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

Student Name: Jinang Shah
Roll Number: 170649
Date: November 27, 2020

QUESTION 1

We are given loss function for Logistic Regression: $L(w) = -\sum_{n=1}^{N}(y_n w^T x_n - log(1+exp(w^T x_n)))$
Here, $y_n, w^T x_n$ are scalar, whereas, $w = [w_1, .., w_d, ..., w_D]^T, x_n = [x_{n1}, .., x_{nd}, ..., x_{nD}]^T$

$\nabla L(w) = \sum_{n=1}^{N}(\frac{exp(w^T x_n)}{1+exp(w^T x_n)} - y_n)x_n$; Assume $p_n^t = \frac{1}{1+exp(-{w^t}^T x_n)} \Rightarrow g^t = \sum_{n=1}^{N}(p_n^t - y_n)x_n$

$\frac{dL(w)}{dw_i dw_j} = \sum_{n=1}^{N}(\frac{exp(w^T x_n)}{1+exp(w^T x_n)})(\frac{1}{1+exp(w^T x_n)})x_{ni}x_{nj} = \sum_{n=1}^{N}(p_n(1-p_n))x_{ni}x_{nj}$
$H^t = \nabla^2 L(w^t) = \sum_{n=1}^{N}(p_n^t(1-p_n^t))x_n x_n^T$, here $p_n^t = \frac{1}{1+exp(-{w^t}^T x_n)}$

To reduce to the asked form of *argmin*, we'll have to start with similar form of Newton's method, which is as follows:

$$\begin{aligned}
w^{t+1} &= argmin_w\{L(w^t) + {g^t}^T(w-w^t) + 0.5(w-w^t)^T H^t(w-w^t)\} \\
&= argmin_w\{{g^t}^T(w-w^t) + 0.5(w-w^t)^T H^t(w-w^t)\} \\
&= argmin_w\{\sum_{n=1}^{N}(p_n^t - y_n)x_n^T(w-w^t) + 0.5(p_n^t(1-p_n^t))(w-w^t)^T x_n x_n^T(w-w^t)\} \\
&= argmin_w\{\sum_{n=1}^{N} -(y_n - p_n^t)(w-w^t)^T x_n + 0.5(p_n^t(1-p_n^t))((w-w^t)^T x_n)^2\} \\
&= argmin_w\{\sum_{n=1}^{N} 0.5(p_n^t(1-p_n^t))[((w-w^t)^T x_n)^2 - 2(w-w^t)^T x_n\frac{(y_n-p_n^t)}{p_n^t(1-p_n^t)}]\} \\
&= argmin_w\{\sum_{n=1}^{N} 0.5(p_n^t(1-p_n^t))(\frac{(y_n-p_n^t)}{p_n^t(1-p_n^t)} - (w-w^t)^T x_n)^2\} \\
&= argmin_w\{\sum_{n=1}^{N} 0.5(p_n^t(1-p_n^t))((\frac{(y_n-p_n^t)}{p_n^t(1-p_n^t)} + {w^t}^T x_n) - w^T x_n)^2\}
\end{aligned}$$

Comparing the derived equation for w^{t+1} with the equation, $argmin_w \sum_{n=1}^{N}\gamma_n^t(\hat{y}_n^t - w^T x_n)^2$
For, $p_n^t = \frac{1}{1+exp(-{w^t}^T x_n)}$, we get,

$\gamma_n^t = \frac{1}{2}p_n^t(1-p_n^t)$

$\hat{y}_n^t = \frac{(y_n-p_n^t)}{p_n^t(1-p_n^t)} + {w^t}^T x_n$, here, it is easily observable that $\hat{y}_n^t$ is a **real number** unlike y_n.

Also, here γ_n^t will achieve its maxima for $p_n^t = 0.5$, giving respective samples the highest importance. It also makes an intuitive sense here, as rather than focusing on the samples with $p_n^t = 0$ *or* 1, for which our model is almost confident about, the model gives higher importance to the samples for which our current model is the most uncertain about.

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

Student Name: Jinang Shah
Roll Number: 170649
Date: November 27, 2020

QUESTION 2

Weight vector learned with the standard perceptron can also be written as, $w = \sum_{n=1}^{N} \alpha_n y_n x_n$

To learn non-linear boundaries with perceptron, we will assume kernel k, feature map ϕ
We will have a transformation of $x_n \Rightarrow \phi(x_n)$, for which $k(x_m, x_n) = \phi(x_m)^T \phi(x_n)$

To use the form of w given in the first line, we will have to initialize w^0 with 0,$w^0 = 0$
For any t, w^t can we written as, $w^t = \sum_{k=1}^{N} \alpha_k y_k \phi(x_k)$, where α_k is the number of times **Kernelized Perceptron** makes mistake on sample k **up to the step** t.

At initialization, $t = 0$, $w^0 = 0$, $\alpha_n = 0; \forall n$,

Pick some sample n out of N randomly, $(\phi(x_n), y_n)$

If current w^t makes **mistake on sample n** i.e.,
$y_n {w^t}^T \phi(x_n) < 0 \implies \sum_{k=1}^{N} \alpha_k y_k y_n \phi(x_k)^T \phi(x_n) < 0 \implies \sum_{k=1}^{N} \alpha_k y_k y_n k(x_k, x_n) < 0$
{
$\implies \alpha_n = \alpha_n + 1$
$\implies t = t + 1$
}

If not converged, then again pick a new sample randomly, and repeat the process.

Here, if you notice, then we are not storing w directly, as it might go up to infinite features due to $\phi(x_k)$, so to update w, we actually only update α_n corresponding to the sample on which our model made mistake. Here, even **mistake condition** is also not dependent directly on w or $\phi(x_n)$. But instead it depends on the values of kernel k.

So, for this problem, **additionally**, we only have to store kernel k of NXN shape, and α_n for all n, as our model.

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

Student Name: Jinang Shah
Roll Number: 170649
Date: November 27, 2020

QUESTION
3

SVM with Unequal Class Importance

Objective function: $\min\limits_{w,b,\xi} \frac{\|w\|^2}{2} + \sum_{n=1}^{N} C_{y_n}\xi_n$, here $y_n \in \{-1,+1\}$

Constraints: $1 \le y_n(w^T x_n + b) + \xi_n$ and $-\xi_n \le 0, \forall n$

Overall objective: Introducing Lagrange multipliers α_n, β_n for each constraint, we get,

$$\min_{w,b,\xi} \max_{\alpha\ge 0,\beta\ge 0} L = \frac{\|w\|^2}{2} + \sum_{n=1}^{N} C_{y_n}\xi_n + \sum_{n=1}^{N} \alpha_n(1 - y_n(w^T x_n + b) - \xi_n) - \sum_{n=1}^{N} \beta_n \xi_n$$

$\frac{\partial L}{\partial w} = 0 \Rightarrow w = \sum_{n=1}^{N} \alpha_n y_n x_n$; $\frac{\partial L}{\partial b} = 0 \Rightarrow \sum_{n=1}^{N} \alpha_n y_n = 0$; $\frac{\partial L}{\partial \xi_n} = 0 \Rightarrow C_n - \alpha_n - \beta_n = 0$

Since, $\beta_n \ge 0$, from the above equation with $\alpha_n \ge 0$, we can also state, $\alpha_n \le C_n$

Substituting derived values in the Lagrangian L, we get the dual problem,

$$\max_{\substack{0\le\alpha_n\le C_{y_n};\forall n\\ \beta\ge 0}} L = \sum_{n=1}^{N} \alpha_n - \frac{1}{2}\sum_{m=1}^{N}\sum_{n=1}^{N} \alpha_m\alpha_n y_m y_n (x_m^T x_n);\ s.t.\ \sum_{n=1}^{N} \alpha_n y_n = 0$$

Since, the dual variables β don't appear in the dual problem, the final objective function is,

$$\max_{0\le\alpha_n\le C_{y_n};\forall n} L = \sum_{n=1}^{N} \alpha_n - \frac{1}{2}\sum_{m=1}^{N}\sum_{n=1}^{N} \alpha_m\alpha_n y_m y_n (x_m^T x_n);\ s.t.\ \sum_{n=1}^{N} \alpha_n y_n = 0$$

In the standard SVM, the cost of misclassification is same for both the classes, which presumes a balanced dataset, but that might not be the case each time. In an unbalanced dataset, the class with the higher number of samples will effectively have a higher cost of misclassification, resulting in a favourable treatment from the standard model, while compensating for training error and margin.

To make the model impartial towards the unbalanced classes, different cost for misclassification C_{y_n} will definitely help. Class with the lower number of samples will have higher cost than the class with the higher number of samples for misclassification. This will help in reduce the partial treatment of the model as now the class with lower number of samples will have higher cost on training error, resulting in a more impartial hyperplane selection.

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

Student Name: Jinang Shah
Roll Number: 170649
Date: November 27, 2020

QUESTION 4

Online K-means Clustering with SGD

Assuming we are already given initialized means μ_k, and number of samples assigned to k^{th} cluster, N_k, for total of all K clusters.

For a new random sample x_n, K-means objective function will be, $L = \sum_{k=1}^{K} z_{nk} \|x_n - \mu_k\|^2$

Step 1
Since, $z_n = [z_{n1}, ..., z_{nk}, ..., z_{nK}]$ is a one hot vector, we can assign x_n to only one cluster, so it would be ideal to have it assigned to the cluster with the nearest mean from sample x_n.

Now, we can assign cluster k' to x_n, for which $k' = argmin_k \|x_n - \mu_k\|^2$

Step 2
Since only $z_{nk'}$ is 1 and only nonzero in z_n, we can write, $L = \|x_n - \mu_{k'}\|^2$

Since L only depends on $\mu_{k'}$, we have to update $\mu_{k'}$ only. For which, $\frac{\partial L}{\partial \mu_{k'}} = -2(x_n - \mu_{k'})$

SGD Update for $\mu_{k'}$: $\mu_{k'} = \mu_{k'} - \eta(-2(x_n - \mu_{k'})) \implies \mu_{k'} = \mu_{k'} + 2\eta(x_n - \mu_{k'})$

This update equation actually makes sense here, as we would just like to update mean of the cluster to which the new sample x_n is assigned in proportion to how apart x_n and $\mu_{k'}$ are.

η in the SGD update equation is the step size here. For K-mean, we would like to set η in such a way that the updated $\mu_{k'}$ becomes the actual mean of x_n and all the samples assigned to cluster k' before, a total of $N_{k'}$.

So, since we have $N_{k'}$, **new actual mean** would be, $\mu_{k'} = \frac{N_{k'}\mu_{k'} + x_n}{N_{k'}+1} = \mu_{k'} + \frac{1}{N_{k'}+1}(x_n - \mu_{k'})$

Comparing both the equations for $\mu_{k'}$, we get step size, $\eta = \frac{1}{2(N_{k'}+1)}$

At last, we will also **have to update** $N_{k'}$ as well, $N_{k'} = N_{k'} + 1$

This also makes an intuitive sense, as in K-means, we would like to have our updated mean at the actual mean of the samples assigned to the respective cluster. And thus, the chosen value of the step size makes sense here, as it actually considers all the samples, definitely indirectly, even in the online mode.

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

Student Name: Jinang Shah
Roll Number: 170649
Date: November 27, 2020

QUESTION 5

Kernelized K-means Clustering

We will assume we have a NXN matrix for kernel k, s.t. $k(x_m, x_n) = \phi(x_m)^T\phi(x_n)$; And here, N_k refers to the number of sample assigned to the cluster k out of a total of K clusters.

For a sample x_n, the kernelized distance with the cluster k mean would be, $\|\phi(x_n) - \phi(\mu_k)\|^2$
$\|\phi(x_n) - \phi(\mu_k)\|^2 = \|\phi(x_n)\|^2 + \|\phi(\mu_k)\|^2 - 2\phi(x_n)^T\phi(\mu_k)$

Here, $\phi(\mu_k) = \frac{1}{N_k} \sum\limits_{m:z_m=k} \phi(x_m)$; Here, z_m is the number of the cluster assigned to x_m.

$$\|\phi(x_n)-\phi(\mu_k)\|^2 = k(x_n, x_n)+((\frac{1}{N_k} \sum_{m:z_m=k} \phi(x_m))^T \frac{1}{N_k} \sum_{m:z_m=k} \phi(x_m))-2\phi(x_n)^T \frac{1}{N_k} \sum_{m:z_m=k} \phi(x_m)$$

$$\|\phi(x_n) - \phi(\mu_k)\|^2 = k(x_n, x_n) + (\frac{1}{N_k^2} \sum_{m_1:z_{m_1}=k} \sum_{m_2:z_{m_2}=k} k(x_{m_1}, x_{m_2})) - \frac{2}{N_k} \sum_{m:z_m=k} k(x_n, x_m)$$

Now, we have a distance metric in the form that doesn't require a direct use of means or samples.

Algorithm

1. Randomly initialize z_n for all n, to assign clusters randomly at the first
2. Reassign z_n for all n s.t. $z_n = argmin_k \|\phi(x_n) - \phi(\mu_k)\|^2$
3. Repeat step 2 until z_n doesn't change for all n during the reassignment

---

Here, we don't actually have or store the means in feature map ϕ unlike standard K-mean, where we actually store the D dimensional mean μ_k vectors, but instead we use the stored kernel k matrix for any required calculations of means μ_k or of samples x_n, during the training.

Ignoring time for memory retrieval, summation and subtraction operations. Only considering the time complexity for multiplication operations assuming them to be dominant.

Time cost of the cluster mean calculations of one input with one mean

Standard K-means: $\mathcal{O}(D)$; Kernelized K-means: $\mathcal{O}(N^2D)$

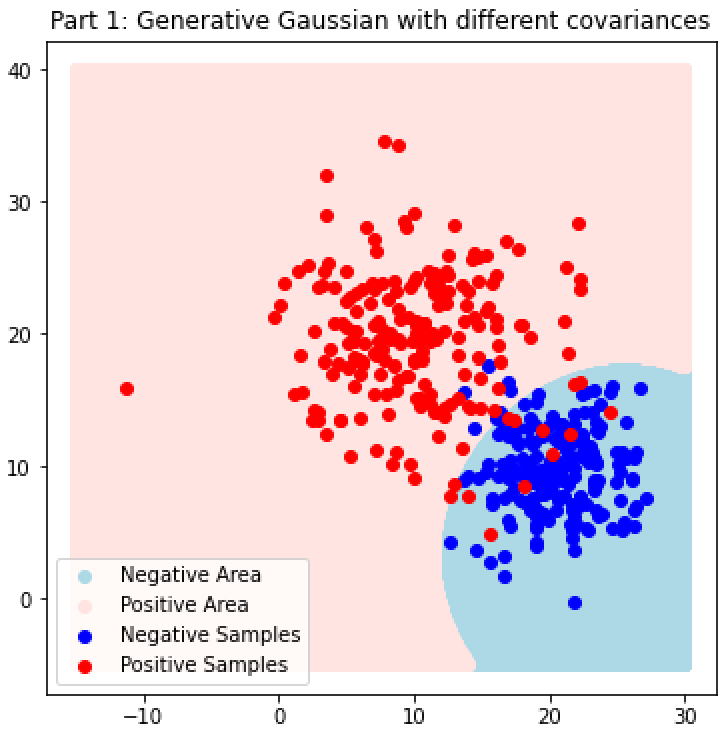
Additional N^2 term in the Kernelized K-means is due to the kernel matrix k of NXN.

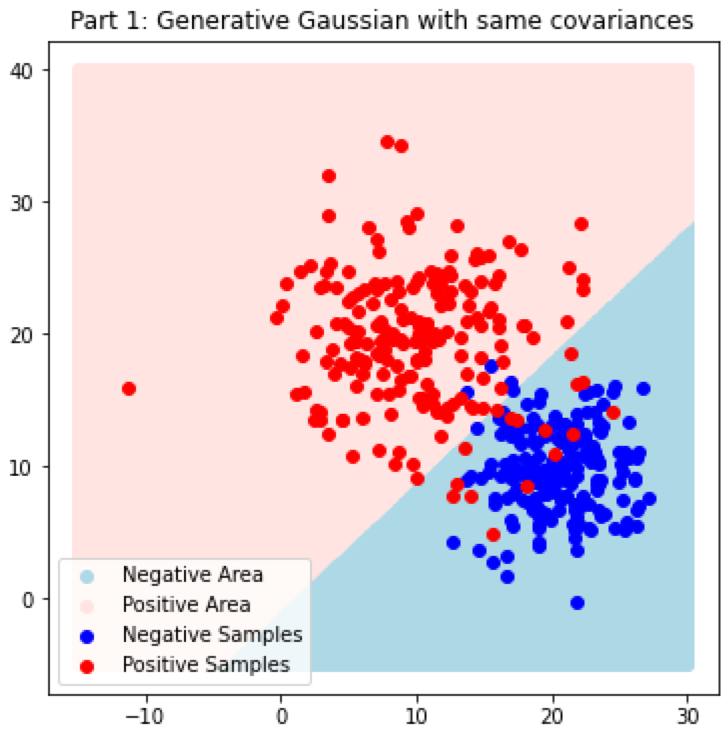
Introduction to ML (CS771), Autumn 2020
Indian Institute of Technology Kanpur
Homework Assignment Number 1

Student Name: Jinang Shah
Roll Number: 170649
Date: November 27, 2020

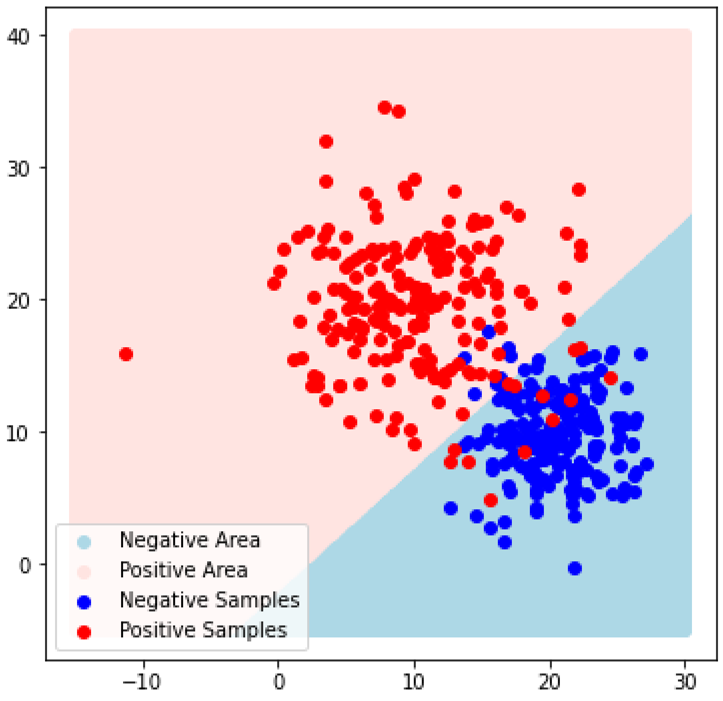
QUESTION

6

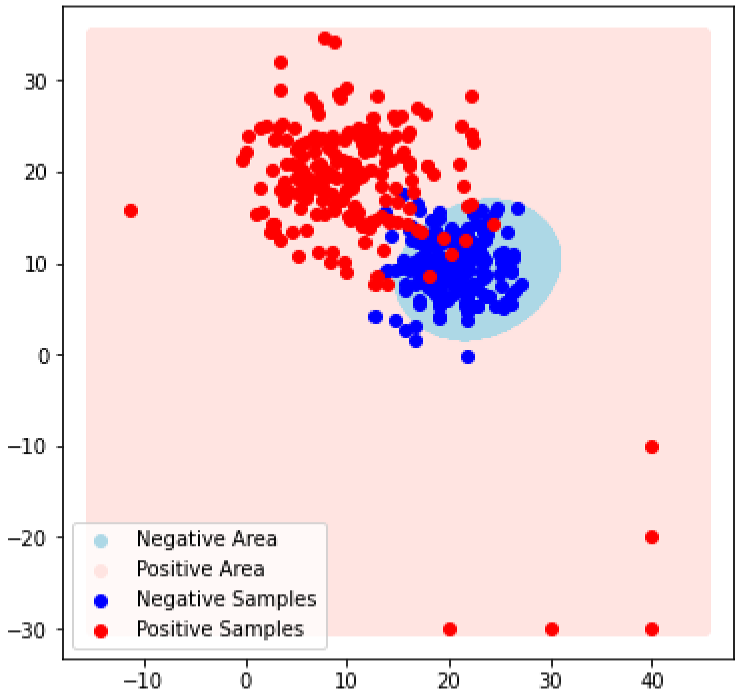


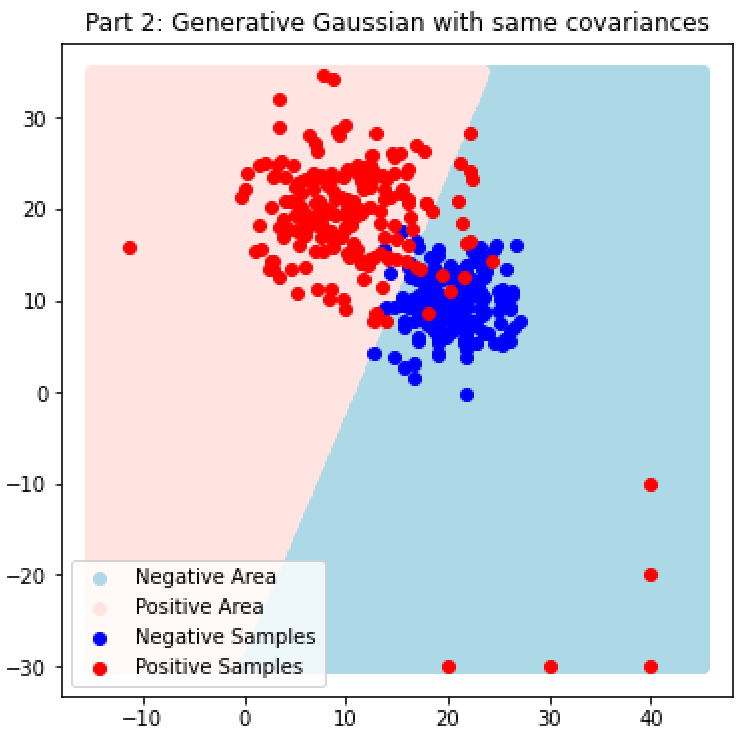


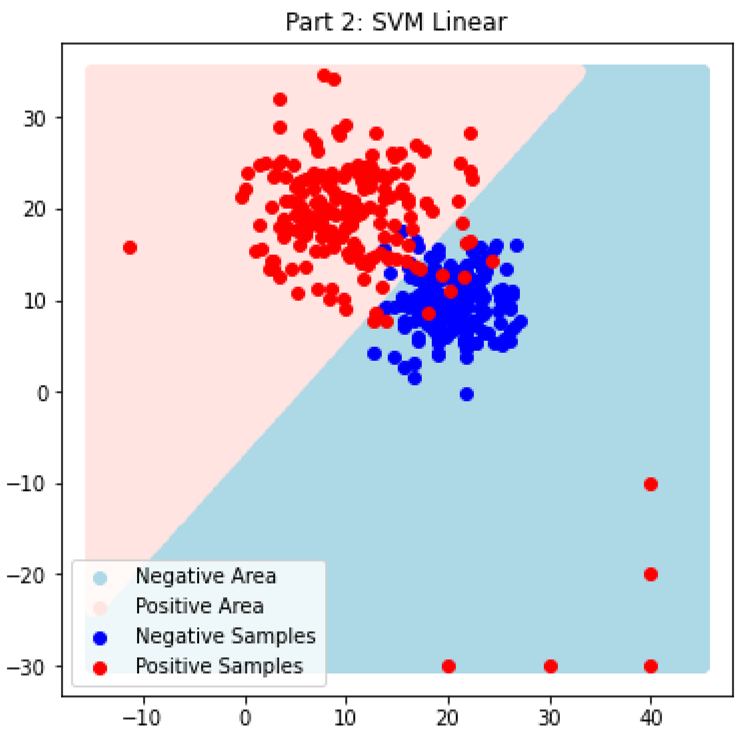
Part 1: SVM Linear



Part 2: Generative Gaussian with different covariances







All the labels and description about the figures are given inside or above each figure.

Here, positive and Negative area refer to the area in the feature space where the model has learnt to classify samples there as positive and negative class respectively. These areas also help in visualising the decision boundary learnt by the classifier.

For the problem of gaussian class conditionals with the same covariance matrices, the average covariance of the two classes has been considered.

It would be inappropriate to compare models which learn non-linear and linear boundaries. But for gaussian class-conditionals linear and SVM Linear, we might be able to do so. For part 1, since there are not that many extreme outliers both seems to be performing really well. But for part 2, SVM seems to be performing better than the gaussian class-conditionals linear. It still is the matter of perspective and it is hard to generalise anything from just two examples, but if I have to choose, then I would put my bet on SVM.