Indian Institute of Technology Kharagpur

Department of Computer Science and Engineering

Assignment 2

Hardik Soni
20CS30023
Saurabh Jaiswal
20CS30047

Machine learning (CS60050)

November 6, 2022

Contents

Page

1 Dataset

1.1 Description

Wine recognition data contains data inferred from the results of a chemical analysis of wines grown in the same region in Italy but derived from three different cultivars. The analysis determined the quantities of 13 constituents found in each of the three types of wines. The dataset has the following data fields:-

- Alcohol
- Malic acid
- Ash
- Alcalinity of ash
- Magnesium
- Total phenols
- Flavanoids
- Nonflavanoid phenols
- Proanthocyanins
- Color intensity
- Hue
- OD280/OD315 of diluted wines
- Proline

The Dataset has 178 training examples and are available in the '.csv' format as 'Dataset_A.csv' in the Assignment's Directory.

1.2 Number of Instances

Class Label	%
1	33.14
2	39.88
3	26.98

Note: The Dataset is available at **https://archive.ics.uci.edu/ml/datasets/Wine**. The Dataset in the Code is named **Dataset_A.csv**

2 Question 1: Unsupervised Learning (30)

2.1 Introduction and Tasks

In this report we analyse the results of K-Means Clustering applied on Wine Data-set using Features Extracted from the PCA Analysis Preserving 95% of Variance.

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 normalised mutual information (NMI). Report the value of K for which the NMI is maximum. (in-built function not allowed for K-Means).
4. Prepare a report including all your results

2.2 Principal Component Analysis (PCA)

2.2.1 Introduction

The central idea of principal component analysis (PCA) is to reduce the dimensionality of a data set consisting of a large number of interrelated variables while retaining as much as possible of the variation present in the data set

2.2.2 Mathematics Behind PCA

Principal Component: Component along the direction **w** such that its variance is maximum along all possible projections. Additionally, the subsequent principal components are perpendicular to the prior principal components.

- Take the whole dataset consisting of **d+1** dimensions and ignore the labels such that our new dataset becomes d dimensional.
- Compute the mean for every dimension of the whole dataset.
- Compute the covariance matrix of the whole dataset.
- Compute eigenvectors and the corresponding eigenvalues.
- Sort the eigenvectors by decreasing eigenvalues and choose k eigenvectors with the largest eigenvalues to form a d $\times$ k dimensional matrix W.
- Use this d $\times$ k eigenvector matrix to transform the samples onto the new subspace.

2.3 K-Means Clustering

2.3.1 Introduction

K-means algorithm is an iterative algorithm that tries to partition the dataset into K pre-defined distinct non-overlapping subgroups (clusters) where each data point belongs to **only one group**. It tries to make the intra-cluster data points as similar as possible while also keeping the clusters as different (far) as possible. It assigns data points to a cluster such that the sum of the squared distance between the data points and the **cluster's centroid** (arithmetic mean of all the data points that belong to that cluster) is at the **minimum**.The homogeneity (similarity) of the data points within a cluster increases as the amount of variance within the cluster decreases.

2.3.2 Algorithm

The way **k-means** algorithm works is as follows:

1. Specify number of clusters **K**.
2. Initialize centroids by first shuffling the dataset and then randomly selecting K data points for the centroids without replacement.
3. Keep iterating until there is no change to the centroids. i.e assignment of data points to clusters isn't changing.
    - Compute the sum of the squared distance between data points and all centroids.
    - Assign each data point to the closest cluster (centroid).
    - Compute the centroids for the clusters by taking the average of the all data points that belong to each cluster.

2.3.3 Mathematics Behind K-Means Algorithm

The approach K-means follows to solve the problem is called **Expectation-Maximization**. The **E-step** is assigning the data points to the closest cluster. The **M-step** is computing the centroid of each cluster. Below is a break down of how we can solve it mathematically:- The objective function is:

$$J = \sum_{i=1}^{m} \sum_{k=1}^{K} w_{ik} \parallel x^i - \mu_k \parallel^2 \tag{1}$$

where $w_{ik} = 1$ for data point x_i if it belongs to cluster k; otherwise, $w_{ik} = 0$. Also, μ_k is the centroid of x_i's cluster. It's a minimization problem of two parts. We first minimize J w.r.t. w_{ik} and treat μ_k fixed. Then we minimize J w.r.t. μ_k and treat w_{ik} fixed

E-Step:- Technically speaking, we differentiate J w.r.t. w_{ik} first and update cluster assignments (E-step).

$$\frac{\partial f}{\partial w_{ik}} = \sum_{i=1}^{m} \sum_{k=1}^{K} \parallel x^i - \mu_k \parallel^2 \tag{2}$$

$$\implies w_{ik} = \begin{cases} 1, & \text{if } k = \text{argmin} \parallel x^i - \mu_j \parallel^2 \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

M-Step:- Then we differentiate J w.r.t. μ_k and recompute the centroids after the cluster assignments from previous step (M-step).

$$\frac{\partial f}{\partial \mu_k} = 2 \sum_{i=1}^{m} w_{ik}(x^i - \mu_k) \tag{4}$$

$$\implies \mu_k = \frac{\sum_{i=1}^{m} w_{ik} x^i}{\sum_{i=1}^{m} w_{ik}} \tag{5}$$

2.3.4 Important Points to Note

- Since clustering algorithms including **K-means** use distance-based measurements to determine the similarity between data points, it's recommended to **standardize** the data to have a mean of zero and a standard deviation of one since almost always the features in any dataset would have different units of measurements such as age vs income.
- Given **K-means** iterative nature and the random initialization of centroids at the start of the algorithm, different initializations may lead to different clusters since **K-means** algorithm may stuck in a local optimum and may not converge to global optimum. Therefore, it's recommended to run the algorithm using **different initializations of centroids(vary the value of K)** and pick the results of the run that that yielded the lower sum of squared distance.
- Assignment of examples isn't changing is the same thing as no change in within-cluster variation:

$$\frac{1}{m_k}\sum_{i=1}^{m_k} \parallel x^i - \mu_{c^k} \parallel^2 \tag{6}$$

2.4 Procedure and Results

2.4.1 Data Preprocessing

The Dataset is preprocessed using **.data_preprocessor(dataset)** function. In Machine Learning, Standard Scalar Normalisation is used to resize the distribution of values so that the mean of the observed values is 0 and the standard deviation is 1.

Standardisation:-

$$z = \frac{x-\mu}{\sigma} \tag{7}$$

with Mean:-

$$\mu = \frac{1}{N}\sum_{i=1}^{N}(x_i) \tag{8}$$

and Standard Deviation:-

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i-\mu)^2} \tag{9}$$

Procedure:

2.4.2 Principal Component Analysis

The dataset contains 13 numerical columns. Since PCA is a variance maximizing algorithm it will load on that column which has a large variance compared to other, so we standardize the data to make PCA independent of such scale differences.

- We need to select a number of **principal components** by **preserving** 95% of the total variance.
- For applying PCA, an in-built function is used by passing an **n-component** argument as 13.
- Then the **.explained_variance_ratio_[:]** method is used to compute the ratio of variance captured by each individual component.
- 10 principal components collectively explains variance data.

$$\textbf{Number of Components selected: 10} \tag{10}$$

$$\textbf{Variance captured: 96.16971684450642 \% (> 95\%)} \tag{11}$$

- The principal component features were extracted by using **.transform()** function om the scaled dataset.
- This new scaled dataset is called **clustering_dataset**

To visualize the **explained variance_ratio**, we plot a graph for explained ratio vs principal components and the cumulative sum of explained **variance_ratio** vs **principal component**.

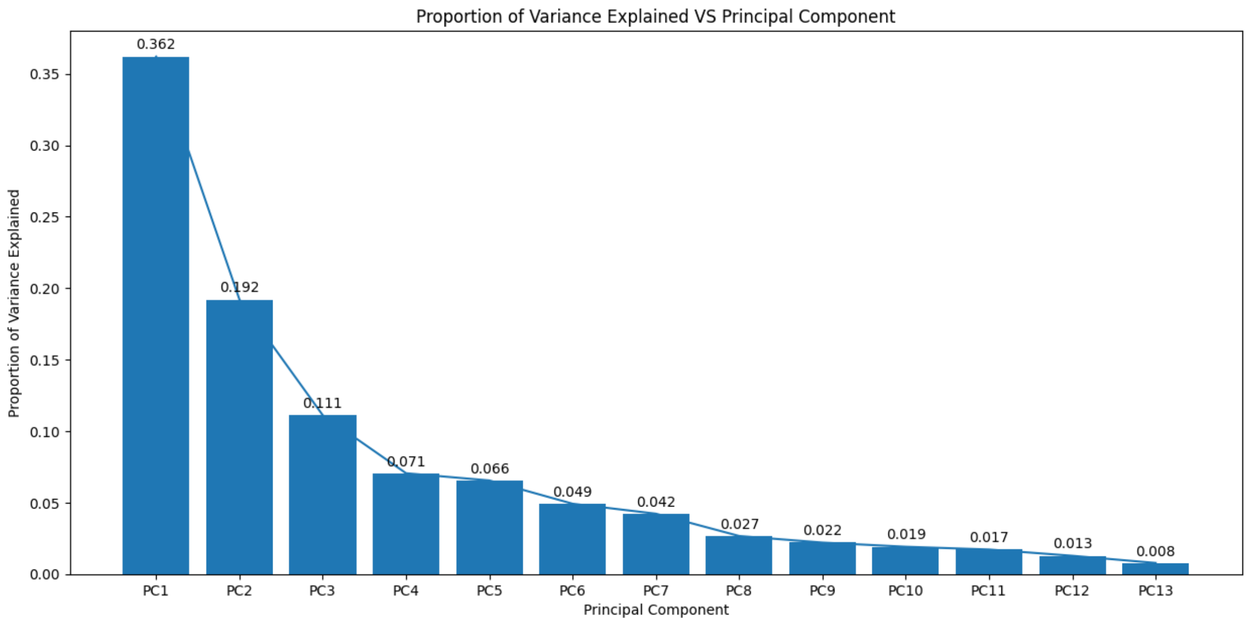


Figure 1: Plot for Variance Captured by Individual Component

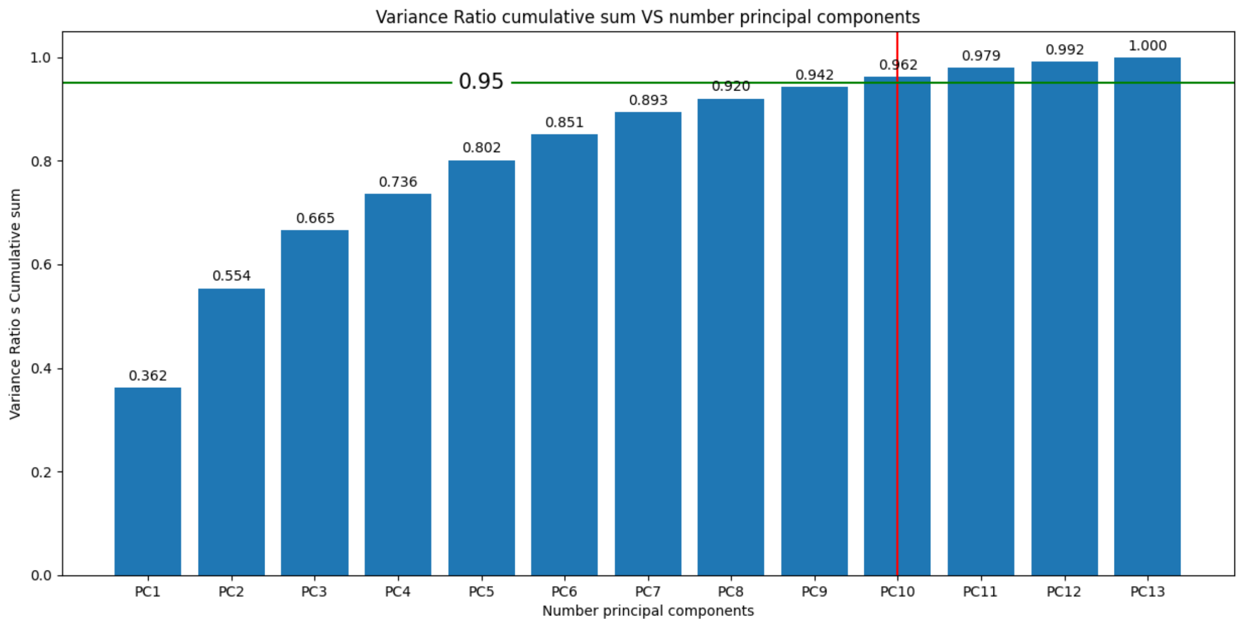


Figure 2: Plot for Cumulative Sum of Variance Captured by Component vs Principal Components

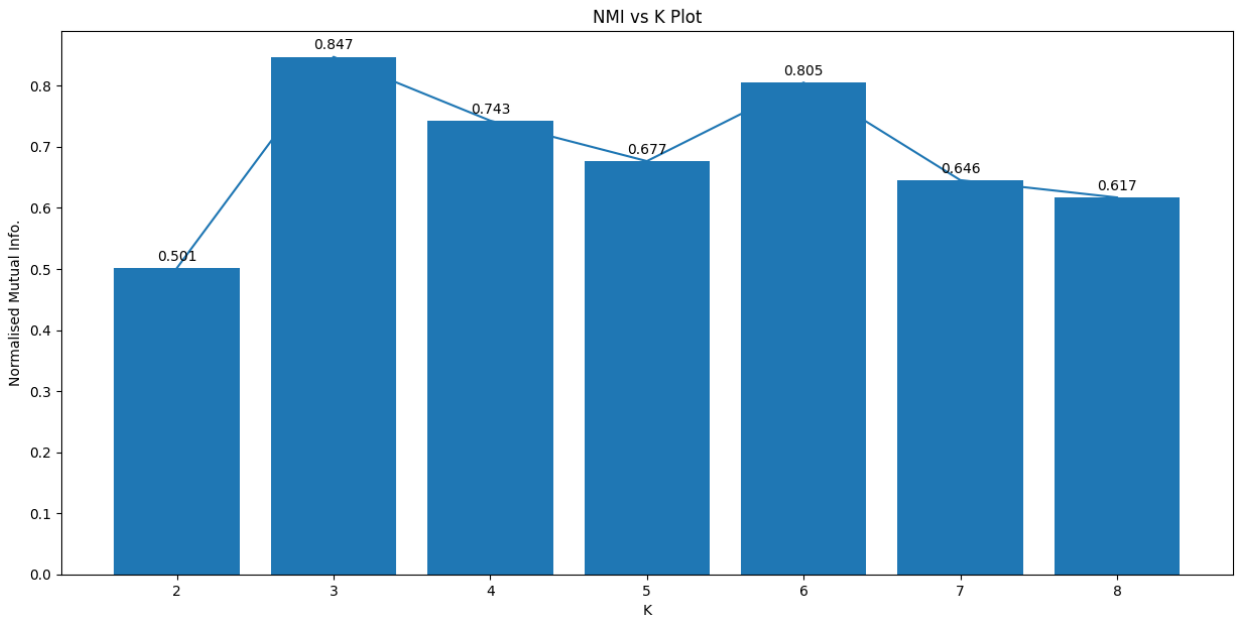


Figure 3: Number of Clusters(K) vs. Normalised Mutual Info. Score

2.4.3 K-Means Clustering using Features extracted from PCA

- Apply K-means clustering algorithm on the feature extracted form PCA.
- Vary the value of kt= from 2 to 8 to get the best K value for clustering.
- Normalised mutual information (NMI) was the criterion for determining the best value of k.
- The K_Means class is used to find the Clusters for Each Sample in the Dataset.
- The Compute_MI function is used to find the NMI Score for Each Iteration.

The Plot for K(number of Clusters) vs. Normalised Mutual Information is shown in Figure 3. The NMI Scores obtained are listed below.

K(No. of Clusters)	NMI Score
2	0.4837708380148453
3	0.8472896471857924
4	0.7612635309132971
5	0.6979912729775097
6	0.6659911094325242
7	0.6032026471650354
8	0.5954225310521523

3 Question 2: Supervised Learning (70)

3.1 Introduction and Tasks

1. Normalise 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 forward selection 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.

3.2 Support Vector Machine

It is a linear discriminant classifier which follows *Vapniks principle.* For classification it is sufficient to compute class boundaries. After training the weight vector can be written in terms of training samples lying in class boundaries. If r^t is the class (1) of x^t then we want $r^t(w^T x^t + w_0) \geq 1$ for all t. The 1 appears because of the margin between the classes. Thus we have a constrained optimisation problem which is to minimise $\frac{\|w\|}{2}$ subject to $r^t(w^T x^t + w_0) \geq 1$ for all t. This can be posed as Lagrangian problem $L_p = \frac{\|w\|}{2}$ - $\sum_{t=1}^{N} {}^t[r^t(w^T x^t + w_0)$ - 1] which needs to be minimized w.r.t w and w_0 and maximized w.r.t. Lagrange multipliers. We get $w = \sum_t \alpha^t r^t x^t$ and $\sum_t {}^t r^t = 0$. A dual problem is obtained and solved using quadratic optimisation technique. Samples with positive α^t are support vectors.

3.3 MLP Classifier

For weights vector W and input vector X it returns the output $y = sign(W^T X)$. Depending on the sign it is classified to one of the two classes. We want to find the optimum W which minimises the classification error. On one side of the hyperplane $W^T X = 0$ it takes positive values and on the other side negative. If a solution exists classes are called linearly separable. We normalize the data by making $Y = X$ (if X in class 1) and $Y = X$ (if X in class 2). Thus for correct classification $W^T Y > 0$Y. One error function is $J(W) = \sum_{Y_{misclassified}} W^T Y$

which we try to minimise. Using gradient descent we iteratively do $W_i = W_{i-1}$ - $\eta(i)\nabla J(W)$ where η is the learning rate. There are other error functions which can also be used such as $J_q(W) = \sum_{Y_{misclassified}} (W^T Y)^2$.

3.4 Procedure and Results

3.4.1 Data Preprocessing

Dataset is clean, with no missing data, so no pre-processing is needed.

3.4.2 Encoding of Features

The Dataset contains no categorical Data, hence no encoding of features is required.

3.4.3 Train Test Split

- The Dataset is Split into two sets for training and testing purpose using a function defined by us. The indices are randomly shuffled and then 80% of that is used for training and 20% is used for testing.
- test_size = 0.2
- test_train_split_data function is used to split dataset into test and training set.

The Size of the Training Set and Testing Set are mentioned in the table below:

Train Data size	143
Test Data size	35

3.4.4 Standard Scalar Normisalisation

Implemented standard scaling to normalize the data using **standard_scalar_normalize() function**

3.4.5 Support Vector Machine

- A supervised learning algorithm in which we try to find such decision boundary which can successfully divide the data into different classes.(for multi-class classification.).
- We use different functions called kernels to project non-separable data into a space where they can be separated easily. These kernels can be linear or non-linear depending upon separability of data or the choice of the programmer.
- sklearn implementation of SVM Classifier is used with linear, poly(degree = 2), radial basis function is used.
- Linear Kernel:-
  - Accuracy (Linear Kernel): 97.14
  - F1 (Linear Kernel): 97.17

- Radial Basis Function Kernel:-
  - Accuracy (RBF Kernel): 100.00
  - F1 (RBF Kernel): 100.00
- Quadratic Kernel:-
  - Accuracy (Quadratic Kernel): 88.57
  - F1 (Quadratic Kernel): 88.62

Maximum Accuracy is Achieved for Radial Basis Function Kernel.
Maximum F1-Score is Achieved for Radial Basis Function Kernel.

3.4.6 Multi-layer Perceptron

- This is a fully-connected, feedforward neural network. The network is composed of multiple neurons which introduce non-linearity on the basis of their activation function. This network is capable of learning both linear and non-linear relationship between the data and labels.
- There are a lot of hyper-parameters to vary and instead of grid searching the best set of parameters, we sequentially tackle one hyper-parameter and find the best value for it.
- sklearn implementation of MLP Classifier is used and we vary the hidden layer sizes to find the best size. The learning rate is varied after this and we find the best learning rate to use for the model

The MLP Classifier is implemented using Stochastic Gradient Descent Optimiser for a :-

- Batch Size = 32
- Learning Rate = 0.001

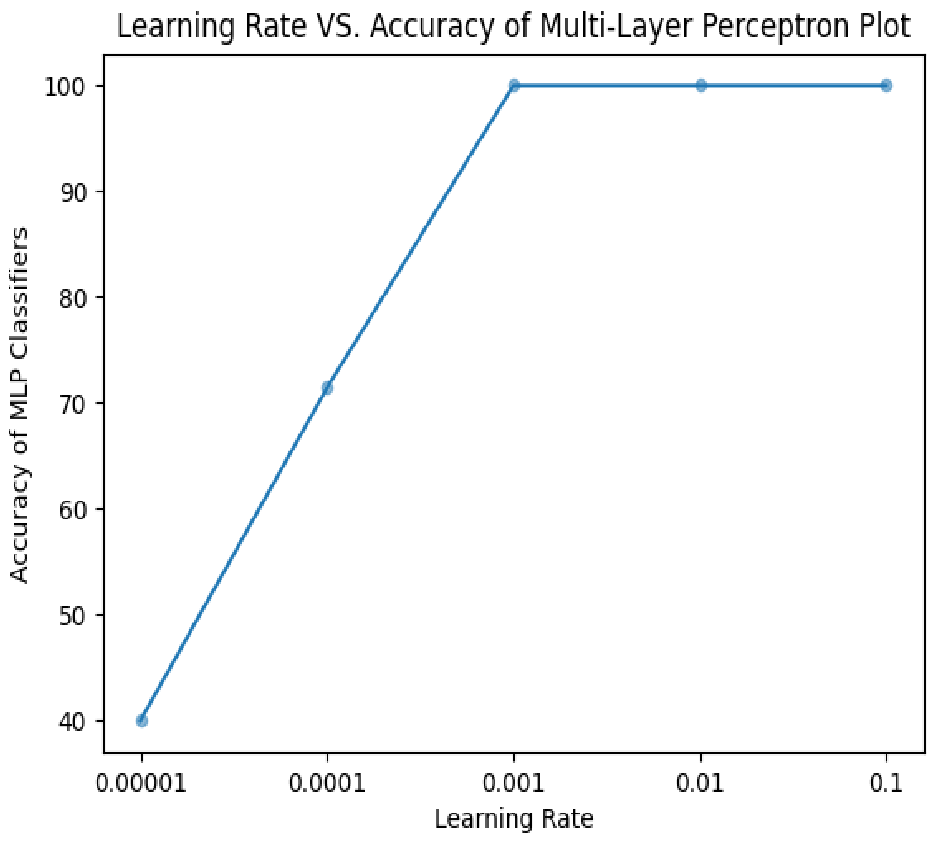
Implemented with varying hidden layer:-

1. 1 hidden layer with 16 nodes
2. 2 hidden layers with 256 and 16 nodes respectively.

Accuracy is 1.0 for both classifier with different Hyper-Parameters. When we vary the Learning Rate from 10^{-1} to 10^{-5}:-

- Accuracy is 40.0000 for Learning Rate: 0.00001
- Accuracy is 71.4286 for Learning Rate: 0.00010
- Accuracy is 100.0000 for Learning Rate: 0.00100
- Accuracy is 100.0000 for Learning Rate: 0.01000
- Accuracy is 100.0000 for Learning Rate: 0.10000

The Plot for the Same is shown below.



3.4.7 Forward Selection Method

- Try to add a feature to the subset of best features and see if accuracy increases, if it does then keep the feature else discard it.
- This process gives us three features: Flavanoids, Alcohol, Ash

The Best Features:-

1. Flavanoids
2. Alcohol
3. Ash

Accuracy for these Best Features Selected from Forward Selection: 1.0

3.4.8 Ensemble Learning

- Method to combine the model outputs of all 3 models:- SVM with quadratic, SVM with radial basis function and the MLP Classifier.

We use max voting method which means taking the mode of the predicted class of all models for each input instance.
Accuracy Found for Hard(Max.) Voting: 97.75281 %

4 Files Present

4.1 Brief Description:

This directory contains the following files:

- **Assignment2_Gr_A.pdf**: Description of problem statement
- **wine_data.csv**: Contains the data used for clustering and training the SVM Classifier and MLP Classifier.
- **q1.py**: Contains the solution to the Question 1: Unsupervised Learning
- **q2.py**: Contains the solution to the Question 2: Supervised Learning
- **requirements_1.txt**: Contains all the necessary dependencies and their versions for Question 1
- **requirements_2.txt**: Contains all the necessary dependencies and their versions for Question 2

Plots and Other Results

1. For Question 1 there are three plots generated:-
   - **variance_ratio_pca.png** :- It's the Plot of the Fraction of Variance Explained for Each Component by Principal Component Analysis
   - **variance_ratio_cumulative sum.png** :- It's the Plot the Cumulative Fraction Sum of Variance Explained for Each Component by Principal Component Analysis. It also marks the component where 95% of Variance is explained.
   - **k_vs_nmi.png** :- It plots the value of K-the number of clusters vs Normalised Mutual Information Score.

   The Results and Outputs are present in **simulation1.txt**
2. For Question 2 there are two plots generated:-
   - **learning_rate_vs_acc.png** :- It plots the learning rate vs. accuracy for the MLP Classifier.

   The Results and Outputs are present in **simulation2.txt**

5 Instructions to Execute the Code

5.1 Question 1: Unsupervised Learning (30)

Make sure that you run the forth-mentioned command to install dependencies in your system also, make sure there is a .csv with the name same as the one of the global variable filename in the file q1.py, also there is a file named simulation1.txt in the directory.

- Navigate to the **q1.py** Python File in the Directory.
- Ensure you are using a latest version of **Python3**, and install all dependencies.
  pip install -r requirements1.txt
- Execute the file **q1.py**: python q1.py **OR** python3 q1.py
- Plots of the **PCA**(Cumulative Variance Explained marked with **95%** and the individual Variance Explained Plot) and the plot for **K vs normalised mutual information (NMI)** will be created in the same directory.
- The Output for the Code is printed on the Console as well as the copied into the file **simulation1.txt**

5.2 Question 2: Supervised Learning (70)

Make sure that you run the forth-mentioned command to install dependencies in your system also, make sure there is a .csv with the name same as the one of the global variable filename in the file q2.py, also there is a file named simulation2.txt in the directory.

- Navigate to the **q2.py** Python File in the Directory.
- Ensure you are using a latest version of **Python3**, and install all dependencies.
  pip install -r requirements2.txt
- Execute the file **q2.py**: python q2.py **OR** python3 q2.py
- Plots of the **Learning Rate vs. Accuracy** for the MLP (**Multi-Layer Perceptron**) will be created in the same directory.
- The Output for the Code is printed on the Console as well as the copied into the file **simulation2.txt**