Credit card fraud detection using a clustering approach

María Camila Vásquez Correa
Mathematics department
Universidad EAFIT
Medellín, Colombia
mvasqu49@eafit.edu.co

***Abstract*—In this article is presented a comparison of 5 clustering algorithms to segregate data on credit card fraud detection, these algorithms are: k-means, an extension to k-means, fuzzy c-means, subtractive and mountain clustering. These algorithms are first evaluated in the iris dataset, and finally in our case study. Several metrics for finding the clusters are compared and a variation of parameters is performed.**

***Index Terms*—Cluster, k-means, credit card, fraud.**

I. Introduction

Given the current global economic context, increasing efforts are being made to both prevent and detect fraud. Credit card financial products can derive in unsecured and unplanned credit card risks and should not be underestimated. Stopping credit card fraud has become a hot issue in academia and industry. In the field of credit card fraud, the mistake of letting go of fraudulent transactions is much more expensive than mistakenly intercepting normal transactions, and the number of fraudulent transactions is far less than the normal number of transactions.
Clustering, as unsupervised data mining technique, deals with the problem of dividing a given set of entities into meaningful subsets. Clusters resulted from this data segmentation are required to be to be homogeneous and/or well separated, entities within the same group being similar while entities within different groups being dissimilar.

II. Literature review

A survey for the methods on fraud detection via clustering is presented in [1]. This survey takes into account the algorithm used and the type of fraud the models are attacking. Is clear that the prevalent method is the k-means, however, some other types of clustering including graph based, hierarchical and density based algorithms were also implemented in the reviewed works.
As an example, some of the methods that make good use of the k-means clustering are: [2], who compares the performance of Support Vector Machines along with clustering algorithms and other classification approaches, and finds put that the former is better at handling highly unbalanced data. [3] uses a Naive Bayes clustering approach (as well as [4]) along with k-means clustering. In these algorithms, there is a clear pattern of hybrid methodologies, some other examples include [5], [6], and [7] that use Hidden Markov models along with k-means clustering with the objective of reducing the false alarms produced by other algorithms of fraud detection and to include information about past transactions to build a costumer profile. However, some authors like [4] use only Naive Bayes clustering along with a Multi Layer perceptron to build a clustering engine and [8] makes use of fuzzy c-means clustering and builds stochastic models along with artificial neural networks to achieve the same objectives. Finally, hierarchical clustering paired with classification techniques is also useful, as shown in [9].
In this work we aim to use only clustering techniques to analyze a dataset containing some fraudulent transactions and the features associated with them. These clustering techniques will be analyzed in order to get some insights about the data and the costumers.

III. Methodology

A. Data

1) *Toy dataset:* The initial test dataset for the algorithms was the iris dataset, that contains 3 classes of iris flowers (versicolor, virginica and setosa) and has 4 features (sepal length and height and petal length and width) with 150 samples.
2) *Credit card dataset:* This dataset was downloaded from Kaggle (https://www.kaggle.com/mlg-ulb/creditcardfraud) and contains transactions made by credit cards in September 2013 by european cardholders. This dataset presents transactions that occurred in two days, where we have 492 frauds out of 284,807 transactions. The dataset is highly unbalanced, the positive class (frauds) account for 0.172% of all transactions.

B. Pipeline

1) *Preprocessing:* Elimination of missing values and normalization. The normalization method performed is dividing by the maximum of each feature. This to ensure that all the data points are mapped into the hypercube [0,1].
2) *Statistical analysis:* In order to asses the complexity of the problem, several statistical tests were taken, including normality tests, Independence tests, dsitribution tests and stationarity tests.

3) *Feature selection and extraction:* This step was performed differently in both datasets, the toy dataset and the credit card dataset. In the first one, 4 characteristics were extracted, augmenting the dimension from 4 to 8. In the second one, 28 characteristics were provided, and a reduction of dimensionality via PCA was performed.
4) *Embbeding:* The T-distributed Stochastic Neighbor Embedding was used as embbeding algorithm, due to its distance conserving property. This embbeding was used to visualize the results and also to learn in lower dimensions.
5) *Learning:* 5 clustering algorithms were implemented in both datasets, performing the learning task in the higher, medium and lower dimensional space and comparing the results.

C. Algorithms

Let $\{x_1, \ldots, x_n\}$ be n data points in an M dimensional space, normalized to an hypercube. Also, let $d(x, y)$ be a distance function in $\mathbb{R}^n$.

1) *Subtractive clustering:* This method takes all the points as cluster centers candidates. The steps for the algorithm are:
   a) Calculate a density function for each pint x_i, given by:
   $$D_i = \sum_{i=1}^{n} \exp\left(-\frac{d(x_i, x_j)^2}{(r_a/2)^2}\right)$$
   where r_a is a positive constant representing a neighborhood radius.
   b) Choose the cluster center x_{ci} as the point having the largest density value D_{c_i}.
   c) Revise each point's density function by eliminating the effect of the previously chosen center as follows:
   $$D_i = D_i - D_{c_i} \exp\left(-\frac{d(x_i, x_{c_i})^2}{(r_b/2)^2}\right)$$
   where r_b is a positive constant which defines a neighborhood that has measurable reductions in density measure.
   d) If a sufficient number of clusters is reached, or the value of the density function is too small, stop. If not, go back to step 2.
2) *Mountain clustering:* This method, instead of taking each point as a cluster candidate, it forms a grid in the data space, where the intersections of the grid lines represent the potential cluster centers. These points are stored in set V. Then, we apply the following algorithm:
   a) Calculate a mountain function for each point v_j in V, given by:
   $$m_j = \sum_{i=1}^{n} \exp\left(-\frac{d(v_j - x_i)^2}{2\sigma^2}\right)$$
   where σ is a positive constant representing the height and the smoothness of the mountain.
   b) Choose the cluster center v_{ci} as the point having the largest mountain value m_{c_i}.
   c) Revise each point's mountain function by eliminating the effect of the previously chosen center as follows:
   $$m_j = m_j - m_{c_i} \exp\left(-\frac{d(v_j - v_{c_i})^2}{(2\beta^2}\right)$$
   where β is a positive constant.
   d) If a sufficient number of clusters is reached, or the value of the mountain function is too small, stop. If not, go back to step 2.
3) *k-means clustering* In this algorithm, the dataset is going to be partitioned in k groups G_i, $i = 1, \ldots, k$. The cost function, based on a distance $d(x, y)$, can be defined by:
   $$J = \sum_{i=1}^{c} J_i = \sum_{i=1}^{c} \left(\sum_{k, x_k \in G_i} d(x_k, c_i)^2 \right)$$
   For the development, we follow these steps:
   a) Randomly initialize the cluster centers c_i, $i = 1, \ldots, k$.
   b) Determine a membership matrix for each group, U by:
   $$u_{ij} = \begin{cases} 1 & ||x_j - c_i||^2 \leq ||x_j - c_k||^2 \qquad \forall k \neq i \\ 0 & \text{Otherwise} \end{cases}$$
   c) Compute the cost function. Stop if it is bellow a certain tolerance ε or its improvement is irrelevant.
   d) Update the cluster centers by:
   $$c_i = \frac{1}{|G_i|} \sum_{k, x_k \in G_i} x_k$$
   Then, go to the second step.
4) *Fuzzy c-means clustering* In this algorithm, each point of the dataset belongs to each of the c clusters in a certain degree of membership, between 0 and 1. The steps to follow are:
   a) Initialize the membership matrix U with random values between 0 and 1 such that:
   $$\sum_{i=1}^{c} u_{ij} = 1 \qquad \forall j = 1, \ldots, n$$
   b) Calculate c fuzzy cluster centers c_i, $i = 1, \ldots, n$ by:
   $$c_i = \frac{\sum_{j=1}^{n} u_{ij}^m x_j}{\sum_{j=1}^{n} u_{ij}^m}$$
   c) Compute the cost function:
   $$J(U, c_1, \ldots, c_c) = \sum_{i=1}^{c} J_i = \sum_{i=1}^{c} \sum_{j=1}^{n} u_{ij}^m d_{ij}^2$$
   where u_{ij} is between 0 and 1, c_i is the cluster center of fuzzy group i, $d_{ij} = d(c_i, x_j)$ and $m \in [1, \infty)$ is a weighting exponent. Stop if it

is either bellow a certain tolerance ε or if there is no improvement.

d) Compute a new membership matrix using the following:

$$u_{ij} = \frac{1}{\sum_{k=1}^{c}\left(\frac{d_{ij}}{d_{kj}}\right)^{\frac{2}{m-1}}}$$

Then, go to step 2.

5) *K-medians clustering* This is a variation of the k-means clustering [10] in which, instead of computing the mean of the points in a cluster, one calculates the median. This algorithm is more robust to outliers in the dataset.

D. Distance functions

As defined above, all of the mentioned algorithms calculate distances between points. Let $x = (x_1, \ldots, x_n)$ and $y = (y_1, \ldots, y_n)$ be two points in ^{n}. The distance functions that are going to be used are the following:

- *Euclidean distance*

$$d(x, y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

- *Manhattan distance*

$$d(x, y) = \sum_{i=1}^{n}|x_i - y_i|$$

- *Cosine similarity*

$$d(x, y) = \frac{\sum_{i=1}^{n} x_i y_i}{\sum_{i=1}^{n} x_i^2 \sum_{i=1}^{n} y_i^2}$$

E. Validation

Both datasets contain groundtruths for the classes of each sample. The previously presented algorithms will be evaluated using some indices (intra cluster and extra cluster) with both of the datasets. Let c_i, $i = 1, \ldots, k$ be the cluster centers and C_i be the sets containing the points of each cluster. The indices used are described in [11] and include:

- *Davies-Bouldin (DB) index* This index us a function of the *within cluster scatter* and the *between cluster separation*. We define:
  1) Within cluster scatter: $S_i = \frac{1}{|C_i|}\sum_{x \in C_i} ||x - z_i||$.
  2) Between cluster separation: Is simply the distance d_{ij} between two cluster centers $||c_i - c_j||$.
  3) $R_{i,qt} = \max_{j, j \neq i}\left\{\frac{S_{i,q} + S_{j+q}}{d_{ij}t}\right\}$

  And the index is defined as:

$$DB = \frac{1}{k}\sum_{i=1}^{k} R_{i,qt}$$

  The objective is to minimize this index for proper clustering.

- *Calinski Harabasz (CH) Index* Let c be the centroid of the entire dataset. The index is defined as

$$CH = \frac{\left[\frac{\sum_{l=1}^{k}|C_l|||c_l - c||^2}{k-1}\right]}{\left[\frac{\sum_{l=1}^{k}\sum_{i=1}^{|C_l|}||x_i - c_l||^2}{n-k}\right]}$$

IV. Results

A. Iris dataset

- **Subtractive clustering**

The iris dataset contains a ground truth of 3 clusters. In Tables I, II and III are described the number of clusters found by the subtractive algorithm with various configurations of neighborhood ratios r_a and metrics. The Figures 1 and 2 show the initial state of the density function and the final state for one configuration ($r_a = 0.5$, metric euclidean),respectively.

r_a	euclidean	cosine	cityblock
0.4	4	1	6
0.5	3	1	4
0.7	2	1	3

TABLE I
NUMBER OF CLUSTERS GIVEN BY THE SUBTRACTIVE CLUSTER FOR THE IRIS DATASET.

r_a	euclidean	cosine	cityblock
0.4	99	4	99
0.5	93	4	99
0.7	8	3	94

TABLE II
NUMBER OF CLUSTERS GIVEN BY THE SUBTRACTIVE CLUSTER FOR THE AUGMENTED IRIS DATASET.

r_a	euclidean	cosine	cityblock
0.4	99	3	99
0.5	91	2	99
0.7	37	2	48

TABLE III
NUMBER OF CLUSTERS GIVEN BY THE SUBTRACTIVE CLUSTER FOR THE EMBBEDED IRIS DATASET.

Despite the fact that the dataset provides clearly at least 2 groups, the results vary among the different parameters for the subtractive algorithm. We observe that, for the embbeded data, the cosine norm is usually better than the others, as well as for the augmented dataset. But the euclidean norm works better for the original dataset, as shown in Figure 3. In Tables IV and V we observe two indices for the clustering performed in the original dataset. Note that, for the cosine norm the number of clusters achieved is 1, and the metric is not defined for less than 2 groups. We can observe that the highest score

r_a	euclidean	cosine	cityblock
0.4	523.017618	NaN	415.515675
0.5	603.592099	NaN	522.011796
0.7	497.999891	NaN	594.486176

TABLE IV
CALINSKI-HARABASZ SCORE FOR THE IRIS DATASET CLUSTERED WITH THE SUBTRACTIVE ALGORITHM.

r_a	euclidean	cosine	cityblock
0.4	0.800206	NaN	1.054853
0.5	0.634911	NaN	0.817205
0.7	0.390944	NaN	0.640144

TABLE V
DAVIES-BOULDIN INDEX FOR THE IRIS DATASET CLUSTERED WITH THE SUBTRACTIVE ALGORITHM.

and the lowest index are achieved with the euclidean norm, as we stated in the results for the number of clusters. In Tables VI and VII are shown the results for the validation on the augmented dataset. As seen before, the cosine norm achieves an appropiate number of clusters for this dataset, however, its indices and scores are the worst performed, probably because the other metrics achieve clusters that, even when they are not much informative, separate well the data and overfits the score.

r_a	euclidean	cosine	cityblock
0.4	76625.576579	3.018177	78208.081628
0.5	53652.621959	3.874866	74559.857679
0.7	2.602149	4.132075	55651.428403

TABLE VI
CALINSKI-HARABASZ SCORE FOR THE AUGMENTED IRIS DATASET CLUSTERED WITH THE SUBTRACTIVE ALGORITHM.

r_a	euclidean	cosine	cityblock
0.4	0.217657	1.903561	0.219811
0.5	0.221893	1.861367	0.228289
0.7	2.739684	1.963818	0.258687

TABLE VII
DAVIES-BOULDIN INDEX FOR THE AUGMENTED IRIS DATASET CLUSTERED WITH THE SUBTRACTIVE ALGORITHM.

r_a	euclidean	cosine	cityblock
0.4	9150.473726	2055.146690	8762.051647
0.5	6450.492142	1626.713464	8176.198965
0.7	3367.100226	1626.713464	3643.894411

TABLE VIII
CALINSKI-HARABASZ SCORE FOR THE EMBBEDED IRIS DATASET CLUSTERED WITH THE SUBTRACTIVE ALGORITHM.

r_a	euclidean	cosine	cityblock
0.4	0.270312	0.423202	0.279279
0.5	0.327560	0.216722	0.290310
0.7	0.626248	0.216722	0.564174

TABLE IX
DAVIES-BOULDIN INDEX FOR THE EMBBEDED IRIS DATASET CLUSTERED WITH THE SUBTRACTIVE ALGORITHM.

Finally, in Tables IX and VIII we observe the valdiation results for the embbeded dataset (that contained only 2 features). Again, as for the augmented dataset, the euclidean and cityblock metrics are performing better, but we observe that is probably because the number of clusters achieved is huge.

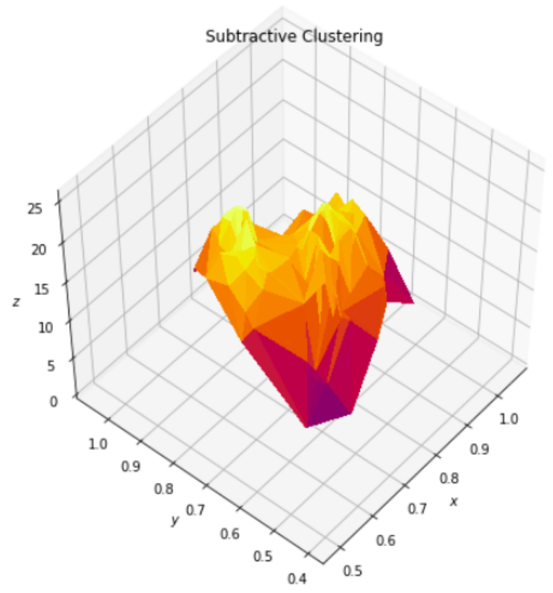

Fig. 1. Initial density function for the subtractive clustering using the euclidean norm.

- **Mountain clustering:** In Tables X, XI and XII are described the number of clusters found by the mountain algorithm with various configurations of smoothing parameters σ and metrics. For $\sigma = 1.5$ and the euclidean metric, the first and last mountain function is shown in Figures 4 and 2, respectively.

σ	euclidean	cosine	cityblock
0.4	5	3	7
0.5	3	2	4
0.7	2	1	3

TABLE X
NUMBER OF CLUSTERS FOUND BY THE MOUNTAIN ALGORITHM IN THE IRIS DATASET.

In comparison with the subtractive algorithm, the mountain algorithm is able to approximate better the number of clusters existing in the iris dataset, for any configuration

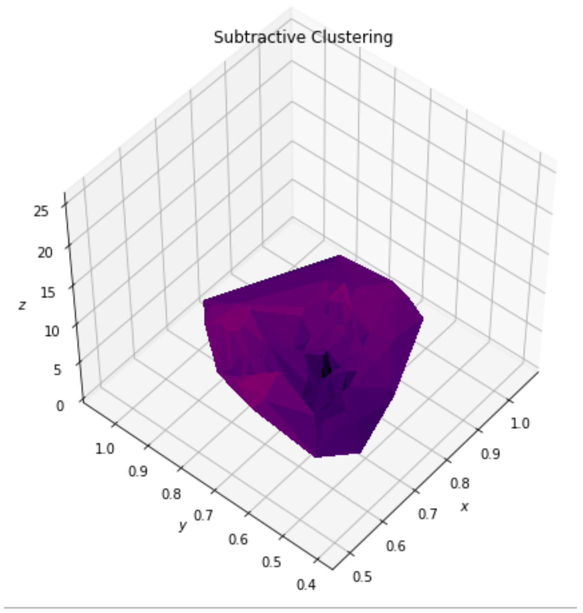

Fig. 2. Final density function for the subtractive clustering using the euclidean norm.

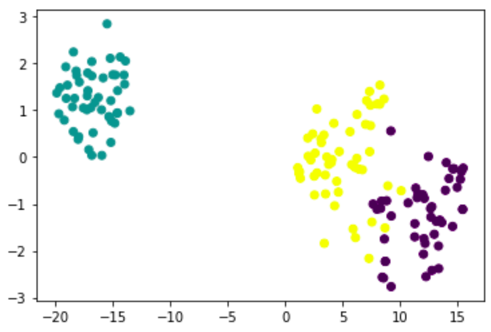

Fig. 3. Clustering for the iris dataset using the subtractive algorithm.

σ	euclidean	cosine	cityblock
0.4	4	4	2
0.5	3	4	2
0.7	2	2	2

TABLE XI
NUMBER OF CLUSTERS FOUND BY THE MOUNTAIN ALGORITHM IN THE AUGMENTED IRIS DATASET.

σ	euclidean	cosine	cityblock
0.4	2	3	2
0.5	2	3	2
0.7	3	2	3

TABLE XII
NUMBER OF CLUSTERS FOUND BY THE MOUNTAIN ALGORITHM IN THE EMBBEDED IRIS DATASET.

of the smoothing parameter σ and metric. However, as shown in the results of Figure 6, some configurations are better than others in finding the correct amount of clusters.

σ	euclidean	cosine	cityblock
0.4	392.333739	NaN	245.874313
0.5	586.577476	NaN	419.084383
0.7	497.999891	NaN	567.385797

TABLE XIII
CALINSKI-HARABASZ SCORE FOR THE IRIS DATASET CLUSTERD BY THE MOUNTAIN ALGORITHM.

σ	euclidean	cosine	cityblock
0.4	0.803257	NaN	1.185689
0.5	0.641318	NaN	0.678465
0.7	0.390944	NaN	0.662073

TABLE XIV
DAVIS BOULDIN INDEX FOR THE IRIS DATASET CLUSTERD BY THE MOUNTAIN ALGORITHM.

In Tables XIII and XIV are shown the validity indices for the original iris dataset. We can see that, despite the fact that the mountain with the cosine metric finds more than 1 cluster, the centroids of some of them are so far away fro the data that one ends up having only one cluster, and that is why the indices are not defined in that point. We observe that, in this case, the euclidean metric gives the best overall performance.

σ	euclidean	cosine	cityblock
0.4	0.807805	4.606519	0.299737
0.5	1.235066	1.092758	0.299737
0.7	0.860060	1.378751	0.299737

TABLE XV
CALINSKI-HARABASZ SCORE FOR THE AUGMENTED IRIS DATASET CLUSTERED BY THE MOUNTAIN ALGORITHM.

σ	euclidean	cosine	cityblock
0.4	4.605520	2.506750	7.163532
0.5	10.291972	2.595677	7.163532
0.7	4.561892	2.061052	7.163532

TABLE XVI
DAVIS BOULDIN INDEX FOR THE AUGMENTED IRIS DATASET CLUSTERED BY THE MOUNTAIN ALGORITHM.

In Tables XV and XVI are shown the results for the validation of the mountain algorithm in the augmented dataset. In this case, we have always more than 1 cluster, which is a good result. Here, the cityblock (Manhattan) distance shows the best performance overall.

Finally, in Tables XVII and XVIII are shown the validity indices for the embbeded dataset. Again, we have more

σ	euclidean	cosine	cityblock
0.4	61.900352	1400.846404	62.435134
0.5	61.900352	1400.846404	62.435134
0.7	106.215548	1626.713464	104.916325

TABLE XVII
CALINSKI-HARABASZ SCORE FOR THE EMBBEDED IRIS DATASET CLUSTERED BY THE MOUNTAIN ALGORITHM.

σ	euclidean	cosine	cityblock
0.4	0.820981	0.522071	0.833421
0.5	0.820981	0.522071	0.833421
0.7	0.685696	0.216722	0.686128

TABLE XVIII
DAVIS BOULDIN INDEX FOR THE EMBBEDED IRIS DATASET CLUSTERED BY THE MOUNTAIN ALGORITHM.

than 1 cluster and the manhattan and euclidean distance show a similar performance overall, being both better than the cosine distance.

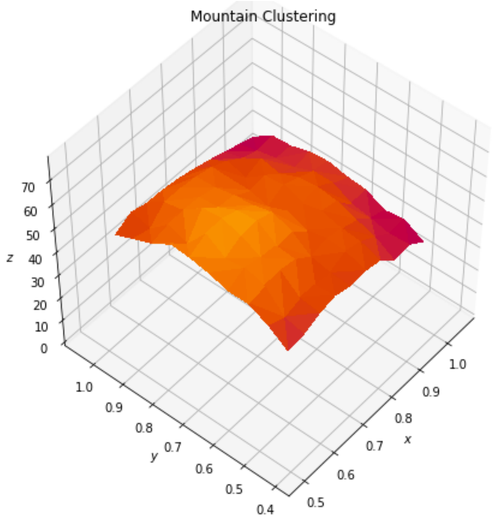

Fig. 4. Initial mountain function for the mountain clustering using the euclidean norm.

1) k-means clustering:

2) c-means clustering:

3) k-medians clustering:

4) Comparison of the methods: In Figures 7, 8, 9 and 10 are shown the different centers provided by the different algorithms explored in this work, for the iris dataset.

Is clear that, even when the number of centers are the same, they vary in terms of localization, sometimes more than others. The three algorithms that need the number of centers tend to give more similar ones, while the ones that explore the whole space differ more in this particular dataset.

B. Credit card dataset

This dataset includes two classes, highly unbalanced: fraud or not fraud.

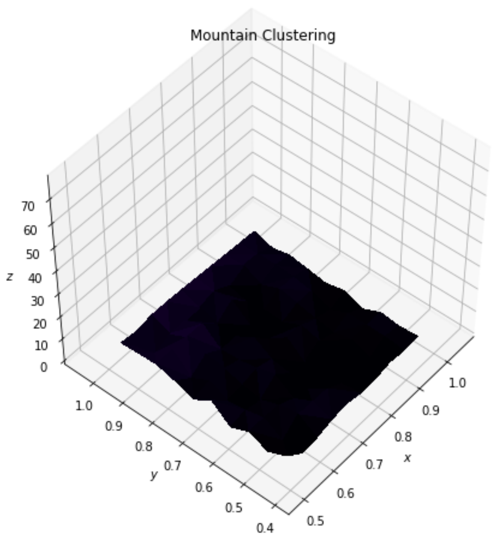

Fig. 5. Final mountain function for the mountain clustering using the euclidean norm.

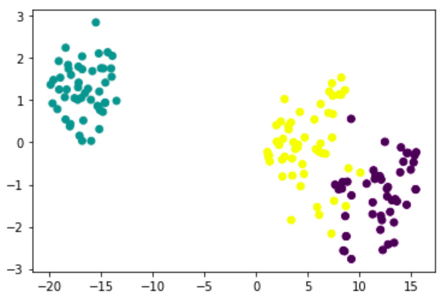

Fig. 6. Clustering for the iris dataset using the mountain algorithm.

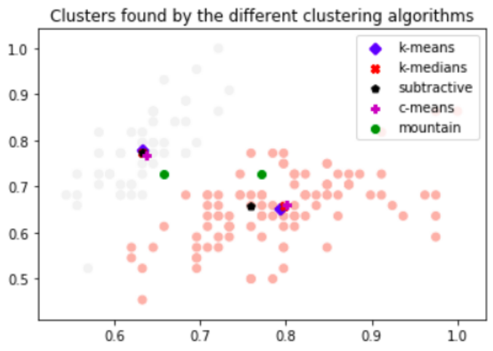

Fig. 7. Clusters given by different methods, $k = 2$.

1) Subtractive clustering: In Tables XIX, XX and XXI are shown the number of clusters found for the three datasets of credit card fraud. In this case, we observe that the cosine norm achieves 2 clusters, the expected ones. But probably more clusters are not a wrong performance, rather than a different form of viewing the data. We will refer then to the validation indices to see when the performance is better. Also,

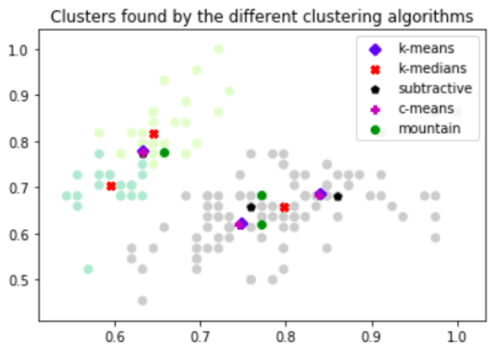

Fig. 8. Clusters given by different methods, $k = 3$.

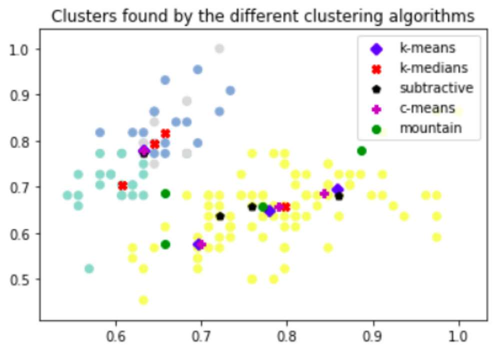

Fig. 9. Clusters given by different methods, $k = 4$.

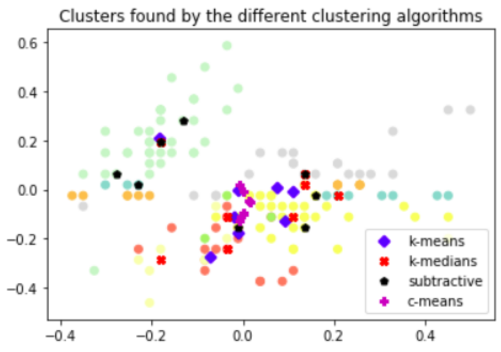

Fig. 10. Clusters given by different methods, $k = 7$.

in Figure 11 is shown the initial density function for the credit card reduced dataset and $r_a = 0.5$, with the euclidean distance. Figure 12 shows the final state.

r_a	euclidean	cosine	cityblock
0.4	10	2	52
0.5	7	2	21
0.7	4	2	8

TABLE XIX
NUMBER OF CLUSTERS FOUND BY THE SUBTRACTIVE ALGORITHM FOR THE CREDIT CARD DATASET.

The algorithm has an emergency stopping criteria, the

r_a	euclidean	cosine	cityblock
0.4	8	2	23
0.5	5	2	11
0.7	3	2	5

TABLE XX
NUMBER OF CLUSTERS FOUND BY THE SUBTRACTIVE ALGORITHM FOR THE CREDIT CARD REDUCED DATASET.

r_a	euclidean	cosine	cityblock
0.4	99	8	99
0.5	99	7	99
0.7	99	4	99

TABLE XXI
NUMBER OF CLUSTERS FOUND BY THE SUBTRACTIVE ALGORITHM FOR THE CREDIT CARD EMBBEDED DATASET.

number of iterations. Due to the fact that they are 100, is clear that for the euclidean distance in the embbeded dataset much more clusters are achieved. This may be due to the spatial distribution of the data, as shwon in the results of Figure 13.

r_a	euclidean	cosine	cityblock
0.4	3.029305	0.981477	3.374885
0.5	3.441991	0.981142	3.349646
0.7	3.737428	0.981221	4.300733

TABLE XXII
DAVIES BOULDIN INDEX FOR THE CREDIT CARD DATASET CLUSTERED BY THE SUBTRACTIVE ALGORITHM.

r_a	euclidean	cosine	cityblock
0.4	34200.815670	181518.597459	4712.546985
0.5	45858.080636	181504.729539	10983.494778
0.7	67334.688094	181520.031384	30413.670844

TABLE XXIII
CALISNKI-HARABSZ SCORE FOR THE CREDIT CARD DATASET CLUSTERED BY THE SUBTRACTIVE ALGORITHM.

In Tables XXII and XXII are shown the results for the validation of the clustering in the original credit card dataset. We saw that, for the cosine metric the number of clusters found are the expected, and is actually revised by these scores, that classify this metric as the one that performs the better.

r_a	euclidean	cosine	cityblock
0.4	2.409494	0.875412	3.139200
0.5	2.088824	0.875412	2.601788
0.7	1.468652	0.875644	3.157173

TABLE XXIV
DAVIES BOULDIN INDEX FOR THE CREDIT CARD REDUCED DATASET CLUSTERED BY THE SUBTRACTIVE ALGORITHM.

r_a	euclidean	cosine	cityblock
0.4	47725.987561	210512.707501	12023.907405
0.5	92681.517486	210512.707501	31914.695617
0.7	113693.471190	210511.841004	59832.246699

TABLE XXV
CALISNKI-HARABSZ SCORE FOR THE REDUCED CREDIT CARD DATASET CLUSTERED BY THE SUBTRACTIVE ALGORITHM.

In Tables XXIV and XXV are shown the indices for the reduced dataset by PCA. Again, the performance shown by these corresponds to the fact that the cosine metric achieves the expected number of clusters.

r_a	euclidean	cosine	cityblock
0.4	1.080875	1.176533	1.253805
0.5	1.050448	1.093338	1.333077
0.7	0.953949	0.866116	1.043586

TABLE XXVI
DAVIES BOULDIN INDEX FOR THE CREDIT CARD EMBBEDED DATASET CLUSTERED BY THE SUBTRACTIVE ALGORITHM.

r_a	euclidean	cosine	cityblock
0.4	83056.373617	174368.043596	77677.563496
0.5	109156.916197	185793.584797	88929.724177
0.7	142020.902180	217192.867615	116444.149920

TABLE XXVII
CALISNKI-HARABSZ SCORE FOR THE REDUCED CREDIT CARD DATASET CLUSTERED BY THE SUBTRACTIVE ALGORITHM.

Finally, Tables XXVI and XXVII are shown the validity indices or the embbeded dataset. These indices show a poor performance in all cases, reflecting the amount of clusters found by the algorithm in this particular dataset. Probably, the information given by the embbeding is not enough to show the full structure of the data.

2) Mountain clustering: Tables XXVIII and XXIX show the number of clusters found by the mountain algorithm in the embbeded and reduced dataset, with variations in the smoothing paramenter σ and the metric.

σ	euclidean	cosine	cityblock
0.4	8	7	2
0.5	4	5	2
0.7	3	3	2

TABLE XXVIII
NUMBER OF CLUSTERS FOUND BY THE MOUNTAIN ALGORITHM IN THE REDUCED CREDIT CARD DATASET.

In this case, we observe that the number of clusters is reduced in comparison with the subtractive algorithm, however, the metric that only finds the two existing clusters in the dataset is the manhattan, and only in the reduced one. The embbeded one, again, does not seem to provide enough information to perform a good clustering.

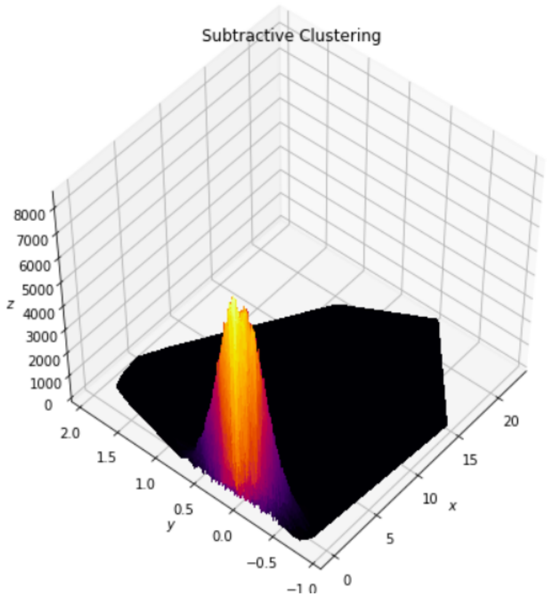

Fig. 11. Initial density function for the credit card dataset in the subtractive clustering algorithm.

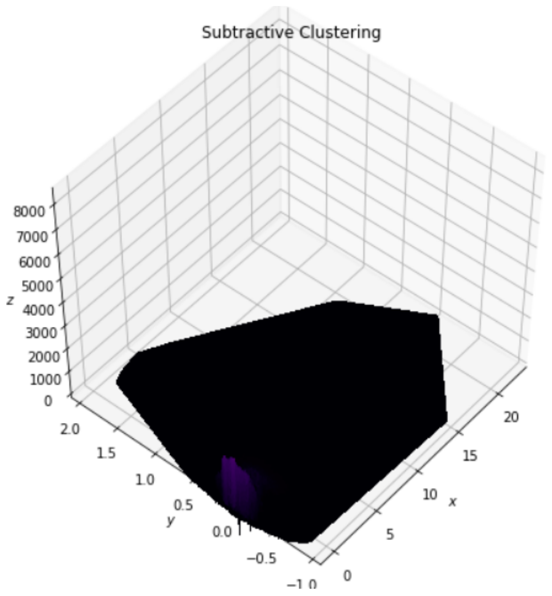

Fig. 12. Final density function for the credit card dataset in the subtractive clustering algorithm.

σ	euclidean	cosine	cityblock
0.4	10	5	9
0.5	9	4	10
0.7	9	4	9

TABLE XXIX
NUMBER OF CLUSTERS FOUND BY THE MOUNTAIN ALGORITHM IN THE EMBBEDED CREDIT CARD DATASET.

In Tables XXX and XXXI are shown the validity indices for the clustering in the reduced dataset.

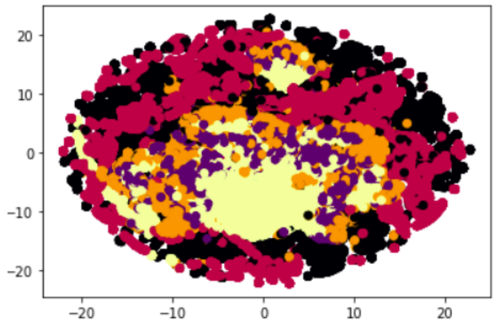

Fig. 13. Clustering for the credit card dataset in the subtractive clustering algorithm.

σ	euclidean	cosine	cityblock
0.4	5.395964	2.539999	6.582867
0.5	5.076245	3.047503	6.582867
0.7	4.635804	1.861457	6.582867

TABLE XXX
DAVIES BOULDIN INDEX FOR THE REDUCED CREDIT CARD DATASET CLUSTERED BY THE MOUNTAIN ALGORITHM.

σ	euclidean	cosine	cityblock
0.4	2612.705876	41225.158046	4029.893595
0.5	3689.213460	58473.394701	4029.893595
0.7	3966.374184	110716.512521	4029.893595

TABLE XXXI
CALINSKI-HARABASZ SCORE FOR THE REDUCED CREDIT CARD DATASET CLUSTERED BY THE MOUNTAIN ALGORITHM.

σ	euclidean	cosine	cityblock
0.4	0.891413	0.967157	0.946406
0.5	0.884992	0.878476	0.891413
0.7	0.884992	0.865683	0.884992

TABLE XXXII
DAVIES BOULDIN INDEX FOR THE EMBBEDED CREDIT CARD DATASET CLUSTERED BY THE MOUNTAIN ALGORITHM.

σ	euclidean	cosine	cityblock
0.4	198415.832838	193511.849750	188520.438083
0.5	208251.768500	207307.628003	198415.832838
0.7	208251.768500	217172.104734	208251.768500

TABLE XXXIII
CALINSKI-HARABASZ SCORE FOR THE EMBBEDED CREDIT CARD DATASET CLUSTERED BY THE MOUNTAIN ALGORITHM.

V. CONCLUSIONS

REFERENCES

[1] A. Sabau, "Survey of Clustering based Financial Fraud Detection Research," *Informatica Economica Journal*, vol. 16, no. 1, pp. 110–122, 2012.

[2] C. Wang and D. Han, "Credit card fraud forecasting model based on clustering analysis and integrated support vector machine," *Cluster Computing*, vol. 22, no. s6, pp. 13 861–13 866, 2019. [Online]. Available: https://doi.org/10.1007/s10586-018-2118-y

[3] L. J. S. Santos, "Bayesian Method with Clustering Algorithm for Credit Card Transaction Fraud Detection," *Revista Română de Statistică*, vol. 66, no. 1, pp. 103–120, 2018.

[4] E. M. Carneiro, L. A. V. Dias, A. M. D. Cunha, and L. F. S. Mialaret, "Cluster Analysis and Artificial Neural Networks: A Case Study in Credit Card Fraud Detection," *Proceedings - 12th International Conference on Information Technology: New Generations, ITNG 2015*, pp. 122–126, 2015.

[5] M. Sathyapriya and V. Thiagarasu, "A cluster based approach for credit card fraud detection system using hmm with the implementation of big data technology," *International Journal of Applied Engineering Research*, vol. 14, no. 2, pp. 393–396, 2019.

[6] S. Kumari and A. Choubey, "Credit Card Fraud Detection Using HMM and K-Means Clustering Algorithm," *International Journal of Scientific Research Engineering & Technology (IJSRET)*, vol. 6, no. 6, pp. 614–619, 2017. [Online]. Available: www.ijsret.org

[7] S. Fashoto, O. Owolabi, O. Adeleye, and J. Wandera, "Hybrid Methods for Credit Card Fraud Detection Using K-means Clustering with Hidden Markov Model and Multilayer Perceptron Algorithm," *British Journal of Applied Science & Technology*, vol. 13, no. 5, pp. 1–11, 2016.

[8] T. K. Behera and S. Panigrahi, "Credit Card Fraud Detection: A Hybrid Approach Using Fuzzy Clustering & Neural Network," *Proceedings - 2015 2nd IEEE International Conference on Advances in Computing and Communication Engineering, ICACCE 2015*, pp. 494–499, 2015.

[9] H. Wang, P. Zhu, X. Zou, and S. Qin, "An ensemble learning framework for credit card fraud detection based on training set partitioning and clustering," *Proceedings - 2018 IEEE SmartWorld, Ubiquitous Intelligence and Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People and Smart City Innovations, SmartWorld/UIC/ATC/ScalCom/CBDCo*, pp. 94–98, 2018.

[10] W. S. Sarle, "Algorithms for clustering data," *Technometrics*, vol. 32, no. 2, pp. 227–229, 1990, https://amstat.tandfonline.com/doi/abs/10.1080/00401706.1990.10484648.

[11] U. Maulik and S. Bandyopadhyay, "Performance evaluation of some clustering algorithms and validity indices," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 12, pp. 1650–1654, Dec 2002.

[12] N. Kasa, A. Dahbura, C. Ravoori, and S. Adams, "Improving credit card fraud detection by profiling and clustering accounts," *2019 Systems and Information Engineering Design Symposium, SIEDS 2019*, pp. 1–6, 2019.

[13] N. O. Francisca, "Data mining application in credit card fraud detection system," *Journal of Engineering Science and Technology*, vol. 6, no. 3, pp. 314–325, 2011.

[14] M. Zamini and G. Montazer, "Credit Card Fraud Detection using autoencoder based clustering," *9th International Symposium on Telecommunication: With Emphasis on Information and Communication Technology, IST 2018*, pp. 486–491, 2019.

[15] V. Hanagandi, A. Dhar, and K. Buescher, "Density-based clustering and radial basis function modeling to generate credit card fraud scores," *IEEE/IAFE Conference on Computational Intelligence for Financial Engineering, Proceedings (CIFEr)*, pp. 247–251, 1996.

[16] M. Hegazy, A. Madian, and M. Ragaie, "Enhanced Fraud Miner: Credit Card Fraud Detection using Clustering Data Mining Techniques," *Egyptian Computer Science Journal*, vol. 40, no. 03, pp. 1110–2586, 2016. [Online]. Available: https://pdfs.semanticscholar.org/2c64/233ad8239884cdd410fb63c63124bd9fb515.pdf

[17] S. Fashoto, O. Owolabi, O. Adeleye, and J. Wandera, "Hybrid Methods for Credit Card Fraud Detection Using K-means Clustering with Hidden Markov Model and Multilayer Perceptron Algorithm," *British Journal of Applied Science & Technology*, vol. 13, no. 5, pp. 1–11, 2016.

[18] E. M. Carneiro, L. A. V. Dias, A. M. D. Cunha, and L. F. S. Mialaret, "Cluster Analysis and Artificial Neural Networks: A Case Study in Credit Card Fraud Detection," *Proceedings - 12th International Conference on Information Technology: New Generations, ITNG 2015*, pp. 122–126, 2015.

[19] T. K. Behera and S. Panigrahi, "Credit Card Fraud Detection: A Hybrid Approach Using Fuzzy Clustering & Neural Network," *Proceedings - 2015 2nd IEEE International Conference on Advances in Computing and Communication Engineering, ICACCE 2015*, pp. 494–499, 2015.

[20] N. Kasa, A. Dahbura, C. Ravoori, and S. Adams, “Improving credit card fraud detection by profiling and clustering accounts,” *2019 Systems and Information Engineering Design Symposium, SIEDS 2019*, 2019.

[21] C. Wang and D. Han, “Credit card fraud forecasting model based on clustering analysis and integrated support vector machine,” *Cluster Computing*, vol. 22, pp. 13 861–13 866, 2019. [Online]. Available: https://doi.org/10.1007/s10586-018-2118-y

[22] K. Hammouda and F. Karray, “A comparative study of data clustering techniques,” *University of Waterloo, Ontario, Canada*, vol. 1, 2000.