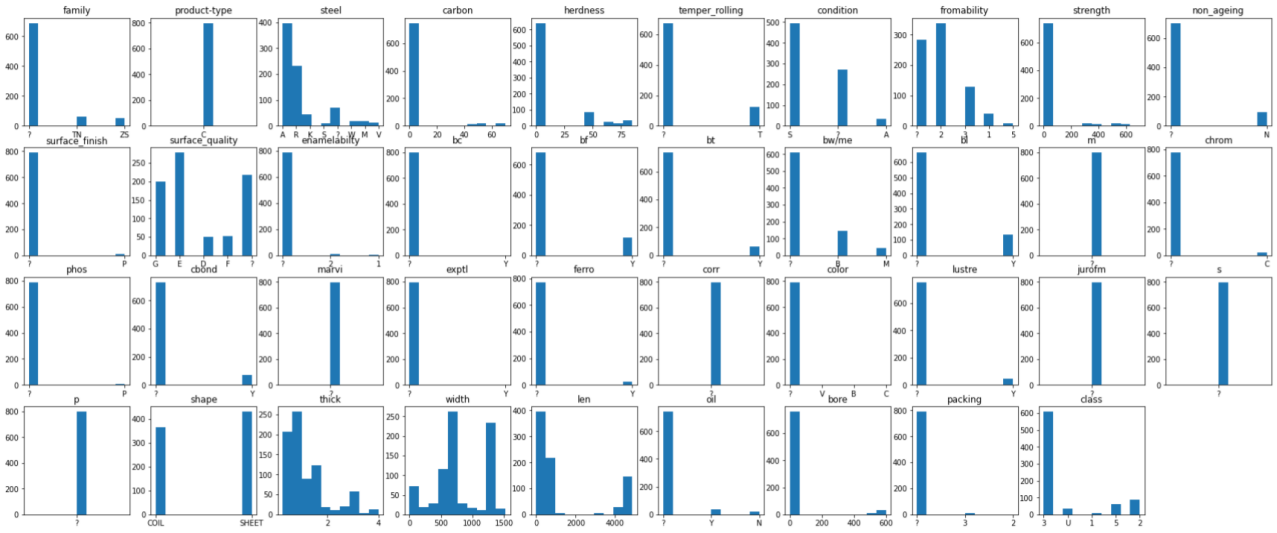
Pattern Recognition and Machine Learning
Lab 4 Assignment
Principal Component Analysis and Linear Discriminant Analysis

Jatin Lohar
B21CS091

Question 1 (PCA)

1. Imported the numpy and pandas and the imported the data using ***'pandas.read_csv'.*** Named the columns and the in order to know about the data and to do the explorative analysis, I plotted the histograms of data. And got the following results.



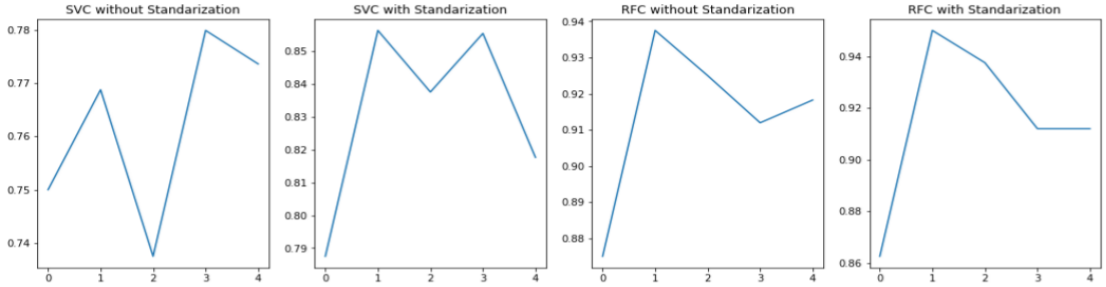
Found that columns = [0, 5, 7, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 35, 37] has large number of '?' values so dropped the features.

Then in the remaining data having some **'?'** values, replaced them with the mode.

2. Using ***'sklearn.preprocessing'*** imported ***'StandardScaler'*** and ***'LabelEncoder'***. Fitted and transformed the data into it. Then using ***'train_test_split'*** splitted the data into 65:35 ratio.

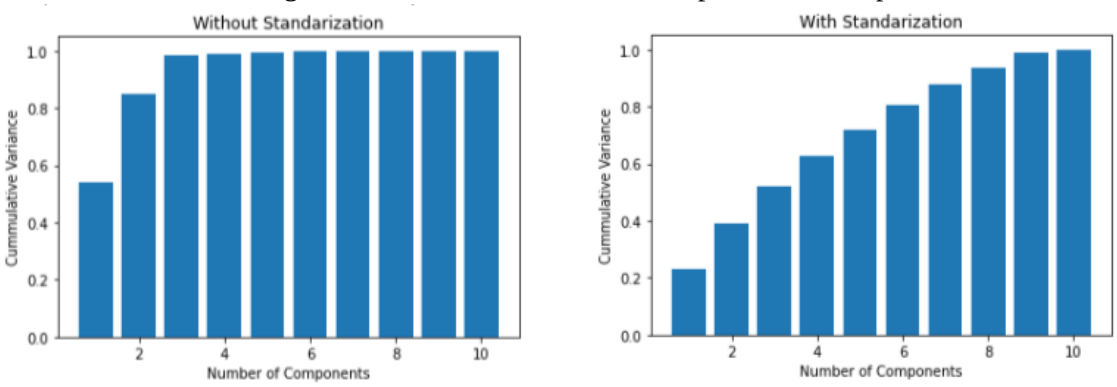
3. Applied SVM, RandomForestClassifier on the data set and applied KFold and got the following results.

```
Accuracy of SVM without Standarization: 0.762
Accuracy of Random Forest Classifier without Standarization: 0.914

Accuracy of SVM with Standarization: 0.831
Accuracy of Random Forest Classifier with Standarization: 0.915

[<matplotlib.lines.Line2D at 0x7fd81b1e61c0>]
```



4. Make a class named PCA and made n_components as the initialiser with default value as 2. The functions worked as follows.
   - FIT : found the mean of each column and the subtracted mean from the data, so as to centralise it. Found covariance of the new data using numpy.dot(). Then found the eigenvalues and eigenvectors of the covariance matrix. Using ZIP function, merged the eigenvalues and the eigenvectors. Then sorted the data according to the eigenvalues using lambda functions. Then added the eigenvectors with highest eigenvalue into variable named ***'components_'*** .
   - Transform : As I already got the ***'components_'*** matrix in the fitting part, I simply multiplied the input data with the ***'components_'*** matrix to get the transformed the data and then returned it.

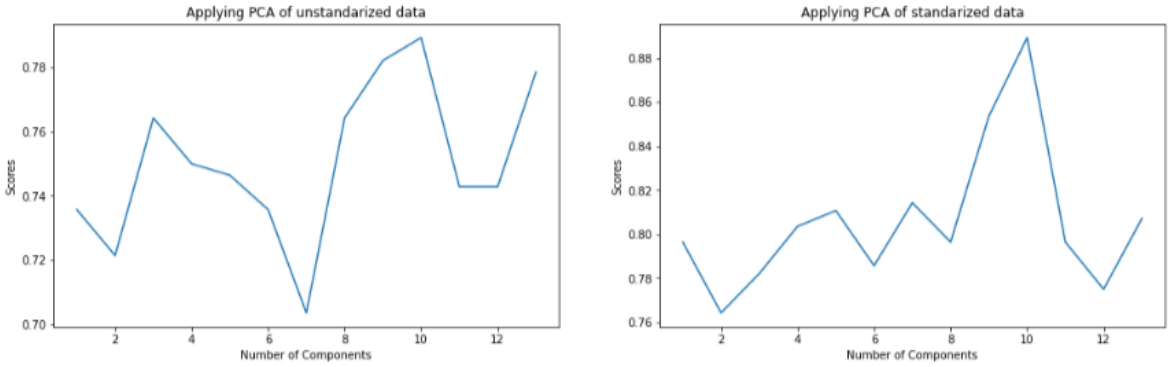
5. Transformed the data using the PCA made from scratch. The plotted the bar plots.



6. Used DecisionTreeClassifier, SVC and RandomForestClassifier to do this part. Applied KFold and split the data into training and testing. Fitted the model differently for standardized data and non-standardized ones. Stored the scores in respective list and the printed the means of the obtained lists.

```
Decision Tree Classifier Accuracy without Standarization : 0.8608569182389937
Random Forest Classifer Accuracy without Standarization : 0.8959512578616351
SVM Accuracy without Standarization : 0.7619418238993712

Decision Tree Classifier Accuracy with Standarization : 0.867122641509434
Random Forest Classifer Accuracy with Standarization : 0.885943396226415
SVM Accuracy with Standarization : 0.8308411949685535
```

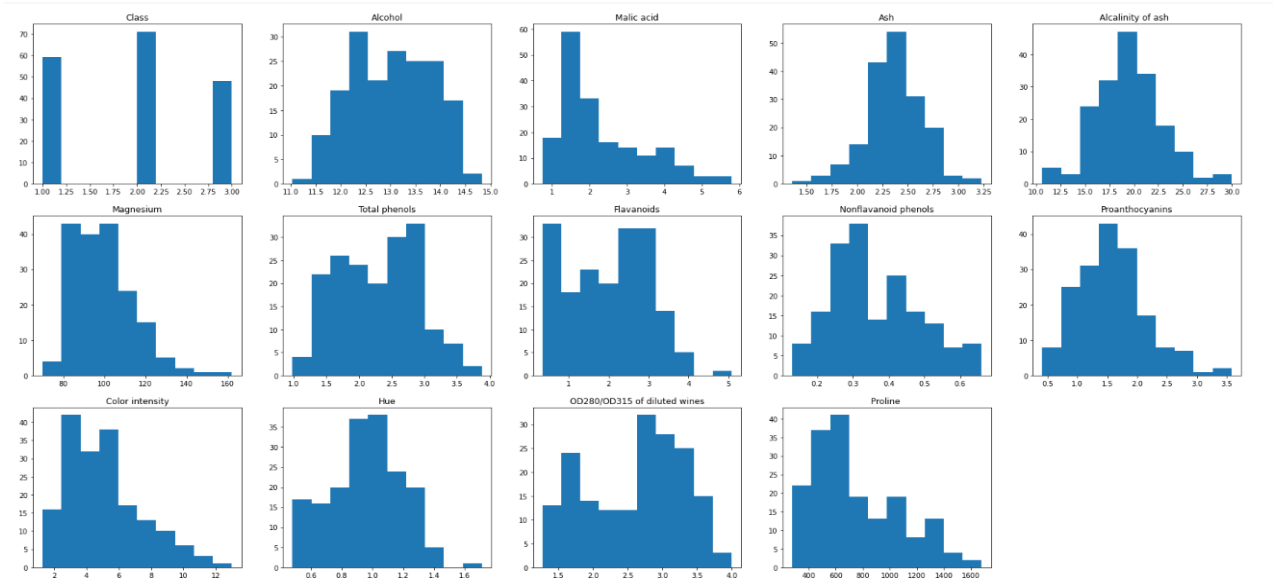
7. Varied ***'n_components'*** and appended the scores of SVC model into a list. Latter plotted the list using line plot.



The optimum number of components for both the data came out to be 10, but also they were fluctuating on multiple runs.

Question 2 (Linear Discriminant Analysis)

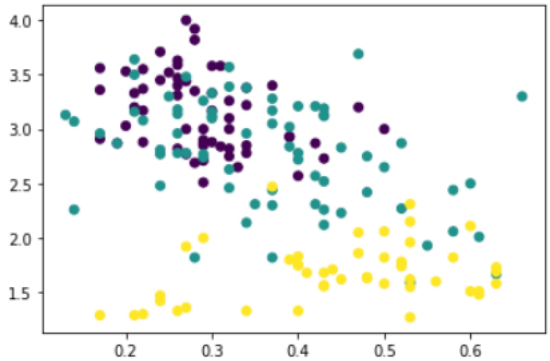
Imported the data using ***'pandas.read_csv'*** and splitted it into X and Y. Plotted the histograms for visualizing the dataset.



1. Used previously made Gaussian Naïve Bayes Classifier, as default LDA classifier. Then, made a class ***'LDA'*** . Used n_components as the initializer. And made the following functions.
   - FIT : Gets input as X and Y and the train the Gaussian Model so as to use in later parts and stores the data and their means according to their classes. Also found the Scatter within and scatter between matrices. Found the eigenvalues and eigenvectors using them. In case n_components is not given, in that case selecting the optimum number using sort of thresholding value given as input or by default 0.95. Then got the indexes and if maximum eigenvalues and then selected the eigenvectors accordingly. And stored into W_matrix.
   - Transform : As obtained W_matrix during fitting, so simply applying dot product and returning the obtained transformed data.
   - Predict : Used to predict some data. Internally calls the GaussianNB and return the predicted value.
   - Score : Used the Test function of GaussianNB and returns the accuracy.
   - Scatter_within : return the scatter within matrix
   - Scatter_between : return the scatter between matrix

2. Found the features with maximum variance after appling MinMaxScaler. And found that ***'Flavoids' and 'Hue'*** have maximum variance.

```
Flavanoids Hue
<matplotlib.collections.PathCollection at 0x7fd81a8ca520>
```



3. PCA on the dataset and transformed it and stored it into ***'X_trans_PCA'.*** Applied DecisionTreeClassifer, GaussianNB using KFold and calculated the accuracies.

```
Mean Accuray of Decision Tree Classifier with LDA = 0.9833333333333332
Mean Accuray of Decision Tree Classifier with PCA = 0.7222222222222222
Mean Accuray of Gaussian Naive Bayes with LDA = 0.9944444444444445
Mean Accuray of Gaussian Naive Bayes with PCA = 0.7676470588235295
```

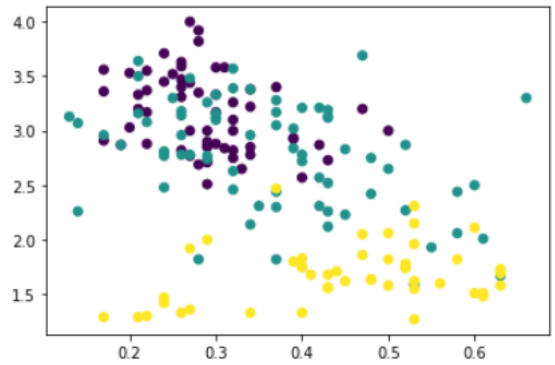
Got better results in LDA then in PCA.

4. Made a DataFrame and stored the obtained values in it.

	DTG	GNB
0	0.983333	0.994444
1	0.722549	0.767647

Aplied MinMaxScaler and found the entris with maximum variance and ploted the scatter plot for that features.

```
[7, 11]
<matplotlib.collections.PathCollection at 0x7fd81b343220>
```



5. Used LDA as a classifier and tested the data and found the following accuracies.

```
[0.9333333333333333, 0.8666666666666667, 0.8666666666666667, 1.0, 0.896551724137931, 0.896551724137931]
```