' \leq C$$
$$\sum_{i=1}^{l} (\mu_i - \mu_i') = 0.$$

1.8 Regression to classification

As already mentioned one problem with the regression formulation of SVM is that there is no physical significance for minimising W.Another very important limitation is that there is no notion of model complexity or VC dimension in case of SVR(Support Vector Regression). In this section we will see how Bi and Bennett tried to convert the regression problem to a classification problem so that all notions and ideas like VC dimension,etc can be applied to a regression problem as well.
The basic idea is that we create two classes for +1 class and for -1 class. Once we have the two classes we can apply all the SVM algorithms developed for classification problem including dual and kernel formulations.

$$D^+ = \left\{ \begin{pmatrix} x_i \\ y_i + \varepsilon \end{pmatrix}, i = 1, \dots l \right\}.$$

$$D^- = \left\{ \begin{pmatrix} x_i \\ y_i - \varepsilon \end{pmatrix}, i = 1, \dots l \right\}.$$

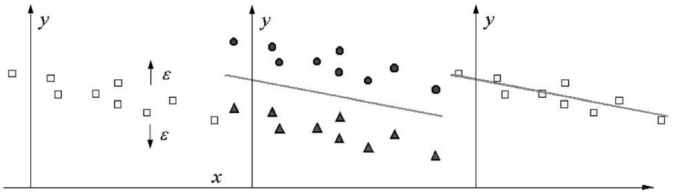

Figure 1.5: Conversion from regression to classification

1.9 Experiments and Implementations

In this section we will showcase the results of implementing the above algorithms in python. The code and results will be available in github at :

`https://github.com/harikuttan7136/Scribing-Kernel-PCA-Kernel-SVM-SVM-Regression/tree/main/Scribing/Code%20and%20Notebooks`

1.9.1 DUAL VS PRIMAL SVM

The optimisation codes for primal and dual SVM's were written and results compared.

Primal SVM results

Weights obtained for the above data: -1.59871784, 1.17456189 Bias obtained:1.74919393

Dual SVM results

Weights obtained for the above data: -1.58388965, 0.94189243 Bias obtained:1.91898122 Number of support vectors=3
As one can observe the weights obtained from primal and dual does not exactly match but they are within an error range.

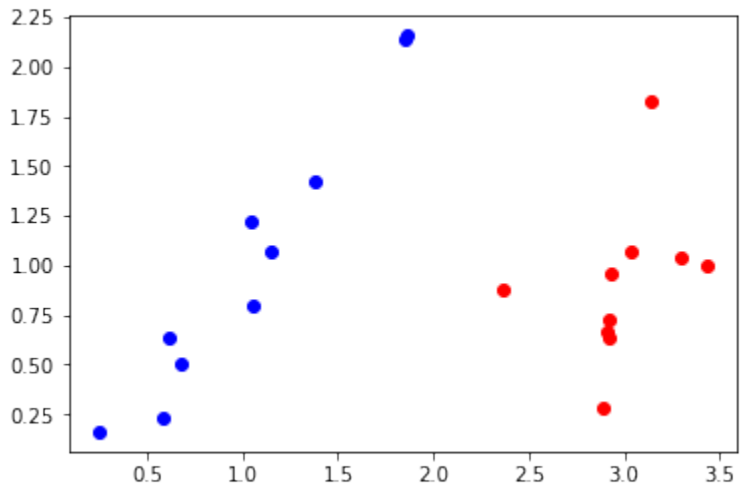

Figure 1.6: Data used for training SVM

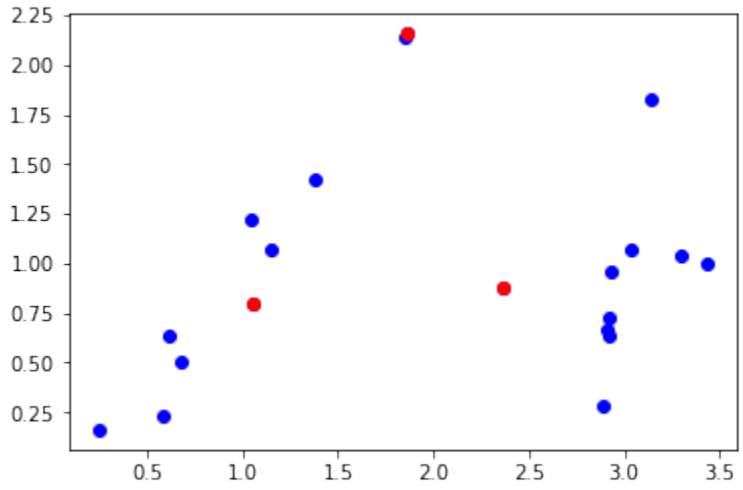

Figure 1.7: Red dots are the support vectors for the data in figure 1.6

1.9.2 Kernel SVM

This subsection deals with results obtained with the kernel implementation of the SVM.We will try to compare the results obtained when data is non-linearly separable in original dimension.

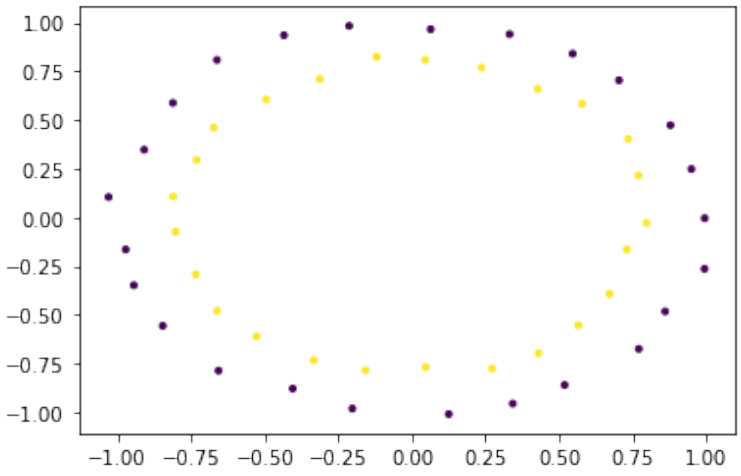

Figure 1.8: Original Data

Kernel SVM accuracy score on training data: 0.52
Linear SVM accuracy score on training data: 0.50
As we can clearly see kernel SVM performs better on non-linear data as it will project the data to a very high dimension where the data will be linearly separable. Note that there is no significant increase because the optimisation code for kernel SVM is not optimised

1.9.3 Kernel PCA

We will now look at how kernel methods can enhance dimensionality reduction techniques like PCA.
From Figure 1.7 and 1.8 it is very clear that normal PCA is unable to infer information about the labels or geometric structure of the data whereas kernel PCA(Guaussian kernel) was able to preserve the geometric information of the data.

Figure 1.9 shows the distribution of original data. When we apply normal PCA on this we get an overlapped region on as we can see in Figure 1.10. But Figure 1.11 shows that this problem does not exist in kernel PCA as data is well separated for the two classes.

1.9.4 SVM Regression

In this section we will compare the results from normal SVM regression and regression using Gaussian kernel.

Non linear case

r2 score obtained for non-linear data with kernel SVM:0.9999933197143883. As it is clear from the r2-score we were able to get good fit even on the non linear data.

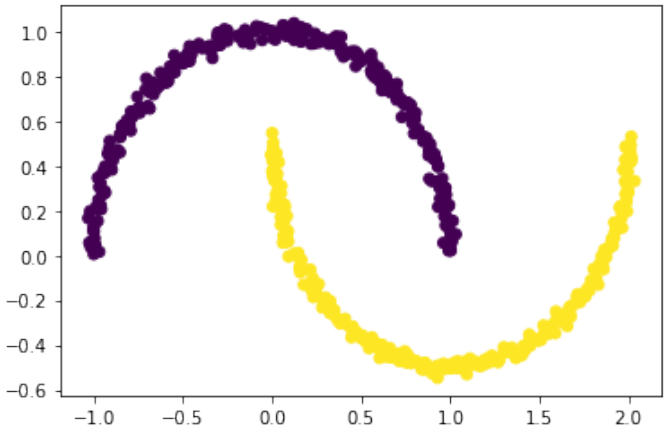

Figure 1.9: Original Data

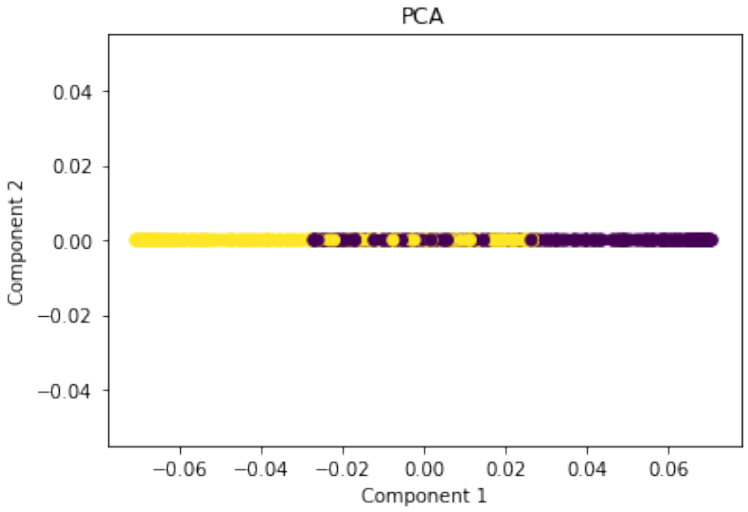

Figure 1.10: Normal PCA in 1D

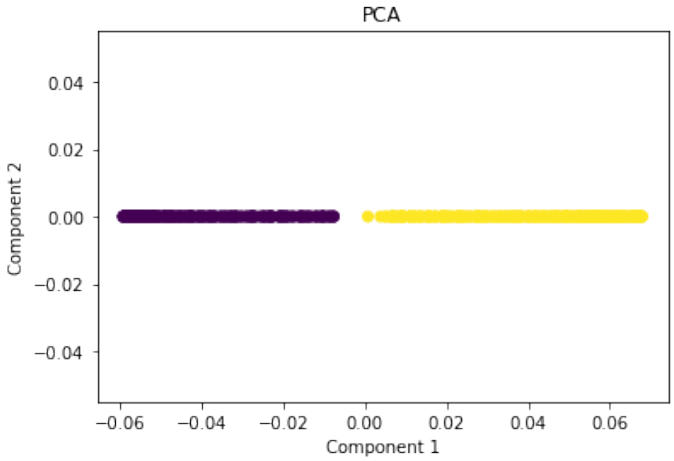

Figure 1.11: Kernel PCA in 1D

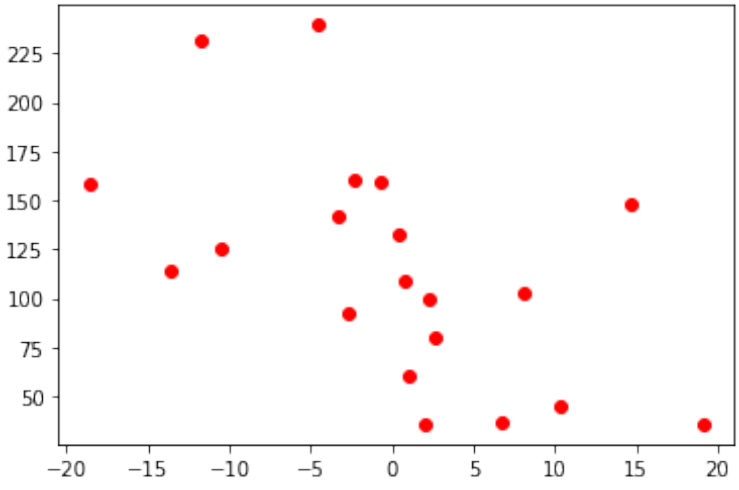

Figure 1.12: Non linear data

1.9.5 Regression to classification

In this section we will see the implementation of Bi and Bennett of converting regression to classification and then using SVM classifier to get the weights of the line.
The basic idea is that we create two classes for +1 class and for -1 class. Once we have the two classes we can apply all the SVM algorithms developed for classification problem including dual and kernel formulations.

$$D^+ = \left\{ \begin{pmatrix} x_i \\ y_i + \varepsilon \end{pmatrix}, i = 1, \dots l \right\}.$$

$$D^- = \left\{ \begin{pmatrix} x_i \\ y_i - \varepsilon \end{pmatrix}, i = 1, \dots l \right\}.$$

Original equation of the line from which data is sampled Y = 6x1+5x2. We make two datasets for +1 class and -1 class. Theory is discussed in detail in section 1.8. The equation obtained by solving the classification problem using SVM is 6.00*x1+5.00*x2. As one can observe we got the same equation of the regression line.

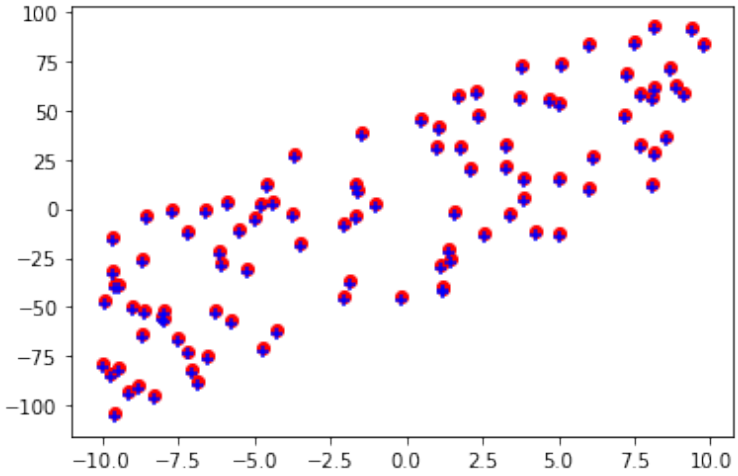

Figure 1.13: Classification problem for the regression problem Y=6x1+5x2.Dots are -1 class and + is +1 class

Bibliography

[1] A geometric approachto support vector regression Jinbo Bi , Kristin P. Bennett

[2] An Introduction to Support Vector Machines P. S. Sastry

[3] Nonlinear Component Analysis as a Kernel Eigenvalue Problem Bernhard Sch¨olkopf Alexander Smola Klaus-Robert M ¨uller

[4] Optimizing the Kernel in the Empirical Feature Space Huilin Xiong, M. N. S. Swamy, Fellow, IEEE, and M. Omair Ahmad, Fellow, IEEE

[5] Wikipedia