Section A

24/9/22 Date .....

ML Assignment -1

A1 1) $y_i = \hat{a}_0 + \hat{a}_1 x_i$

$\Rightarrow y_1 = \hat{a}_0 + \hat{a}_1 x_1$

$y_2 = \hat{a}_0 + \hat{a}_1 x_2$

$\vdots$

$y_n = \hat{a}_0 + \hat{a}_1 x_n$

$$\Rightarrow \sum_{i=1}^{n} y_i = \sum_{i=1}^{n} \hat{a}_0 + \hat{a}_1 \sum_{i=1}^{n} x_i$$

$$\sum_{i=1}^{n} y_i = n\hat{a}_0 + \hat{a}_1 \sum_{i=1}^{n} x_i$$

$$\Rightarrow \frac{\sum_{i=1}^{n} y_i}{n} = \hat{a}_0 + \frac{\hat{a}_1 \sum_{i=1}^{n} x_i}{n}$$

$$\therefore \bar{y}_i = \hat{a}_0 + \hat{a}_1 \bar{x}_i$$

Hence, proved

2) Given, X and Y are highly correlated with a third variable Z.
This does not imply that X and Y are also highly correlated with each other.

We know that,

$$r_{xy} = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2 \sum (y_i - \bar{y})^2}}$$

using this,

Date .....

$$\rho_{XY} = \rho_{XZ} \cdot \rho_{YZ} - \sqrt{(1-\rho_{XZ}^2)(1-\rho_{YZ}^2)}$$

eg take $\rho_{XZ} = 0.65$, $\rho_{YZ} = 0.75$

$\Rightarrow \rho_{XY} = 0.235$

So, X and Y are not highly correlated.

Taking a real life example, let us say that I want to reach a place using a car. Whether I would be able to reach the place depends on if the car is in proper working condition and does not break down on the way. It also depends on the amount of traffic on the road. However, there is no direct correlation b/w the condition of the car and the amount of traffic on road.

3) The weak law of large numbers (WLLN) states that →

Suppose $x_1, x_2, \ldots x_n$ are independent and identically distributed (iid) random variables, each with $E[x_i] = \mu < \infty$, $i = 1, 2, \ldots n$ and $S_n = x_1 + x_2 + \ldots x_n$ then, the average in probability converges to μ.

Proof →

$$S_n = x_1 + x_2 + \ldots x_n$$

$$\bar{S}_n = \frac{S_n}{n}$$

$$= \frac{x_1 + x_2 + \ldots x_n}{n} = \frac{n\mu_x}{n} = \mu_x$$

$$Var(\bar{S}_n) = Var\left\{\frac{x_1 + x_2 + \ldots x_n}{n}\right\}$$

$$= \frac{1}{n^2}\{Var(x_1) + Var(x_2) + \ldots Var(x_n)\}$$

$$= \frac{\sigma_x^2}{n}$$

Spiral

Date .....

From Chebyshev's Inequality,

~~$P[|\quad - E[x_n]| \geq \varepsilon] \leq Var(S_n$~~

$$P[|\bar{S}_n - \mu_2| \geq \varepsilon] \leq \frac{Var(\bar{S}_n)}{\varepsilon^2}$$

$$= \frac{\sigma_x^2}{n\varepsilon^2}$$

$$\therefore \lim_{n \to \infty} P[|\bar{S}_n - \mu_2| \geq \varepsilon] = 0$$

Hence, proved

4) For Maximum A Posteriori (MAP), we find the weights for our model, given data D.
So we find the most likely weights according to our data

$$P(W|D) = \frac{P(D|W) \times P(W)}{P(D)}$$

↳ weights
↳ data

$$\Rightarrow P(W|y,x) = \frac{P(y|x,W) \times P(W)}{P(y|x)}$$

where x, y are input and output. ①

$$\log(W|y,x) = \log P(y|x,W) + \log[P(W)] - \log[P(y|x)]$$

const

Now, $P(x|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}(x-\mu)^2}$

$$\Rightarrow P(W) = \frac{1}{\sqrt{2\pi\beta^2}} e^{-\frac{1}{2\beta^2}(W^T W)}$$

$$\log P(W) = \log\left(\frac{1}{2\pi\beta^2}\right) - \frac{1}{2\beta^2}(W^T W)$$

const

Spiral

Date .....

Now, we choose W that is most probable

$$\therefore \underset{W}{\operatorname{argmax}} \log(P(W|y,x))$$

$$= \underset{W}{\operatorname{argmax}} \log P(y|x,W) - \frac{1}{2\beta^2}(W^TW)$$

$$= \underset{W}{\operatorname{argmax}} \left(-\log P(y|x,W) - \frac{1}{2\beta^2}(W^TW)\right)$$

From ①,

$$= \underset{W}{\operatorname{argmax}} \frac{1}{2\sigma^2}\{(Y - W^Tx)^2 + \frac{1}{\beta^2}(W^TW)$$

$$W^{MAP} = \underset{W}{\operatorname{argmax}} \left(\sum_{i=1}^{n} \left(\frac{y_i - W^Tx_i}{2\sigma^2}\right)^2 + \frac{1}{\beta^2}W^TW\right)$$

$$W^{MAP} = \frac{d(J)}{dW}$$

$$0 = \frac{1}{\sigma^2}(W^Tx^Tx - y^Tx) + \frac{1}{\beta^2}W^T$$

$$= W^T\left(\frac{x^Tx}{\sigma^2} + \frac{1}{\beta^2}\right) - \frac{y^Tx}{\sigma^2}$$

$$W^T_{MAP} = \frac{y^Tx}{\sigma^2}\left(\frac{x^Tx}{\sigma^2} + \frac{1}{\beta^2}\right)^{-1}$$

$$W^T_{MAP} = \left(x^Tx + \frac{\sigma^2}{\beta^2}\right)^{-1} xy^T$$

Hence, proved.

Section B

Part A

The optimal value of K comes out to be **5** using a learning rate of 0.005 and 250 iterations, as it has the lowest RMSE. This is because when using k=5, the data is effectively split into 80:20, which is higher than 2 fold (50:50 split), 3 fold (67:33 split) and 4 fold (75:25 split). A higher amount of data to train means higher accuracy and lower RMSE.

Mean RMSE for two folds =

```
[[14.68573098]
 [14.64185808]]
14.663794531657334
```

Mean RMSE for three folds =

```
[[10.04605486]
 [ 9.63758499]
 [10.7391896 ]]
10.140943149898648
```

Mean RMSE for four folds =

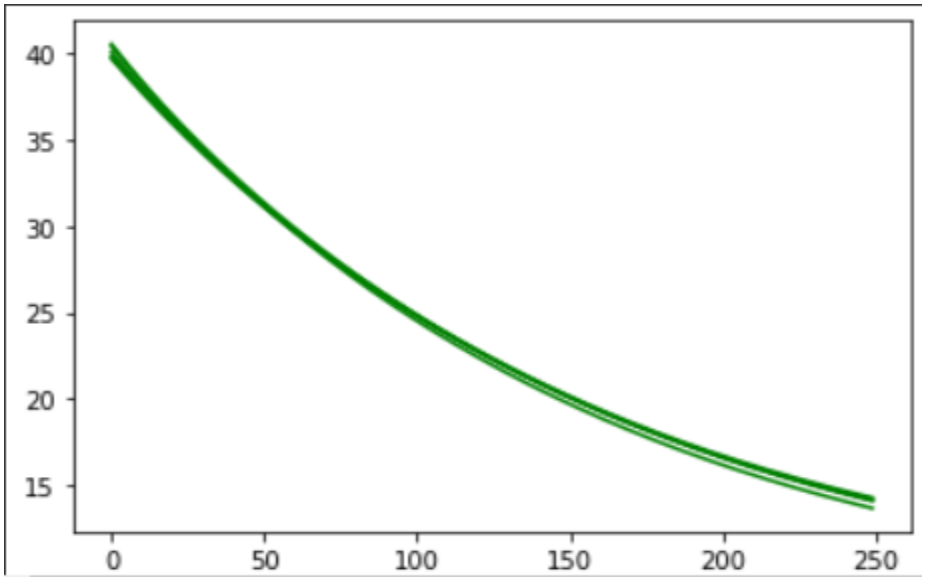
```
[[8.28165221]
 [8.13315413]
 [7.87379918]
 [8.78287795]]
8.267870868383518
```

Mean RMSE for five folds =

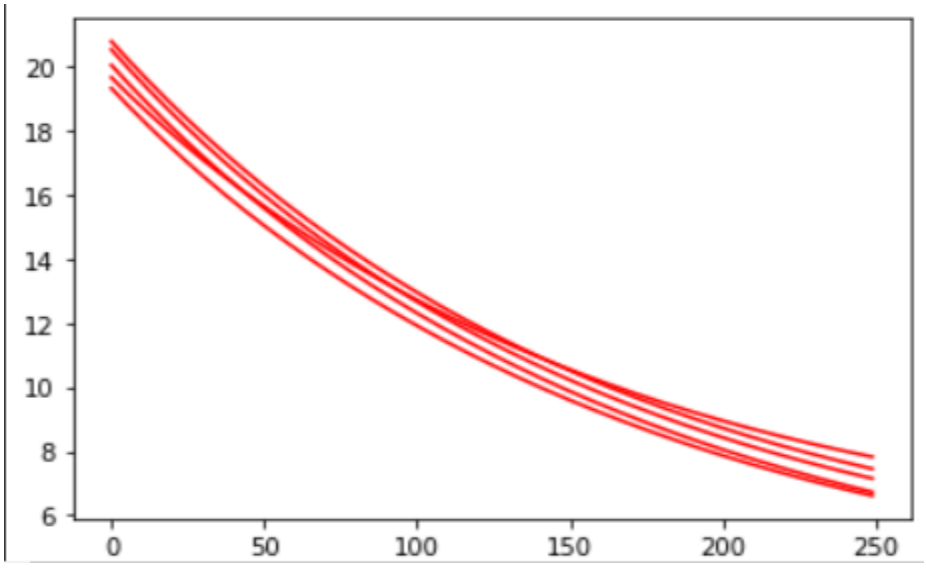
```
[[7.14780641]
 [6.72598242]
 [6.61139442]
 [7.44873727]
 [7.82236334]]
7.15125677258799
```

Part B

Training RMSE for K=5 folds



Testing RMSE for K=5 folds



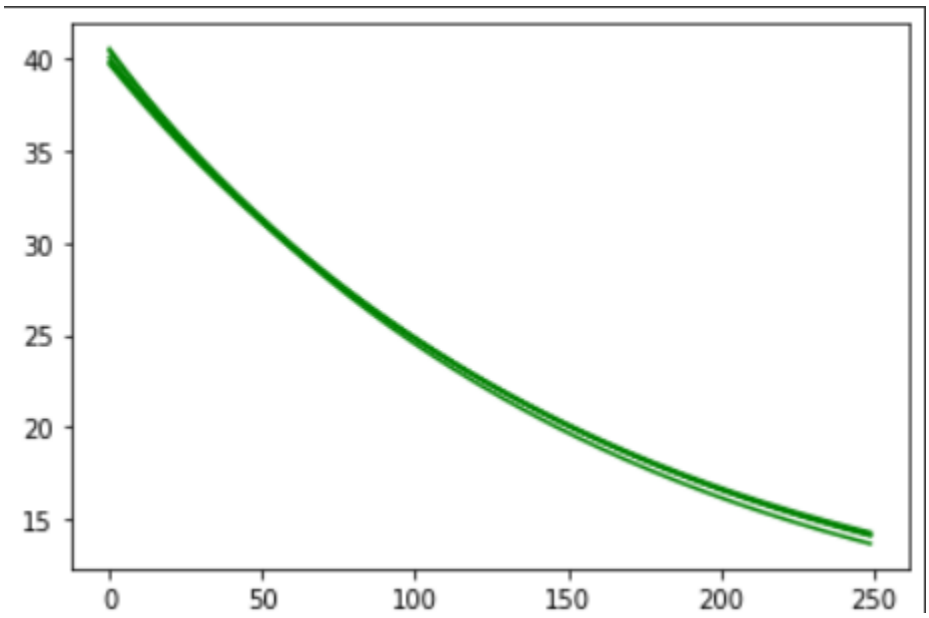
RMSE for five folds and mean RMSE =

```
[[7.14780641]
 [6.72598242]
 [6.61139442]
 [7.44873727]
 [7.82236334]]
7.15125677258799
```

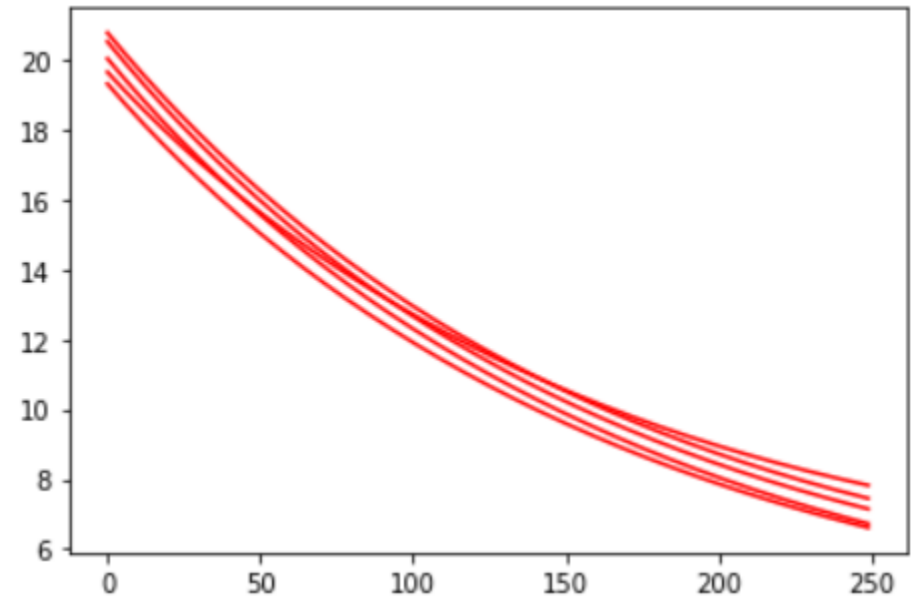
Part C

Lasso Regularisation

Best value for lambda = 0.01
Training RMSE for K=5 folds

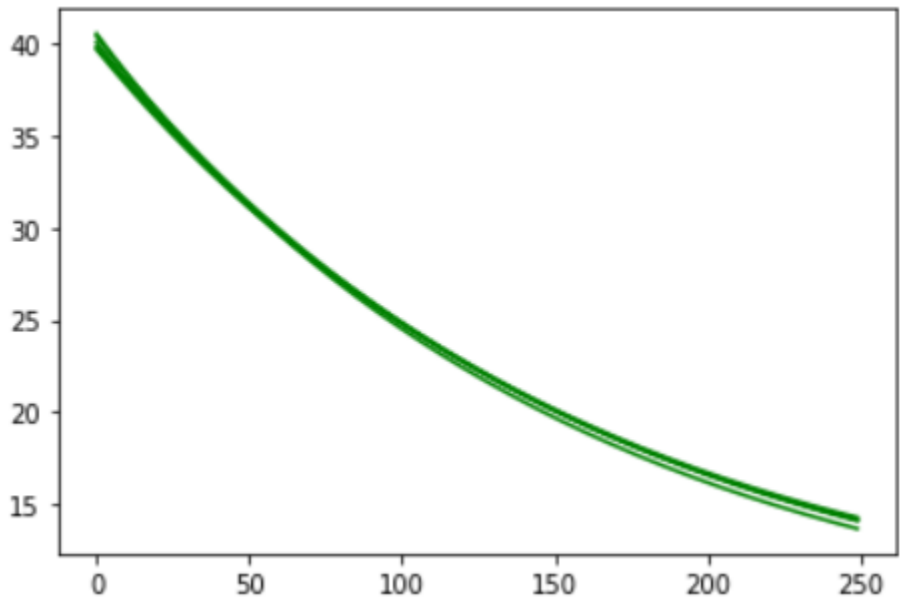


Testing RMSE for K=5 folds

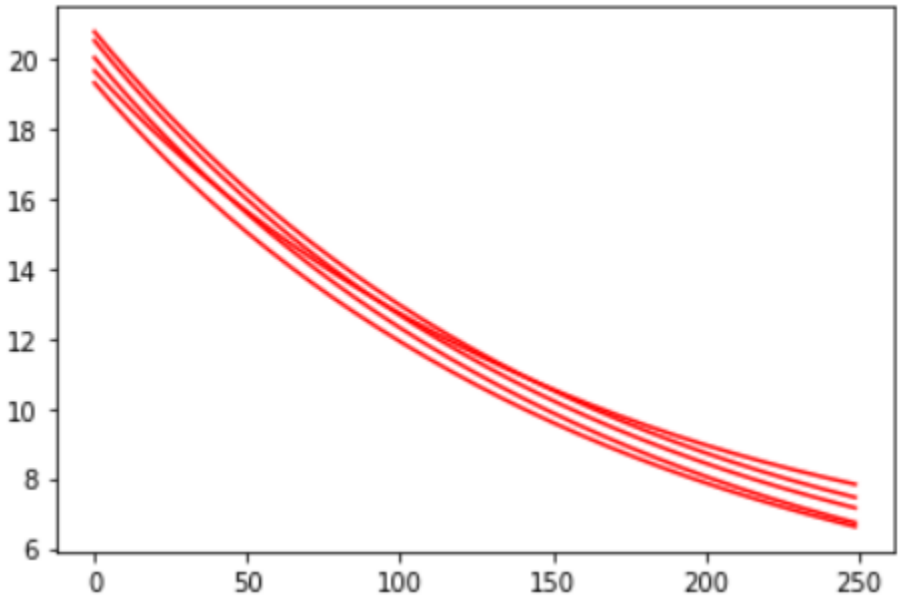


Ridge Regularisation

Best value for lambda = 0.01
Training RMSE for K=5 folds



Testing RMSE for K=5 folds



Part D

Optimal weights for the five folds were calculated using the normal equation for K=5 folds.

```
optimal w:  [[37.95420236]
 [ 1.54779742]
 [-3.13595461]
 [-5.71924389]
 [ 3.48461246]
 [ 2.77305268]
 [-0.28587288]]
```

```
optimal w:  [[37.83169413]
 [ 1.72541862]
 [-3.04887824]
 [-5.62170708]
 [ 3.17300579]
 [ 2.72987566]
 [-0.0453296 ]]
```

```
optimal w:  [[37.97031463]
 [ 1.26347703]
 [-3.16913675]
 [-6.05539978]
 [ 3.26232353]
 [ 2.47369292]
 [-0.27897062]]
```

```
optimal w:  [[38.06431501]
 [ 1.29715828]
 [-3.41891693]
 [-5.76454832]
 [ 3.20336447]
 [ 3.26390266]
 [-0.66724619]]
```

```
optimal w:  [[38.10111259]
 [ 1.41430806]
 [-2.60277439]
 [-5.08726653]
 [ 3.49876741]
 [ 2.80973873]
 [ 0.39938718]]
```

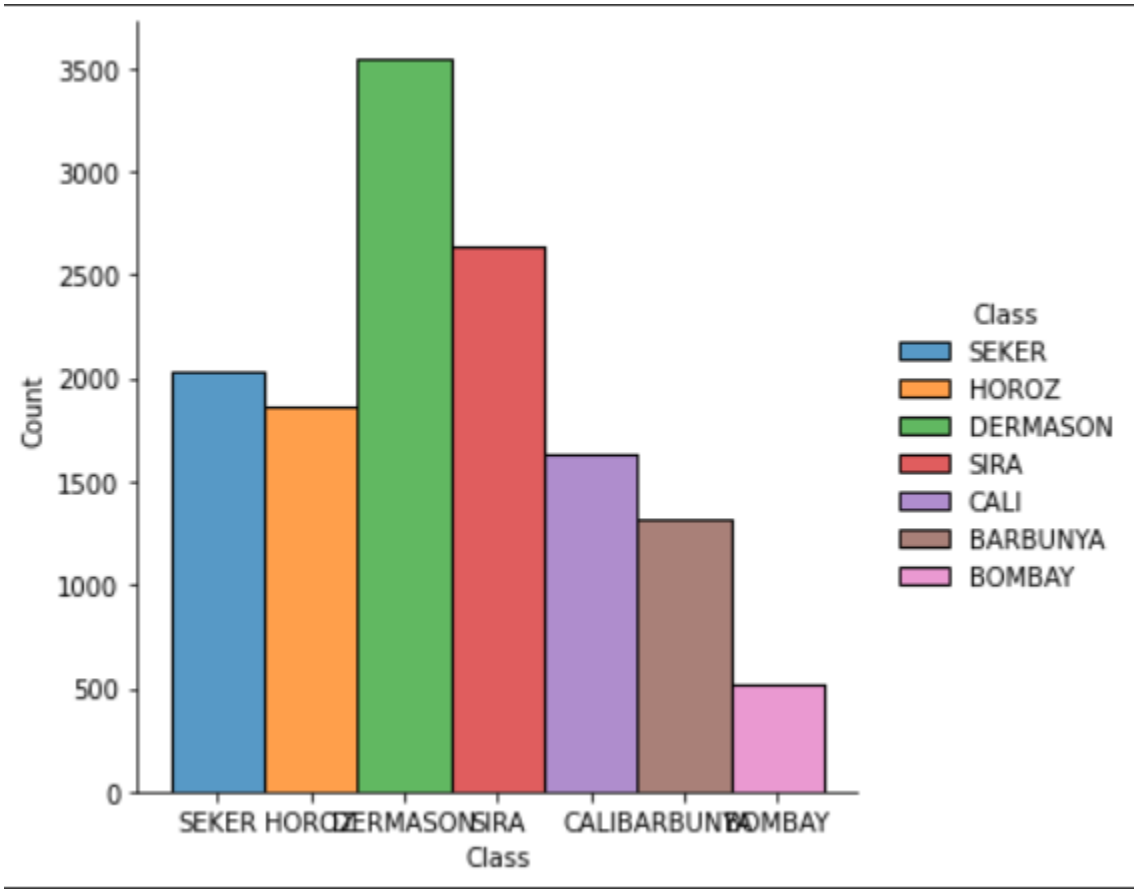
Mean RMSE =

```
[[7.14780641]
 [6.72598242]
 [6.61139442]
 [7.44873727]
 [7.82236334]]
7.15125677258799
```

Section C

Part A

From the displot, we infer that Bombay occurs the least number of times, whereas Dermason occurs the most.



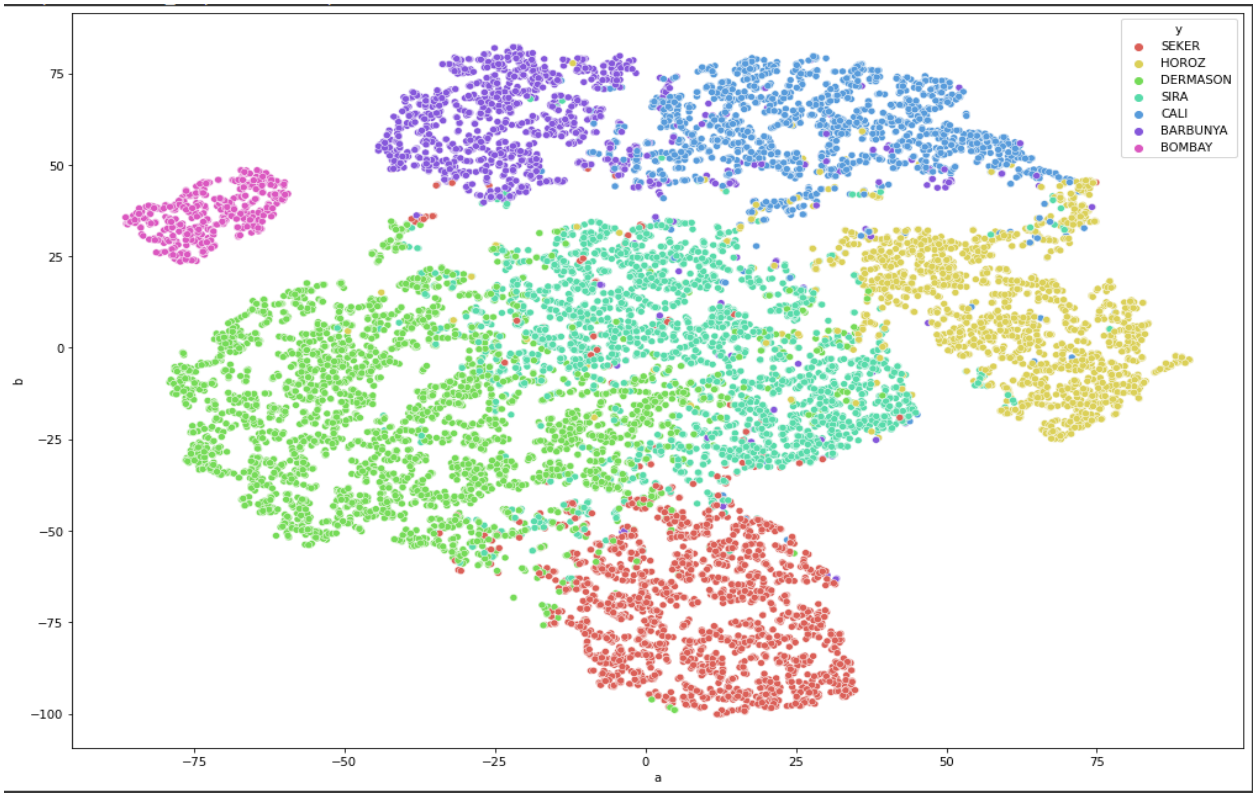
Part B

We get the following inferences from the data:

a) Some features are very skewed, and there are outliers in the distribution.
b) Bombay class differs significantly from other classes in that it has a larger Area, Perimeter and Minor/Major Axis Length.
c) Barbunaya and Cali have similar distributions.
d) There is a lot of overlap between the Dermason and Sira Classes.
e) There are a lot of linearly correlated features, as can be seen through the heatmap.
f)

Part C

The class Bombay is entirely separable from the rest of the classes. The rest of the classes have some amount of separation among them.



Part D

Gaussian Naive Bayes is for a continuous feature dataset, whereas Bernoulli Naive Bayes is for a binary feature dataset. This is why Gaussian NB gives better accuracy than Bernoulli NB.

```
Gaussian Naive Bayes model accuracy(in %): 89.34998163789938
Gaussian Naive Bayes model precision(in %): 89.44691361783833
Gaussian Naive Bayes model recall(in %): 89.34998163789938
```

```
Bernoulli Naive Bayes model accuracy(in %): 71.31839882482555
Bernoulli Naive Bayes model precision(in %): 72.61726000532246
Bernoulli Naive Bayes model recall(in %): 71.31839882482555
```

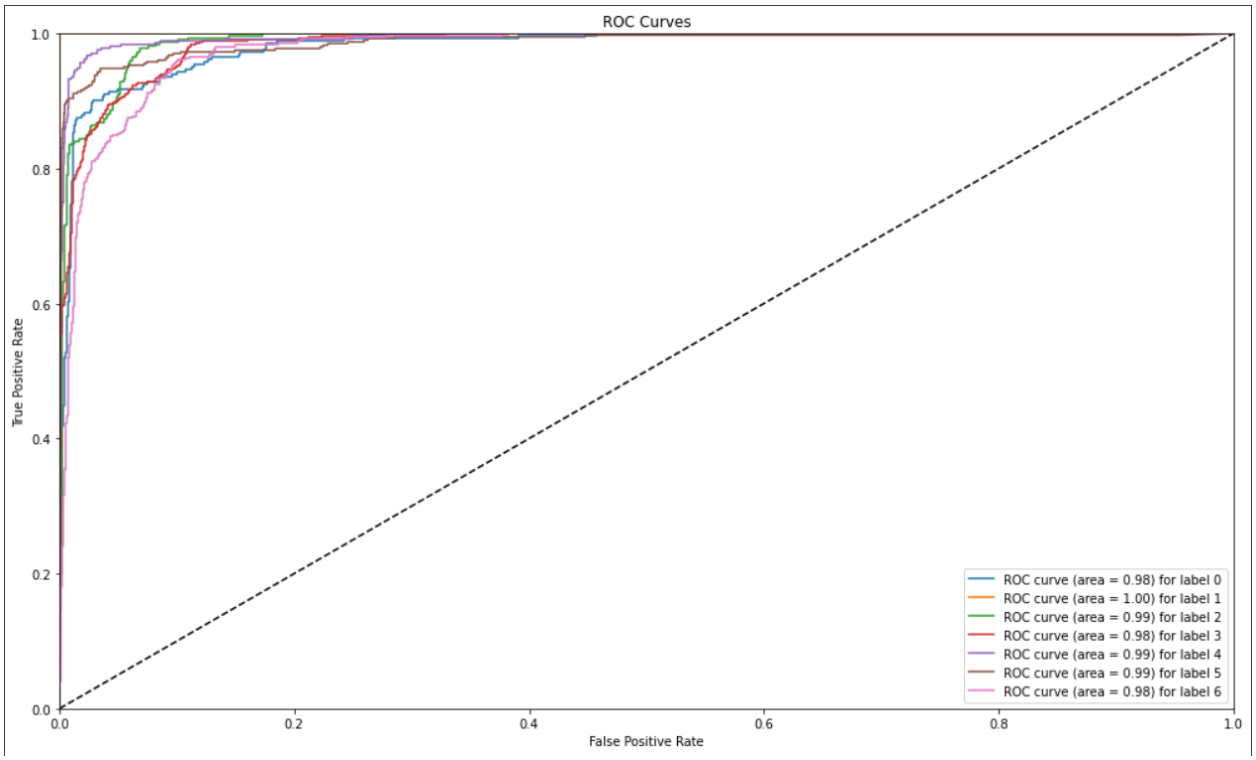
Part E

PCA is changing due to a change in number of components. PCA reduces the dataset's dimensionality, resulting in different accuracy and other metric scores.

```
PCA model accuracy(in %) for n = 4: 87.0062753783684
PCA model precision(in %) for n = 4: 87.16021324806914
PCA model recall(in %) for n = 4: 87.0062753783684
PCA model f1 score(in %) for n = 4: 86.90930051552206

PCA model accuracy(in %) for n = 6: 89.11037283130307
PCA model precision(in %) for n = 6: 89.53578296026893
PCA model recall(in %) for n = 6: 89.11037283130307
PCA model f1 score(in %) for n = 6: 89.19194994959248

PCA model accuracy(in %) for n = 8: 89.03654485049833
PCA model precision(in %) for n = 8: 89.72577457079089
PCA model recall(in %) for n = 8: 89.03654485049833
PCA model f1 score(in %) for n = 8: 89.14878509619865

PCA model accuracy(in %) for n = 10: 88.33517903285345
PCA model precision(in %) for n = 10: 89.5965329244226
PCA model recall(in %) for n = 10: 88.33517903285345
PCA model f1 score(in %) for n = 10: 88.46856405249926

PCA model accuracy(in %) for n = 12: 88.26135105204872
PCA model precision(in %) for n = 12: 89.48966576549992
PCA model recall(in %) for n = 12: 88.26135105204872
PCA model f1 score(in %) for n = 12: 88.38576161812856
```

Part F

The following curve shows high AUC scores for the classes. It represents the model's high performance at distinguishing between the positive and the negative classes.



Part G

The parameter “multi_class” is set to **“multinomial”** instead of “ovr” as it has higher accuracy among the two models.
The accuracy of Logistic Regression with the label “multinomial” is slightly higher than the accuracy of the Gaussian Naive Bayes model.