Collin Paiz (1576109)

Professor Anastasiu

COEN 140

1 June 2023

Program 3

Clustering

Rank & NMI

Rank: 5

NMI: 0.2848

Leaderboard: Visible

My Approach

My approach to this clustering problem begins with initialization, importing the necessary packages and reading in the *train.dat* dataset. I then take the *train* dataset and isolate its values, storing them in *X_train*, which is later used for preprocessing and fitting.

Next up in my approach is to preprocess the data to ensure that it is fit for clustering. The first step for preprocessing is to use the *StandardScaler* to standardize the features of the *train* dataset and storing the results to *X_standard*. The next step in preprocessing is normalizing the samples using the *Normalizer*. This step helps deal with data that varies in magnitude/scale, which fits perfectly for our EKG dataset. The normalized data is then stored to *X_normal*.

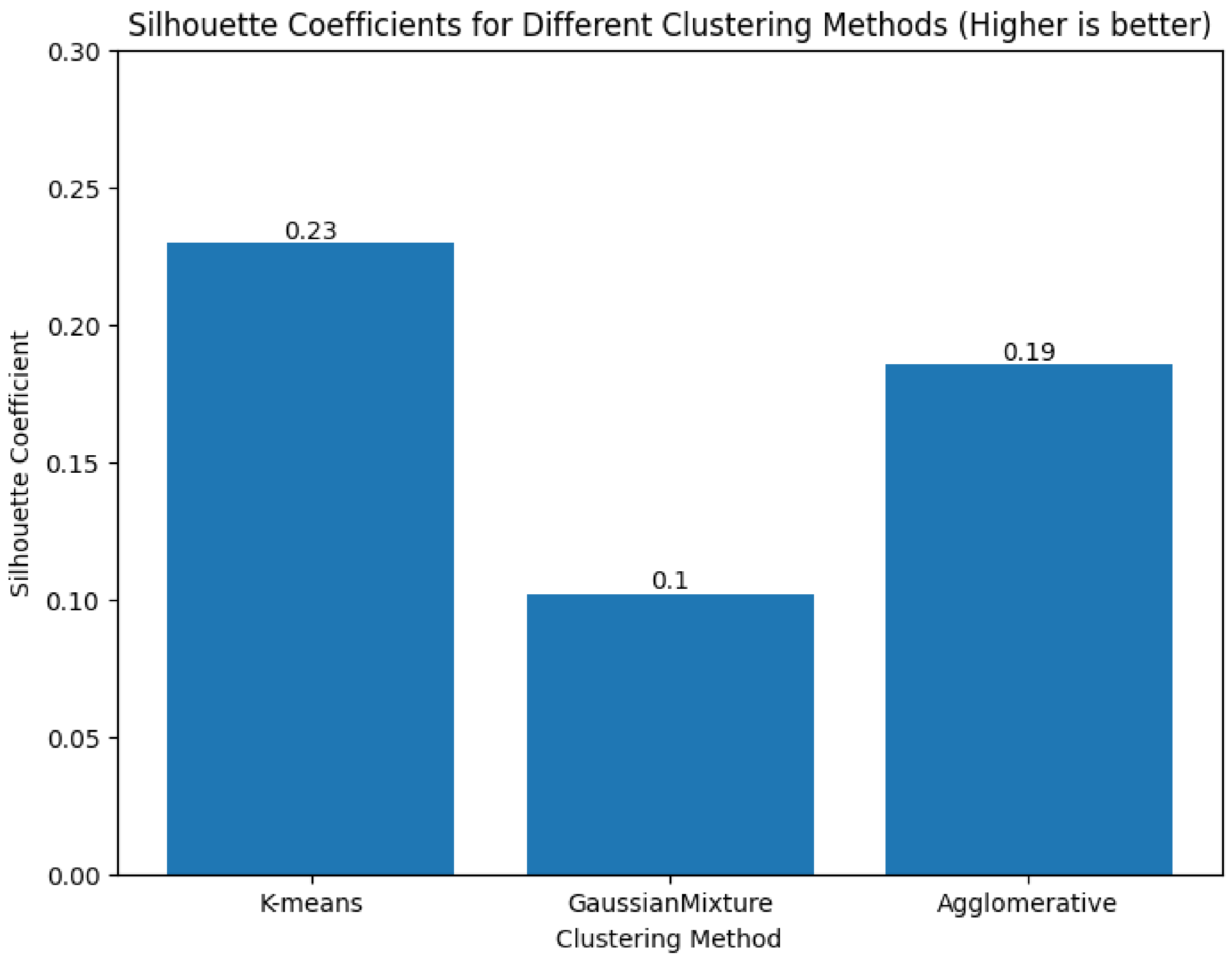
After this, I move into feature extraction/reduction. I tried a handful of different approaches here before eventually settling on *UMAP*. I first started with *PCA*, to perform feature extraction, which yielded decent results. Next, I tried a non-linear approach by applying autoencoding using *Keras*. This didn't seem to achieve any better results than *PCA* upon my initial tests (before tweaking the approach and trying different parameters/layers. After that I tried using *UMAP*, another non-linear approach to feature reduction. This yielded even better results and is the feature reduction method that I decided to move forward with.

Following all the preprocessing and preparation, I move into actually fitting the model. I first opted to use *K-Means* for my model because of its efficiency and its slightly improved performance to *AgglomerativeCluststering.* I then decided to try out the mixture of Gaussian model, which showed to have slightly improved results, which is the model I settled for.

After fitting the model, I use *X_umap* to make my clustering predictions and store this in *clusters*. I then take *clusters* and store it in a *panda's DataFrame*, and finally output it to a file, *output.dat*.

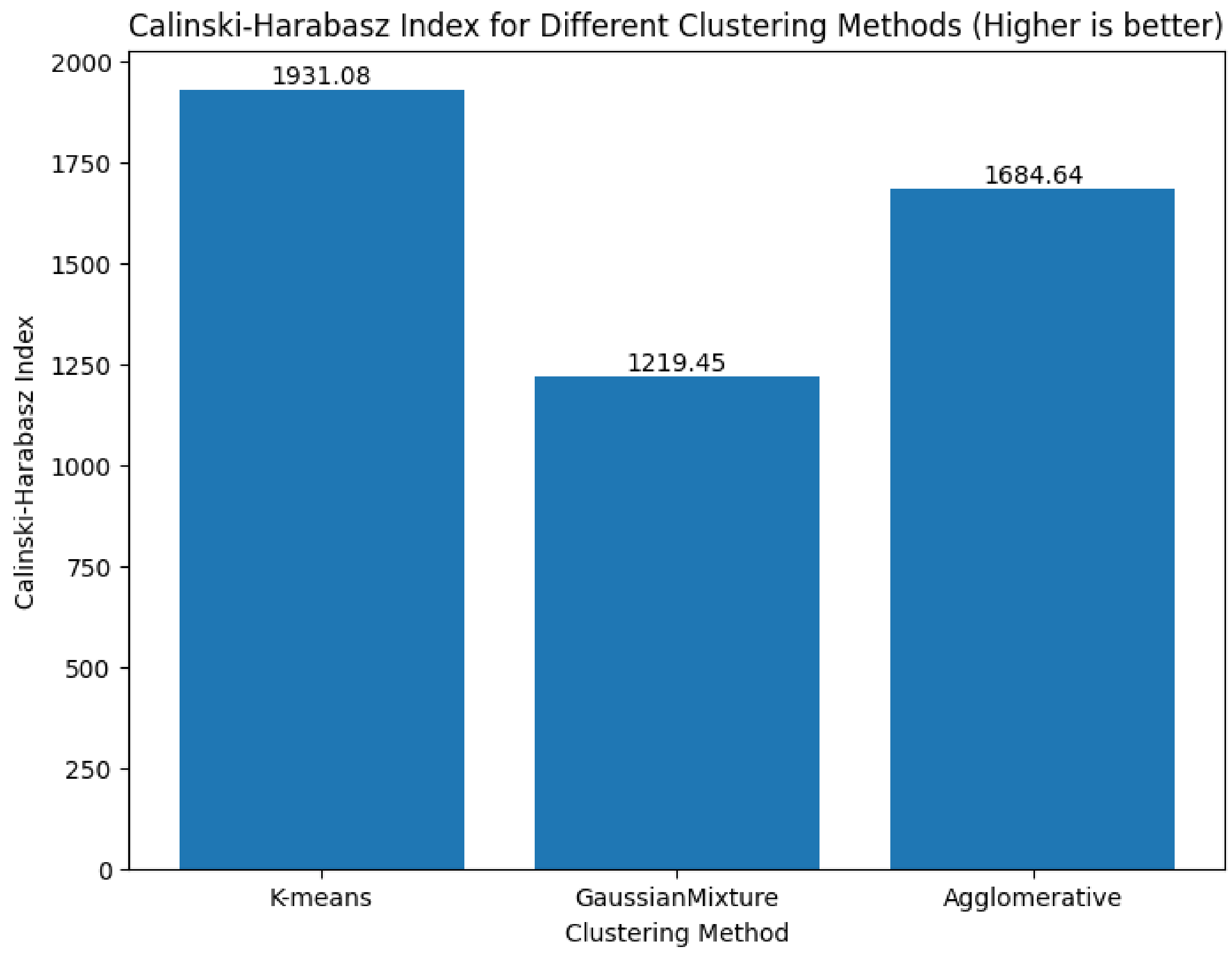
Evaluation

For my first internal evaluation metric, I chose to use Silhouette Score, which measures the quality of the clustering results by assessing the compactness and separation of clusters for the samples in the dataset.



I ended up getting results that went against the NMI score that I received from submitting my clustering results. *GaussianMixture* yielded the best NMI score in the CLP submission, but had a drastically lower silhouette score, which indicates that the clusterings were less well-separated compared to *K-means* and *Agglomerative Clustering*.

Due to these contrasting results, I decided to test out another evaluation metric, Calinski-Harabasz Score (another internal evaluation metric), which also measures the quality of clustering results by assessing the compactness and separation of clusters by calculating the ratio of the sum of between-cluster dispersion and of within-cluster dispersion.



I again found that *K-means* and *Agglomerative Clustering* had a better clustering separation compared to *GaussianMixture*, which was interesting to see because that did not quite agree with the NMI score I received from CLP.

Despite these internal evaluation metrics, I still decided to move forward with *GaussianMixture* because it resulted in the highest NMI score from CLP, which is evaluated using the true clustering labels.

Feature Selection/Reduction

My method for feature reduction was quite simple. I chose to use *UMAP*, a popular non-linear feature reduction/dimensionality reduction technique. I played around with various values for *n_components*, to test out how changing the reduced number of features affects the final clustering results. After trying out different values, I ended up using a value of 10 for *n_components* because it yielded slightly better results compared to other values. *UMAP* worked well because it is a non-linear feature reduction technique, which bypasses the assumption that the data is linearly correlated from linear feature reduction techniques.

Suggestions For Running

- Simply run all cells in order as is
- Results outputted to *output.dat*