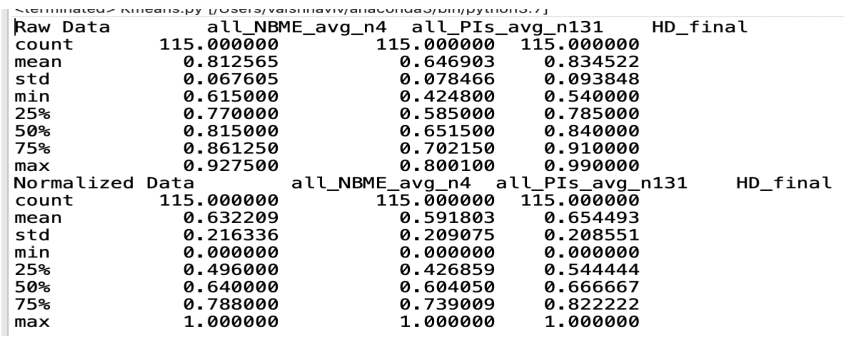
Machine learning - Assignment 1

[redacted] [redacted]

1. K-means clustering with different number of clusters (40 points):

a. Apply k-means clustering on the BSOM dataset with 3 features: 'all_NBME_avg_n4', 'all_PIs_avg_n131', 'HD_final', given the number of clusters k = 3. Visualize your clusters using a 3D scatter plot.

K-means attempts to group observations by spatial proximity. If you were to specify 2 clusters (k=2), for example, you might find that there were two groups of clusters that were (hopefully) spaced far apart. We are using Euclidean distance, calculating those distances for variant data would be meaningless if feature scales are not uniform. But this is not the case with the BSOM data however I have normalized the data before proceeding with the experiments. Raw and Normalized data results displayed in Figure 1.



```
<terminated> Kmeans.py [/Users/vaishnaviv/anaconda3/bin/python3.7]
Raw Data         all_NBME_avg_n4  all_PIs_avg_n131    HD_final
count       115.000000         115.000000  115.000000
mean          0.812565           0.646903    0.834522
std           0.067605           0.078466    0.093848
min           0.615000           0.424800    0.540000
25%           0.770000           0.585000    0.785000
50%           0.815000           0.651500    0.840000
75%           0.861250           0.702150    0.910000
max           0.927500           0.800100    0.990000
Normalized Data         all_NBME_avg_n4  all_PIs_avg_n131    HD_final
count       115.000000         115.000000  115.000000
mean          0.632209           0.591803    0.654493
std           0.216336           0.209075    0.208551
min           0.000000           0.000000    0.000000
25%           0.496000           0.426859    0.544444
50%           0.640000           0.604050    0.666667
75%           0.788000           0.739009    0.822222
max           1.000000           1.000000    1.000000
```

Figure 1:Raw data – Normalized data

***3D visualization of clusters**: Figure 2 shows the 3D scatter plot for k =3 where each color represents each single cluster with their centroids marked in bold black. For our understanding of analysis further in this report we label orange plots as cluster 1, Green as cluster 2 and Red as cluster 3.*

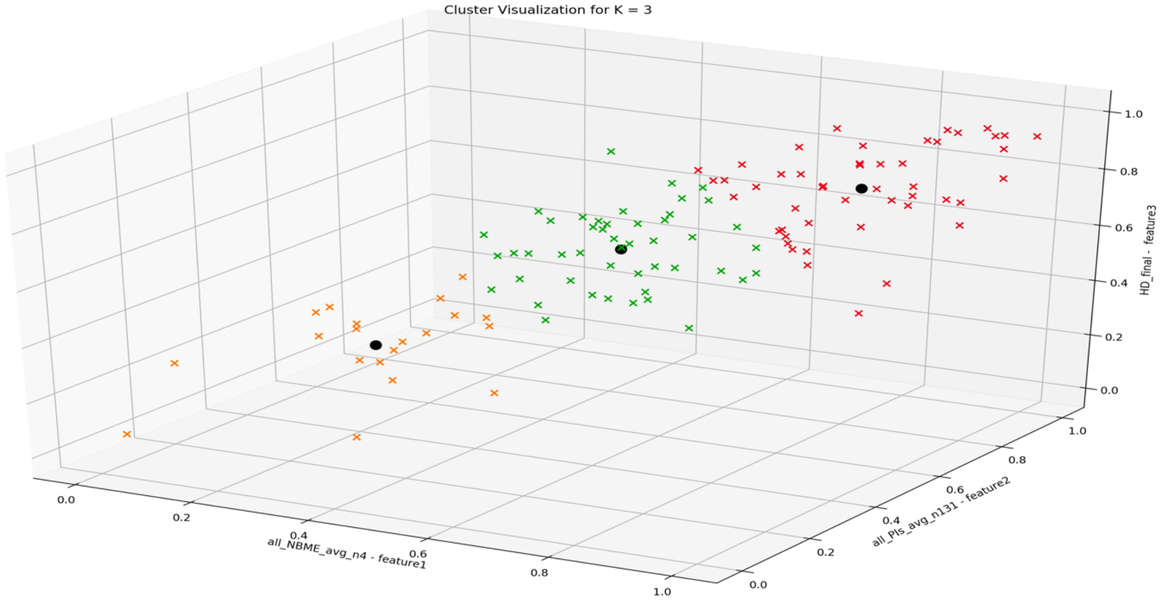


Figure 2: 3D scatter plot visualization for k =3

b. Test with different number of clusters k, from k = 2 to k = 10. Which one you believe is the best number of clusters? Justify your response. (Hint: you may compare the 3D scatter plots with different number of clusters.)

<u>Analysis of best number of clusters from below 3D scatter plots:</u>

Conclusion: we can say k=2 is the best number of clusters through visualization since Intercluster distance is higher than other number of clusters from k=3 to 10 with overlap of clusters as we increase the cluster count.

Analysis resulted in above conclusion:

K=2: Figure 3 gives visualization of two clusters. For k=2 we can see clear distinction between two clusters each cluster is dense enough. Many points are closely around centroids except few.

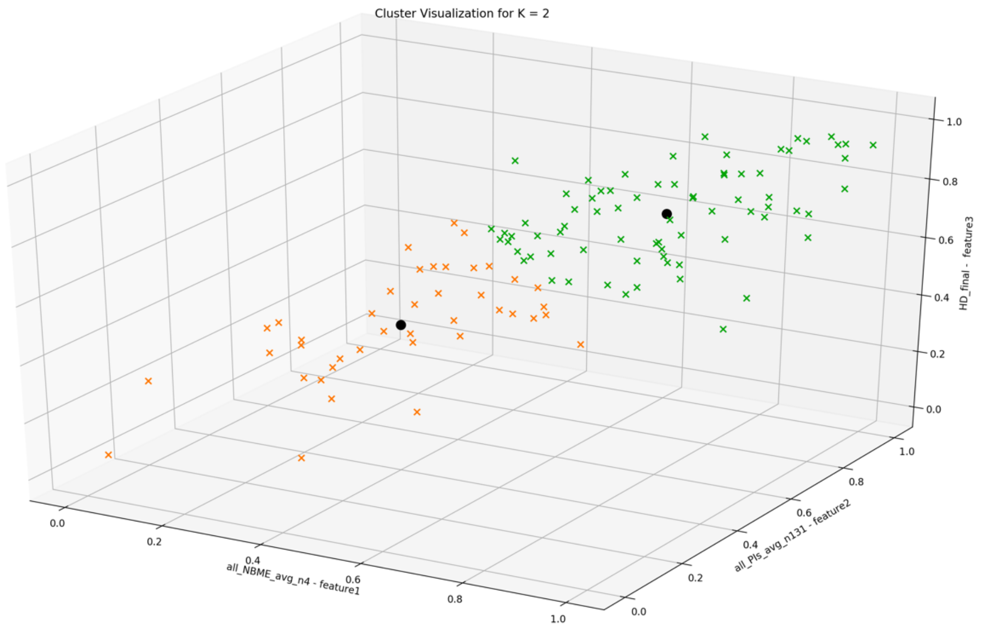


Figure 3: 3D scatter plot visualization for k =2

K=3: Figure 4 gives visualization for 3 clusters. Among three, two clusters are dense and one of the clusters is not dense, since K-means assume all data in circular shape, cluster in orange can overlap with cluster in green.

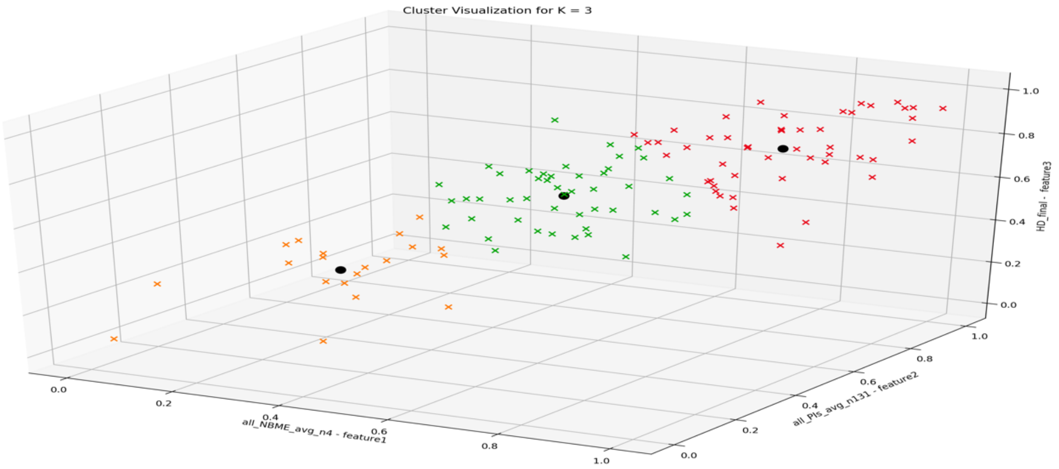


Figure 4 :3D scatter plot visualization for k =3

From Figure 5 to 11 for K=4 to 10 respectively we can observe most clusters overlaps and the points around the centroids are becoming sparser as we increase number of cluster initialization. Cluster in orange have only three points in its clusters which appears like an outlier.

3D visualization for K=4

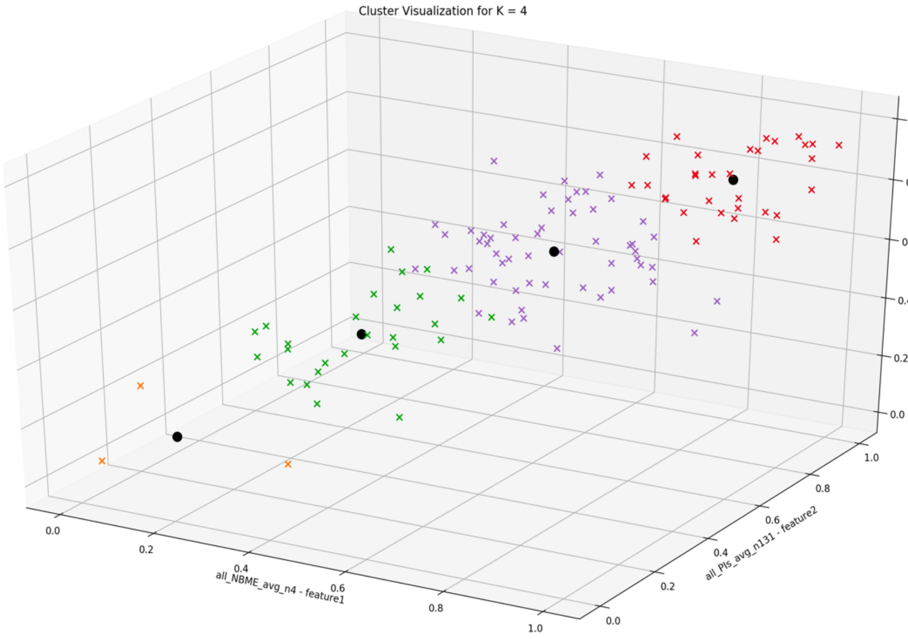


Figure 5: 3D scatter plot visualization for k = 4

3D visualization for K=5

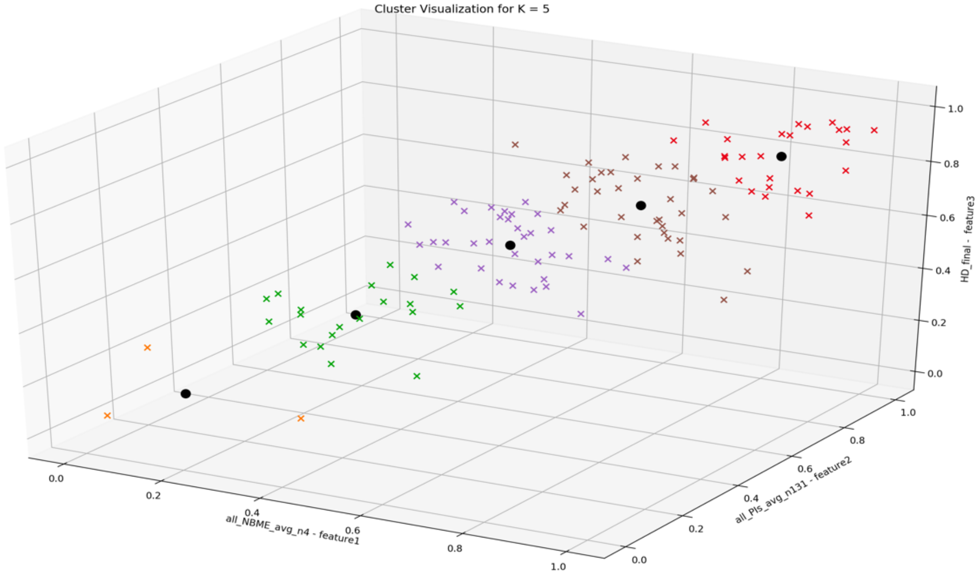


Figure 6: 3D scatter plot visualization for k =5

3D visualization for K=6

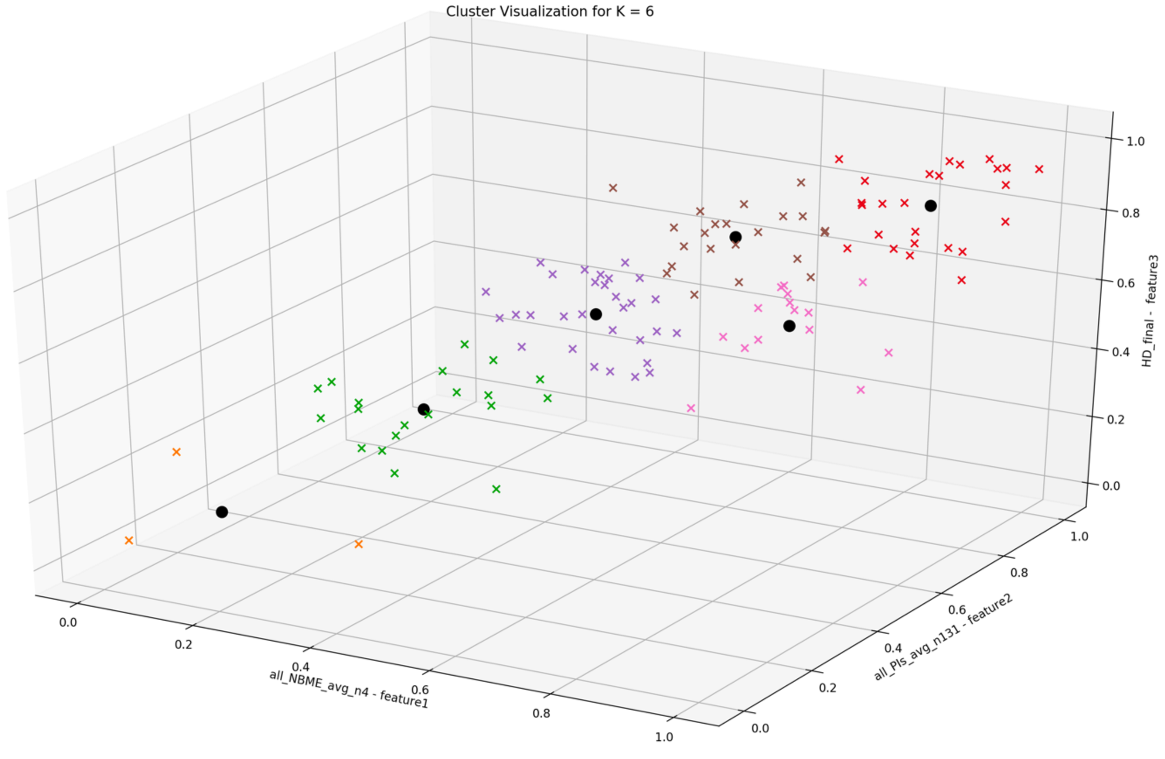


Figure 7: 3D scatter plot visualization for k =6

3D visualization for K=7

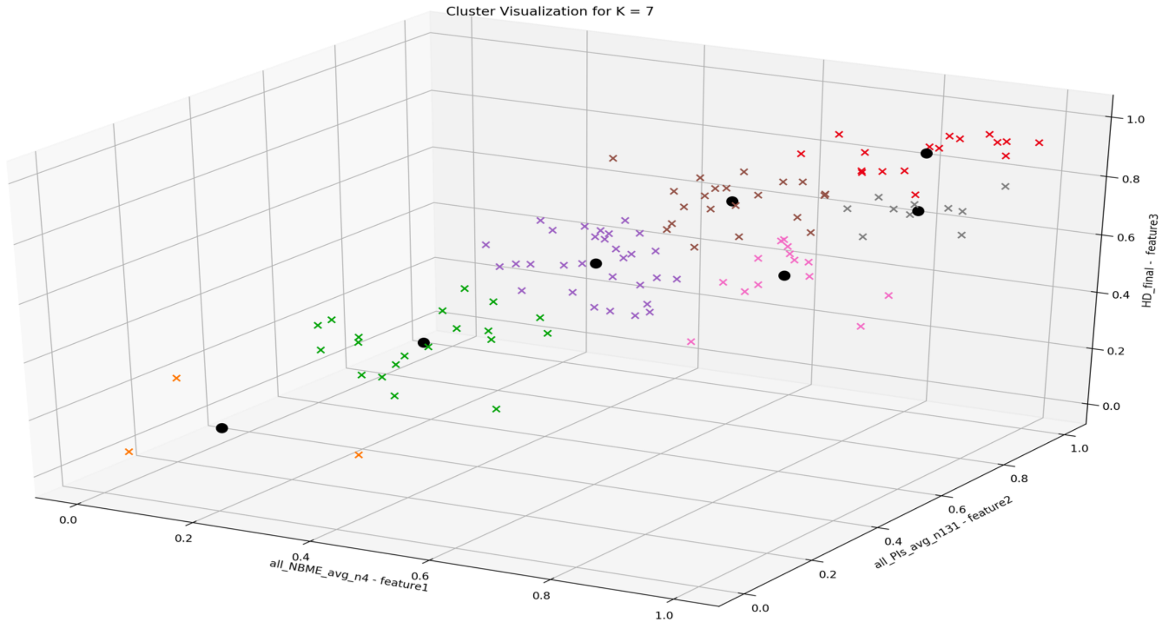


Figure 8: 3D scatter plot visualization for k =7

3D visualization for K=8

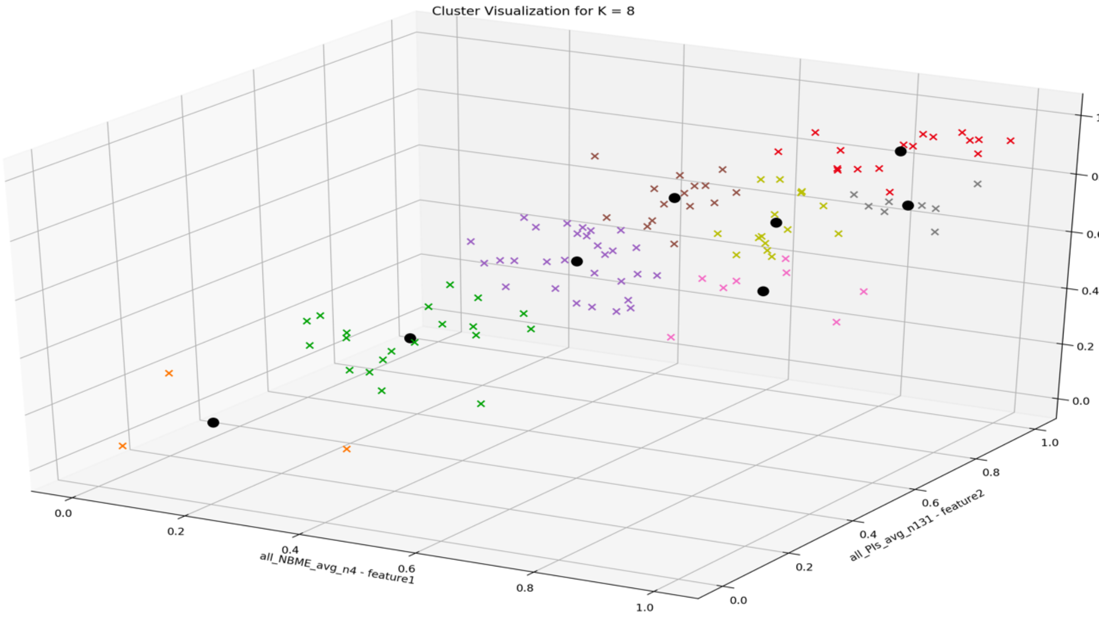


Figure 9: 3D scatter plot visualization for k =8

3D visualization for K=9

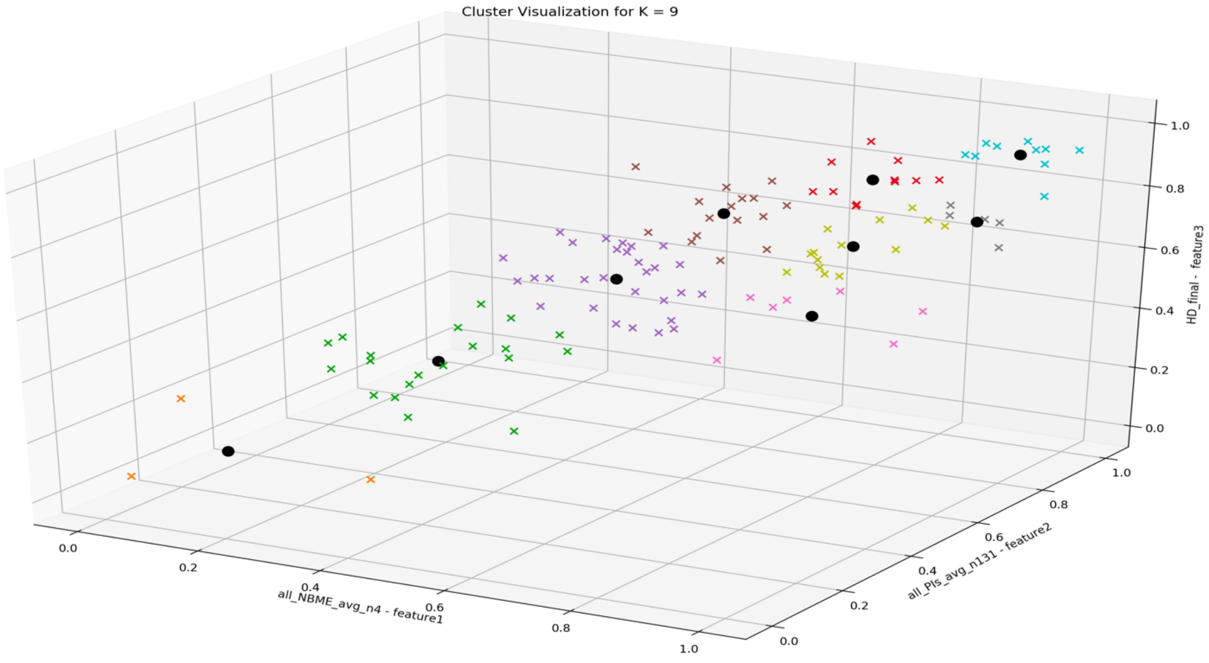


Figure 10: 3D scatter plot visualization for k =9

3D visualization for K=10

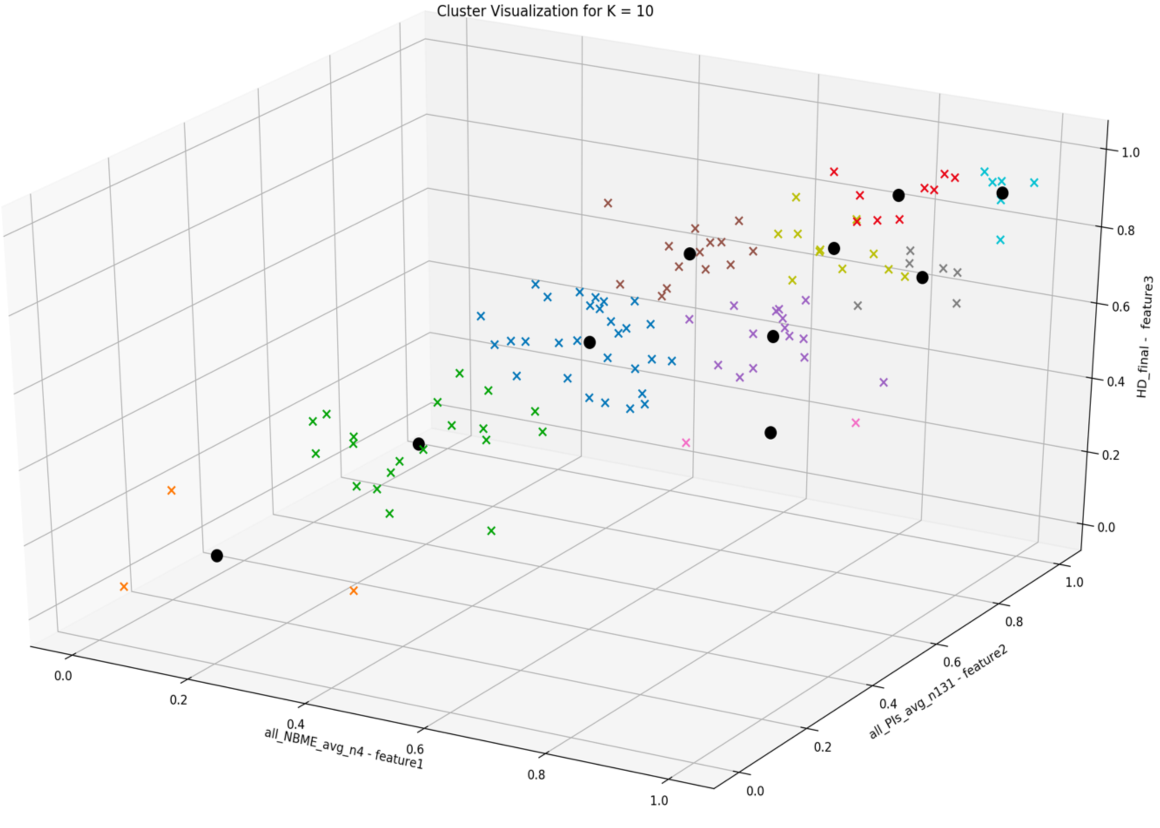


Figure 11: 3D scatter plot visualization for k =10

c. Implement Davies-Bouldin (DB) validity measure. Repeat experiments in problem 1b and calculate corresponding DB indices. Which one you believe is the best number of clusters using the validity measure? Does it agree with your initial observation in problem 1b?

Analysis: Davies-Bouldin validity measure is an internal cluster validation method where results are evaluated based of clustered data without external information as reference. For this validity measure **lower the DB index value better is the clustering** which means **a best cluster minimizes the DB index**.

Figure 12 show the plot of the DB index against cluster values from k= 2 to 10. It can be observed that for k=2 the DB index value is 0.7990049818342608 which is lowest ,so it can be interfered that by using DBI as validity measure, **k=2 gives good quality cluster which is acceptable from the observation in problem 1b where k=2 provided good cluster visualization.** However for k=3 and k=4 the change is minor so we can even initialize number of cluster with given k values but can result in overlapping clusters.

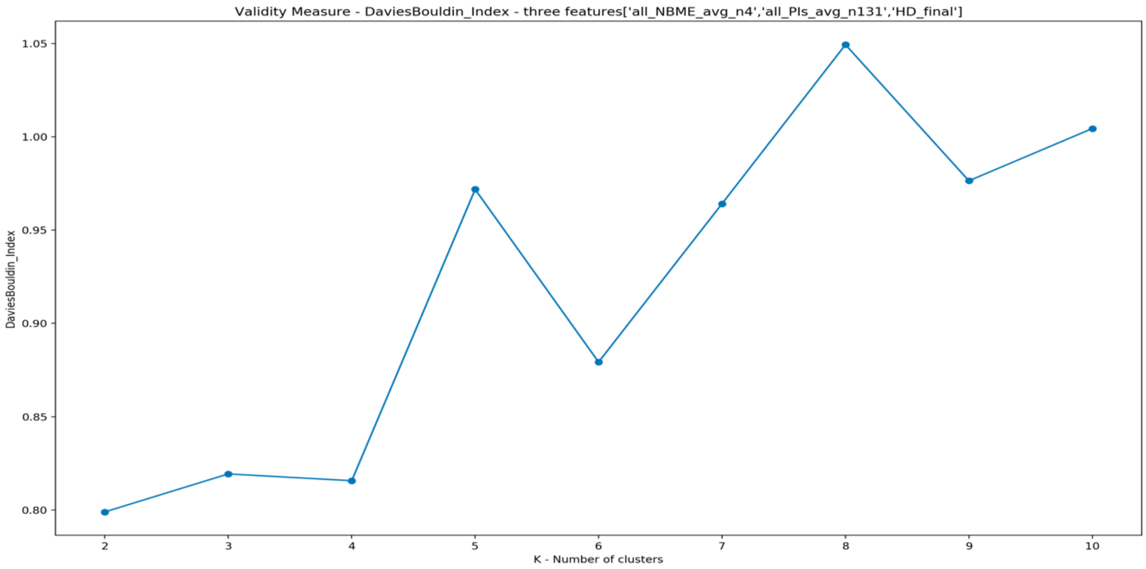


Figure 12: 2D plot of number of clusters against DB index for three features ----Kmeans algorithm

2. K-means clustering with different features (20 points):
a. Based on the best number of clusters you obtained in problem 1c and the 3 features, does adding the 'all_irats_avg_n34' (total 4 features) improve the clustering results? Using validity measures to justify your response.

Analysis: Figure 13 show the plot of the DB index against cluster values from k= 2 to 10 with newly added fourth feature. **Adding one more feature "all_irats_avg_n34" didn't improve the clustering results**. For k =2 the DBI index obtained for three features is "0.7990049818342608" which is less than "0. 8262209991214161" for four features with k=2.It can be interpreted that the information from fourth feature added might not correlated with the cluster assignment and as data dimensions increase, the Euclidean distances suffer from the curse of dimensionality having impact on the result.

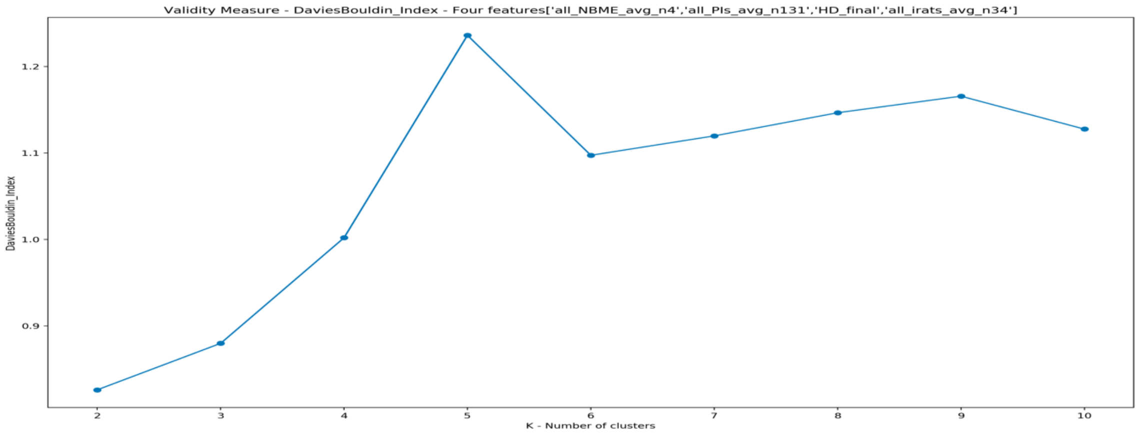


Figure 13: 2D plot of number of clusters against DB index for four features ----Kmeans algorithm

b. Based on model in problem 2a, does adding the 'HA_final' (total 5 features) improve the clustering results? Using validity measures to justify your response.

Analysis: No, adding the 5th feature "HA_final" did not improve the clustering result. Figure 14 explains about DB index validity measure plots for feature count 3 ,4 and 5. It can be inferred that adding 5th feature **did not yield good DBI value** 0.8704137935346903 when compared with 0.8262209991214161, 0.7990049818342608 obtained for feature count four and three respectively. As number of features increases the amount of information is higher which means higher flexibility and freedom.

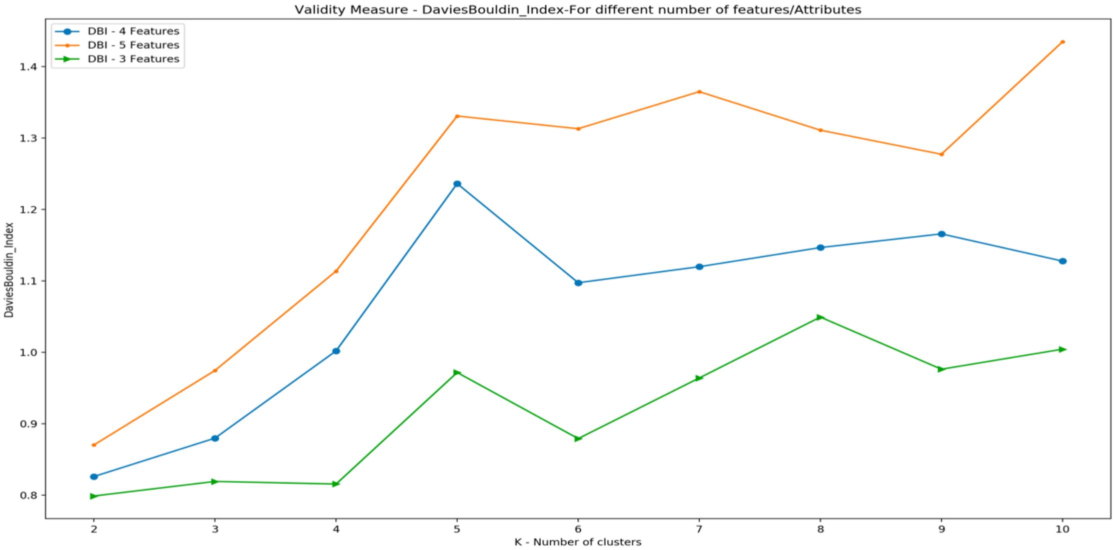


Figure 14: 2D plot of number of clusters against DB index for three, four and five features ----Kmeans algorithm

3. Fuzzy C-means clustering (40 points): a. Implement Fuzzy C-means and apply it with the best number of clusters you selected in problem 1 and the best combination of features you selected in problem 2. Was there any difference in the clusters as compared to the k-means clusters? (Compare using visualization tools, using centroid values, OR using some labels and observing the differences).

Best conditions from problem 1 and 2: Best **number of clusters is k =2** and best combination of features using DB index as validity measure is three namely **'all_NBME_avg_n4','all_PIs_avg_n131','HD_final'.**

Figure 15 and Figure 16 shows 3D plots for "fuzzy C-means" and "K -means" algorithm. Based on the experiment, the centroids obtained for FCM and Kmeans is given below.
Kmeans Centroids k = 2{1:[0.41580488, 0.37833798, 0.45528455]),2:[0.75210811, 0.71007338, 0.76486486])}
FCM Centroids for k = 2{1:[0.4332309, 0.39960813, 0.47909936]), 2: [0.77607345, 0.73921161, 0.7799264])}

Analysis: There has been **difference in the clusters as compared to k-means**. In Fig. 15 and Fig. 16 cluster marked in orange have more points for Fuzzy C-means than K-means, this can be observed at axis 0.2 perpendicular to feature 3. It means few points from green cluster in k-means shifted to orange cluster in fuzzy-c. This is because Fuzzy C-means labels are based on maximum of membership value calculated using membership function. While K-means uses closet Euclidean distances from centroids rather square of the Euclidean distance and weights w.r.t to each data points. Due to classification of points to orange cluster from green cluster using fuzzy C-means there is change in cluster size (for around 8 points) for fuzzy resulted in change of centroid points for fuzzy (centroids shifted by 0.02 positively) w.r.t k-means. The same can be concluded from centroid points mentioned above.

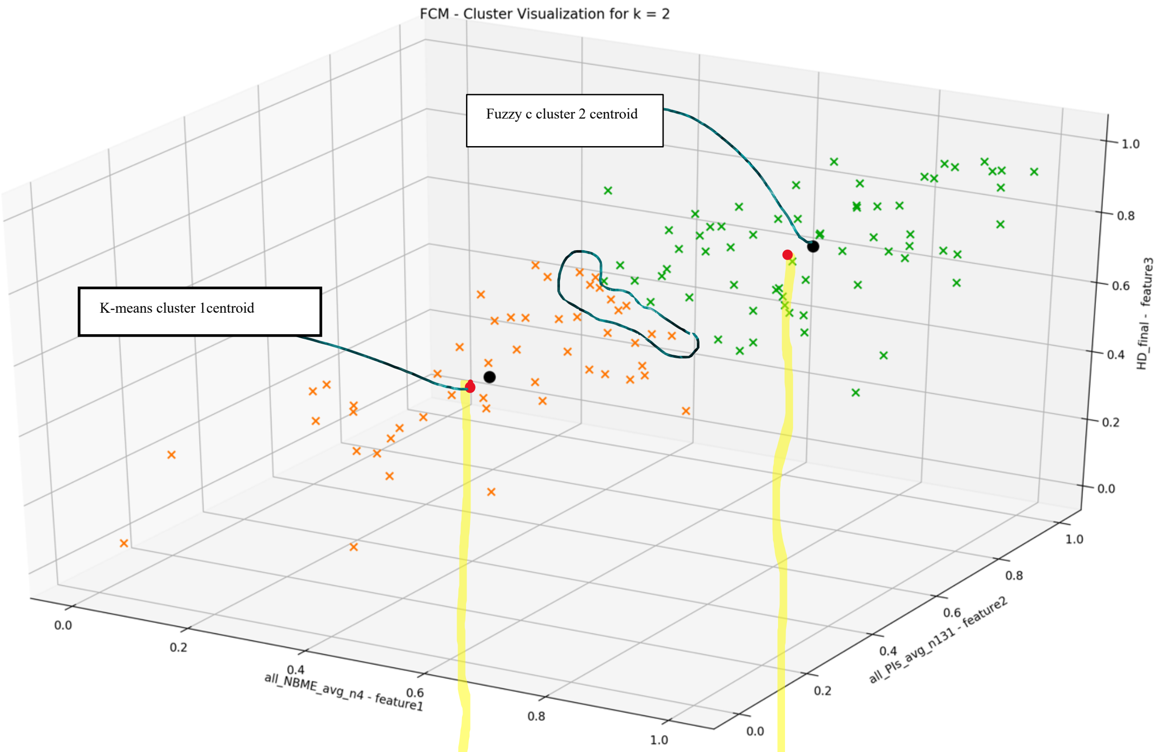


Figure 15:3D plot for Fuzzy C means for k =2 with projection of centroids of k-means algorithm

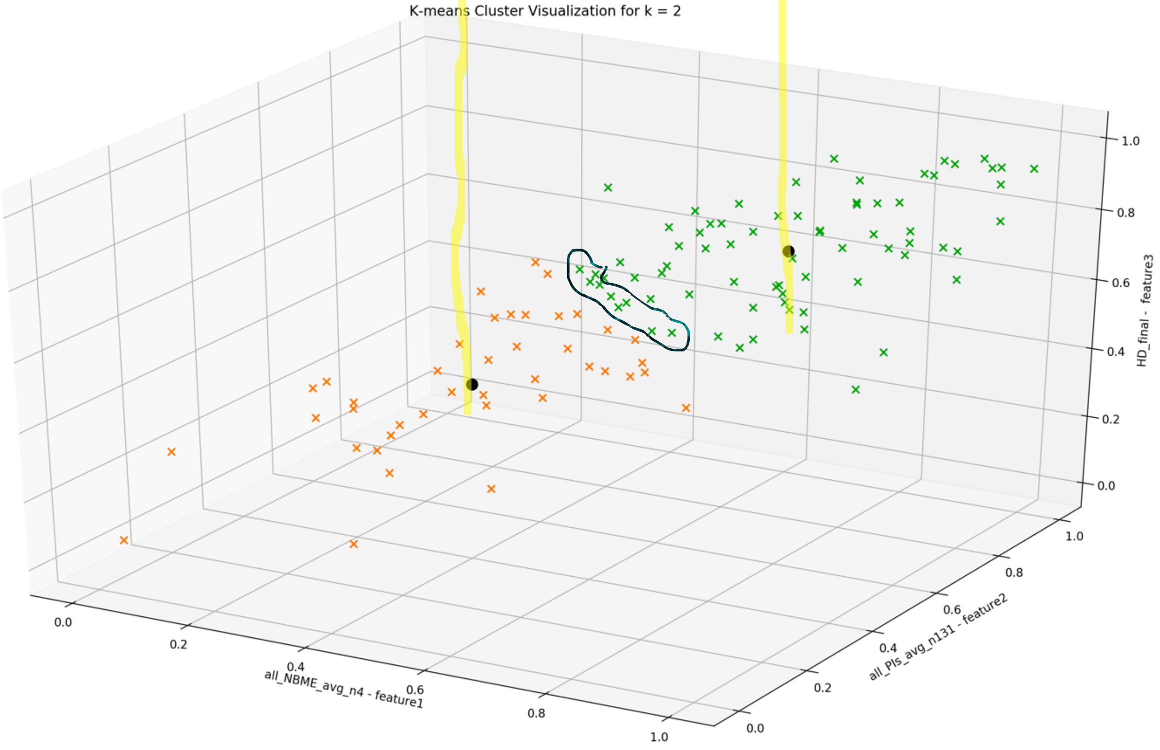


Figure 16: 3D plot for K-means for k =2 with projection of k mean centroids to Figure 15

b. Harden the cluster assignment of Fuzzy C-means and use DB index to compare it with the k- means clustering result. Which clustering algorithm you think produce better clusters and why?

Analysis: **Fuzzy C-means produced better results for selected k=2 and number of features = 3**. The DB index value 0.7895064748211907 is less than k-means (0.7990049818342608). Fuzzy C-means algorithm membership calculation involves weighing the data points along with fuzzifier in order to know the degree of truth of the datapoint belonging to a particular cluster. So Fuzzy C-Means produce better results than K-Means as it does not restrict the datapoint belonging to one single cluster. Furthermore, for given data set by hard clustering the Fuzzy results, its performance is degraded which can be observed from Figure 17. We could interpret it as follow. During Hard clustering we take maximum of membership values, sometimes those membership values might not be above 50% threshold which might result in assigning them to wrong cluster when compared with k-means. The DB index for FCM is better than K-means at k=2 only not for k = 3 to 10.

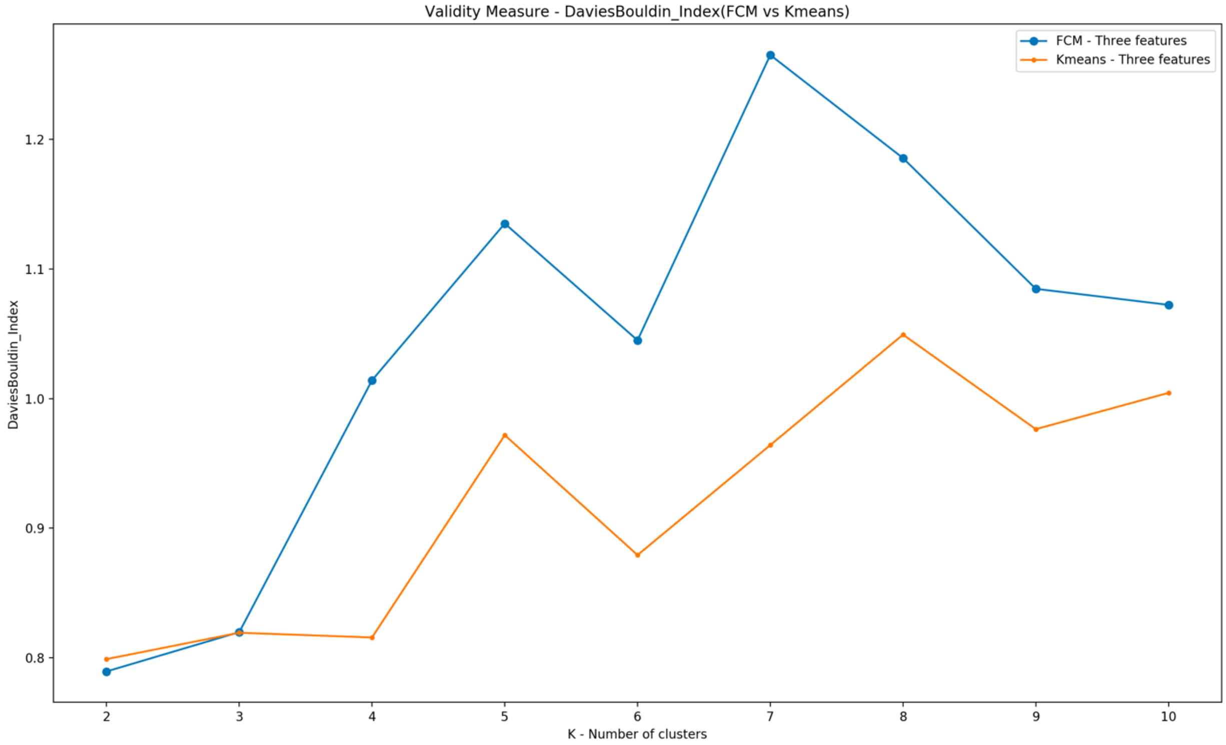


Figure 17:2D plot of number of clusters and DB index for FCM vs K-means when number of features =3

c. Add one more feature into the to the model in problem 3a. Does adding this new feature improve the clustering results? If so, why or why not?

Analysis: **Result is not improved**. By experimenting with another additional feature “all_irats_avg_n34 “into problem 3a and here we are validating the cluster results using DB index validity measure. We can observe from Figure 18 that validity measure (DB index = 0.7895064748211907) of FCM is yielding better cluster for three features than for four features (DB index= 0.8225975213339006).This is because when dimensions increases the points move or become sparse from centroid .So distance from centroids increase resulting in increase in intracluster distance which pave path for increase of DB index .**Since Lower DB index contribute to better cluster results we can conclude there is not improvement in clustering result by adding additional feature**.

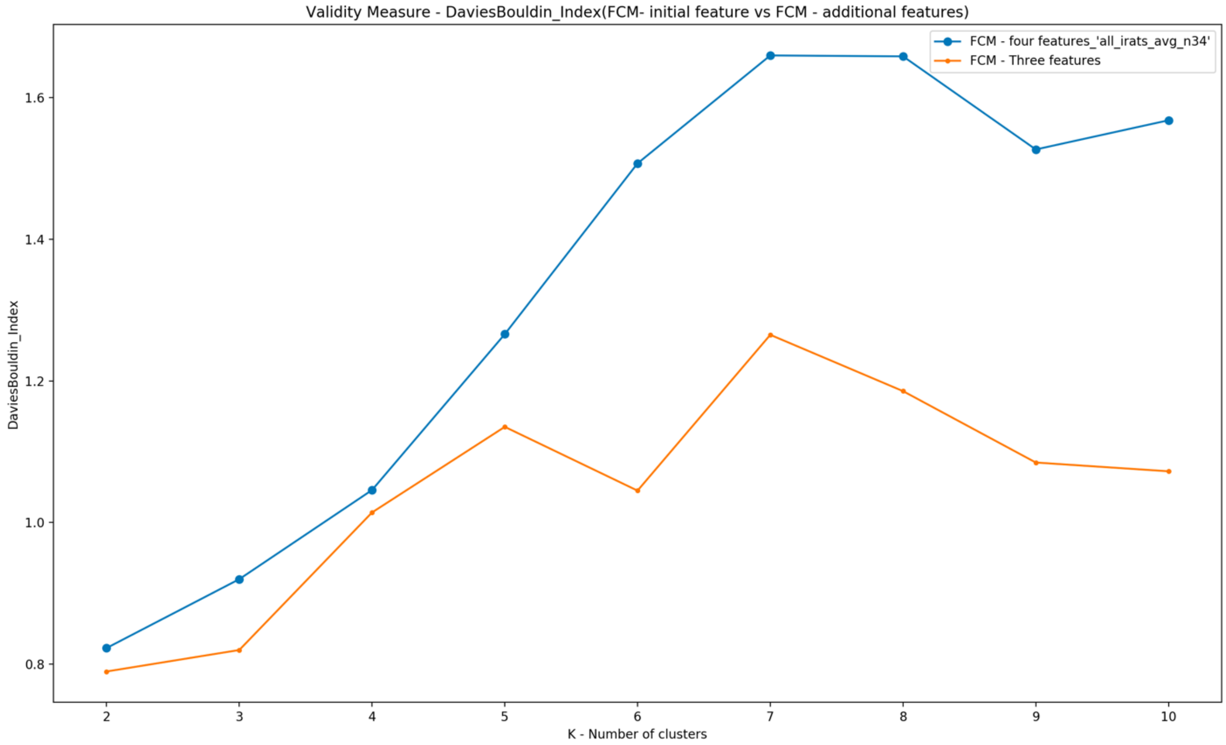


Figure 18:2D plot of DB index validity measure for different number of clusters using Fuzzy C-Means algorithm or 3 and 4 features.