Film Rating Prediction Using Unsupervised learning

Juliana Henao
Ingeniería Matemática
Universidad EAFIT)
Medellín, Colombia
jhenaoa4@eafit.edu.co

***Abstract*—Film Industry is one of the most important industries in the world, making a big impact in our society. But film making requires many resources. Producers are interested in finding the future commercial performance of a film project. In this project, a set of neural networks predict if a film will be nominated to an Oscar Award. This neural networks try to find patterns within the films that have the most award success. This research compares neural networks, making combinations of number of neurons per layer and hyper-parameters, on the data set formed by the processing some characteristics of a film.**

***Index Terms*—non supervised, clustering, films, Mountain clustering, Subtractive clustering, K means clustering, Fuzzy c-means clustering**

I. Introduction

Cinematography is a way of art that everyone has enjoyed in their lives, the seventh art has a direct impact in society, culture and economics. That is one reason why Film Industry is one of the most important in the world and worth billions of dollars. But it is well known that the cost of the production of a film can be very high, and this budget affects directly the audience rating and the performance in the awards, this is why the accuracy when predicting financial performance is important when making an investment.

One of this applications in Motion Picture Industry is the prediction of what films people will want to see. Researchers from film studio 20Th Century Fox say that understanding detailed audience composition is important for movie studios that invest in stories of uncertain commercial outcome [1]. This film positioning can be based in many different criteria, based on genre, on directors value, synopses and others. Besides, the analysis of audience success based on genre can be done in many different ways, by analysing different components of a movie or any kind of motion picture. Other interesting researches in this topic are made by Y. J. Lim and Y. W. Teh in the paper "Variational bayesian approach to movie rating prediction" [3], and the work of J. D. Mcauliffe and D. M. Blei in 2008 [4].

Data science is a very useful tool to find implicit patterns that are intuitively perceived but difficult to grasp it self. So that, Artificial Intelligence has reached the enough development to be applied to creative industries like music, painting, literature and of course cinematography, which seemed impossible a few decades ago.

For this reasons, there is a big opportunity of research in this field. In this project the approach that is going to be taken is to use non supervised learning to explore the data space and find patterns within.

II. Conceptual Framework

In this research it will be used several clustering methods. The algorithms that are going to explore are: Mountain clustering, Subtractive clustering, K means clustering, Fuzzy c-means clustering and Trimmed k-means clustering. Also exploring using different similarity metrics and changing the hyper-parametres. This algorithms are explained bellow using the definitions in the curse's book, Machine Intelligence for Human Decision Making [5].

A. Mountain Clustering

Mountain clustering is a method to estimate the cluster centers from a dataset based on a density measure called the mountain function. The entire space is subdivided in clusters that actually look like a mountain. The process of finding the optimal groups starts with and assumption of gridding the entire space, and the size of the grid depends more on the designer. The intersections of the grid lines are the potential cluster centers.

The algorithm is described in the following image.

Begin:

1. An equally spaced grid on the entire data space is generated. Let V be the set of all of the points or nodes where the grid lines intersect each other.
2. Set $i = 1$. Compute the value of the mountain function m_i at each point $\mathrm{v} \in V$ as follows:

$$m_i(\mathbf{v}) = \sum_{j=1}^{n} \exp\left(-\frac{\|\mathbf{v} - \mathbf{x}_j\|^2}{2\sigma^2}\right)$$

   where σ is an application specific constant.
3. Determine the point $\mathbf{v}$ at which the function m_i reaches the highest value and designate this point as the cluster center c_i.
4. Compute the value of the new mountain function m_{i+1} at each point $\mathbf{v} \in V$ as follows:

$$m_{i+1}(\mathbf{v}) = m_i(\mathbf{v}) - m_i(\mathbf{c}_i) \exp\left(-\frac{\|\mathbf{v} - \mathbf{c}_i\|^2}{2\beta^2}\right)$$

   where β is an application specific constant.
5. Set $i = i + 1$.
6. Repeat steps 3 to 5 while $i \leq K$.

End

Fig. 1. Mountain algorithm. [5]

Where Two parameters σ and β act as the kernel influence for mountain function construction and update. In particular, application of constant σ influences the height and the smoothness of the resultant mountain function m_i.

B. Subtractive clustering

This is a exploratory algorithm similar to the mountain algorithm, except fixing the computational issues of the grid. This algorithm begins by considering each object (point) in the dataset as a potential cluster center.

The potential for each object is a function of its distance to the remaining objects in the dataset. Consequently, an object with many nearby objects (i.e. with a high density of surrounding objects) will have a high potential value.
The algorithm is described in the following image.

Begin:

1. Set $i = 1$. Calculate a density measure D_i at each object $\mathbf{x}_j$ as follows:

$$D_i(\mathbf{x}_j) = \sum_{l=1}^{n} \exp\left(-\frac{\|\mathbf{x}_j - \mathbf{x}_l\|^2}{(r_a/2)^2}\right)$$

2. Find the object $\mathbf{x}_j$ with the highest density measure D_i and designate it as the cluster center $\mathbf{c}_i$.
3. Calculate a new density measure D_{i+1} at each object $\mathbf{x}_j$ as follows:

$$D_{i+1}(\mathbf{x}_j) = D_i(\mathbf{x}_j) - D_i(\mathbf{c}_i) \exp\left(-\frac{\|\mathbf{x}_j - \mathbf{c}_i\|^2}{(r_b/2)^2}\right)$$

4. Set $i = i + 1$.
5. Repeat steps 2 to 4 while $i \leq K$.

End

Fig. 2. Subtractive algorithm. [5]

Where r_a is a positive radius used to define a neighborhood around each object in order to measure its potential value; and r_b is used to define the neighborhood of a found cluster that will experiment a subtraction (reduction) of its potential.

C. Classic K-means clustering

K-means aims to partition a set of n objects/ data points $x_1, x2, ..., x_n$ into K clusters $C_1, C_2, ..., C_K$ such that the following cost function J is minimized:

$$\mathcal{J} = \sum_{i=1}^{K} \sum_{j=1}^{n} u_{ij} \|\mathbf{x}_j - \mathbf{c}_i\|^2$$

where

$$u_{ij} = \begin{cases} 1, & \|\mathbf{x}_j - \mathbf{c}_i\|^2 \leq \|\mathbf{x}_j - \mathbf{c}_l\|^2, l \neq i \\ 0, & \text{otherwise} \end{cases}$$

And $c_i$is the center (the prototype, the most representative member, etc) of the cluster C_i. The optimal center c_i that minimizes the cost function J is the sample mean for the cluster C_i and can be determined based on the entries of membership matrix U as follows:

$$\mathbf{c}_i = \frac{1}{N_i} \sum_{j=1}^{n} u_{ij} \mathbf{x}_j$$

where $N_i = \sum_{j=1}^{n} u_{ij}$ is the size of the cluster C_i. The algorithm is described in the following image.

Begin:

1. Select randomly K objects from the dataset and designate them as cluster centers $\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_K$.
2. Determine all of the entries u_{ij} of the membership matrix U according to Equation 6.7.
3. Update the center of each cluster based on the entries of the membership matrix U using the Equation 6.8.
4. Compute the cost function J according to Equation 6.6.
5. Repeat steps 2 to 4 until cost function J converges.

End

Fig. 3. Classic k.means algorithm. [5]

D. Fuzzy C-means Clustering

Fuzzy clustering considers that each point of the available data can belong to more than one cluster with certain degree of membership. This algorithm consist on the minimization of the c-means functional J formulated by Bezdek:

$$\mathcal{J} = \sum_{i=1}^{K} \sum_{j=1}^{n} (u_{ij})^m \left\| \mathbf{x}_j - \mathbf{c}_i \right\|^2$$

FCM aims to partition a set of n data points $x_1, x_2, ..., x_n$ into C fuzzy clusters such that the cost function J in the equation above is minimized.

The cost function J can be minimized with an iterative procedure that updates the following equations for the centers.

$$\mathbf{c}_i = \frac{\sum_{j=1}^{n} (u_{ij})^m \mathbf{x}_j}{\sum_{j=1}^{n} (u_{ij})^m},$$

and the membership values,

$$u_{ij} = \left[\sum_{i=1}^{C} \left(\frac{\left\| \mathbf{c}_i - \mathbf{x}_j \right\|}{\left\| \mathbf{c}_l - \mathbf{x}_j \right\|} \right)^{2/(m-1)} \right]^{-1}.$$

The u_{ij} are the entries of the fuzzy membership or partition matrix $\mathcal{U}$, and each of them describes the degree of membership of the data point $\mathbf{x}_j$ in the i-the fuzzy cluster with a value (between 0 and 1) that is inversely proportional to the distance of these data points to the cluster center $\mathbf{c}_i$. The degree of membership of a data point to every fuzzy cluster is fixed, so, the sum of elements in each column of the matrix $\mathcal{U}$ is equal to 1.

To implement this algorithm, we have as inputs n objects $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n$; the number of clusters C; and the fuzzification parameter m. Expecting as outputs, we have the Cluster centers $\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_C$; and the membership matrix U.

E. Trimmed k-means clustering

Trimmed k-means method is defined through the search of k centers $m_1, ..., m_k \subset R^p$ solving the double minimization problem

$$\arg\min_{\mathbf{Y}} \min_{m_1,\ldots,m_k} \sum_{x_i \in \mathbf{Y}} \min_{j=1,\ldots,k} \left\| x_i - m_j \right\|^2,$$

where $\mathbf{Y}$ ranges on the class of subsets of size $[n(1-\alpha)]$ within the sample $\{x_1, \ldots, x_n\}$. Notice that this definition includes the ordinary k-means as a limit case when $\alpha = 0$. When using trimmed k-means, we are not forced to classify all the observations because we allow for a proportion α of observations (hopefully the most outlying ones) to be left unassigned. [6].

The algorithm is the following:

1) Random starts: Draw k random initial centers $m_1^0, \ldots, m_k^0$.
2) Concentration steps:
   a) Keep the set H made of the $[n(1-\alpha)]$ observations closest to the centers $m_1^l, \ldots, m_k^l$.
   b) Partition H onto k subsets $\{H_1, \ldots, H_k\}$, where H_j contains the observations in H closer to the center m_j^l than to the other centers.
   c) Update the centers $m_1^{l+1}, \ldots, m_k^{l+1}$ such that each center m_j^{l+1} is the sample mean of the observations in H_j.
3) Repeat several times Step 1 and Step 2 and keep the best solution in the sense of minimizing the objective function.

III. Method

The data used for this research is from the combination of two Kaggle data sets [2]. The resulting data set contains the following features.

- Title of the movie.
- Rating, a categorical variable indicating the Motion Picture Association film rating.
- Genres, a categorical variable indicating the main genre of the movie.
- Duration of the movie in minutes.
- Number of years since the film premiere.
- Votes in IMDb.
- Score in IMDb.
- Country, 1 if the country is The United States, 0 if not.
- Budget of the film.

A. Visualization

In Figure 4, there is a scatter plot for each pair of features. It can be seen that there is not any lineal relation between this variables.

Fig. 4. Scatter plot of each pair of features

In Figure 5, there is a scatter plot for each pair of 3 features. The color respresents an extra variable, added just for this plot, and it indicates whether a film was nominated for an Oscar or not.

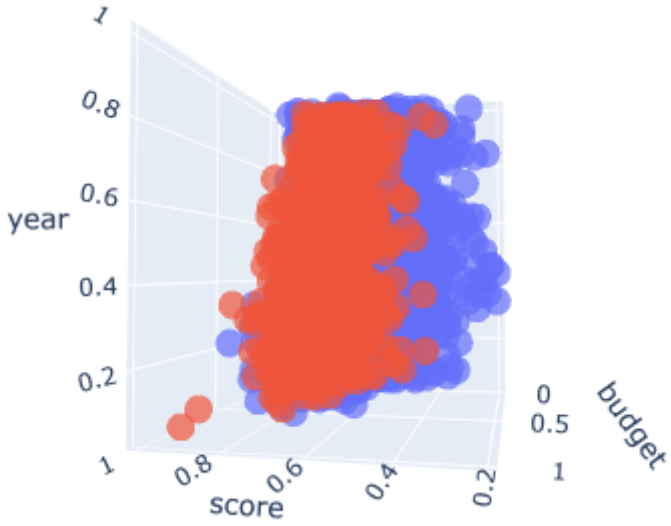

Fig. 5. Scatter plot 3 variables, where the blue color represents the non nominated movies and red represents the nominated ones.

B. Learning considerations

As mentioned above, all the methods will explore the space using different hyper-parameters and metrics. The metrics that are going to be used are: euclidean distance, mahalanobis distance and cosine similarity.

IV. Results

In the following tables and figures, it can be seen the performance of the methods used to explore the dataset for each algorithm and hyper-parameters.

A. Mountain clustering

The computational expense is the main drawback of the mountain algorithm. It increases exponentially along with the dimension of the problem since the method must evaluate the mountain function over all grid points. This is the reason why, for this problem and this data set, the mountain algorithm, despite being simple and intuitive, is not the best option. Also, since the computational resources are not enough, in this case I will focus on the subtracting algorithm.

B. Subtractive clustering

In the following figures, are the results in the exploration of the subtractive algorithm. Since this is a exploratory algorithm, the number of clusters is not defined by the programmer. And for all the variations of the parameters in the subtractive algorithm it only found one cluster. The plots shown in this section are the scatter plot of two variables.

The figures 7 and 8 are the results of using the Euclidean distance and the Mahalanobis distance respectively. With $r_a = 0.5$ and $r_b = 0.8$

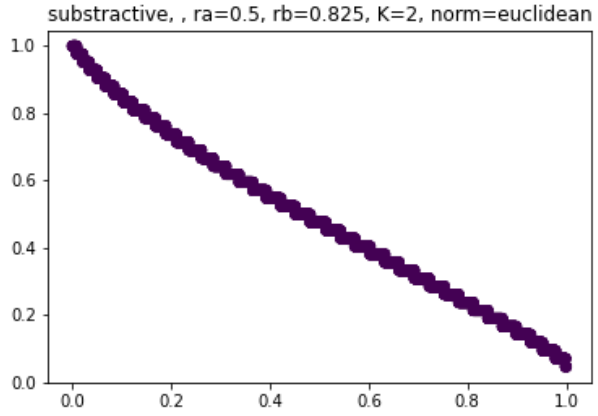

Fig. 6. Subtractive algorithm results

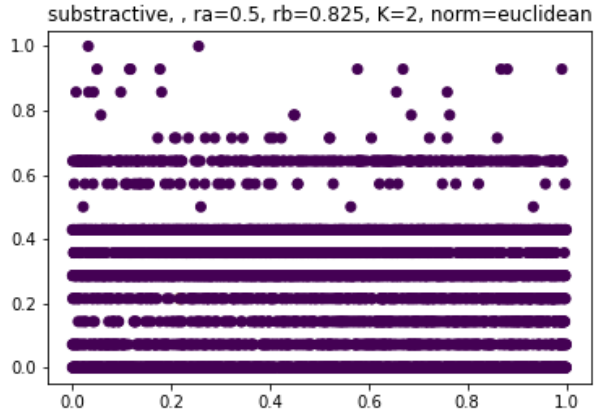

Fig. 7. Subtractive algorithm results

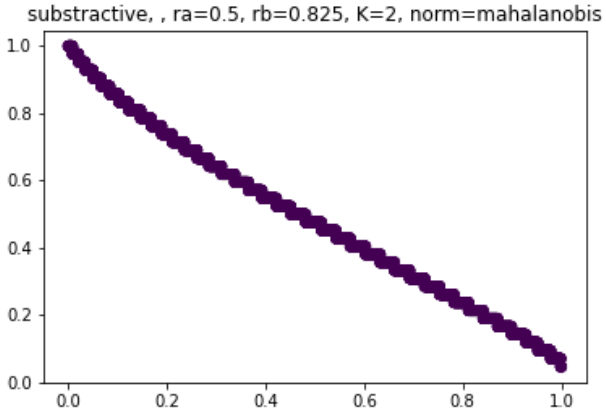

Fig. 8. Subtractive algorithm results

For the experiments, the value of the parameters changed, so $r_a = 0.8$ and $r_b = 1.32$. The figures 9 and 10 are the results of using the Euclidean distance and the Mahalanobis distance respectively.

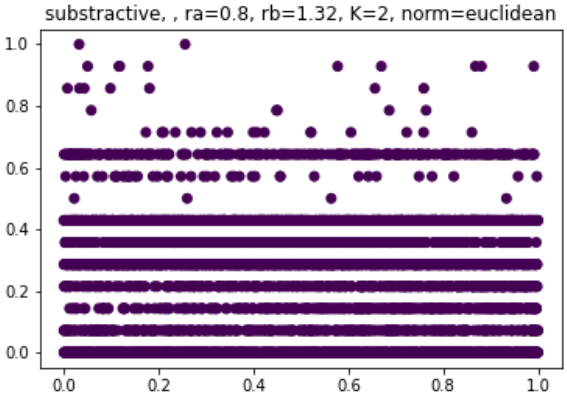

Fig. 9. Subtractive algorithm results

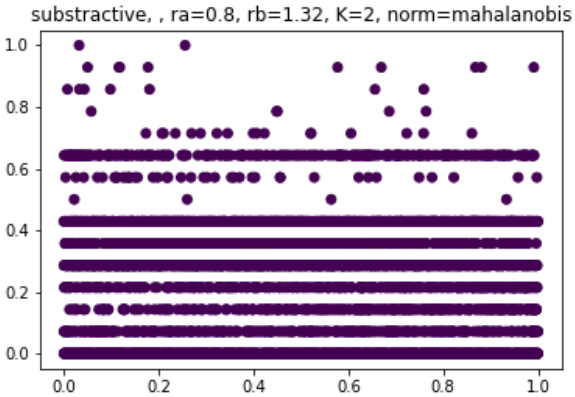

Fig. 10. Subtractive algorithm results

As all the metrics and parameters variations gave the same result, it is not necessary to show them.

C. Classic k-means

In the following figures, are the results in the exploration of the k-means algorithm. In this algorithm the number of clusters is defined by the programmer.

The figures 11 and 11 are the results of using the Euclidean distance, but the plots are of two different pairs of variables. The number of clusters in this case was 2, it can be seen that the algorithm splits the space in half.

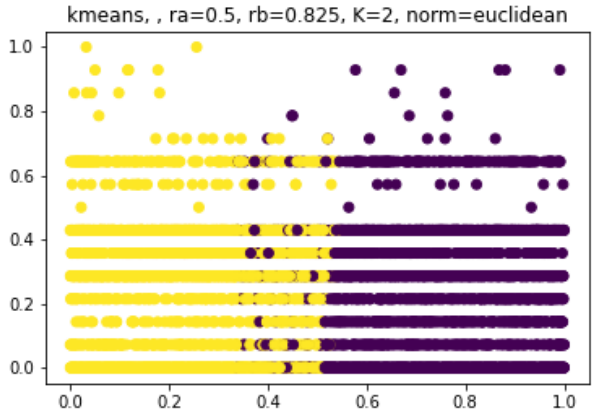

Fig. 11. K-means algorithm results

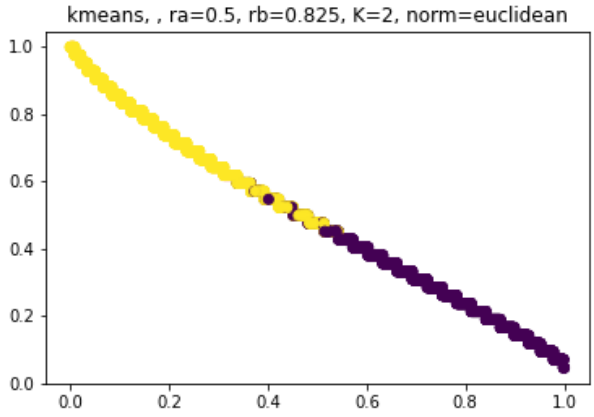

Fig. 12. K-means algorithm results

The figures 13 and 14 are the results of using also the Euclidean distance, plotting two different pairs of variables, but the number of clusters in this case was 3. It can be seen that, even given 3 clusters the algorithm also splits the space in half.

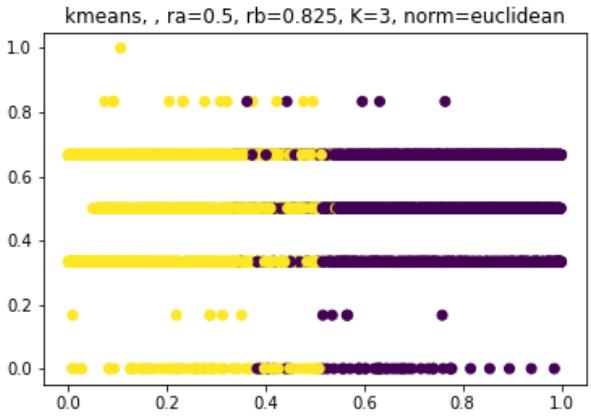

Fig. 13. K-means algorithm results

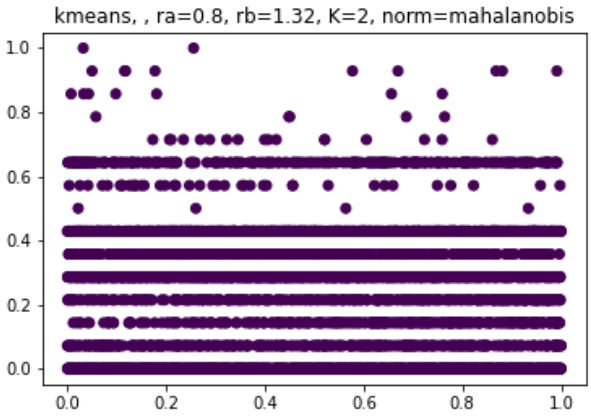

Fig. 15. K-means algorithm results

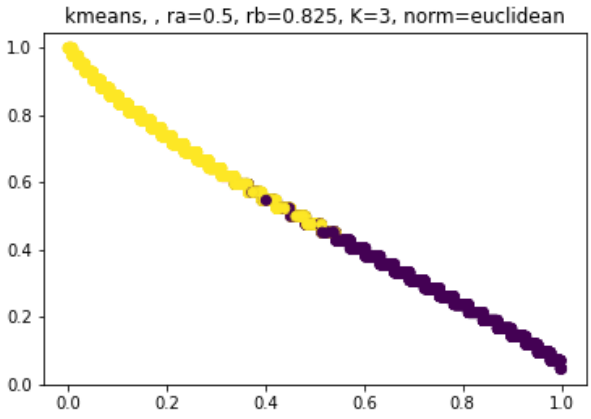

Fig. 14. K-means algorithm results

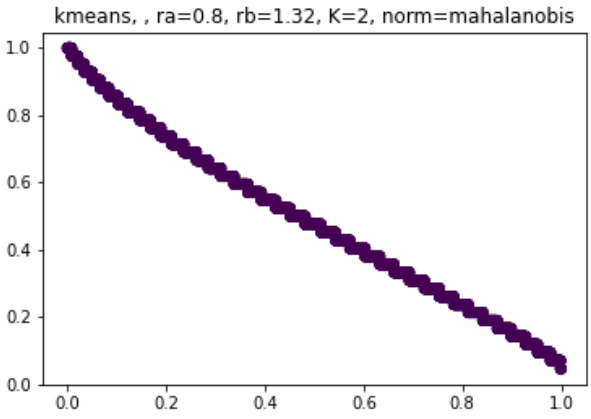

Fig. 16. K-means algorithm results

The figures 15 and 16 are the results of using Mahalanobis distance, plotting two different pairs of variables and the number of clusters in this case was 2. It can be seen that, even given 2 clusters the algorithm does not split at all the space, all the elements belong in the same cluster.

The figures 17 and 18 are the results of using Cosine distance, plotting two different pairs of variables and the number of clusters in this case was 2. It can be seen that, even given 2 clusters the algorithm does not split at all the space, all the elements belong in the same cluster.

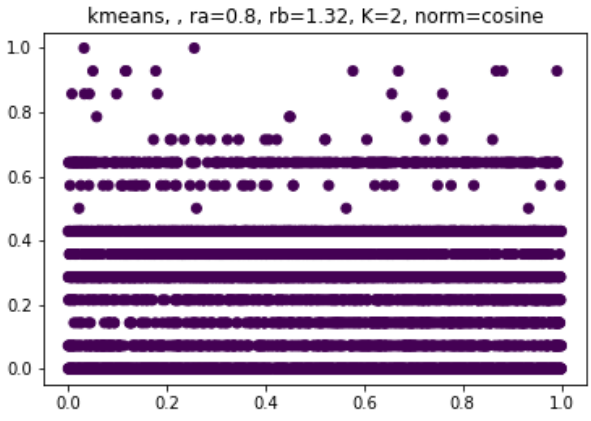

Fig. 17. K-means algorithm results

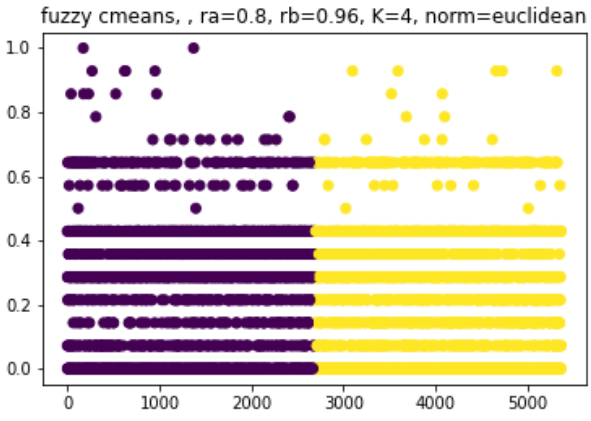

Fig. 19. Fuzzy c-means algorithm results

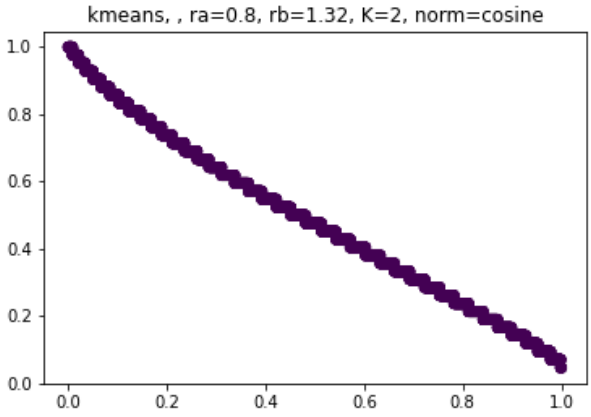

Fig. 18. K-means algorithm results

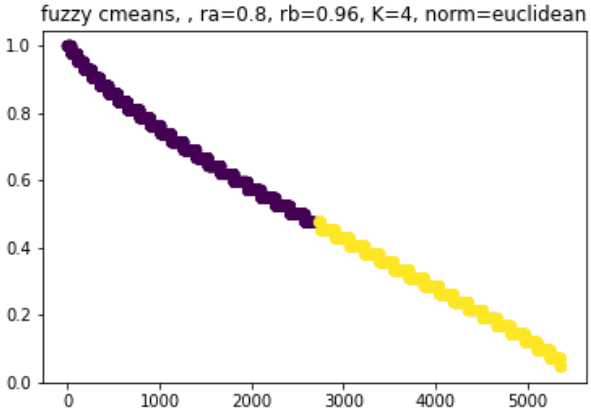

Fig. 20. Fuzzy c-means algorithm results

D. Fuzzy c-means

In the following figures, are the results in the exploration of the fuzzy c-means algorithm. In this algorithm the number of clusters is defined by the programmer, the value of the fuzzy parameter is 0.5.

The figures 19 and 20 are the results of using the Euclidean distance, but the plots are of two different pairs of variables. The number of clusters in this case was 2, it can be seen that the algorithm splits the space in half.

The figures 21 and 22 are the results of using Cosine distance, plotting two different pairs of variables and the number of clusters in this case was 2. It can be seen that, unlike the k-means algorithm where all the elements belong in the same cluster, in this case it also splits the space in half.

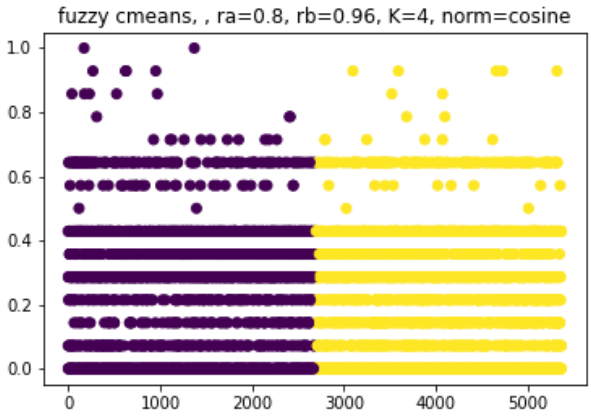

Fig. 21. Fuzzy c-means algorithm results

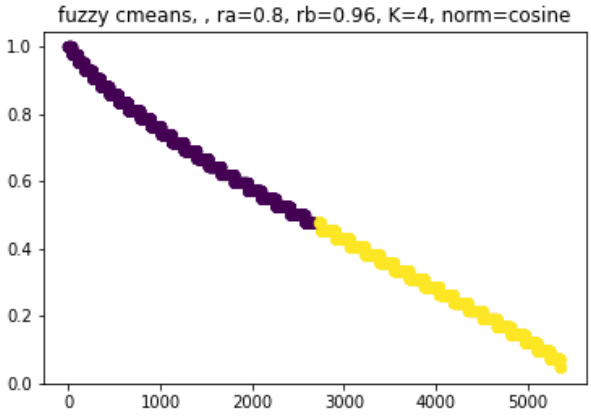

Fig. 22. Fuzzy c-means algorithm results

The results of this algorithm was very similar to the ones of k-means.

E. Trimmed k-means

In the following figures, are the results in the exploration of the trimmed k-means algorithm. In this algorithm the number of clusters is defined by the programmer, the value of the trim parameter is 0.1.

The figures 23 and 24 are the results of using the Euclidean distance, but the plots are of two different pairs of variables. The number of clusters in this case was 2, it can be seen that the algorithm splits the space in half, but with more accuracy than k-means.

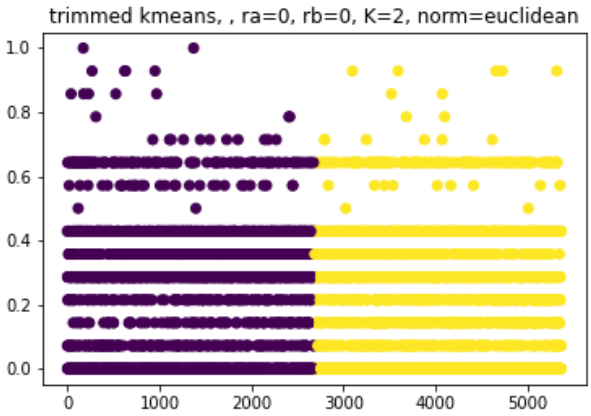

Fig. 23. Trimmed K-means algorithm results

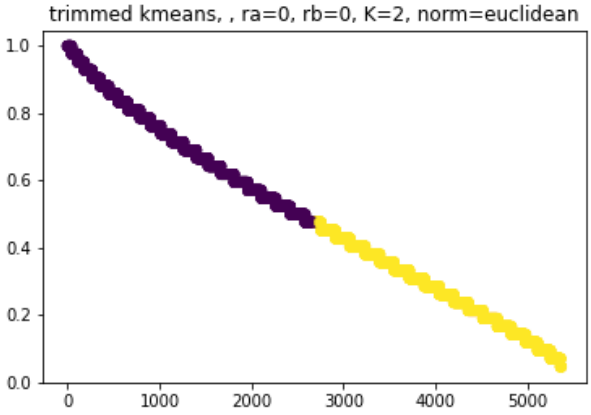

Fig. 24. Trimmed K-means algorithm results

The figures 25 and 26 are the results of using the Euclidean distance, but the plots are of two different pairs of variables. The number of clusters in this case was 3. It can be seen that the algorithm splits the space in 3. This results are the same for the other two metrics.

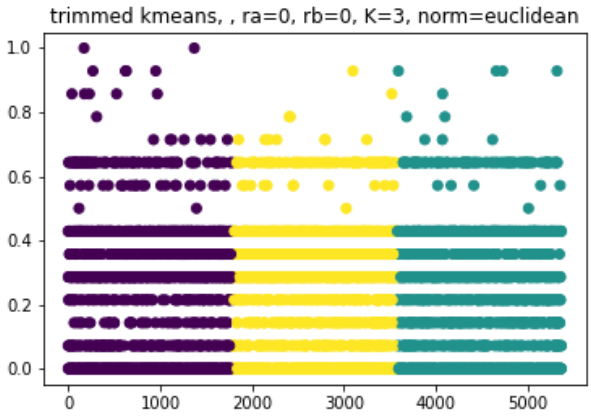

Fig. 25. Trimmed K-means algorithm results

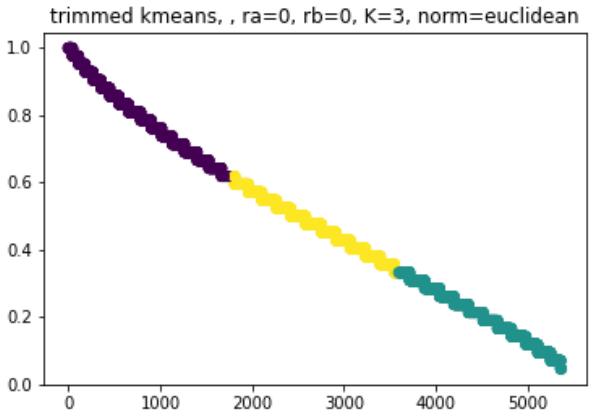

Fig. 26. Trimmed K-means algorithm results

The figures 25 and 26 are the results of using the Mahalanobis distance, but the plots are of two different pairs of variables. The number of clusters in this case was 2. It can be seen that the algorithm splits the space in 2. This results are the same for the cosine distance.

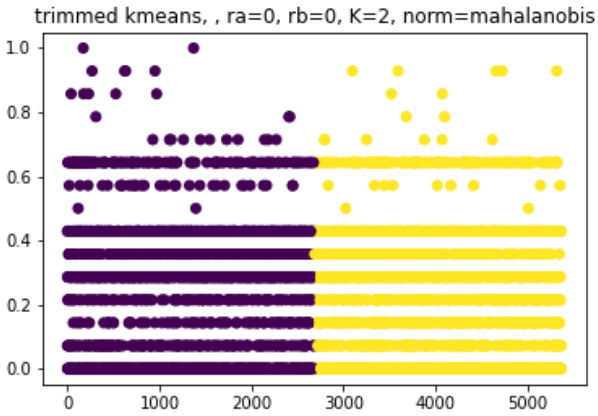

Fig. 27. Trimmed K-means algorithm results

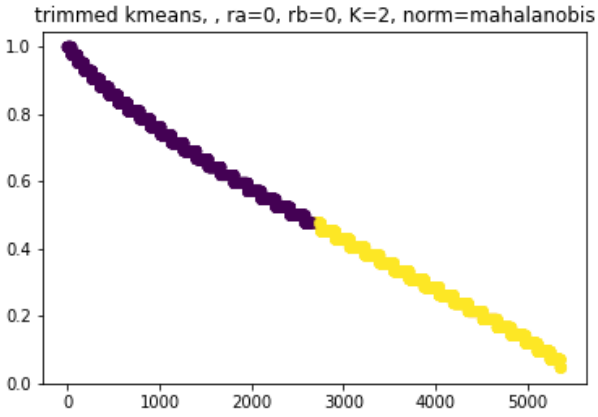

Fig. 28. Trimmed K-means algorithm results

V. Conclusions

In this work we can remark that the exploratory phase of the clustering process is very important. In this case we have seen, that the subtractive algorithm have not identified any kind of cluster in the data. That is the reason why with the other algorithms have not found any groups. And when a number of clusters is imposed on them it clearly splits the space in the number of clusters.

Another important finding is that it is important to choose the correct hyper-parameters of the methods, on that depends the learning process. This methods are sensitive to the rate in the algorithms, so that, if we do not choose suiting rates for the problem, the algorithm does not converge.

Finally, I have observed that the complexity of the algorithms has to be taken in count. Calculating the distances consumes a lot of resourses. And the algorithms that take the most time to finish are the Fuzzy k-means and the subtractive.

REFERENCES

[1] Hsieh, Cheng-Kang, Campo, Miguel, Taliyan, Abhinav, Nickens, Matt, Pandya, Mitkumar, JJ, Espinoza. 2018. Convolutional Collaborative Filter Network for Video Based Recommendation Systems. ArXiv.

[2] IMDb. 2021. Sort by Popularity - Most Popular Movies and TV Shows tagged by keywords.

[3] Y. J. Lim and Y. W. Teh, "Variational bayesian approach to movie rating prediction," in Proceedings of KDD Cup and Workshop, vol. 7, 2007, pp. 15–21.

[4] J. D. Mcauliffe and D. M. Blei, "Supervised topic models," in Advances in Neural Information Processing Systems, 2008, pp. 121–128.

[5] O. L. Quintero and J. Hopcrof, "Machine Intelligence for Human Decision Making,", 2019.

[6] Luis Angel García-Escudero and Alfonso Gordaliza and Carlos Matrán and Agustín Mayo-Isca, "A review of robust clustering method," in Advances in Data Analysis and Classification, 2010.