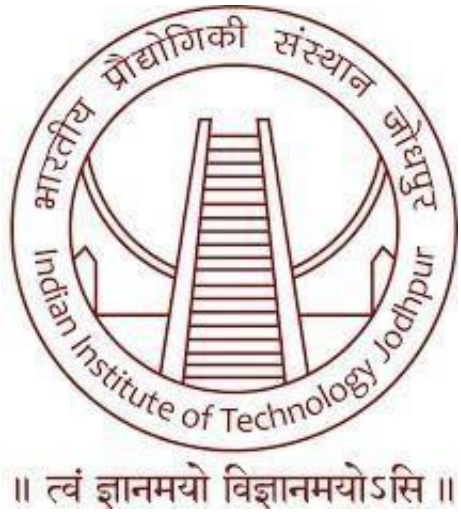
Pattern Recognition and Machine Learning

Indian Institute of Technology, Jodhpur



LAB 6

Report

Harshil Kaneria

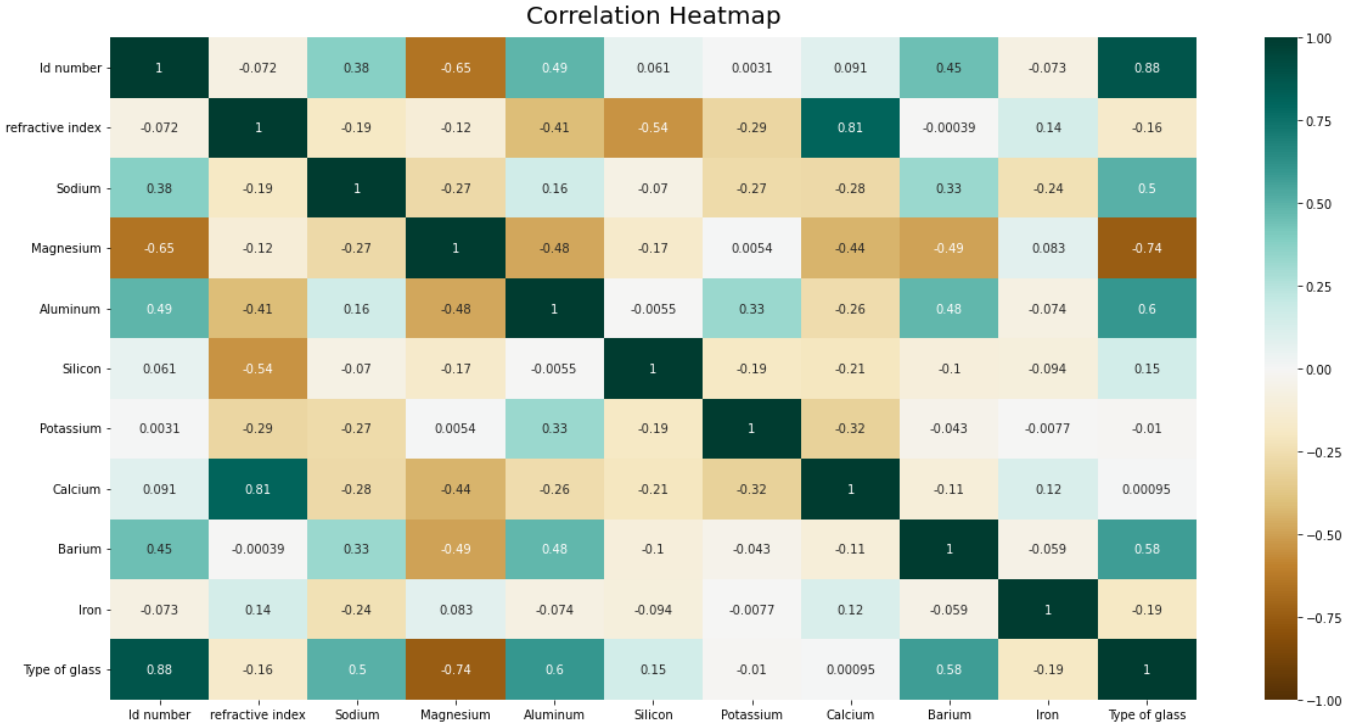
Department of Computer Science, IIT Jodhpur

March 7, 2023

1)

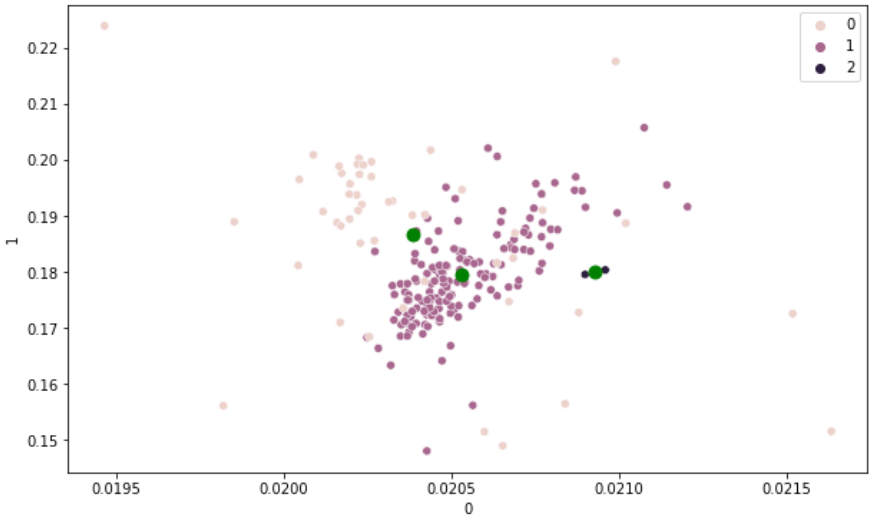
Pre-Processing, Visualisation and Labelling:

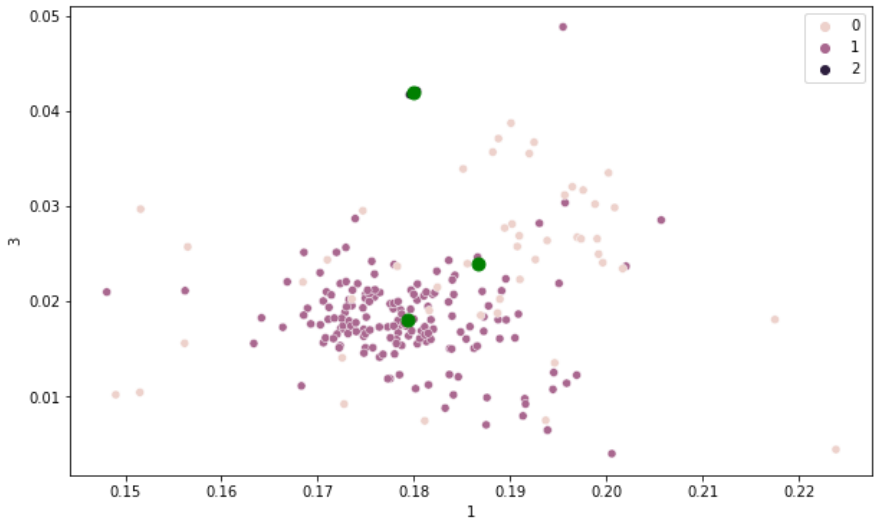
- Labelled the features and target columns in dataset itself.
- Plotted Correlation heatmap and found that there is high correlation between “Calcium” and “refractive index” features. Also, the id number is unique for all the instances/data-points. So, dropped the “Calcium” and “Id number” columns for further question.



- Visualised the data by plotting some features with others in a 2D scatter plots.

a) Built a **K-means Clustering Algorithm** and implemented it with using k=3. Also, plotted various 2D scatter plots of dataset choosing 2 random features and plotted the cluster centres along with it. Following are some of the scatter plots:





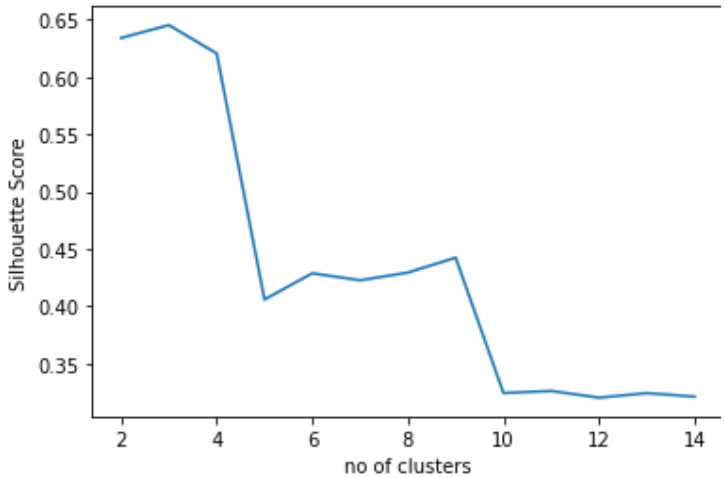
Green points are the cluster centres. The first plot is between “feature 0” and “feature 1”. The second plot is between “feature 1” and “feature 3”.

b) Silhouette Score:

Calculated silhouette score for different values of k ranging from (2,15). The following are the scores:

```
The silhouette score for k = 2  is : 0.633911788706452
The silhouette score for k = 3  is : 0.6451688601633173
The silhouette score for k = 4  is : 0.6203941177676084
The silhouette score for k = 5  is : 0.4057598889079089
The silhouette score for k = 6  is : 0.4286472532333569
The silhouette score for k = 7  is : 0.4225192167877065
The silhouette score for k = 8  is : 0.4292743631441461
The silhouette score for k = 9  is : 0.44224660327488274
The silhouette score for k = 10  is : 0.3241365654137125
The silhouette score for k = 11  is : 0.3259652575796045
The silhouette score for k = 12  is : 0.3201256620412471
The silhouette score for k = 13  is : 0.32410120610960796
The silhouette score for k = 14  is : 0.32102774258777383
```

The corresponding graph of silhouette score vs k is as follows:



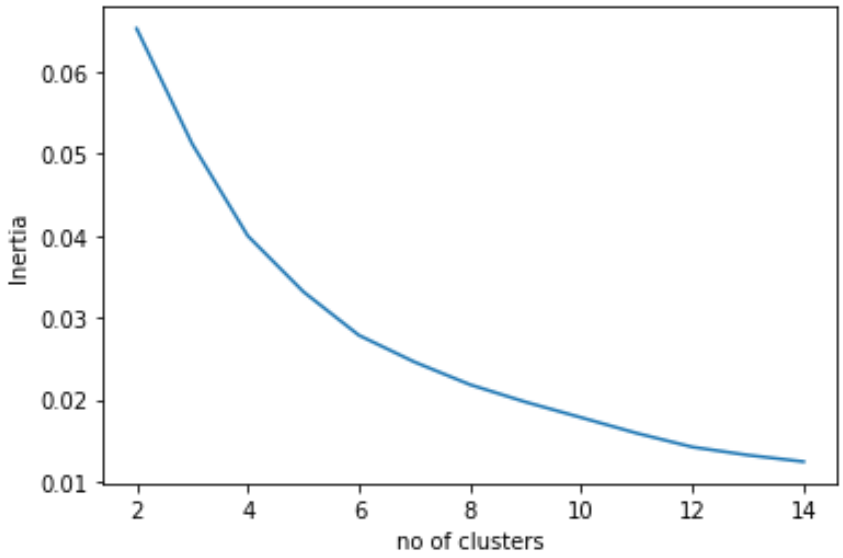
The maximum Silhouette score is 0.6451 for k=3.

The silhouette score ranges from -1 to 1, with a higher score indicating more effective clustering. A score close to one implies that cluster samples are densely packed, whereas a score close to zero suggests that samples lie along the boundary of two clusters.

c) Elbow Method:

We have utilised several inertia values for various K values. Inertia is the sum of the squared distances between cluster centroids and sample points. A model with low values of K and Inertia is a good one. As one of the properties of the built-in K-Means algorithm in 'sklearn', we can obtain the inertia value directly.

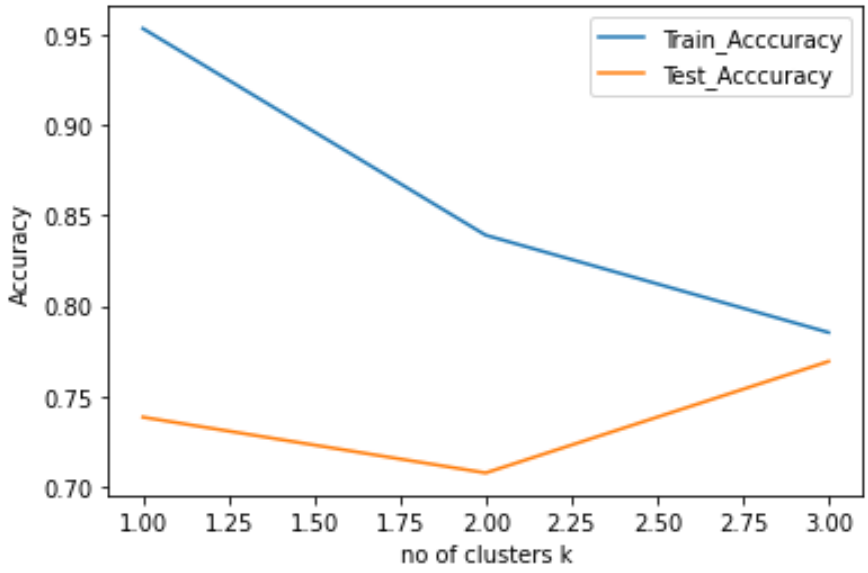
The following is the plot of "no of clusters: k" vs "Inertia":

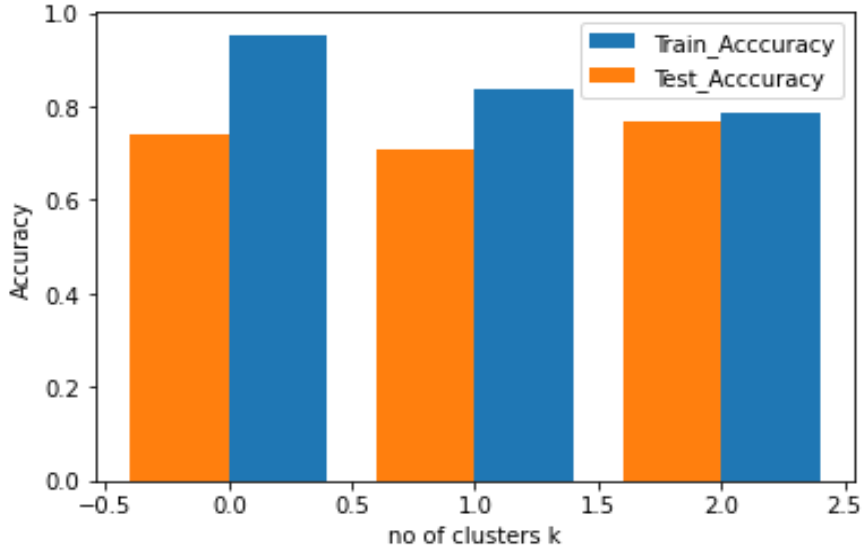


We can see that the elbow method didn't give effective value of k but we can consider k = 4 or k = 5 to be optimal value for clustering according to above graph.

d) Bagging Classifier with KNN as base model:

Implemented Bagging classifier with KNN as base model for k = 1,2 and 3. The following are the plots of comparisons:





Observations:

- Because there is a reduction in the amount of train error when we increase the value of K, the KNN classifier's bias will increase as we do so. And as a result, the gap between the training error and the test error is also getting less, which means that the variance is getting smaller as well.
- Due to the fact that KNN has a high variance for low values of k, the application of bagging will result in a reduced variance. Also, the precision of each and every model will advance. Yet, bagging has no effect on bias, and as a result, the model's inherent bias is not diminished as a result of applying bagging.

2)

Flattening and Pre-processing Dataset:

Flattened the data and data is already pre-processed.

a) Implemented K-means Clustering Algorithm from Scratch:

The class name is "KMeans_Cluster".

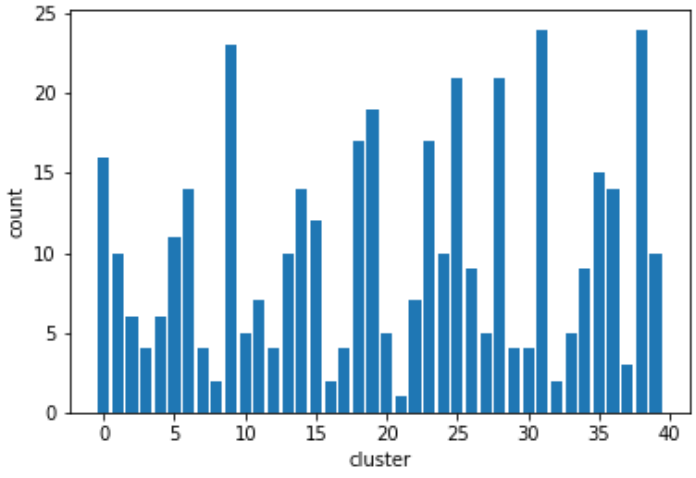
It contains following methods:

- **__init__** - constructor of the class
- **_norm** – calculates the norm of two vectors
- **fit** – Trains data and forms clusters
- **predict** – Predicts cluster of input data frame

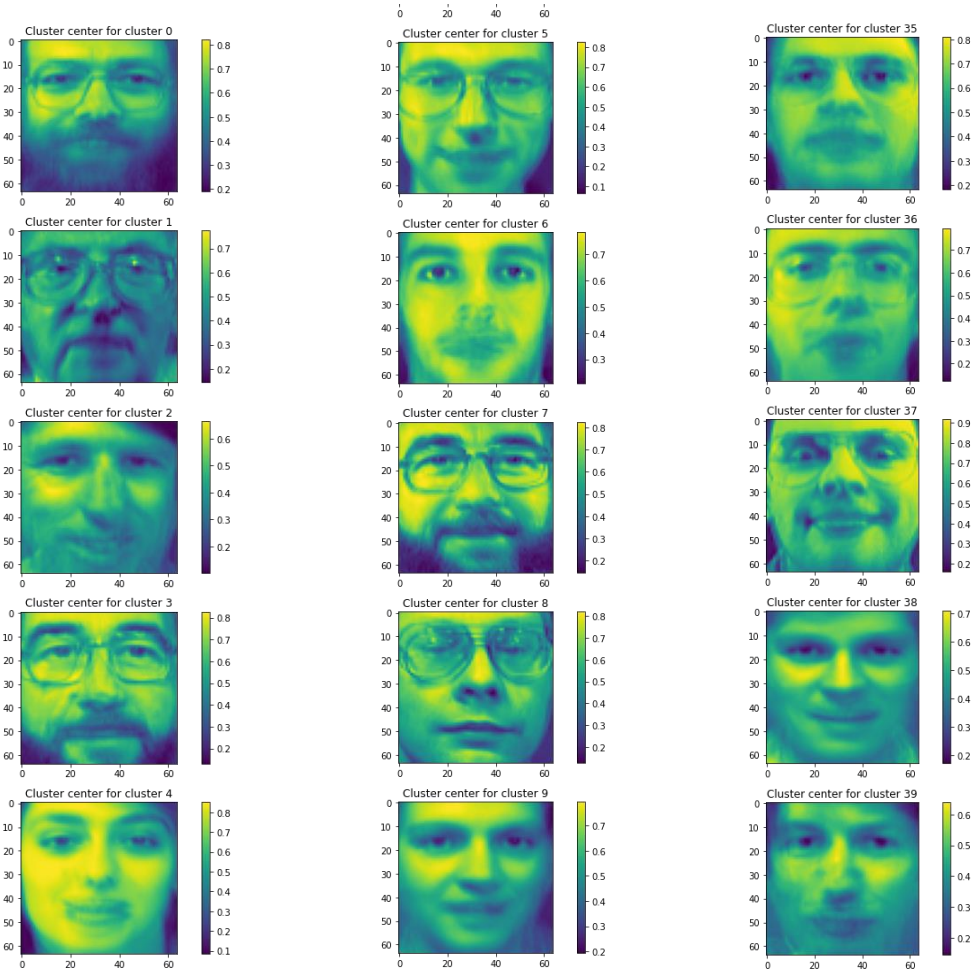
b) The class does the following tasks:

- Stores the cluster centres in class variable **self.centriods**
- Takes input value of **k** – number of clusters
- Takes initial centres as a input data frame from user and is stored in class variable **self.centriods**
- The fit method stops iterating when it reaches value **max_iter** i.e. either given input by user or its default value is max_iter = 200 or it stops iterating when new centres are close to each other with tolerance of 10^(-4).

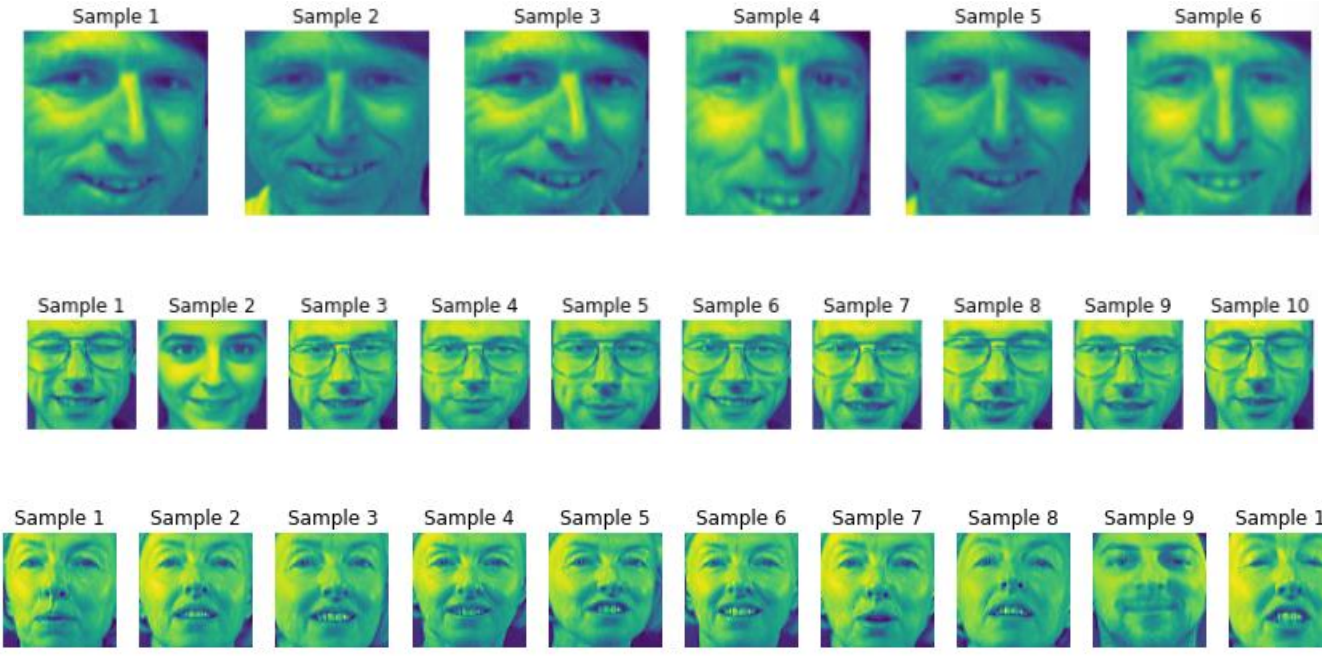
c) Trained K-means model for Olivetti dataset by choosing 40 points as initial centres randomly from dataset. The following is the plot of count of points in each cluster:



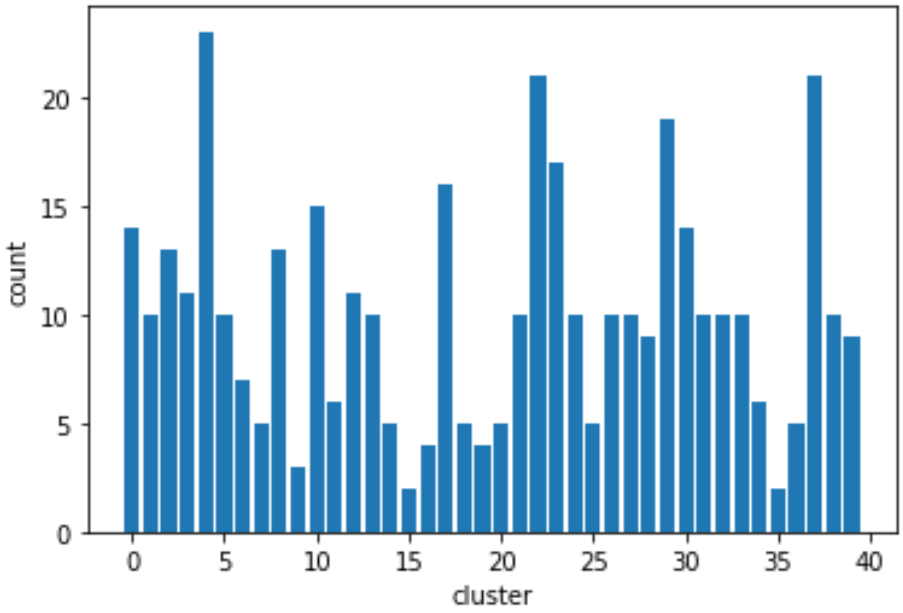
d) Some of the cluster centres for above trained model:



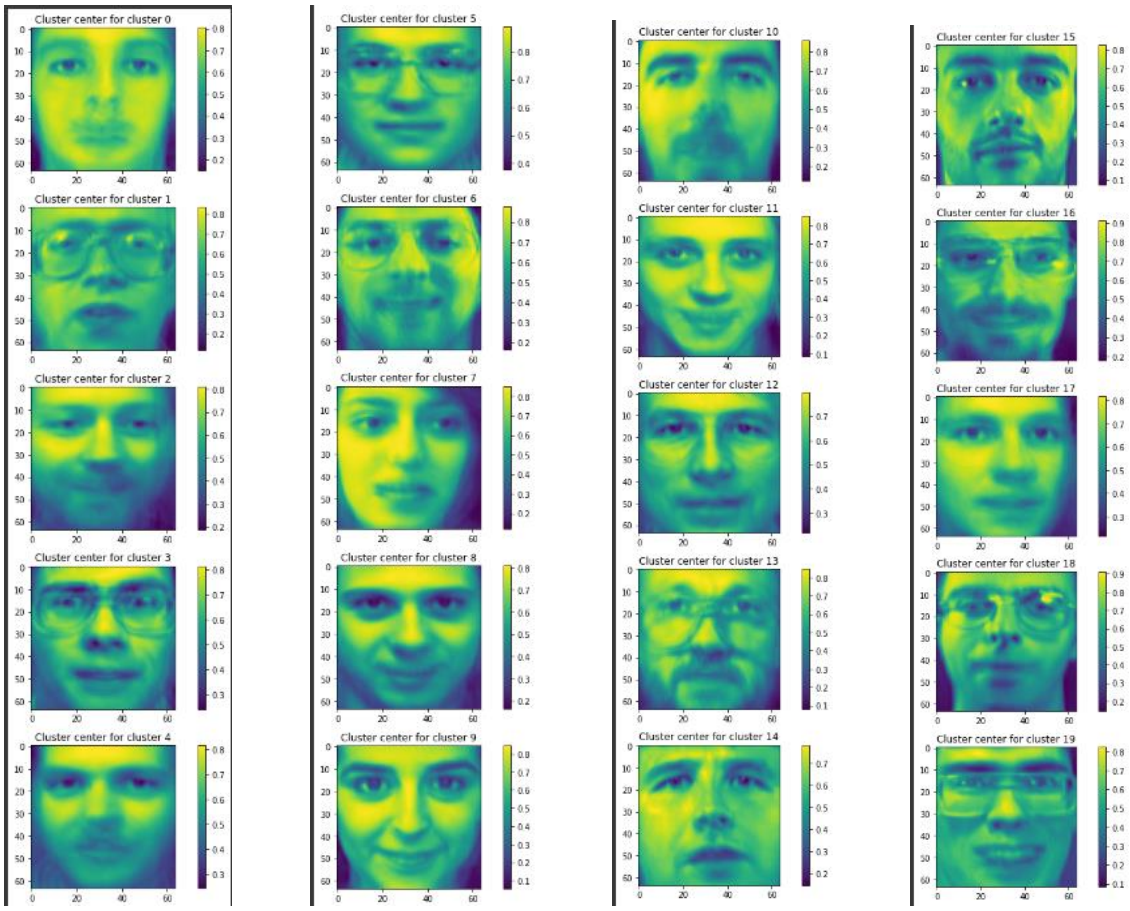
e) Since I am not getting 10 images in each cluster some of the Images of a particular cluster:



f) Trained a K-Means Clustering model by giving initial centres one of each class as input. The following is the plot of no of points in each cluster:



The following images are some of the cluster centres:



g) Some of the Images of a particular cluster:

h) The Sum of Squared Error (SSE) for each of the two models trained above is as follows:

```
SSE for part 1:  197691.42611971736
SSE for part 2:  162071.51258567243
```

The SSE score of second model is less than first model so the second cluster is better than first.

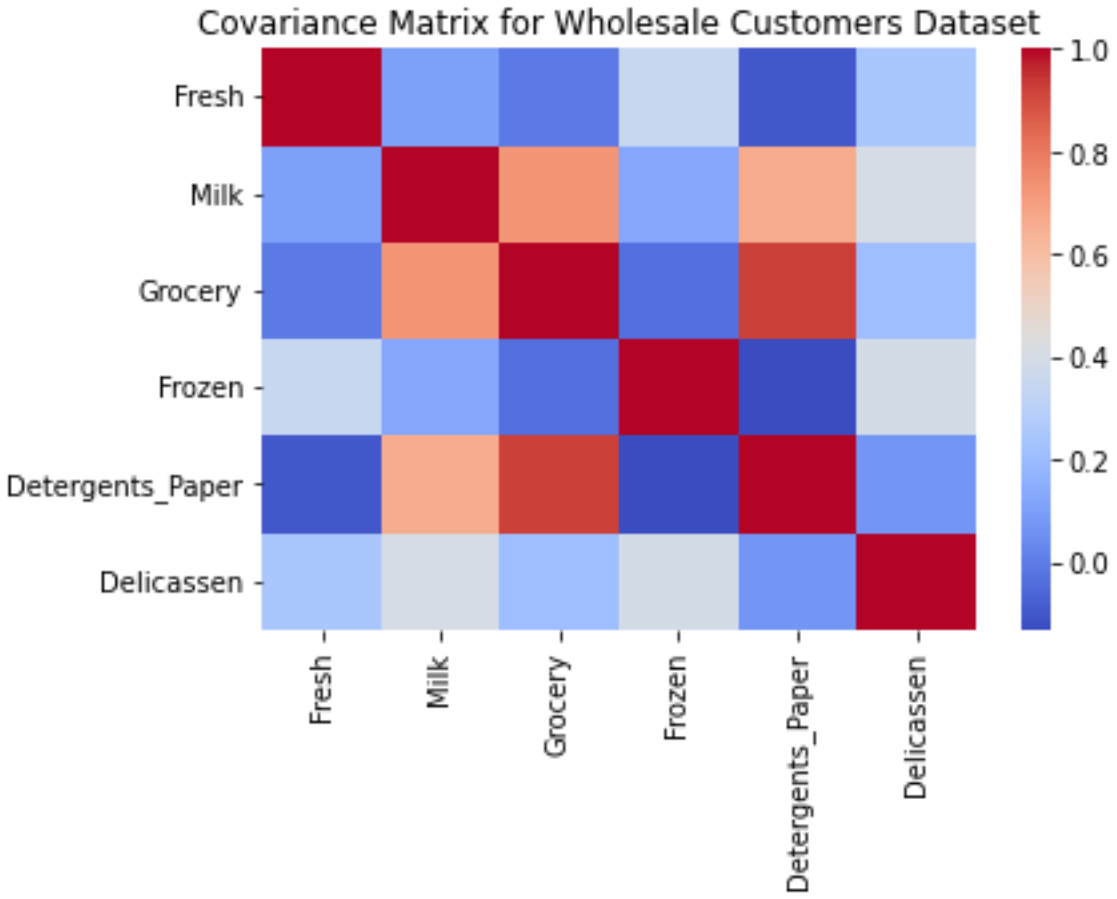
3)

a) Pre-Processing and Scaling:

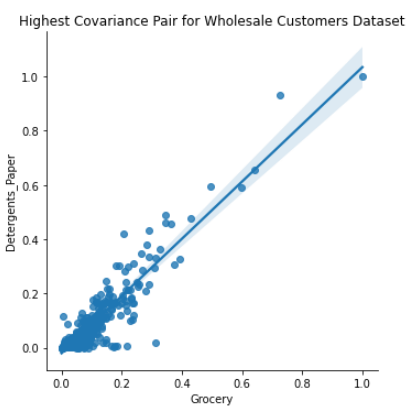
- In pre-processing the dataset, I have removed the features “Channel” and “Region” as they are categorical data. Also, plotted the correlation heatmap to check for correlation between different features.
- In scaling , I have Max-Min scaled the dataset. This scales the data in range [0,1] both inclusive.

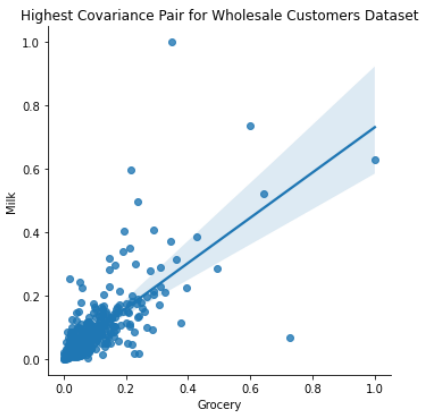
b) Visualising Outliers:

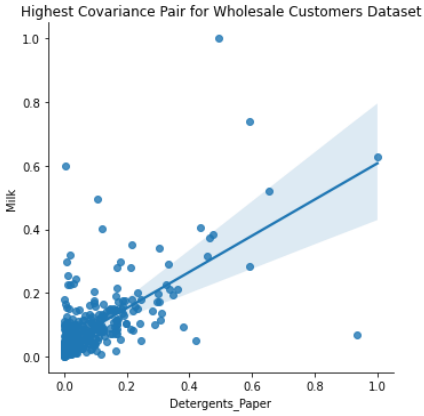
The following is the correlation heatmap after pre-processing and scaling step is done:

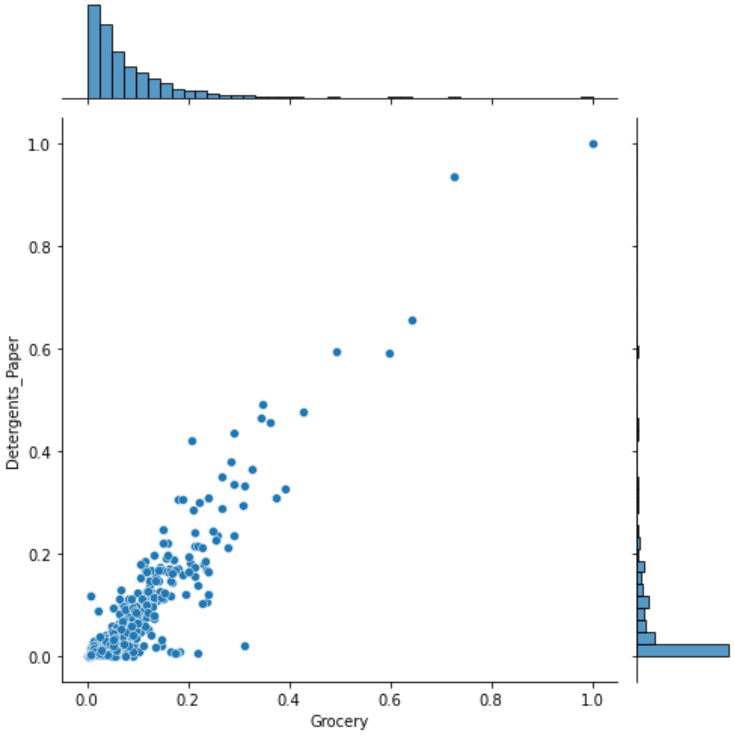


As we can see the correlation between features “Grocery” and “Detergents Paper” is highest. Next comes the correlation between features “Grocery” and “Milk” then “Detergent Paper” and “Milk”. So to visualise the outliers the features with maximum correlation is used.







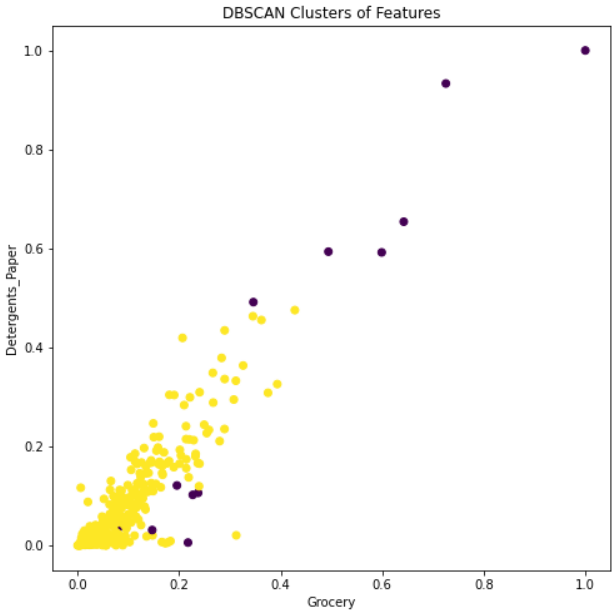


Plot: 'Grocery' vs 'Detergent_Paper'

We can clearly see the outliers from the graph shown above. The points in upper right part of the plot are outliers from visual perception.

c) DBSCAN Clustering:

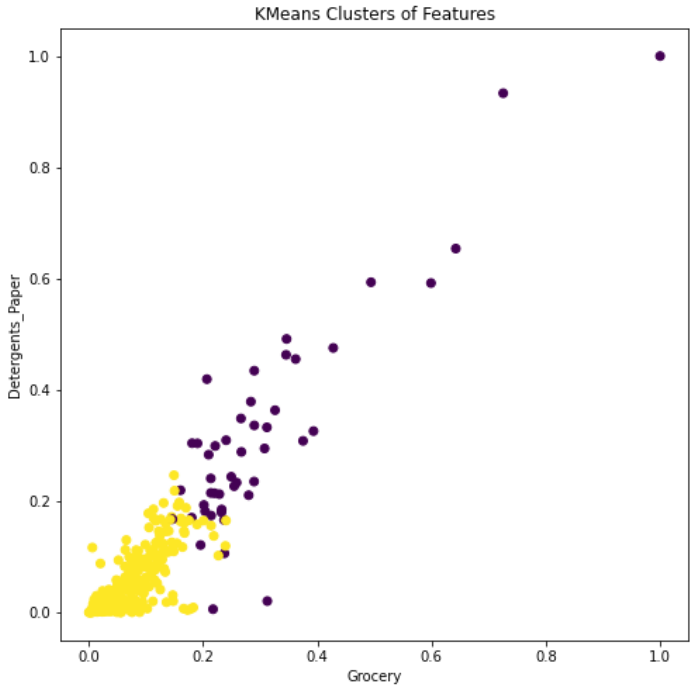
Trained a DBSCAN Clustering Model on dataset given and the below shown is the graph to visualise outliers:



Plot DBSCAN: 'Grocery' vs 'Detergent_Paper'

d) K-means Clustering on the same dataset:

Trained a K-Means Clustering Model on dataset given and the below shown is the graph to visualise outliers:

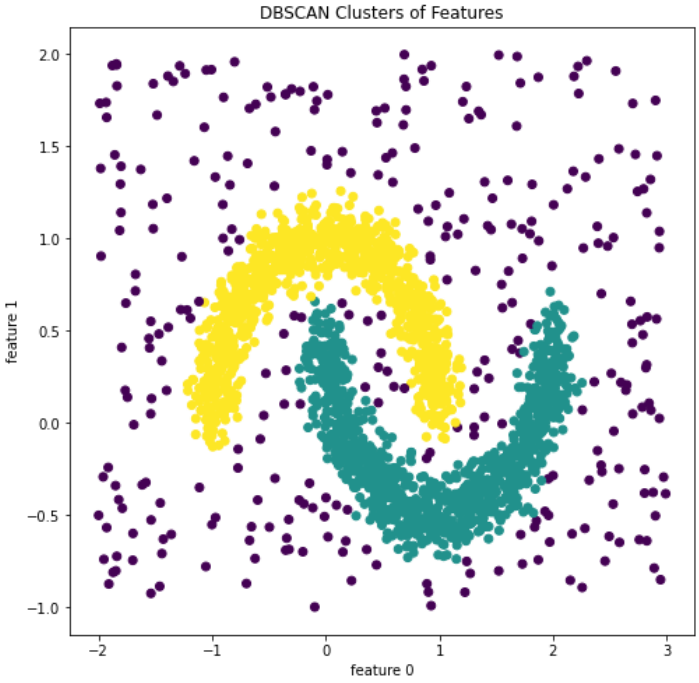


Plot K-Means: 'Grocery' vs 'Detergent_Paper'

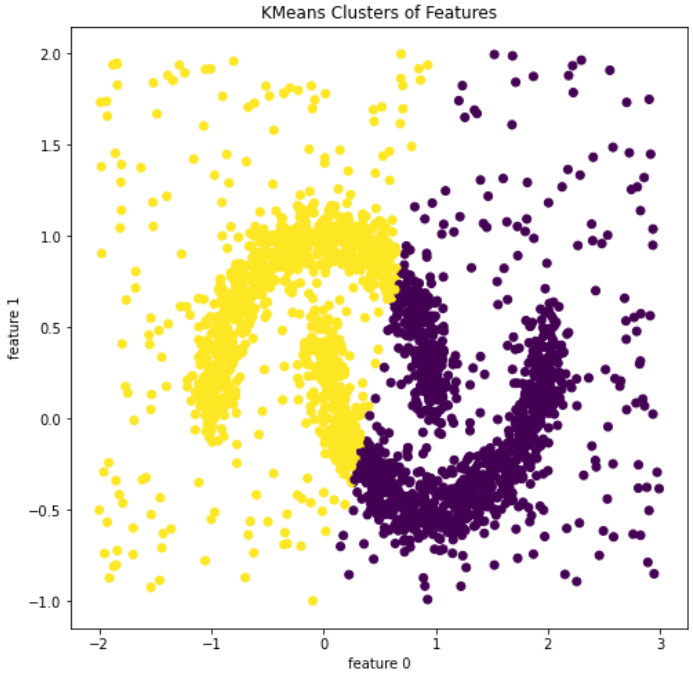
Observations:

We are able to see that DBSCA is able to categorise outliers as having a value of -1, whereas k-Means is unable to do so. K-Means is unable to cluster the two moons, but DBSCAN, after being given the necessary values for the parameters, is successful in doing so. We are able to reach the conclusion that DBSCAN, in contrast to K-Means, is not influenced by the presence of extreme examples. It is possible for DBSCAN to group together points that are located in a region with a random shape.

e) DBSCAN vs K Means on Moons Dataset:



DBSCAN Clusters: 2 clusters + outliers



K-Means Clusters: k=2

We can observe that DBSCAN has ability to classify Clusters and outliers separately and K-means cannot do that. This makes DBSCAN a better clustering algorithm because real world problems have outliers in considerable amount.