

Contents

[***1. Introduction***](#_heading=h.gjdgxs) ***2***

[***2. Preprocessing***](#_heading=h.30j0zll) ***3***

[**a. Pen-based**](#_heading=h.1fob9te) **3**

[**b. Vowel**](#_heading=h.3znysh7) **3**

[**c. Adult**](#_heading=h.2et92p0) **3**

[***3. Choosing the best value of k***](#_heading=h.tyjcwt) ***4***

[**a. First approach: Choosing the best k value using the SSE metric value**](#_heading=h.3dy6vkm) **4**

[**b. Second approach: Using the elbow method and the silhouette scores**](#_heading=h.1t3h5sf) **5**

[***4. Algorithms of clustering analysis conducted***](#_heading=h.4d34og8) ***7***

[**a. Fuzzy C Means**](#_heading=h.2s8eyo1) **7**

[**b. Agglomerative Clustering**](#_heading=h.17dp8vu) **8**

[**c. Mean Shift Algorithm**](#_heading=h.3rdcrjn) **9**

[**d. K Means, K Harmonic Means & Bisecting K Means**](#_heading=h.26in1rg) **9**

[***5. Results of cluster analysis***](#_heading=h.lnxbz9) ***11***

[**a. Vowel Dataset**](#_heading=h.35nkun2) **12**

[**b. Pen Based Dataset**](#_heading=h.1ksv4uv) **13**

[**c. Adult Dataset**](#_heading=h.44sinio) **14**

[***6. Conclusions***](#_heading=h.2jxsxqh) ***16***

# Introduction

The aim of the exercise is to implement and analyze different clustering algorithms: Agglomerative Clustering, Mean Shift, K-Means, Bisecting K-Means, K-Harmonic Means and Fuzzy C-means). To do that, we use the data from 3 datasets from the UCI repository: a numerical and medium size data set (Pen-based), a mixed and large data set (Adult) and a mixed and small data set (Vowel). Table 1 shows a summary of each of these three data sets.

| Domain | # Cases | # Num | # Nom | # Cla | Dev. Cla. | Maj. Cla. | Min. Cla. | Missing values |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Adult | 48,842 | 6 | 8 | 2 | 26.07% | 76.07% | 23.93% | 0.95% |
| Pen-based | 10,992 | 16 | - | 10 | 0.40% | 10.41% | 9.60% | - |
| Vowel | 990 | 10 | 3 | 11 | 0.00% | 9.09% | 9.09% | - |

*Table 1 - Data sets summary*

In this document, the different relevant steps to perform this analysis are shown and explained. Section 2 explains some data pre-processing that must be done before running any clustering algorithm. Section 3 analyzes the characteristics of the data sets in order to decide the best value of k clusters for each. Section 4 introduces the important concepts for the implementation of K-Means, Bisecting K-Means, K-Harmonics Means and Fuzzy C-means (FCM). Finally, section 5 presents the results of the clustering for each algorithm.

# Preprocessing

## Pen-based

The pen-based dataset consists of 15 numerical features, all of them in the range of values [0, 1]. Since this is the initial situation, the preprocessing steps for the implementation of the word for this dataset are quite simple: we simply deleted the class column.

The class of each instance is stored in this dataset in the last feature, called ‘a17’. For our preprocessing, we had to delete it since it is not used in our clustering algorithms.

## Vowel

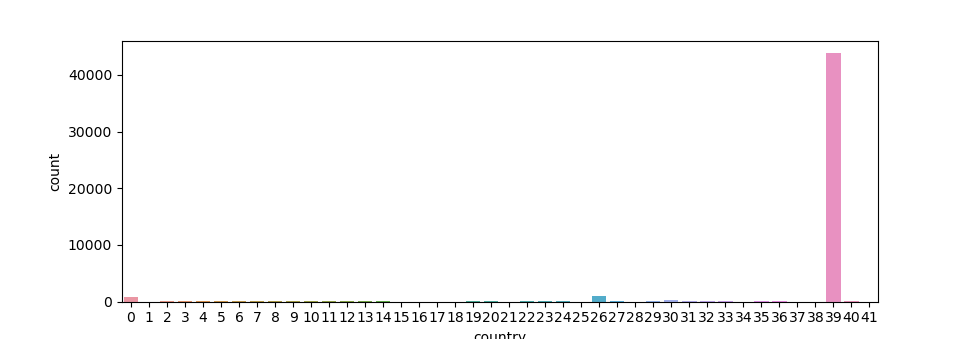
The vowel datasets contain both numerical and categorical data. This is important because he must apply some preprocessing to the dataset in order to convert all features into the same type.

Our approach has been to apply a label encoding to the categorical features and convert them to numerical values. This label encoding step will transform each categorical value into a numerical exclusive one, keeping the redundant information but in a different data type, convenient for the performance of the clustering algorithms.

Apart from this, we also deleted the class feature, called ‘Class’.

## Adult

We choose this dataset first because it’s a large dataset but it’s also a mixt dataset: it contains Categorical as well as Numerical value. The dataset contains 2 classes, has a number of values that exceeds 48 000 values and most importantly got 0,90% of missing values. So the preprocessing step will focus on some changes regarding these characteristics.

The dataset is actually a biased dataset. We actually got to this conclusion because of some EDA we applied on the dataset. First, we tested with the variable Country and plotted the distribution. We got this result. 

*Figure 1 : Plot of the class country to see the biase*

As we see in the plot result of the variable country, the USA got the most of the values present in the dataset. This is a proof that some of the dataset variables are biased. For the variable Hours Per Week, it’s again the same issue.



*Figure 2 : Results of the hours per week class*

To have logical clustering analysis results, we will perform the preprocessing functions below.

First we will start by encoding our dataset using the Label Encoder. So as every categorical variable turns into a numerical one.

We will then take care of the missing values by replacing the “?” with np.nan. This special character needed to be changed since it represents nothing. We will then be dropping the missing values since for us, dropping them won't affect the dataset much.

Encoding inclus as well the variable Class, where we affect 0 and 1 to < 50 and > 50 respectively.

Since our data is biased, we can scale it a bit by using the minmaxscaler. This will help us to get a more scaled and equilibrated data.

# Choosing the best value of k

## First approach: Choosing the best k value using the SSE metric value

For the different algorithms we implemented it is necessary to specify a priori a value of k, being k the number of clusters for the algorithm to find. This k value must be set in a certain criterion, since it is a critical parameter that will define the whole algorithm.

In our case, to choose the best value of k we calculated for each cluster k the sum of squared errors of the distances between each datapoint and the centroid of the cluster. We decided to calculate this with the fuzzy c means algorithm.

As this is a sum of errors, we want to determine the value of k that minimizes it.

In the following plots, we can see for each dataset the relation between the number of clusters defined a priori and the SSE metric value.



*Figure 3 - K selection curve for each dataset*

From the figures above (figure 1), we decided to choose the following number of clusters for each dataset (this is, the k value):

* Adult: 2
* Vowel: 11
* Pen-based: 10

We can see that the values chosen don’t necessarily correspond to the minimum value of the SSE metric, this is because the behavior of the curve is also analyzed and considered. For example, in the adult dataset curve (first plot of the figure 1) we can see that k=2 is the first value where the curve starts to flatten. Before this (k<2) the curve is descending in a really pronounced way, and after (k>2), it slightly descends, with little variations in the results obtained. This kind of behavior has been also considered to try to obtain the optimal k value.

For the vowel dataset (plot in the middle of the figure 1) we can notice an increase in the value of the SSE after k=11. Until that point, values are decreasing with different levels of pronunciation, but from k=11 to k=12, it can be observed that the SSE value increases, therefore we decided to define k=11 as the optimal number of cluster values.

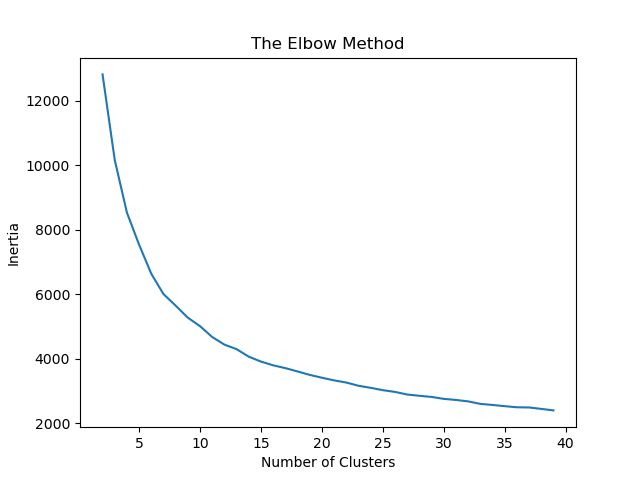
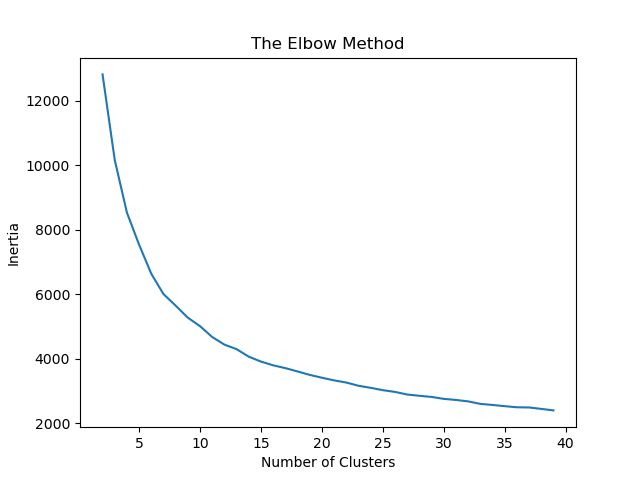
Similarly, in the pen-based dataset (third plot in figure 1) we set the optimal number of clusters in k=10, since as in the vowel case, it is the last value of k before the SSE increases.

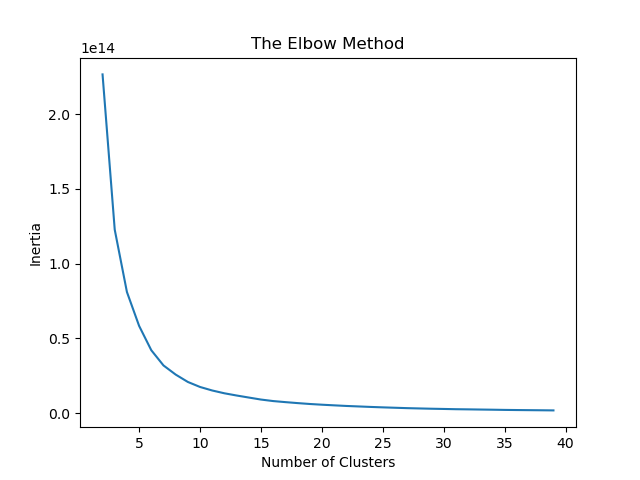
## Second approach: Using the elbow method and the silhouette scores

For the second approach, we will try to use the Elbow Method and calculate the Silouhette score to be able to choose the best k to use. This method will help us to get the best k per dataset, so as we get the best results while using our different algorithms.

Normally, to use the elbow method, we needed to follow this general rule : Calculate the Within-Cluster-Sum of Squared Errors (WSS) for different values of k, and choose the k for which WSS first starts to diminish. In the plot of WSS-versus-k, this is visible as an elbow.

So started by understanding the elbow method. First, the Squared Error for each point is the square of the distance of the point from its representation. Secondly, the WSS score is the sum of these Squared Errors for all the points. And finally, any distance metric like the Euclidean Distance or the Manhattan Distance can be used in our case.

We first start by defining the possible k values, and for each of our values, iterate through, take each value from, fit it in our dataframe and then append the inertia to our array. We can then plot the results. For each of the dataset, we get the respective plots: 

*Figure 4 : Vowel Elbow Method Figure 5 : Pen Based Elbow Method*

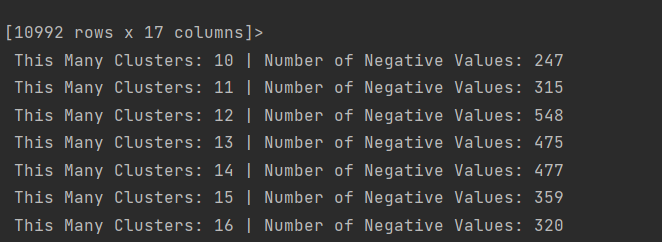
‘

*Figure 6 : Adult Elbow Method*

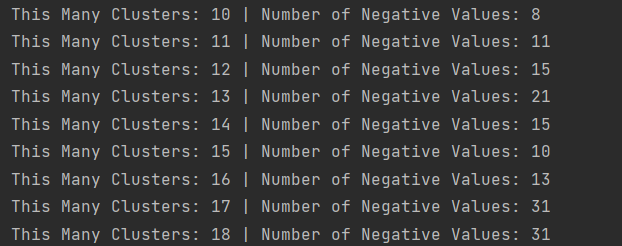
From the figures, we can get a clear idea of where the elbow is located, but then we get to see different values of K where the elbow is present.

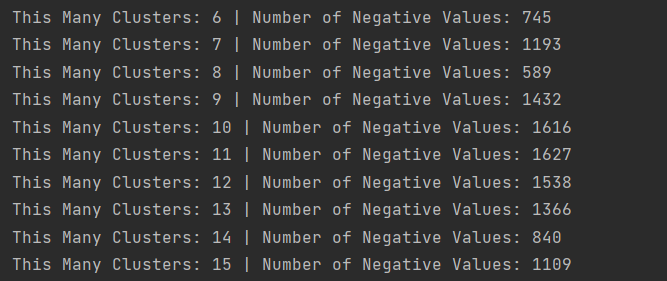
This is why, we are using a second method which is the silhouette method to get to see what’s the best k we should be using in our case. To choose the best k, we will calculate the silhouette score.

Actually the silhouette method measures how similar a point is to its own cluster which is what we call cohesion compared to other clusters which is called separation. The silhouette score is a value between +1 and -1. The closer to 1, the better it is. For our case, we are calculating all the negative values per cluster number. If a cluster has a lot of negative values then it’s not the best k cluster. In our case, we are choosing the number of clusters with less negative values.



*Figure 8 : S Score of the Vowel Dataset*



*Figure 9 : S Score of the Pen Based Dataset* 

*Figure 10 : S Score of the Adult Dataset*

Calculating the scores will help us find the best K, we need to take the k which got the less negative values. So finally, we get the same Ks we got using the first method for the Vowel and Pen Based Dataset, but for the Adult Dataset, we get different results. The K chosen for the Adult Dataset is 8.

# Algorithms of clustering analysis conducted

## Fuzzy C Means

The Fuzzy C Means algorithm is especially interesting because it defines the cluster of each sample by taking the maximum value from a membership matrix. This is based in the idea that each sample has a degree of belonging to each cluster, therefore we could say that each datapoint belongs with a different degree (membership value) to all clusters. From this, the determination of a cluster for each sample is done by evaluating the membership values and taking the maximum one as result.

This procedure is implemented in a loop that stops under a performance condition or a defined number of iterations. For each iteration, the membership values of each data sample are calculated with the updated values of the centroids of each cluster.

The computation of the Fuzzy C Means algorithm has, accordingly to the explained above, two main parts: the computation of new centroids and the computation of the membership matrix.

These calculations are made with the following equations:

* Membership matrix (U)
* Centroids matrix (V)

For the Fuzzy C Means implementation is necessary to define the parameters that will define the behavior of the method. These parameters that must be set are:

* *m* (degree of fuzziness)
* *c* (number of clusters to find)

For our approach, we defined *m = 2* as it is the common value in the literature. The number of centroids has been defined for each data set as explained in section 3 in this document.

Furthermore, the maximum number of iterations for the algorithm to find clusters has been set by default to 100, and the number of times to run the algorithm before returning results has been set to 10. After these 10 iterations, the best result of all is kept by selecting the minimum value of a defined performance index. This index is based on intra-class and inter-class similarity (distance).

## Agglomerative Clustering

The implementation of the agglomerative clustering procedure has been done using the agglomerative clustering function from the *sklearn* python library. For the different test cases we had to create, two parameters were modified:

* *affinity*
* *linkage*

These parameters refer to the metric used to compute distance and linkage criterion to use, respectively. The affinities we have worked on are the Euclidean and Cosine distance, and the different linkage methods used have been: complete, average and single.

## Mean Shift Algorithm

As in the case of the agglomerative clustering, the mean shift algorithm used for the work has been the one from the *sklearn* python library.

This algorithm intends to discover clusters in a smooth density of samples. It is a centroid-based algorithm, which sets the value of the centroids to the mean of the data points in a region.

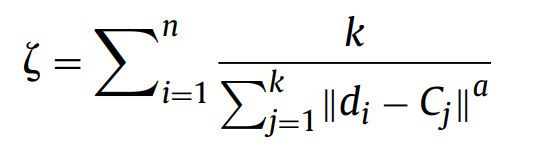
## K Means, K Harmonic Means & Bisecting K Means

**K-means Algorithm :** In the case of our project. The k-means we are using is our proper k mean algorithm implementation. We followed the general algorithm logic to be able to create the adequate function. In fact K-mean clustering tries to group similar kinds of items in form of clusters. It finds the similarity between the items and groups them into the clusters. K-means clustering algorithm works in three steps.

First, we select the k value. Secondly, we initialize the centroids and finally we select the group and find the average. But if we want to explain in details the steps, we will dividing them into steps:

* Step-1: Select the number K to decide the number of clusters.
* Step-2: Select random K points or centroids. (It can be different from the input dataset).
* Step-3: Assign each data point to their closest centroid, which will form the predefined K clusters.
* Step-4: Calculate the mean of all points belonging to once centroid and place a new centroid of each cluster.
* Step-5: Repeat the third steps, which means reassign each datapoint to the new closest centroid of each cluster.
* Step-6: If any reassignment occurs, then go to step-4 else go to FINISH.
* Step-7: The model is ready.

**K- harmonic means Algorithm :** Cluster centers updating is a problem for K-means clustering algorithm. To overcome this problem we can use the KHM clustering algorithm with a novel function to update the clusters. The KHM clustering algorithm uses the harmonic means of the distances from data points to the cluster centers in its cost function. KHM clustering algorithm uses the following cost function defined by for n data points;

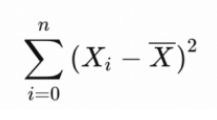


* di is the ith data point,
* Cj is the jth cluster center.
* k is the number of desired clusters.
* a is a positive number.
* The cost function uses the squared Euclidean distance (a = 2)
* KHM clustering algorithm has following steps for data having n data points and k desired clusters;

**Bisecting K-Means Algorithm :** The bisecting K-means algorithm will be as well developed by us in this project. We will use the previous K-mean algorithm developed to be able to apply the bisecting one. Bisecting K-Mean Algorithm is a simple development of the basic K-means algorithm that depends on a simple concept such as to acquire K clusters, split the set of some points into two clusters, choose one of these clusters to split, etc., until K clusters have been produced.

The steps of the Bisecting K-Means Algorithm that we followed to develop our function are :

* Step-1: Set K to define the number of clusters.
* Step-2: Set all data as a single cluster.
* Step-3: Measure the distance for each intra cluster “Sum of square Distance”.



* Step-4: Select the cluster that has the largest distance and split it into 2 clusters using K-means.
* Step-5: Repeat step 3–5 until the number of leaf clusters = K.

So as a result we get the best clusters. This algorithm performs better than the K-mean one. It’s indeed a hybrid approach between Divisive Hierarchical Clustering (top down clustering) and K-means Clustering.

# Results of cluster analysis

All datasets have been tested with all algorithms implemented in the work. The results have been obtained from different metrics, concretely, the following:

* Accuracy score: Provides the accuracy between two multilabel classification set. It assumes that the set of labels predicted for a sample must exactly match the corresponding set of labels in true classification. Note that this metric can return a zero level of accuracy even if the predicted clustering match with a 100% accuracy with the true clustering, due to the cluster labeling.
* Adjusted Random Score / Adjusted Random Index (ARS / ARI): Computes a similarity measure between two clustering results considering that the label of the clusters may differ. It considers all pairs of samples and counts pairs that are assigned in the same or different clusters in the predicted and true clusterings.
* F-Score: It is based in precision and recall. The F-Score can be interpreted as the harmonic mean of both. Also, the contribution of precision and recall is equal.
* Homogeneity score: This metric result satisfies homogeneity if all its clusters contain only data points which are members of a single class. It is important to point out that this metric is independent of the absolute value of the labels, meaning that a permutation of the class or cluster label values won’t change the score value in any way.

## Vowel Dataset

**Agglomerative clustering**

The results from applying the agglomerative clustering to the vowel dataset are the following:

|  | **ARS** | **Homogeneity score** | **Accuracy score** | **F-score** |
| --- | --- | --- | --- | --- |
| **Euclidean distance** |  |  |  |  |
| Single linkage | -0.003 | 0.043 | 0.091 | 0.096 |
| Average linkage | 0.015 | 0.099 | 0.079 | 0.007 |
| Complete linkage | 0.055 | 0.186 | 0.095 | 0.096 |
| **Cosine distance** |  |  |  |  |
| Single linkage | 0.003 | 0.081 | 0.073 | 0.015 |
| Average linkage | 0.003 | 0.093 | 0.079 | 0.04 |
| Complete linkage | 0.061 | 0.164 | 0.068 | 0.045 |

*Table 2 – Results from agglomerative clustering for vowel dataset*

**Mean Shift**

| **ARS** | **Homogeneity score** | **Accuracy score** | **F-score** |
| --- | --- | --- | --- |
| -0.001 | 7.2·10-4 | 0.091 | 0.028 |

*Table 3 - Results from the Mean Shift algorithm for the vowel dataset*

**Our own implemented algorithms**

| **Algorithm** | **ARS** | **Homogeneity score** | **Accuracy score** | **F-score** |
| --- | --- | --- | --- | --- |
| Fuzzy C Means | 0.015 | 0.072 | 0.093 | 0.088 |
| Bisecting K Means | 0.054 | 0.191 | 0.050 | 0.044 |

*Table 4 - Results from the different algorithms implemented for our own for the vowel data set*

**Conclusion**

From the results shown in the Table 4, Fuzzy C Means performs better than the Bisecting K Means in terms of the Accuracy score, and in terms of the F score. This can be indicating that the Fuzzy C Means works better and gives better results when the dataset is categorical. Or maybe it generally performs better than the Bisecting K Mean Algorithm. Yet even if it performs better, the computational time needed to apply the FCM is much more higher than the BKM.

## Pen Based Dataset

**Agglomerative clustering**

The results from applying the agglomerative clustering to the vowel dataset are the following:

|  | **ARS** | **Homogeneity score** | **Accuracy score** | **F-score** |
| --- | --- | --- | --- | --- |
| **Euclidean distance** |  |  |  |  |
| Single linkage | -5.6·10-6 | 0.001 | 0.104 | 0.02 |
| Average linkage | 0.414 | 0.567 | 0.017 | 0.021 |
| Complete linkage | 0.355 | 0.534 | 0.085 | 0.07 |
| **Cosine distance** |  |  |  |  |
| Single linkage | 1·10-4 | 0.007 | 0.103 | 0.02 |
| Average linkage | 0.311 | 0.482 | 0.079 | 0.038 |
| Complete linkage | 0.327 | 0.5 | 0.178 | 0.17 |

*Table 5 – Results from agglomerative clustering for pen-based dataset*

**Mean Shift**

| **ARS** | **Homogeneity score** | **Accuracy score** | **F-score** |
| --- | --- | --- | --- |
| 0.0 | 0.0 | 0.104 | 0.02 |

*Table 6 - Results from mean Shift algorithm for pen-based dataset*

**Our own implemented algorithms**

| **Algorithm** | **ARS** | **Homogeneity score** | **Accuracy score** | **F-score** |
| --- | --- | --- | --- | --- |
| Fuzzy C Means | 0.547 | 0.651 | 0.143 | 0.136 |
| Bisecting K Means | 0.59 | 0.70 | 0.09 | 0.08 |

*Table 7 - Results from the different algorithms implemented for our own for the pen-based data set*

**Conclusion**

From the results shown in the Table 7, Fuzzy C Means performs better than the Bisecting K Means in terms of the Accuracy score, and in terms of the F score. This can be indicating that the Fuzzy C Means works better and gives better results when the dataset is numerical. Or maybe it generally performs better than the Bisecting K Mean Algorithm. Yet even if it performs better, the computational time needed to apply the FCM is much more higher than the BKM.

## Adult Dataset

**Agglomerative clustering**

The results from applying the agglomerative clustering to the vowel dataset are the following:

|  | **ARS** | **Homogeneity score** | **Accuracy score** | **F-score** |
| --- | --- | --- | --- | --- |
| **Euclidean distance** |  |  |  |  |
| Single linkage | 8.923·10-5 | 5.321·10-5 | 0.761 | 0.657 |
| Average linkage | 3.309·10-5 | 2.514·10-6 | 0.239 | 0.092 |
| Complete linkage | -0.019 | 0.041 | 0.515 | 0.54 |

*Table 8 – Results from agglomerative clustering for adult dataset*

**Mean Shift**

| **ARS** | **Homogeneity score** | **Accuracy score** | **F-score** |
| --- | --- | --- | --- |
| -0.027 | 0.055 | 0.495 | 0.524 |

*Table 9 - Results from the Mean Shift algorithm for the adult dataset*

**Our own implemented algorithms**

| **Algorithm** | **ARS** | **Homogeneity score** | **Accuracy score** | **F-score** |
| --- | --- | --- | --- | --- |
| Fuzzy C Means | -0.032 | 0.046 | 0.502 | 0.526 |
| Bisecting K Means | 0.023 | 0.177 | 0.229 | 0.328 |

*Table 10 - Results from the different algorithms implemented for our own for the adult data set*

**Conclusion**

From the results shown in the Table 10, Fuzzy C Means performs better than the Bisecting K Means in terms of the Accuracy score, and in terms of the F score. This can be indicating that the Fuzzy C Means works better and gives better results when the dataset is numerical. Or maybe it generally performs better than the Bisecting K Mean Algorithm. Yet even if it performs better, the computational time needed to apply the FCM is much higher than the BKM.

# Conclusions

With the results obtained in section 5 we can observe that the best performance is accomplished with the Fuzzy C Means algorithm, in our case, for all the data sets in general, but the maximum score is obtained concretely with the agglomerative algorithm applied to the adult data set, obtaining a F-score of 65.7% and an accuracy score of 76.1%.

It is important to notice that the larger the dataset is, the best results we can obtain. For the adult data set, which is nearly 50,000 instances long, the results obtained are much better than the ones obtained from the rest of the data sets. Also, we must consider the fact that the adult data set has only two classes, that is, clusters to classify the instances into, and this can also be a reason for the performance when working with this data set in respect to the others.

Even so, we must highlight that running these algorithms for such a long data set takes much more computational time when compared to the shorter ones. This is important because in our work some algorithms could not be performed on this data set, since the time for them to run the code and obtain the results was hours long. In these cases, the use of a powerful computer and an efficient algorithm is advisable. Concretely, the agglomerative clustering with cosine distance was not reproduceable. Also, we must point out that the agglomerative clustering algorithm with Euclidean distance was performed but it lasted hours to end up with some results. We don’t recommend testing it since it might take too much time.

We cannot conclude that there is a difference between the results obtained for the numerical data set (pen-based) and the mixt data sets (vowel and adult). We think that the differences between the results might be because of the number of instances of each one, and the number of classes that they must be clustered into.

To conclude with, we would like to say that it is surprising what a simple algorithm is capable of. In our opinion, these simple algorithms have been able to, with higher or lower accuracy, obtain some results similar to the ground truth, which means that, with a lot of other steps yet to add (such as PCA analysis), they have obtained quite good performances. Adding these other steps, would increase the accuracy of the results, and so, define models more effective and valid. Nevertheless, it has been shown that the application of a cluster algorithm to unsupervised data, with some a priori analysis, can lead us to a first understanding of the data in a short time, compared with the task of doing it manually, independently of the number of instances and features this data has. However, it has also been seen that large data sets also carries a higher computational cost.

**References**

[1] C. L. DuBois, “UCI network data repository.” http://networkdata.ics.uci.edu, 2008.

[2] J. V. D. Robert L. Cannon and J. C. Bezdek, “Efficient implementation of the fuzzy c-means clustering algorithms,” 1986.

[3] Bisecting Kmeans Clustering. Bisecting k-means is a hybrid approach… | by Afrizal Firdaus | Medium [WWW Document], n.d. URL<https://medium.com/@afrizalfir/bisecting-kmeans-clustering-5bc17603b8a2> (accessed 10.23.22).

[4] Bisecting K-Means Algorithm Introduction, 2020. . GeeksforGeeks. URL<https://www.geeksforgeeks.org/bisecting-k-means-algorithm-introduction/> (accessed 10.23.22).

[5] Di, J., Gou, X., 2018. Bisecting K-means Algorithm Based on K-valued Selfdetermining and Clustering Center Optimization. Journal of Computers 588–595.<https://doi.org/10.17706/jcp.13.6.588-595>

[6] coding, D.K.D.K. has years of experience as a S.D.S.H. enjoys, teaching, everyone, has created this website to make M.L. accessible to, 2022. K-Means Accuracy Python With Silhouette Method » EML. URL<https://enjoymachinelearning.com/blog/k-means-accuracy-python-silhouette/> (accessed 10.23.22).

[7] Selecting the number of clusters with silhouette analysis on KMeans clustering [WWW Document], n.d. . scikit-learn. URL<https://scikit-learn/stable/auto_examples/cluster/plot_kmeans_silhouette_analysis.html> (accessed 10.23.22).

[8] sklearn.cluster.BisectingKMeans — scikit-learn 1.1.2 documentation [WWW Document], n.d. URL<https://scikit-learn.org/stable/modules/generated/sklearn.cluster.BisectingKMeans.html> (accessed 10.23.22).

[9] Zhang, B., Hsu, M., & Dayal, U. (1999). K-Harmonic Means–A Data Clustering Algorithm (Technical Report HPL-1999-124). *Palo Alto: Hewlett-Packard Laboratories*.