Potential Loan Default Detection

Naiyi Zhang, Yongzheng Li

1 Introduction

In the realm of financial institutions, identifying potential loan defaulters is a critical task for ensuring the stability and profitability of lending portfolios. The primary question is how to effectively predict which borrowers are likely to default on their loans. This question is of immense interest to banks, credit unions, and other lending entities, as defaulting borrowers can lead to significant financial losses and disrupt the functioning of the institution. Therefore, developing robust models for detecting potential loan defaults is a priority in the financial sector.

Our project focuses on unsupervised machine learning to detect potential loan defaults by analyzing financial data patterns. The aim is to identify default probabilities for proactive risk mitigation. However, analyzing this data is challenging due to its multidimensional nature, making feature selection and model interpretation complex. Furthermore, evolving market dynamics add to the difficulty of building robust detection systems.

2 Dataset

2.1 Our dataset

Our dataset, accessible on Kaggle [1], originates from the International Institute of Information Technology Bangalore (IIIT-B). This dataset was collected as part of a social experiment conducted by IIIT-B to offer insights into the loan application process and associated risks. The dataset provides a comprehensive set of information related to loan applicants, including demographic information, credit card balances, payment history, and other relevant financial indicators.

2.2 Related work

Several studies have investigated financial anomaly detections using supervised and unsupervised machine learning techniques. Vasarhelyi & Issa [2] applied k-means clustering to detect fraudulent refund transactions from a telecommunication company, using SSE to select optimal clusters and detect fraud based on instance distance to centroid. Miloš et al. [3] used hybrid unsupervised techniques to detect tax evasion, employing k-means to find structure-based outliers and training autoencoder models on non-suspicious instances determined by experts, cross-checking results from both methods.

For supervised fraud detections, Xuan et al. [4] used random tree based random forest and CART-based random forest to classify normal and abnormal credit card transactions. Randhawa et al. [5] proposed majority voting methods with AdaBoost that combine 12 standard models including naive bayes, random forest, and support vector machines to detect credit card default. Vaishnavi [6] clustered customers based on amount and applied a sliding window method to aggregate the transactions to learn the behavior patterns.

We would like to find a similar dataset to detect abnormal loans of clients using unsupervised models. Unlike advanced techniques described above, we would experiment with a basic clustering based model and evaluate their performance.

2.3 Related implementation

In a notebook made by MANH [7], they selected features based on missing value < 500 and imputed the remaining missing value with mean/median. They applied label encoder to category variables and applied Logistic regression, KNN classifier, GaussianNB, Decision tree classifier to detect the outliers. Our method is better in feature selection and data preprocessing because we believe our selected features are closely related to the target.

In another notebook made by VAMSI [8], the author selected 8 features and imputed missing values with mean. They used boxplot to remove outliers and applied Random forest to detect fraud. Our method is better because we keep more features and we choose to standardize the feature instead of removing outliers if we do not have assumptions about our data.

3 Methods

In our analysis, we utilized three distinct unsupervised machine learning techniques, DBSCAN, K-means clustering, and CBLOF, to detect potential loan defaults within financial transaction data. These methods offer unique advantages in identifying anomalous patterns within the data without the need for labeled examples.

3.1 Data pre-processing & Analysis

We carefully selected 31 columns from our dataset of 182,811 rows, focusing on key features relevant to potential loan defaults. To handle missing values, we chose a pragmatic approach: dropping rows with missing entries. This maintains data integrity without introducing bias or skewing analysis, although other strategies like imputation could be considered.

	TARGET	NAME_CONTRACT_TYPE	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDREN	AMT_INCOME_TOTAL	AMT_CREDIT	AMT_ANNUITY	NAM
0	1	Cash loans	N	Y	0	202500.0	406597.5	24700.5	
1	0	Cash loans	N	N	0	270000.0	1293502.5	35698.5	
2	0	Revolving loans	Y	Y	0	67500.0	135000.0	6750.0	
3	0	Cash loans	N	Y	0	121500.0	513000.0	21865.5	
4	0	Cash loans	N	Y	0	99000.0	490495.5	27517.5	

Figure 1: Head rows of our data frame containing necessary columns.

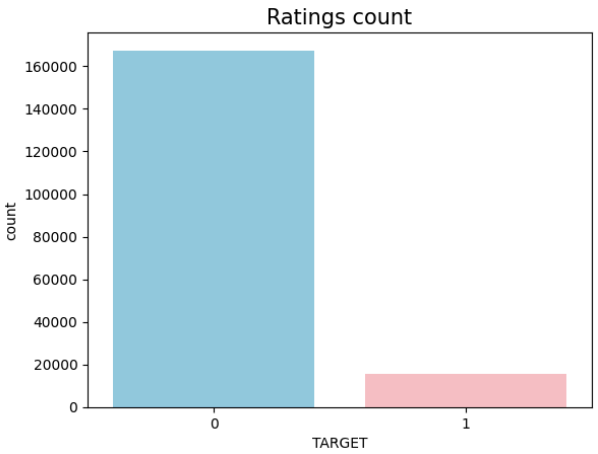


Figure 2: Target count where 1 indicates that the applicant has payment difficulties.

By visualizing the dataset, as shown in Figure 2, we noticed a contrast between the abundance of normal rows and the fewer instances of fraud. But in our approach with unsupervised fraud detection, it does not require a balanced dataset since our algorithms operate directly within the feature space, making class labels irrelevant.

Given the varying scales and distributions of numerical features, we standardized the numerical data using z-score normalization, which is calculated as:
$z = \frac{x - \mu}{\sigma}$ wher μ is the mean, and σ is the standard deviation of the dataset.
This involved scaling each feature to have a mean of zero and a standard deviation of one.

To handle categorical variables, we employed a mapping technique to convert textual values into numerical representations. By assigning unique numerical labels to each category within categorical columns, we facilitated the integration of these variables into our analysis without introducing ordinality or hierarchy.

Recognizing the potential for high dimensionality in the dataset, we applied Principal Component Analysis (PCA) to reduce the number of features while preserving the variance within the data.

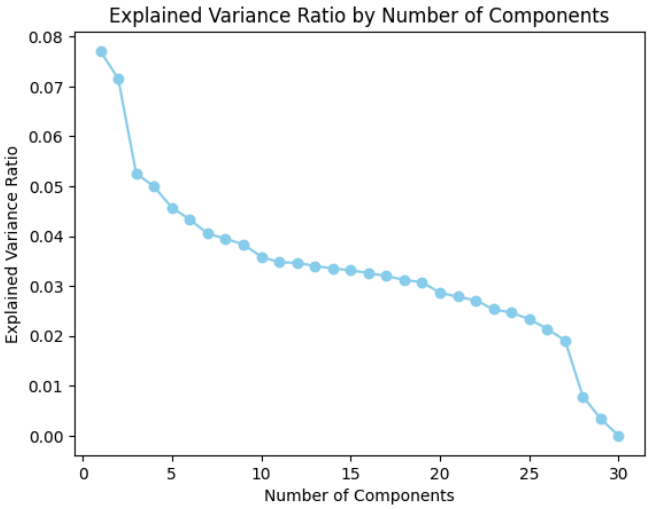


Figure 3: Explained variance ratio vs. number of components plot.

To determine the optimal number of components for PCA, we generated a plot illustrating the explained variance ratio versus the number of components, as shown in Figure 3. This plot showcases how much variance each principal component accounts for. Notably, a significant change in slope occurred between the second and fifth components. Consequently, we opted to use 5 principal components for our PCA.

3.2 DBSCAN

DBSCAN is a robust clustering algorithm known for its ability to detect clusters of various shapes and sizes in data, while also identifying outliers. Its key feature is handling clusters of arbitrary shapes and sizes without needing prior knowledge of the cluster count. This flexibility suits datasets with irregular shapes or inseparable clusters.

In addition to its flexibility in handling different cluster shapes, DBSCAN also excels in handling various cluster shapes and densities effectively. Unlike some algorithms, it doesn't

assume uniform density, making it great for datasets with varying density regions, like spatial data or unevenly distributed clusters.

DBSCAN's knack for automatically spotting noise points is handy for datasets with noisy or sparse regions. It sorts points into core, border, and noise categories, helping filter out irrelevant data and improving clustering accuracy. This feature is particularly useful for datasets with noise or outliers, allowing for better focus on meaningful patterns.

The success of using DBSCAN in our dataset stems from its ability to handle the inherent complexities present in the data, such as irregular cluster shapes, varying cluster densities, and noise. By effectively capturing the underlying structure of the data, DBSCAN facilitates the extraction of meaningful insights and patterns, which is crucial for any clustering task. These properties make DBSCAN a versatile algorithm that can be applied to a wide range of datasets across different domains.

DBSCAN operates by grouping together closely packed data points while marking those in sparse regions as outliers. Its core principle relies on the concept of density, where clusters are formed based on the density of points within a specified neighborhood. In DBSCAN, a core point is defined as a data point that has at least a specified number of points (MinPts) within a specified radius (ε). Border points are points that are within the neighborhood of a core point but do not have enough points surrounding them to be considered core points themselves. Noise points, also referred to as outliers, are points that are neither core points nor border points. They lie in low-density regions and are often discarded or considered as separate clusters.

The DBSCAN algorithm works in the following steps:

1. Initialization: Choose an arbitrary point from the dataset.
2. Clustering: Expand the cluster by adding all reachable points within the neighborhood defined by Eps and MinPts. (Mark each point as a core, border, or noise point.)
3. Iteration: Repeat the process for all core points and their connected components until all points have been assigned to a cluster or identified as noise.

The formula for DBSCAN's core density-based notion can be defined as follows:
$Core\ Density\ (p) = |\{q \in D : dist(p, q) \leq \varepsilon\}|$ and dist is typically the Euclidean distance.

We determined the minimum sample parameter by multiplying the shape of our PCA dataset (which is 5) by 2. This decision was based on empirical observations and a desire to ensure an adequate number of samples for the formation of dense regions. Next, we focused on tuning the epsilon parameter by conducting a systematic search over a range of epsilon values and evaluating the clustering performance at each step. Our evaluation criteria included metrics such as accuracy, F-1 score, and visual inspection of clustering results. Through iterative experimentation, we adjusted the epsilon parameter and evaluated two settings: 0.45 and 0.2. Each setting yielded different clustering performances.

3.3 K-means

K-means clustering is an unsupervised machine learning algorithm used for partitioning data into distinct clusters. The algorithm aims to minimize the within-cluster variance by iteratively assigning data points to the nearest centroid and updating the centroids based on the mean of the points in each cluster. Mathematically, the objective function of K-means can be represented as:

$J = \sum_{i=1}^{k} \sum_{x \in C_i} ||x - \mu_i||^2$, where k is the number of clusters, C_i is the ith cluster, μ_i is the centroid of the ith cluster.
The algorithm iteratively minimizes this objective function until convergence, resulting in a set of k centroids representing the cluster centers. We find the optimal number of clusters by using the elbow method where the point of rate of decrease in WCSS(within cluster sum of squares) slows down abruptly.

We assumed normal data instances belong to large and dense clusters, while anomalies do not. Under this assumption, K-means is a computationally efficient algorithm that groups similar loans into clusters and detects defaults that deviate from typical patterns. K-means is also useful to compare with other clustering methods.

To detect anomalies in K-means, points that are significantly farther away from the centroid than the majority of points in the same cluster can be considered anomalies. The distance threshold for considering a point as a default is determined using percentile. We choose 95 percentile as the threshold to balance Sensitivity and Specificity.

3.4 CBLOF

Cluster-Based Local Outlier Factor (CBLOF) is a clustering based algorithm used for detecting anomalies. It estimates the density for each cluster. CBLOF applies k-means to cluster the dataset. Then the algorithm split the resulting clusters into two categories: large and small. The calculation of the outlier score is accomplished by using the distance of each data point to the central data point of its cluster, multiplied by the data points that belong to its cluster. In a small cluster, the distance to the nearest large cluster is used.
The CBLOF score is represent as [9]:
if t in SC: $CBLOF = |C_i| * min(dist(t, C_j))$
where C_j belongs to LC, $|C_i|$ represent the size of cluster
if t in LC: $CBLOF = |C_i| * dist(t, C_i)$ where C_i belongs to LC

CBLOF identifies data points with outlier scores significantly higher than the average, indicating potential anomalies. Like K-means, CBLOF can effectively identify default loan information that deviates from normal loans if we assume non default loans belong to a large cluster. We choose 5 numbers of clusters, threshold set to 0.05 for CBLOF parameters.

4 Results

In assessing the performance of our algorithms for detecting potential loan defaults, we utilized several key evaluation metrics: accuracy, F1 score, precision, and recall. These metrics offer valuable insights into the model's effectiveness in classifying instances and detecting anomalies within the dataset.

The F1 score is a metric commonly used to evaluate classification models, especially in scenarios where the classes are imbalanced. It considers both the precision and recall of the model to provide a single score that reflects the model's overall performance, where precision measures the accuracy of the positive predictions made by the model and recall measures the ability of the model to capture all the positive instances in the dataset. The F1 score is calculated using the following formula: $F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}$ where

$Precision = \frac{TP}{TP+FP}$ and $Recall = \frac{TP}{TP+FN}$, while accuracy is calculated as the ratio of the number of correctly predicted instances to the total number of instances in the dataset.

4.1 DBSCAN

As we experimented with different values of our DBSCAN parameters, we noticed an increase in the accuracy and a decrease in the F1 score as the value of epsilon increases, as shown below.

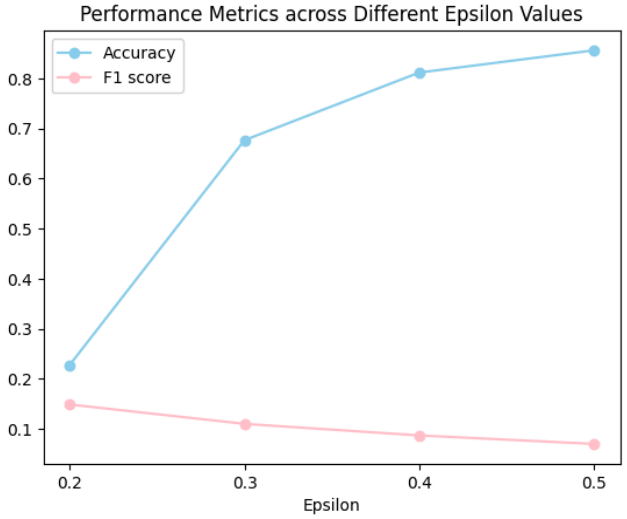


Figure 5: Variation in accuracy and F1 score across different epsilon thresholds.

After careful consideration, we have finalized our focus on two sets of parameters: setting the minimum samples to twice the dimensionality of the dataset and epsilon to 0.45 and 0.2. The adjustment of epsilon to 0.45 led to an impressive accuracy of approximately 84.89%. Anomalies, indicating potential loan defaults, were detected in the cluster labeled as -1, which contained 14276 instances. Moreover, our algorithm identified 154182 instances of non-default (true negatives) and accurately pinpointed 1000 instances of loan defaults (true positives). For the alternate epsilon value of 0.2, we have observed an accuracy of 22.75%. Despite the lower overall accuracy, this setting notably captured most of the loan defaults.

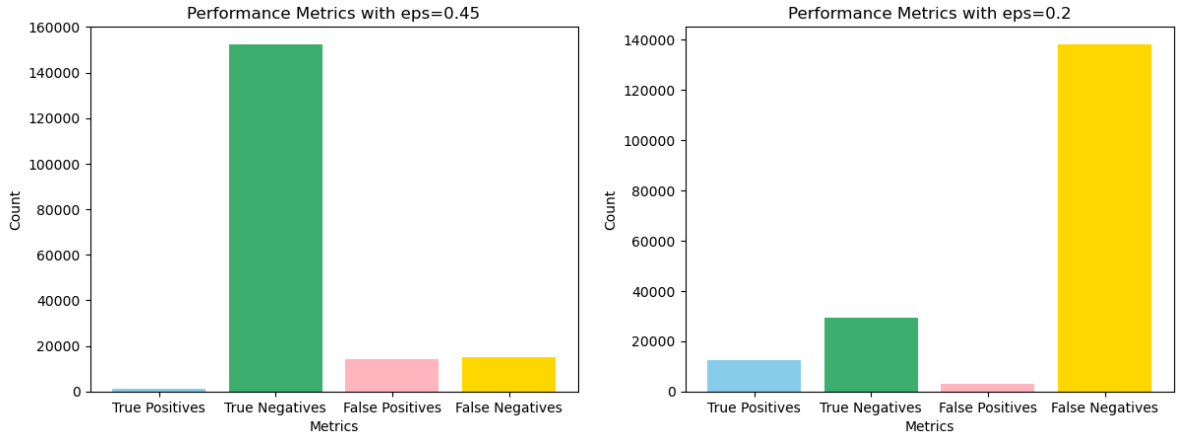


Figure 6: Performance metrics with finalized parameters

While DBSCAN is primarily employed for density-based clustering and does not inherently generate class labels that are conducive to F-1 score calculation like conventional classification models, our dataset features a TARGET column denoting whether an application poses a risk of loan default. Leveraging this, we classified outliers as potential loan defaults and all the remaining entries as normal applications.

	TP	TN	FP	FN	Accuracy	F-1 Score
eps=0.45	1195	152511	14158	14947	0.84	0.07
eps=0.2	12310	29287	3043	138171	0.23	0.15

Figure 7: Results of our DBSCAN algorithm.

4.2 K-means and CBLOF

After applying the elbow method for k-means clustering, k=6 is the elbow point. Using k=6 with a threshold of 95% percentile, we got precision = 0.068, recall =0.0404, F-1 score = 0.051. We only captured 620 fraud loans among 15353. For CBLOF, the result has slightly improved compared with K-means(precision = 0.092, recall =0.055, F-1 score = 0.069). Both results are worse compared with DBSCAN. It suggests that our data after PCA transformation does not have a natural clustering and we would better use other techniques for anomaly detection.

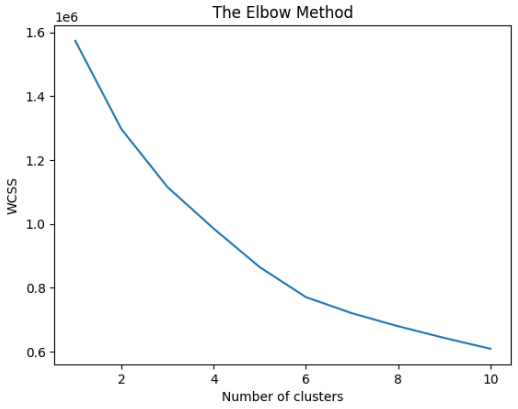


Figure 8: elbow method for K-means clustering

	TP	TN	FP	FN	F-1 Score
K-Means k=6	620	158937	8521	14733	0.051
CBLOF k=6	842	159159	8299	14511	0.069

Figure 9: K-means and CBLOF Result

4.3 F-1 scores

The F1 score of our algorithm was found to be low because while the model showed promising performance in identifying one performance metric, it struggled with other ones. Several factors may contribute to this observation:

1. Dataset quality: One potential reason for the discrepancy in performance metrics could be attributed to the quality of the dataset itself. It is plausible that inconsistencies or inaccuracies within the dataset affected the performance of our algorithm. We couldn't find many datasets online not processed through PCA transformation. However, we also aimed to demonstrate our ability to perform data standardization and PCA during our exploration.
2. Attribute selection: Another factor could be the attribute selection process. While we chose 31 attributes based on their perceived relevance to loan default prediction, it is possible that the omission of certain columns may have led to crucial information being overlooked. Furthermore, the choice to exclude certain columns due to computational constraints may have impacted the effectiveness of our algorithm. In future iterations, a more comprehensive evaluation of feature importance and experimentation with different attribute combinations could be explored to improve the performance of our model.

5 Discussion

According to a recent review of anomaly detection techniques[10], the trend has suggested researchers have adopted unsupervised and semi-supervised models to study anomaly in finance. The popular unsupervised model includes isolation forest, self-organizing maps and autoencoders. For semi-supervised models, HMM(hidden markov models) and GAN(generative adversarial networks) are the focused model. Many hybrid techniques are also under investigation(GAFCM:genetic algorithm based Fuzzy C-Means).

Our models do not achieve the similar outcome as the existing works. One reason is due to the methods we choose. The data does not fit well for clustering after PCA. Other unsupervised techniques might improve the outcomes like auto encoder and isolation forest. Also, we could choose supervised models to detect potential loan default like random forest and logistic regression. Another reason our models failed is due to the imbalance of data. We initially believe that imbalance data would not impact on unsupervised methods since these methods do not require labels. However, we need to further investigate the data structure and try SMOTE, undersampling to see if the outcome improved.

Our solution does not work well and needs more development. The low F1 score indicating both precision and recalls are low. The methods couldn’t capture enough loan default clients and those who are labeled as default clients are mostly innocent. We would suggest the desired F1 score > 0.5 for someone to use. Under different business cases, we would focus more on recall or precision.

References

[1] Mishra5001. (2019, July 15). Credit Card Fraud Detection. Kaggle. https://www.kaggle.com/datasets/mishra5001/credit-card/data

[2] Issa, Hussein and Vasarhelyi, Miklos A., Application of Anomaly Detection Techniques to Identify Fraudulent Refunds (August 16, 2011). Available at SSRN: https://ssrn.com/abstract=1910468 or http://dx.doi.org/10.2139/ssrn.1910468

[3] Miloš Savić, Jasna Atanasijević, Dušan Jakovetić, Nataša Krejić, Tax evasion risk management using a Hybrid Unsupervised Outlier Detection method, Expert Systems with Applications, Volume 193, 2022, 116409, ISSN 0957-4174, https://doi.org/10.1016/j.eswa.2021.116409.

[4] S. Xuan, G. Liu, Z. Li, L. Zheng, S. Wang and C. Jiang, "Random forest for credit card fraud detection," 2018 IEEE 15th International Conference on Networking, Sensing and Control (ICNSC), Zhuhai, China, 2018, pp. 1-6, doi: 10.1109/ICNSC.2018.8361343.

[5] Randhawa, Kuldeep, et al. “Credit Card Fraud Detection Using AdaBoost and Majority Voting.” IEEE Access, vol. 6, 2018, pp. 14277–14284., doi:10.1109/access.2018.2806420.

[6] Vaishnavi Nath Dornadula, S Geetha, Credit Card Fraud Detection using Machine Learning Algorithms, Procedia Computer Science, Volume 165, 2019, Pages 631-641, ISSN 1877-0509, https://doi.org/10.1016/j.procs.2020.01.057.

[7] Mnhcngbi (2023) *Fraud_detection*, *Kaggle*. Available at: https://www.kaggle.com/code/mnhcngbi/fraud-detection (Accessed: 23 April 2024).

[8] Vamsichennakesava (2022) *Credit Card Fraud Detection*, *Kaggle*. Available at: https://www.kaggle.com/code/vamsichennakesava/credit-card-fraud-detection (Accessed: 23 April 2024).

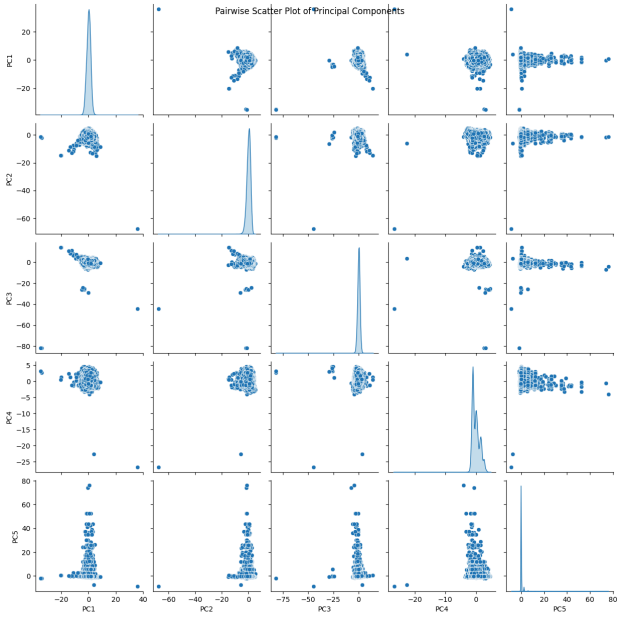
[9] He, Z., Xu, X. i Deng, S. (2003). Discovering cluster based local outliers, Pattern Recognition Letters, 24 (9-10), str. 1651-1660

[10] Waleed Hilal, S. Andrew Gadsden, John Yawney, Financial Fraud: A Review of Anomaly Detection Techniques and Recent Advances, Expert Systems with Applications, Volume 193, 2022, 116429, ISSN 0957-4174, https://doi.org/10.1016/j.eswa.2021.116429.

Appendix A: Code Repository

https://github.com/Tiffanyxk3/PotentialLoanDefaultDetection

Appendix B: Pairwise scatter plot of principal components



The pairwise scatter plots of the components we generated using PCA show the distribution of variance, aiding our analysis.

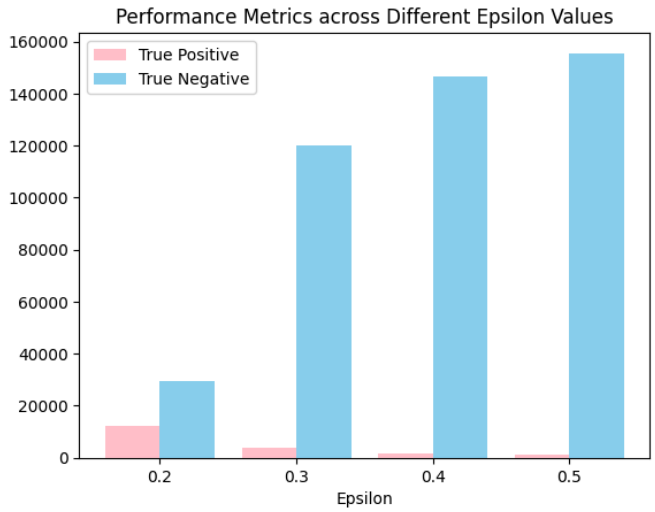
Appendix C: DBSCAN Results across Different Parameters

C.1: Table demonstrating the numerical results of DBSCAN

	TP	TN	FP	FN	Accuracy	F-1	# clusters
eps=0.2	12310	29287	3043	138171	0.23	0.15	344
eps=0.3	3642	120099	11711	47359	0.68	0.11	128
eps=0.4	1630	146819	13723	20639	0.81	0.09	47
eps=0.5	983	155587	14370	11871	0.86	0.07	26

This table showcases DBSCAN results across different parameters. The table includes true positives, true negatives, false positives, false negatives, accuracy, F-1 Score, and the number of clusters generated for varying epsilon values. As epsilon increases from 0.2 to 0.5, the number of clusters decreases while accuracy improves, indicating the impact of parameter selection on DBSCAN's clustering performance.

C.2: Plot demonstrating the differences in TP and TN



The number of true positives decreases and the number of true negatives increases as the value of epsilon increases.

Statement of Contributions

All group members contributed equally to this project. Below is a summary of each member's contributions:

- Naiyi Zhang: Worked on data pre-processing, feature extraction, and implemented the DBSCAN algorithm.
- Yongzheng Li: Implemented the K-means algorithm and CBLOF (Cluster-based Local Outlier Factor). Worked on related works and discussion sections.