Introduction to ML (CS771), Autumn 2018
Indian Institute of Technology Kanpur
Homework Assignment Number 3

Student Name: Sahil Bansal
Roll Number: 150614
Date: November 2, 2018

QUESTION 1

We are given the soft-margin linear SVM problem as

$$\underset{0\leq\boldsymbol{\alpha}\leq C}{\operatorname{argmax}} f(\boldsymbol{\alpha})$$

where $f(\boldsymbol{\alpha}) = \boldsymbol{\alpha}^T - \frac{1}{2}\boldsymbol{\alpha}^T\mathbf{G}\boldsymbol{\alpha}$
Our goal is to derive a co-ordinate ascent algorithm for the vector $\boldsymbol{\alpha}$. We will update it as $\boldsymbol{\alpha}_n = \boldsymbol{\alpha}_n + \delta_*$ where δ_* is given as

$$\delta_* = \underset{\delta}{\operatorname{argmax}} f(\boldsymbol{\alpha} + \delta\mathbf{e}_n)$$

Therefore, We have to solve this maximization problem in order to get δ_*

$$f(\boldsymbol{\alpha} + \delta\mathbf{e}_n) = (\boldsymbol{\alpha} + \delta\mathbf{e}_n)^T 1 - \frac{1}{2}(\boldsymbol{\alpha} + \delta\mathbf{e}_n)^T\mathbf{G}(\alpha + \delta\mathbf{e}_n)$$

Simplifying,

$$f(\boldsymbol{\alpha} + \delta\mathbf{e}_n) = \boldsymbol{\alpha}^T 1 + \delta\mathbf{e}_n^T 1 - \frac{1}{2}\boldsymbol{\alpha}^T\mathbf{G}\boldsymbol{\alpha} - \delta\mathbf{e}_n^T\mathbf{G}\boldsymbol{\alpha} - \frac{1}{2}\delta^2\mathbf{e}_n^T\mathbf{G}\mathbf{e}_n$$

Derivate w.r.t to δ, we get

$$\mathbf{e}_n^T 1 - \mathbf{e}_n^T\mathbf{G}\boldsymbol{\alpha} - \delta\mathbf{e}_n^T\mathbf{G}\mathbf{e}_n$$

To get the optimal value of δ i.e. δ_*, set it to 0

$$\delta_* = \frac{\mathbf{e}_n^T 1 - \mathbf{e}_n^T\mathbf{G}\boldsymbol{\alpha}}{\mathbf{e}_n^T\mathbf{G}\mathbf{e}_n}$$

Now, $\mathbf{e}_n$ denotes a vector of all zeros except a 1 at entry n, therefore, $\mathbf{e}_n^T 1 = 1$. So, the above equation reduces to

$$\delta_* = \frac{1 - \mathbf{e}_n^T\mathbf{G}\boldsymbol{\alpha}}{\mathbf{e}_n^T\mathbf{G}\mathbf{e}_n}$$

Also, the constraint $0 \leq \boldsymbol{\alpha}_n \leq C$ needs to be satisfied at all points in time. Therefore, following cases hold

- If $\boldsymbol{\alpha}_n^{(t-1)} + \delta_* < 0$, then $\boldsymbol{\alpha}_n^{(t)} = 0$
- If $\boldsymbol{\alpha}_n^{(t-1)} + \delta_* > C$, then $\boldsymbol{\alpha}_n^{(t)} = C$
- Else $\boldsymbol{\alpha}_n^{(t)} = \boldsymbol{\alpha}_n^{(t-1)} + \delta_*$

Sketch of overall co-ordinate ascent algorithm:
1. Choose a uniformly random entry $\boldsymbol{\alpha}_n$ of the vector $\boldsymbol{\alpha}$
2. Update it according to procedure discussed above i.e. $\boldsymbol{\alpha}_n^{(t)} = \boldsymbol{\alpha}_n^{(t-1)} + \delta_*$ with all the constraints met at all times.
3. Repeat the above steps until convergence.

Introduction to ML (CS771), Autumn 2018
Indian Institute of Technology Kanpur
Homework Assignment Number 3

Student Name: Sahil Bansal
Roll Number: 150614
Date: November 2, 2018

QUESTION 2

Function f is such that $f_n = f(x_n)$ is the cluster assignment for point x_n.
$\mathcal{L}_w$ is defined as the sum of squared distances between all pairs of points that are within the same cluster, i.e.,

$$\underset{f}{\text{argmin}}\mathcal{L}_w = \underset{f}{\text{argmin}} \sum_{n,m} \mathbb{I}[f_n = f_m]||x_n - x_m||^2$$

Now, sum of the squared distances between all the points is a constant value, say c. Therefore,

$$||x_n - x_m||^2 = \sum_{n,m} \mathbb{I}[f_n = f_m]||x_n - x_m||^2 + \sum_{n,m} \mathbb{I}[f_n \neq f_m]||x_n - x_m||^2 = c$$

The second term in the above summation corresponds to sum of squared distances between all pairs of points that are in different clusters. Now, as the sum of two summations is a constant, therefore, maximizing one will minimize the other one or vice-versa. Hence, minimizing the sum of squared distances between all pairs of points that are within the same cluster is equivalent to maximizing sum of squared distances between all pairs of points that are within the different clusters.

Introduction to ML (CS771), Autumn 2018
Indian Institute of Technology Kanpur
Homework Assignment Number 3

Student Name: Sahil Bansal
Roll Number: 150614
Date: November 2, 2018

QUESTION 3

We are given N observations $\mathbf{x}_i$ where $i \in \{1, 2, .., n\}$ where there are two parts to each $\mathbf{x}_n$ i.e. $\mathbf{x}_n = [\mathbf{x}_n^o, \mathbf{x}_n^m]$. Here, $\mathbf{x}_n^o$ corresponds to the observed part and $\mathbf{x}_n^m$ to the missing part. We have to find the mean and covariance estimates for $\mathbf{x}_n$. Let μ_n and Σ_n have the following form:

$$\mu_n = \begin{bmatrix} \mu_n^o \\ \mu_n^m \end{bmatrix}$$

$$\Sigma_n = \begin{bmatrix} \Sigma_n^{oo} & \Sigma_n^{om} \\ \Sigma_n^{mo} & \Sigma_n^{mm} \end{bmatrix}$$

Expression for $p(\mathbf{x}^m|\mathbf{x}^o, \mu, \Sigma)$:

$$p(\mathbf{x}_n^m|\mathbf{x}_n^o, \mu, \Sigma) = \mathcal{N}(\mathbf{x}_n^m|\mathbf{x}_n^o, \mu_n, \Sigma_n)$$

which is equal to

$$\mathcal{N}(\mathbf{x}_n^m|\mu_n^{m|o}, \Sigma_n^{m|o})$$

Here, $\mu_n^{m|o} = \mu_n^m + \Sigma_n^{mo}(\Sigma_n^{oo})^{-1}(x_n^o - \mu_n^o)$ and $\Sigma_n^{m|o} = \Sigma_n^{mm} - \Sigma_n^{mo}(\Sigma_n^{oo})^{-1}\Sigma_n^{om}$

Expected CLL:
Complete data log-likelihood is given as:

$$\sum_{n=1}^{N} \log p(\mathbf{x}_n|\mu, \Sigma) = -\frac{N}{2}\log\Sigma - \frac{1}{2}\sum_{n=1}^{N}(\mathbf{x}_n - \mu)^T\Sigma^{-1}(\mathbf{x}_n - \mu)$$

Taking expectation

$$-\frac{N}{2}\log\Sigma - \frac{1}{2}\sum_{n=1}^{N} E\big[(\mathbf{x}_n - \mu)^T\Sigma^{-1}(\mathbf{x}_n - \mu)\big]$$

which is equivalent to

$$-\frac{N}{2}\log\Sigma - \frac{1}{2}\sum_{n=1}^{N}\Big(E[\mathbf{x}_n]^T\Sigma^{-1}E[\mathbf{x}_n] - E[\mathbf{x}_n]^T\Sigma^{-1}\mu - \mu^T\Sigma^{-1}E[\mathbf{x}_n] + \mu^T\Sigma^{-1}\mu\Big)$$

Update equations for μ and Σ:
Derivate above equation w.r.t to μ

$$-\frac{1}{2}\sum_{n=1}^{N}(-2\Sigma^{-1}E[\mathbf{x}_n] + 2\Sigma^{-1}\mu)$$

Equating it to 0, we get

$$\mu = \frac{1}{N}\sum_{n=1}^{N} E[\mathbf{x}_n]$$

The above expression for expected CLL can also be written as:

$$-\frac{N}{2}\log\Sigma - \frac{1}{2}\sum_{n=1}^{N} Tr\{\Sigma^{-1}E[(x_n-\mu)(x_n-\mu)^T]\}$$

Derivating above equation w.r.t to Σ^{-1}, we get

$$\Sigma = \frac{1}{N}\sum_{n=1}^{N} E[(x_n-\mu)(x_n-\mu)^T]$$

EM Algorithm:
1. Initialize $\theta = (\mu, \Sigma)$ as $\theta^{(0)}$ and set t = 1
2. E step :
Calculate $(\mu_n^{m|o})^{(t-1)} = (\mu_n^m)^{(t-1)} + (\Sigma_n^{mo})^{(t-1)}((\Sigma_n^{oo})^{-1})^{(t-1)}(x_n^o - (\mu_n^o)^{(t-1)})$ and $(\Sigma_n^{m|o})^{(t-1)} = (\Sigma_n^{mm})^{(t-1)} - (\Sigma_n^{mo})^{(t-1)}((\Sigma_n^{oo})^{-1})^{(t-1)}(\Sigma_n^{om})^{(t-1)}$
3. M step :

$$\mu^{(t)} = \frac{1}{N}\sum_{n=1}^{N} E[\mathbf{x}_n]$$

$$\Sigma^{(t)} = \frac{1}{N}\sum_{n=1}^{N} E[(x_n-\mu^{(t)})(x_n-\mu^{(t)})^T]$$

Here,

$$E[\mathbf{x}_n] = \begin{bmatrix} \mathbf{x}_n^o \\ E[\mathbf{x}_n^m] \end{bmatrix} = \begin{bmatrix} \mathbf{x}_n^o \\ \mu_n^{m|o^{(t-1)}} \end{bmatrix}$$

and,

$$E[\mathbf{x}_n\mathbf{x}_n^T] = \begin{bmatrix} \mathbf{x}_n^o\mathbf{x}_n^{oT} & \mathbf{x}_n^o\mu_n^{m|o^{(t-1)^T}} \\ \mu_n^{m|o^{(t-1)}}\mathbf{x}_n^o & \Sigma_n^{m|o^{(t-1)}} \end{bmatrix}$$

4. Set t = t + 1 and go to step 2 if not yet converged

Introduction to ML (CS771), Autumn 2018
Indian Institute of Technology Kanpur
Homework Assignment Number 3

Student Name: Sahil Bansal
Roll Number: 150614
Date: November 2, 2018

QUESTION 4

We have N labeled examples along with additional M unlabeled examples, therefore, N+M examples in total.
While discussing MLE for Generative classification with gaussian class conditionals (in case of unlabeled examples), we have seen in class the following equation:

$$\log p(\mathbf{X}, \mathbf{Z}|\theta) = \sum_{n=1}^{N+M} \sum_{k=1}^{K} z_{nk} \big[\log(\pi_k) + \log(\mathcal{N}(\mathbf{x}|\mu_k, \Sigma_k) \big]$$

Now, expected complete data log-likelihood is defined as follows,

$$\mathcal{Q}(\theta, \theta^{old}) = E_{p(\mathbf{Z}|\mathbf{X}, \theta^{old})}[\log p(\mathbf{X}, \mathbf{Z}|\theta)]$$

which is equivalent to

$$\mathcal{Q}(\theta, \theta^{old}) = \sum_{n=1}^{N+M} \sum_{k=1}^{K} E[\mathbf{z}_{nk}] \big[\log(\pi_k) + \log(\mathcal{N}(\mathbf{x}|\mu_k, \Sigma_k) \big]$$

where again the expectation is w.r.t to the $p(\mathbf{Z}|\mathbf{X}, \theta^{old})$ distribution.
Now, for the first N training examples, $E[\mathbf{z}_{nk}] = 1$ if $\mathbf{y}_n = k$ else 0 and for the remaining M examples $E[\mathbf{z}_{nk}]$ is defined again the same way as discussed in the class, i.e.,

$$E[\mathbf{z}_{nk}] = \frac{\pi_k \mathcal{N}(\mathbf{x}|\mu_k, \Sigma_k)}{\sum_{l=1}^{K} \pi_l \mathcal{N}(\mathbf{x}|\mu_l, \Sigma_l)}$$

EM Algorithm:

1. Initialize $\theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^{K}$ as $\theta^{(0)}$ and set $t = 1$
2. E step: compute the expectation of each $\mathbf{z}_n$ (we need it in M step)

- For $n \in \{1, N\}$, $E[\mathbf{z}_{nk}] = 1$ if $\mathbf{y}_n = k$ else 0
- For $n \in \{N+1, N+M\}$, $E[\mathbf{z}_{nk}^{(t)}] = \gamma_{nk}^{(t)} = \frac{\pi_k^{(t-1)} \mathcal{N}(\mathbf{x}|\mu_k^{(t-1)}, \Sigma_k^{(t-1)})}{\sum_{l=1}^{K} \pi_l^{(t-1)} \mathcal{N}(\mathbf{x}|\mu_l^{(t-1)}, \Sigma_l^{(t-1)})}$

3. Given responsibilities $\gamma_{nk} = E[\mathbf{z}_{nk}]$, and $N_k = \sum_{n=1}^{N+M} \gamma_{nk}$, re-estimate θ via MLE

$$\mu_k^{(t)} = \frac{\sum_{n=1}^{N+M} \gamma_{nk}^{(t)} x_n}{N_k}$$

$$\Sigma_k^{(t)} = \frac{\sum_{n=1}^{N+M} \gamma_{nk}^{(t)} (x_n - \mu_k^{(t)})(x_n - \mu_k^{(t)})^T}{N_k}$$

$$\pi_k^{(t)} = \frac{N_k}{N+M}$$

4. Set $t = t + 1$ and go to step 2 if not yet converged

Introduction to ML (CS771), Autumn 2018
Indian Institute of Technology Kanpur
Homework Assignment Number 3

Student Name: Sahil Bansal
Roll Number: 150614
Date: November 2, 2018

QUESTION 5

1. Brief Explanation :

The given model is learning K separate linear regressors instead of a single linear regressor as in the case of standard probabilistic linear model. The standard probabilistic model works well only in case the data is linear but the given model would work better even in case when the data is linear in K different parts.

2. Alt-Opt Algorithm :

$$\textit{Likelihood} \qquad \mathcal{L}(Z,\theta) = \sum_{n=1}^{N} \log p(y_n, z_n | \mathbf{x}_n, \theta)$$

$$\mathcal{L}(Z,\theta) = \sum_{n=1}^{N} \Big(\log \mathcal{N}(y_n | \mathbf{w}_{z_n}^T \mathbf{x}_n, \beta^{-1}) + \log p(z_n|\theta) \Big)$$

$$\mathcal{L}(Z,\theta) = \sum_{n=1}^{N} \sum_{k=1}^{K} z_{nk} \Big(\log \pi_k - \frac{\beta}{2}(y_n - \mathbf{w}_k^T \mathbf{x}_n)^2 \Big)$$

MLE estimates for π_k and $\mathbf{w}_k$:

$$\pi_k = \frac{\sum_{n=1}^{N} z_{nk}}{N}$$

$$\mathbf{w}_k = \Big(\sum_{n=1}^{N} z_{nk} \mathbf{x}_n \mathbf{x}_n^T \Big)^{-1} \Big(\sum_{n=1}^{N} z_{nk} \mathbf{x}_n y_n^T \Big)$$

Alt-Opt Steps:

- Initialize $\theta = \{\pi_k, \mathbf{w}_k\}_{k=1}^K$ as $\theta^{(0)}$ and set $t = 1$
- Calculate z_n as follows:

$$z_n^{(t)} = \arg\max_k \pi_k^{(t-1)} \log \mathcal{N}(y_n | \mathbf{w}_k^{(t-1)^T} \mathbf{x}_n, \beta^{-1})$$

- Update step:

$$\pi_k^{(t)} = \frac{\sum_{n=1}^{N} z_{nk}^{(t)}}{N}$$

$$\mathbf{w}_k^{(t)} = \Big(\sum_{n=1}^{N} z_{nk}^{(t)} \mathbf{x}_n \mathbf{x}_n^T \Big)^{-1} \Big(\sum_{n=1}^{N} z_{nk}^{(t)} \mathbf{x}_n y_n^T \Big)$$

- Set $t = t + 1$ and go to step 2 if not yet converged

Case when $\pi_k = \frac{1}{K}$:

In this case, z_n can be expressed as:

$$z_n = \arg\max_k \log \mathcal{N}(y_n | \mathbf{w}_k^T \mathbf{x}_n, \beta^{-1})$$

$$z_n = \arg\min_k (y_n - \mathbf{w}_k^T \mathbf{x}_n)^2$$

Here, simply assigning that cluster to the point for which the distance is minimum between predicted output and true output.

Student Name: Sahil Bansal
Roll Number: 150614
Date: November 2, 2018

QUESTION
6

Programming Problem 1:
1. Kernel Ridge Regression

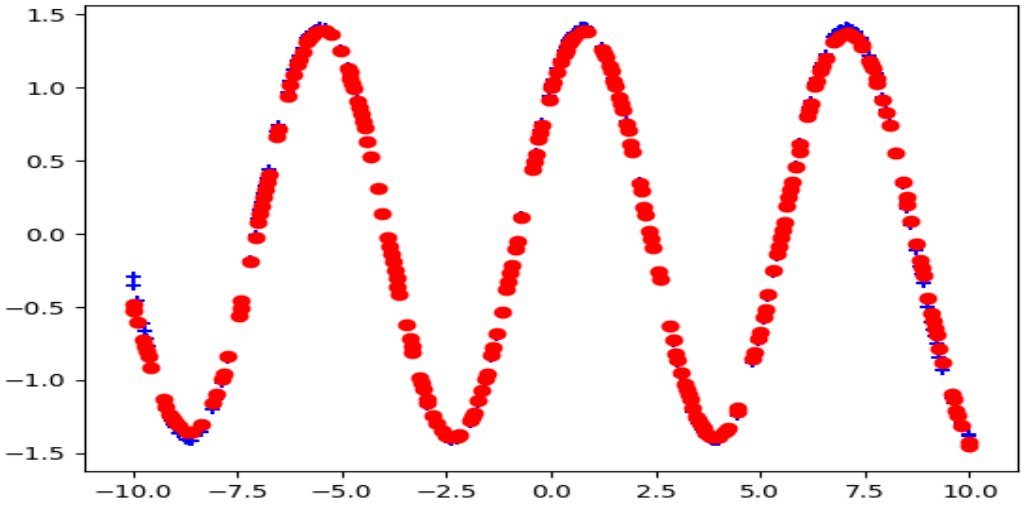


Figure 1: $\lambda = 0.1$

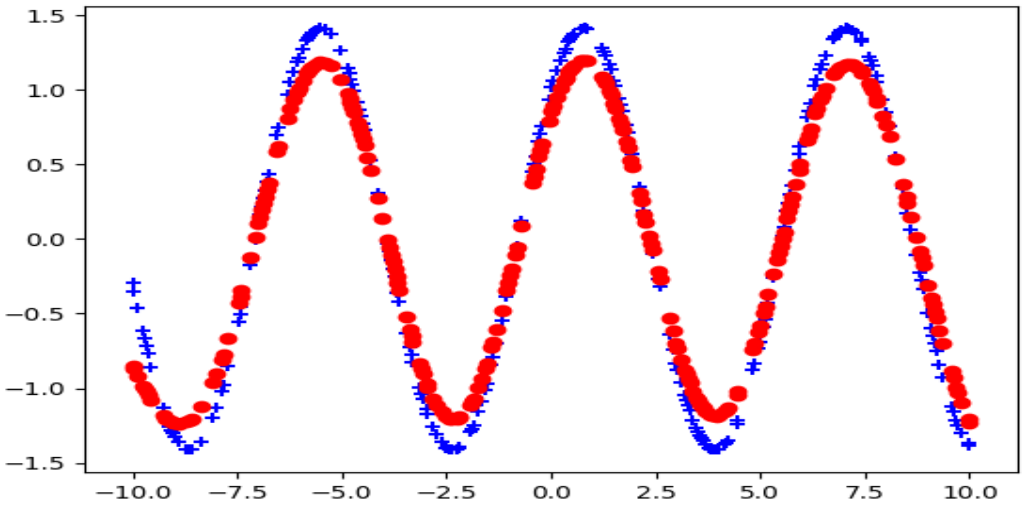


Figure 2: $\lambda = 1.0$

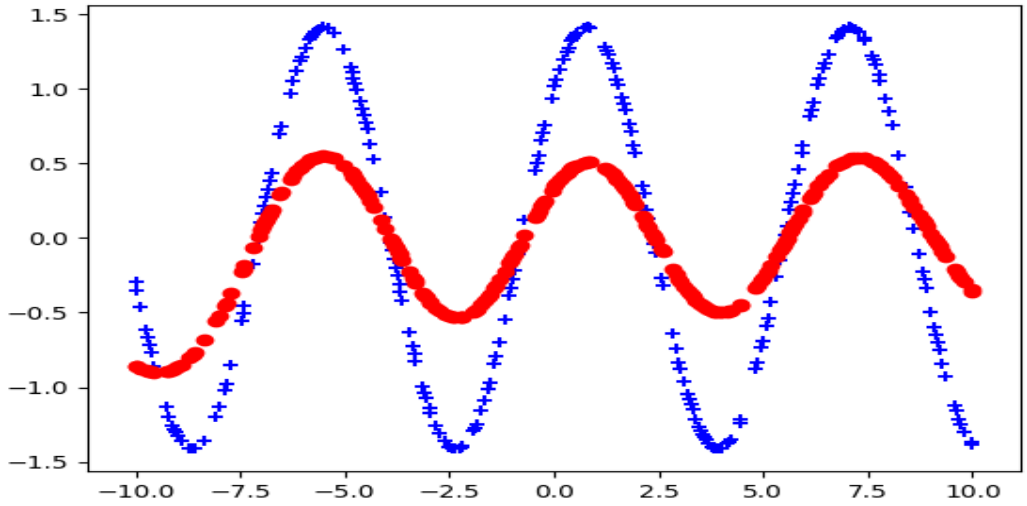


Figure 3: $\lambda = 10.0$

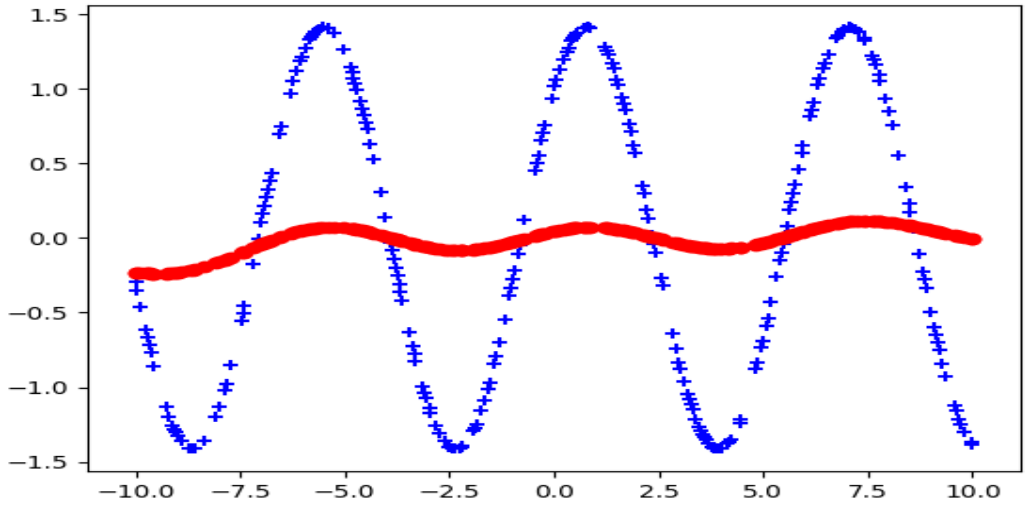


Figure 4: $\lambda = 100.0$

λ	RMSE
0.1	0.03257767029357752
1.0	0.17030390344202498
10.0	0.6092671596540067
100.0	0.9110858052767243

Observation:

As the value of λ increases, the RMSE value is also increasing. This is precisely because as the value of λ increases, the model is underfitting.

2. Landmark Ridge Regression

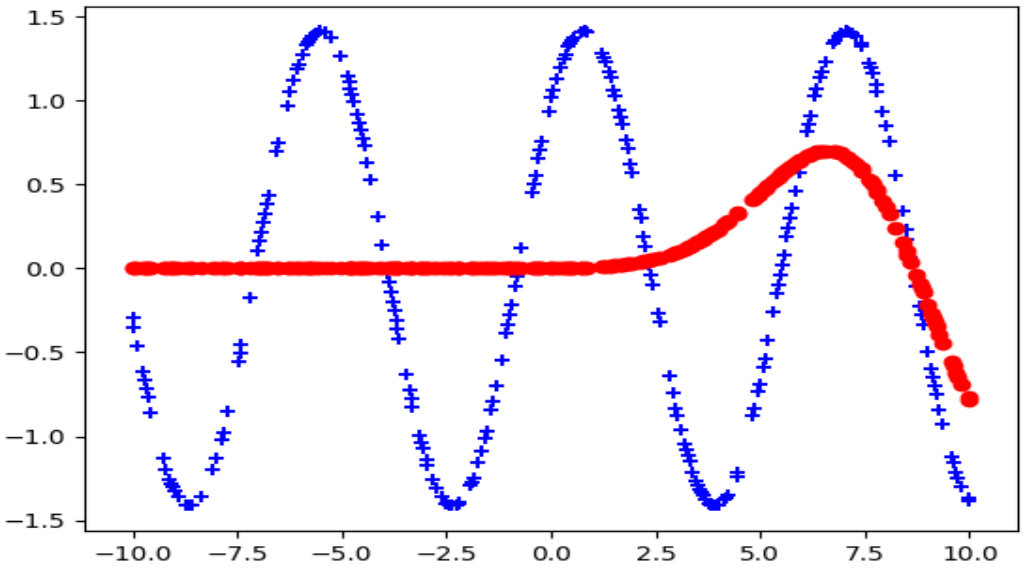


Figure 5: L = 2

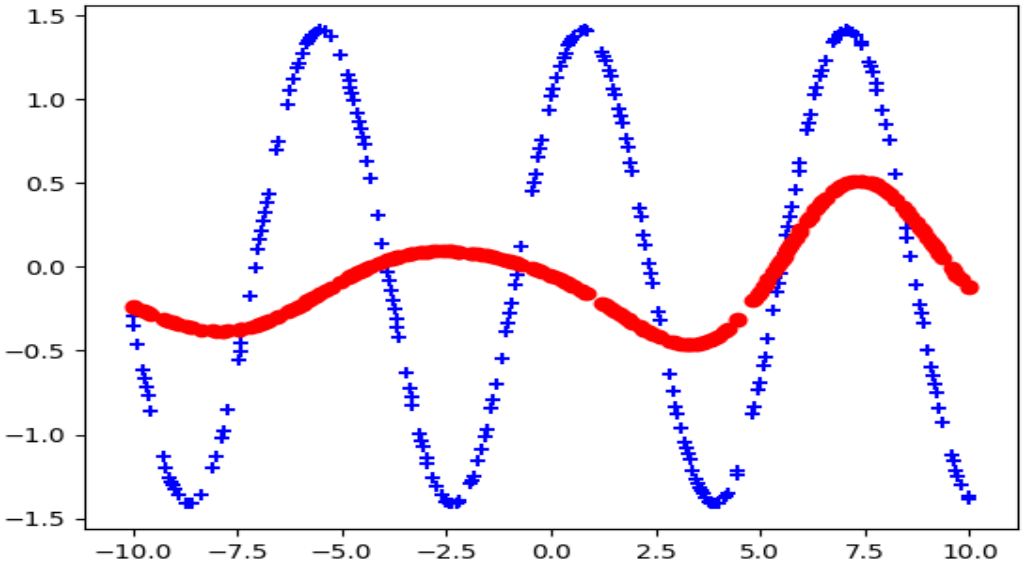


Figure 6: L = 5

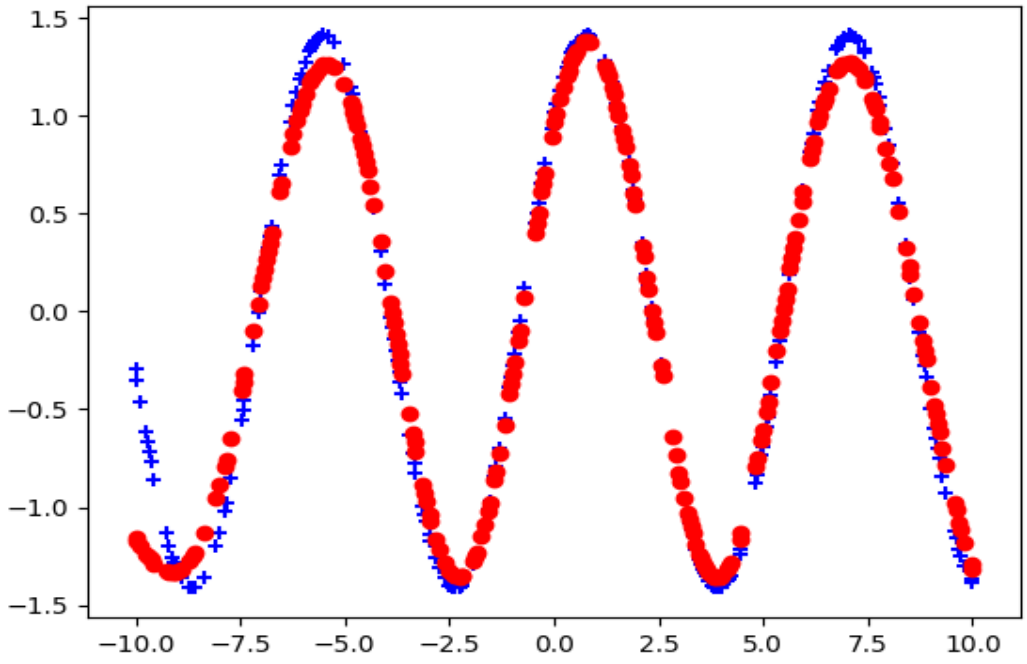


Figure 7: L = 20

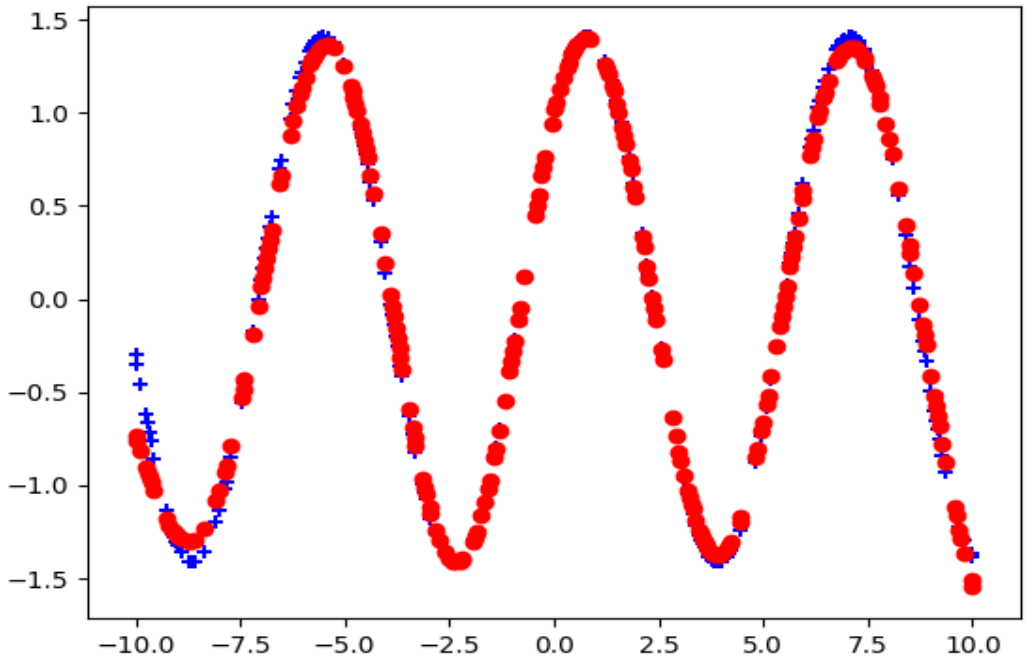


Figure 8: L = 50

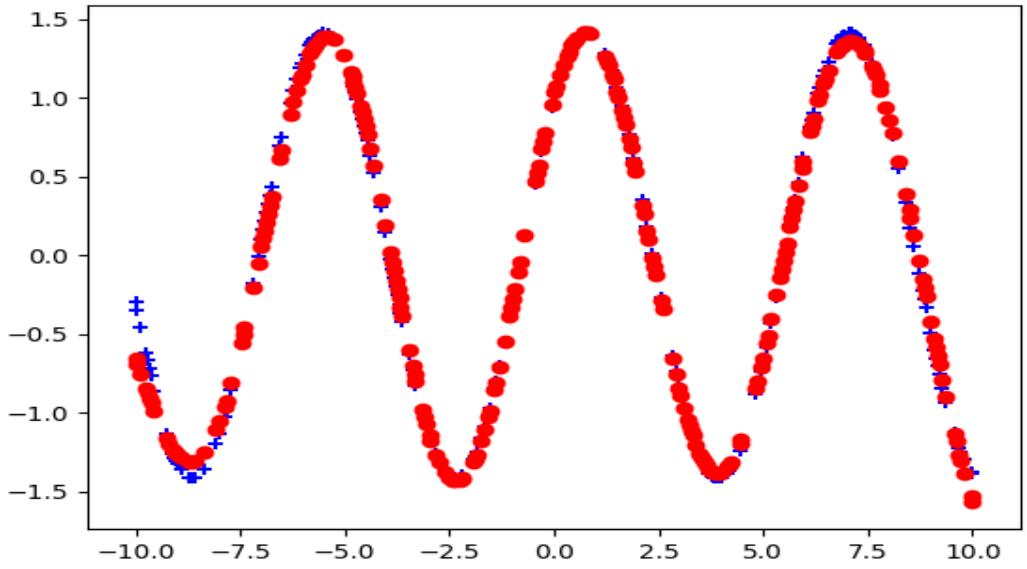


Figure 9: L = 100

L	RMSE
2	0.9345757225022168
5	0.9154503532994157
20	0.14539722761311874
50	0.07137027860665845
100	0.0610035661909464

Observation:

We can clearly see that $L = 100$ gives the best results i.e. RMSE values observed is lowest for $L = 100$.Further, it can be seen that as the number of landmark points increase, correspondingly the RMSE value also decreases. Better results are observed with more number of landmark points.

Programming Problem 2:
1. Using Hand-crafted Features

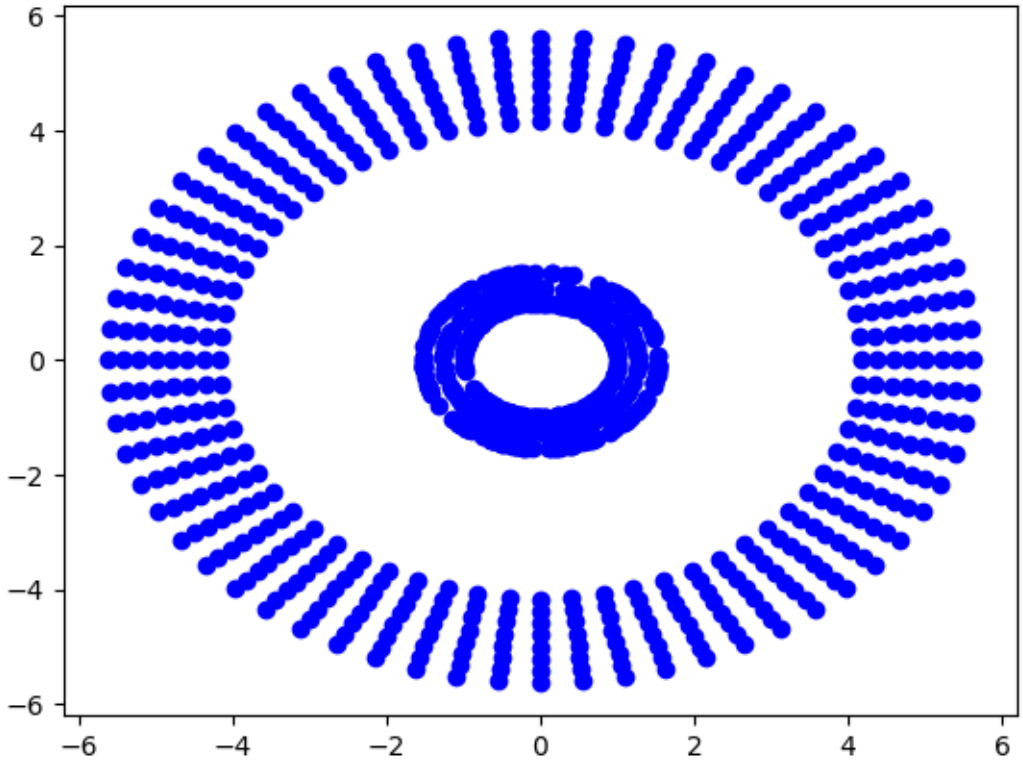


Figure 10: Original Data

Proposed transformation :

$$\Phi(x_1, x_2) = (x_1, x_2, x_1^2 + x_2^2)$$

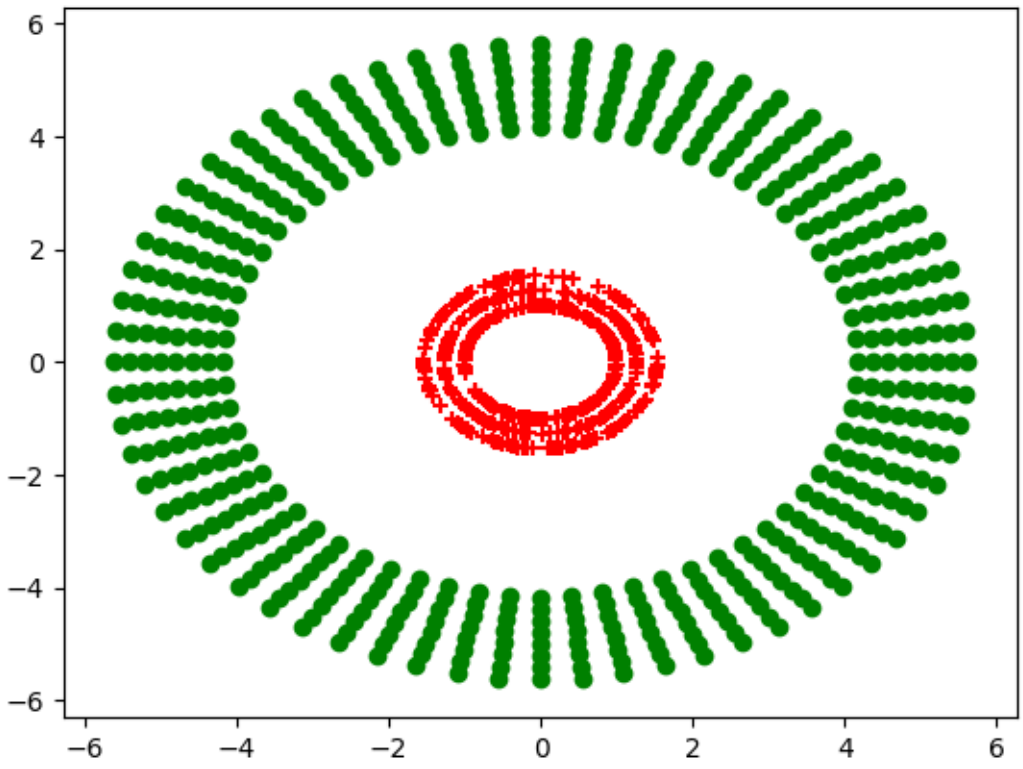
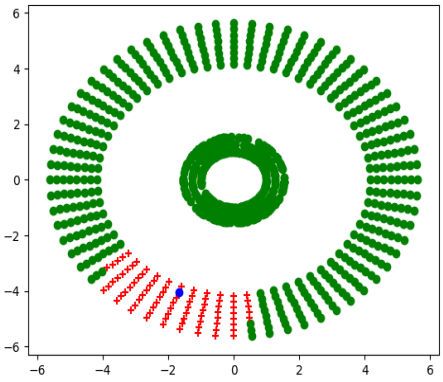
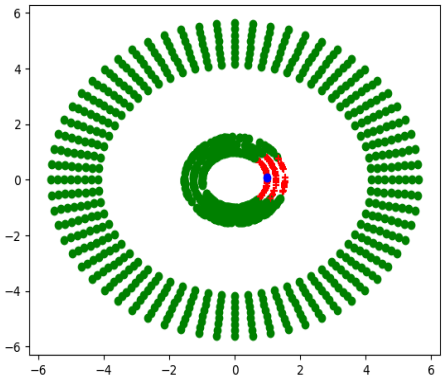
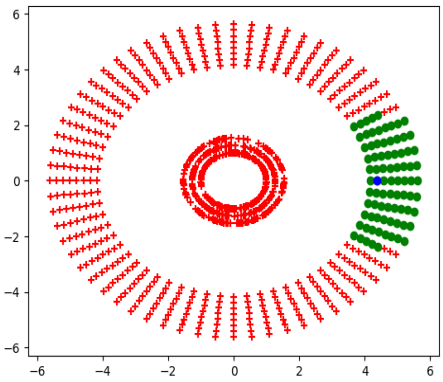
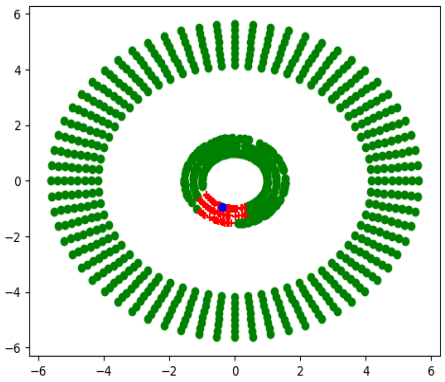
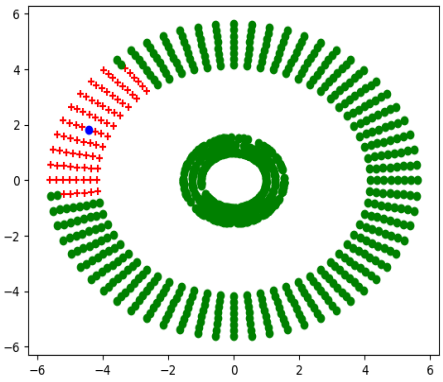
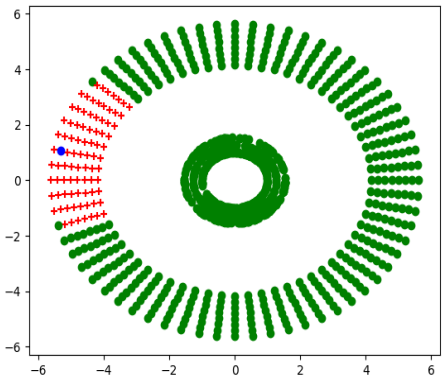
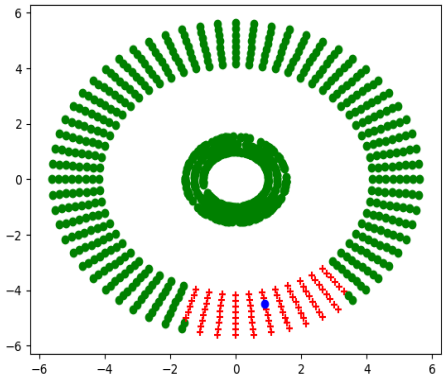
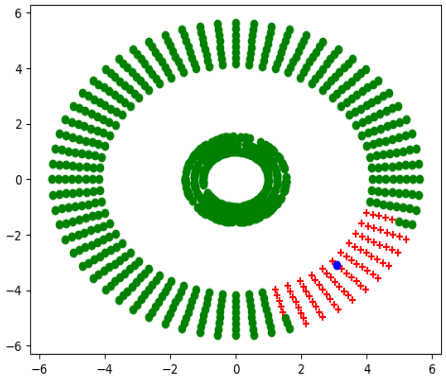
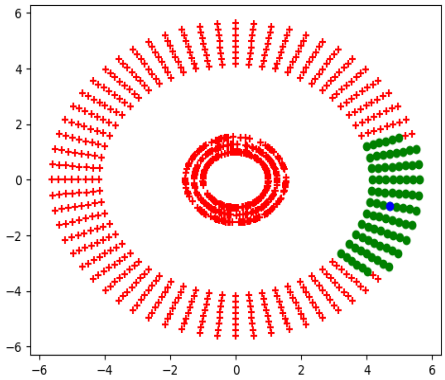
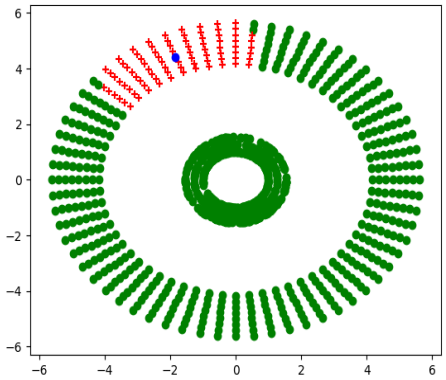


Figure 11: Clustered Data

2. Using Kernels

Observation:

When the given data is transformed using the RBF kernel then the new features obtained are inversely proportional to the euclidean distance from the landmark point. So, ideally two clusters should be formed, one is the set of points closer to landmark point and other, the set of points farther away from the landmark point. One landmark point proves to be insufficient in this case. The results would be best when the landmark point is close to the center of the two circles or better when the landmark point is as close to the center as possible.