Machine Learning (CS60050) - Assignment 1 Report

Group :
Gangaram Arvind Sudewad- 20CS30017
Chenna Keshava Reddy- 20CS10014

Dataset:

Dataset Description:
The data describes 3 types of pathological lung cancers.
Attribute Information:
Label: Column 1
All predictive attributes are nominal, taking on integer values 0-3

UNSUPERVISED LEARNING

Unsupervised learning is a type of algorithm that learns patterns from untagged data. The hope is that through mimicry, which is an important mode of learning in people, the machine is forced to build a concise representation of its world and then generate imaginative content from it.

Question 1:

Tasks :

1) Apply PCA (select number of components by preserving 95% of total variance). (in-built function allowed for PCA).
2) Plot the graph for PCA.
3) Using the features extracted from PCA, apply K-Means Clustering. Vary the value of K from 2 to 8. Plot the graph of K vs normalized mutual information (NMI). Report the value of K for which the NMI is maximum. (in-built function not allowed for K-Means).

The given data has some missing values with '?' in it, these are taken care of by filling them with the mean of that column.

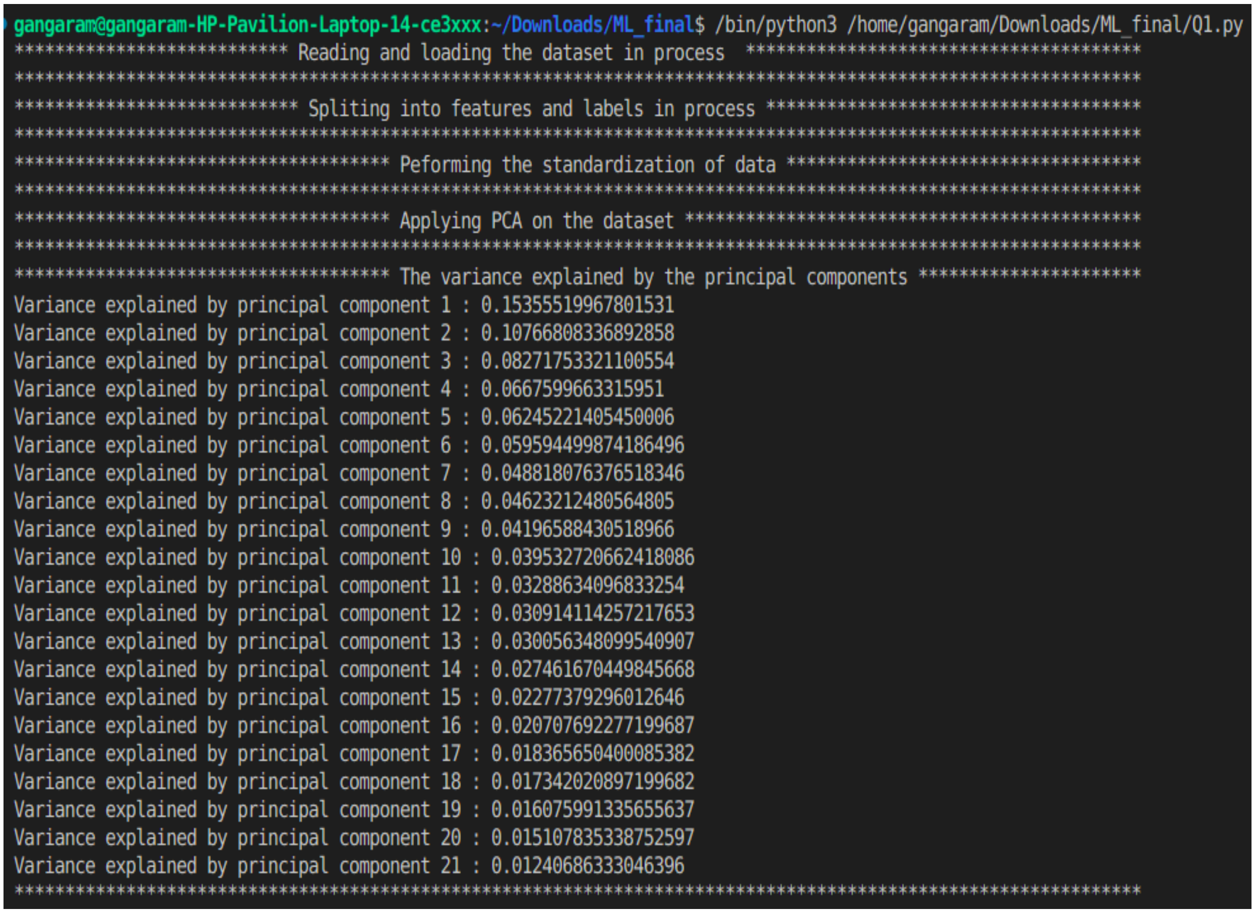
Principal Components Analysis (PCA):

Principal Component Analysis is an unsupervised learning algorithm, It is a statistical process that converts the observations of correlated features into a set of linearly uncorrelated features with the help of orthogonal transformation. Principal Components are the newly transformed features.

To achieve co variance more than 95%, we have to select a number of components at least 21.
But for visualizing the graph we have taken the number of components as 2.

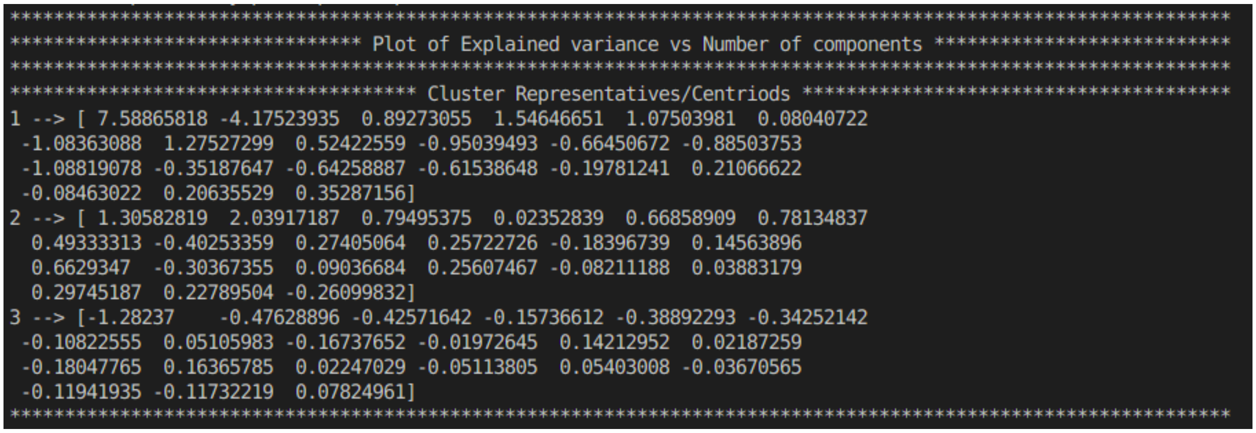
K-Means Clustering is an Unsupervised Learning algorithm, which groups the unlabeled dataset into different clusters. Here K defines the number of predefined clusters that need to be created in the process. It is a centroid-based algorithm, where each cluster is associated with a centroid. The main aim of this algorithm is to minimize the sum of distances between the data point and their corresponding clusters. Determines the best value for K center points or centroids by an iterative process.Assigns each data point to its closest k-center. Those data points which are near to the particular k-center, create a cluster.
For determining the quality of clustering Normalized mutual information (NMI) is used, Since it's normalized we can measure and compare the NMI between different clusterings having different number of clusters.

THE VARIANCE EXPLAINED BY THE PRINCIPAL COMPONENTS:



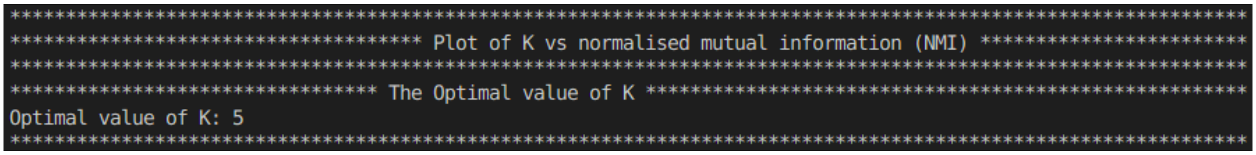
```
gangaram@gangaram-HP-Pavilion-Laptop-14-ce3xxx:~/Downloads/ML_final$ /bin/python3 /home/gangaram/Downloads/ML_final/Q1.py
*************************** Reading and loading the dataset in process  ***************************************
***************************************************************************************************************
**************************** Spliting into features and labels in process *************************************
***************************************************************************************************************
************************************* Peforming the standardization of data ***********************************
***************************************************************************************************************
************************************* Applying PCA on the dataset *********************************************
***************************************************************************************************************
************************************* The variance explained by the principal components ***********************
Variance explained by principal component 1 : 0.15355519967801531
Variance explained by principal component 2 : 0.10766808336892858
Variance explained by principal component 3 : 0.08271753321100554
Variance explained by principal component 4 : 0.0667599663315951
Variance explained by principal component 5 : 0.06245221405450006
Variance explained by principal component 6 : 0.059594499874186496
Variance explained by principal component 7 : 0.048818076376518346
Variance explained by principal component 8 : 0.04623212480564805
Variance explained by principal component 9 : 0.04196588430518966
Variance explained by principal component 10 : 0.039532720662418086
Variance explained by principal component 11 : 0.03288634096833254
Variance explained by principal component 12 : 0.030914114257217653
Variance explained by principal component 13 : 0.030056348099540907
Variance explained by principal component 14 : 0.027461670449845668
Variance explained by principal component 15 : 0.02277379296012646
Variance explained by principal component 16 : 0.020707692277199687
Variance explained by principal component 17 : 0.018365650400085382
Variance explained by principal component 18 : 0.017342020897199682
Variance explained by principal component 19 : 0.016075991335655637
Variance explained by principal component 20 : 0.015107835338752597
Variance explained by principal component 21 : 0.01240686333046396
***************************************************************************************************************
```

CLUSTER REPRESENTATIVES/CENTROIDS:



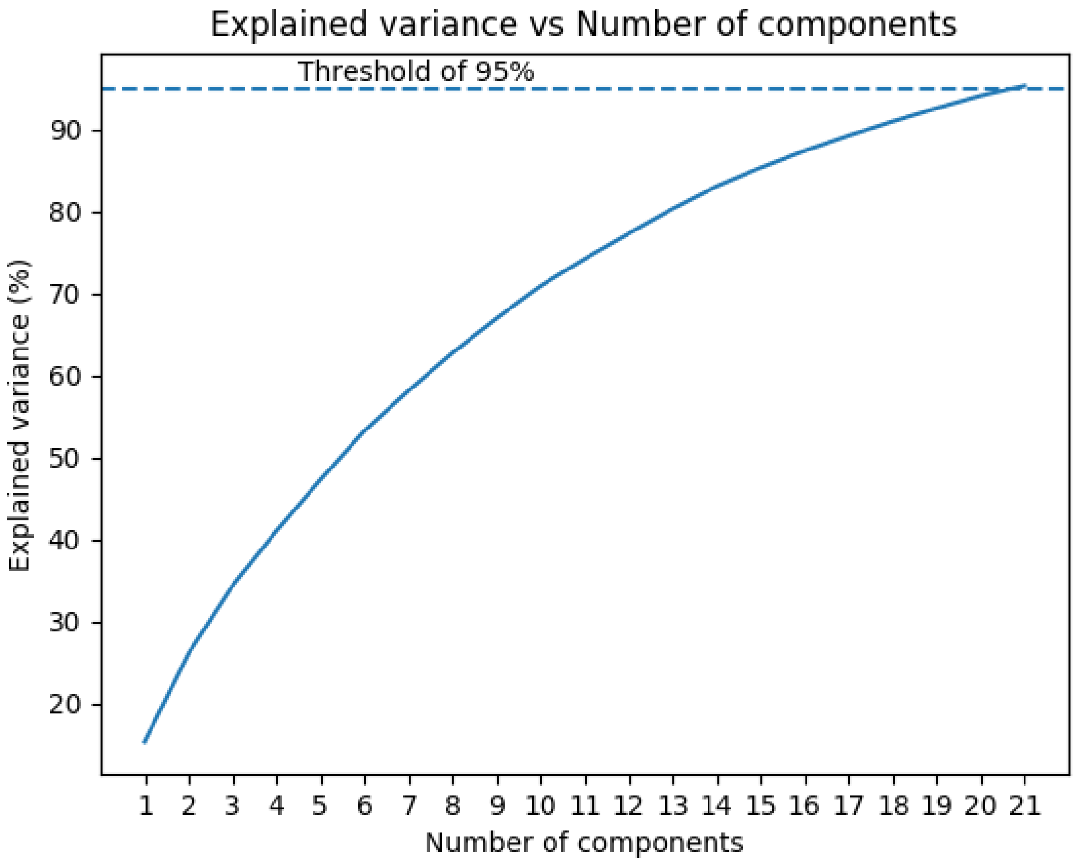
```
***************************************************************************************************************
******************************** Plot of Explained variance vs Number of components ***************************
***************************************************************************************************************
************************************* Cluster Representatives/Centriods ***************************************
1 --> [ 7.58865818 -4.17523935  0.89273055  1.54646651  1.07503981  0.08040722
 -1.08363088  1.27527299  0.52422559 -0.95039493 -0.66450672 -0.88503753
 -1.08819078 -0.35187647 -0.64258887 -0.61538648 -0.19781241  0.21066622
 -0.08463022  0.20635529  0.35287156]
2 --> [ 1.30582819  2.03917187  0.79495375  0.02352839  0.66858909  0.78134837
  0.49333313 -0.40253359  0.27405064  0.25722726 -0.18396739  0.14563896
  0.6629347  -0.30367355  0.09036684  0.25607467 -0.08211188  0.03883179
  0.29745187  0.22789504 -0.26099832]
3 --> [-1.28237    -0.47628896 -0.42571642 -0.15736612 -0.38892293 -0.34252142
 -0.10822555  0.05105983 -0.16737652 -0.01972645  0.14212952  0.02187259
 -0.18047765  0.16365785  0.02247029 -0.05113805  0.05403008 -0.03670565
 -0.11941935 -0.11732219  0.07824961]
***************************************************************************************************************
```

FIND THE OPTIMAL VALUE OF K:

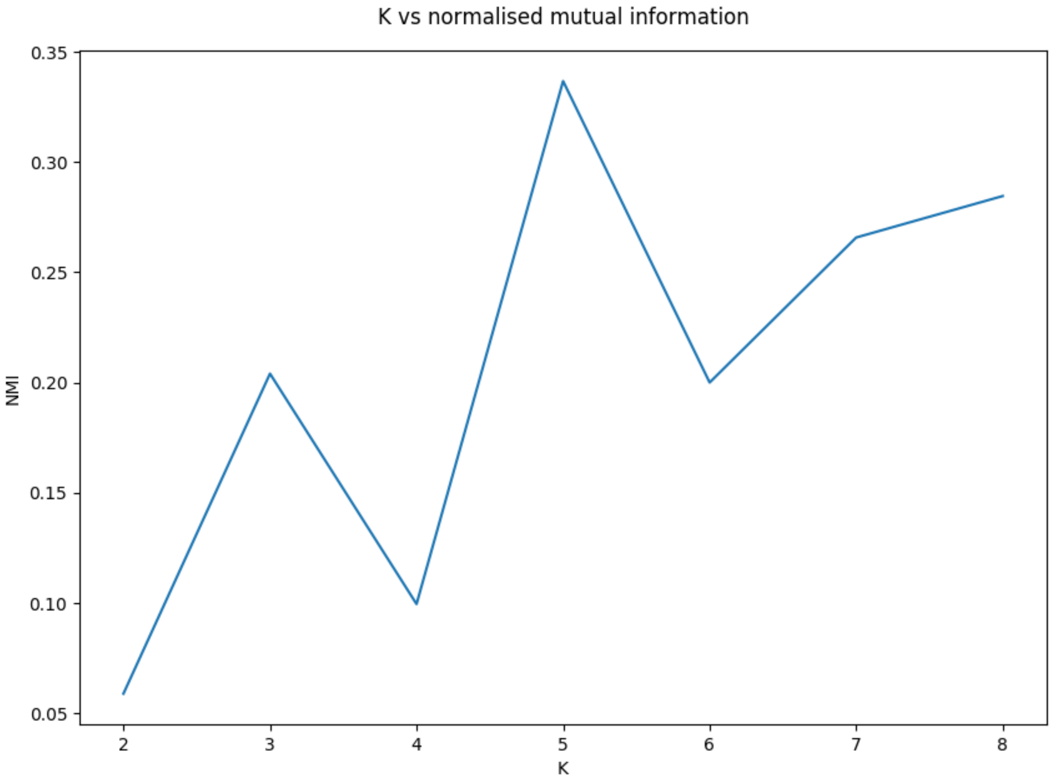


```
*****************************************************************************************************************
************************************* Plot of K vs normalised mutual information (NMI) **************************
*****************************************************************************************************************
********************************* The Optimal value of K ********************************************************
Optimal value of K: 5
*****************************************************************************************************************
```

PCA PLOT :


Explained variance vs Number of components
Threshold of 95%
Explained variance (%)
20
30
40
50
60
70
80
90
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
Number of components

PLOT K vs NORMALISED MUTUAL INFORMATION :



SUPERVISED LEARNING

Question 2:

Tasks:

1) Normalize the data using Standard Scalar Normalisation. Randomly divide the Dataset into 80% for training and 20% for testing. Encode categorical variables using appropriate encoding method (in-built function not allowed for normalization, sampling and encoding).
2) Implement the binary SVM classifier using the following kernels: Linear, Quadratic, Radial Basis function. Report the accuracy for each. (in-built function allowed).
3) Build an MLP classifier (in-built function allowed). for the given dataset. Use stochastic gradient descent optimiser. Keep learning rate as 0.001 and batch size of 32. Vary the number of hidden layers and number of nodes in each hidden layer as follows and report the accuracy of each:
    a. 1 hidden layer with 16 nodes
    b. 2 hidden layers with 256 and 16 nodes respectively.
4) Using the best accuracy model from part 3, vary the learning rate as 0.1, 0.01, 0.001, 0.0001 and 0.00001. Plot the learning rate vs accuracy graph.
5) Use backward elimination method on the best model found in part 3 to select the best set of features. Print the features.
6) Apply ensemble learning (max voting technique) using SVM with quadratic, SVM with radial basis function and the best accuracy model from part 3. Report the accuracy.

Sampling

Since the provided lung-cancer dataset has some missing values denoted by ?, we have replaced those by the mode of the remaining values in the respective columns. We have used the custom scalar function for normalising the data and then we have used the custom sampler function to split the data into 80% training data and 20% test data.

Categorical encoding was not necessary since no string or coded data was involved.

Then the binary SVM classifier was used with the kernel as linear, quadratic and radial basis function

Procedure:
Then the MLP Classifier was used with the given respective parameters:

1 stochastic gradient descent optimiser
2. learning rale as 0.001
3. batch size of 32
4. for the first classifier 1 hidden layer with 16 nodes
5. for the second classifier: 2 hidden layers with 256 and 16 nodes
Then we selected the MLP Classifier model with best accuracy and used it with learning rates as 0.1. 0.01, 0.001, 0.0001 and 0.00001

Then we used forward feature selection, on the best MLP Classifier and listed out the selected features
Finally, we used ensemble learning with max voting technique

Three models were considered
1. Quadratic SVM
2 Radial basis function SVM
3 Best MLP Classifier

For each datapoint, the mode of results was considered. For all such models, the accuracy was calculated

We have used both Binary SVM classifiers (with Linear, Quadratic, and Radial Basis Function as kernels) and Multi-Layer Perceptron classifiers with different learning rates for the assignment. The data used for training was in .csv format. We have shuffled the data. We considered a split of 80:20 for the Train and Test set.

Standard Scalar Normalization:
It scales each input variable separately by subtracting the mean and dividing by the standard deviation to shift the distribution to have a mean of zero and a standard deviation of one.

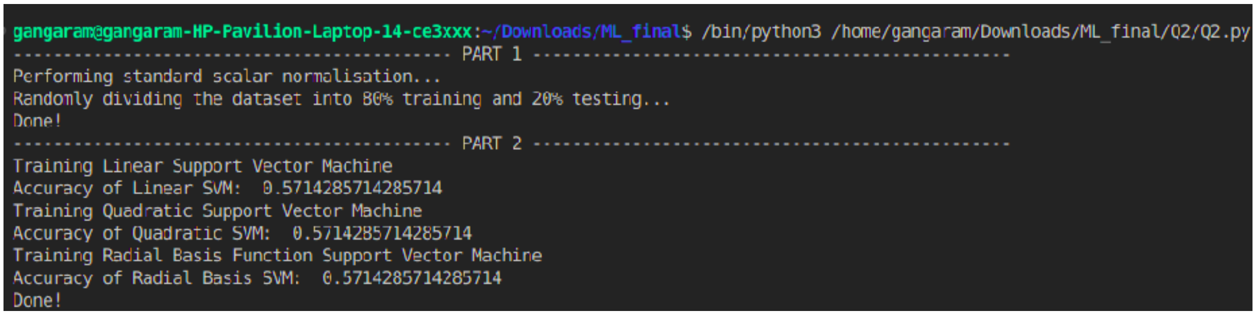
X = (x – mean) / std or we can use
max_min normalization =X = (x-min)/(max-min)

Support vector classifier:

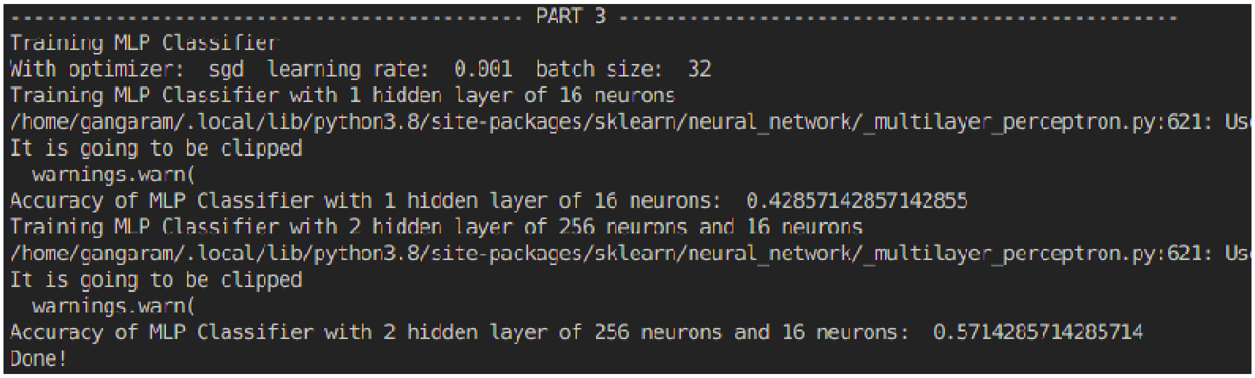
i. Binary SVM Classifier is a linear discriminant classifier.
ii. **Uses Vapnik's principle**: to never solve a more complex problem as a first step before the actual problem.
iii. After training the weight vector can be written in terms of training samples lying in class boundaries.
iv. The Primal problem for soft-margin hyperplanes and kernel function: Minimize w.r.t '**w**'

$$L_p = \frac{1}{2}||w||^2 + C* \Sigma_t s^t - \Sigma_t \alpha^t [r^t (w^T \phi(x^t) + w_0) - 1 - s^t] - \Sigma_t \mu^t s^t$$

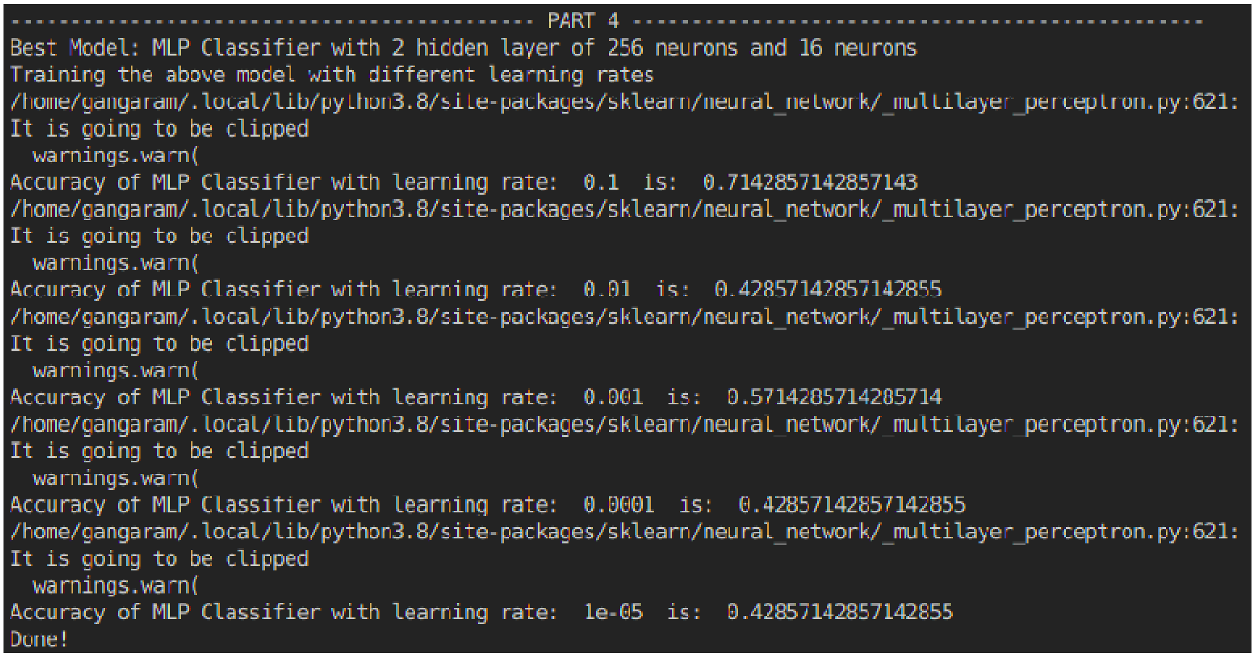
Multi-Layer Perceptron :
i. It is a network of perceptrons arranged in a particular fashion, called the architecture of the model.
ii. We consider a multi-layered feed-forward network, this means there is no feed-back or loop in the network.



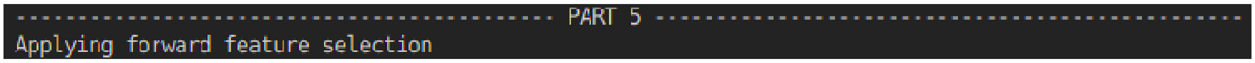
```
gangaram@gangaram-HP-Pavilion-Laptop-14-ce3xxx:~/Downloads/ML_final$ /bin/python3 /home/gangaram/Downloads/ML_final/Q2/Q2.py
-------------------------------------------- PART 1 --------------------------------------------
Performing standard scalar normalisation...
Randomly dividing the dataset into 80% training and 20% testing...
Done!
-------------------------------------------- PART 2 --------------------------------------------
Training Linear Support Vector Machine
Accuracy of Linear SVM:  0.5714285714285714
Training Quadratic Support Vector Machine
Accuracy of Quadratic SVM:  0.5714285714285714
Training Radial Basis Function Support Vector Machine
Accuracy of Radial Basis SVM:  0.5714285714285714
Done!
```



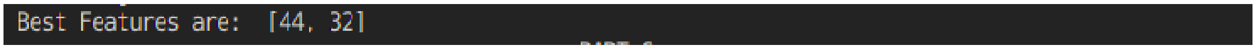
```
-------------------------------------------- PART 3 ------------------------------------------------
Training MLP Classifier
With optimizer:  sgd  learning rate:  0.001  batch size:  32
Training MLP Classifier with 1 hidden layer of 16 neurons
/home/gangaram/.local/lib/python3.8/site-packages/sklearn/neural_network/_multilayer_perceptron.py:621: Us
It is going to be clipped
  warnings.warn(
Accuracy of MLP Classifier with 1 hidden layer of 16 neurons:  0.42857142857142855
Training MLP Classifier with 2 hidden layer of 256 neurons and 16 neurons
/home/gangaram/.local/lib/python3.8/site-packages/sklearn/neural_network/_multilayer_perceptron.py:621: Us
It is going to be clipped
  warnings.warn(
Accuracy of MLP Classifier with 2 hidden layer of 256 neurons and 16 neurons:  0.5714285714285714
Done!
```



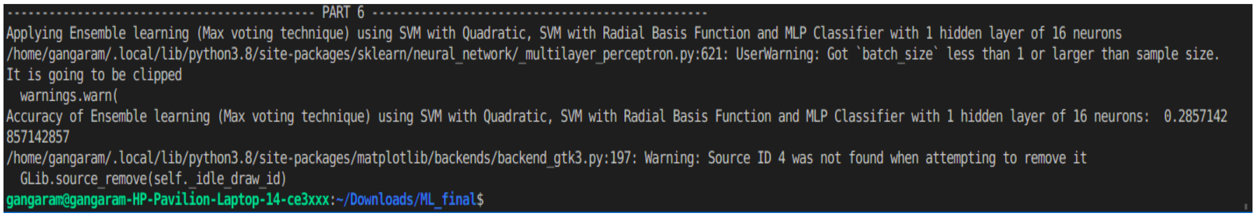
```
-------------------------------------------- PART 4 ------------------------------------------------
Best Model: MLP Classifier with 2 hidden layer of 256 neurons and 16 neurons
Training the above model with different learning rates
/home/gangaram/.local/lib/python3.8/site-packages/sklearn/neural_network/_multilayer_perceptron.py:621:
It is going to be clipped
  warnings.warn(
Accuracy of MLP Classifier with learning rate:  0.1  is:  0.7142857142857143
/home/gangaram/.local/lib/python3.8/site-packages/sklearn/neural_network/_multilayer_perceptron.py:621:
It is going to be clipped
  warnings.warn(
Accuracy of MLP Classifier with learning rate:  0.01  is:  0.42857142857142855
/home/gangaram/.local/lib/python3.8/site-packages/sklearn/neural_network/_multilayer_perceptron.py:621:
It is going to be clipped
  warnings.warn(
Accuracy of MLP Classifier with learning rate:  0.001  is:  0.5714285714285714
/home/gangaram/.local/lib/python3.8/site-packages/sklearn/neural_network/_multilayer_perceptron.py:621:
It is going to be clipped
  warnings.warn(
Accuracy of MLP Classifier with learning rate:  0.0001  is:  0.42857142857142855
/home/gangaram/.local/lib/python3.8/site-packages/sklearn/neural_network/_multilayer_perceptron.py:621:
It is going to be clipped
  warnings.warn(
Accuracy of MLP Classifier with learning rate:  1e-05  is:  0.42857142857142855
Done!
```



```
-------------------------------------------- PART 5 ------------------------------------------------
Applying forward feature selection
```



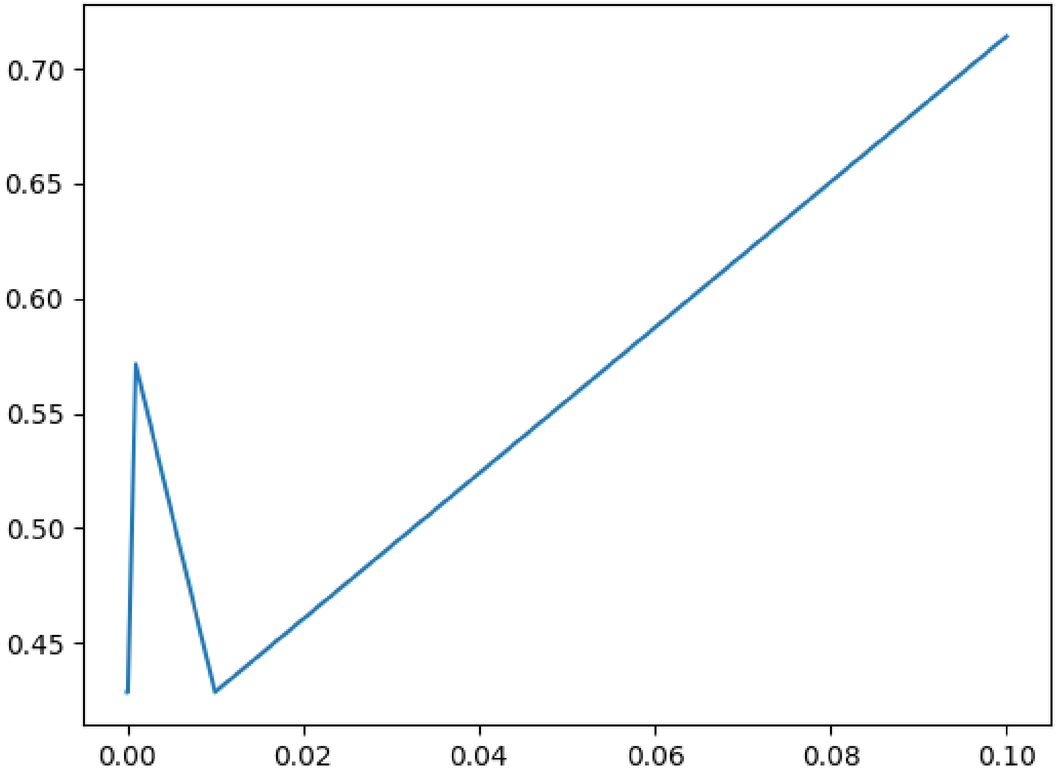
```
Best Features are:  [44, 32]
```



```
-------------------------------------------- PART 6 ------------------------------------------------
Applying Ensemble learning (Max voting technique) using SVM with Quadratic, SVM with Radial Basis Function and MLP Classifier with 1 hidden layer of 16 neurons
/home/gangaram/.local/lib/python3.8/site-packages/sklearn/neural_network/_multilayer_perceptron.py:621: UserWarning: Got `batch_size` less than 1 or larger than sample size.
It is going to be clipped
  warnings.warn(
Accuracy of Ensemble learning (Max voting technique) using SVM with Quadratic, SVM with Radial Basis Function and MLP Classifier with 1 hidden layer of 16 neurons:  0.2857142
857142857
/home/gangaram/.local/lib/python3.8/site-packages/matplotlib/backends/backend_gtk3.py:197: Warning: Source ID 4 was not found when attempting to remove it
  GLib.source_remove(self._idle_draw_id)
gangaram@gangaram-HP-Pavilion-Laptop-14-ce3xxx:~/Downloads/ML_final$
```

Using SVM with Quadratic, SVM with Radial Basis Function and MLP Classifier with 1 hidden layer of 16 neurons: **0.2857142857142857**

PLOT THE GRAPH OF LEARNING RATE v/s ACCURACY:


0.70
0.65
0.60
0.55
0.50
0.45
0.00
0.02
0.04
0.06
0.08
0.10