Credit Card Fraud and Anomaly Detection Using Clustering Algorithms

Abhinav Nippani, Adwait Patil, Ayush Patel, Shivansh Verma

YouTube Link: https://www.youtube.com/watch?v=9ijjT_ojuRA

ABSTRACT

This project addresses the critical issue of credit card fraud in the digital financial landscape by implementing and analyzing four clustering algorithms: K-means, GMM, DBSCAN and OPTIC. These algorithms, built from scratch, aim to detect and prevent fraudulent activities in credit card transactions. The project leverages the unique strengths of each algorithm K-Means' similarity-based grouping, and DBSCAN's dense region detection—to develop a comprehensive fraud detection system, OPTIC—optimized and generalized version of DBSCAN. A significant aspect includes optimizing the Expectation-Maximization (EM) step in GMM, optimizing clustering in K-Means and incorporating both BFS and DFS approaches in DBSCAN. The primary objective is to create an efficient, robust system capable of identifying anomalous transaction patterns and enhancing the security of digital financial transactions against evolving cyber threats.

1. INTRODUCTION

In the contemporary era of digital banking and e-commerce, credit card fraud has emerged as a significant threat to personal financial security and the stability of the global financial ecosystem. With the surge in online financial activities, the vulnerability to such fraudulent activities has alarmingly increased, necessitating advanced technological interventions. This project aims to address this pressing concern by implementing and analyzing the effectiveness of four prominent clustering algorithms: DBSCAN, OPTIC, K-Means, and GMM.

Each team member brings a unique perspective and motivation to the project, professional backgrounds in finance, and a keen interest in applying machine learning techniques to real-world problems. The project's central question focuses on the effective implementation of these algorithms to accurately identify and mitigate fraudulent activities in credit card transactions. The approach is multi-faceted, beginning with KNN's ability to discern transaction patterns through proximity analysis in a feature space, supplemented by K-Means clustering for grouping transactions based on similarities and enhanced by DBSCAN's capacity to detect dense regions indicative of irregularities, and a generalized version of DBSCAN.

A key innovation in this project is optimizing clustering in K-Means algorithm, which is vital for refining cluster assignments and centroid updates, thus influencing the overall performance of the algorithm. Additionally, the project explores GMM along with the efficacy of both Breadth-First

Search (BFS) approaches in the implementation of DBSCAN, aiming to determine which method yields better performance in the context of credit card fraud detection. Furthermore applying OPTICS which provides a more flexible approach than DBSCAN by producing a reachability plot that captures the density-based structure of the data.

2. ANALYSIS

2.1 DATA

The dataset encapsulates various range of credit card usage metrics. It includes 'BALANCE', indicating available credit, 'BALANCE_FREQUENCY', showing how often the balance is updated, and 'PURCHASES', the total expenditure. Specific spending types are detailed in 'ONEOFF_PURCHASES' and ‘INSTALLMENTS_PURCHASES', while 'CASH_ADVANCE' reflects cash advances taken. Transaction frequencies are captured in 'PURCHASES_FREQUENCY', 'ONEOFF_PURCHASES_FREQUENCY', and 'PURCHASES_INSTALLMENTS_FREQUENCY', with 'CASH_ADVANCE_FREQUENCY' and 'CASH_ADVANCE_TRX' tracking cash advance activities. 'PURCHASES_TRX' notes the number of shopping transactions. Financial limits and behaviours are represented by 'CREDIT_LIMIT', 'PAYMENTS', 'MINIMUM_PAYMENTS', and 'PRC_FULL_PAYMENT', which records the percentage of full payment made. Finally, 'TENURE' indicates the duration of credit card service usage. This comprehensive dataset is pivotal in our analysis, offering insights into spending patterns and aiding in the identification of potentially fraudulent activities. We have also pre-processed the dataset by handling the null values. Data scaling is often used to ensure machine learning algorithms operate more effectively. Specifically, many algorithms can produce misleading results due to different scales among features. Therefore, it's important to bring all features to a similar scale. The scaling method we used is StandardScaler. StandardScaler scales each feature to have a mean of 0 and a standard deviation of 1. This is also known as z-score normalization. Before starting the clustering analysis, it is important to check whether the dataset is suitable for clustering. For this purpose, the Hopkins Statistic is used to check whether the dataset is randomly distributed. The Hopkins Statistic is a test that measures how suitable a dataset is for clustering. A value close to 0.5 indicates that the data has a random distribution, and therefore is not suitable for clustering. On the other hand, if the value is greater than 0.7, the dataset is likely suitable for clustering. On our cleaned dataset we got a score of 0.967 which indicates it is suitable for clustering. Then we also preformed PCA (Principal Component Analysis) and is used to represent the data with fewer features.

2.2 Exploratory Data Analysis (EDA)

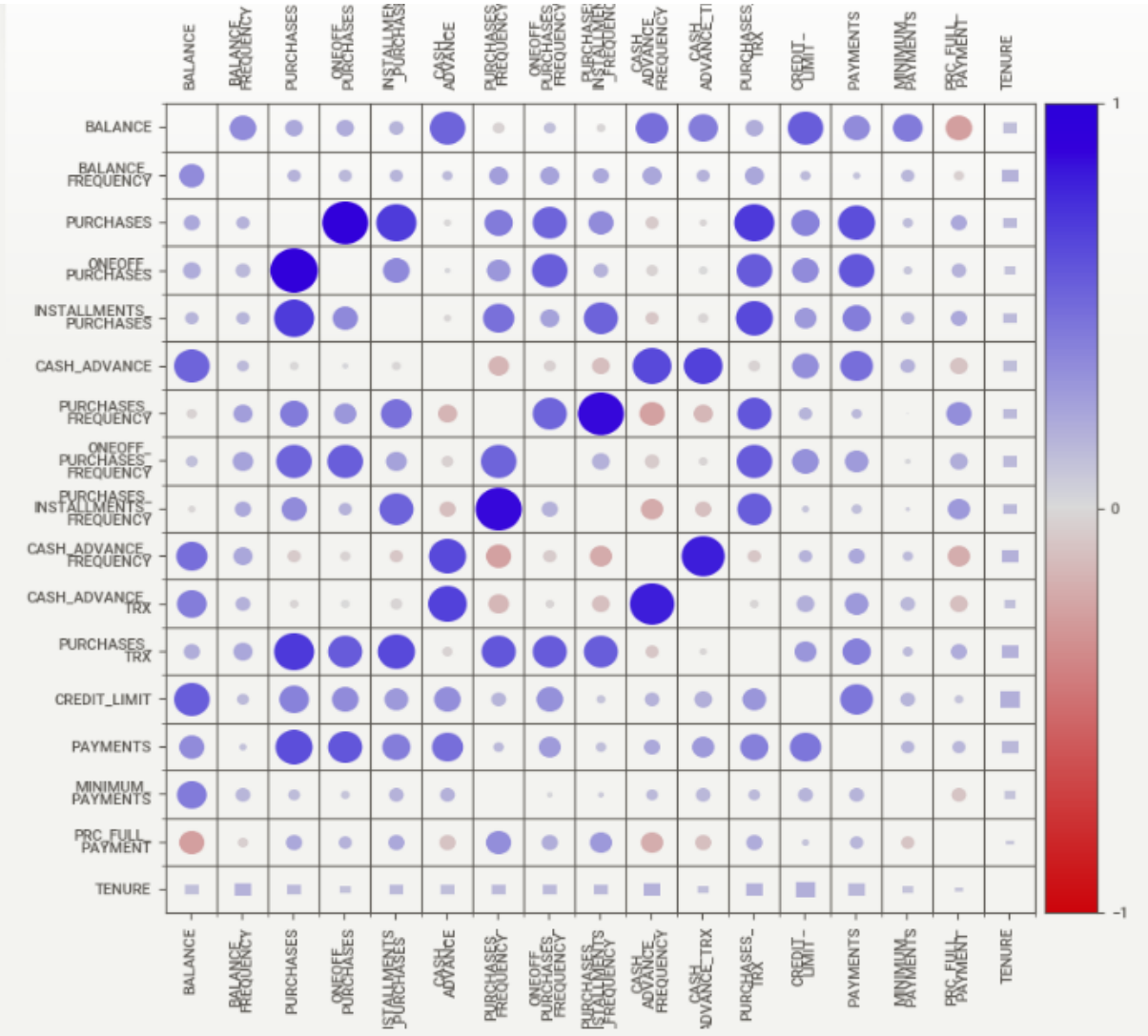

Fig. 1: Correlation matrix between each feature.

The correlation matrix heatmap visualizes the strength and direction of relationships between credit card usage variables. Blue and red circles indicate positive and negative correlations, respectively, with larger circles signifying stronger correlations. The diagonal shows perfect self-correlations. Notably, ‘PURCHASES’ strongly correlates with ‘ONEOFF_PURCHASES’, ‘INSTALLMENTS_PURCHASES’, and ‘PURCHASES_TRX’. Lighter and smaller circles represent weaker correlations, providing insights crucial for feature selection in clustering algorithms.

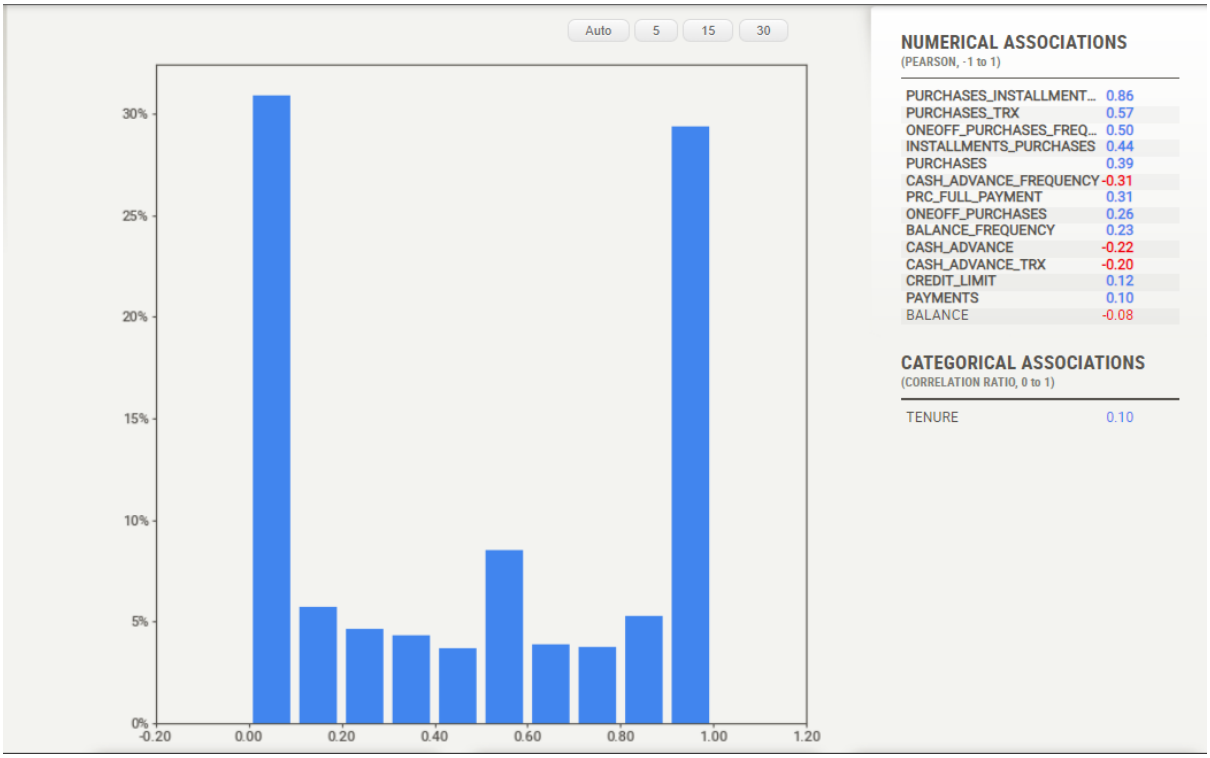

Fig. 2: Purchase Frequency distribution.

Except for PURCHASES_FREQUENCY, none of the headers have a balanced distribution. Moreover, even PURCHASES_FREQUENCY does not have a normal distribution! There is a significant skewness in some headers. As a result, a more detailed examination is needed to identify outliers. Also, conventional clustering algorithms may not be efficient enough.

3. ALGORITHMS

3.1 GMM

Gaussian Mixture Model (GMM) is another clustering algorithm that extends beyond K-Means by incorporating probabilistic models. Unlike K-Means, GMM assumes that the data is generated from a mixture of several Gaussian distributions, each representing a cluster. GMM provides a more flexible framework where data points are not strictly assigned to a single cluster but instead have probabilities of belonging to each cluster.

The Expectation-Maximization (EM) algorithm is employed in the context of GMM to iteratively update parameters. The EM algorithm consists of an E-step, where probabilities (responsibilities) of data points belonging to each cluster are computed, and an M-step, where the parameters of the Gaussian distributions are updated based on these probabilities.

In the E-step, the algorithm calculates the probability of each data point belonging to each cluster, considering the Gaussian distributions with their means, covariances, and weights. In the M-step, the algorithm updates the parameters, including the means, covariances, and weights of the Gaussian distributions, based on the computed probabilities. Similar to the K-Means approach, the optimal

number of clusters in GMM can be determined using techniques such as the elbow method, Silhouette score, or other model selection criteria.

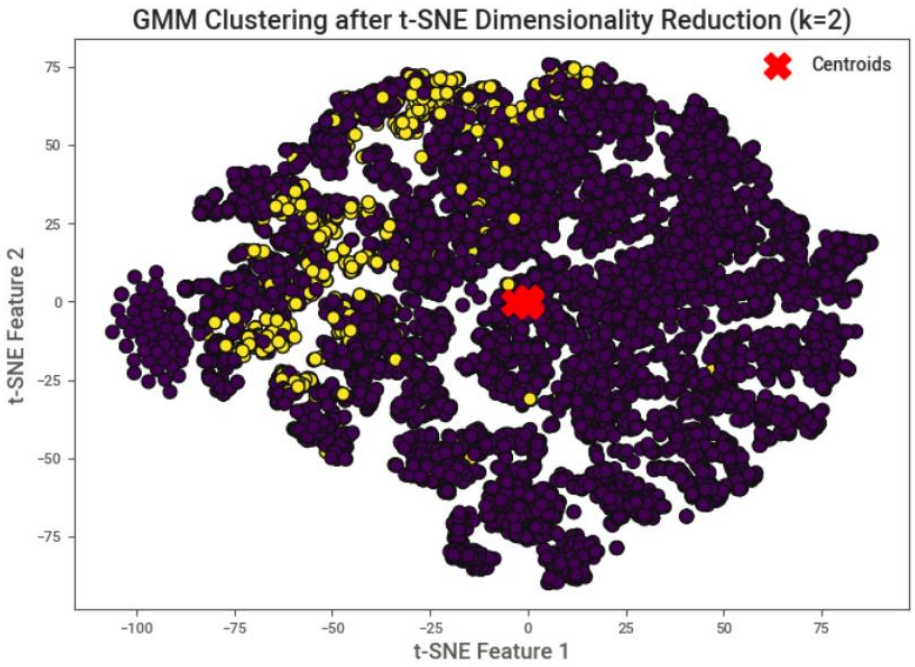

Fig. 3: GMM Plot with t-SNE

3.2 K-Means

Implementing K-Means clustering on the pre-processed data. In this application we try to optimize the clustering step using NumPy array comprehensions to create clusters with optimum time complexity. We then compare this to scikit-learn's KMeans to verify if the clustering meets the standards.

K-Means is a clustering algorithm used to divide the data into ‘k’ number of clusters. The algorithm operates by randomly initializing each cluster center, and then iteratively updating these centers to be the average of the data points in the clusters. This process continues until a defined criterion (e.g., a situation where the canters do not move) is met. The elbow method and Silhouette score were used as supportive to the K-Means process.

Elbow Method One of the toughest decisions for the K-means algorithm is determining how many clusters the data should be divided into. The elbow method is a common technique used for this decision. This method involves calculating the total within-cluster sum of squares (WCSS) for different k values. As the value of k increases, WCSS decreases. However, after a certain k value, this reduction amount becomes insignificant. This point called the "elbow" helps us to determine the optimal k-value.

Silhouette score measures how well the clusters are defined. This score takes values between -1 and 1. A value close to 1 indicates that the points are similar within their clusters and different from other clusters, while a value close to -1 indicates that these points are clustered incorrectly. A value of 0 indicates uncertainty about how far apart the clusters are from each other.

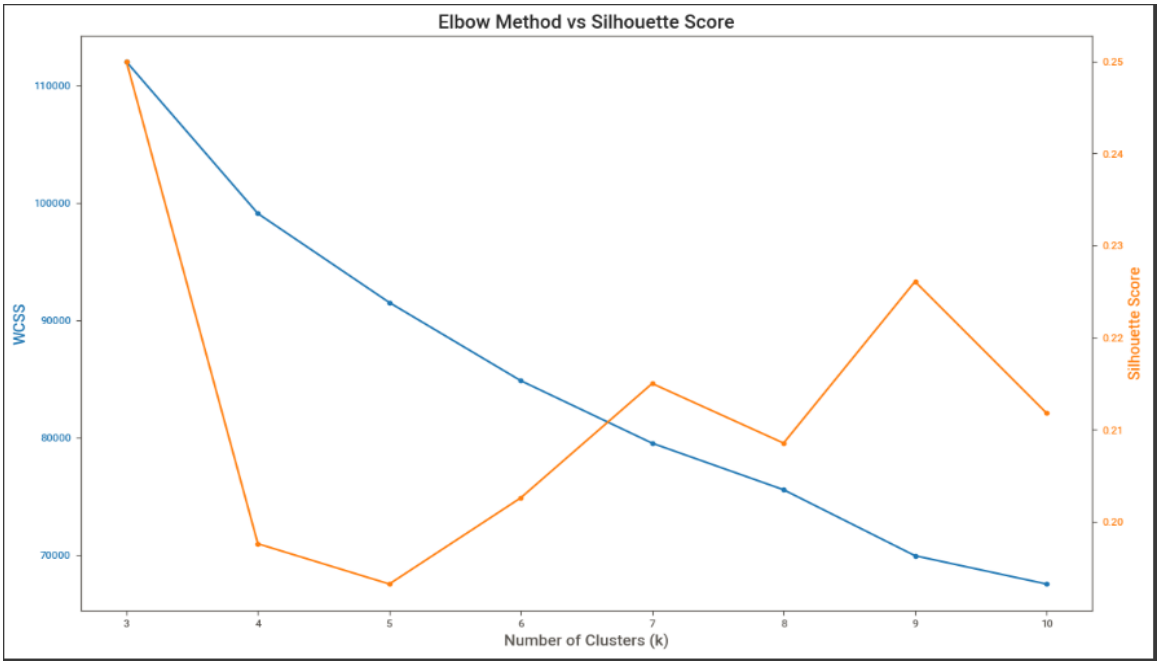

Fig. 4: Silhouette Score vs Number of Clusters

1. We can observe that as the number of clusters increases the silhouette score decreases
2. This states that the quality of clusters reduces with increase in the number of clusters
3. The optimum number of clusters for this problem is 2.

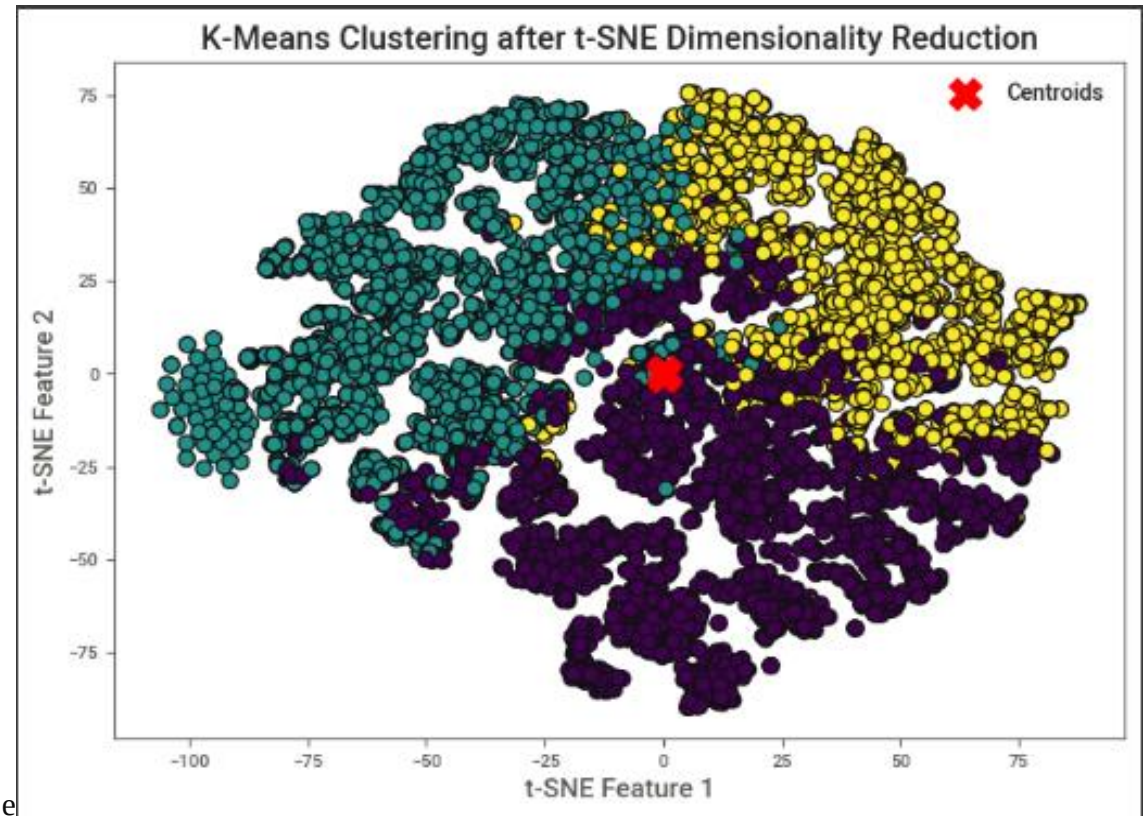

Fig. 5: K-means clusters with t-SNE for K = 3

3.3 DBSCAN

DBSCAN - Density-Based Spatial Clustering of Applications with Noise. Finds core samples of high density and expands clusters from them. Good for data which contains clusters of similar density.

The worst-case memory complexity of DBSCAN is O(N^2), which can occur when the eps param is large and min_samples is low.

This non-parametric approach for density-based clustering takes a set of points in space and groups the points that are tightly spaced (i.e., have many nearby neighbors). It labels as outliers the points that are isolated in low-density areas (i.e., whose nearest neighbors are too far away). One of the most widely used and frequently quoted clustering methods is DBSCAN.

DBSCAN has two parameters:

1.eps: It designates the area surrounding a data point; that is, two points are deemed to be neighbors if their distances are less than or equal to eps. A sizable portion of the data will be regarded as anomalies if the eps value is set too low. The clusters will merge and most of the data points will be in the same clusters if it is selected in a very high amount. The k-distance graph can be used as a basis for determining the eps value.

2.min_points: The smallest number of neighbors (data points) in an eps radius is known as MinPts. A bigger value of MinPts must be selected for larger datasets. Generally speaking, MinPts >= D+1, which is the minimal MinPts, may be obtained from the number D of dimensions in the dataset. MinPts must have a minimum value of at least 3.

We have implemented Density Based Spatial Clustering of Algorithms with Noise (DBSCAN) Algorithm by using a queue and taking inspirations from Breadth First Search (BFS). We are first creating neighbourhoods by finding all the points that are within epsilon distance of each other and then we run DBSCAN by using BFS and checking their neighbours and assigning them the cluster labels while visiting them.

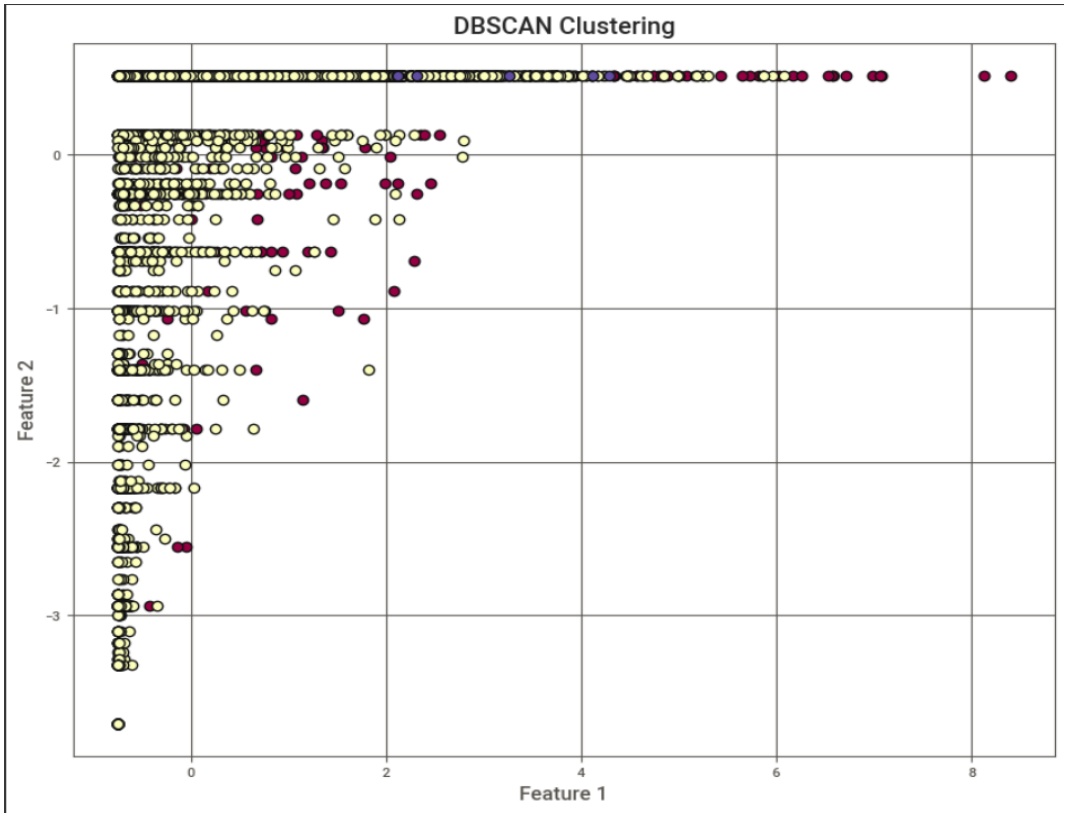

Fig 6. Plotting DBSCAN results for two features and seeing the segregation in classes.

3.4 OPTICS

OPTICS (Ordering Points To Identify the Clustering Structure) is a density-based clustering algorithm and can be considered as a generalization of DBSCAN. However, it is known for its ability to identify structures of different densities in the dataset, instead of using a single eps value. This is algorithm uses a BallTree data structure. Constructing a BallTree has a time complexity of O(n log n), where n is the number of points in the dataset.

Working Principle: - The reachability distance is calculated for each data point. This determines how close a point is to another point at a certain density. - Based on these distances, a reachability graph is created. - Valleys in the graph indicate density-based clusters.

The _query_region method queries the BallTree for neighbors within max_eps radius of a point. BallTree queries have a time complexity of O(log n) for a single query. However, since this is called for each point that is not processed yet, and in the worst case, all points could be queried, the total time complexity for all queries in the worst case could be O(n log n).

_expand_cluster_order Method: This method iterates through the neighbors of a given point to expand the cluster order. In the worst case, where the data points are densely packed, and each point has many neighbors, this could lead to a situation where nearly every point is considered a neighbor. This method could potentially touch every other point in the dataset in a single call, leading to a worst-case complexity of O(n) for a single call. But since _expand_cluster_order is called within a loop in the fit

method and could potentially process each point in the dataset, this results in a nested loop situation with a worst-case complexity of O(n^2).

fit Method: The fit method contains a for-loop that iterates over each point in the dataset, calling _expand_cluster_order for each unprocessed point. The complexity of _expand_cluster_order is O(n) in the worst case, as explained above, and thus the total complexity of the fit method is O(n^2) in the worst case.

Combining these complexities, the overall worst-case time complexity of the OPTICS implementation is O(n^2), dominated by the nested iterations in the fit method and the _expand_cluster_order method. However, it's important to note that the average time complexity might be better than the worst case, depending on the data distribution and the parameter settings for min_samples and max_eps. In practice, OPTICS can perform better than O(n^2) when the dataset has a sparse distribution or when max_eps limits the number of points to be considered in each neighbourhood query.

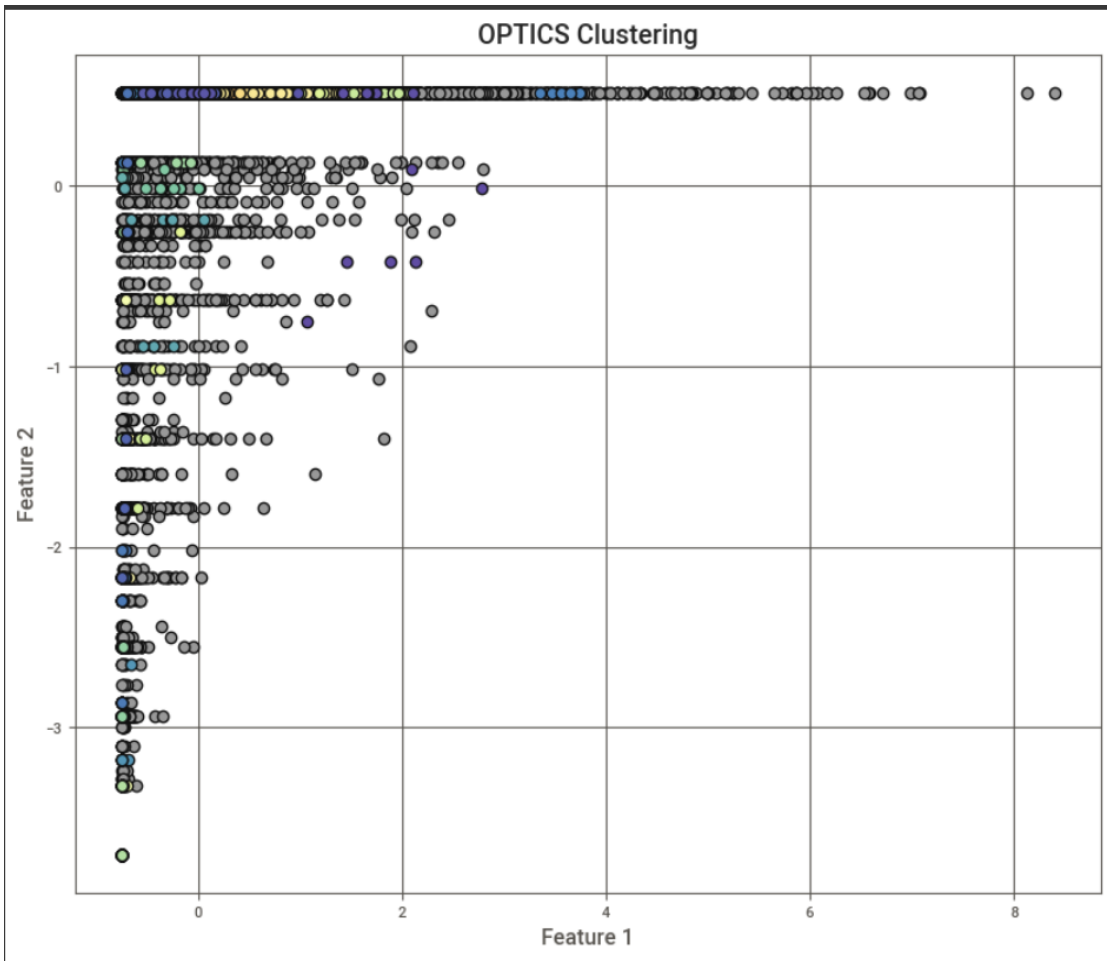

Fig 7. OPTIC Clusters without dimensionality reduction

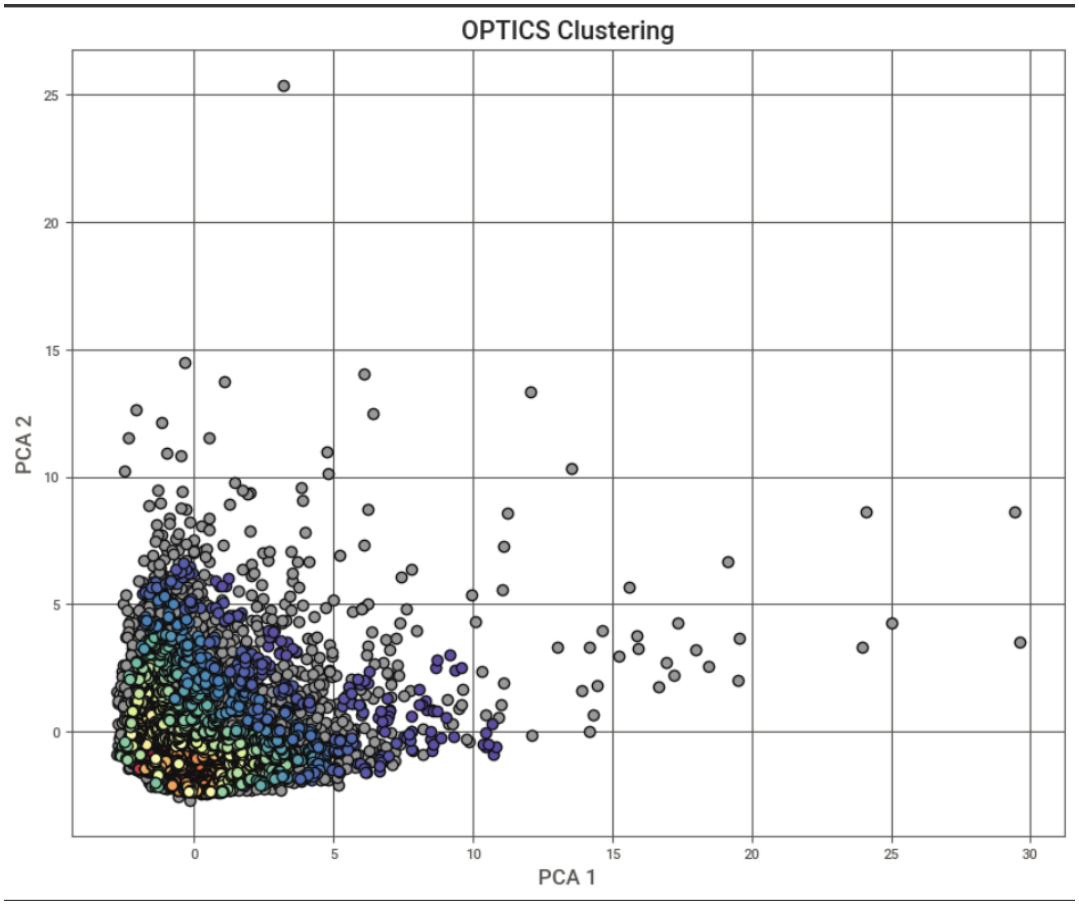

Fig. 8: Optic Clustering with PCA

4. INFERENCES

Cluster 0:

Balance: The customers of this group carry an average level balance. This means they have neither very high nor very low spending capacity.

Purchases: Customers in this group make purchases at an average level, however, they make these purchases both at once and in instalments.

Cash Advance: There is an average level of cash advance usage.

Purchase Frequency: These customers shop regularly, but not very frequently.

Credit Limit: The credit limit of this group may be lower compared to other groups.

They can be evaluated as average users. These users can use their credit cards for both daily shopping and larger purchases. The medium level of cash advance usage shows that they sometimes meet their cash needs from their cards.

Cluster 1:

Balance: The customers of this group carry quite a high balance, which means they have a high spending capacity.

Purchases: They shop less, which means this group is less active.

Cash Advance: A high amount of cash advance usage is evident in this group, which shows these customers often have cash needs.

Purchase Frequency: These customers shop less frequently.

Credit Limit: The credit limit of this group is higher than the other groups.

This group seems to include individuals who use the credit card for larger purchases or high-priced acquisitions. High balances and credit limits indicate that this group has a high financial capacity and spending potential.

Cluster 2:

Balance: The balance of this group is high, but not as much as Cluster 1.

Purchases: Customers in this group make very high amounts of purchases and these purchases are made both at once and in instalments.

Cash Advance: Cash advance usage is slightly above average.

Purchase Frequency: This customer group shops frequently.

Credit Limit: The credit limit of this group is average compared to the other two groups.

There's a high probability that this group consists of young individuals or students, this group may represent individuals who use their cards for daily small expenses. Hence, this could be the reason for their frequent shopping and generally small amount expenditures.

However, to increase the accuracy of these interpretations, additional demographic, or behavioural information (such as age, occupation, income level) may be needed. Such additional information could help us describe each cluster more detailed and accurately. Unfortunately, these are not available in the dataset.

5. RESULTS

K-Means:

Normal

Silhouette Coefficient (SC): 0.25 This metric ranges between -1 and 1. Higher values indicate that the object is well-fitted to its own cluster and has a weak match with neighbouring clusters. The score of 0.25 suggests that the clusters overlap to some extent and there is room for improvement.

Calinski-Harabasz Index (CHI): 1604.40 It is the ratio of within-cluster variance to between-cluster variance. Higher values indicate that clusters are dense and well-separated.

Davies-Bouldin Index (DBI): 1.60 It is the average similarity ratio of each cluster with its most similar cluster. Values close to 0 are better. The score of 1.60 indicates a moderate clustering structure.

DBSCAN Results

DBSCAN generally produced better SC scores for normal and pca data clusters compared to KMeans. Especially with a 0.80 SC, the pca data set shows very well-defined clusters. DBI scores also reflect better cluster separation, especially for the pca data set, compared to KMeans. However, the CHI score, while being better for the normal data set, drops significantly for the UMAP data set. This indicates less dense clusters or greater cluster overlap.

OPTICS

Metrics for these data sets are not promising. Negative or low silhouette scores and very low CHI scores indicate poor cluster definition and density. Silhouette score of -0.18.

6. CONCLUSION

KMeans with UMAP or PCA KMeans combined with UMAP or PCA seems to yield the best results. Although PCA shows a slight advantage in silhouette coefficient, both methods indicate well-defined clusters. This could make it worthwhile to evaluate both approaches for a specific application.

The performance of GMM is currently unsatisfactory, but extending the number of iterations could potentially enhance its results.

DBSCAN The combination of DBSCAN with PCA has a fairly high silhouette coefficient, indicating that this approach also defines clusters quite well. However, the results obtained with UMAP are not as successful for DBSCAN.

OPTICS The performance of OPTICS is generally lower compared to the other three methods depending on the dataset used. Particularly, the negative silhouette coefficients indicate that the clusters are quite poorly defined.

In conclusion, if your primary goal is to obtain well-defined clusters, it is recommended to combine KMeans with PCA or UMAP. On the other hand, it might be worth trying DBSCAN to deal with more complex structures, especially when combined with PCA. Despite the lower overall performance of OPTICS for this particular dataset, it might produce different results on different datasets, therefore, it's important to evaluate before using it.

7. REFERENCES

- Bishop, C. M. (2006). Pattern Recognition and Machine Learning. Springer.
- MacKay, D. J. C. (2003). Information Theory, Inference, and Learning Algorithms. Cambridge University Press.
- Hastie, T., Tibshirani, R., & Friedman, J. (2009). The Elements of Statistical Learning. Springer.
- Murphy, K. P. (2012). Machine Learning: A Probabilistic Perspective. MIT Press.
- Theodoridis, S., & Koutroumbas, K. (2009). Pattern Recognition. Academic Press.
- Scikit-learn Contributors. (2023). Scikit-learn: Machine Learning in Python. Scikit-learn GMM Documentation.
- Lloyd, S. P. (1982). Least squares quantization in PCM. IEEE Transactions on Information Theory, 28(2), 129–137.
- MacQueen, J. (1967). Some Methods for classification and Analysis of Multivariate Observations. In Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability (Vol. 1, pp. 281–297).
- Hartigan, J. A., & Wong, M. A. (1979). Algorithm AS 136: A K-Means Clustering Algorithm. Journal of the Royal Statistical Society. Series C (Applied Statistics), 28(1), 100–108.
- Hastie, T., Tibshirani, R., & Friedman, J. (2009). The Elements of Statistical Learning. Springer.