Lab Assignment 4
Prakhar Gupta
B21AI027

Question 1:

Preprocessing and exploratory analysis.-

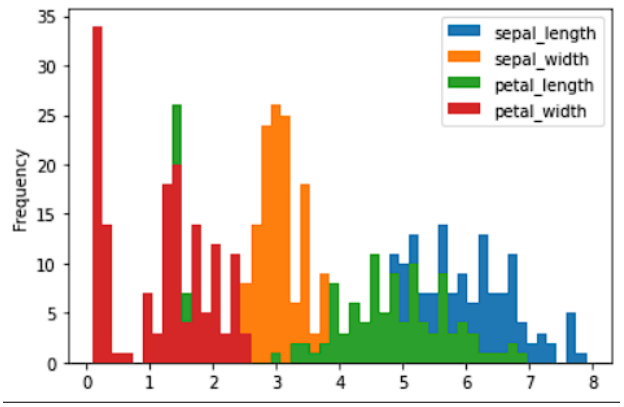
- Downloaded dataset using wget command called using os.system
- Load **iris.csv** in df using pd.read_csv
- Gave column names as column_names = ['sepal_length', 'sepal_width', 'petal_length', 'petal_width', 'species']
- Checked for not filled rows using df.isnull().sum()
- Printed column wise value count
- Plotted df
- Find unique values in ‘species’ column
- Converted df to X,y
- Applied MinMaxScalar()
- Splitted the X,y into train and test set
- Using train_test_split to split the X,y

Part 1 & 2-

- Used three constructor case

  Case $\Sigma_i = \sigma^2.I$ (I stands for the identity matrix)
  Case $\Sigma_i = \Sigma$ (covariance of all classes are identical but arbitrary!)
  Case Σi = actual covariance

- Implemented scratch built covariance function
- Implemented scratch built GaussianBayesClassifier with functions-gauss_pdf_value,train,test,plot_decision_boundary,predict



Part 3-

- Trained the Gaussian Bayes model on the training dataset and plot the decision boundary for each case of three cases
- We got performance as

```
Performance of Identity:
Accuracy:  93.33333333333333
Confusion Matrix:
 [[19  0  0]
 [ 0 10  3]
 [ 0  0 13]]
Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        19
           1       1.00      0.77      0.87        13
           2       0.81      1.00      0.90        13

    accuracy                           0.93        45
   macro avg       0.94      0.92      0.92        45
weighted avg       0.95      0.93      0.93        45



Performance of actual:
Accuracy:  100.0
Confusion Matrix:
 [[19  0  0]
 [ 0 13  0]
 [ 0  0 13]]
Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        19
           1       1.00      1.00      1.00        13
           2       1.00      1.00      1.00        13

    accuracy                           1.00        45
   macro avg       1.00      1.00      1.00        45
weighted avg       1.00      1.00      1.00        45



Performance of identical_but_arbitrary:
Accuracy:  97.77777777777777
Confusion Matrix:
 [[19  0  0]
 [ 0 12  1]
 [ 0  0 13]]
Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        19
           1       1.00      0.92      0.96        13
           2       0.93      1.00      0.96        13

    accuracy                           0.98        45
   macro avg       0.98      0.97      0.97        45
weighted avg       0.98      0.98      0.98        45
```

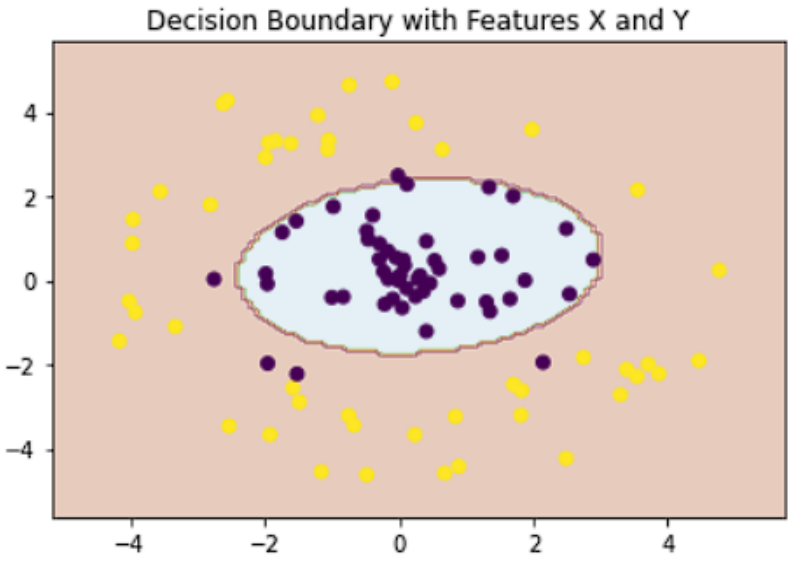
- It can be seen that all three models perform well on the iris dataset, with the Model of actual covariance matrix performing the best.

Part 4-

- Performed K-Fold cross validation on training dataset
- We got average accuracy as
- `Average accuracy: 0.9333333333333333  using model: Identity`
- `Average accuracy: 0.9566666666666667  using model: actual`
- `Average accuracy: 0.9577777777777778  using model: identical_but_arbitrary`
- High accuracy scores for k fold validation suggest that the model has good generalizability.In comparison to the "Identity" model, the models utilising "actual" and "identical but arbitrary" had greater accuracy scores, suggesting that these two models are more generalizable.However, It is important to remember that these accuracy scores are only one metric, and that a more thorough analysis is required to evaluate whether the models are generalizable.

Part 5-

- Using a function find 100 point that lie in and on the circle of radius 5
- Labelled the points according to whether they lie outside radius 3 or inside
- Made the scatter plot for it
- Using actual covariance in GaussianBayesClassifier as a model, trained the model on the points
- Plotted the decision_boundary for the points
- Got `Accuracy: 0.95`



Question 2:

Part a-

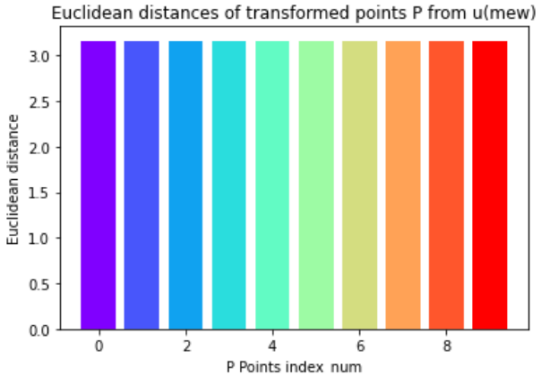
- Created a scratch built function for calculating covariance
- Using `np.random.multivariate_normal` created random point X(array of points) having 1000 points
- Then created covariance matrix from the X
- Using `np.linalg.eig` calculated eigenvalues and eigenvectors
- Plotted the random points along with eigenvectors
- Eigenvalues=
```
[0.93958821 2.07654301]
```
- EigenVectors
```
[[-0.71949162 -0.69450112]
 [ 0.69450112 -0.71949162]]
```

Part b-

- Using `scipy.linalg` find the Y
- Then find covariance of Y using scratch built covariance function
- Got Y=
```
[[1.00000000e+00 3.81597531e-16]
 [3.81597531e-16 1.00000000e+00]]
```
-
- The purpose of this transformation is to reduce the correlation between the variables. This can be seen by the fact that the new covariance matrix has lower (almost identity matrix) values compared to the initial covariance matrix, indicating that the variables are less correlated. The new covariance matrix is a Identity matrix implying the correlation between features is zero

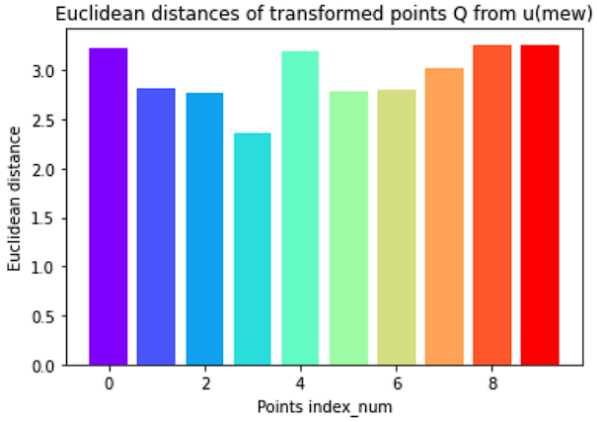
Part c-

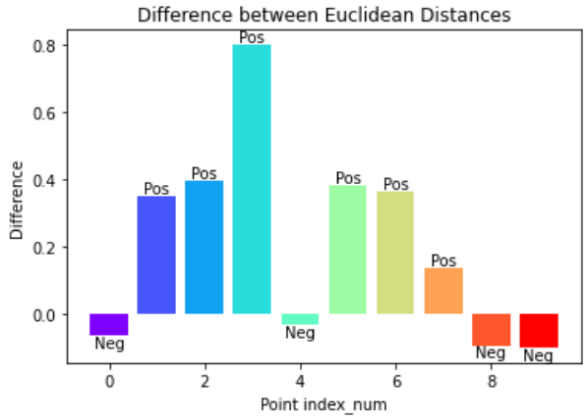
- Defined a function to find random point on a circle of given radius
- Plotted 10 points generated via above function
- Plotted Euclidean distances of points from u(mew) on bar graph



Part d-

- Transformed P to Q as `Q=(sigma)^-0.5 P`
- Plotted difference between point euclid_dist of points in P from u(mew) with corresponding euclid_dist of points in Q from u(mew)





- From the above difference between Euclidean Distance of P-Q . It is clearly evident that the points after transformation has shifted closer to the mean point. This is due to the fact that the transformation aims to reduce the correlation between the variables and normalise the data due to which points come closer to the mean point