Pattern Recognition and Machine Learning 2022 Winter Semester
Assignment 8

Question 1

Exploratory Data Analysis (EDA)

The given 'Wine Quality Dataset' is imported and read in as a pandas data frame along with appropriate column names obtained from the given link. The 'Class' target is 0-indexed. Statistics of the data are analyzed via '.head()', '.info()', and '.describe()'. Following observations are made:

- There are thirteen(13) features and one hundred and seventy-seven (177) data points.
- Only continuous features are available.
- The scale of continuous features is different.
- No missing (NaN) values.

The dataset is split based on 'features' and 'target'.

KMeans input data requirements:

- Numerical variables only: KMeans uses distance-based measurements to determine the similarity between data points.
- Data has no noises or outliers: KMeans is very sensitive to outliers and noisy data.
- Data has a symmetric distribution of variables (it isn't skewed): Read data always has outliers and noise, and it's difficult to get rid of it. Transformation of data to normal distribution helps to reduce the impact of these issues.
- Variables on the same scale: They have the same mean and variance, usually in a range of (-1.0, 1.0) standardized data or (0.0, 1.0) normalized data. For the ML algorithms to consider all attributes as equal, they must have the same scale.
- There is no collinearity (a high level of correlation between two variables): Correlated variables are not useful for ML segmentation algorithms because they represent the same characteristics of a segment. So correlated variables are nothing but noise.
- Few number of dimensions: As the number of dimensions (variables) increases, a distance-based similarity measure converges to a constant value between any given examples. The more variables the more difficult to find strict differences between instances.

Numerical variables only

'dataset.info()' shows that only numerical variables are present in the given dataset.

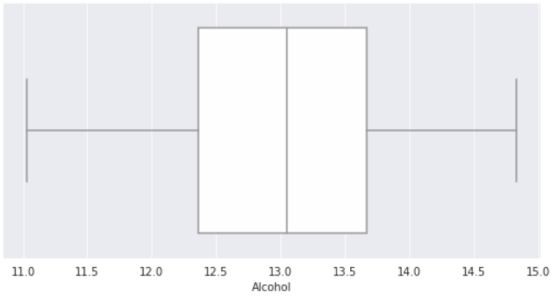
Missing values

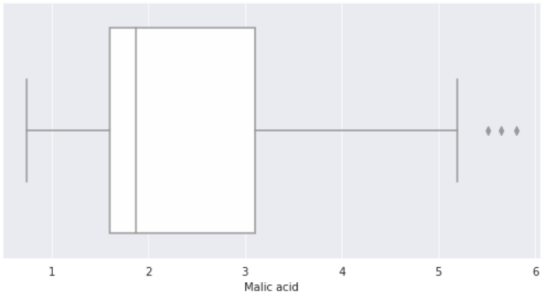
There are no missing values in the given dataset.

Data has no noises or outliers

Plotting the boxplots of features, it can be concluded that outliers are present in the following features:

- Malic acid
- Ash
- Alkalinity of ash
- Magnesium
- Proanthocyanins
- Color intensity
- Hue





To remove them, the following logic is used:

$$For\ all\ values\ >\ Q3\ +\ 1.5\ *\ IQR\ ;\ value\ will\ be\ reduced\ to\ Q3\ +\ 1.5\ *\ IQR$$
$$For\ all\ values\ <\ Q1\ -\ 1.5\ *\ IQR\ ;\ value\ will\ be\ incremented\ to\ Q1\ -\ 1.5\ *\ IQR$$

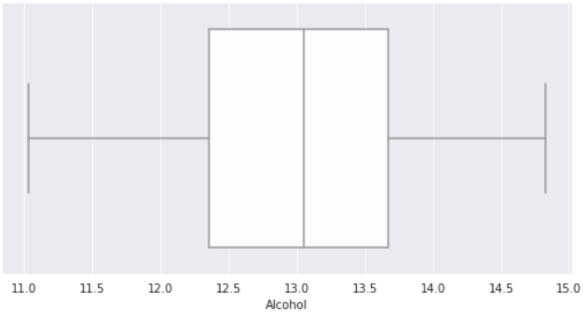
where:

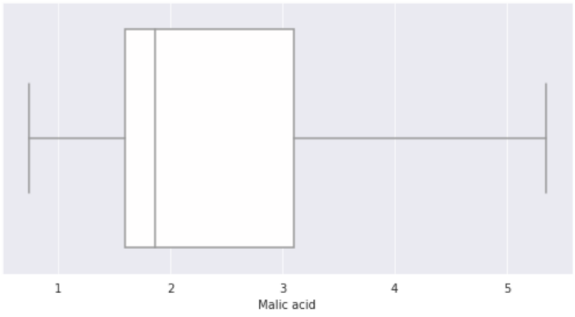
Q3: 3rd quartile or 75th percentile

Q1: 1st quartile or 25th percentile

IQR: Interquartile range

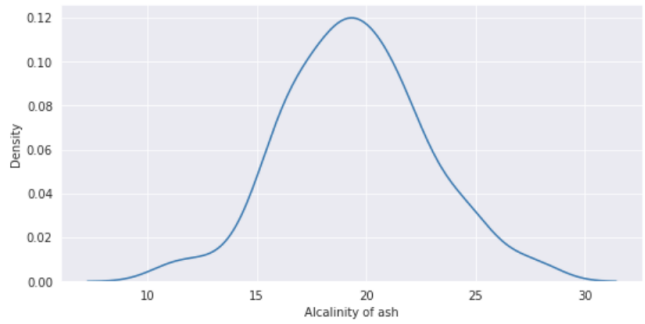
After implementing the above logic in the form of a function on the mentioned features the following box-plots(sample) are observed.

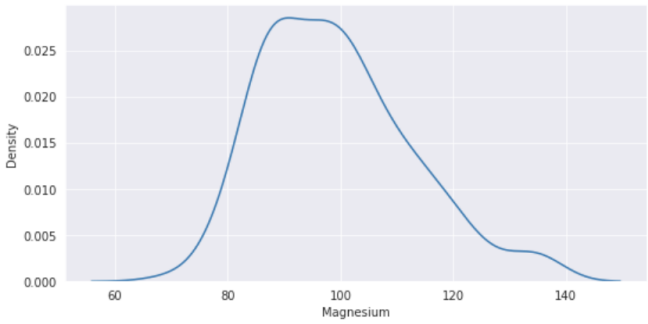




Data has symmetric distribution of variables (it isn't skewed)

Plotting the KDE plots of features, it can be concluded that though the data doesn't follow a normal distribution, it is not highly skewed. And hence requires no handling technique.





Variables on the same scale

'dataset.head()' shows that the numerical variables are not on the same scale. The following techniques will be used to scale them:

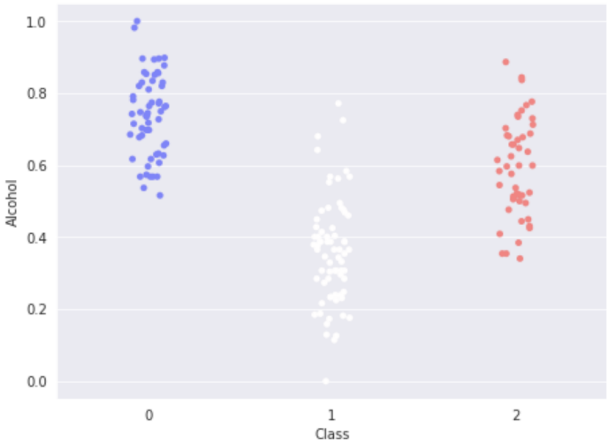
- Standardize features: 'log' transform is one of the possible ways to standardize data.
- Scale features: 'min_max' scaler is a way to get data in the range of 0 to 1.

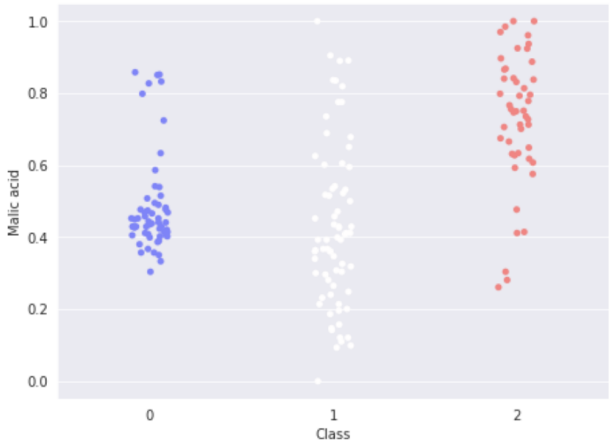
There is no collinearity (a high level of correlation between two variables)

Observing the correlation between each pair of variables via '.corr()', it can be concluded that 'Flavonoids' have high correlation values with 'Total phenols (0.815622)' and 'OD280/OD315 of diluted wines (0.801287)'. Hence, drop 'Flavanoids'.

Visualizing feature-target dependence

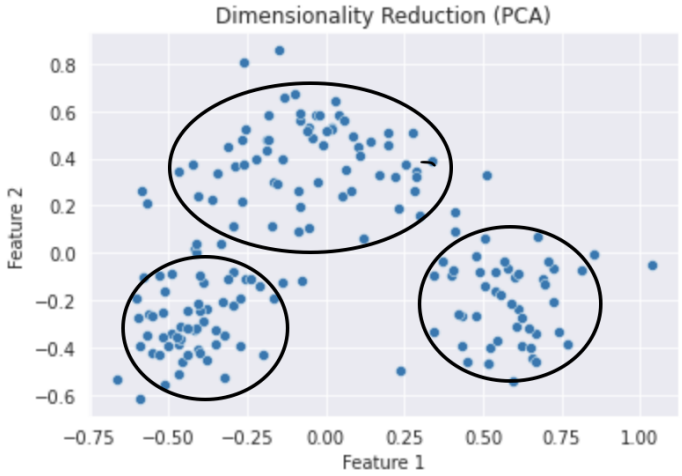
Plotting the stripplot of each feature with class.





Subpart - 1

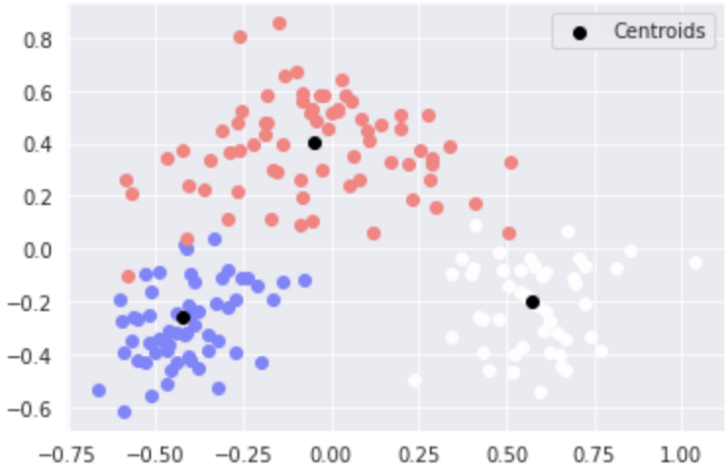
Principle component analysis (PCA) dimension reduction technique with 'n_components = 2' (for two-dimensional data) is used. Visualization of the data after implementing the mentioned technique is as follows:



By looking at the plot, it can be observed that 'k = 3' will be the best-suited value. This can be justified by the fact that we can observe three(3) different clusters (upper half, lower left, and lower right) in the plots.

Subpart - 2

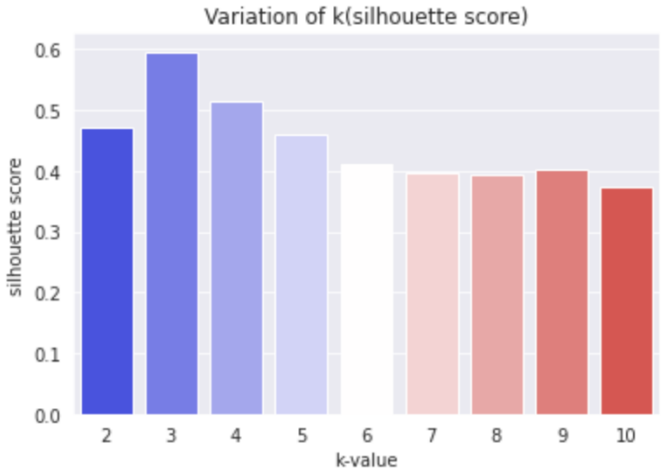
A k-means clustering algorithm (using sklearn library) is built and implemented using 'k = 3', as mentioned in Subpart - 1. Visualization of clusters along with the centroids is as follows:



Subpart - 3

Different values of ‘k’ are used to find the respective ‘Silhouette Score’. The results are tabulated and plotted as follows:

k	Silhouette Score
2	0.47049924273174437
3	0.5962118726536606
4	0.5143762907383391
5	0.45985055334122726
6	0.41050300438062665
7	0.3964546462822677
8	0.39344558634936055
9	0.40341671988251226
10	0.37483468529852985803

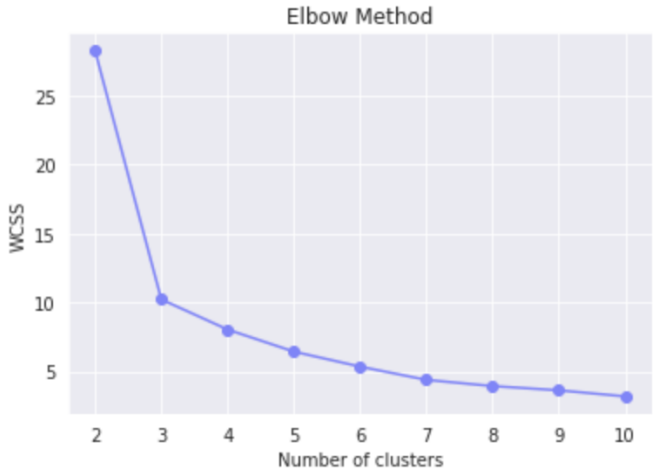


The Silhouette Score of one(1) means that the clusters are very dense and nicely separated. A score of zero(0) means that clusters are overlapping. A score of less than zero(0) means that the data belonging to clusters may be wrong/incorrect.

Considering the maximum silhouette score is ‘0.5962118726536606’ observed at ‘k = 3’, hence it is the optimal value of the same.

Subpart - 4

Elbow Method is used to find the optimal value of k. The plot is as follows:



The intuition is that increasing the number of clusters will naturally improve the fit (explain more of the variation) since there are more parameters (more clusters) to use, but at some point this is over-fitting, and the elbow reflects this.

The 'elbow' occurs at 'k = 3'. That is, the decrease in slope becomes less rapid at k = 3. Hence, as per the 'Elbow Method', 'k = 3' is the optimal value.

Question - 2

Exploratory Data Analysis (EDA)

The given 'fashion-MNIST dataset' is imported and read in as a pandas data frame along with appropriate column names. Statistics of the data are analyzed via '.head()', '.info()', and '.describe()'. Following observations are made:

- There are seven hundred and four(784) features and sixty-thousand(60000) data points.
- Only continuous features are available.
- The scale of continuous features is different.
- No missing (NaN) values.

The dataset is split based on 'features' and 'target'.

Subpart - 1 & 2

KMeans clustering algorithm is implemented from scratch. The class takes as input the following hyper-paramaters:

- centroids: initial cluster center points from the user as its initializations.
- n_clusters: a value 'k' from the users to give 'k' clusters.
- max_iteration: maximum iterations

The class has the following functionalities:

- Able to store the cluster centers.
- Abe to store clusters and their respective sample points.
- Able to predict cluster index for each sample point

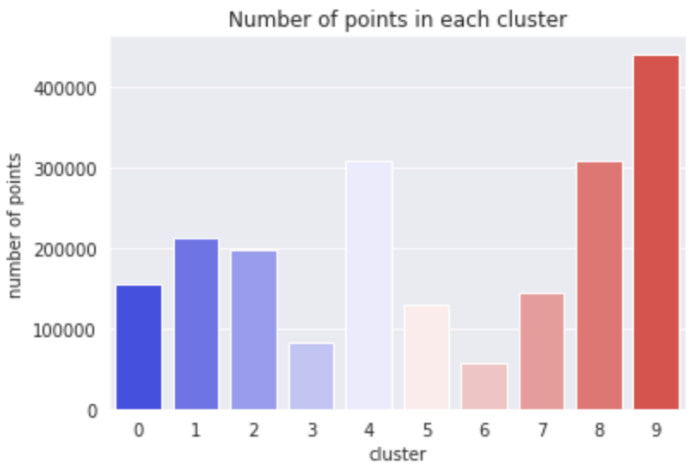
The intricate working of the KMeans algorithm (implemented) is explained as follows:

- For each iteration
    - For each sample
        - Euclidean distance is calculated of the sample from each cluster.
        - The sample is assigned to the cluster with a minimum Euclidean distance.
    - Centroids are updated based on the samples assigned to each cluster.
    - New centroids are compared with the original centroids to check if the values are changing or not. If the values are changing, the loop continues else breaks.

Subpart - 3

The KMeans model is trained on f-MNIST data with ‘k = 10’ and ‘10 random 784-dimensional points (in input range)’ as initializations’. The number of points in each cluster is tabulated and visualized below:

Cluster	Number of points
0	156383
1	212294
2	198577
3	81937
4	308026
5	129245
6	57670
7	144602
8	309231
9	442035



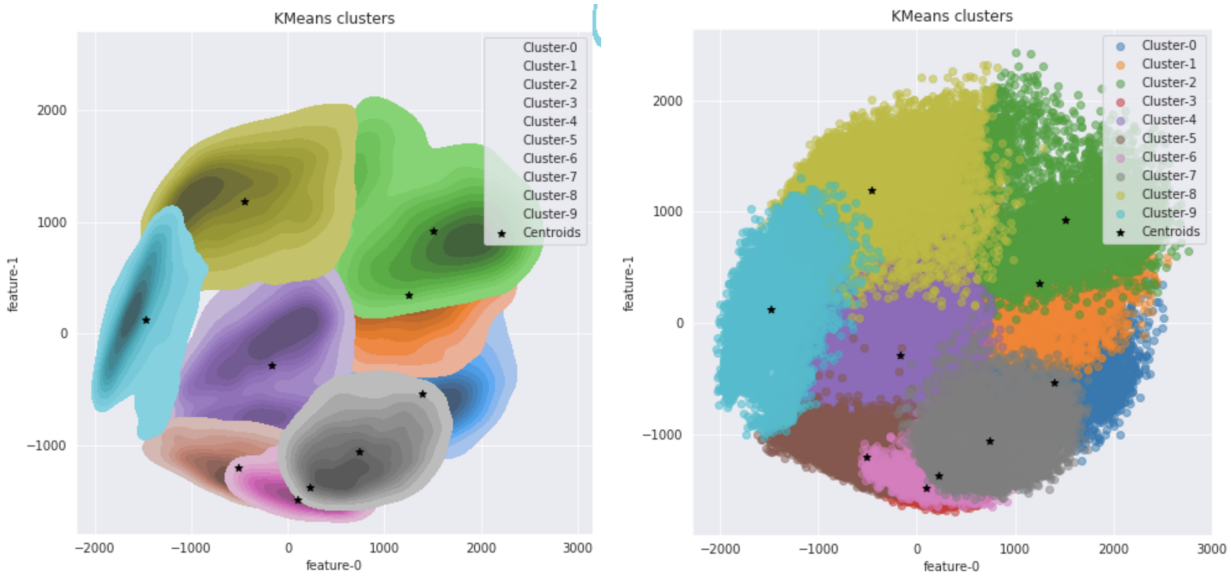
Subpart - 4

Visualisation of the cluster centers of each cluster as 2-d images of all clusters is as follows:

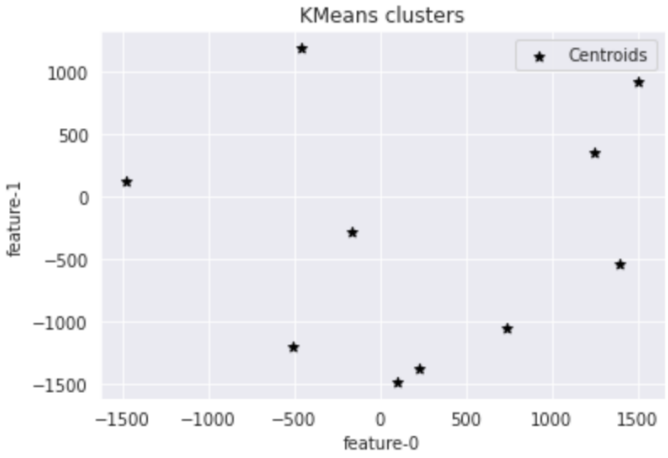
Plot I

Plot of 'feature 1' against 'feature 0' with ten(10) clusters and their respective cluster centers.

NOTE: Considering the given fashion-MNIST dataset is 784-dimensional, it is not possible to plot a two-dimensional plot with all(784) features present. Hence, for simplicity dimension reduction technique (PCA) is applied to the dataset X and centroids.

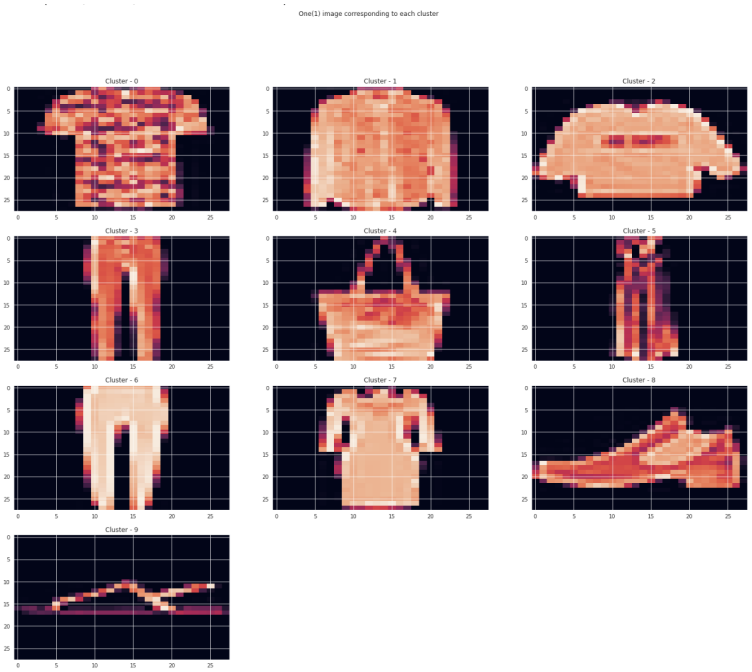


Plot II
Plot of ‘feature 1’ against ‘feature 0’ with ten(10) cluster centers.
NOTE: Considering the given fashion-MNIST dataset is 784-dimensional, it is not possible to plot a two-dimensional plot with all(784) features present. Hence, for simplicity dimension reduction technique (PCA) is applied to the dataset X and centroids.

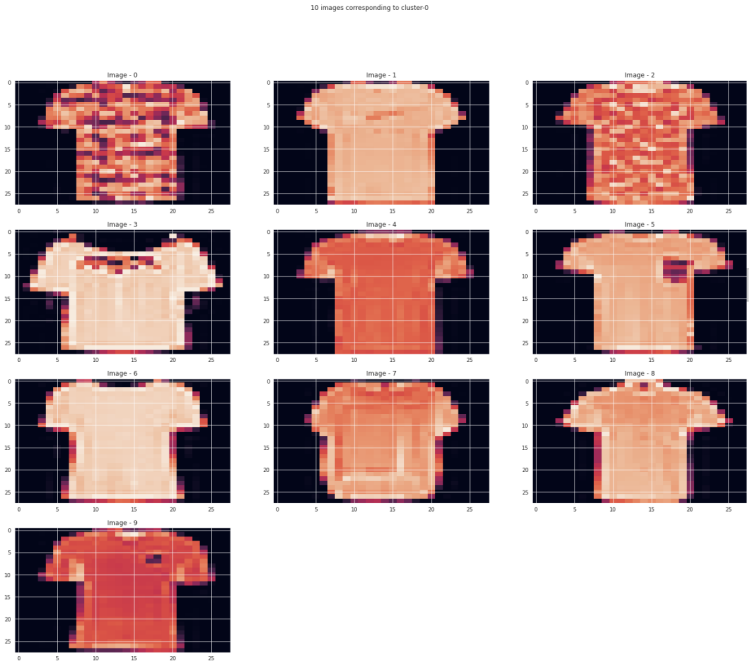


Subpart - 5
Visualization of ten(10) images corresponding to each cluster is as follows:

Plot I
One(1) image from each cluster.



Plot II
Ten(10) images corresponding to each cluster.
The sample for cluster-0 is as follows:

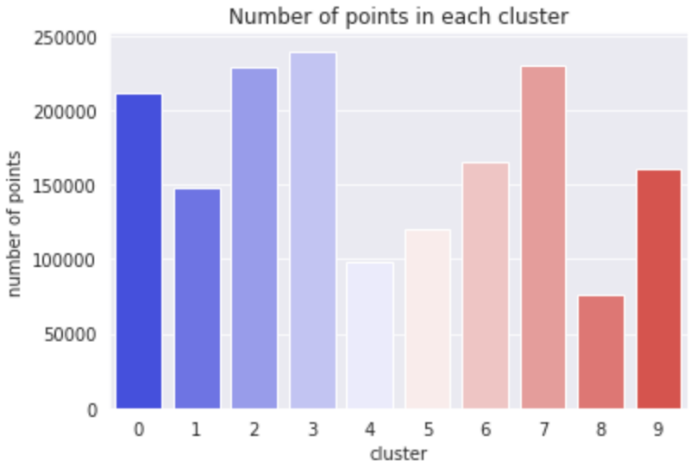


Subpart - 6

Another KMeans model is trained with '10 images from each class' as initializations. The number of points in each cluster is tabulated and visualized below:

Cluster	Number of points
0	211561
1	148230
2	229576
3	239805
4	98448
5	120762
6	165800
7	229823
8	75788

9	160207

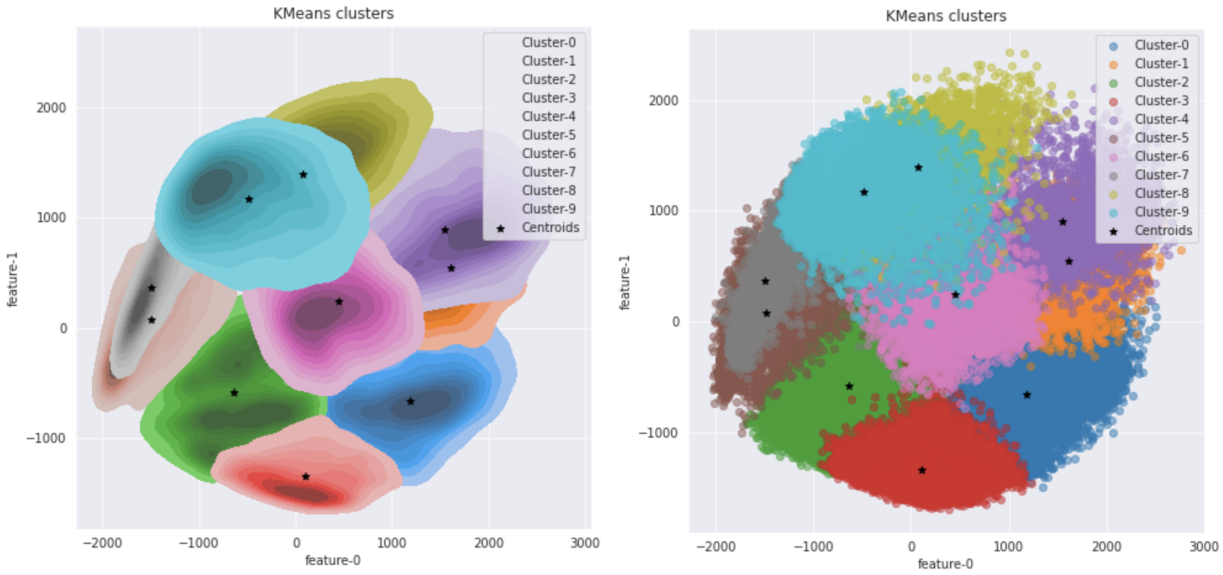


Visualisation of the cluster centers of each cluster as 2-d images of all clusters is as follows:

Plot I

Plot of ‘feature 1’ against ‘feature 0’ with ten(10) clusters and their respective cluster centers.

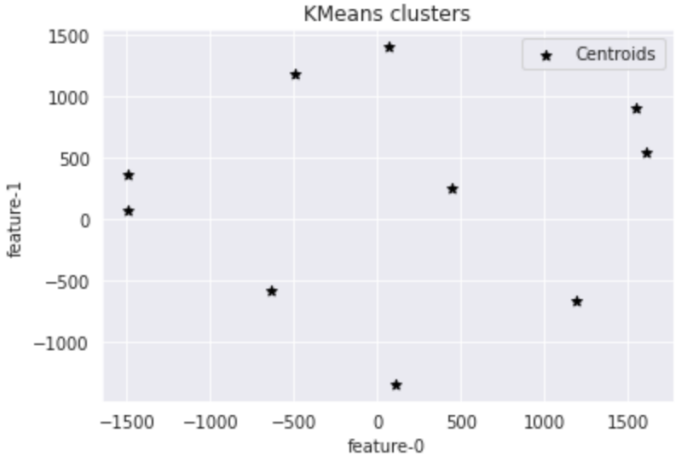
NOTE: Considering the given fashion-MNIST dataset is 784-dimensional, it is not possible to plot a two-dimensional plot with all(784) features present. Hence, for simplicity dimension reduction technique (PCA) is applied to the dataset X and centroids.



Plot II

Plot of ‘feature 1’ against ‘feature 0’ with ten(10) cluster centers.

NOTE: Considering the given fashion-MNIST dataset is 784-dimensional, it is not possible to plot a two-dimensional plot with all(784) features present. Hence, for simplicity dimension reduction technique (PCA) is applied to the dataset X and centroids.

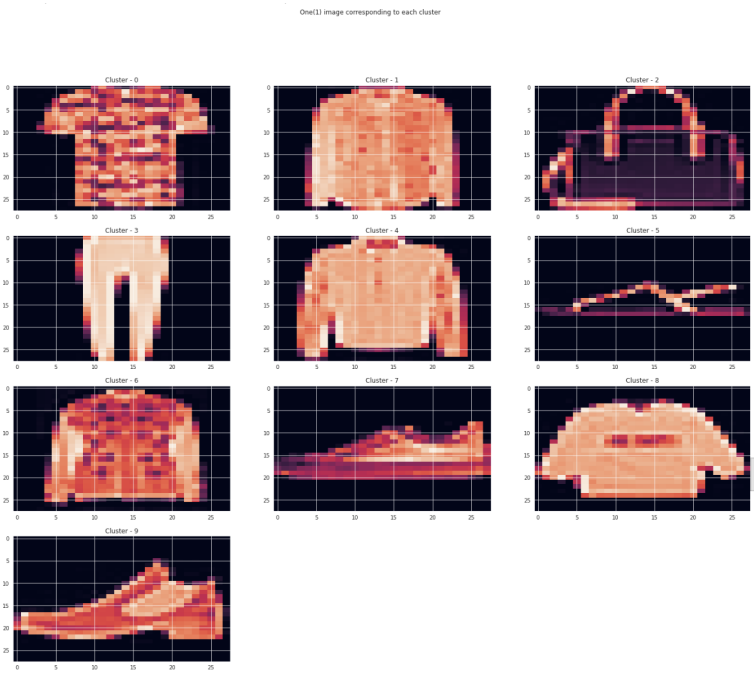


Subpart - 7

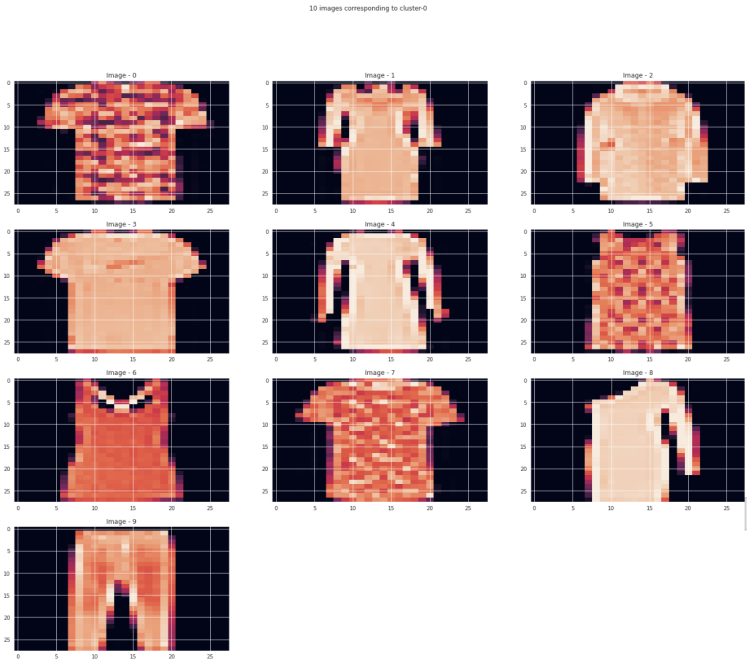
Visualization of ten(10) images corresponding to each cluster is as follows:

Plot I

One(1) image from each cluster.



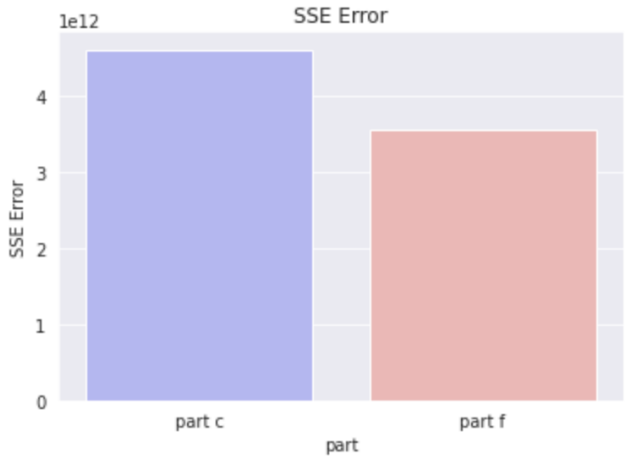
Plot II
Ten(10) images corresponding to each cluster.
The sample for cluster-0 is as follows:



Subpart - 8

Clusters of parts c and f are evaluated with the Sum of Squared Error (SSE) method. The respective scores and tabulated and visualized below:

Part	Sum of Squared Error (SSE)
c	4614390458473.608
f	3568268078025.8765



Observations and Conclusions
From the tabulated data as well as the barplot above, it can be observed that ‘part-f’, that is ‘10 images from each class as initialization’ performs better clustering. This can be justified as follows:
In part-c, where we take ‘10 random 784-dimensional points (in input range)’ as initialization the initial centroids (given by the user) may as well all belong to the same class in worst case scenario (because of the random nature of the technique). This results in poorly formed clusters that require a higher number of iterations to form separate clusters and that too are not guaranteed to yield good results.
However, in part-f, where we take ‘10 images from each class’ as initialization the initial centroids all belong to different and unique classes. This results in better-formed clusters that require a lower number of iterations to form separate clusters. Furthermore considering the centroids were from different and unique classes initially, they are likely to contain sample points of that particular class only.

Question - 3

Subpart - 1

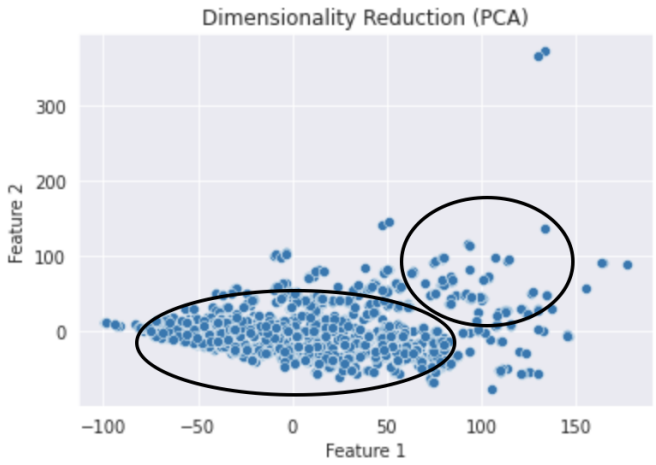
A joint dataset containing 1500 images from the given ‘Yes’ folder, and 1500 images from the given ‘No’ folder (total of 3000 images) is constructed externally and uploaded on Google Drive. The images from the said folder are read in and resized (100 x 100) using the cv2 library, and stored as a list. The list is converted to a NumPy array, resized (3000, 10000), and finally converted to a Pandas Dataframe with appropriate column names and class labels. Statistics of the data are analyzed via ‘.head()’, ‘.info()’, and ‘.describe()’. Following observations are made:

- There are ten thousand(10000) features and three thousand(3000) data points.
- Only continuous features are available.
- The scale of continuous features is different.
- No missing (NaN) values.

The features are normalized via StandardScaler() so that the scale of each variable will be the same.

Subpart - 2

Principle component analysis (PCA) dimension reduction technique with ‘n_components = 2’ (for two-dimensional data) is used. Visualization of the data after implementing the mentioned technique is as follows:

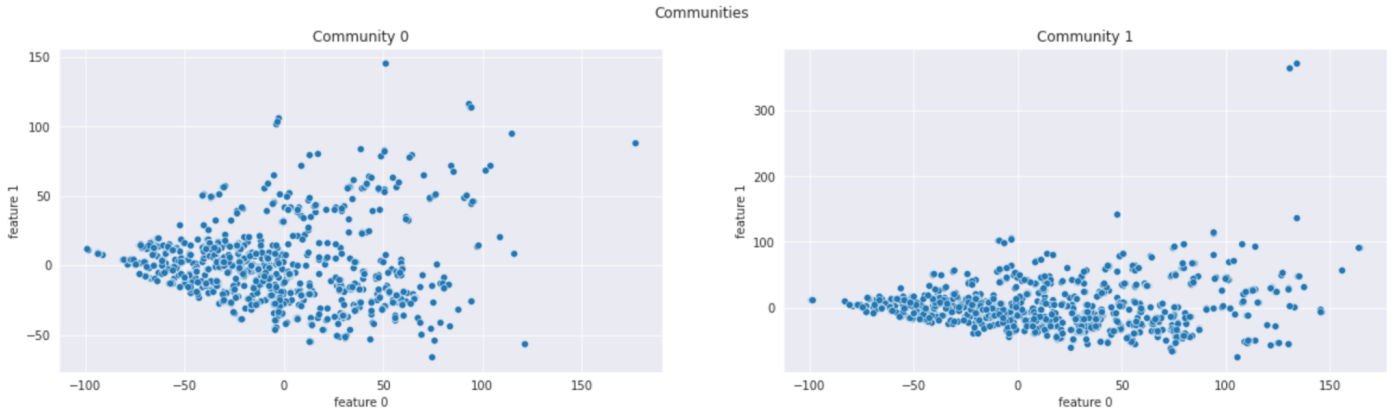


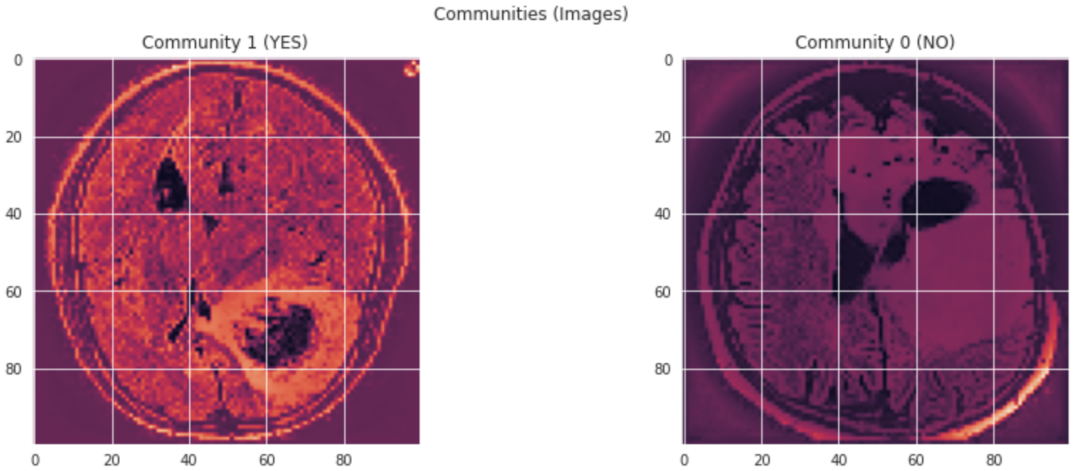
From the above scatter-plot it can be seen that there are two communities:

- Lower left
- Upper right

Subpart - 3

Visualization of the communities from ‘Part A’ is as follows:





Subpart - 3

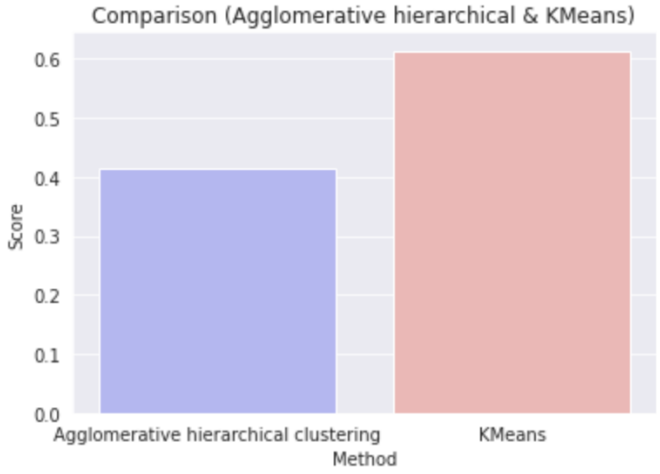
Agglomerative hierarchical clusterings (using sklearn) is applied on the mentioned dataset.

Subpart - 4

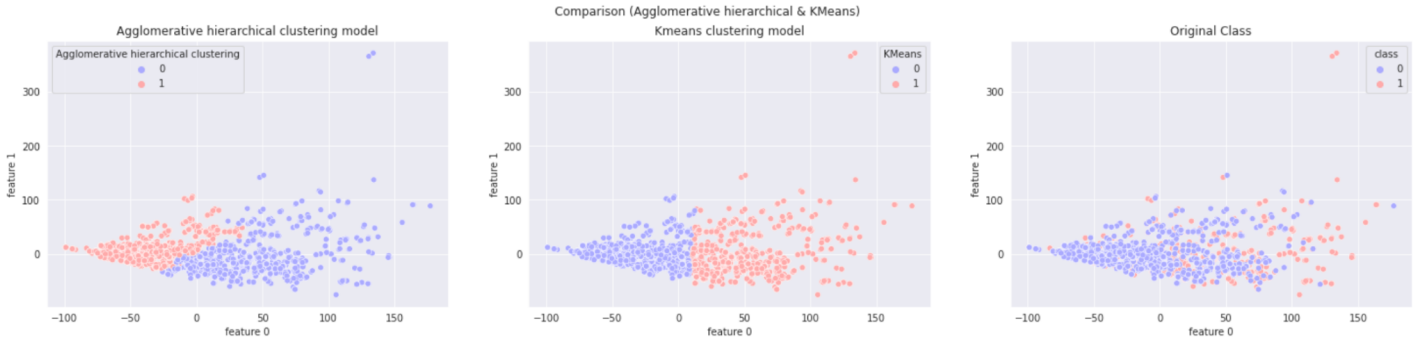
KMeans clusterings (using sklearn) are applied to the mentioned dataset.

Comparison based on accuracy score is tabulated and visualized below:

Clustering Technique	Accuracy Score
Agglomerative hierarchical clustering	0.41267
KMeans clustering	0.61334



Comparison based on clustering is visualized below:



Observations

Based on the above-mentioned data, the following observations can be made:

- KMeans clustering has a better accuracy score (61.34%).
- KMeans divides the given samples into strictly two halves(left and right) from the middle.

- Agglomerative hierarchical clustering divides the given samples along with a diagonal (left upper and right lower).

Conclusions

It can be concluded that though KMeans has a better accuracy score and separated unique clusters, clustering implemented by Agglomerative hierarchical clustering is more efficient in the sense that it resembles the original classification. This can be justified by the fact that though KMeans cluster based on distance from the centroid of the cluster sample points, Agglomerative hierarchical clustering follows a bottom-up approach. A structure that is more informative than the unstructured set of clusters returned by flat clustering.