Machine Learning : Assignment 1

Gaurav Parashar - 17093

June 07 2020

Question 1:

a) Necessary and sufficient conditions on matrix $\mathbb{A} \in \mathbb{R}^{d\times d}$ needed to ensure that the function $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ defined as $k(x, y) = x^T \mathbb{A} y$ is a valid kernel :
Derivation of condition on $\mathbb{A}$:
Let $k(x, y) = x^T \mathbb{A} y$ be a valid kernel function.
$\implies$ $k(x, x) \geq 0$ (from the properties of kernel function). $\implies$ $x^T \mathbb{A} x \geq 0$
$\mathbb{A}$ is a Positive Semidefinite matrix.[From the quadratic form of a positive semi-definite matrix]

Also, From the symmetry property of kernel function:
$k(x, y) = k(y, x)$
$\implies$ $x^T \mathbb{A} y = y^T \mathbb{A} x$
Also, we know that $k(x, y) = \phi(x)^T \phi(y) = \phi(y)^T \phi(x) \implies (x^T \mathbb{A} y)^T = y^T \mathbb{A} x$
$\implies$ $(y\mathbb{A}^T x = y^T \mathbb{A} x$ [since, $(AB)^T = B^T A^T$]
$\implies$ $\mathbb{A}^T = \mathbb{A}$
$\mathbb{A}$ is a symmetric matrix.

Proving the other direction:
if $k(x, y) = x^T A y$ and A is symmetric and positive semi-definite, then k is a kernel function.
proof:
Since, $\mathbb{A}$ is symmetric and psd, we can write,
$\mathbb{A} = LDL^T$, where D is diagonal matrix containing non-negative eigen-values of A.L is an orthogonal matrix.
$\implies$ $\mathbb{A} = LD'D'L^T$, where $D' = \sqrt{D}$ is diagonal matrix formed by taking square root of eigen-values.
$\mathbb{A} = CC^T$, where C = LD'
$\implies$ $x^T \mathbb{A} y = x^T C\ C^T y$
$\implies$ $(x, y) = (C^T x)^T C^T y$
$\implies$ $(x, y) = \phi(x)^T \phi(y)$, where $\phi(w) = C^T w$
Hence, k is a valid kernel.
Therefore, $k(x, y) = x^T A y$ is a valid kernel, if and only if A is symmetric and

positive semi-definite.

proof:

b) If we form the kernel matrix for the 100 points, where $K(i,j) = k_l(x^{(i)}, x^{(j)})$, then the function $k_l : \mathbb{R}^3 \times \mathbb{R}^3 \to \mathbb{R}$ implemented in $k_l.pkl$ is likely to be a valid kernel, if K_l is **symmetric as well as positive semi-definite.**
For testing if K_l is symmetric or not, a function is written from scratch, and for testing positive semi-definite, a function is written which uses numpy.linalg.eigvalsh to calculate the eigen-values of the matrix K_l.

(i) The first matrix K_1 was symmetric but not positive semi-definite, hence function implemented in $k_1.pkl$ is **definitely not a kernel function**. *The smallest eigen value came out to be* -1035

(ii) The second matrix K_2 was symmetric and positive semi-definite(for 5 runs), hence function implemented in $k_1.pkl$ is **likely to be a kernel function**.

(iii) The third matrix K_3 was not a symmetric matrix, hence function implemented in $k_3.pkl$ is **definitely not a kernel function**.

(iv) The fourth matrix K_4 was symmetric but not positive semi-definite, hence function implemented in $k_4.pkl$ is **definitely not a kernel function**. *The smallest eigen value came out to be* -2

(v) The fifth matrix K_5 was symmetric but not positive semi-definite, hence function implemented in $k_5.pkl$ is **definitely not a kernel function**. *The smallest eigen value came out to be* -2.22

Question 2:

(a) Data was sampled using $numpy.random.uniform()$.Plot (ref. fig. 1) of the generated data using a scatter plot and the points colored based on the labels :

(b) The accuracy after training the SVM classifier(refer fig. 2), using cvxopt:

Train accuracy = 1.0
Test accuracy = 1.0
optimal C = 2.8869
$W = (-1.3812 \ -4.0043)^T$
$b = -4.3730$

(c) (i) The plot of the randomly sampled data is (refer fig. 3) :

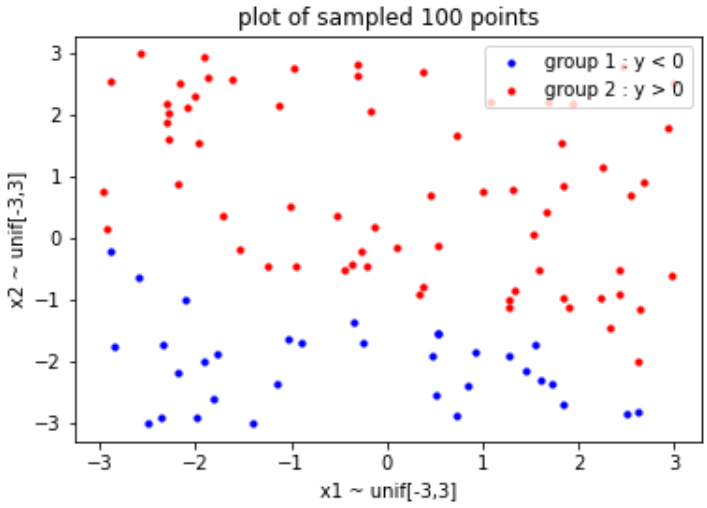


Figure 1: The plot of 100 sampled points, which are grouped to two classed according to $w^T x + b \ < \ 0 \ or \ > \ 0$

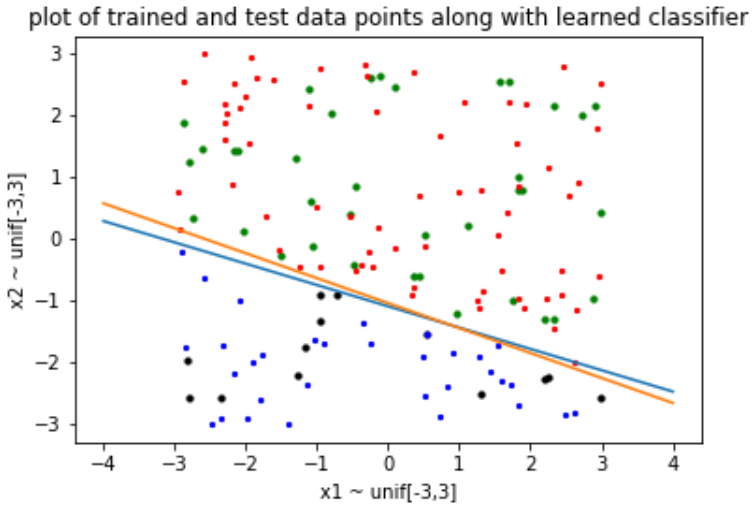
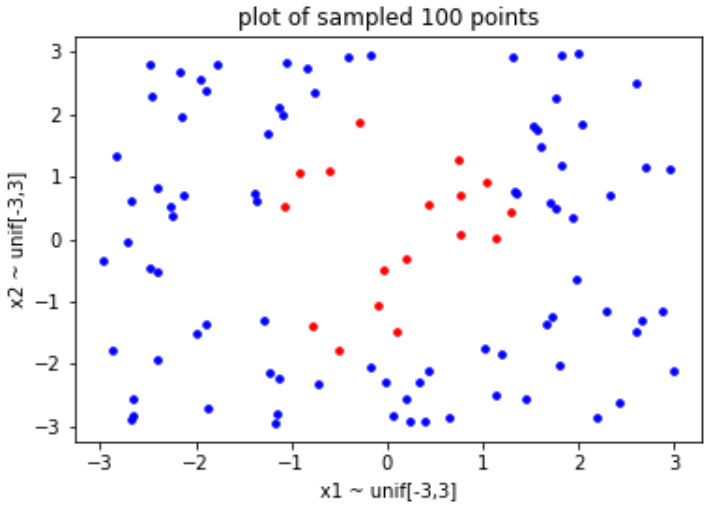


Figure 2: The plot of train (red,blue) and test data (green,black), with original hyperplane(orange) and the learned hyperplane(blue)

(ii) The accuracy after training the SVM classifier, using cvxopt, on the data generated in part (i):



)

Figure 3: The plot of the sampled points along with their class color. Red represents the case when $y = +1$ and blue when $y = -1$

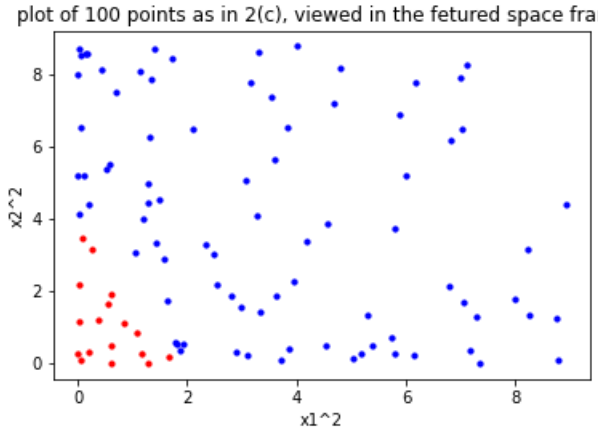
Accuracy on train data = 0.83 or 83%
Accuracy on test data = 0.72 or 72%
optimal C = 0.001
$W = (4.02248e-08\ 3.12828e-08)^T$
$b = -1.000$

(d) (i) The plot(refer fig 4) of the data as generated in 2(c),after applying the feature transformation is :

(ii) The accuracy after training the SVM classifier (refer fig. 5), using cvxopt, on the data mentioned in part (i):

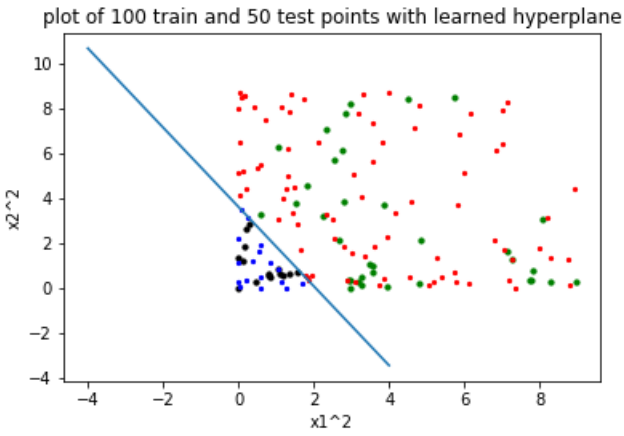
Accuracy on train data = 1.0 or 100%
Accuracy on test data = 1.0 or 100%
optimal C = 0.208
$W = (-1.35214\ -0.76402)^T$
$b = 2.76256$

(iii) Yes, the performance observed is much better. This is because, we used a $2-dimensional$ hyper-plane (a plane : $x_1^2 + x_2^2/2 = 2$) to classify the points. So, when we performed transformation, our axes became x_1^2 and x_2^2, so w.r.t this frame, the plane $x_1^2 + x_2^2/2 = 2$, becomes a



)

Figure 4: The plot of the sampled points after applying feature transformation. Here red points represent the class for which y = +1.0 and blue points represent the class for which y = -1.0



)

Figure 5: The plot of the trained(red and blue) and test(green and black) data, and the hyperplane learned using softmax SVM.

straight line, and the data points linearly seperable, as seen in figure (4) and (5)

(e) The equation of the dual optimization problem is :
$max_\alpha[\sum_{i=1}^{n} \alpha_i \ - \ \frac{1}{2} \sum_{i=1}^{n} \sum j = 1^n(\alpha_i \alpha_j y_i y_j x_i^T x_j)]$
subject to :

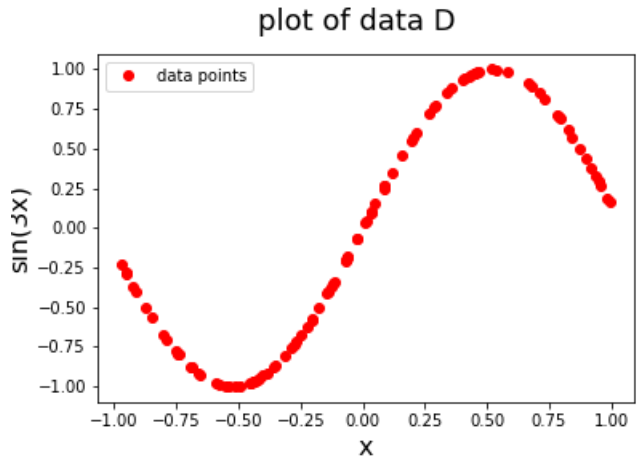
$0 \leq \alpha_i \leq$ ¸$\forall\ i \in [1..n]$
$\sum_{i=1}^{n} \alpha_i y_i = 0$
And the class of any query point is given by:
$y_q = sign(\sum_{i=1}^{n}(\alpha_i y_i x_i^T x_q))$
Here, if we replace $x_i^T x_j$ *by* some kernel function $k(x_i, x_j) = \phi(x_i)^T \phi(x_j)$, then our SVM becomes kernelized SVM. The value of learned α_i is 0 for non-support vector point and is > 0 for support vector points.

Expression of the Kernel function used : Gaussian Kernel
$k(x, y) = e^{\frac{\|\|x-y\|\|_2^2}{-2\sigma^2}}$

Here σ is the hyperparameter and the optimal value for it was found to be 0.01.
The learned value of C is 0.99
The accuracy on Training set data was observed to be 1.0 *or* 100%
The accuracy on test set data was observed to be 0.82 *or* 82%

Question 3:

(a) Plot (refer fig. 6) of the data points:

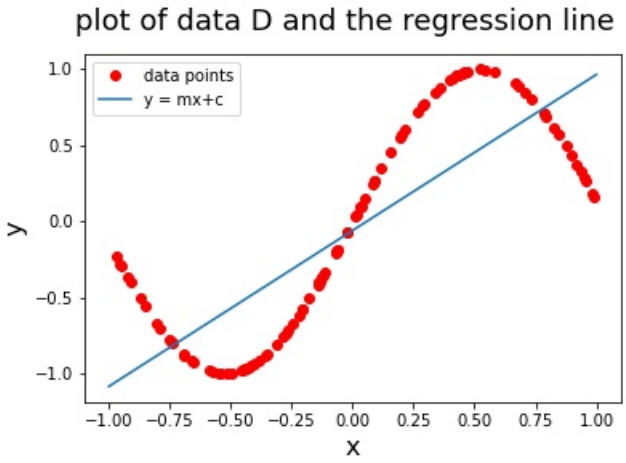


)

Figure 6: The plot of the sampled data points and corresponding $sin(3x)$ values.

(b) Mean square error observed was : 0.166195 or 16.6195% Plot(refer fig.7) of the line after performing Linear Regression:

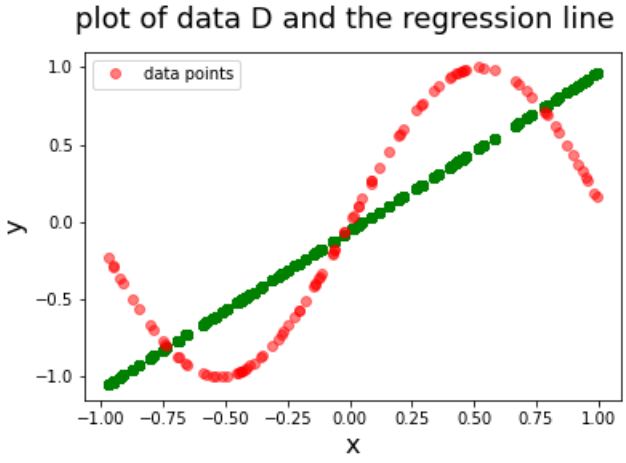
(c) Mean square error for different values of K(1 *to* 10) are:



)

Figure 7: The plot of the sampled data points and corresponding $sin(3x)$ values,along with the line learned after linear regression.

K = 1 ; Mean square error = 0.166195 (Same as performing linear regression)(refer fig. 8)

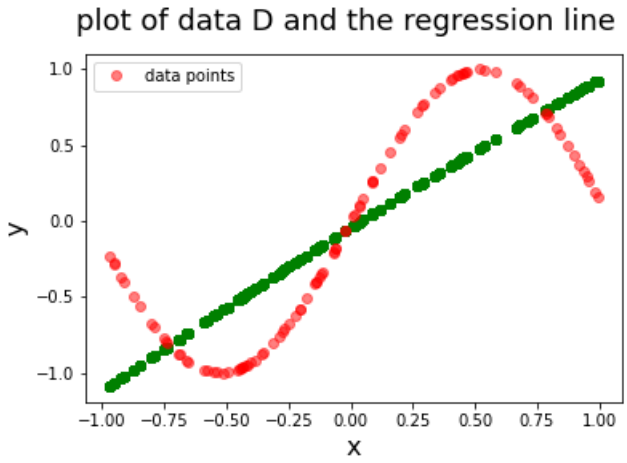


)

Figure 8: The plot of the sampled data points(red) and corresponding curve learnt(green) for k = 1.

K = 2 ; Mean square error = 0.165932 (Slightly started bending!)(refer fig. 9)

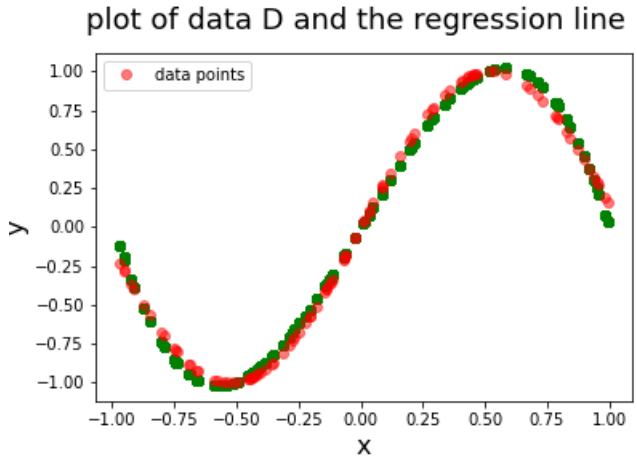
K = 3 ; Mean square error = 0.002691 (Took the rough shape of the curve!)(refer fig. 10)



)

Figure 9: The plot of the sampled data points(red) and corresponding curve learnt(green) for k = 2.

K = 4 ; Mean square error = 0.165932 (refer fig. 11)

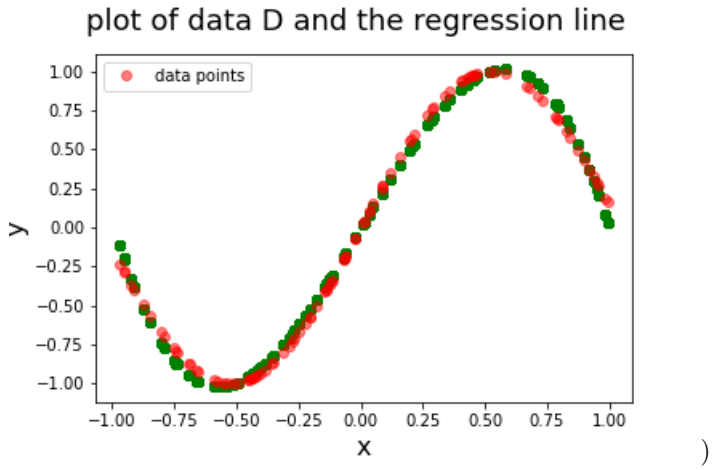


)

Figure 10: The plot of the sampled data points(red) and corresponding curve learnt(green) for k = 3.

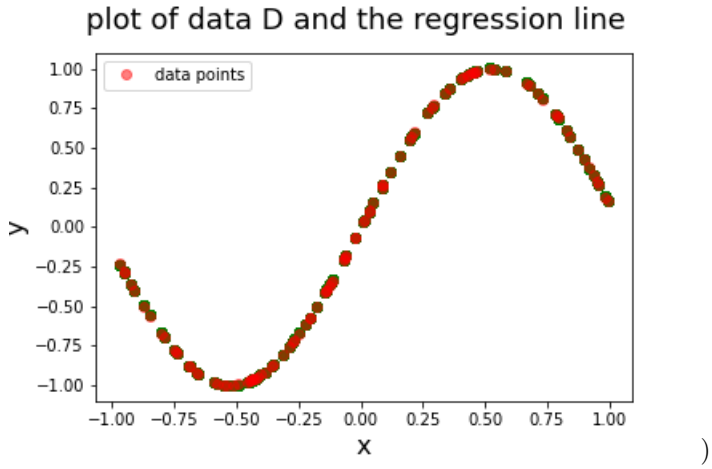
K = 5 ; Mean square error = $8.007575\text{e}^{-06}(Almost0msefromhereonwards)(refer fig.\textbf{??})$
$K = 6; Meansquareerror = 7.813379e^{-}06(refer fig.13)$

K = 7 ; Mean square error = 1.0135549e-08 (refer fig. 14)
K = 8 ; Mean square error = 9.574353e-09 (refer fig. 15)
K = 9 ; Mean square error = 3.925042e-12 (refer fig. 16)



)

Figure 11: The plot of the sampled data points(red) and corresponding curve learnt(green) for k = 4.



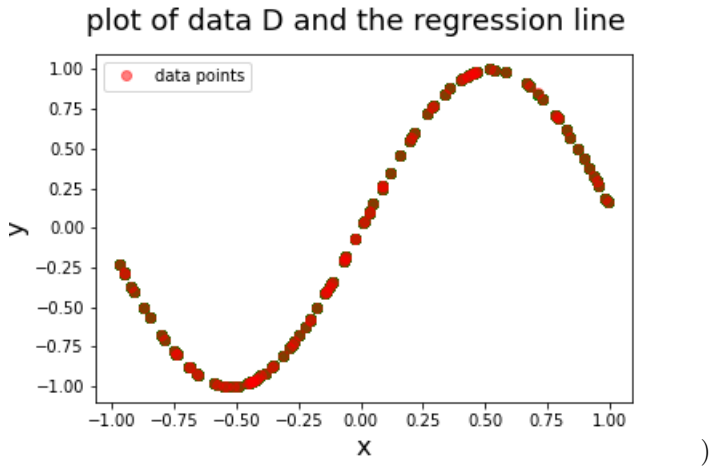
)

Figure 13: The plot of the sampled data points(red) and corresponding curve learnt(green) for k = 6.

K = 10 ; Mean square error = 3.653355e-12 (best fit)(refer fig. 17)

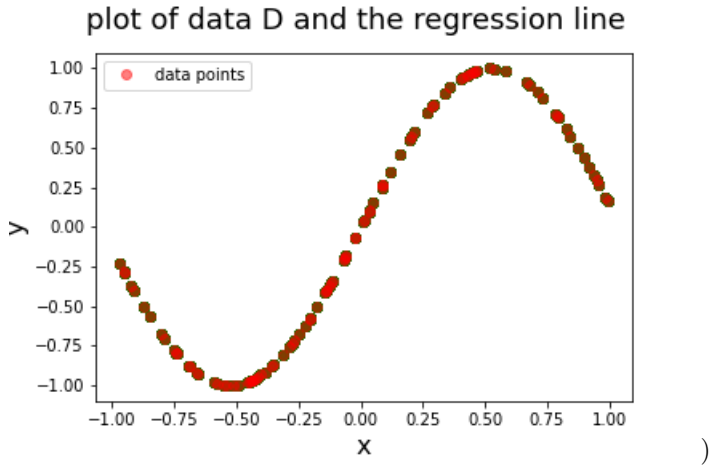
(d) (i) Kernels used for this part were : Gaussian kernel, polynomial kernel.

(ii) Training set Mean square error :4.7878181e-07 at $\sigma = 0.002$ *(hyperparameter for gaussian kernel)*



)

Figure 14: The plot of the sampled data points(red) and corresponding curve learnt(green) for k = 7.

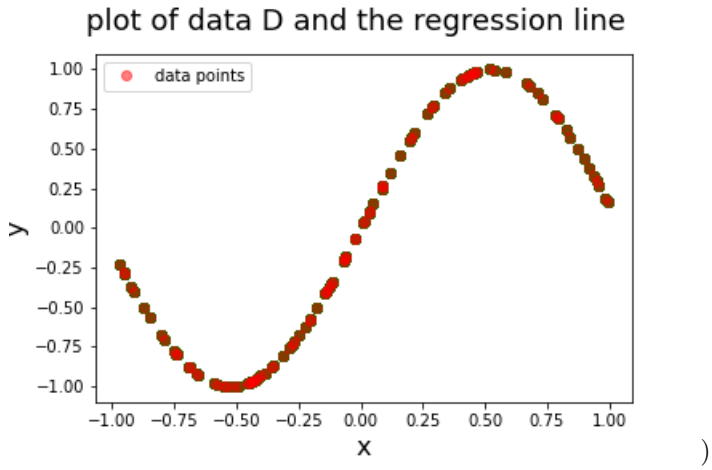


)

Figure 15: The plot of the sampled data points(red) and corresponding curve learnt(green) for k = 8.

Test set Mean square error : 0.0003416285

(iii) plot(refer fig. 18) of the learned function :



)

Figure 16: The plot of the sampled data points(red) corresponding curve learnt(green) for k = 9.

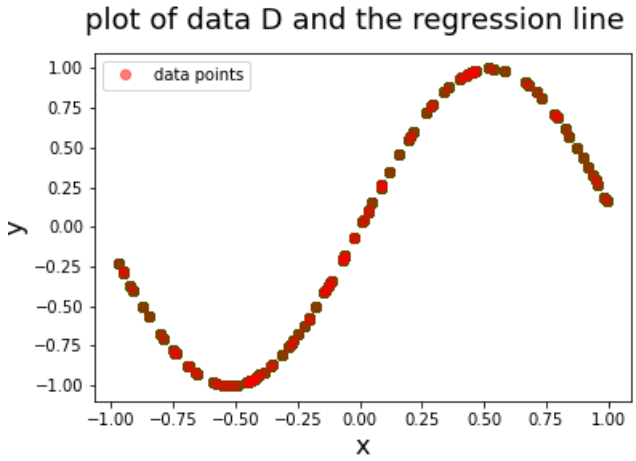


Figure 17: The plot of the sampled data points(red) and corresponding curve learnt(green) for k = 10.

Question 4:

(a) $\mathbb{D}_{10\times 10}$ *matrix* was observed to be Zero matrix and hence, sum of all entries of $\mathbb{D}$ was 0.

(b) The value of $\sum_{i=1}^{d} d_i$ for $d = 10$ was also observed to be Zero.(run code for output)

(c) For $k = 2$, $d = 2$, $n = 100$, the following clusters were observed(refer

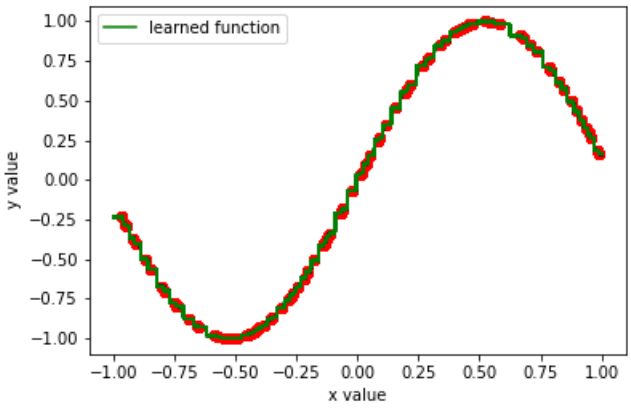


Figure 18: The data points(in red) and the learned curve(in green). This curve is highly overfitted and needs regularization

fig. 19):

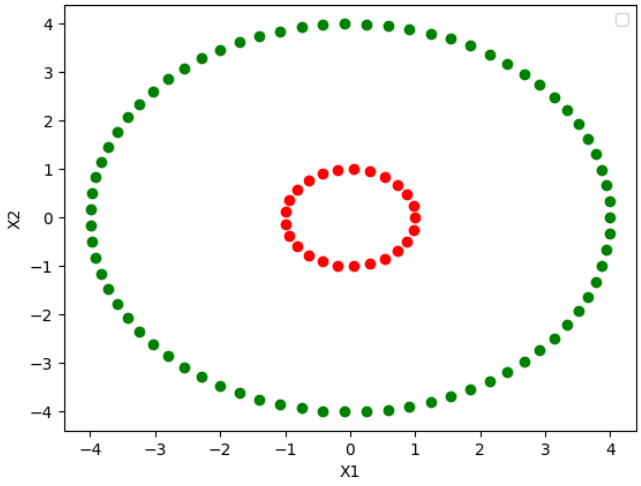


Figure 19: The data points(in red) form one cluster and the data points(in green) form the another cluster.

Question 5:

(a) The $1-D$ plot(refer fig.20) of the sampled data points.

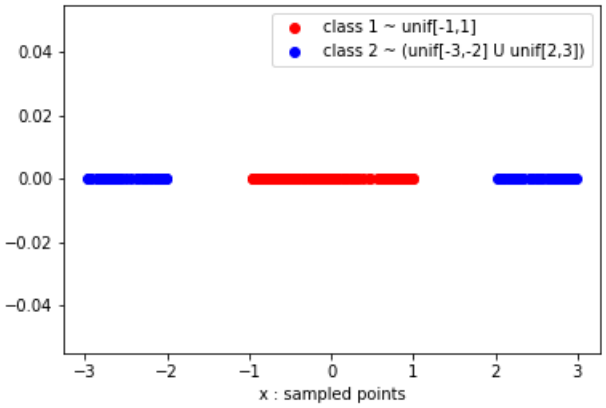


Figure 20: The $1-D$ plot of the sampled data points.

(b) Scatter plot (refer fig. 21) of the transformed data points according to $\phi : \mathbb{R} \to \mathbb{R}^2$.

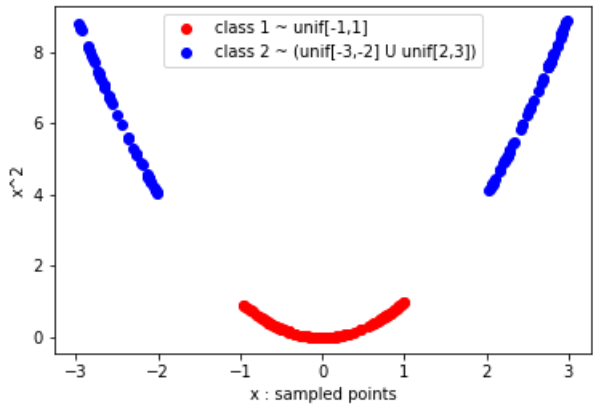


Figure 21: The transformed data points according to ϕ.

(c) After applying Fisher's Linear Discriminant Analysis on the data given in $\tilde{C}_1$ and $\tilde{C}_2$, the following direction of W was observed (refer fig. 22).

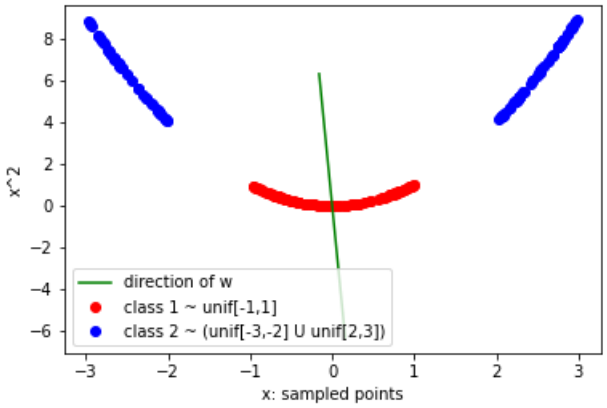


Figure 22: The direction of w is given by green line.

(d) *Note: code of this part is in separate notebook, and hence data was sampled again.*
This part was done with both Gaussian kernel and polynomial kernel, and the plot was as observed(refer fig. 25 24). If we kept on increasing σ for Gaussian Kernel, the accuracy remained same as 100%, but the separation between the two classes kept on decreasing(refer fig **??** 26). Hence σ_opt was chosen to be 1.

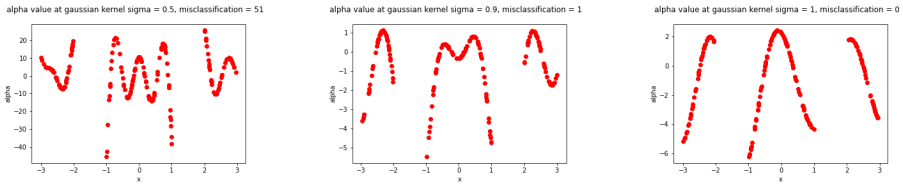


Figure 23: α *vs.* x for different $\sigma = 0.5, 0.9, 1.0$ of Gaussian Kernel

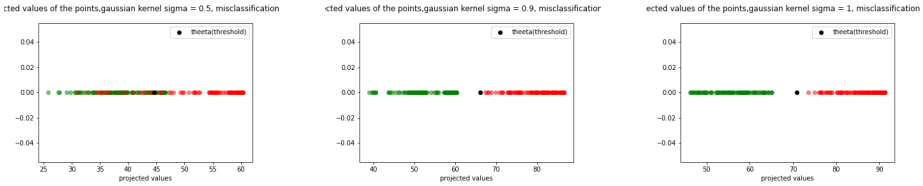


Figure 24: The projection of the points in $1 - Dimension$, calculated using $y(x) = \sum_{i=1}^{n} \alpha_i k(x_i, x)$, for different values of $sigma = 0.5, 0.9, 1.0$. When $sigma = 0.5$,miss-classifications $= 51$, and hence accuracy $= 74.5\%$
When $sigma = 0.9$,miss-classifications $= 1$, and hence accuracy $= 99.5\%$
When $sigma = 1.0$,miss-classifications $= 0$, and hence accuracy $= 100\%$

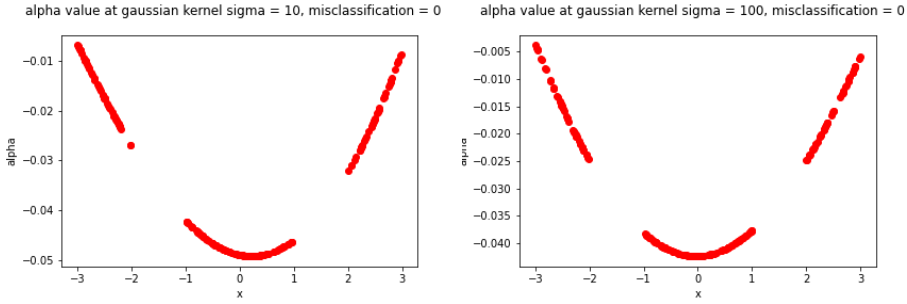


Figure 25: α $vs.$ x for different $\sigma = 10, 100$ of Gaussian Kernel

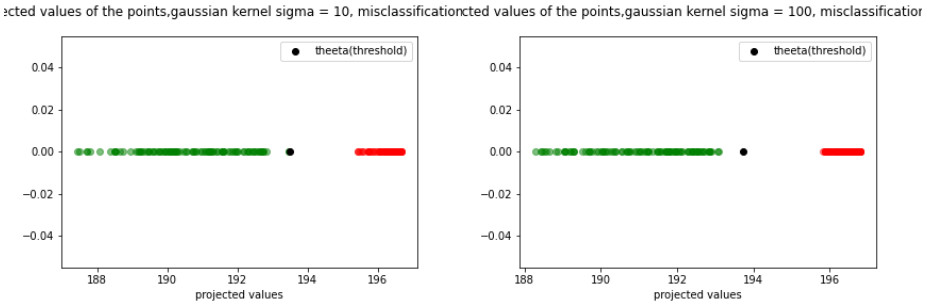


Figure 26: The projection of the points in $1 - Dimension$, calculated using $y(x) = \sum_{i=1}^{n} \alpha_i k(x_i, x)$, for different values of $sigma = 10, 100$. Although accuracy is 100% in both the cases, distance between the classes decreases as σ increases.