Report

Cluster Detection and Interpretation Approach for Financial Data

Ria (190001051)
Deepkamal Singh (190001011)
Mahanya Kochhar (190005021)

Submitted to:
Dr. Kapil Ahuja
Professor
Department of Computer Science and Engineering
Indian Institute of Technology, Indore

Contents

1 Introduction

Labelled data of ground truth is highly scarce in many financial applications such as fraud detection , reject inference and credit evaluation.Unsupervised algorithms like clustering help identify sub-patterns of user data and identify potential risks and opportunities. Due to complexity of human behaviours and changing social environments, finding optimal clusters with the appropriate clustering algorithm is challenging and real-world model activation is further needed to strengthen and verify the results given by the clustering.
Clustering algorithms like K-Means need the selection of k manually and are very sensitive to initial centroid selection.Further K-Means has trouble clustering data of varying sizes and densities and is highly sensitive to noise and outliers.
With this background, an integrated cluster detection approach is needed to guide base clustering algorithms to detect clusters adaptively. Ada-Ellip technique, proposed in the research work guides base clustering algorithms such as K-Means to detect the cluster number automatically. We have implemented Ada Ellip and observed its results for the financial dataset. To solve the issue of noise and outliers in datasets, the automatic cluster algorithm DBSCAN which is robust to noise has also been evaluated and proposed by us for financial datasets.

2 Dataset

For our entire work, we have worked with CC General dataset from Kaggle that has also been used in our base research paper **"An Integrated Cluster Detection, Optimization, and Interpretation Approach for Financial Data Tie Li , Gang Kou , Yi Peng , and Philip S. Yu , Life Fellow, IEEE"** [1]

2.1 Dataset Characteristics

Our Credit Card dataset consisted of 8950 entries having 18 attributes [2]. Some of the attributes are:

BALANCE : Balance amount left in their account to make purchases.

BALANCE FREQUENCY : How frequently the Balance is updated, score between 0 and 1 (1 = frequently updated, 0 = not frequently updated).

PURCHASES : Amount of purchases made from account.

CREDIT LIMIT : Limit of Credit Card for user.

ONEOFF PURCHASES : Maximum purchase amount in single transaction.

INSTALLMENTS PURCHASES : Amount of purchase done in installment by the user.

CASH ADVANCE : Cash in advance paid by the user.

PURCHASES FREQUENCY : Frequency of Purchases are being made by the user, score between 0 and 1 (1 = frequently purchased, 0 = not frequently purchased).

ONEOFF PURCHASES FREQUENCY : Frequency of Purchases happening in single transaction being done by user(1 = frequently purchased, 0 = not frequently purchased).

TENURE : Tenure of credit card of an user.

2.2 Preprocessing the Data

2.2.1 Data Cleaning

We remove all the rows having NULL value as one of their attribute values. Similarly filling with mean value of feature is another possible solution. After data cleaning we are left with 8636 entries. We also remove the customer id attribute used for indexing as it is redundant attribute.

2.2.2 Standardization of the Data

We used standardize method which removes the mean and does scaling to unit variance.
This standard score of a sample x is calculated as:

$$z = (x - u)/s$$

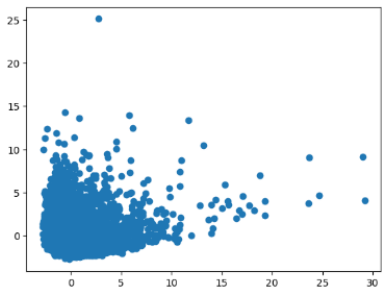
where u is the mean of the training samples or zero, and s is the standard deviation of the training samples. Centering and scaling happen independently on each feature by computing the relevant statistics on the samples in the training set. Mean and standard deviation are then stored to be used on later data using transform. Standardization of a dataset is a common requirement for many machine learning estimators: they might behave badly if the individual features do not more or less look like standard normally distributed data

2.2.3 Dimensionality reduction

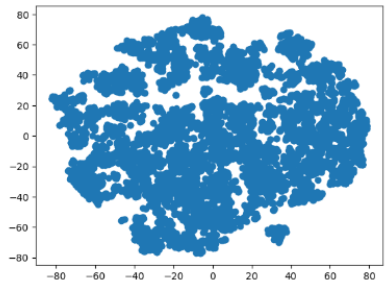
We used **t-SNE** [3] and **PCA** to do dimensionality reduction. We found that results given by **t-SNE** were more appropriate for clustering as **t-SNE** preserves local structure of data whereas **PCA** preserve global structure of data. If our data has n samples with m attributes then **t-SNE** works in $O(n^2)$ where as **PCA** works in $O(m^2n)$. Although **t-SNE** is slower it is preferred method for small sample sizes as in our case.

S.NO.	PCA	t-SNE
1.	It is a linear Dimensionality reduction technique.	It is a non-linear Dimensionality reduction technique.
2.	It tries to preserve the global structure of the data.	It tries to preserve the local structure(cluster) of data.
3.	It does not work well as compared to t-SNE.	It is one of the best dimensionality reduction technique.
4.	It does not involve Hyperparameters.	It involves Hyperparameters such as perplexity, learning rate and number of steps.
5.	It gets highly affected by outliers.	It can handle outliers.
6.	PCA is a deterministic algorithm.	It is a non-deterministic or randomised algorithm.

Difference between PCA and TSNE



PCA



t-SNE

Results obtained for PCA and TSNE

2.3 PCA : Dimensionality Reduction Technique

Principal component analysis, or PCA [4], is a dimensionality reduction method that is often used to reduce the dimensionality of large data sets, by transforming a large set of variables into a smaller one that still contains most of the information in the large set.

Reducing the number of variables of a data set naturally comes at the expense of accuracy, but the trick in dimensionality reduction is to trade a little accuracy for simplicity. Because smaller data sets are easier to explore and visualize and make analyzing data points much easier and faster for machine learning algorithms without extraneous variables to process.

Steps involved in PCA are as follows:

1. Standardize the range of continuous initial variables
2. Compute the covariance matrix to identify correlations
3. Compute the eigenvectors and eigenvalues of the covariance matrix to identify the principal components
4. Create a feature vector to decide which principal components to keep
5. Recast the data along the principal components axes

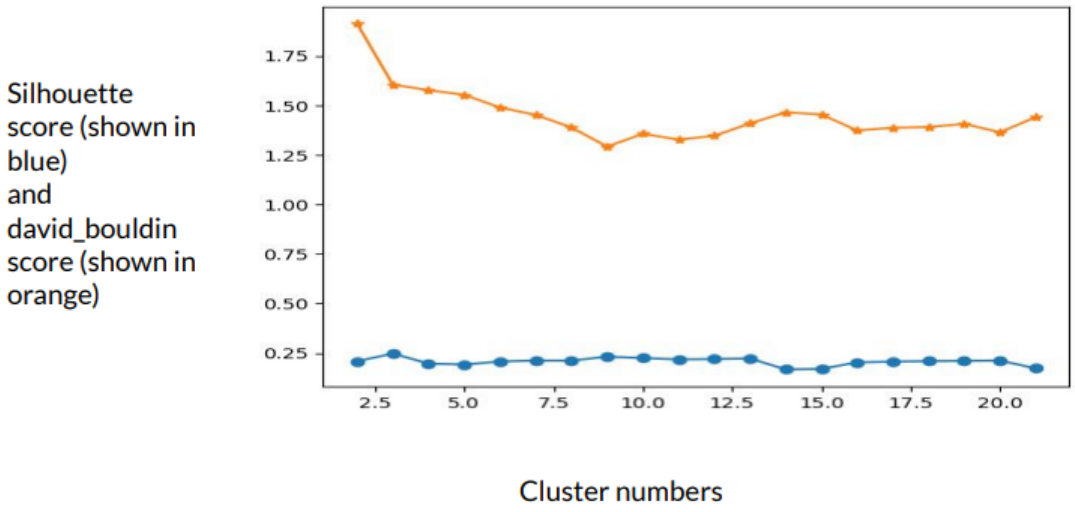
3 Clustering Algorithms

Before heading onto automatic cluster detection and robust to noise clustering algorithms, we perform K-Means on the data to:
1. To get an idea of the clusters and their distributions

2. Evaluated possible cluster numbers with silhouette and davies bouldin index(DBI) as metrics.
3. Perform cluster interpretation on these results

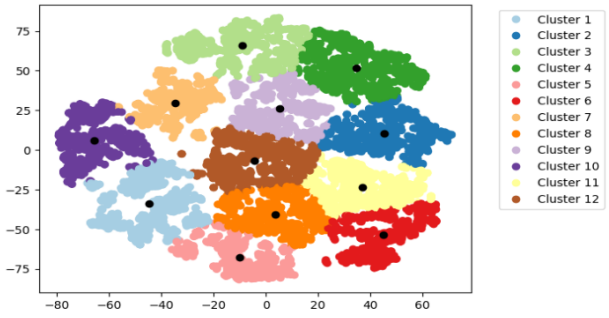
3.1 Methodology

We bruteforced on the no of clusters between 2 to 22. The following possible cluster range was chosen as beyond this range, choosing too many clusters will lead to artificially deflating the inertia metric and could lead to substantial overheads.
Silhouette values closer to 1 and DBI values closer to 0 indicate good cluster number selection. As per the cluster number estimation results, cluster numbers between 10-13 are good choices as these cluster numbers lead to maximum silhouette values and minimum DBI values.
We have chosen 12 as an ideal cluster number based on the silhouette and DBI metrics.



3.2 Cluster prediction based on K-Means

K-Means implementation was done using scikit-learn library [5] K-means class using default parameters.



Clustering using K-Means

3.3 Time Complexity

Time complexity of k-means clustering is $O(NTK)$ where N is total number of datasets, K is the number of partitions and T is the number of iterations.T is not deterministic and varies in each K-Means invocation. Typically, K-Means converged in around 20-25 steps in our implementation after observing several runs.

3.4 Cluster Characterization

Column Name	Metrics	Cluster 1	Cluster 10	Cluster 11	Cluster 12	Cluster 2	Cluster 3	Cluster 4	Cluster 5	Cluster 6
BALANCE	mean	193.549577	189.821055	2508.814551	1025.243687	3428.512165	1332.055473	4086.210959	97.112117	1304.146068
BALANCE_FREQUENCY	mean	0.667290	0.449253	0.992255	0.990018	0.986443	0.979567	0.987579	0.961572	0.967455
PURCHASES	mean	536.809858	324.755376	2767.108087	595.399641	1197.914454	29.811154	145.342862	930.576937	5058.943378
ONEOFF_PURCHASES	mean	144.570840	248.633830	2014.943128	426.193601	661.598326	25.369832	110.175767	107.396849	3312.355574
ISTALLMENTS_PURCHASES	mean	393.339609	76.187702	752.164958	169.380458	536.751048	4.465361	35.277726	823.336743	1747.601318
CASH_ADVANCE	mean	198.201974	390.319981	311.316174	80.392500	2501.076820	819.999957	4165.566113	10.926912	206.137978
PURCHASES_FREQUENCY	mean	0.633445	0.175082	0.939603	0.421173	0.746093	0.022281	0.084848	0.944553	0.927639
_PURCHASES_FREQUENCY	mean	0.071273	0.087984	0.737542	0.217265	0.288594	0.016672	0.051218	0.057072	0.752645
ISTALLMENTS_FREQUENCY	mean	0.534999	0.084357	0.586615	0.218899	0.574960	0.005108	0.036281	0.915474	0.635924
SH_ADVANCE_FREQUENCY	mean	0.036864	0.046047	0.051462	0.017080	0.365978	0.186262	0.468056	0.003815	0.023969
CASH_ADVANCE_TRX	mean	0.646154	0.969359	1.022346	0.290842	9.522562	3.409856	13.869423	0.056338	0.589527
PURCHASES_TRX	mean	9.351479	3.032033	38.861732	8.471535	21.276565	0.373798	1.707291	19.088028	63.136824
CREDIT_LIMIT	mean	2980.608870	4011.699164	6425.139665	2484.135914	5946.870451	2095.032051	6724.700762	3826.320423	7957.478501
PAYMENTS	mean	813.084775	1031.717434	2486.950392	836.726000	2697.764141	873.871032	3185.633939	983.266024	4935.876871
MINIMUM_PAYMENTS	mean	192.009359	161.920189	1065.008927	543.653571	1318.759837	498.751530	1378.558117	194.112599	682.127463
PRC_FULL_PAYMENT	mean	0.474830	0.087965	0.045361	0.022135	0.039414	0.024532	0.030697	0.648812	0.593099
TENURE	mean	10.472189	11.837047	11.963687	11.928218	11.883552	11.966346	11.840044	11.959507	11.944257

Cluster Characterization

- We can observe that cluster 4 has the highest balance which indicates this cluster include more wealthy users which is supported by the fact this cluster also has the highest credit limit and cash advance.
- We can also see cluster 11 and cluster 6 with higher balance frequency consists of customers making more number of purchases which is reflected in their payments value as it is also very high.
- We can see cluster 2 have high credit limit, high cash advance and balance.

3.5 Why not K-Means ?

- Not adaptive, need to select K
- Affected by Outliers
- Doesn't work well when clusters are not spherical in nature

This motivates us to switch to adaptive clustering

4 Adaptive Clustering with Ada Ellip

Ada Ellip is a framework to add adaptivity to base clustering algorithms such that automatic clustering is achieved.

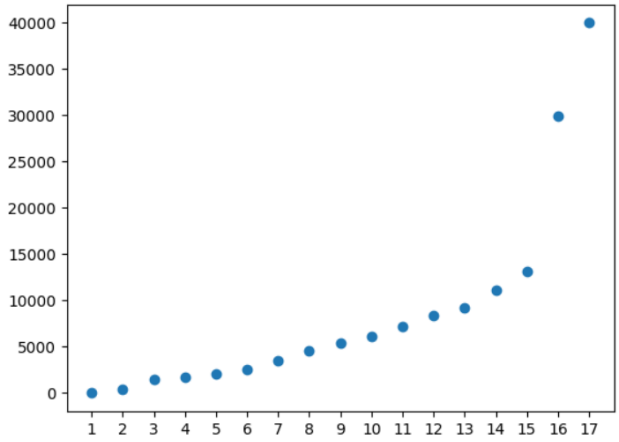
4.1 Theoretical Background

A kernel matrix K is defined as $K = \begin{bmatrix} k_{11} & k_{12} & ... & k_{1m} \\ k_{21} & k_{22} & ... & k_{2m} \\ ... & ... & ... & ... \\ k_{n1} & k_{n2} & ... & k_{nm} \end{bmatrix}$

where k is the kernel function of pairwise data points. This kernel function is used to measure the similarity between pairwise data points and is defined as $k(x_i, x_j) = \phi(x_i)^T * \phi(x_i)$: where x_i and x_j are pairwise data points and ϕ is a transformation function that projects data into reproducing kernel space. Since our data can be separated well after dimensionality reduction, we do not need this transformation.
This Kernel Matrix is $N \times N$ where N is the number of data points. Spectral Clustering aims to compute the eigen values of this kernel matrix. The position of the biggest eigengap can be used to infer the possible cluster number when the eigenvalues are ordered increasingly [6]. Although this approach has a strong background, it is infeasible for large N.

4.1.1 Eigen Value Computation

To obtain eigenvalues of the Kernel Matrix, we make use of SVD [7]. SVD states that the eigenvalues of $X^T X$ is the same as eigenvalues of XX^T.In our case, X is an $N \times M$ matrix with N data points and M features. So, $X^T X$ is an $M \times M$ matrix where M is the number of features and hence eigen value decomposition is much more efficient for this matrix. We have used the same and computed eigenvalues as follows :



Computed Eigen values

Clearly, at 15th eigenvalue we observe the biggest eigen gap and thus can infer the possible cluster number. However this appoach has the following lim-

itations:

- It is unavailable when the eigenvalue curve is smooth and there is no evident big gap between the eigenvalues

4.2 Ada Ellip

Ada - Ellip is a recursive clustering approach which repeatedly divides some cluster until we obtain clusters which are acceptable.
We consider all our data points as a single cluster initially and keep dividing them repeatedly as per

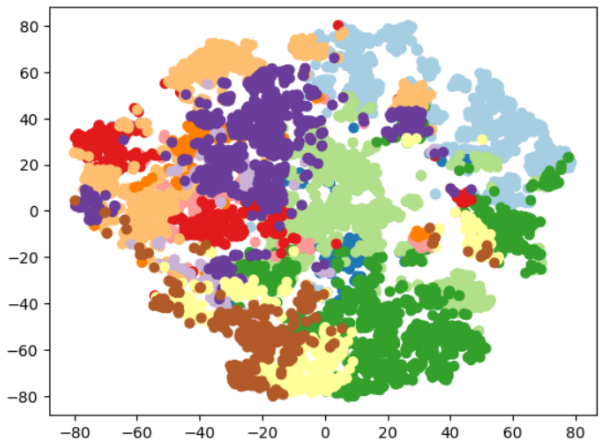
4.2.1 Algorithm

This division occurs in the following manner :

- First for a given cluster its eigenvalues are calculated as per SVD
- Then we calculate qci value which is $q_{ci} = s_1^2 / \sum_{j=1}^{m'} s_j^2$ Where s_1 is the largest Eigenvalue and s_i are given values sorted in descending order. q_{ci} is a cluster quality evaluation metric where a dominantly large eigenvalue indicates nodes are densely connected and the data points show great similarities.
- If q_{ci} is greater than the threshold we don't divide further else
- We calculate k_s which is minimum k such that $\sum_{j=1}^{k} s_j / \sum_{j=1}^{m'} s_j^2$ is greater than the threshold value selected by us.
- Then we apply K-means with number of cluster as ks and recursive solve for each given cluster that does not meet the threshold cluster quality value.

4.3 Result

We implemented Ada-Ellip as per the research paper and observed the following results:



Ada Ellip Result

The results are sensitive to the threshold value selected for the calculation of ks. Furthermore, the silhouette value 0.0749 obtained was not very large.

5 Dealing with Noisy Datasets

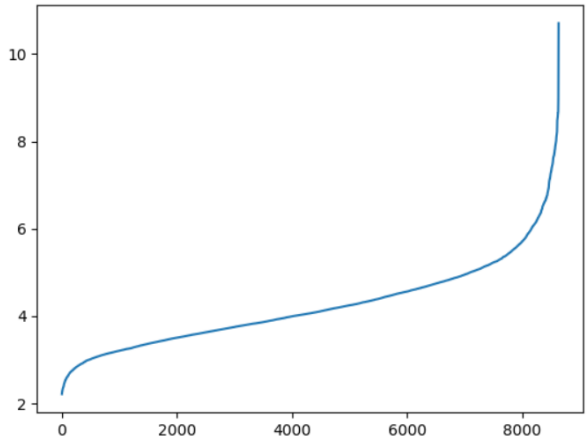
Since financial datasets are noisy, consist of outliers and fraud-related data points(countermeasures taken by criminals to avoid anti-fraud mechanisms), we need a clustering algorithm that is robust to noise, can handle outliers but is also adaptive at the same time like Ada Ellip was.

5.1 DBSCAN

DBSCAN : Density-Based Spatial Clustering of Applications with Noise is a possible algorithm choice that can find clusters of arbitrary sizes and shapes. It finds core samples of high density and expands clusters from them. A core sample is defined as being a sample in the dataset such that there exist min samples other samples within a distance of ϵ, which are defined as neighbours of the core sample.

The selection of its input parameters ϵ and min samples is of crucial importance to guarantee good clustering. In Generalized DBSCAN, Sander [8] suggested using the $(2 * dim - 1)$ nearest neighbor distance as ϵ and $minsamples = 2 * dim$. In our case dim is 17, so min samples is chosen as 34 .
For the calculation of ϵ, a popular technique in literature [9] involves calculating the average distance between each point and its k nearest neighbours, where k = the min samples value you selected. The kth-distances are then plotted in ascending order on a k-distance graph. The optimal value for ϵ is found at the

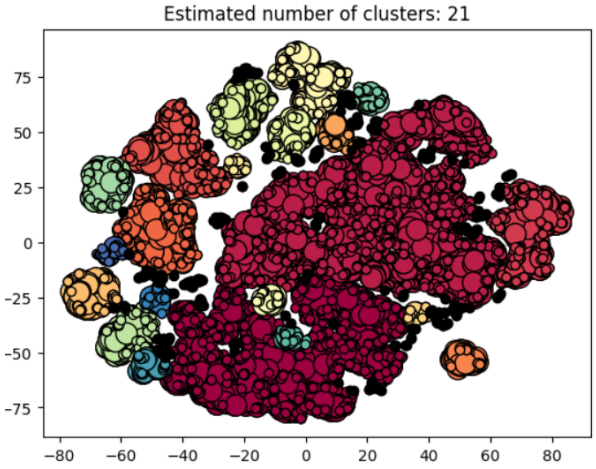
point of maximum curvature. We have done the same and found ϵ as 4.5 for our dataset.



DBSCAN

5.2 DBSCAN Output

DBSCAN output is as follows :



DBSCAN Output

We observe well-separable clusters with large diameter circles representing core samples of that respective cluster.

6 Conclusions

While we added adaptivity to K-Means with Ada-Ellip to automatically detect clusters, we further needed some mechanism to resist noise and cluster centroids being affected by outliers. DBSCAN is a good automatic clustering algorithm choice that is robust to outliers and we observed well separable clusters with outlier detection as well.
Finally we would like to conclude by saying that measurements from model activation is the real moment of truth. If the measurements are not showing up great, one should always re-run the model with different parameters, feature engineering and number of clusters !

7 References

1. T. Li, G. Kou, Y. Peng and P. S. Yu, "An Integrated Cluster Detection, Optimization, and Interpretation Approach for Financial Data," in IEEE Transactions on Cybernetics, vol. 52, no. 12, pp. 13848-13861, Dec. 2022, doi: 10.1109/TCYB.2021.3109066.

2. https://www.kaggle.com/datasets/arjunbhasin2013/ccdata

3. Visualizing Data using t-SNE, Laurens van der Maaten, Geoffrey Hinton

4. https://builtin.com/data-science/step-step-explanation-principal-component-analysis

5. Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.

6. U. Von Luxburg, "A tutorial on spectral clustering," Stat. Comput.,vol. 17, no. 4, pp. 395–416, 2007.

7. https://www.geeksforgeeks.org/singular-value-decomposition-svd/

8. Jörg Sander, Martin Ester, Hans-Peter Kriegel, and Xiaowei Xu. 1998. Density-based clustering in spatial databases: The algorithm GDBSCAN and its applications. Data Mining and Knowledge Discovery 2, 2 (1998), 169–194.

9. Erich Schubert, Jörg Sander, Martin Ester, Hans Peter Kriegel, and Xiaowei Xu. 2017. DBSCAN Revisited, Revisited: Why and How You Should (Still) Use DBSCAN.