ML ASSIGNMENT 3

OMKAR BHANDARE (22CS30016)

Part B: K-Means Clustering - Anuran Calls Dataset (MFCCs)

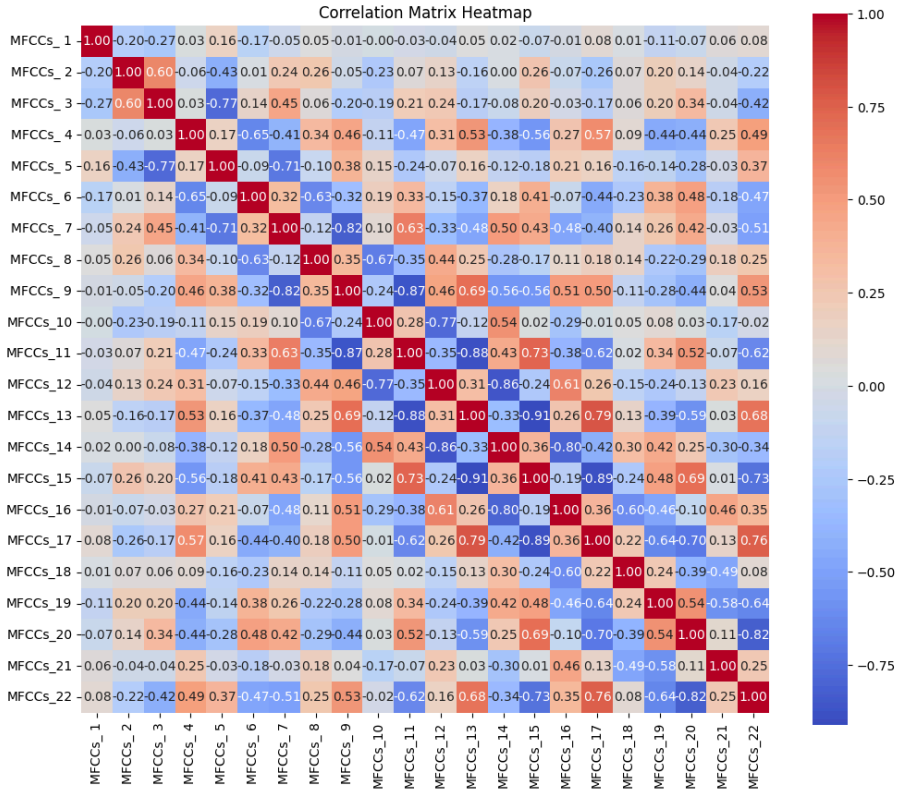
Introduction

The Anuran Cells Dataset was created by segmenting 60 audio records belonging to 4 families, 8 genera and 10 species. Each audio corresponds to one specimen (an individual frog). The recordings were processed with the help of modern means of processing, and 22 MFCCs were created out of it. These are treated as features in the dataset.

The objective of the assignment is to apply advanced clustering techniques, starting with K-Means, to group the frogs into clusters based on the MFCC features and explore clustering performance using additional evaluation methods.

Exploratory Data Analysis

Histogram plots with KDE were used to analyse the distribution of the features (all the graphs can be seen in the submitted Jupyter Notebook).
The NA values in the dataset were observed, but none were found.
The columns were standardised using the StandardScaler of the sklearn. It helped standardise features to have a balanced scale.
Next, outliers were detected using the Z-score method; a higher threshold of 4 was kept to eliminate the “far trouble-causing” outliers. The outliers were detected feature-wise, and then respective data points were removed from the dataset; this method of eliminating the outliers helps in a more nuanced understanding of the dataset, thus helping for a better approach. The summary of outliers feature-wise has been included in the code itself.

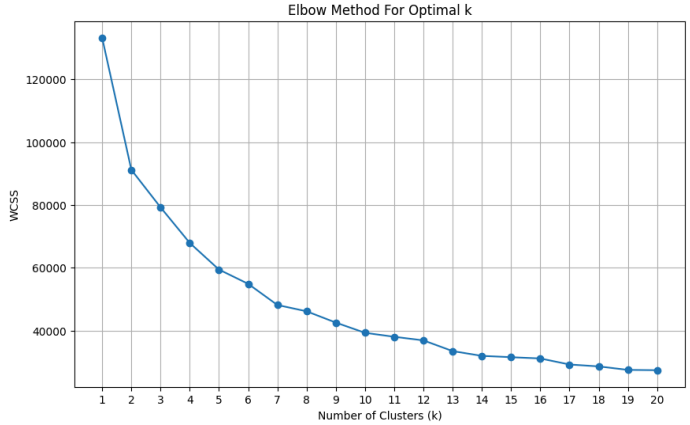


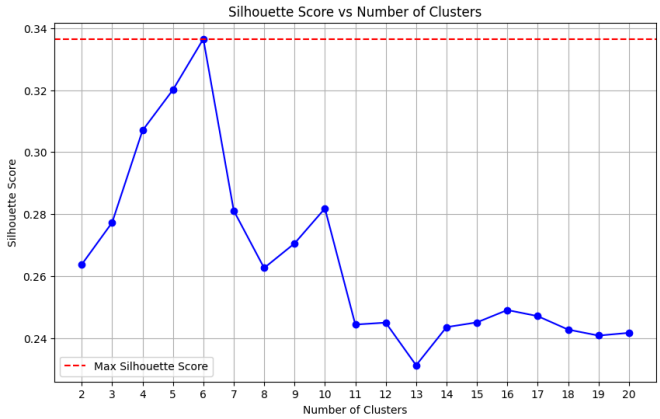
Since I do not carry any domain knowledge, developing exquisite features based on the combination of the available features and their study was not possible. However, some feature engineering techniques which do not require domain knowledge were tried. Out of those, one stood out. The correlation matrix was plotted to observe the closely related features, and a reasonably good threshold was set to consider the correlated pairs. Now, for the correlated pairs, a new feature that contained the product of values of the correlated pairs was developed, and the original features were removed. By doing this, in essence, we eliminated the problem of multicollinearity in the data by removing the correlated features; at the same time, we retained the information that the feature essentially held by taking the product. This method was adopted because after trying out the polynomial feature generation, this yielded better results than those. Since feature engineering is a vast study area, many more techniques can be explored there.

K-Means Clustering

K-Means Clustering is a popular unsupervised machine learning algorithm that groups similar data points into clusters.
It aims to partition a dataset into K distinct clusters, where K is a predetermined number. The algorithm iteratively assigns data points to the nearest cluster centroid and updates the centroids based on the designated points. This process continues until the cluster assignments stabilize.
The elbow method is used to determine the optimal number of clusters in a dataset. It involves calculating the Within-Cluster Sum of Squares (WCSS) for a range of K values. WCSS measures the sum of squared distances between data points and their assigned cluster centroids.

As the number of clusters(K) increases, the WCSS generally decreases. However, at a certain point, the rate of decrease slows down, forming an "elbow" shape in the plot of WCSS versus K. The optimal K value is typically chosen at this elbow point, where adding more clusters doesn't significantly reduce the WCSS

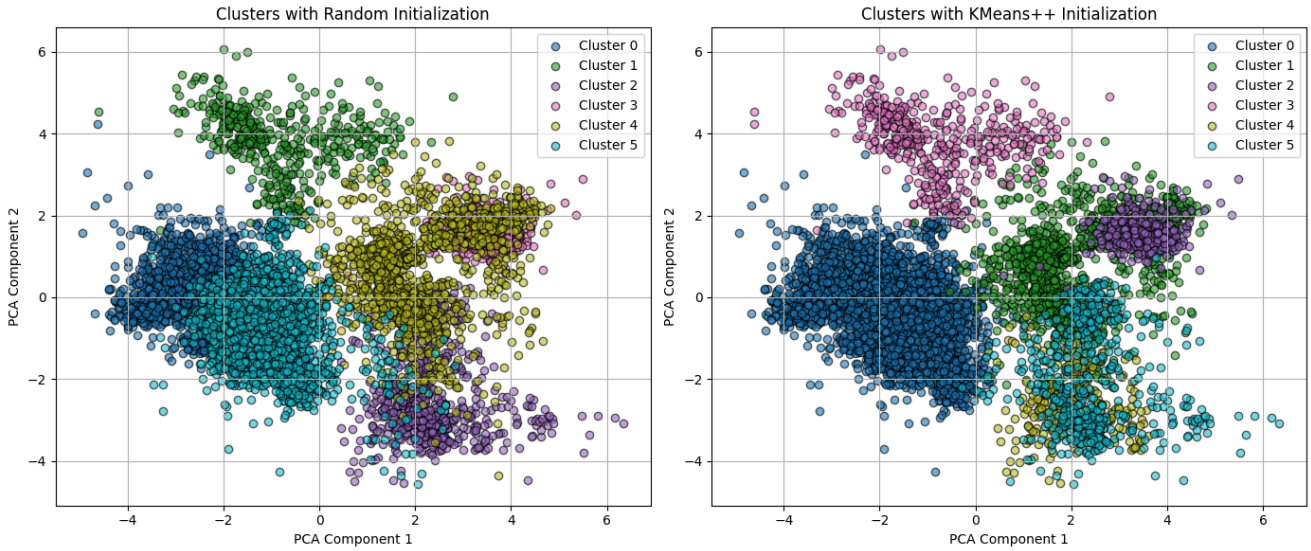




The Silhouette Coefficient, or the Silhouette Score, is a metric to evaluate the clustering quality.
It measures how similar a data point is to its cluster (cohesion) compared to other clusters (separation).
Based on the Elbow graph and the Silhouette Score plot, the optimal number of clusters for the given dataset and proposed preprocessing can be seen as 6. Further, all the reports will be based on this optimal number of clusters.

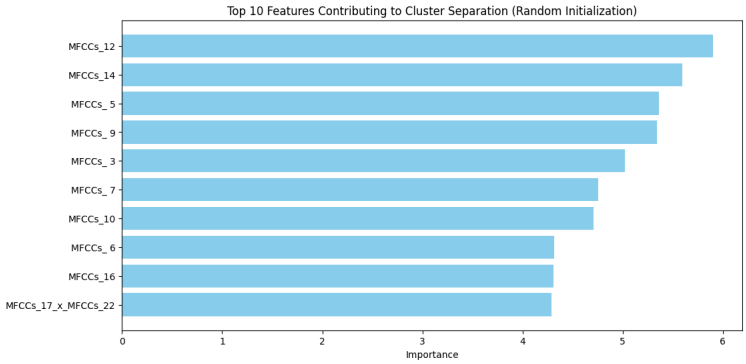
The K-Means clustering can have two different methods of initialization of the centroids.
Random Initialization: Randomly selects K data points from the dataset as initial cluster centroids.
K-Means++ Initialization: Selects the first centroid randomly. For each subsequent centroid, select a data point with probability proportional to its distance from the nearest existing centroid.
Both of the above-mentioned initialization methods were explored, and their performance was compared on the basis of the Silhouette Score, keeping the number of clusters optimal, as mentioned above.

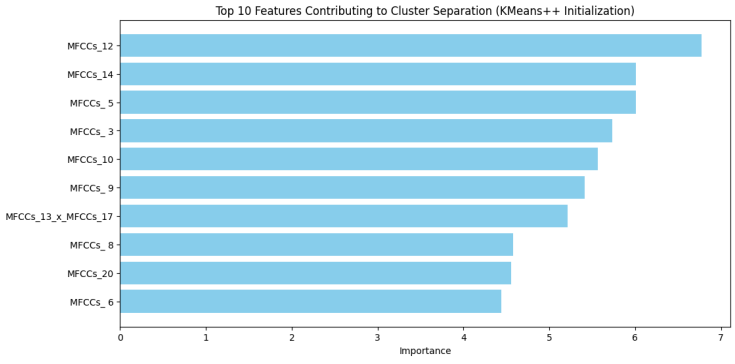
```
Silhouette Score (Random Initialization): 0.2688431410941967
Silhouette Score (KMeans++ Initialization): 0.33649080750501986
```



As expected, the KMeans++ initialization performed better in the scenario, but not by a more significant margin; this may be because of the dataset or the number of clusters. This can be further explored by adapting different preprocessing techniques, changing optimal number of clusters, etc.

The method used here to extract feature importance is based on calculating the absolute difference between the cluster centroids and the overall mean of the data for each feature. This technique provides insight into how much each feature contributes to the clustering structure compared to the average data distribution.

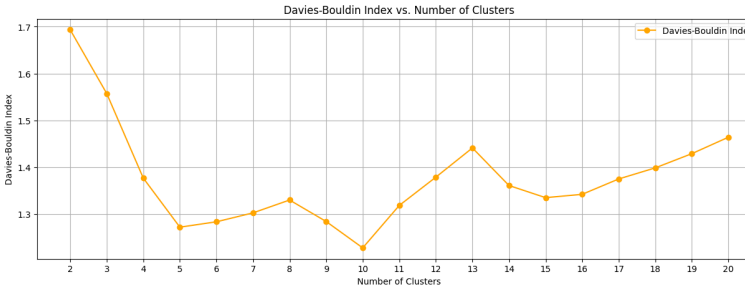


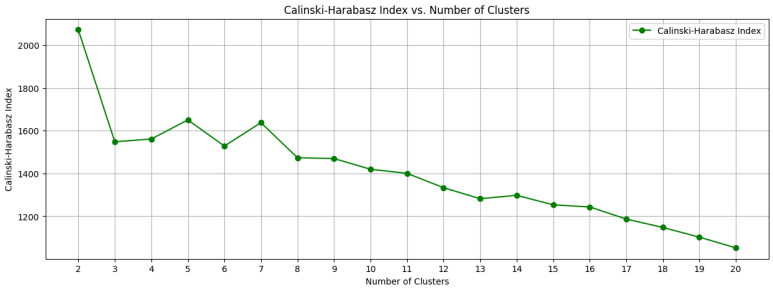


In addition to the Silhouette Score, two other cluster evaluation metrics were used to analyse the quality of the clusters, namely:

Davies-Bouldin Index: It measures the average similarity between each cluster and its most similar cluster; a lower DB index suggests that the clusters are well separated and internally compact.

Calinski-Harabasz Index: The ratio of between-clusters dispersion and within-cluster dispersion; a higher CH index indicates well-separated and internally compact clusters.





As we can observe from the Index vs Number of Clusters graph above, the selected optimal value of k=6 is backed by a low DB index and reasonably high CH index values.

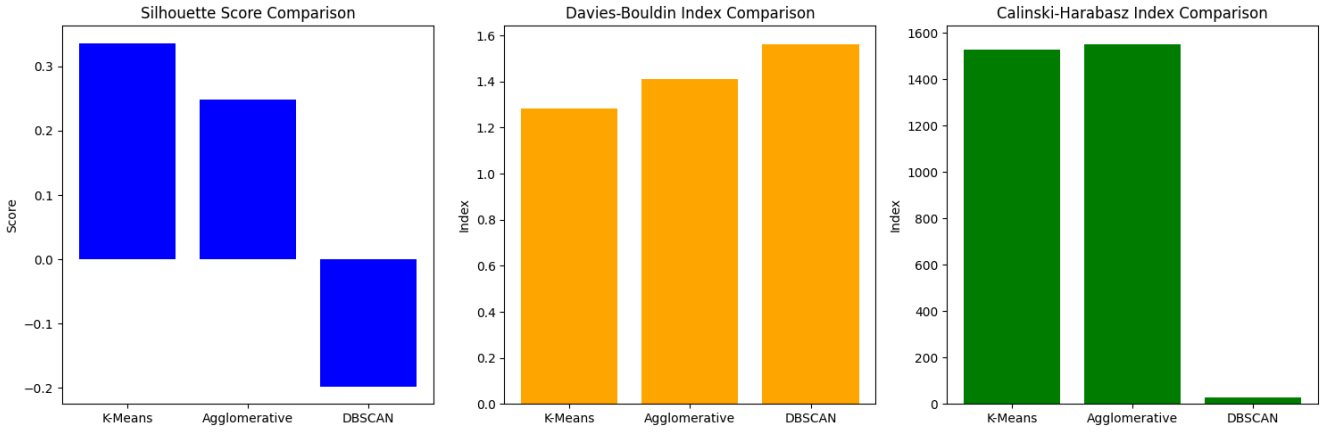
Comparison with other Clustering Algorithms

For comparison, we will consider Agglomerative clustering and DBSCAN clustering algorithms.

Agglomerative clustering is a hierarchical clustering technique that starts by treating each data point as a separate cluster. It then iteratively merges the closest pair of clusters based on a distance metric until all data points belong to a single cluster.

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is a popular density-based clustering algorithm. Unlike distance-based methods like K-Means, DBSCAN groups together points that are closely packed, and marks as outliers points that lie alone in low-density regions.

The comparison will be done based on the Silhouette Score, DB index, and CH index.



A negative Silhouette Score for the DBSCAN algorithm is suggestive that, the algorithm is not able to group the data points in a meaningful way based on the current parameter settings (default parameters used). The Agglomerative clustering is working reasonably, as good as the K-Means. DBSCAN for this

dataset performs well when the DB index is considered; however, it performs poorly when the other two metrics are considered.

Limitations of Clustering Algorithms

- K-Means:
    - K-Means is limited by its reliance on spherical clusters, as it assumes clusters are convex and similar in size. For datasets with non-spherical or unevenly distributed clusters, K-Means may underperform, as seen in some metrics for this dataset.
    - Sensitivity to outliers can affect K-Means clustering, although the preprocessing steps aimed to mitigate this.
- Agglomerative Clustering:
    - While more flexible, agglomerative clustering can be computationally intensive for large datasets. Its hierarchical nature, however, made it adaptable to the dataset's natural grouping structure.
- DBSCAN:
    - DBSCAN, being density-based, struggled with this dataset, possibly due to inadequate parameter tuning or a lack of density-based clusters. The negative silhouette score suggests that DBSCAN was unable to capture meaningful groupings within the data.

Key Insights and Conclusions

- Clustering Effectiveness: K-Means with five clusters was the most effective method, as confirmed by multiple metrics and supported by visualization.
- Evaluation Metrics: The combined use of the silhouette score, DBI, and CHI provided a robust framework for evaluating clustering quality, reinforcing the choice of 5 clusters.
- Algorithm Suitability: K-Means was suitable due to the dataset's characteristics, while DBSCAN may not be well-suited for datasets without clear density separations. Agglomerative Clustering could have been equally impactful as that of K-Means, as seen in its performance using metrics