Introduction to ML (CS771), Autumn 2020
Indian Institute of Technology Kanpur
Homework Assignment Number 1

Student Name: Abhas Kumar
Roll Number: 20111001
Date: November 27, 2020

QUESTION 1

We have,

$$\mathcal{L}(\mathbf{w}) = -\sum_{n=1}^{N} y_n \mathbf{w}^T \mathbf{x}_n - \log(1 + \exp(\mathbf{w}^T \mathbf{x}_n))$$

and the newton's update rule is

$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \mathcal{H}^{(t)^{-1}} \mathbf{g}^{(t)}.$$

$$\mathbf{g} = \frac{\partial \mathcal{L}(\mathbf{w})}{\partial \mathbf{w}} = -\sum_{n=1}^{N} y_n \mathbf{x}_n - \frac{\exp(\mathbf{w}^T \mathbf{x}_n)}{1 + \exp(\mathbf{w}^T \mathbf{x}_n)} \mathbf{x}_n = -\sum_{n=1}^{N} y_n \mathbf{x}_n - \boldsymbol{\alpha_n} \mathbf{x}_n = \sum_{n=1}^{N} (\boldsymbol{\alpha_n} - y_n) \mathbf{x}_n = \mathbf{X}^T (\boldsymbol{\alpha} - \mathbf{y})$$

where, $\boldsymbol{\alpha_n} = \dfrac{\exp(\mathbf{w}^T \mathbf{x}_n)}{1 + \exp(\mathbf{w}^T \mathbf{x}_n)}$.

Now Hessian, $\mathcal{H} = \dfrac{\partial(\sum_{n=1}^{N} (\boldsymbol{\alpha_n} - y_n)\mathbf{x}_n)}{\partial \mathbf{w}} = \dfrac{\partial(\sum_{n=1}^{N} \boldsymbol{\alpha_n}}{\partial \mathbf{w}} \mathbf{x}_n$

$$\frac{\partial \boldsymbol{\alpha_n}}{\partial \mathbf{w}} = \frac{\partial \left(\frac{\exp(\mathbf{w}^T \mathbf{x}_n)}{1 + \exp(\mathbf{w}^T \mathbf{x}_n)} \right)}{\partial \mathbf{w}} = \frac{(1 + \exp(\mathbf{w}^T \mathbf{x}_n)) \exp(\mathbf{w}^T \mathbf{x}_n) \mathbf{x}_n - \exp(\mathbf{w}^T \mathbf{x}_n) \exp(\mathbf{w}^T \mathbf{x}_n) \mathbf{x}_n}{(1 + \exp(\mathbf{w}^T \mathbf{x}_n))^2}$$

$$= \mathbf{x}_n \left(\frac{\exp(\mathbf{w}^T \mathbf{x}_n)}{(1 + \exp(\mathbf{w}^T \mathbf{x}_n))^2} \right) = \mathbf{x}_n \left(\frac{\exp(\mathbf{w}^T \mathbf{x}_n)}{1 + \exp(\mathbf{w}^T \mathbf{x}_n)} \right) \left(1 - \frac{\exp(\mathbf{w}^T \mathbf{x}_n)}{1 + \exp(\mathbf{w}^T \mathbf{x}_n)} \right) = \mathbf{x}_n \boldsymbol{\alpha_n} (1 - \boldsymbol{\alpha_n})$$

Hence $\mathcal{H} = \sum_{n=1}^{N} \boldsymbol{\alpha_n} (1 - \boldsymbol{\alpha_n}) \mathbf{x}_n^T \mathbf{x}_n = \mathbf{X}^T \mathbf{D} \mathbf{X}$.

Where $\mathbf{D} = \begin{bmatrix} \alpha_1(1-\alpha_1) & 0 & 0 & 0 & 0 \\ 0 & \alpha_2(1-\alpha_2) & 0 & 0 & 0 \\ . & . & . & . & . \\ . & . & . & . & . \\ 0 & . & . & . & \alpha_n(1-\alpha_n) \end{bmatrix}$

According to Newton's update method we have,

$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \mathcal{H}^{(t)^{-1}} \mathbf{g}^{(t)}$$

$$= \mathbf{w}^{(t)} - (\mathbf{X}^T \mathbf{D}^{(t)} \mathbf{X})^{-1} \mathbf{X}^T (\boldsymbol{\alpha}^{(t)} - \mathbf{y})$$

$$= \mathbf{w}^{(t)} + (\mathbf{X}^T \mathbf{D}^{(t)} \mathbf{X})^{-1} \mathbf{X}^T (\mathbf{y} - \boldsymbol{\alpha}^{(t)})$$

$= (\mathbf{X}^T\mathbf{D}^{(t)}\mathbf{X})^{-1}(\mathbf{X}^T\mathbf{D}^{(t)}\mathbf{X}\mathbf{w}^{(t)} + \mathbf{X}^T(\mathbf{y} - \boldsymbol{\alpha}^{(t)}))$

$= (\mathbf{X}^T\mathbf{D}^{(t)}\mathbf{X})^{-1}\mathbf{X}^T(\mathbf{D}^{(t)}\mathbf{X}\mathbf{w}^{(t)} + (\mathbf{y} - \boldsymbol{\alpha}^{(t)}))$
$= (\mathbf{X}^T\mathbf{D}^{(t)}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{D}^{(t)}(\mathbf{X}\mathbf{w}^{(t)} + {\mathbf{D}^{(t)}}^{-1}(\mathbf{y} - \boldsymbol{\alpha}^{(t)}))$

Let, $\boldsymbol{\gamma}^{(t)} = \mathbf{D}^{(t)}$ and $\hat{\mathbf{y}}^{(t)} = \mathbf{X}\mathbf{w}^{(t)} + {\mathbf{D}^{(t)}}^{-1}(\mathbf{y} - \boldsymbol{\alpha}^{(t)})$

Hence, $\mathbf{w}^{(t+1)} = (\mathbf{X}^T\boldsymbol{\gamma}^{(t)}\mathbf{X})^{-1}\mathbf{X}^T\boldsymbol{\gamma}^{(t)}\hat{\mathbf{y}}^{(t)}$

Clearly it has resemblance with the least square sum solution i.e $(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$ with $\boldsymbol{\gamma}^{(t)}$ acting as the weight(importance) of an input and modified $\mathbf{y}$ as $\hat{\mathbf{y}}^{(t)}$.

So it is same as solving this objective function ,

$$\arg\min_{w} \sum_{n=1}^{N} \boldsymbol{\gamma}_n^{(t)} \left(\hat{\mathbf{y}}_n^{(t)} - \mathbf{w}^T\mathbf{x}_n\right)^2$$

Introduction to ML (CS771), Autumn 2020
Indian Institute of Technology Kanpur
Homework Assignment Number 1

Student Name: Abhas Kumar
Roll Number: 20111001
Date: November 27, 2020

QUESTION 2

We know that, **Primal Perceptron Update rule** is

if $y_n(\mathbf{w}^T\mathbf{x}_n) < 0$:
. $\quad \mathbf{w} = \mathbf{w} + y_n\mathbf{x}_n$

To derive a kernelized version of the perceptron algorithm, we must first formulate it in **dual form**.

We have $\mathbf{w} = \sum_{n=1}^{N} \alpha_n y_n \mathbf{x}_n$, where α_n is the number of times $\mathbf{x}_n$ was misclassified.

$$\implies \mathbf{w}^T\mathbf{x} = \left(\sum_{n=1}^{N} \alpha_n y_n \mathbf{x}_n\right)^T \mathbf{x}$$

$$\implies \mathbf{w}^T\mathbf{x} = \sum_{n=1}^{N} \alpha_n y_n \mathbf{x}_n \cdot \mathbf{x}$$

Dual Perceptron Update rule: instead of storing and updating $\mathbf{w}$, it updates α.

if $y_m\left(\sum_{n=1}^{N} \alpha_n \mathbf{x}_n \cdot \mathbf{x}_m\right) < 0$:
. $\quad \alpha_m = \alpha_m + 1$

We can replace the dot product in the dual perceptron by a kernel function $\mathbf{K}$, to get the effect of a feature map $\phi(\mathbf{x})$.

i.e $\sum_{n=1}^{N} \alpha_n(\mathbf{x}_n \cdot \mathbf{x}_m) = \sum_{n=1}^{N} \alpha_n(\phi(\mathbf{x}_n) \cdot \phi(\mathbf{x}_m)) = \sum_{n=1}^{N} \alpha_n \mathbf{K}(\mathbf{x}_n \cdot \mathbf{x}_m)$

Kernel Perceptron.

1. Initialize α to a zero vector of size n
2. For each training example $(\mathbf{x}_m, \mathbf{y}_m)$
. $\quad$ if $\mathbf{y}_m\left(\sum_{n=1}^{N} \alpha_n \mathbf{K}(\mathbf{x}_n \cdot \mathbf{x}_m)\right) < 0$:
. $\quad\quad \alpha_m = \alpha_m + 1$
3. Repeat 2 until convergence

Introduction to ML (CS771), Autumn 2020
Indian Institute of Technology Kanpur
Homework Assignment Number 1

Student Name: Abhas Kumar
Roll Number: 20111001
Date: November 27, 2020

QUESTION 3

Objective function for this problem can be formulated as

$$\min_{\mathbf{w},b,\xi} \left\{ \frac{\|\mathbf{w}\|^2}{2} + \mathcal{C}_+\Big(\sum_{n:y_n=+1}^{N} \xi_n\Big) + \mathcal{C}_-\Big(\sum_{n:y_n=-1}^{N} \xi_n\Big) \right\}$$

such that, $y_n(\mathbf{w}^T\mathbf{x}_n + b) \geq 1 - \xi_n$ and $\xi_n \geq 0, \forall n$

where $\mathcal{C}_+$ and $\mathcal{C}_-$ are misclassification costs for positive class and negative class examples respectively.
Lagrangian problem of this objective function, can be formulated using two lagrangian multilpliers $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, ..., \alpha_N]$ and $\boldsymbol{\beta} = [\beta_1, \beta_2, ..., \beta_N]$.

$\mathcal{L}(\mathbf{w}, b, \xi) =$

$$\min_{\mathbf{w},b,\xi} \max_{\alpha,\beta\geq 0} \frac{\|\mathbf{w}\|^2}{2} + \mathcal{C}_+\Big(\sum_{n:y_n=+1}^{N} \xi_n\Big) + \mathcal{C}_-\Big(\sum_{n:y_n=-1}^{N} \xi_n\Big) + \sum_{n=1}^{N} \alpha_n\{1 - y_n(\mathbf{w}^T\mathbf{x} + b) - \xi_n\} - \sum_{n=1}^{N} \beta_n\xi_n$$

$$= \min_{\mathbf{w},b,\xi} \max_{\alpha,\beta\geq 0} \frac{\|\mathbf{w}\|^2}{2} + \mathcal{C}_+\Big(\sum_{n:y_n=+1}^{N} \xi_n\Big) + \mathcal{C}_-\Big(\sum_{n:y_n=-1}^{N} \xi_n\Big) + \sum_{n=1}^{N} \alpha_n - \sum_{n=1}^{N} \alpha_n y_n \mathbf{w}^T\mathbf{x} - \sum_{n=1}^{N} \alpha_n y_n b - \sum_{n=1}^{N} \alpha_n\xi_n - \sum_{n=1}^{N} \beta_n\xi_n$$

On Taking derivative with respect to primal variables $\mathbf{w}, b, \xi$ and setting it to 0.

$$\frac{\partial\mathcal{L}(\mathbf{w}, b, \xi)}{\partial\mathbf{w}} = \mathbf{w} + 0 + 0 + 0 - \sum_{n=1}^{N} \alpha_n y_n \mathbf{x}_n - 0 - 0 - 0 = 0 \implies \mathbf{w} = \sum_{n=1}^{N} \alpha_n y_n \mathbf{x}_n.$$

$$\frac{\partial\mathcal{L}(\mathbf{w}, b, \xi)}{\partial b} = 0 + 0 + 0 + 0 + 0 - \sum_{n=1}^{N} \alpha_n y_n - 0 - 0 = 0 \implies \sum_{n=1}^{N} \alpha_n y_n = 0.$$

$$\frac{\partial\mathcal{L}(\mathbf{w}, b, \xi)}{\partial\xi} = 0 + \mathcal{C}_+ + \mathcal{C}_- + 0 + 0 - 0 - \alpha_n - \beta_n = 0 \implies \mathcal{C}_+ + \mathcal{C}_- - \alpha_n = \beta_n$$

$\implies \mathcal{C}_+ - \alpha_+ \geq 0$ and $\mathcal{C}_- - \alpha_- \geq 0$, since $\beta_n \geq 0$

$\implies \alpha_+ \leq \mathcal{C}_+$ and $\alpha_- \leq \mathcal{C}_-$, where α_+ and α_- represent the Lagrangian multipliers of positive and negative examples, respectively.

putting $\mathbf{w} = \sum_{n=1}^{N} \alpha_n y_n \mathbf{x}_n$, $\sum_{n=1}^{N} \alpha_n y_n = 0$ and $\mathcal{C}_+ + \mathcal{C}_- - \alpha_n = \beta_n$ back in the equation we get,

The **Dual Lagrangian form** as

$$\max_{\boldsymbol{\alpha}} \left\{ \frac{1}{2} \sum_{n,m=1}^{N} \boldsymbol{\alpha_m \alpha_n} y_m y_n (\mathbf{x}_m \cdot \mathbf{x}_n) + \sum_{n=1}^{N} \alpha_n - \sum_{n,m=1}^{N} \boldsymbol{\alpha_m \alpha_n} y_m y_n (\mathbf{x}_m \cdot \mathbf{x}_n) \right\}$$

$$= \max_{\boldsymbol{\alpha}} \left\{ \sum_{n=1}^{N} \alpha_n - \frac{1}{2} \sum_{n,m=1}^{N} \boldsymbol{\alpha_m \alpha_n} y_m y_n (\mathbf{x}_m \cdot \mathbf{x}_n) \right\}$$

such that $\sum_{n=1}^{N} \alpha_n y_n = 0$,$0 \leq \alpha_+ \leq \mathcal{C}_+$ and $0 \leq \alpha_- \leq \mathcal{C}_-$

**By assigning a higher misclassification cost for the positive class examples than the negative class examples (i.e $\boldsymbol{\mathcal{C}_+ > \mathcal{C}_-}$), model can handle the case where sometimes it costs us a lot more to classify negative points as positive than positive points as negative. In standard SVM dual problem just the single hyperparameter $\mathcal{C}$ can not handle such cases.

Introduction to ML (CS771), Autumn 2020
Indian Institute of Technology Kanpur
Homework Assignment Number 1

Student Name: Abhas Kumar
Roll Number: 20111001
Date: November 27, 2020

QUESTION 4

Standard k-means loss function is given as

$$\mathcal{L}(\mathbf{X}, \mathbf{Z}, \boldsymbol{\mu}) = \sum_{n=1}^{N} \sum_{k=1}^{K} \mathbf{z}_{nk} \|\mathbf{x}_n - \boldsymbol{\mu_k}\|^2$$

SGD K-means

STEP 1:To Make it online by taking a random example $\mathbf{x}_n$ at a time, and then assigning $\mathbf{x}_n$ "greedily" to the "best" cluster.

Using **ALT-OPT** technique
Fix $\boldsymbol{\mu} = \hat{\boldsymbol{\mu}}$ and solve for $\mathbf{z}_n$.

$$\hat{\boldsymbol{z_n}} = \arg\min_{\hat{\boldsymbol{z_n}}} \sum_{k=1}^{K} z_{nk} \|\mathbf{x}_n - \hat{\boldsymbol{\mu}}_{\boldsymbol{k}}\|^2$$

$$= \arg\min_{\hat{\boldsymbol{z_n}}} z_{nk} \|\mathbf{x}_n - \hat{\boldsymbol{\mu}}_{\boldsymbol{z_n}}\|^2$$

To perform Step 1 we need to assign a cluster to $\mathbf{x}_n$ using the above equation for each of the example $\{\mathbf{x}_n\}_{n=1}^{N}$.

STEP 2:To update the cluster means.
Solving for $\boldsymbol{\mu}$ using SGD on the objective function by fixing $\mathbf{z} = \hat{\mathbf{z}}$

$$\hat{\boldsymbol{\mu}} = \arg\min_{\mu} \mathcal{L}(\mathbf{X}, \hat{\mathbf{Z}}, \mu) = \arg\min_{\mu} \left\{ \sum_{n=1}^{N} \sum_{n:\hat{\mathbf{z}}_\mathbf{n}=k} ||\mathbf{x_n} - \mu_\mathbf{k}||^2 \right\}$$

$$\hat{\mu_\mathbf{k}} = \arg\min_{\mu_k} \left\{ \sum_{n:\hat{\mathbf{z}}_\mathbf{n}=k} ||\mathbf{x_n} - \mu_\mathbf{k}||^2 \right\}$$

At any iteration $\mathbf{t}$, choose an example $\mathbf{x}_n$ uniformly randomly and approximate $\mathbf{g}$ as

$$\mathbf{g} \approx \mathbf{g_n} = \frac{\partial}{\partial \mu_\mathbf{k}} (||\mathbf{x_n} - \mu_\mathbf{k}||^2) = -2(\mathbf{x_n} - \mu_\mathbf{k})$$

Now mean can be updated as, $\boldsymbol{\mu}_k^{(t+1)} = \boldsymbol{\mu_k}^{(t)} - \eta \mathbf{g}^{(t)}$

$$\implies \boldsymbol{\mu}_k^{(t+1)} = \boldsymbol{\mu_k}^{(t)} + 2\eta(\mathbf{x_n}^{(t)} - \mu_\mathbf{k}^{(t)})$$

Step size can be $\eta \propto \frac{1}{N_k}$, where N_k is number of data points in k^{th} cluster so that the updated mean would also be the in ratio of the sum of features of every data point to the total numbers of data points in the that cluster.

Introduction to ML (CS771), Autumn 2020
Indian Institute of Technology Kanpur
Homework Assignment Number 1

Student Name: Abhas Kumar
Roll Number: 20111001
Date: November 27, 2020

QUESTION 5

Standard k-means loss function is given as

$$\mathcal{L}(\mathbf{X}, \mathbf{Z}, \boldsymbol{\mu}) = \sum_{n=1}^{N} \sum_{k=1}^{K} \mathbf{z}_{nk} \|\mathbf{x}_n - \boldsymbol{\mu_k}\|^2$$

To implement the kernel K-means, we can replace the Euclidean distance/similarity computations in this loss function by kernelized version of it, i.e $\|\mathbf{x}_n - \boldsymbol{\mu_k}\|^2$ by $\|\phi(\mathbf{x}_n) - \phi(\boldsymbol{\mu_k})\|^2$.

$$\|\phi(\mathbf{x}_n) - \phi(\boldsymbol{\mu_k})\|^2 = \|\phi(\mathbf{x}_n)\|^2 + \|\phi(\boldsymbol{\mu_k})\|^2 - 2\phi(\mathbf{x}_n)^T \phi(\boldsymbol{\mu_k})$$

$$= \mathbf{K}(\mathbf{x}_n, \mathbf{x}_n) + \mathbf{K}(\boldsymbol{\mu_k}, \boldsymbol{\mu_k}) - 2\mathbf{K}(\mathbf{x}_n, \boldsymbol{\mu_k})$$

Even though we can neither store the kernel-induced feature map representation of the data points nor can store the cluster means in the kernel-induced feature space but still we implement the kernel K-means algorithm in practice as,the implicit function ϕ doesn't have to be computed/stored for data $\{\mathbf{x}_n\}_{n=1}^N$ or for the cluster means $\{\boldsymbol{\mu_k}\}_{k=1}^K$ because computations will only depend on kernel evaluations.

Kernel K-means.

Redefine the mean as

$$\boldsymbol{\mu_k} = \frac{\sum_{n=1}^{N} z_{nk} \phi(\mathbf{x}_n)}{\sum_{n=1}^{N} z_{nk}}$$

Now the distance between the transformed data and this mean can be calculate as.

$$\|\phi(\mathbf{x}) - \boldsymbol{\mu_k}\|^2 = \mathbf{k}(\mathbf{x}, \mathbf{x}) - 2\sum_{n=1}^{k} z_{nk} \mathbf{K}(\mathbf{x}, \mathbf{x}_n) + \sum_{m,n=1}^{N} z_{mk} z_{nk} \mathbf{k}(\mathbf{x}_m, \mathbf{x}_n)$$

1.Initialise means to the first $\phi(\mathbf{x}_k)$, k=1,2,...,K.
2.Repeat 3 and 4 until converged

. 3. Assign each $\mathbf{x}_n$, $n > k$ to the closest mean $\boldsymbol{\mu_\alpha}$
. i.e $\mathbf{Z}_{n,\alpha} = 1 \ \forall \alpha \neq \beta$ if $\|\phi(\mathbf{x}) - \boldsymbol{\mu_\alpha}\|^2 < \|\phi(\mathbf{x}) - \boldsymbol{\mu_\beta}\|^2$ else 0.

. 4.Update the mean using $\frac{\sum_{n=1}^{N} z_{nk} \phi(\mathbf{x}_n)}{\sum_{n=1}^{N} z_{nk}}$

**The complexity of standard k-means is $\mathcal{O}(NKDt)$ where N is the numbers of data points, K is number of clusters, D is dimension. If we have $N \times N$ kernel matrix then complexity of kernel k-means can be given as $\mathcal{O}(N^2KDt)$ where $\mathbf{t}$ is the number of iterations taken to converge.Updation of mean every time in Kernal k-means involves computation with all N data points which is not the case with standard k-means.

Introduction to ML (CS771), Autumn 2020
Indian Institute of Technology Kanpur
Homework Assignment Number 1

Student Name: Abhas Kumar
Roll Number: 20111001
Date: November 27, 2020

QUESTION

6

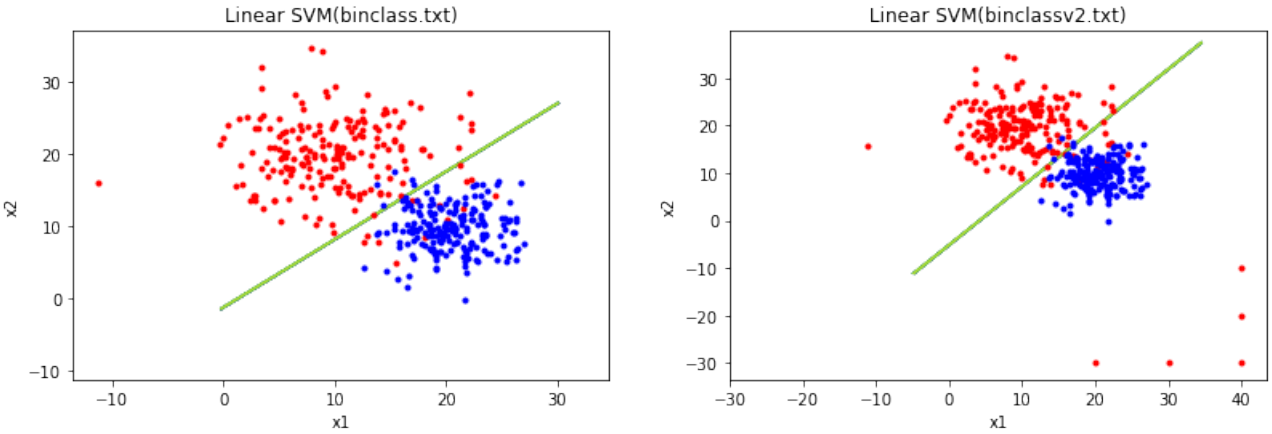

Figure 1: Plots obtained for SVM Linear classifier(by using scikit-learn).

1. Generative classification model for the binclass.txt and binclassv2.txt dataset assuming Gaussian class-conditional distribution with same variance learns a linear decision boundary while with different variance for positive and negative class learns non-linear decision boundary[figure 2].

2. Dataset binclassv2.txt contained some outliers points for the positive class(red points)[figure 1][Figure 2].

3. In generative case, the decision boundary comes more close to the blue clusters, possibly because variance for red class is more due to the presence of outliers.[figure 2]

4. With the linear SVM[figure 1] there is a slight shift of decision boundary towards the red class as the decision boundary has tried to shift a little to decrease number of crossovers.[figure 1](more number of data points on the decision boundary for dataset binclassv2.txt).

5. In general Guassian seems to overfit the data[figure1] while SVM generalises well on both the dataset[figure 2].

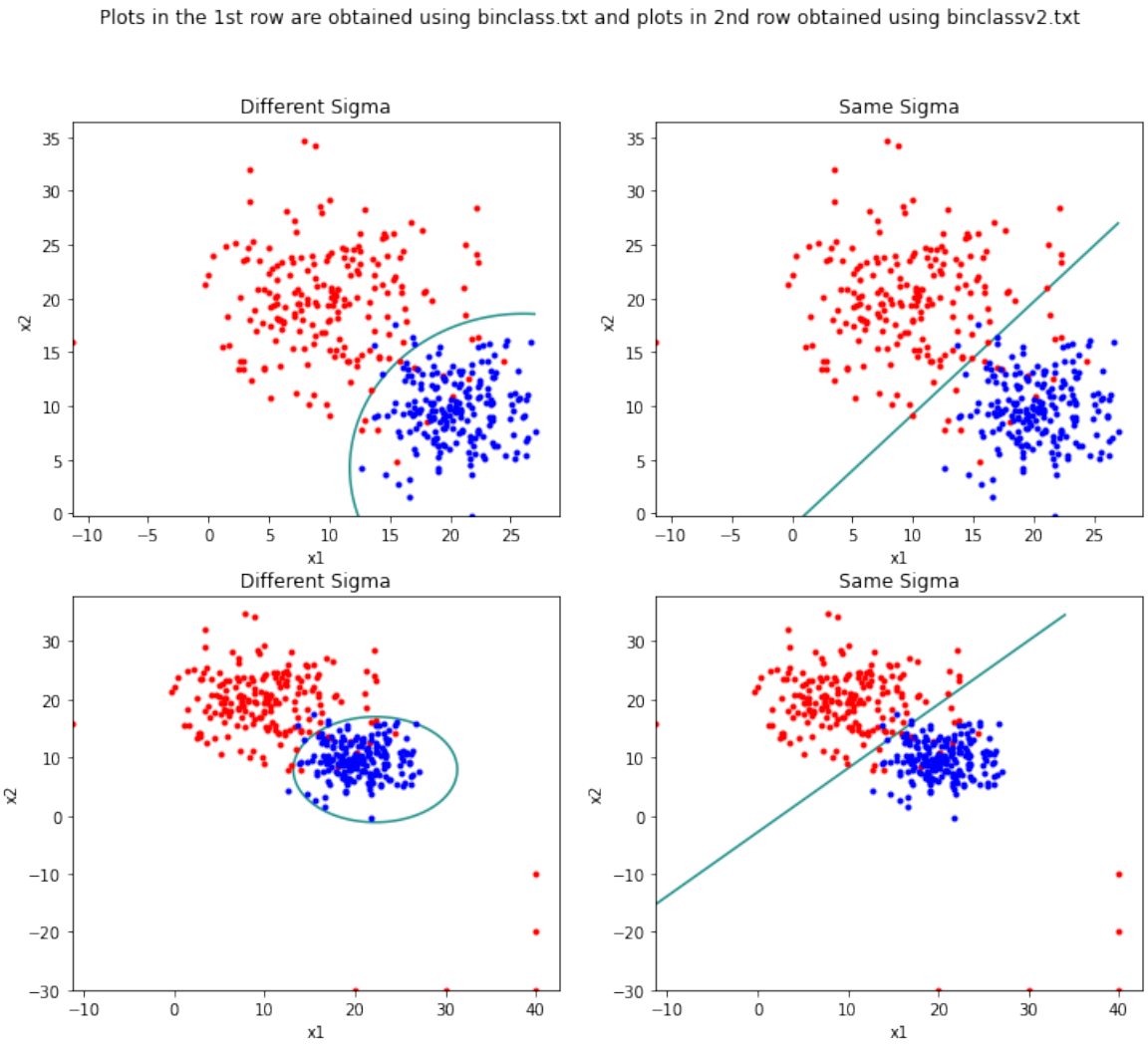

Figure 2: Plots obtained for generative classification model.