DATA MINING AND NEURAL NETWORKS

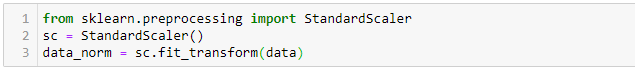
COMPUTATIONAL TASK 2

**ROSHAN RAJ RAMESH (rrr13)**

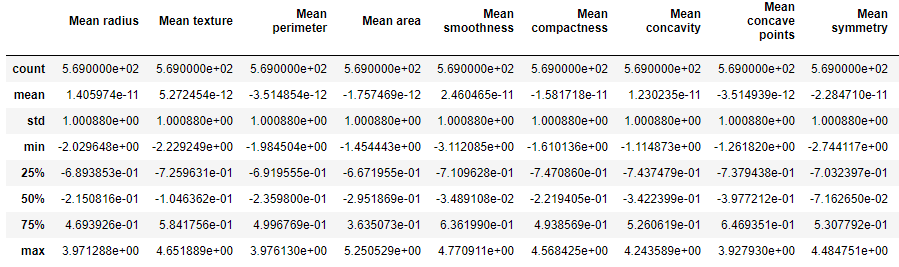
**209030158**

**Question 1:**

Centralise and normalize the data.

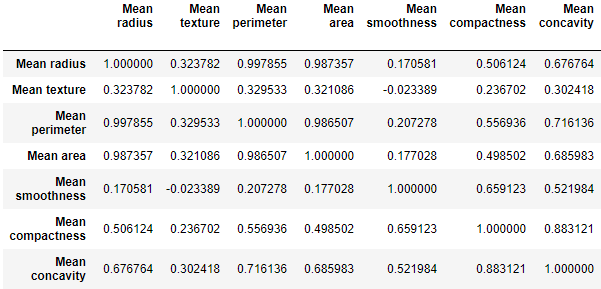


We are using StandardScaler method from sklearn.preprocessing library to normalize the dataset resulting in mean = 0 and std = 1.



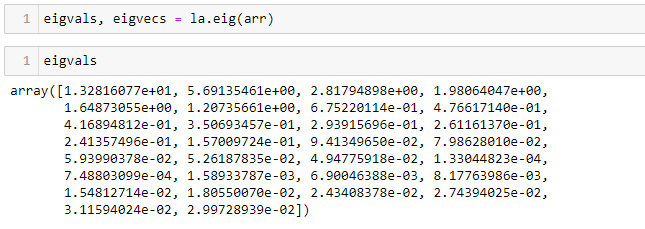
Similarly, for all 30 attributes is done.

Correlation matrix is calculated by corr() method.

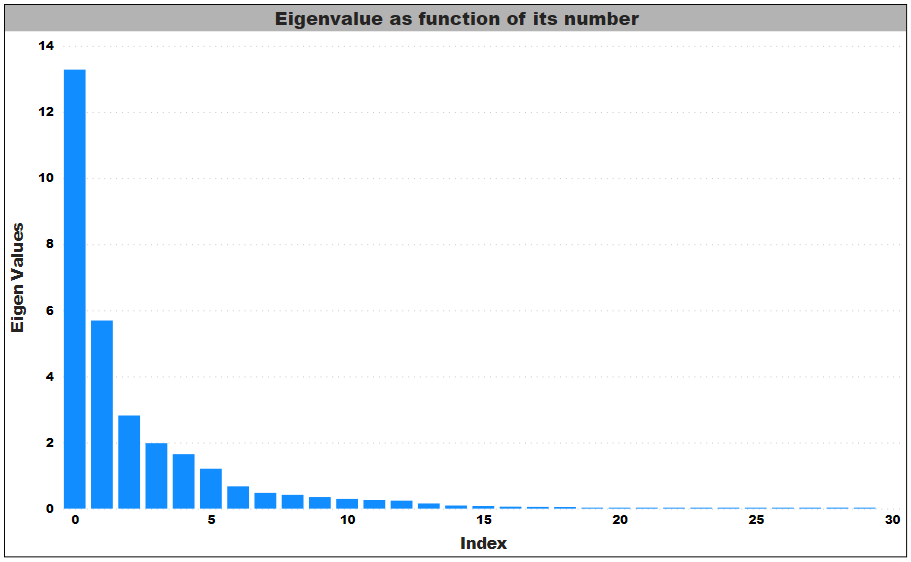


Gilmse of correlation matrix

By using linalg method from numpy library, eigen values are extracted from the correlation matrix.



With the results obtained above (Eigen values), a plot has been created w.r.t to the function of its number.



Keiser Rule and Conditional number rule are used to determine the major components and retains the major component with their respective condition.

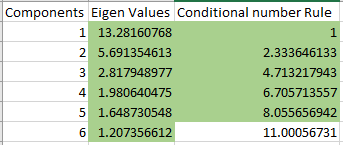
|  |  |
| --- | --- |
| Keiser Rule |  |
| Conditional Number Rule |  |

Keiser Rule:

|  |  |
| --- | --- |
| 1 | 13.281 |
| 2 | 5.691 |
| 3 | 2.817 |
| 4 | 1.980 |
| 5 | 1.648 |
| 6 | 1.207 |

Conditional Number Rule:

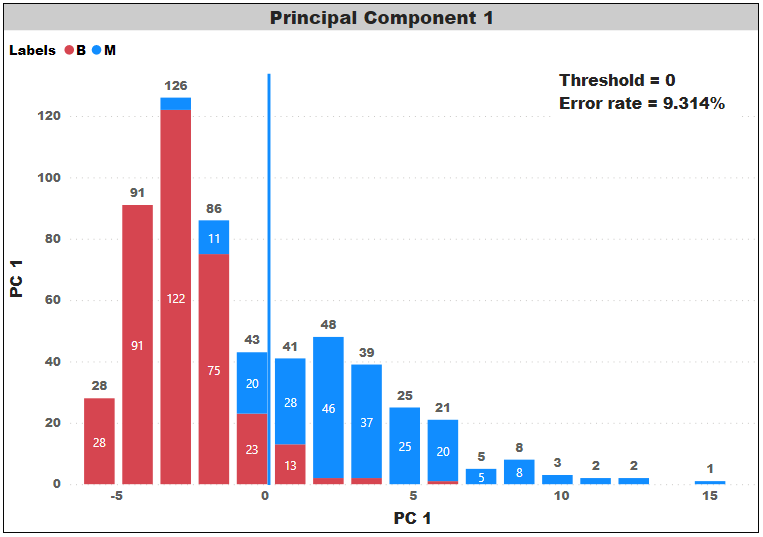
|  |  |
| --- | --- |
| 1 | 1 |
| 2 | 2.333 |
| 3 | 4.713 |
| 4 | 6.705 |
| 5 | 8.055 |

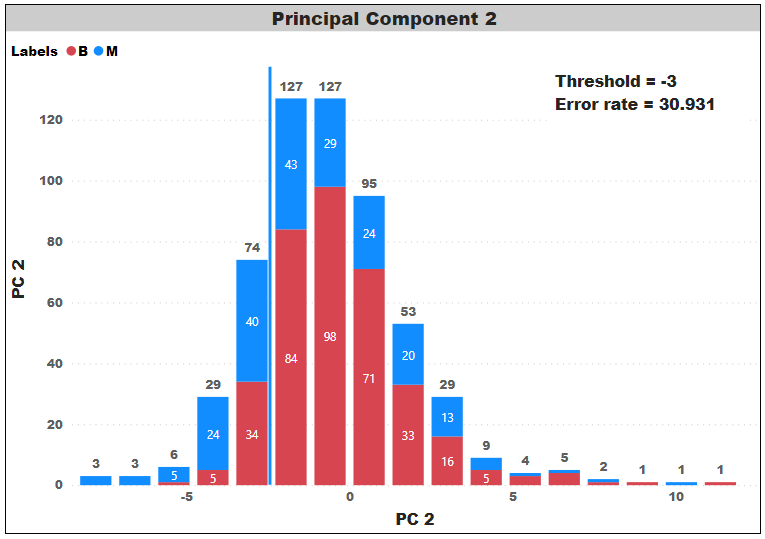


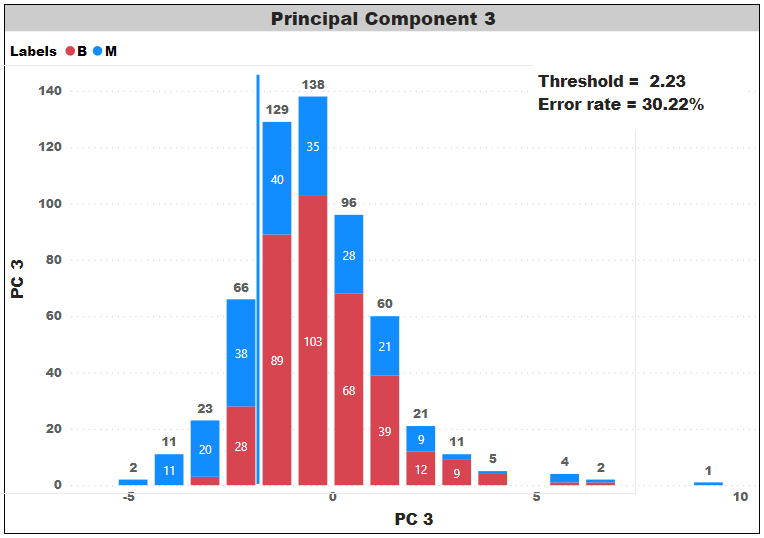
**Question 2**

Following is the result obtained from one attribute predictor from CT1 Assignment.

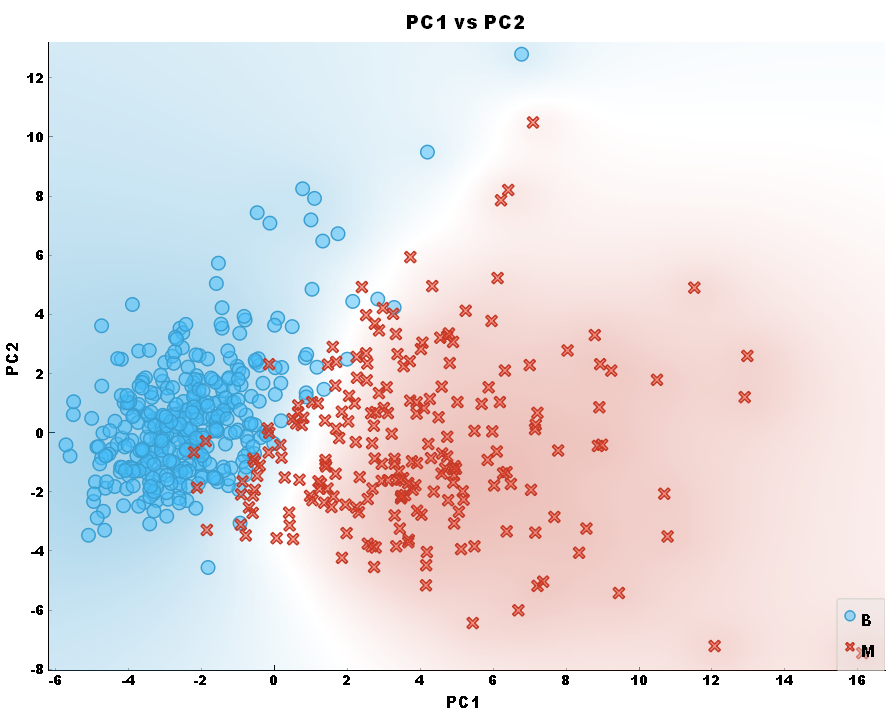
|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Threshold** | **Error** | **Prediction Ability** |
| 1 | 13 | 28% | 16th |
| 2 | 19 | 41% | 26th |
| 3 | 85 | 9.83% | 5th |
| 4 | 380 | 15% | 8th |
| 5 | 0.10 | 33% | 21st |
| 6 | 0.08 | 20% | 13th |
| 7 | 0.04 | 13.5% | 7th |
| 8 | 0.04 | 9.8% | 4th |
| 9 | 0.19 | 29.5% | 19th |
| 10 | Could not find\* | NA |  |
| 11 | 0.10 | 18.6% | 12th |
| 12 | Could not find\* | NA |  |
| 13 | 0.75 | 18.4% | 11th |
| 14 | 6.5 | 11.8% | 5th |
| 15 | Could not find\* | NA |  |
| 16 | 0.02 | 35.38% | 24th |
| 17 | 0 | 34.38% | 22nd |
| 18 | 0.01 | 34.56% | 23rd |
| 19 | Could not find\* | NA |  |
| 20 | 0 | 40.85% | 25th |
| 21 | 13.5 | 16.26% | 9th |
| 22 | 27 | 29.43% | 18th |
| 23 | 90.5 | 8.22% | 2nd |
| 24 | 592 | 8.43% | 3rd |
| 25 | 0.13 | 29.26% | 17th |
| 26 | 0.23 | 20.08% | 14th |
| 27 | 0.13 | 16.46% | 10th |
| 28 | 0.12 | 7.38% | 1st |
| 29 | 0.30 | 21.45% | 15th |
| 30 | 0.09 | 31.03% | 20th |

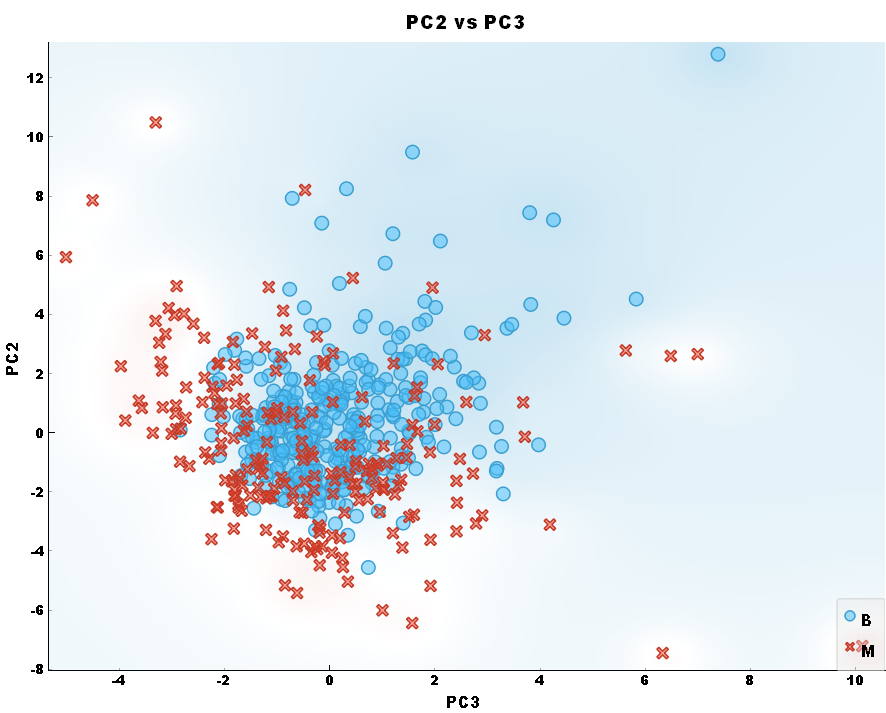


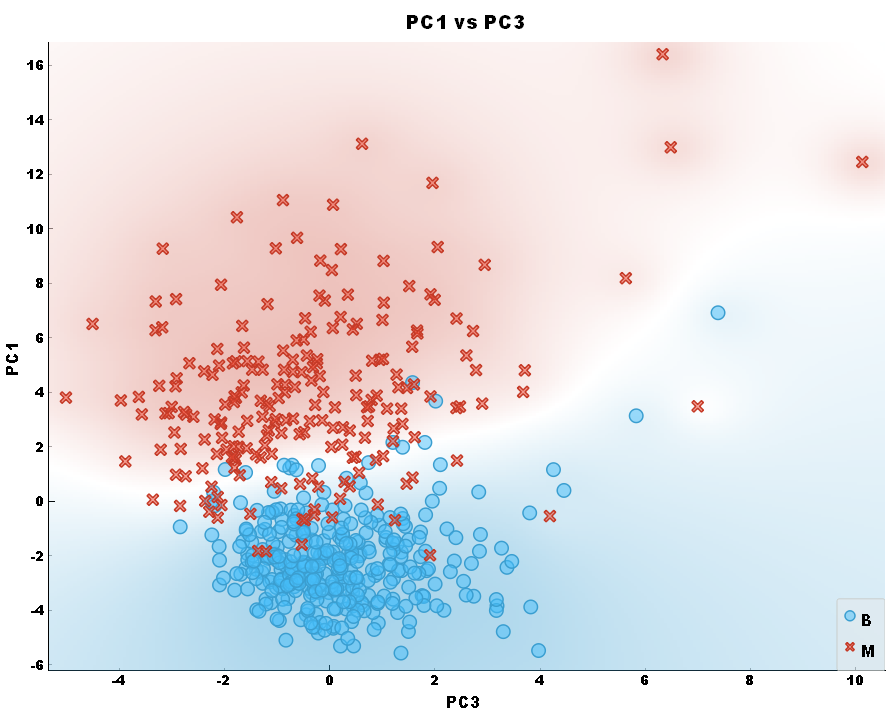




|  |  |  |
| --- | --- | --- |
| **Attribute** | **Threshold** | **Error** |
| PC1 | 0 | 9.31% |
| PC2 | 3 | 30.93% |
| PC3 | 2.23 | 30.22% |



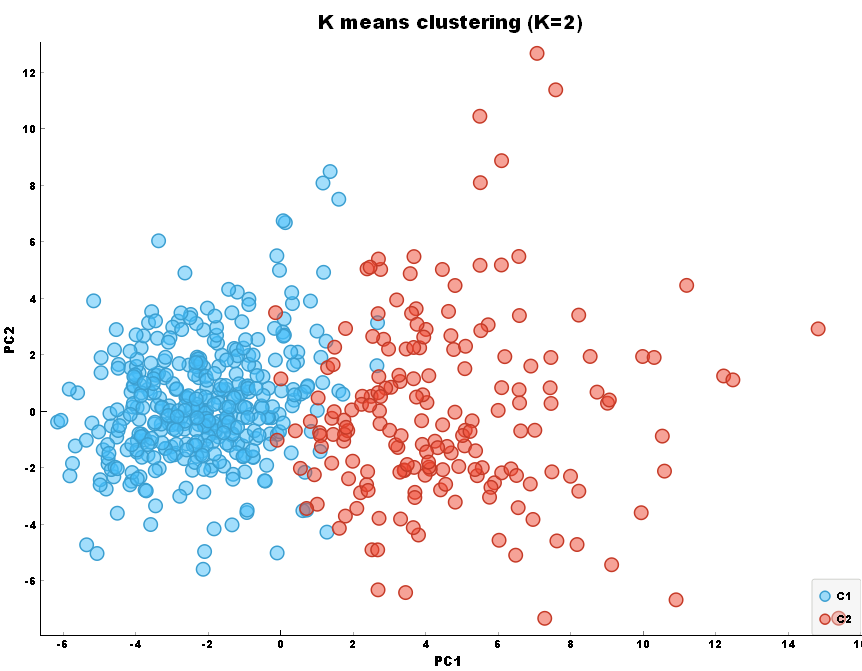


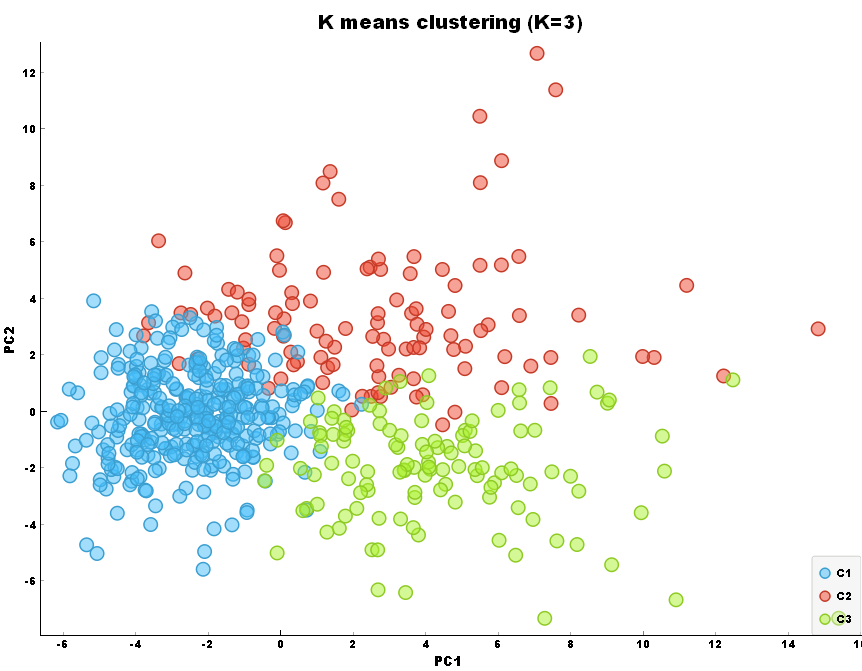


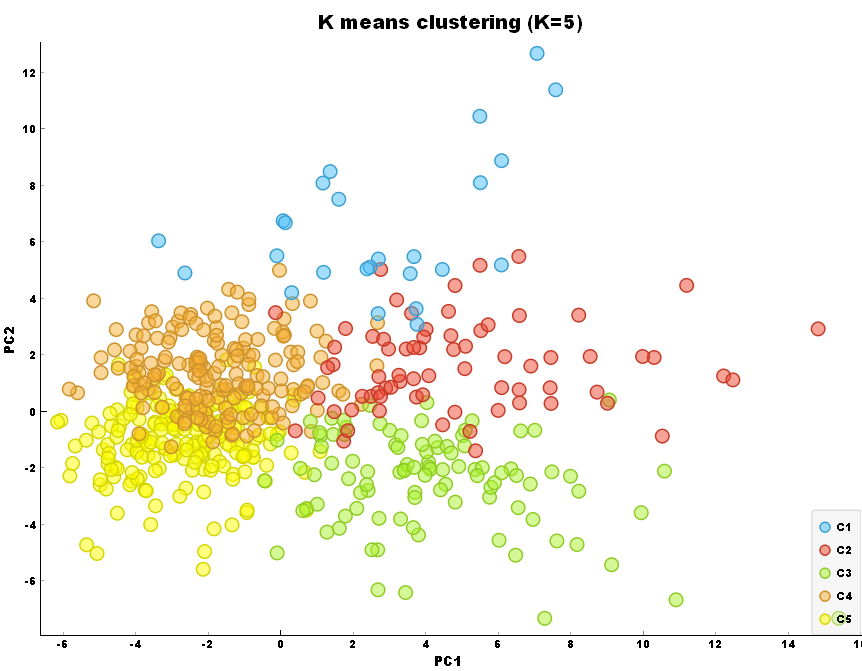
PC1 vs PC2 has good data separability. PC2 vs PC3 has poor data separability.

**Question 3**

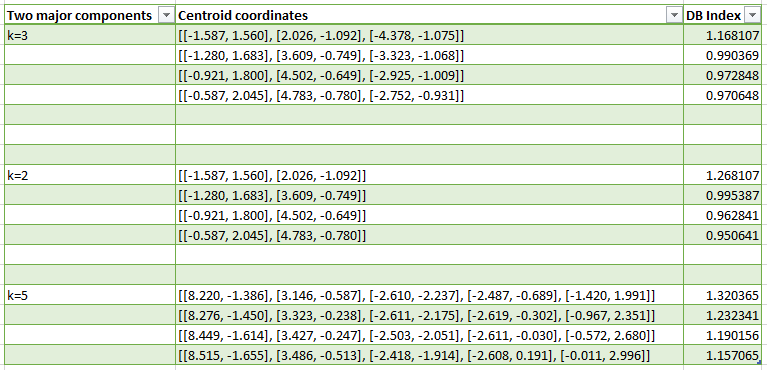
From sklearn.cluster library import KMeans method for implementing K-Means algorithm. PC1 and PC2 (Dimensionality reduced dataset) is used to implement the K-Means algorithm. Following are the results obtained by K-Means (K=2, K=3, K=5).





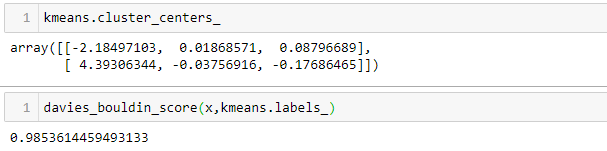


Initialize starting centroid coordinates and implement k-means algorithm and compute DB index for each iteration. Results are documented below. DB index is decreasing when the iteration goes on. This trend goes on.

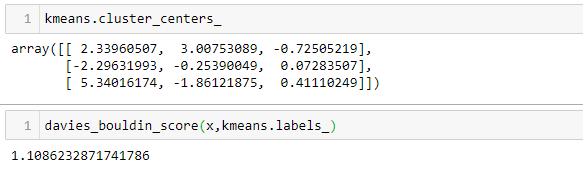


From sklearn.metrics library import davies\_bouldin\_score method for calculating K-Means algorithm.

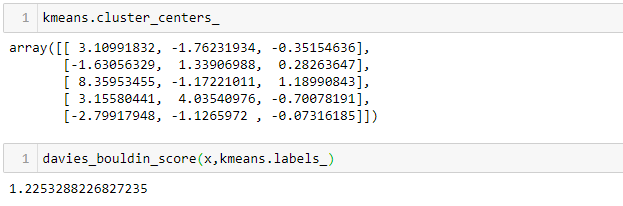
K = 2



K = 3



K = 4



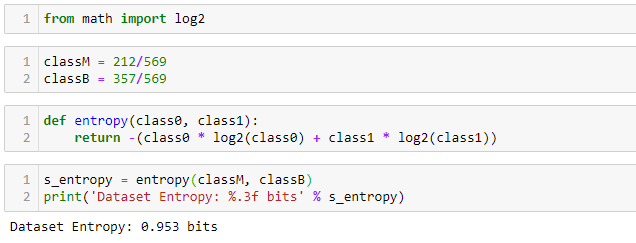
Above results (Cluster centroids and DB Index) is obtained when setting maximum iterations in Kmeans method to default, Hence this is the converged result.

**Question 4**

Accuracy/Error rate cannot be calculated for unsupervised algorithms like clustering. Purity is a measure of the extent to which clusters contain a single class. Accuracy is a completely different term than purity as Accuracy is metric used for supervised and Purity is metric used unsupervised algorithm.

Purity closer to one means that we can consider using the results obtained from K-means clustering as an evaluation metrics (Predicted) for a classification problem.

Following is the code snippet for calculating the overall dataset entropy.



**Software Used:**

Jupyter (Python)

Orange

PowerBI