Shyama Sastha Krishnamoorthy Srinivasan
PhD21022
shyamas@iiitd.ac.in

ECE563: Machine Learning W2022

Assignment: Linear / Logistic Regression

Report

02/03/2022

Contents

PROGRAMMING QUESTIONS

0.1 Basic operations + Data visualization

0.1.1 IRIS Dataset

(a) Loading the dataset

```
# Reading in the IRIS dataset - Q.1.1.a.
iris_df = pd.read_csv('iris.csv')
iris_df = iris_df.drop("Id", axis=1)
```

(b) Column information

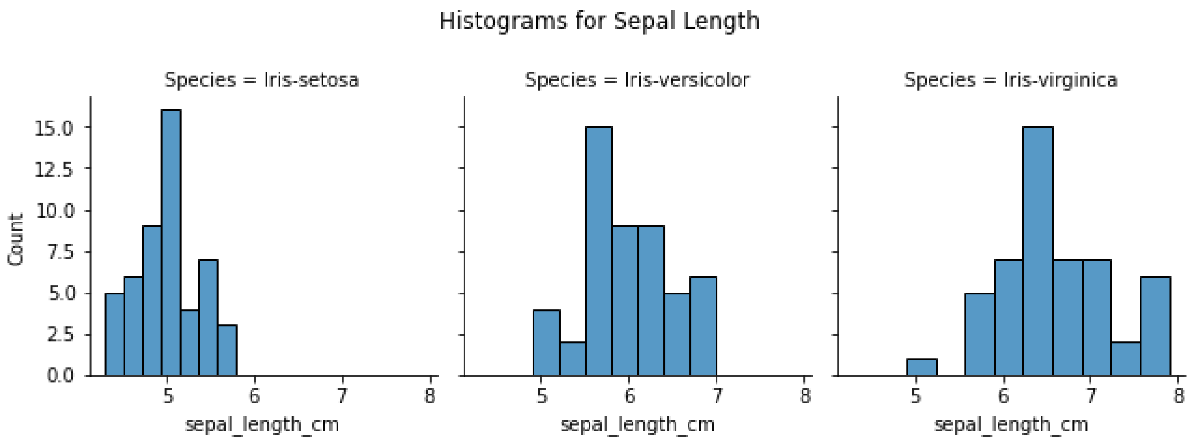
```
# Printing the column information - Q.1.1.b
iris_df.info()
```

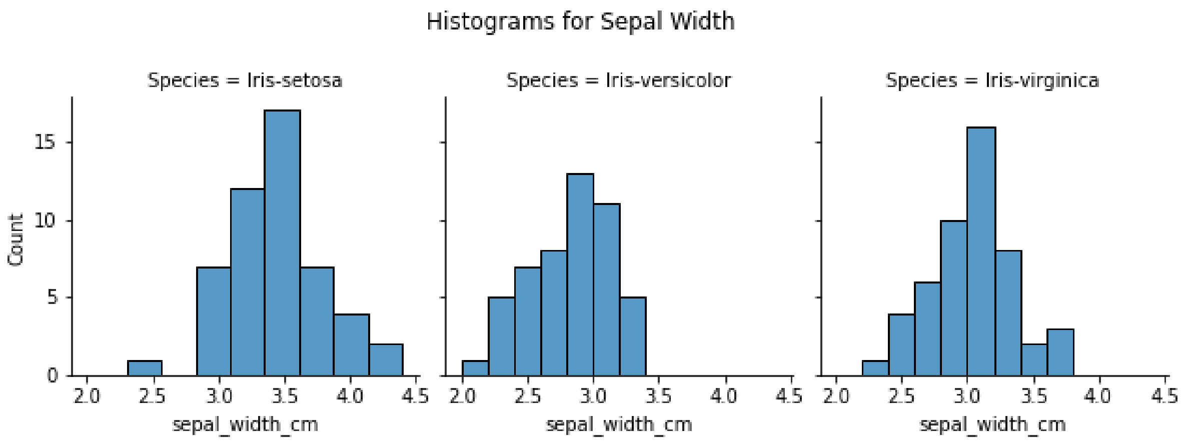
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   sepal_length_cm  150 non-null    float64
 1   sepal_width_cm   150 non-null    float64
 2   petal_length_cm  150 non-null    float64
 3   petal_width_cm   150 non-null    float64
 4   Species          150 non-null    object
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
```

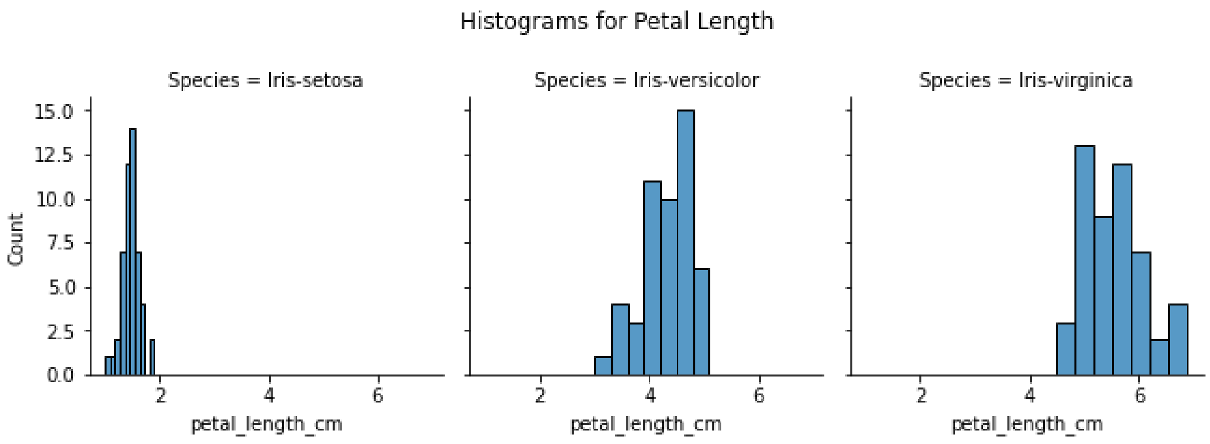
```
# Printing column metrics
iris_df.describe().T
```

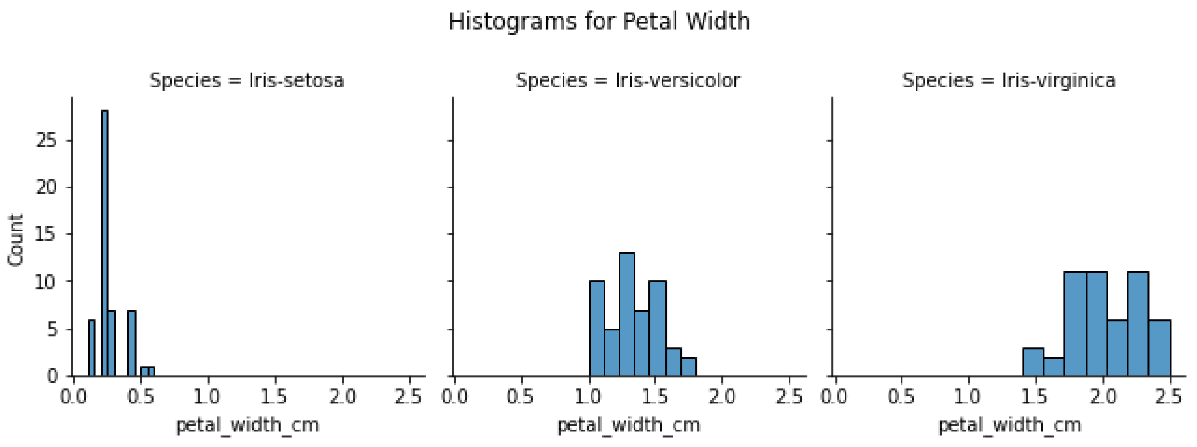
	count	mean	std	min	25%	50%	75%	max
sepal_length_cm	150.0	5.843333	0.828066	4.3	5.1	5.80	6.4	7.9
sepal_width_cm	150.0	3.054000	0.433594	2.0	2.8	3.00	3.3	4.4
petal_length_cm	150.0	3.758667	1.764420	1.0	1.6	4.35	5.1	6.9
petal_width_cm	150.0	1.198667	0.763161	0.1	0.3	1.30	1.8	2.5

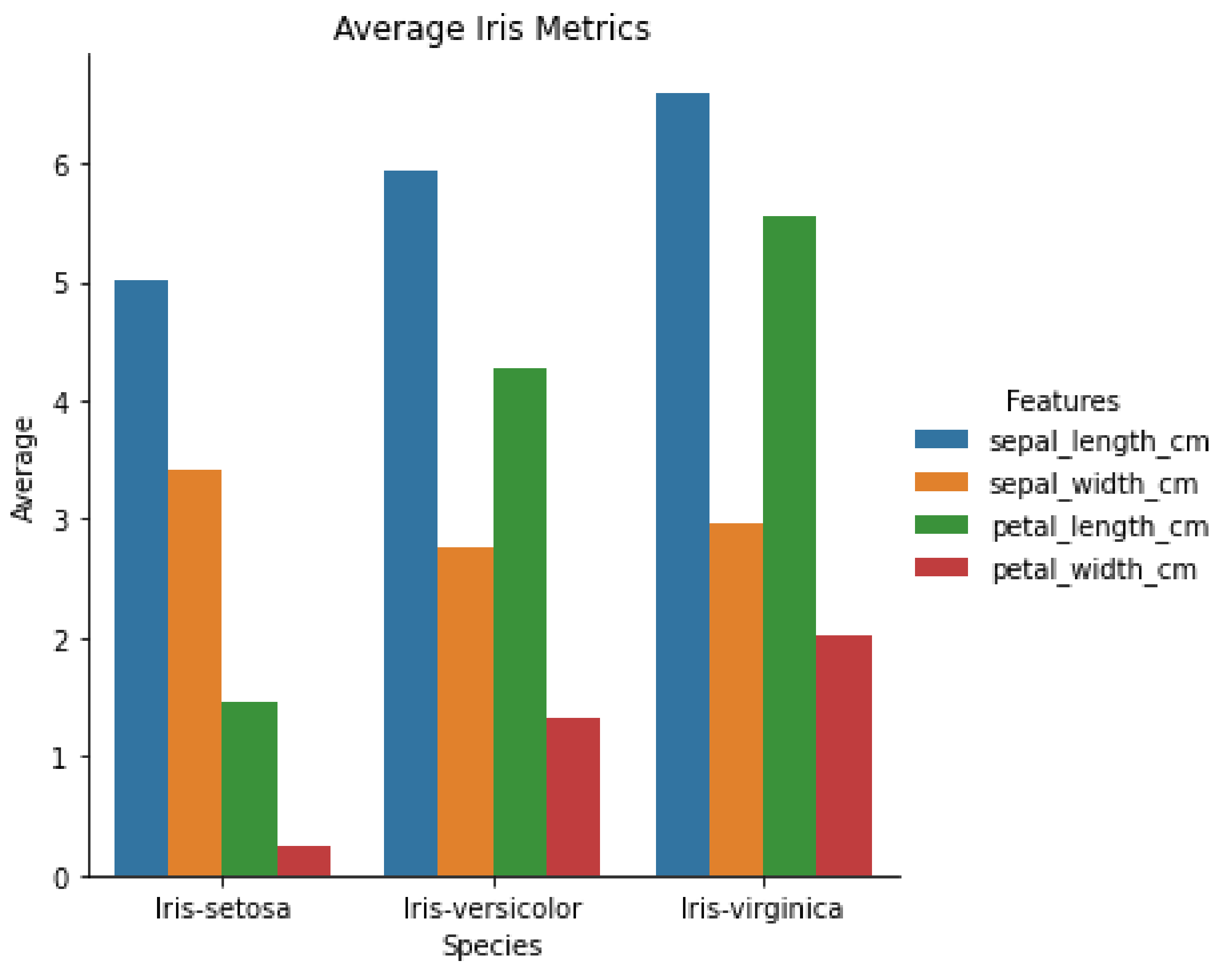
(c) Plotting Histograms and Bar plots











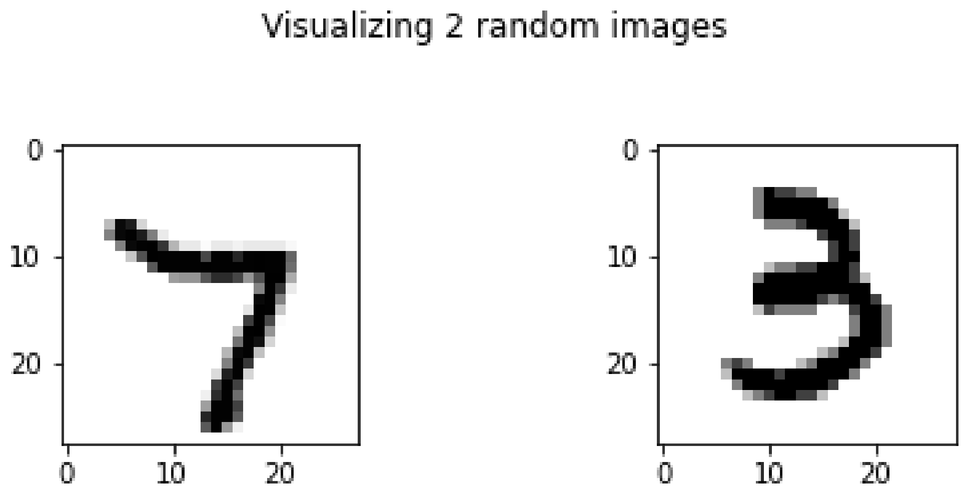
0.1.2 MNIST Dataset

(a) Loading the dataset

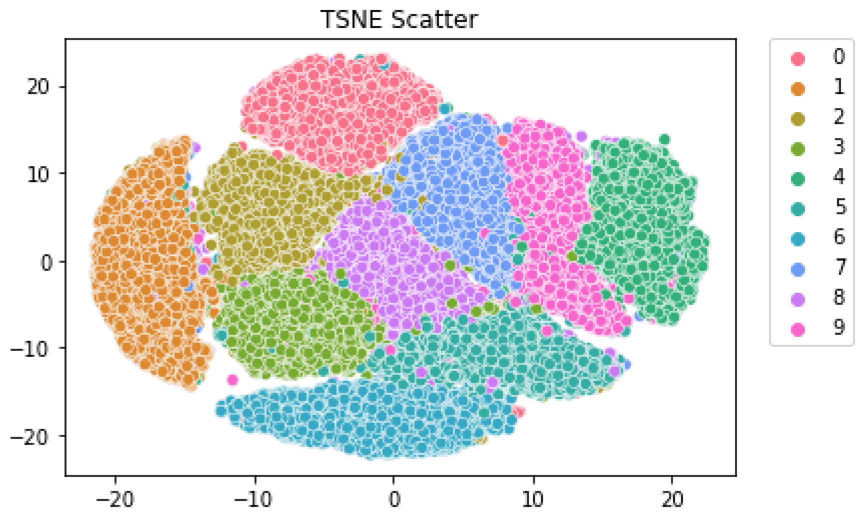
```
# Loading training data
mnist_train_images = idx2numpy.convert_from_file('train-images.idx3-ubyte')
```

```
# Loading testing data
mnist_test_images = idx2numpy.convert_from_file('t10k-images.idx3-ubyte')
```

(b) Visualizing Images



(c) Performing TSNE for 500 iterations and plotting the Scater plot

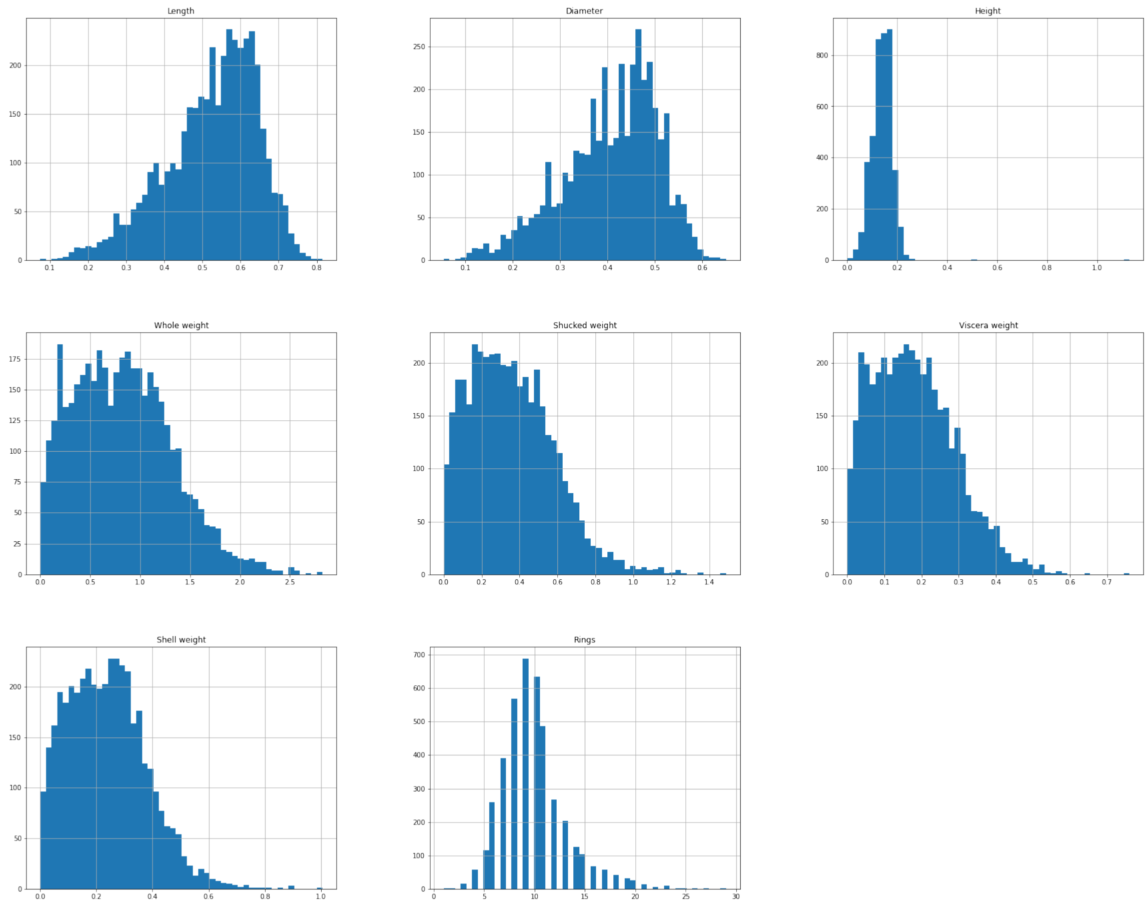


Observations: After running the tSNE for 500 iterations on the entire dataset, it can be inferred that there is enough information for the separability of the labels even when the dimension is reduced to two.

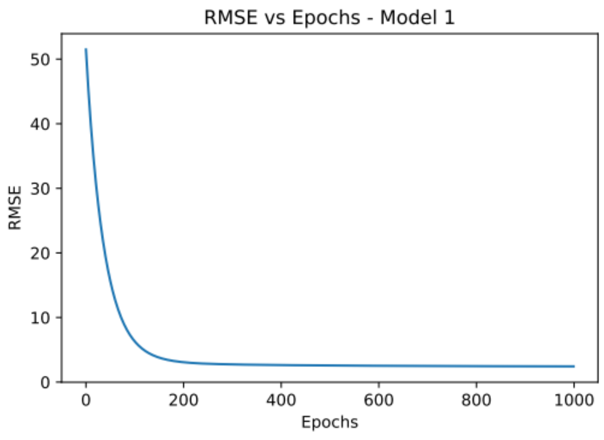
0.2 Linear Regression - Abalone Dataset

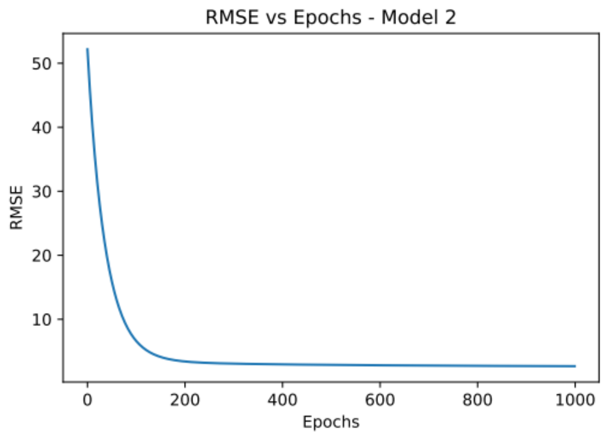
Implementation of Linear Regression with gradient descent from scratch on Abalone dataset

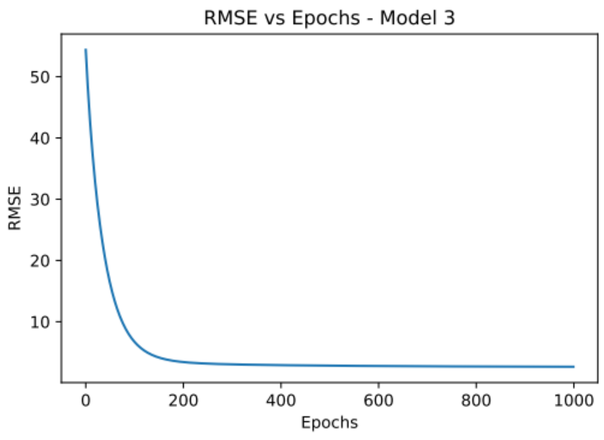
Histograms for all continuous columns:

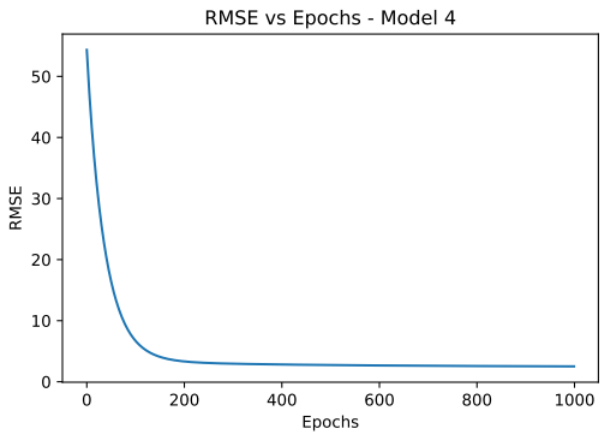


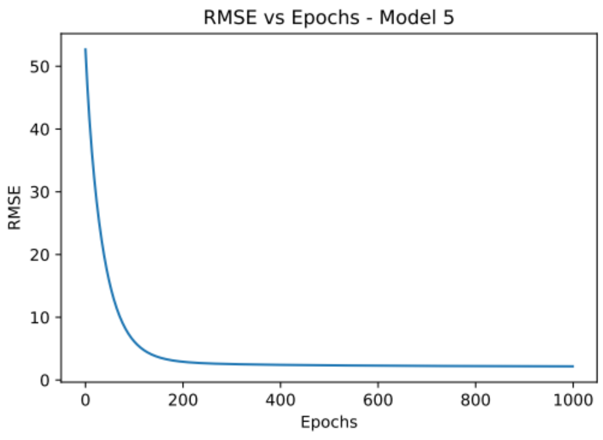
(a) RSME vs Epochs plots for Linear Regression



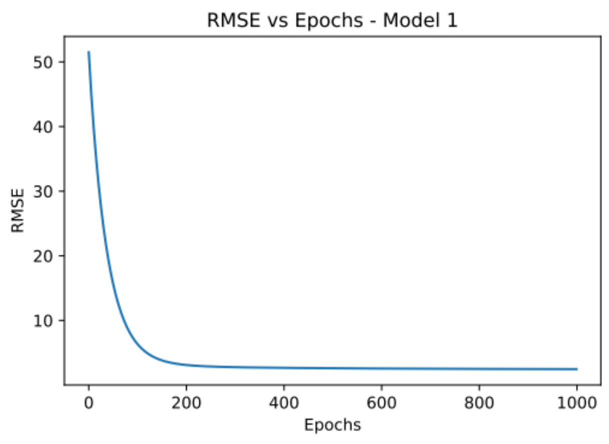


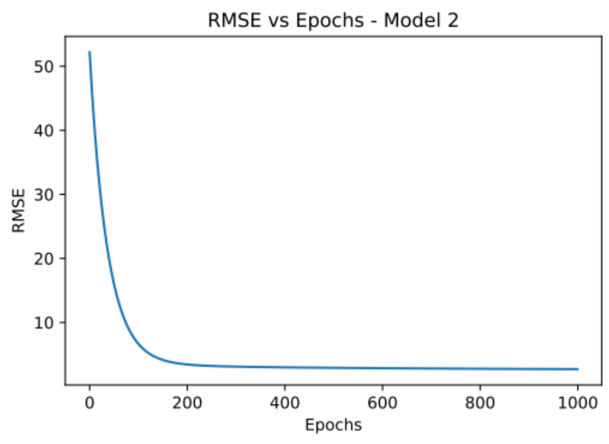


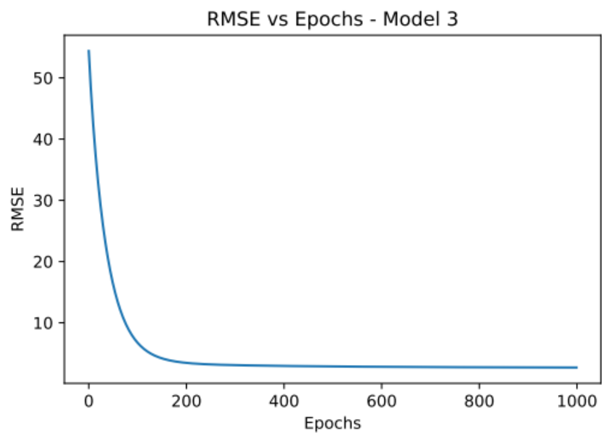


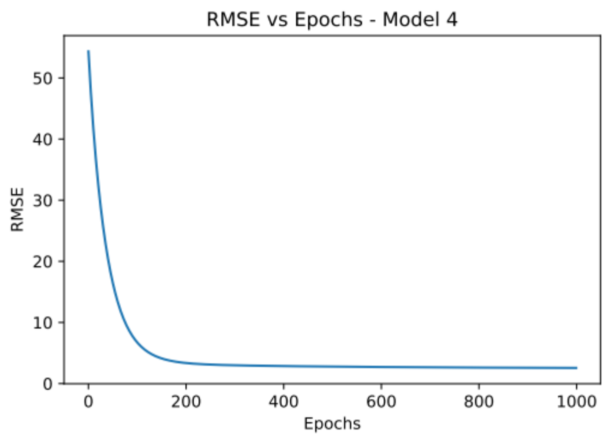


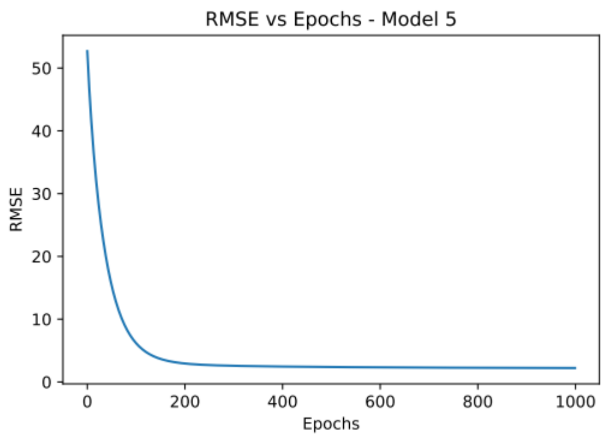
(b) RSME vs Epoch for L1 Regularized Regression





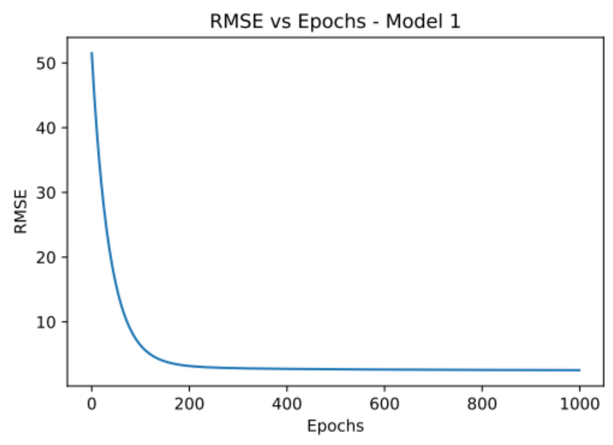


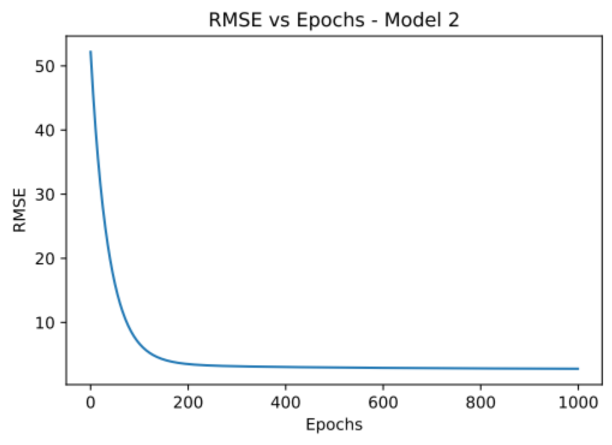


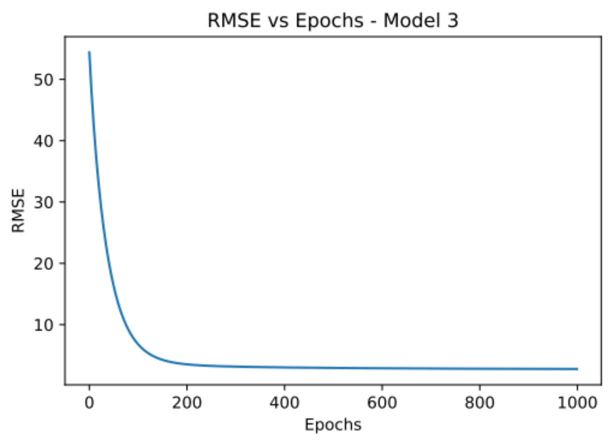


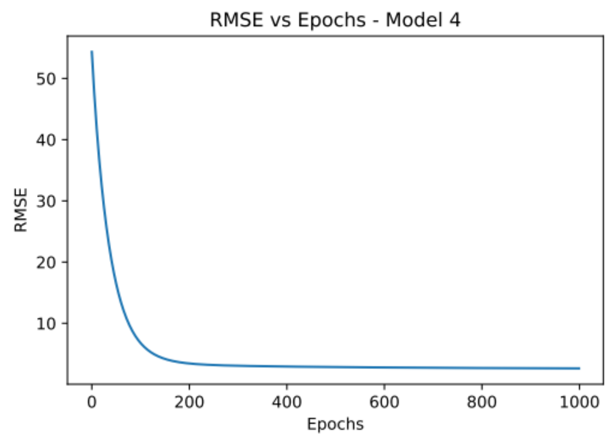
The best lasso parameter came out to be 0.01 with an RMSE of 2.2177746167864254.

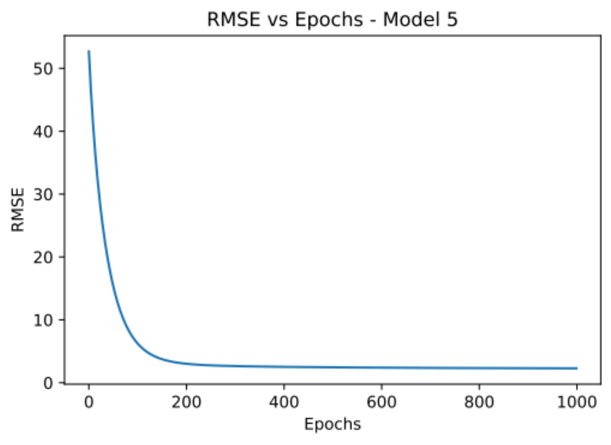
RSME vs Epoch for L2 Regularized Regression





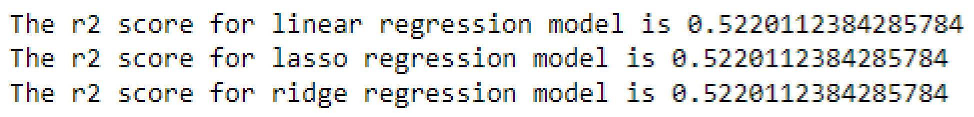






The best ridge parameter came out to be 0.01 with an RMSE of 2.2878374686981324.

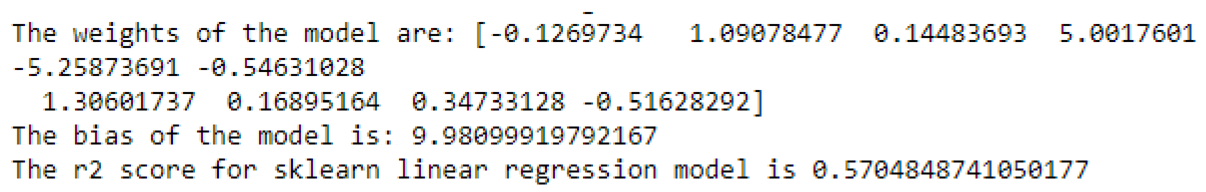
(c) Best Models Test set Accuracy:



```
The r2 score for linear regression model is 0.5220112384285784
The r2 score for lasso regression model is 0.5220112384285784
The r2 score for ridge regression model is 0.5220112384285784
```

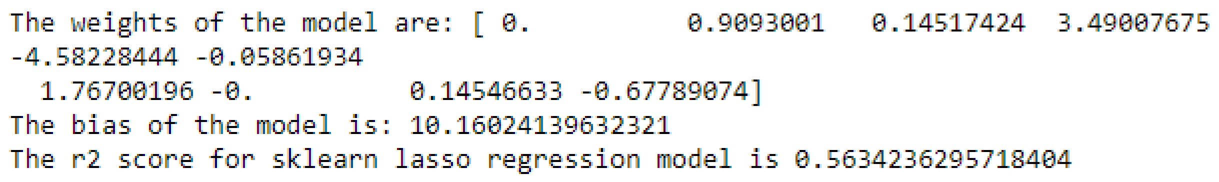
(d) Best Models for sklearn implementation of simple, L1 regularized and L2 regularized

Simple sklearn Implementation



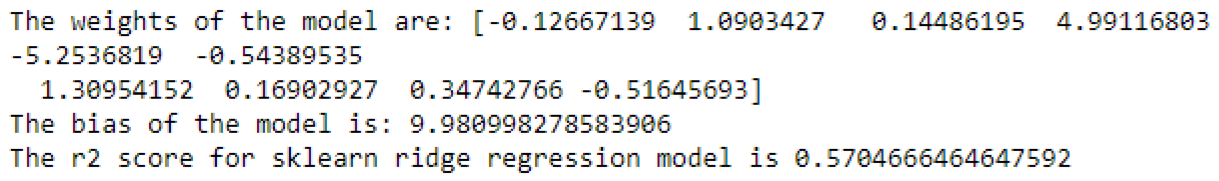
```
The weights of the model are: [-0.1269734   1.09078477  0.14483693  5.0017601
-5.25873691 -0.54631028
  1.30601737  0.16895164  0.34733128 -0.51628292]
The bias of the model is: 9.98099919792167
The r2 score for sklearn linear regression model is 0.5704848741050177
```

L1 sklearn Implementation



```
The weights of the model are: [ 0.          0.9093001   0.14517424  3.49007675
-4.58228444 -0.05861934
  1.76700196 -0.          0.14546633 -0.67789074]
The bias of the model is: 10.16024139632321
The r2 score for sklearn lasso regression model is 0.5634236295718404
```

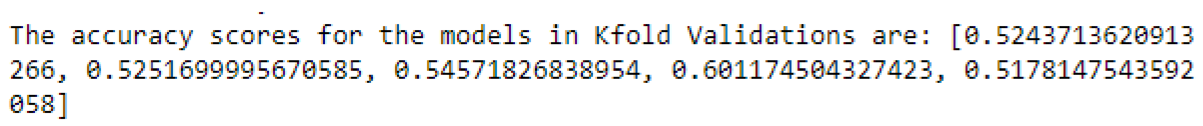
L2 sklearn Implementation



```
The weights of the model are: [-0.12667139  1.0903427   0.14486195  4.99116803
-5.2536819  -0.54389535
  1.30954152  0.16902927  0.34742766 -0.51645693]
The bias of the model is: 9.980998278583906
The r2 score for sklearn ridge regression model is 0.5704666464647592
```

Observations: The test accuracy (r2score) of the sklearn implementations came out to be slightly higher than the ”from the scratch” implementations. Which might infer that either the learning rate, regularization parameters or the gradient updates could have been tuned better in the ”from the scratch” implementations.

(e) The accuracy metrics for closed form implementation of Kfold are:

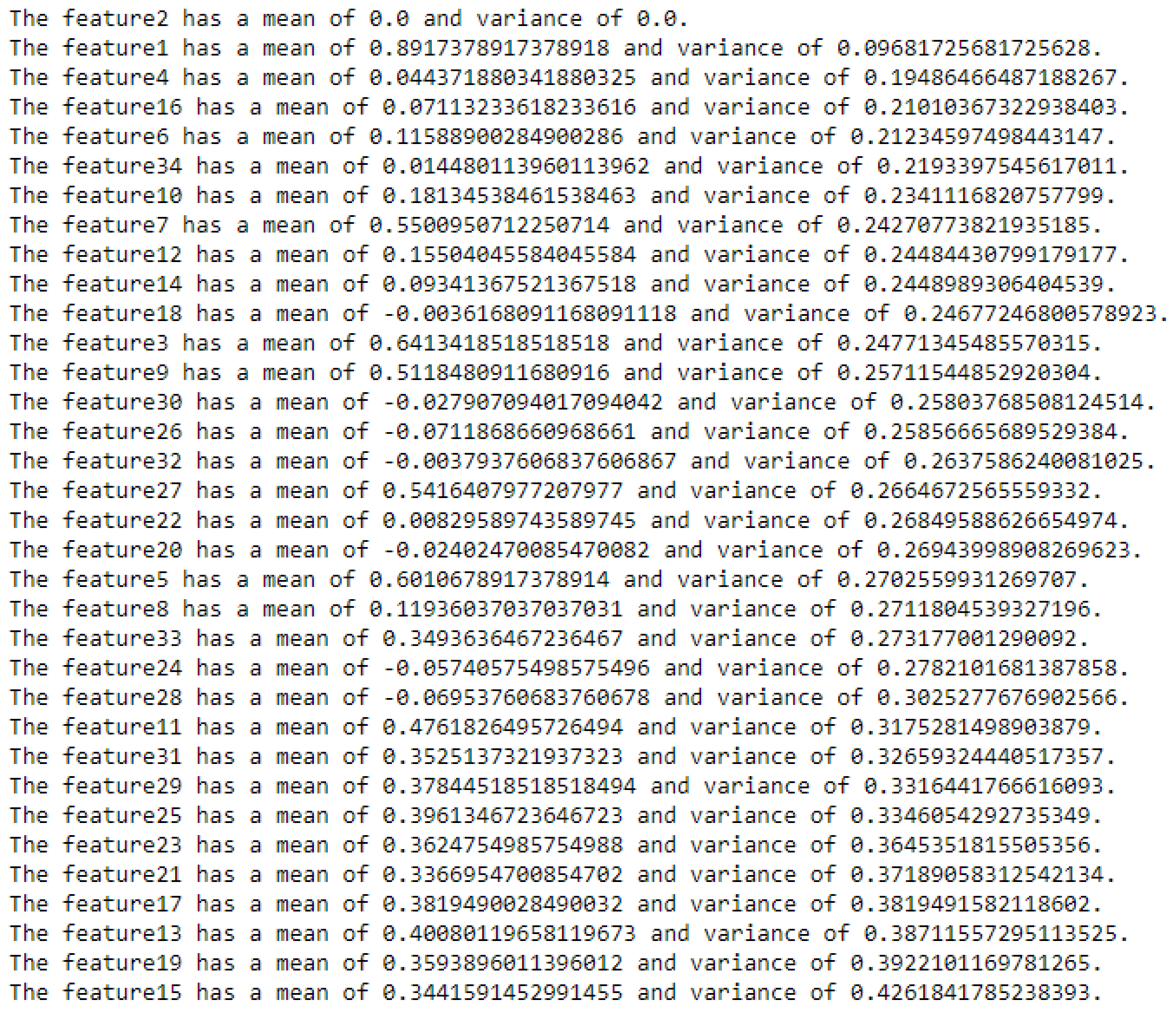


```
The accuracy scores for the models in Kfold Validations are: [0.5243713620913
266, 0.5251699995670585, 0.54571826838954, 0.601174504327423, 0.5178147543592
058]
```

0.3 Logistic Regression

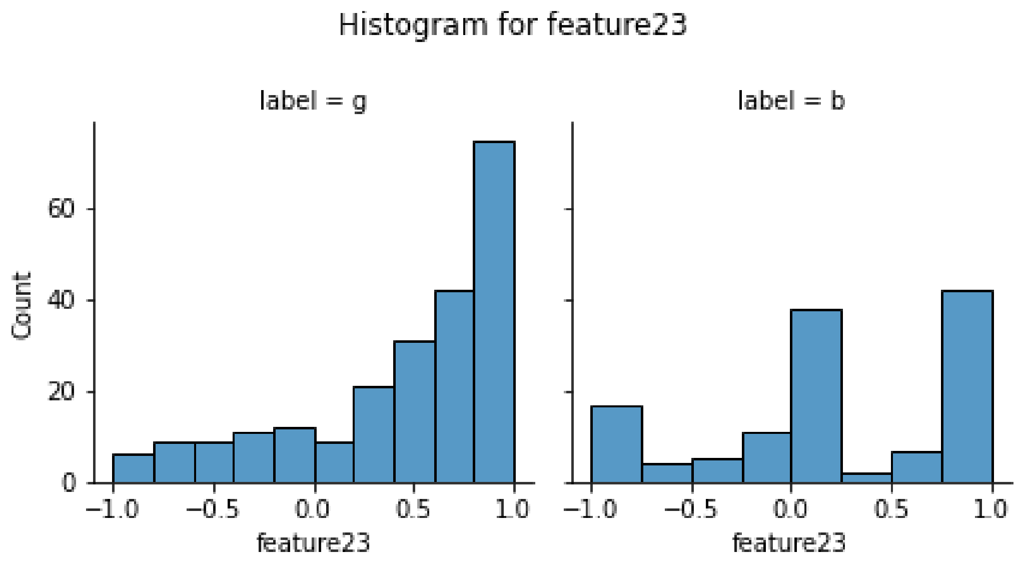
0.3.1 UCI Ionosphere Dataset

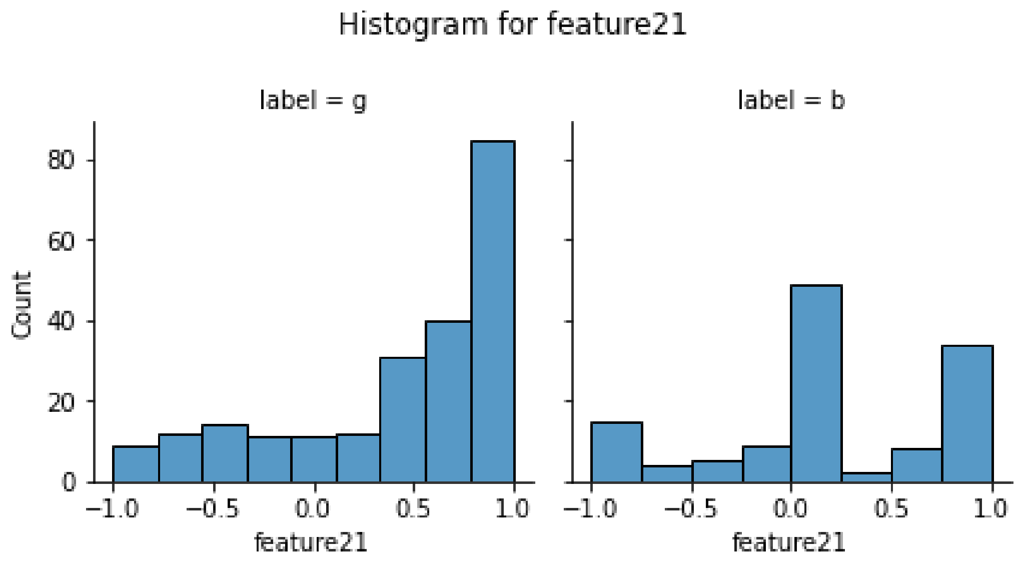
Means and Variances:

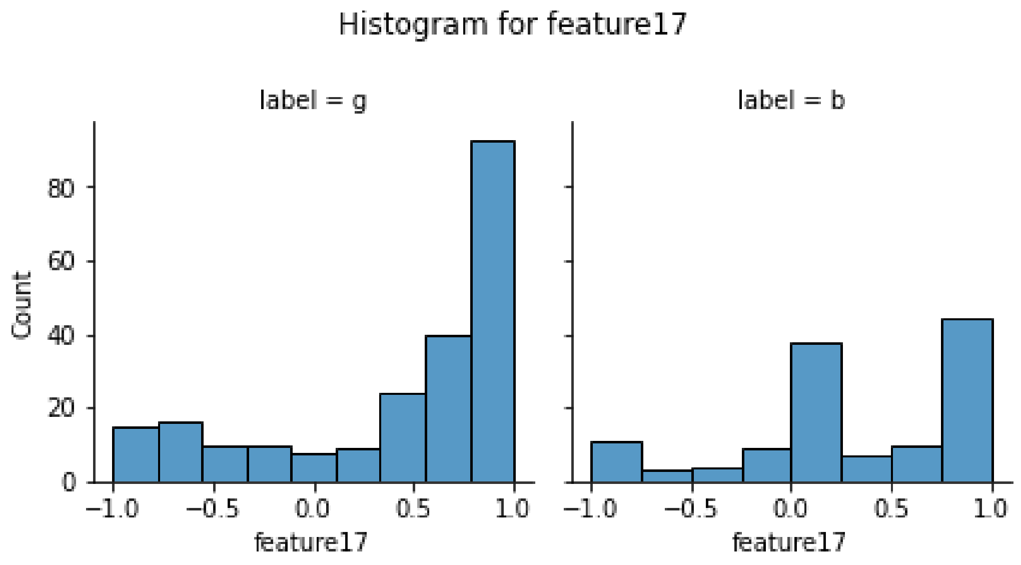


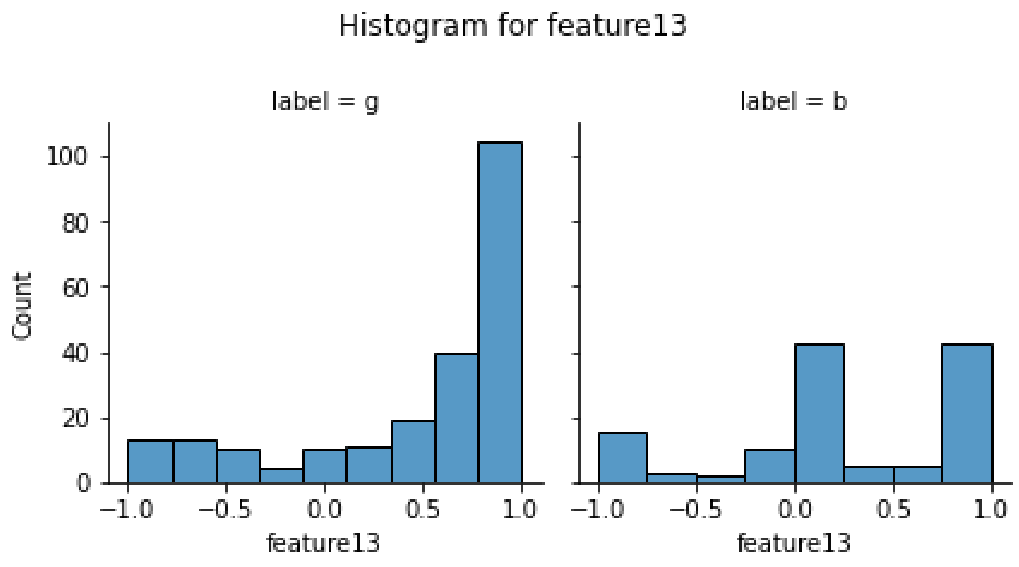
```
The feature2 has a mean of 0.0 and variance of 0.0.
The feature1 has a mean of 0.8917378917378918 and variance of 0.09681725681725628.
The feature4 has a mean of 0.044371880341880325 and variance of 0.19486466487188267.
The feature16 has a mean of 0.07113233618233616 and variance of 0.21010367322938403.
The feature6 has a mean of 0.11588900284900286 and variance of 0.21234597498443147.
The feature34 has a mean of 0.014480113960113962 and variance of 0.2193397545617011.
The feature10 has a mean of 0.18134538461538463 and variance of 0.2341116820757799.
The feature7 has a mean of 0.5500950712250714 and variance of 0.24270773821935185.
The feature12 has a mean of 0.15504045584045584 and variance of 0.24484430799179177.
The feature14 has a mean of 0.09341367521367518 and variance of 0.2448989306404539.
The feature18 has a mean of -0.0036168091168091118 and variance of 0.24677246800578923.
The feature3 has a mean of 0.6413418518518518 and variance of 0.24771345485570315.
The feature9 has a mean of 0.5118480911680916 and variance of 0.25711544852920304.
The feature30 has a mean of -0.027907094017094042 and variance of 0.25803768508124514.
The feature26 has a mean of -0.0711868660968661 and variance of 0.25856665689529384.
The feature32 has a mean of -0.003793760683760867 and variance of 0.2637586240081025.
The feature27 has a mean of 0.5416407977207977 and variance of 0.2664672565559332.
The feature22 has a mean of 0.00829589743589745 and variance of 0.26849588626654974.
The feature20 has a mean of -0.02402470085470082 and variance of 0.26943998908269623.
The feature5 has a mean of 0.6010678917378914 and variance of 0.2702559931269707.
The feature8 has a mean of 0.11936037037037031 and variance of 0.2711804539327196.
The feature33 has a mean of 0.3493636467236467 and variance of 0.273177001290092.
The feature24 has a mean of -0.05740575498575496 and variance of 0.2782101681387858.
The feature28 has a mean of -0.06953760683760678 and variance of 0.3025277676902566.
The feature11 has a mean of 0.4761826495726494 and variance of 0.3175281498903879.
The feature31 has a mean of 0.3525137321937323 and variance of 0.32659324440517357.
The feature29 has a mean of 0.37844518518518494 and variance of 0.3316441766616093.
The feature25 has a mean of 0.3961346723646723 and variance of 0.3346054292735349.
The feature23 has a mean of 0.3624754985754988 and variance of 0.3645351815505356.
The feature21 has a mean of 0.3366954700854702 and variance of 0.37189058312542134.
The feature17 has a mean of 0.3819490028490032 and variance of 0.3819491582118602.
The feature13 has a mean of 0.40080119658119673 and variance of 0.38711557295113525.
The feature19 has a mean of 0.3593896011396012 and variance of 0.3922101169781265.
The feature15 has a mean of 0.3441591452991455 and variance of 0.4261841785238393.
```

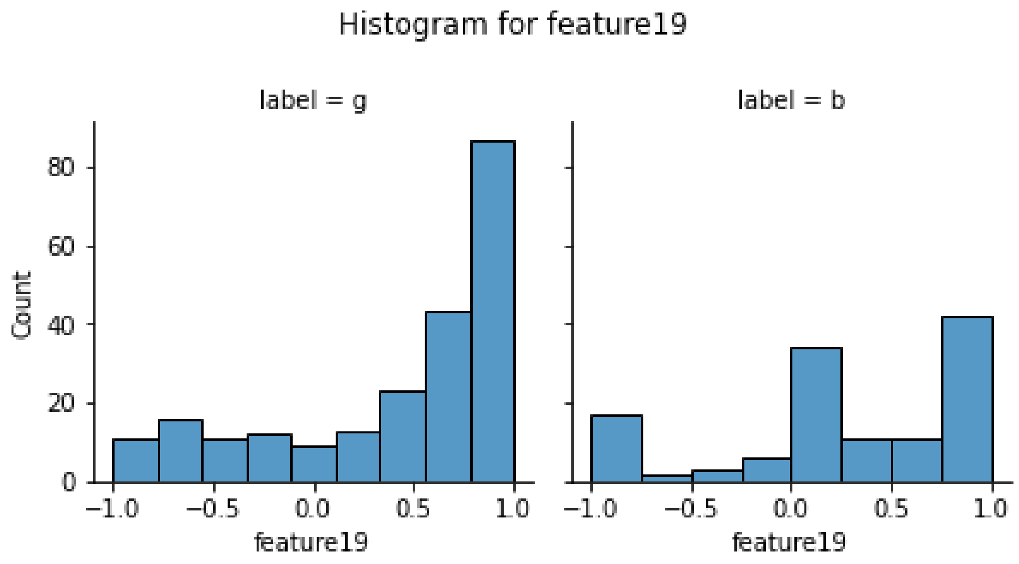
Histograms of top 5 Features:



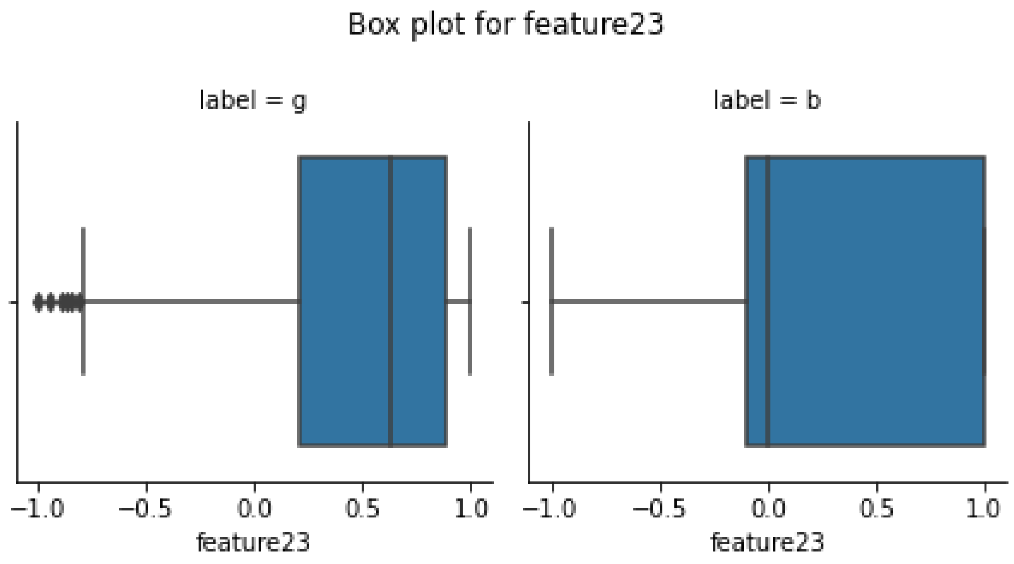


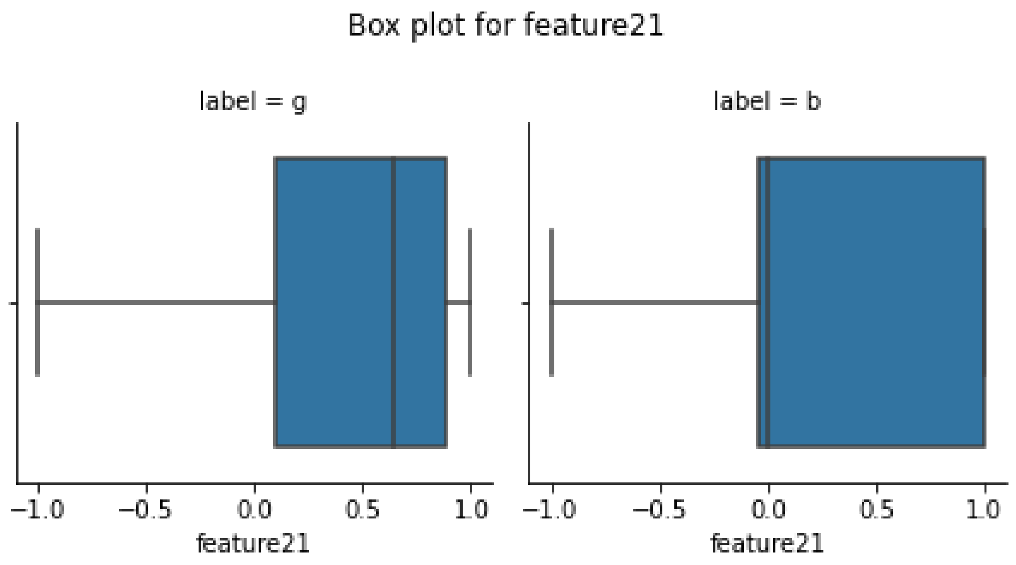


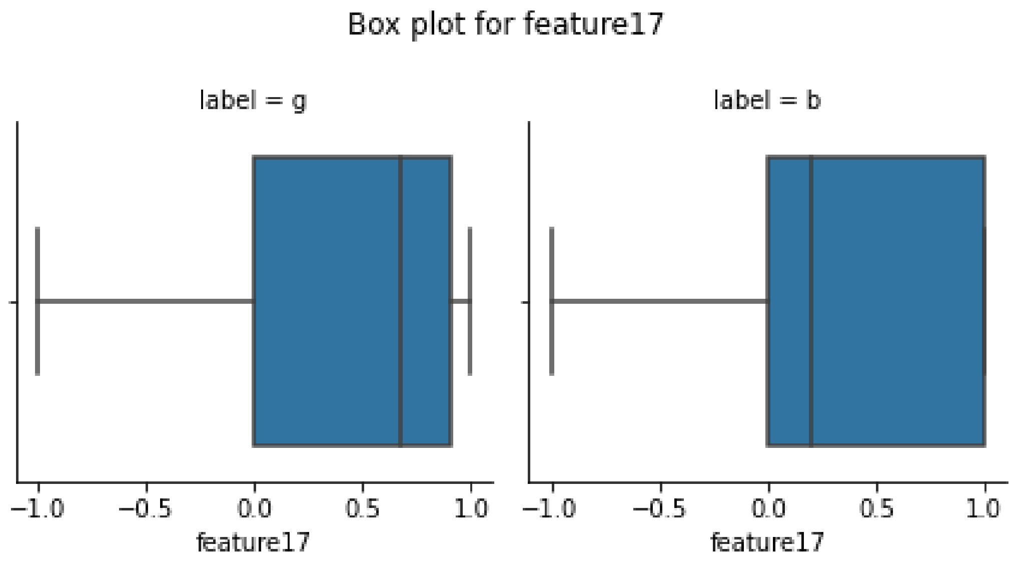


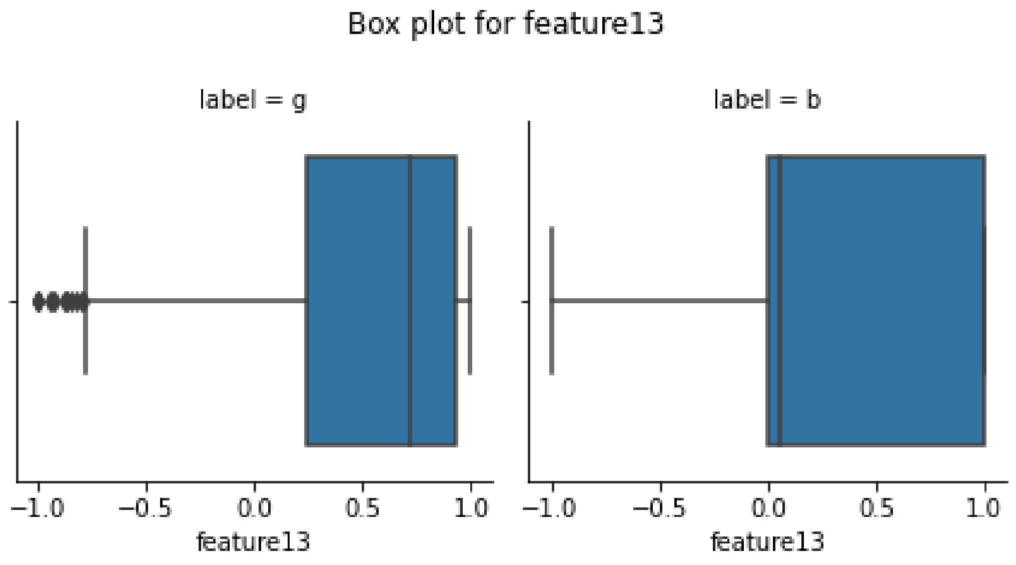


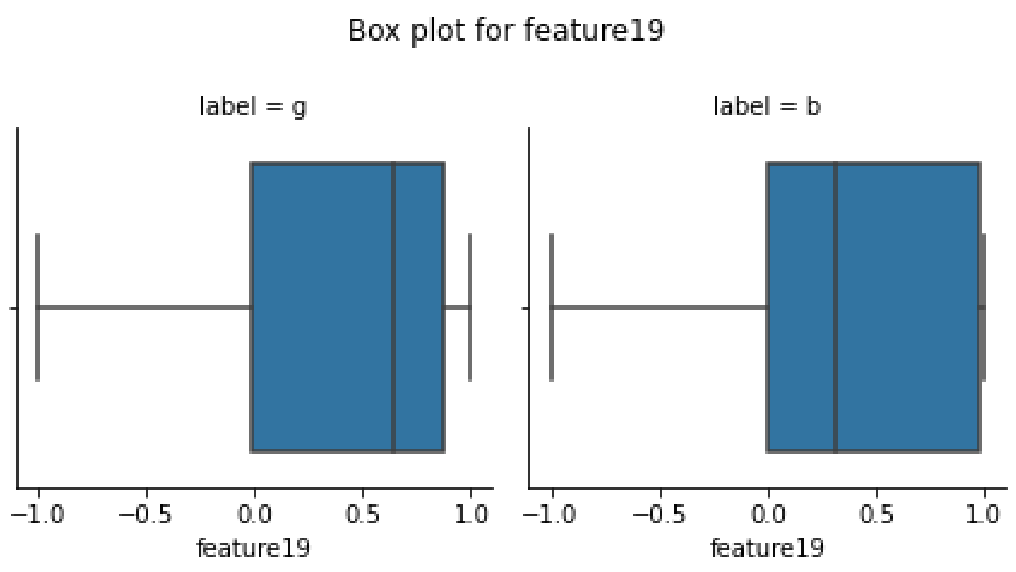
Box plots of top 5 Features:




Box plot for feature21
label = g
label = b
−1.0
−0.5
0.0
0.5
1.0
feature21
feature21


Box plot for feature17
label = g
label = b
−1.0
−0.5
0.0
0.5
1.0
feature17
feature17

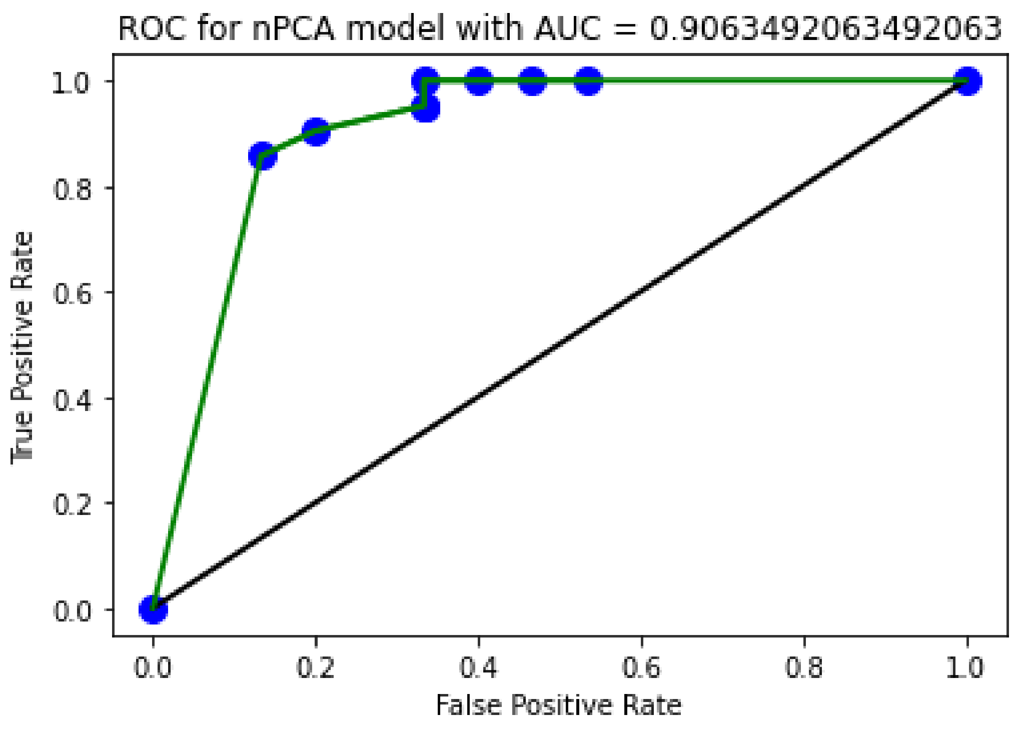

Box plot for feature13
label = g
label = b
−1.0
−0.5
0.0
0.5
1.0
feature13
feature13

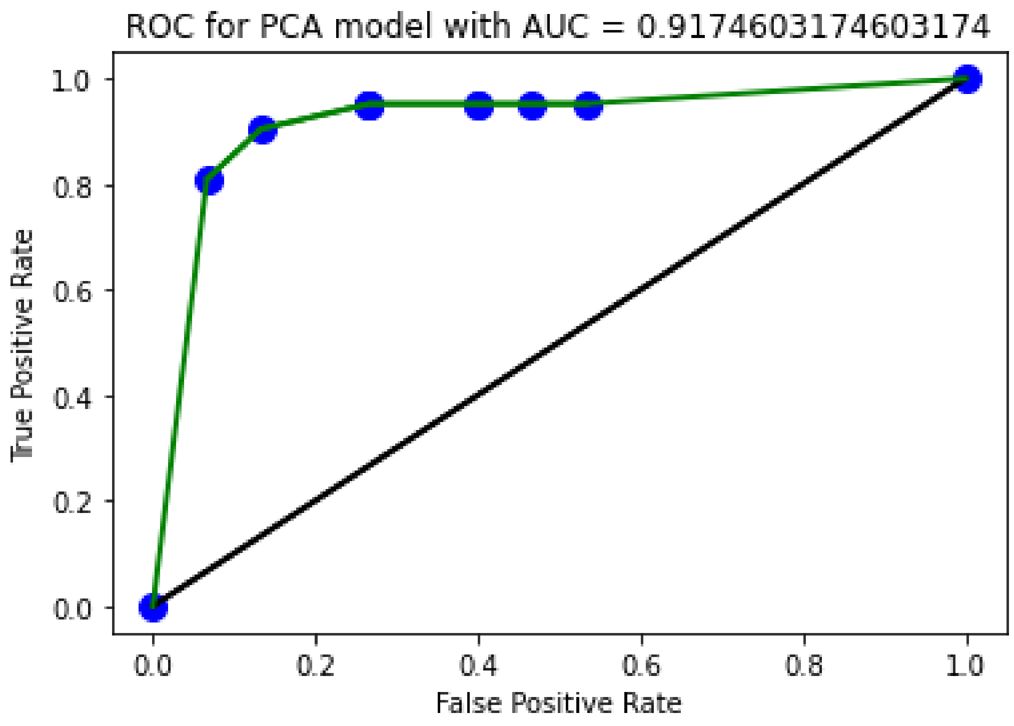


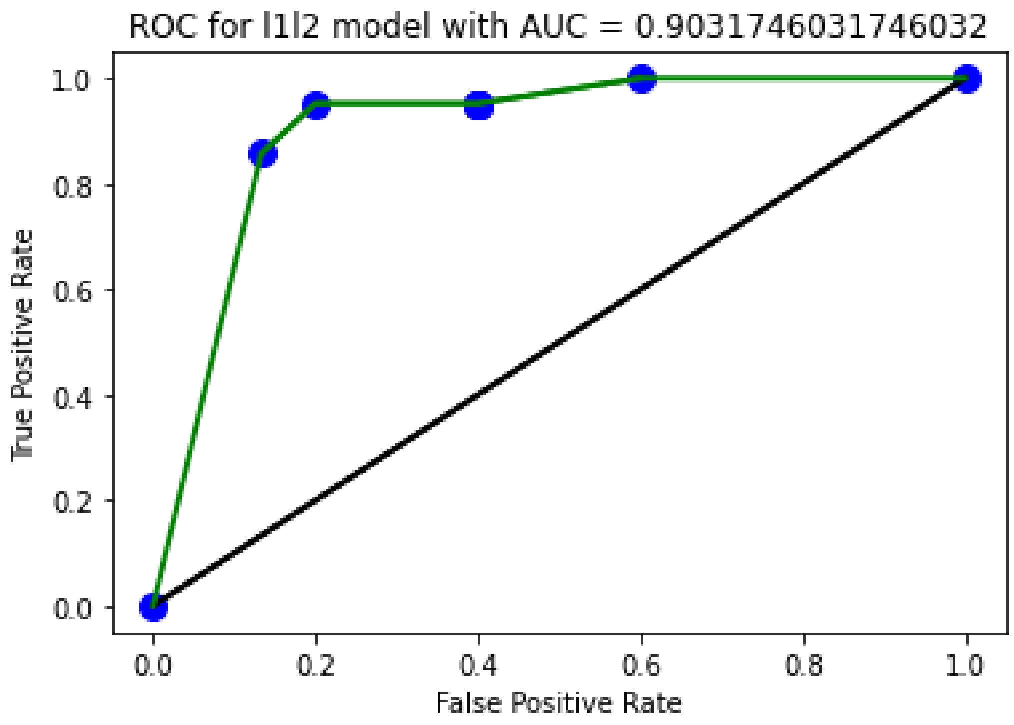
(a) and (b) Comparing results for the best PCA vs non PCA vs L1L2 models

```
The evaluation metrics of the best non PCA model on the test set:
Accuracy: 0.8333333333333334, Precision: 0.8, Recall: 0.9523809523809523, F-1 Score: 0.8695652173913043
The evaluation metrics of the best PCA model on the test set:
Accuracy: 0.8333333333333334, Precision: 0.8260869565217391, Recall: 0.9047619047619048, F-1 Score: 0.8636363636363636
The evaluation metrics of the best l1l2 model on the test set:
Accuracy: 0.8055555555555556, Precision: 0.7692307692307693, Recall: 0.9523809523809523, F-1 Score: 0.8510638297872339
```

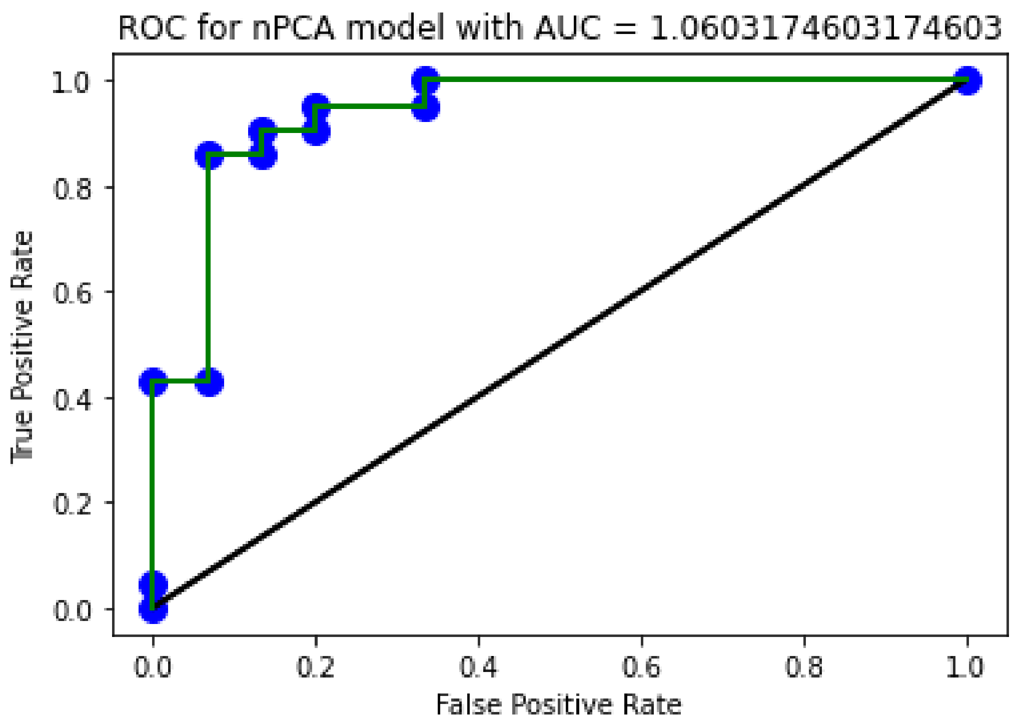
(c) From the scratch implementations of ROC-AUC curves for nonPCA, PCA, and L1L2 Models

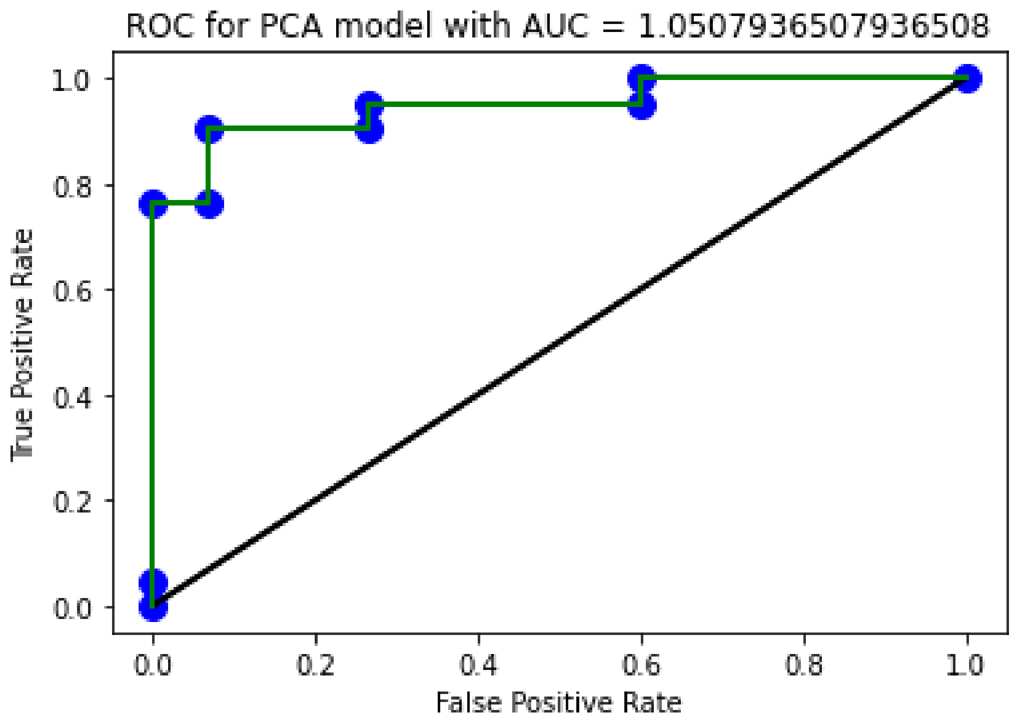


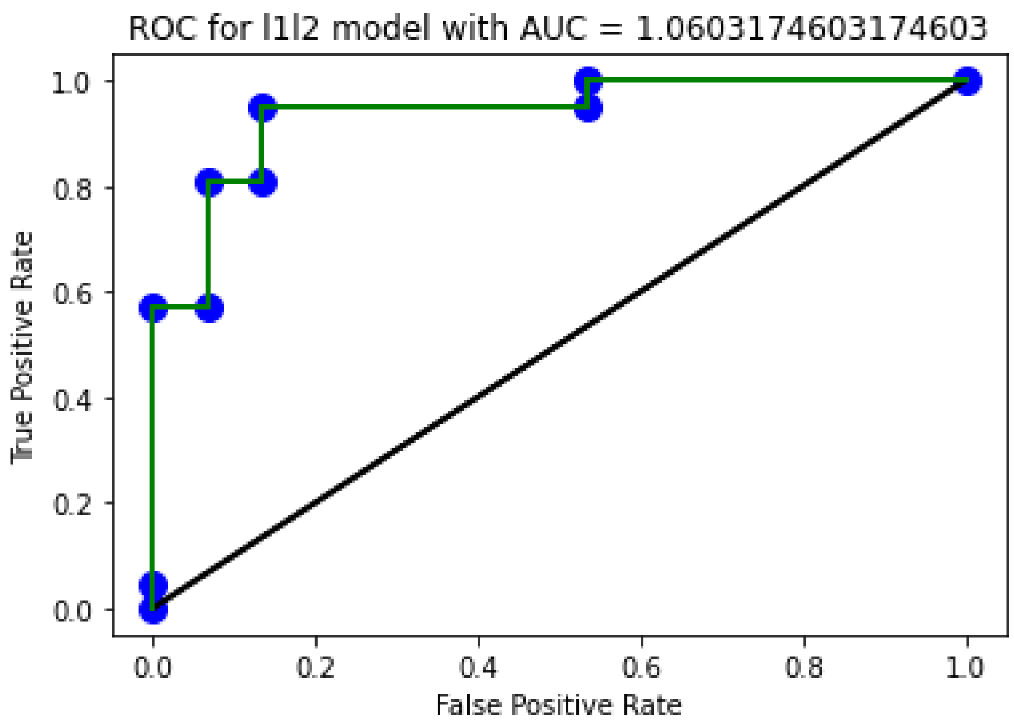

ROC for PCA model with AUC = 0.9174603174603174
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0


ROC for l1l2 model with AUC = 0.9031746031746032
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0

(d) The sklearn implementations of ROC-AUC curves for nonPCA, PCA, and L1L2 Models







Observations: Comparing the sets of each model, it can be seen that the true positive rates and false positive rates take sharper jumps in the sklearn model compared to the ”from the scratch” implementation. It is based on the choice of thresholds.

0.3.2 Multi-class Logistic Regression for the MNIST Dataset

(a) Results on OVO with simple Logistic regression and L2 regularized Logistic regression:

```
simple Logistic Regression OVO:
Accuracy: 0.9278, Precision: 0.9278, Recall: 0.9278, F-1 Score: 0.9278

l2 Logistic Regression OVO:
Accuracy: 0.9278, Precision: 0.9278, Recall: 0.9278, F-1 Score: 0.9278
```

(b) Results on OVR with L2 regularized Logistic regression:

```
Logistic Regression OVR:
Accuracy: 0.9178, Precision: 0.9178, Recall: 0.9178, F-1 Score: 0.9178
```

THEORY QUESTIONS (for everyone)

0.4 Linear Regression

1. Derive the closed form solution to the linear regression problem for the dataset $\mathcal{D} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots$
where $\mathbf{x}_i \in \mathbb{R}^d$, and $y_i \in \mathbb{R}$, for $i = 1, 2, \ldots, N$. Let $\mathbf{y} = \mathbf{X}\theta + \epsilon$ be the regression model, where $\mathbf{y}$ is an $N \times 1$ vector constructed by concatenating the target variables $y_i, i = 1, \ldots, N$, and the matrix $\mathbf{X}_{N\times d} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N]^\top$ contains the input data vectors. The regression parameters θ needs to be estimated.

Sol: From the given assumptions the Loss function of can be written as sum of the least squares of actual - predicted. Which gives,

$$Loss = J(\theta) = \textstyle\sum_{i=1}^{N} \left(w^\top x_i - y_i\right)^2$$

From matrix algebra, this can be written as $J(\theta) = \frac{1}{N}(X\theta - Y)^T(X\theta - Y)$

To find the value of θ which minimizes the loss, we differentiate $J(\theta)$ with respect to θ.

$$\frac{\partial}{\partial\theta}J(\theta) = \frac{1}{N}\frac{\partial}{\partial\theta}(X\theta - Y)^T(X\theta - Y)$$

$$= \frac{1}{N}\frac{\partial}{\partial\theta}\left(\theta^T X^T X\theta - \theta^T X^T Y - Y^T X\theta + Y^T Y\right)\}$$

$$\left(X^T X\theta - X^T Y\right)$$

To find the value of θ set $\frac{\partial}{\partial\theta}J(\theta) = 0$

Which gives $\left(X^T X\theta - X^T Y\right) = 0$

Solving it for θ gives, $\theta = \left(X^T X\right)^{-1} X^T Y$

2. Write the conditions under which the closed form solution to Linear Regression exists.

Sol: The conditions for closed form is that the input feature matrix X cannot be ill conditioned or have a huge number of features and samples, the computation cost becomes exponential and cannot exist.

3. If we have a closed form solution for Linear Regression, why do we use Gradient Descent? Give an example of a situation where Gradient Descent is a better option than closed form calculations.

Sol: For a feature vector has a lot of features and samples, say in the order of 10^5 with a high percentage of non-zero entries. The computing power required for the term $\mathbf{X}^\intercal\mathbf{X}$ which would make it impractical to store and make the closed form model unfeasible. Which makes iterative models like Gradient descent a much better option.

4. Prove that for simple linear regression, the least square fit line always passes through the point $(\bar{X}, \bar{Y})$, where $\bar{X}$ and $\bar{Y}$ represent the arithmetic mean of the independent variables and dependent variables respectively.

Sol: The net expression for each sample in a given set of observations can be represented as $Y_i = \hat{\theta_0} + \hat{\theta_1} X_i + \hat{\mu_i}$

where θ_0 is the bias term, $theta_1$ is the coefficient of feature X_i, μ_i is the residual $(\hat{Y_i} - Y_i)$, and Y_i are the desired value for each sample.

When summing over all N samples, the sum of all residuals is 0 for least square fit. If the summation is divided by the number of samples in each side, we get:

$\bar{Y} = \hat{\theta_0} + \hat{\theta_1} \bar{X}$ where, $\bar{X}$ and $\bar{Y}$ are the mean values.

5. Can we use Linear regression for classification? If yes, how?

Sol: Linear regression cannot be directly used for classification. However, plugging in the model of linear regression into a sigmoid function makes it logistic regression which is the fundamental model for classification.

THEORY QUESTIONS (For PG Students only)

0.5 Which of the below expressions are linear regression models? Justify.

The solution for this question lies in establishing what is a linear regression model. In a simplified way, a linear regression model describes the relationship between a dependent variable, y, and one or more independent variables, X. A multiple linear regression model can be represented as,

$$y_i = \theta_0 + \theta_j X_{ij} + \theta_j X_{ij} + \cdots + \theta_p X_{ip} + \varepsilon_i, \quad i = 1, \cdots, n, \quad j = 1, \cdots, k$$

where,
y_i is the i th response.
θ_k is the k th coefficient, and θ_0 is the constant term in the model.
X_{ij} is the i th observation on the j th predictor variable, $j = 1, \ldots, p$.
ε_i is the i th noise term, that is, random error.
A linear regression model can also be of the form

$$y_i = \theta_0 + \sum_{k=1}^{K} \theta_k f_k \left(X_{i1}, X_{i2}, \cdots, X_{ip}\right) + \varepsilon_i, \quad i = 1, \cdots, n$$

where $f(.)$ is a scalar-valued function of the independent variables, X_{ij} s.
The functions, $f(X)$, might be in any form including nonlinear functions or polynomials.
The linearity, in the linear regression models, refers to the linearity of the coefficients θ_k.
That is, the response variable, y, is a linear function of the coefficients, θ_k.

source: https://www.mathworks.com/help/stats/what-is-linear-regression.html

Now the equations to identify are as follows:

1. $\theta_0 x_0 + \theta_1 x_1 + \ldots + \theta_n x_n = 0$

Sol: The expression is linear in coefficients but there is no dependent variable to map to. Hence not a linear regression model.

2. $\theta_0 \sin(x_0) + \theta_1 \sin(x_1) + \ldots. + \theta_n \sin(x_n) = 0$

Sol: The expression is linear in coefficients but there is no dependent variable to map to. Hence not a linear regression model.

3. $\sin(\theta_0 x_0) + \sin(\theta_1 x_1) + \ldots. + \sin(\theta_n x_n) = 0$

Sol: The expression does not even have linear coefficients, so it is definitely not a linear regression model.

4. $y_i = w_0 + \sum_{j=1}^{N} w_j \sinh(x_{i,j})$

Sol: This expression is the only linear model where the parameters are $\sinh$ functions of features but the coefficients are still linear in nature and are mapped to a dependent variable y_i. Tt can be inferred that for every i th set of features, the output can be computed using the given linear regression model.

0.6 Logistic Regression

1. What is the loss function used to train a Logistic Regression model. Write the expression for the probabilistic model used and the mathematical expression for the loss function.

Sol: Log Loss is the loss function used for logistic regression.

The probabilistic model for logistic regression is created by putting a linear model into a sigmoid function that gives a bounded value between 0 and 1 as the probability of predicting the output. The probabilistic model is given by the expression: $g(z) = \frac{1}{1+e^{(-z)}}$, where $z = \left(-\theta^T x\right)$

Hence, the expression $h_\theta(x) = \frac{1}{1+e^{(-\theta^r x)}}$

where, $g(z)$ is the output of the logistic regression model for a particular sample. z is the linear model given by $z = h_\theta(x) = \theta_0 x_0 + \theta_1 x_1 + \ldots + \theta_j x_j = \theta^T x$, with weights θ_j, bias θ_0, and features x.

$g(z)$ is also called as *log-odds* due to the invertability of the sigmoid function, and also because it gives the log of the probability of predicting a given output divided by the probability of not predicting it.

The The loss function is a representation of the loss function of linear model mapped to a probabilistic space between 0 and 1.

The loss function is given by $\text{Loss}\left(h_\theta(x), y\right) = \begin{cases} -\log\left(h_\theta(x)\right) & \text{if } y = 1 \\ -\log\left(1 - h_\theta(x)\right) & \text{if } y = 0 \end{cases}$.

Which can be written as $\text{Loss}\left(h_\theta(x), y\right) = -y \log\left(h_\theta(x)\right) - (1-y)\log\left(h_\theta(x)\right)$

Mapping this over all training samples and summing it makes the loss function to be:

$J(\theta) = \frac{1}{m}\left[\sum_{i=1}^{m} -y^{(i)} \log\left(h_\theta\left(x^{(i)}\right)\right) + \left(1 - y^{(i)}\right)\log\left(1 - h_\theta\left(x^{(i)}\right)\right)\right]$

where, $m =$ number of samples

2. Modify the above expression to include (a) Gaussian and (b) Laplacian (doubly exponential) regularization.

Sol: The Gaussian prior corresponds to L2 regularization and the Laplacian prior corresponds to L1 regularization. Plugging those in the log loss function gives us:

a. $J(\theta) = \frac{1}{m}\left[\sum_{i=1}^{m} -y^{(i)} \log\left(h_\theta\left(x^{(i)}\right)\right) + \left(1 - y^{(i)}\right)\log\left(1 - h_\theta\left(x^{(i)}\right)\right)\right] + \frac{\lambda}{2m}\sum_{j=1}^{n} \theta_j^2$

where, $m =$ number of samples, $n =$ number of features

b. $J(\theta) = \frac{1}{m}\left[\sum_{i=1}^{m} -y^{(i)} \log\left(h_\theta\left(x^{(i)}\right)\right) + \left(1 - y^{(i)}\right)\log\left(1 - h_\theta\left(x^{(i)}\right)\right)\right] + \frac{\lambda}{m}\sum_{j=1}^{n} |\theta_j|$

where, $m =$ number of samples, $n =$ number of features