


PRML Minor Project

YouTube Comment Classification

Arvind Kumar Sharma (B21AI006)

Nitish Bhardwaj (B21AI056)

Renu Sankhla (B21AI028)



Problem Statement : This dataset contains videos from 4 different YouTubers and all the comments made on those videos. The primary objective of this dataset is to cluster the comments to identify a cluster that contains all the spam comments and fix the issue once and for all.

Pipeline:

1. Data Preprocessing
2. Data Cleaning
3. Feature Engineering
4. Model Selection
5. Prediction



1. Data Preprocessing

Firstly, we have seen the attributes in the dataset. From all the attributes, only 'comment (displayed)' is used for further processing. We searched for nan values and dropped them. Since the dataset was quite large, we worked on a small dataset of 10,000 samples.

2. Data Cleaning

The nltk (Natural language Toolkit) library was initially utilized for stopping and lemmatization. Stopping means removing the unnecessary words which don't carry much meaning like 'the', 'and', 'a', 'an', 'of', etc.. As a result, our feature vector is put to better use. Lemmatizing means reducing a word to its essential form; for example, run will replace running and ran. Lemmatizing is preferred over stemming because the latter can lead to meaning loss; for example, "caring" would become "car" if stemmed.

Not considering this is a wise move.

3. Feature Engineering

To convert the comment into the feature, we are using countVectorizer from sklearn. It converts the word in the number of frequencies and each sample in the feature vector represents the number of times that word has come into that comment. Since the features were quite large, we also worked on a reduced dimension dataset. — melspectro ka dal

4. Model Selection

There are three models which we have selected for this problem. Those three models are:

- Agglomerative Clustering
- K means
- Spectral Clustering

1. Agglomerative Clustering

Agglomerative Clustering is a clustering technique in which initially all the data points are considered clusters, and we merge those two clusters that satisfy some metric. For this clustering algorithm, we have taken five different metrics to find the best suitable distance. We repeat this process until we

These distances are:

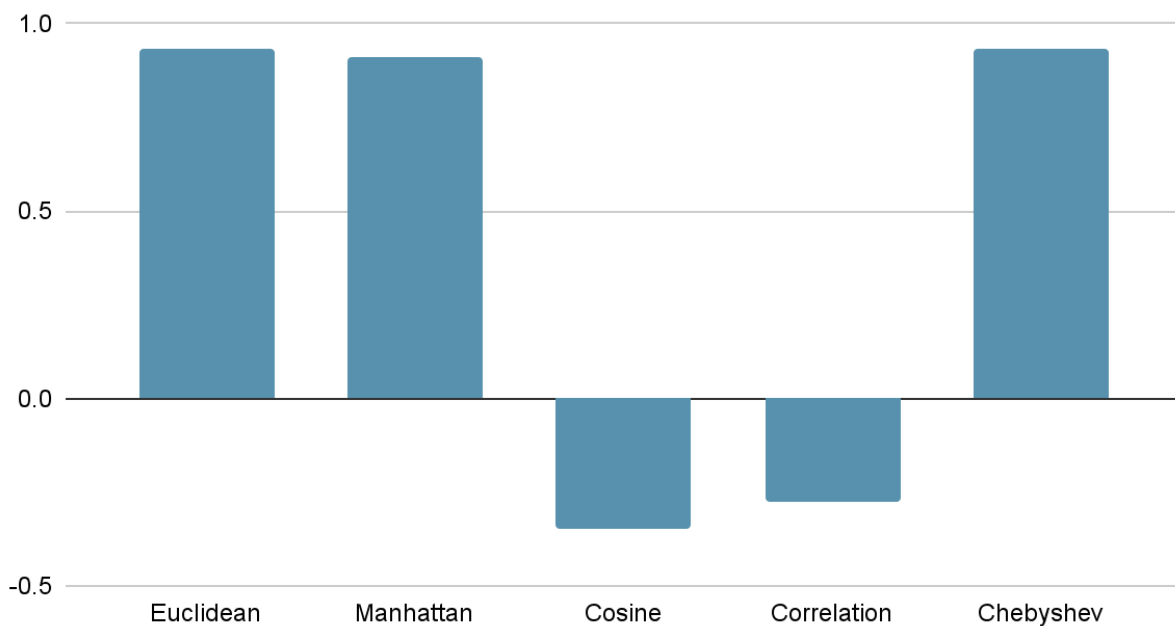
- Euclidean
- Manhattan
- Cosine
- Correlation
- Chebyshev

As well as three evaluation methods are used to quantify the quality of the clusters. They are:

- Silhouette Score
- Calinski Harabasz Score
- Davies Bouldin Score

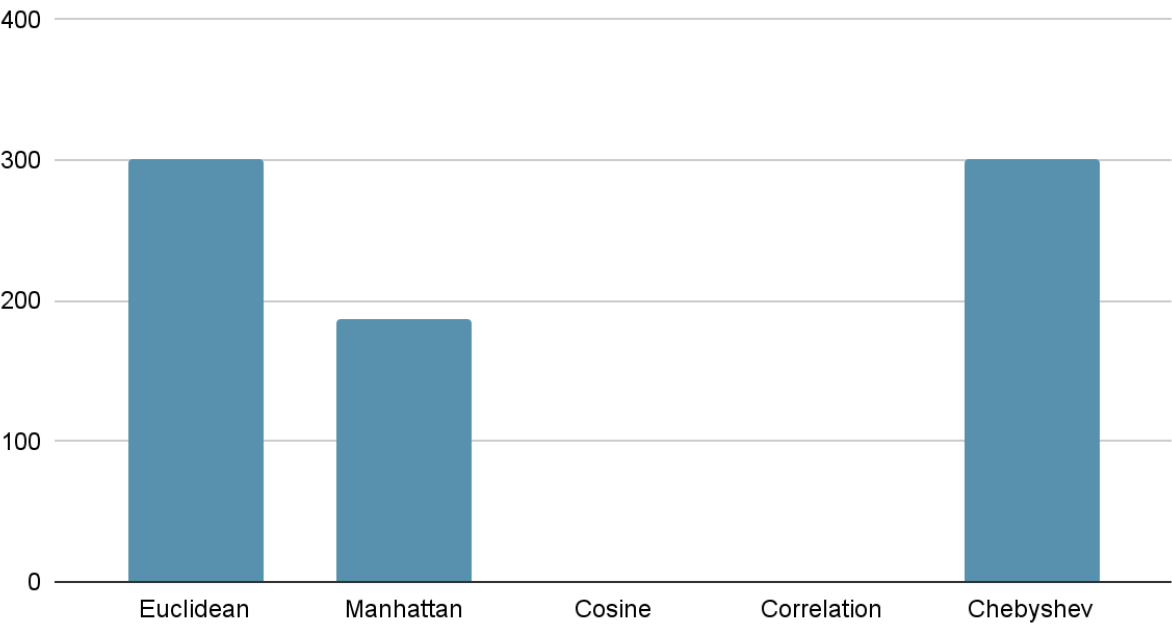
The result are as follows:

Silhouette Score

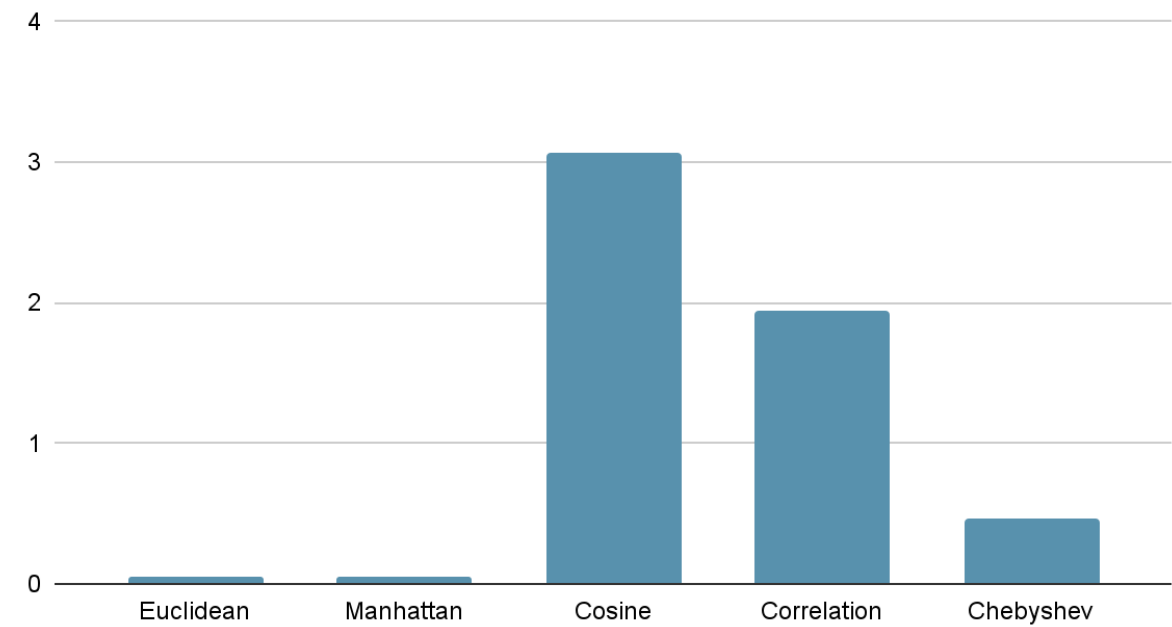




Calinski Harabasz Score



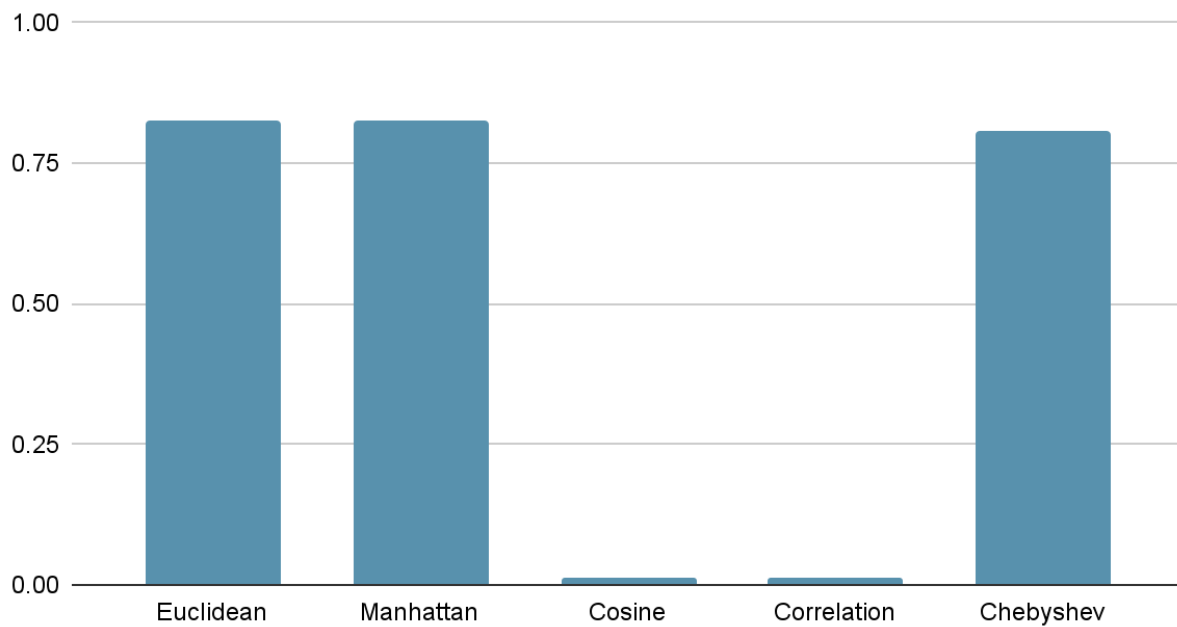
Davies Bouldin Score



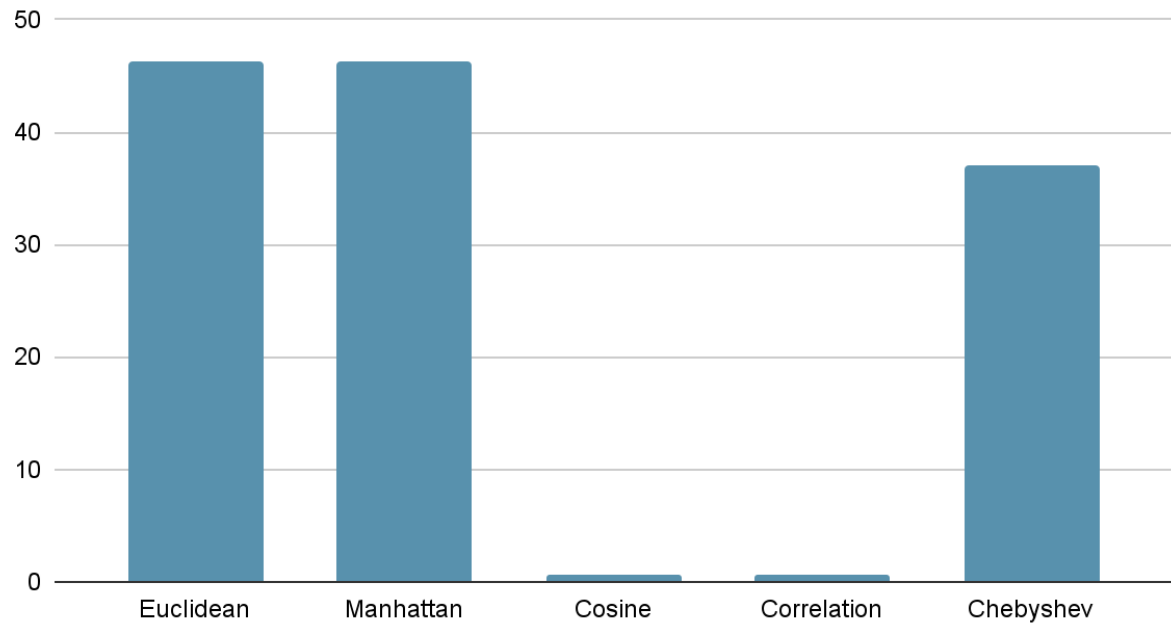
While Silhouette Score, Calinski Harabasz Score measure the difference between the clusters, Davies Boulder Score measures the similarity between the clusters.

PCA is also applied to see if we could decrease the number of features and still conserve the accuracy. With the help of PCA, the number of features is reduced to 2000. The score are as follows:

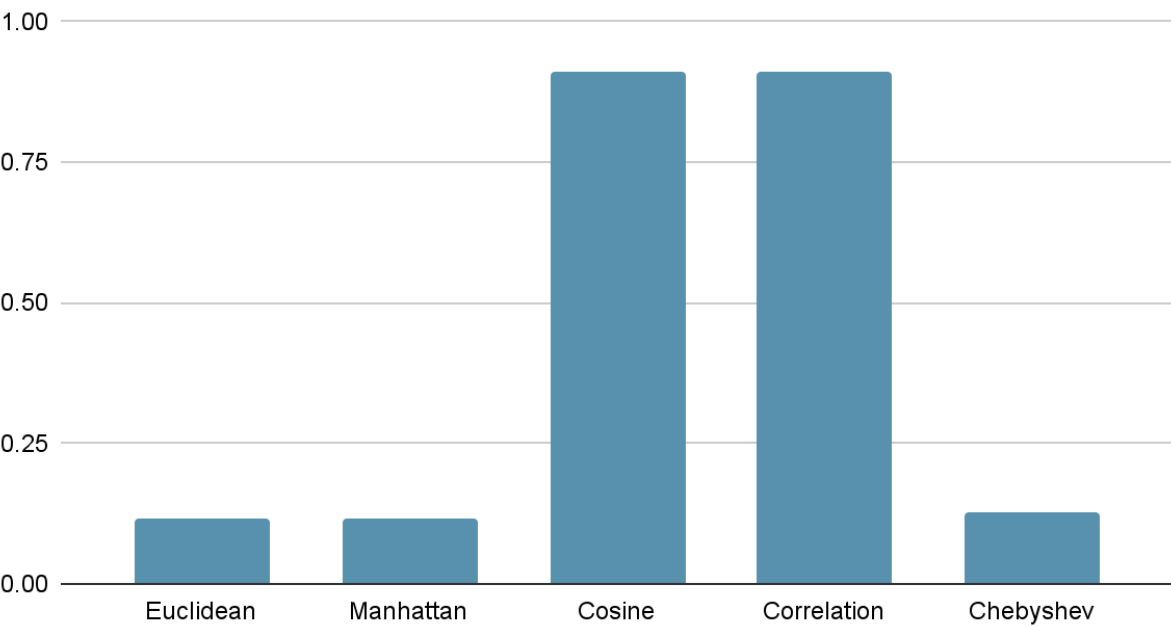
Silhouette Score



Calinski Harabasz Score



Davies Bouldin Score



2. K Means

K-means is a well-known algorithm for unsupervised machine learning that groups data points into groups based on how similar they are. The algorithm aims to partition n observations into k clusters, where each observation belongs to the cluster with the nearest mean.

Working of K-means clustering :

1. Choose the number of clusters k . Here we chose $k=2$, because we have to make clustering for two classes, spam and non-spam.
2. Randomly initialize k centroids.
3. Assign each observation to the cluster with the nearest centroid.
4. Update the centroids of each cluster as the mean of all the observations assigned to it.
5. Repeat steps 3 and 4 until the centroids no longer move or a maximum number of iterations is reached.

The algorithm attempts to minimize the sum of squared distances between each data point and its assigned centroid.

We varied the 'thresh' hyper-parameter provided in the scipy k means and calculated the scores.

'Thresh' terminates the k-means algorithm if the change in distortion since the last k-means iteration is less than or equal to threshold.

Scores calculated are as shown in table:

	Threshold	Cluster sizes	calinski_harabasz_score	silhouette_score	davies_bouldin_score
0	exp(-0)	[9123, 762]	557.544690	0.477448	2.776159
1	exp(-1)	[9135, 750]	559.702499	0.478050	2.752241
2	exp(-2)	[9170, 715]	564.061598	0.488529	2.728606
3	exp(-3)	[703, 9182]	566.104280	0.489635	2.706399
4	exp(-4)	[670, 9215]	568.993821	0.501548	2.694415
5	exp(-5)	[9281, 604]	579.205572	0.521519	2.632048
6	exp(-6)	[9378, 507]	597.220440	0.551378	2.513900
7	exp(-7)	[475, 9410]	603.934291	0.558776	2.453375
8	exp(-8)	[370, 9515]	621.980237	0.588888	2.261132
9	exp(-9)	[485, 9400]	601.833029	0.557252	2.476814

3. Spectral Clustering :

Spectral clustering is a technique in machine learning for clustering data points into groups based on their similarity. It involves transforming the clustering into a high dimensional space using a matrix operation and then partitioning the transformed data points into clusters.

In this part we perform clustering analysis on a dataset named X_disp4_056 using the Spectral Clustering algorithm with two different types of affinity measures: RBF and nearest neighbors. The number of clusters is set to two in both cases. The purpose of this analysis is to determine which affinity measure produces better clustering results.

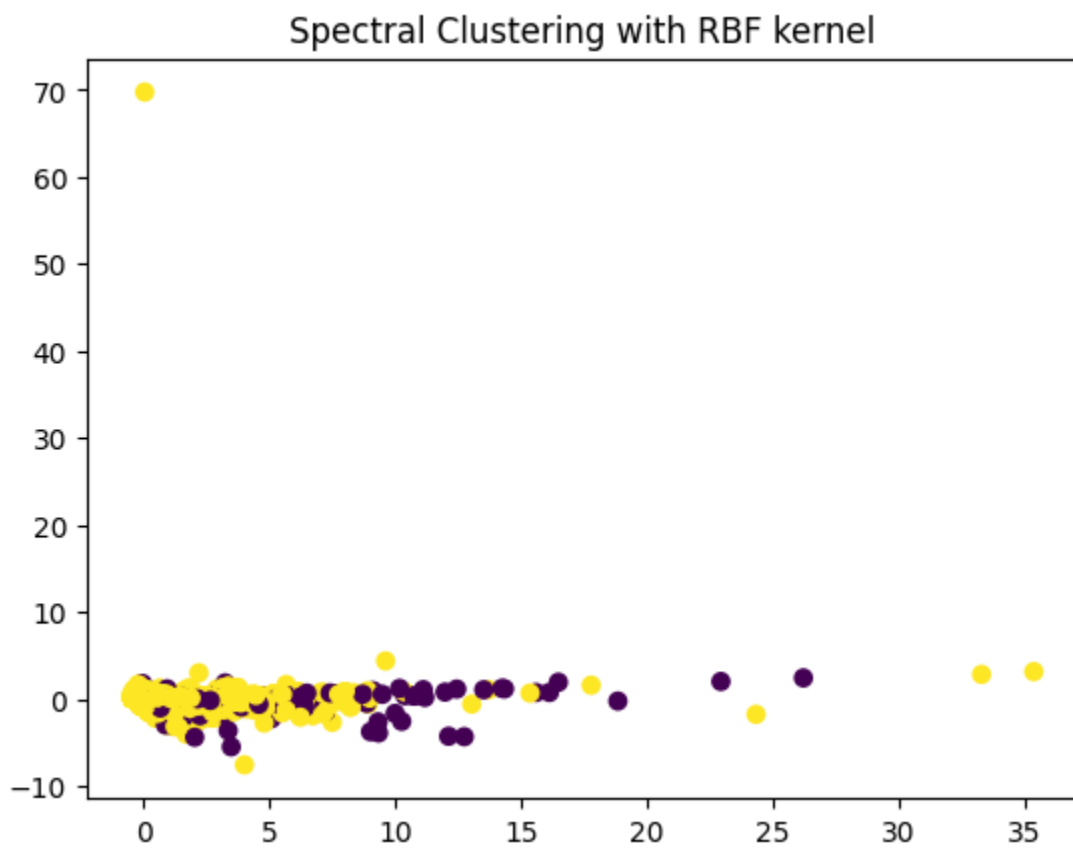
The clustering labels obtained from the two algorithms are stored in Y_111 and Y_122, respectively. The number of data points assigned to each cluster for each algorithm is shown in the arrays returned by `int(np.unique(Y_111,return_counts=True))` and `np.unique(Y_122,return_counts=True)`.

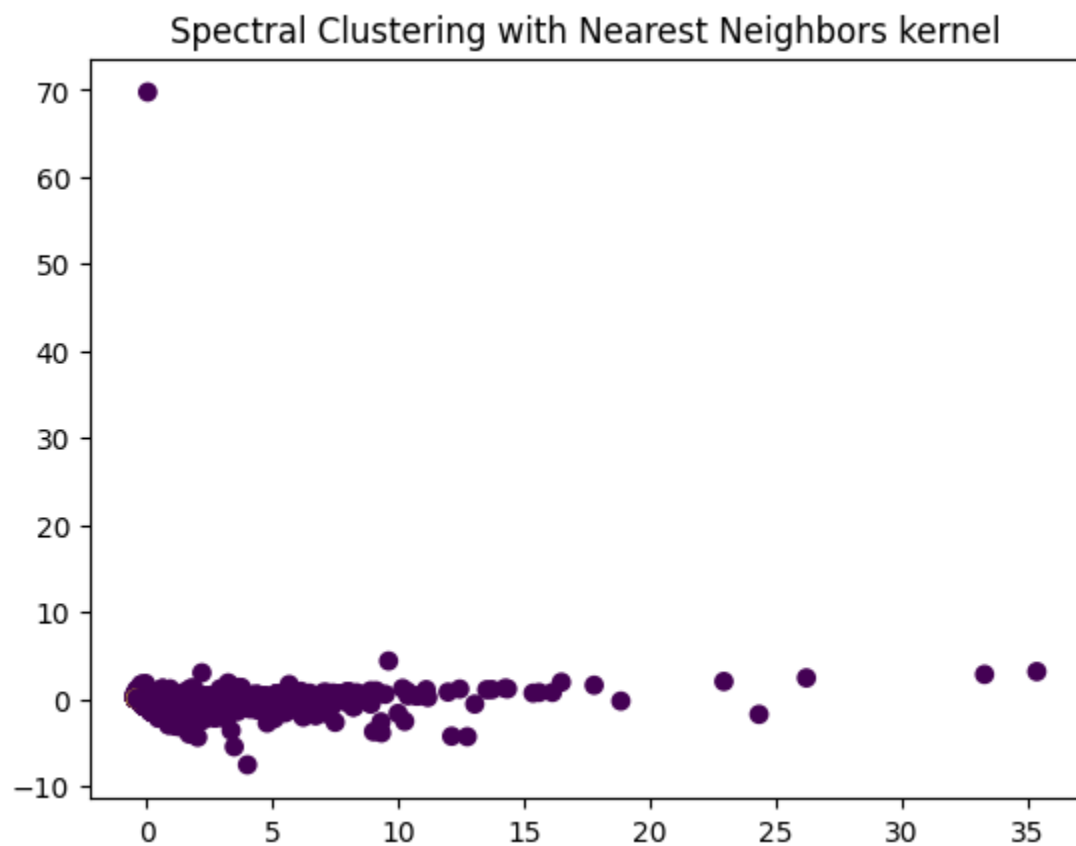
To evaluate the quality of the clustering results, three metrics are used: Calinski-Harabasz score, Silhouette score, and Davies-Bouldin score. These metrics are printed for both clustering algorithms using the `calinski_harabasz_score()`, `silhouette_score()`, and `davies_bouldin_score()` functions from the `sklearn.metrics` module.

The results of the clustering analysis show that the second clustering algorithm that used nearest neighbors as the affinity measure produced better clustering results compared to the first algorithm that used RBF as the affinity measure. Specifically, the second algorithm achieved higher values for all three evaluation metrics:

	Random state	Cluster sizes	calinski_harabasz_score	silhouette_score	davies_bouldin_score
0	0	2	187.213733	0.668853	3.545747
1	1	2	14.427727	-0.177267	1.456660

Therefore, it can be concluded that the Spectral Clustering algorithm with nearest neighbors as the affinity measure produced better clustering results for this dataset. These results should be reported in any analysis or study that uses this clustering algorithm on this dataset.



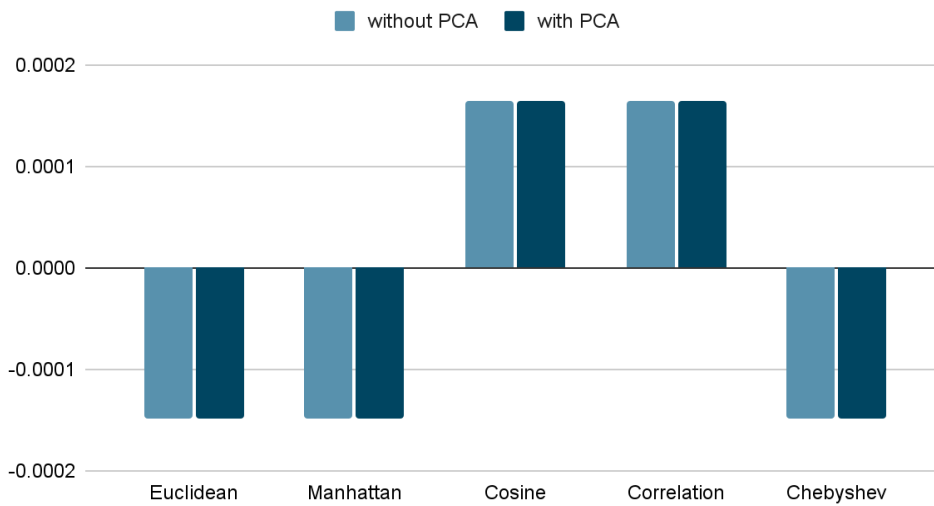


5. Prediction

In prediction, we have manually labeled 1000 data points to evaluate our model.

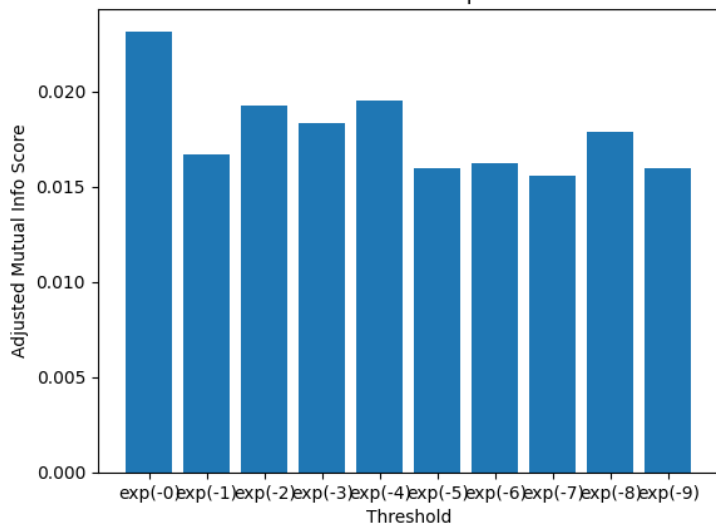
For Agglomerative Clustering the mutual info scores are:

Mutual info score

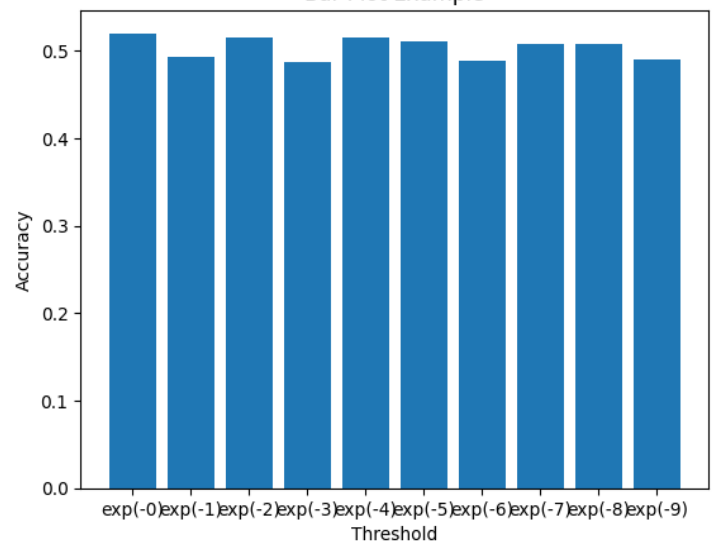


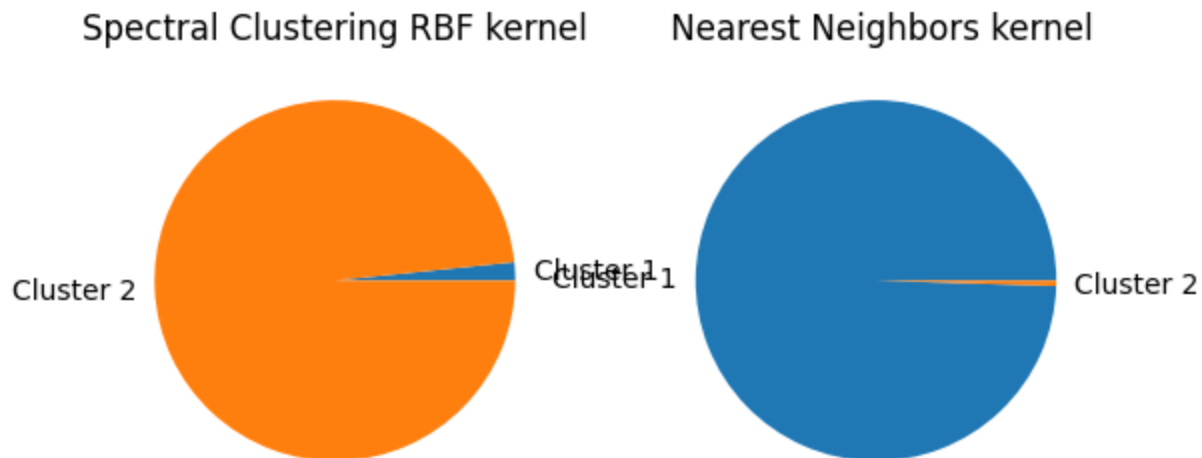
For Agglomerative Clustering the mutual info scores and accuracy scores are:

Bar Plot Example



Bar Plot Example





Conclusion :

Agglomerative clustering does not work properly. The reason is that there could be an outlier that is interfering with the algorithm. It is possible that sampling is too small for accurate distribution but without access to larger RAM or Colab Pro we are not able to run the algorithm. Although cosine and correlation seem to give better separation but not much.

Spectral clustering does not work properly. The reason is that there could be an outlier that is interfering with the algorithm. It is possible that sampling is too small for accurate distribution but without access to larger RAM or Colab Pro we are not able to run the algorithm. Although the nearest neighbor's kernel gives better separation as compared to the RBF kernel , not much.

Kmeans clustering performed better than agglomerative as well as spectral clustering.

Overall all the clustering algorithms did not perform well because the dataset we worked on was very small as compared to actual dataset, because of the limitations of google colab.

