Anomaly Detection For Credit Card Transaction

1st Felicia Janice Pranoto
Computer Science
BINUS University
Tangerang, Indonesia
felicia.pranoto@binus.ac.id

2nd Kelila Karenza Kumala
Computer Science
BINUS University
Tangerang, Indonesia
kelila.kumala@binus.ac.id

3rd Rafael Zefanya Jaya Surya
Computer Science
BINUS University
Tangerang, Indonesia
rafael.surya@binus.ac.id

I. Introduction

Credit Card fraud has been a prevalent problem everywhere in the world. According to recent reports, by 2026 the estimated global credit card fraud will reach a total of 43 Billion US Dollars. So evidently, the rate of which global credit card fraud counts has been increasing drastically in the past few years. This graph below shows the increase of global losses from credit card fraud each year.

Which is why it has become important for banks to be able to detect fraudulent credit card transactions quickly and accurately. Before the existence of machine learning models which are widely used today to quickly detect any anomalies within transactions, they had to manually detect credit card fraudulent transactions by either using predefined ruleset to filter the normal transactions from the anomaly transactions, or they had to manually review one transaction after another. The older methods obviously took a lot of time and effort, and were very inefficient. However, with the rising popularity and performance of machine learning models, and their ability to cluster or classify transactions in order to detect fraudulent transactions, these models have been widely used by banks and financial institutions to help detect anomalies.

Our project will include experimenting on three different algorithms that will detect fraudulent transactions from publicly available datasets on kaggle, and see which model performs the best, and how different preprocessing methods benefit each of the algorithms. The algorithms we will be using for this project are Isolation Forest, SVM, and K-Means Clustering. We will then evaluate each model using standard metrics such as accuracy, f1 score, etc.

II. Theoretical Basis

A. Anomaly Detection

An anomaly is a deviation that distinguishes itself from others. Therefore, anomaly detection refers to the processes of identifying unusual patterns or outliers in a dataset. The primary goal is to recognize data points that significantly deviate from the majority of data instances.

There are multiple algorithms that can be used for anomaly detection such as isolation forest, local outlier factor, support vector machines, k-means, angle-based outlier detection, etc. As mentioned previously, in this paper we will focus on the three algorithms: isolation forest, support vector machines, and k-means.

B. Isolation Forest

Isolation forest is one of the most popular algorithms used for anomaly detection. It is an unsupervised method meaning that it does not require labeled data and is known for being fast and efficient with low overhead. Isolation forest identifies anomalies by recursively partitioning the dataset which allows it to detect data that deviates from the norm. What makes this method unique is that it uses random partitioning instead of proximity measures that traditional methods use.

Random partitioning is performed by selecting a feature at random to split the data, as the name suggests. The algorithm will then select a random threshold within the range of the data and divide the dataset into subsets. The first subset contains data points less than or equal to the threshold while the second subset contains data points greater than the threshold. The process of random partitioning continues recursively until each leaf contains only one data point or a predefined stopping point is reached. The predefined stopping point depends on the requirements set by the user. Splitting can stop when a maximum tree depth is reached or a minimum number of data points per subset is obtained.

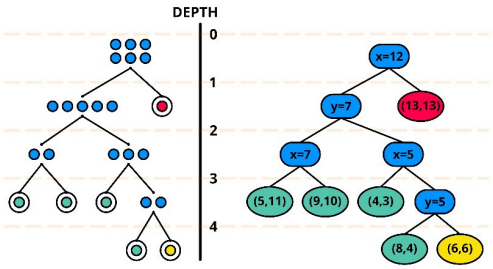


Fig. 1: Predefined stopping with a minimum 2 data per subset

As the name suggests, this method creates multiple trees each with different selected features used to split the dataset, creating a “forest”. Having multiple trees introduces randomness and ensures that the model generalizes better, preventing over-fitting to a specific data or pattern. It also provides a more stable and reliable measure of how isolated a point is by averaging the path length across all trees. Multiple trees allow the model to include all the features from the dataset and detect diverse data patterns.

Anomalies are identified by using anomaly scores. The anomaly score formula is defined as follows:

$$s(x,n) = 2^{-\frac{E(h(x))}{c(n)}} \tag{1}$$

where E(h(x)) represents the path length and c(n) represents the normalization constant for data with the size of n. Path length refers to the number of edges or splits required for a data point to be isolated. The lower the value of the path length, the higher the probability that the data point is an anomaly. This is because it takes fewer splits to isolate an anomaly as it is quite separated from the norm. The normalization constant is required to adjust the path length for a dataset. It represents the average number of splits required to isolate a point at a random tree. The formula for c(n) is defined as follows:

$$c(n) = \begin{cases} 2H(n-1) - \frac{2(n-1)}{n}, & \text{if } n > 2 \\ 1, & \text{if } n = 2 \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

The anomaly score s(x,n) states that the shorter the path lengths, the higher the anomaly score. A higher anomaly score means a higher chance of the data point being an anomaly.

C. Support Vector Machine (SVM)

Support vector machine is a supervised machine learning method used for both classification and regression, though it is best used for classification. SVM focuses on finding the optimal hyperplane for separation between data points, dividing them into classes. It ensures that the distance between the nearest points of each class, vector points, is maximized. This is referred to as hard margin. In some cases, a dataset could contain overlapping classes, which would be a struggle for hard margin. To overcome this limitation, soft margin could be applied. Soft margin allows for some misclassifications with certain conditions. Slack variable is used to quantify the extent to which a data point can violate the margin constraints. The total sum of all slack variables of the misclassifications is called the classification error. Therefore, the goal of soft margin is to not only maximize the margin between the closest data points, but to also minimize the classification error.

The dimension of the hyperplane depends on the number of features. If the features increase, the dimension of the hyperplane increases as well. What makes SVM unique is that it can map data points as if they are in a higher dimension using the kernel trick, enabling it to classify data that is not linearly separable in the original dimension. The kernel trick works by applying a kernel function and calculating the dot product of two data points to simulate the effect of computing dot products of the data points in a higher dimension. The dot product measures how similar two data points are. SVM uses this to calculate distances and find the optimal decision boundary. The "trick" of this method is that SVM does not actually map the data points to a higher dimension as that would be computationally expensive and time consuming. Instead, it simply calculates the equivalent result in the original space as if the data points are in the higher dimensional space by encoding the effects of the transformation to a higher dimension on the data points. The kernel function is responsible for this effect. There are several types of popular kernel functions:

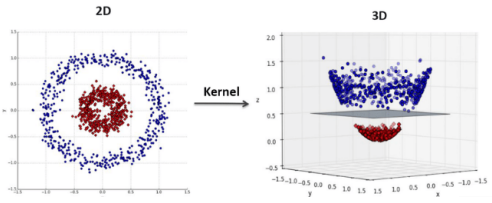


Fig. 2: SVM's kernel trick

1) *Polynomial Kernel*
   A non-linear kernel function that applies polynomial functions to transform data points to a higher dimension. Polynomial kernel uses the following function:

$$K(x_1, x_2) = x_1 x_2 + c)^d \tag{3}$$

   where x_1 and x_2 are the data points (vectors) and c is the constant that shifts the input vectors before raising them to the power of d.
2) *Gaussian (RBF) Kernel*
   A non-linear kernel function, also known as radial basis function, that applies Gaussian function to transform data points to a higher dimension. This method can capture complex, non-linear relationships between inputs. The Gaussian kernel uses the following function:

$$K(x_1, x_2) = exp(-\frac{||x_1 - x_2||^2}{2\sigma^2}) \tag{4}$$

   where x_1 and x_2 are the data points (vectors), σ is the parameter that controls the spread of the Gaussian function, and $||x_1 - x_2||^2$ is the Euclidean distance between the vectors squared.
3) *Laplacian Kernel*
   A non-parametric kernel function, also known as exponential kernel, that applies exponential functions to transform data points to a higher dimension. This method is similar to the Gaussian kernel but differs in the distance measure. The Laplacian kernel uses the following function:

$$K(x_1, x_2) = exp(-\frac{||x_1 - x_2||_1}{\sigma}) \tag{5}$$

   where x_1 and x_2 are the data points (vectors), σ is the parameter that controls the spread of the Gaussian function, and $||x_1 - x_2||_1$ is the Manhattan distance (L1 norm) between the vectors.

As high dimensions do not limit SVM, it is suitable for handling datasets with many features. It is less prone to overfit and handles imbalanced data well. With its objective of finding an optimal hyperplane to separate data points, this method is fitting for anomaly detection. Utilizing the

kernel function allows for non-linear separation of outliers and without high computational cost. On top of that, SVM outputs clear decision boundaries which provide information on why or which feature causes a data point to be classified as an anomaly.

D. K-Means Clustering

K-Means clustering is an unsupervised machine learning algorithm that clusters data points based on their distance to the centroids of the clusters. Data points of the same cluster would most likely be more similar to each other than other clusters. The value k represents the number of clusters that the dataset would be divided into. This is usually initialized in the beginning of the calculation process. A higher k value means smaller clusters with higher detail while a lower k value means larger clusters with lower detail. K-Means clustering is categorized as an exclusive clustering method, meaning that each data point can only belong to just one cluster. This method is an iterative process to minimize the sum of distances between data points and their cluster's centroid.

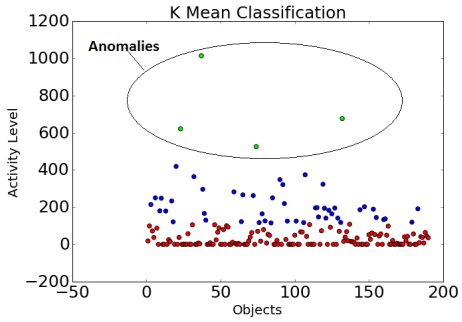


Fig. 3: K-Means clustering for anomaly detection

Initially, the centroids are randomly selected from any points in the dataset. The Euclidean distance of each data point is then calculated to each of the centroids. The data point would then be categorized into the cluster whose centroid has the smallest distance to it. The next iteration is calculated with a new centroid which is taken from the mean or the median value of all the data points within that cluster. This process will keep repeating until there are no changes to the centroids.

K-Means clustering is very simple to implement, but it is very sensitive to outliers. Hence, it is crucial to carefully select the appropriate number of clusters. This can be done using the elbow method. This method chooses the optimal number of clusters based on where the change in "within cluster sum of squares" (WCSS) levels off. WCSS assesses how similar data points are within a cluster. It sums up the squared distance of each data point to the cluster centroid. The lower the WCSS value, the more similar the data points are within a cluster.

K-Means clustering is a suitable method for anomaly detection, especially because it performs well on unseen data. Anomalies would typically have greater distances to cluster centers than other data points or do not belong in any cluster at all. Using this method is also beneficial because it is very simple to implement and interpret. It is also scalable and can handle large datasets well.

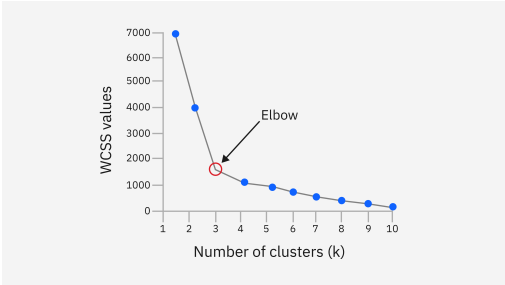


Fig. 4: Implementation of the elbow method

III. Methodology

Our proposed workflow consists of Exploratory Data Analysis (EDA), data preparation, data mining, model evaluation, and knowledge presentation.

A. Exploratory Data Analysis

In this study, two datasets, Dataset 1 and Dataset 2, will be merged before continuing to the training and testing step. Both datasets used are taken from Kaggle. The data set contains details of credit card transactions that will be explored in more detail in this section and a column that indicates whether the transaction is fraud or not.

1) Dataset 1 (Credit Card Transactions Dataset): Our first dataset contains a total of 1,296,675 rows of data describing transactions using credit cards, full of information and attributes such as the transaction amount, the credit card number, the merchant, the purchase category, and more with a total of 24 columns of attributes, including the label that describes whether the transaction is fraudulent or not. This dataset doesn't contain any duplicated data, but it contains a total of 195,973 rows that yields null values in one of the attributes. Important attributes that will be processed later are categorical data that needs to be encoded into numerical data, as well as an attribute that describes the date and time of the transaction.

2) Dataset 2 (Credit Card Fraud data): The second dataset contains a total of 14,446 rows of data similar to the first dataset, describing transactions using credit cards with a total of 15 columns, with attributes that intersect with the first data like merchant, purchase category, amount, but also contains attributes that doesn't intersect with the first dataset. This dataset also contains a label attribute that describes whether a transaction is fraudulent or not. this dataset contains a total of 63 rows of duplicated data, but doesn't contain rows that yield null values. Similar to the first dataset, important attributes that we will process later are categorical data, as well as the transaction date and time attribute.

After pre-processing the data, we will analyze the correlation between attributes to find out which attribute contributes most to potentially determining whether a transaction is fraudulent or not. We will try to find any potential patterns between attributes and comparing them between non-fraudulent and

fraudulent data. Later on, we will then compare the performance of the algorithms when given all of the attributes, and when we give only the selected attributes, to see which method of data selection performs better.

We will explore different analyses on the attributes of the combined data set.

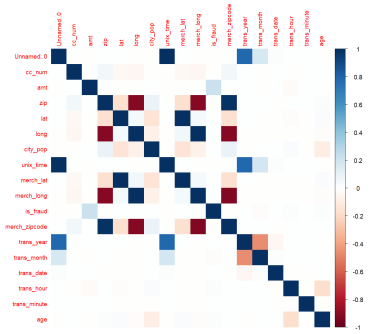


Fig. 5: Heatmap of the first dataset

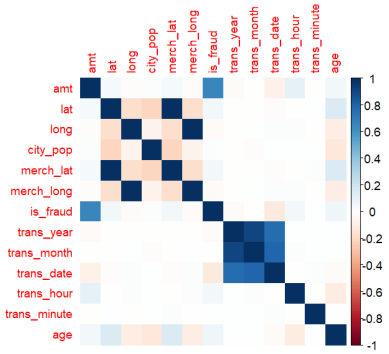
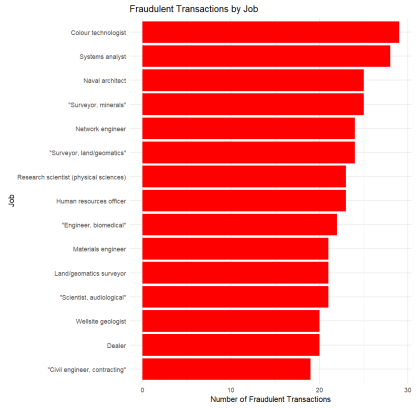


Fig. 6: Heatmap of the second dataset

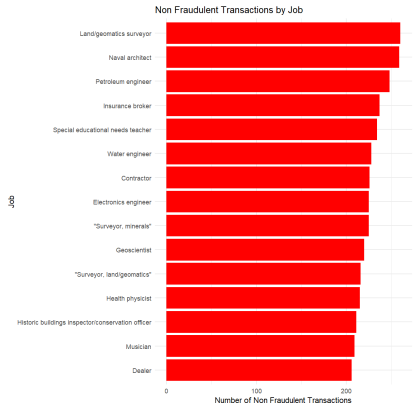
Figure 5 and 6 shows the correlation heatmap of both datasets. We can see that in the first dataset, the most notable fact is that in both first and second dataset, amt is fairly positively correlated to the `is_fraud` label, other than that, in the second dataset, we can see that `trans_date` is slightly negatively correlated to `is_fraud`, and `merch_lat` and `lat` is slightly positively correlated to `is_fraud`.

Figure 7a and 7b shows the top frequencies of jobs for fraudulent, and non-fraudulent transactions. We can see that there is a clear distinction between jobs that appear in fraudulent and non-fraudulent transactions. Thus, we can assume some jobs yield higher "risk" of a transaction being fraudulent.

Figure 8 shows the frequency of trasnsactions per hour, grouped by fraudulent and non-fraudulent transactions. We can see that the frequency of transactions for the non-fraudulent class starts of high from midnight, and the frequency stays until the afternoon, where the frequency rises quite considerably where it keeps that rate up until midnight. This pattern shouldn't be a surprise since most people would be more active in the afternoon, and we would have expected the frequency to rise around that time.



(a) Job frequency by fraud transactions



(b) Job frequency by non-fraud transactions

Fig. 7: Job Frequency for fraud vs non-fraud transactions

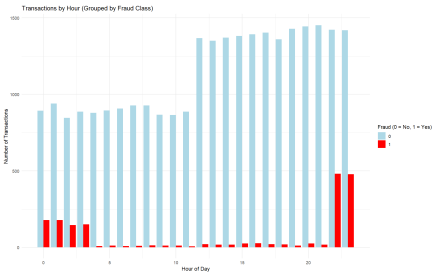


Fig. 8: Transactions frequency per hour based on fraudulent and non-fraudulent

On the other hand, when we look at the fraudulent transactions frequency, we can see that there are some transactional activities starting from midnight, but the frequency considerably drops starting from around 5 or 6 in the morning, and we see this constant absence of transactional activity until 11 to 12 at night, where we see significant rise of transactions frequency, almost doubling the previous rate. We can draw a very clear conclusion that fraudulent transactions are mostly done either very early in the day, or very late at night. Which means the hour of which the transaction takes place may determine the "risk" of that transaction being fraudulent.

Figure 9 shows the density of transactions based on the

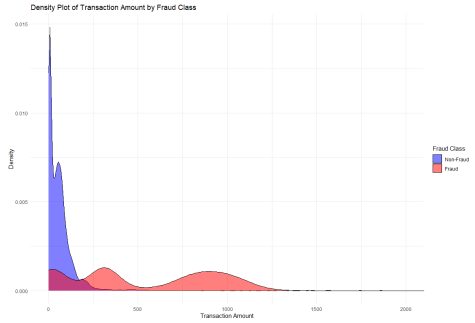


Fig. 9: Density of transactions based on amount, grouped by fraudulent and non-fraudulent transactions

amount, divided by fraudulent and non-fraudulent data. We can see that the non-fraudulent transaction has a particularly small deviation, and it peaks around the lower amount number, it has outliers for higher amounts but not significant enough. Meanwhile, if we see the density for fraudulent transactions, we can see that it has three peaks, two is closer to the lower amount, while the last one peaks at around 800-1000, which is a very distinct difference compared to the non-fraudulent transactions. We can draw a conclusion that fraudulent transactions may yield a higher transaction amount compared to non-fraudulent transactions, and this attribute is quite important in determining potentially fraudulent transactions.

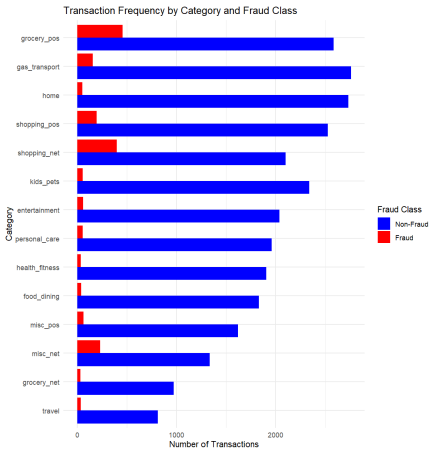


Fig. 10: Total transactions based of categories grouped by fraudulent and non-fraudulent

Figure 10 graphs as a form of barplot the number of transactions based on the category of transaction, grouped by fraudulent and non-fraudulent transactions. We can see that the fraudulent transactions resulted in a different distribution than the non-fraudulent transactions, and we can see that `grocery_pos`, `shopping_net`, and `misc_net` has the highest total transactions for fraudulent transactions. Thus, we can conclude that the category of transaction may potentially help in determining whether a transaction is fraudulent or not.

B. Data Preparation

1) Data Cleaning: Both datasets contains several `NULL` values as well as duplicated data. In this data cleaning step, we remove all missing values and duplicates to obtain a clean dataset.

2) Data Integration: We integrate the two datasets to create one large dataset for our data mining step. This allows us to identify more intricate patterns and reduces overfitting. In order to obtain balanced information from both datasets, we took a sample of 15000 data from each dataset and combined them to get a new total of 30000 rows of data. This is done to prevent any dataset from overshadowing the other, considering the total rows in Dataset 1 is far more than in Dataset 2.

3) Data Transformation: Transformation of data is a crucial step to make the values interpretable by our anomaly detection algorithms. We conducted frequency encoding for the `job`, `merchant`, and `city` columns to convert the data to numerical data type. Next, the `trans_date_trans_time` column was separated into separate columns for `date` and `time`. `dob` was also converted to numerical values by subtracting date's the year from the current year. One-hot encoding was used to convert `category` and `state` using the library `fastDummies`.

4) Data Selection: After transforming our combined dataset into numerical values, we dropped several features that has a low correlation to identifying anomalies. These includes the `cc_num`, `first`, `last`, `gender`, `street`, `zip`, `unix_time`, and `merch_zipcode` columns. We will then compare the performance of our algorithms when we use all the attributes with when we use only the selected data to see which method performs better, and to see whether selecting certain attributes is actually necessary.

C. Data Mining

1) Isolation Forest: The isolation forest algorithm can be automatically implemented using the `isotree` built-in package in R. Once imported, we can directly use the `isolation.forest` function to create our isolation forest. We created a total of 1000 trees with sample size 256 per tree and a dimension value of 1 which means that our trees split the dataset based on 1 feature. We also set a parallel thread value of 4 trees, considering our CPU processing power.

The anomaly scores of each data point are calculated using the `predict` function of the package. We determined the threshold as the 80th percentile of the anomaly scores. Any value that falls above the 80th percentile of the anomaly scores would be classified as an outlier. We used binary values to classify normal and anomalous data with values 0 and 1, respectively.

2) SVM: Support Vector Machine (SVM) is a machine learning algorithm that is widely used for linear and non-linear classification, numerical regression, as well as outlier tasks. For our purposes, we will use SVM to detect any outlier or anomalies in our credit card transaction dataset. SVM works by attempting to separate data into their corresponding classes, and it separates them by using a hyperplane. The hyperplane

depends on the dimension of the dataset, for example if we have two dimensional dataset, then the hyperplane would just be a line. If the dataset is three dimensional, the hyperplane would be a 2d shape. Because of the usage of this hyperplane, SVM is still very effective even when working with data with high dimensionality. So compared to other algorithms, SVM can better resist the curse of dimensionality, making it superior to other algorithms in that sense.

For our purpose, we chose SVM to be one of the algorithm to do the anomaly detection task because we are working with a dataset with quite high dimensionality, especially after preprocessing the data. So we expect high performing results for this algorithm. We will train the model as well as testing it with new unseen data to ensure the model is performing well (Not underfit or overfit), so we will split the dataset by 70% for the training data split, and 30% for the testing data split. The preprocessing steps will be the exact same for all three algorithms, and at the end, we will evaluate the model, as well as comparing between using a select few attributes based on our EDA as well as using all the attributes to see which method yields a higher result.

The model will then train based off the given dataset, and it will attempt to separate the data based of the class (in our case fraud and non-fraud) using hyperplanes. After the model has trained, we can then run the trained model on the unseen test dataset split to evaluate the model's performance.

3) K-Means Clustering: The K-Means Clustering algorithm can be implemented using a built-in package in R, namely `stat`, `ClusterR`, and `cluster`. The `kmeans()` function from the `stat` package is used to perform K-Means clustering. To determine the optimal number of centers (k), the elbow method was applied. The resulting graph visualization of the elbow method can be further seen in Figure 11.

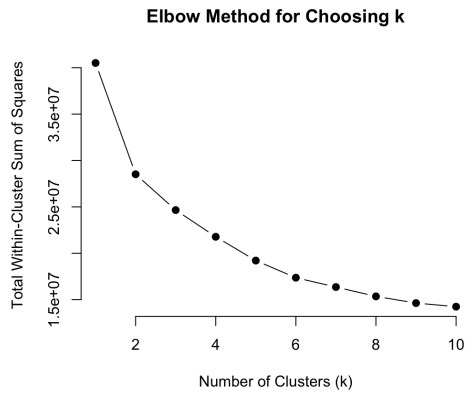


Fig. 11: Elbow Method

From the graph, the ideal number of centers can be either 2 or 3, but we chose 2 because we decided that it is the 'elbow' of the curve. After training the model, ensure that the number of dimensions of your dataset match with K-Means' model clusters before calculating the distance of each data point (row) to each cluster center. Similar to the isolation forest model, the threshold was determined as the 80th percentile of the distances. Data points with distances greater than the threshold are detected as anomalies. Binary values are used to classify normal and anomalous data with values 0 and 1, respectively.

D. Model Evaluation

1) Confusion Matrix: Confusion matrix is an N x N matrix used for evaluating the performance of a classification model, where N is the number of target classes. This helps us better visualize the errors while mapping out its type. There are 4 possible outputs in a confusion matrix.

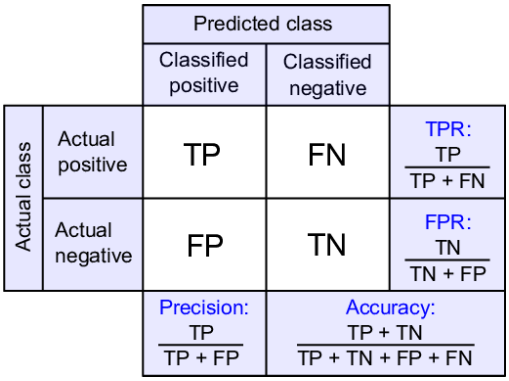


Fig. 12: Confusion Matrix

- True Positives : When the actual value is positive and the model also predicts positive
- True Negatives : When the actual value is negative and the model also predicts negative
- False Positives (Type-I error) : When the actual value is negative but the model predicts positive
- False Negatives (Type-II error) : When the actual value is positive but the model predicts negative

The results of our confusion matrix are used to calculate other evaluation metrics, namely accuracy and recall score.

2) Accuracy: Accuracy measures how often the classifier makes the correct prediction. It's the ratio between the number of correct predictions and the total number of predictions. The equation for accuracy is as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{6}$$

3) Recall Score: Recall, also known as sensitivity, is a measure of actual observations which are predicted correctly. It calculates how well our model can correctly predict positive instances out of all actual positive samples in the dataset. The higher the recall value, the better the model performance. Recall score can be computed using the following formula:

$$Recall = \frac{TP}{TP + FN} \tag{7}$$

IV. Result

A. Isolation Forest

The anomaly detection on our dataset using isolation forest resulted in an accuracy of 80%. This value is considerably

good considering the abstractness of our combined dataset, especially with the specificity score being at 0.82.

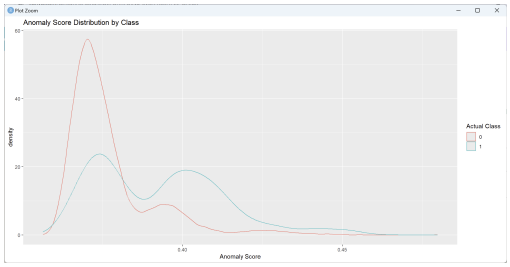


Fig. 13: Anomaly Score Distribution by Class

The density graph above shows the anomaly score distribution for normal transactions and anomalous transactions, marked as red and blue respectively. The normal transaction graph is skewed right with a low anomaly score which shows that our model correctly classifies normal transactions. On the other hand, the anomalous transaction graph has peaks with a higher anomaly score which also proves that fraudulent transactions are correctly classified.

TABLE I: Confusion Matrix

	Negative	**Positive**
Negative	22683	821
Positive	4,837	1040

However, the recall score lies at a low 50%. This low performance is caused by two reasons. The first is due to the fact that all the features in the dataset are chosen to perform random partitioning while not all of them contribute to whether a transaction is considered an anomaly or not. Because of this, we decided to conduct a second calculation using only features that correlates to the `is_fraud` column. These features are chosen based on our Exploratory Data Analysis process where we discovered that `job`, `trans_hour`, `amt`, and `category` highly affects anomalous transactions.

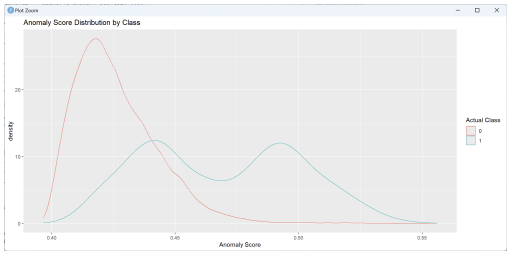


Fig. 14: Anomaly Score Distribution by Class based on selected features

After running the model again with just 4 highly-correlated features, the recall score improved drastically to 73% and the accuracy increased to 83%. The difference between normal and fraudulent transactions in the new density graph is also much more distinct with anomalous data receiving much higher anomaly scores than before.

TABLE II: Updated Confusion Matrix

Predicted	**Negative**	**Positive**
Negative	23,015	489
Positive	4,505	1,372

The second reason is because we are using a combination of 2 different datasets that may lead to some inconsistencies. Each dataset has their own patterns which might not compliment the other dataset. To prove this theory, we performed isolation forest on each dataset individually.

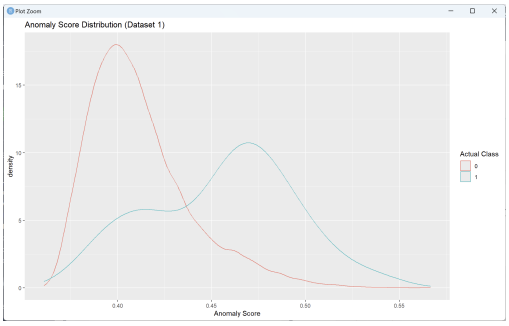


Fig. 15: Anomaly Score Distribution for Dataset 1

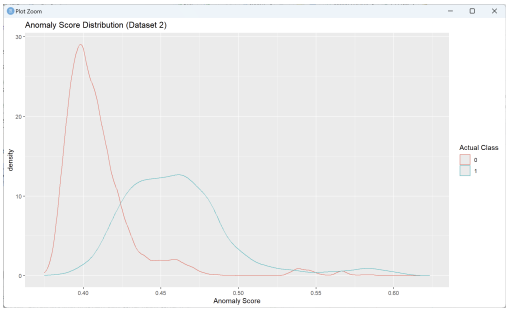


Fig. 16: Anomaly Score Distribution for Dataset 2

The resulting density graphs in Figure 15 and Fig 16 display an excellent separation between normal and anomalous transactions for, both, Dataset 1 and Dataset 2 with the anomaly scores of the fraudulent transaction being much higher than than normal transactions. Hence, this verifies our analysis that the combination of 2 different datasets might impact the patterns and insights identified for anomaly detection. Moreover, this further proves that our isolation forest model is highly compatible and performs fittingly.

B. SVM

Training the SVM model with all attributes of the combined dataset, then testing it with the unseen test data, resulted in a high 97% accuracy, as well as specificity of 99%, although the sensitivity metric falls behind with only 60%. Table III shows the confusion matrix of the SVM's model by training it with all attributes of the dataset.

We will then compare the result of the SVM model when we train it using the selected attributes, and the result shows

TABLE III: Confusion Matrix SVM

Predicted	Negative	Positive
Negative	8238	221
Positive	13	341

an accuracy of 98%, and a significant increase in sensitivity, resulting in 75%, with specificity yielding the same result of 99%. This shows that selecting the proper attributes by analyzing the data through EDA can yield better performance for the algorithm, and this improvement stays consistent throughout our three tested algorithms. Table IV shows the confusion matrix for the newly trained model.

TABLE IV: Updated Confusion Matrix SVM

Predicted	Negative	Positive
Negative	8225	139
Positive	26	423

C. K-Means Clustering

Anomaly detection using the K-Means model with all attributes of the combined dataset resulted in an accuracy of 77% with a sensitivity of 26% and a specificity of 80%. The density graph on Figure 17 shows the distance distribution by class, with class 0 or the red graph as the normal transaction and class 1 or the blue graph as the anomalous transactions. From the graph, we can see that there is an overlap between the two classes. This indicates that both anomalous and normal transactions have similar distances to their nearest clusters, making it more difficult to differentiate the type of transactions based on their distance.

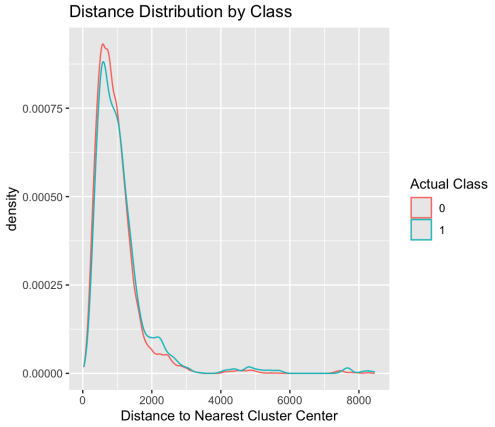


Fig. 17: Anomaly Score Distribution by Class

Table V shows the confusion matrix for the K-Means model by training it with all of the attributes from the combined dataset.

TABLE V: Confusion Matrix K-Means

Predicted	Negative	Positive
Negative	22,120	1,384
Positive	5,400	476

Another training of the K-Means model is done using a dataset with selected attributes. The result shows that there is a slight improvement in accuracy, increasing from 77% to 79%, a significant improvement in sensitivity, reaching 40%, and a slight increase in specificity, reaching 81%. The density graph in Figure 18 illustrates reduced overlap between the two classes. This suggests that training the K-Means model using the dataset with selected attributes allows for a more distinct distribution between anomalous and normal transactions, thereby improving the model performance in detecting anomaly transactions.

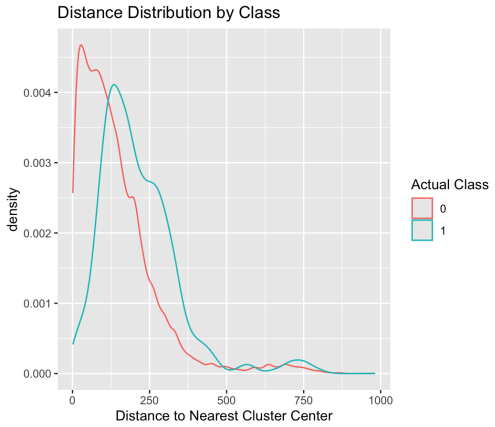


Fig. 18: Anomaly Score Distribution by Class based on selected features

Table VI shows the updated confusion matrix of the re-trained K-Means Model.

TABLE VI: Updated Confusion Matrix K-Means

Predicted	Negative	Positive
Negative	22,396	1,108
Positive	5,124	752

V. Conclusion

In this project, we have experimented with three different algorithms in order to detect anomalies in credit card transactions data. After exploring different EDA on different attributes, we have found that certain attributes have higher correlation with the fraud label from the dataset, with the help of correlation matrix. However, outside of correlation matrix, we also found other attributes that may be important in determining potential fraud transactions, by analyzing any patterns when we group them based on the fraud class.

Out of the three algorithms, SVM yields the highest accuracy, this may be explained by the fact that SVM is a more complex algorithm compared to the other two, and the fact that SVM performs better even in higher dimensions, though it is more computationally expensive compared to the others as a trade-off. Doing data selection by selecting the attributes that has correlation to the fraud label, or has shown distinct patterns from the EDA, and feeding the selected data to the

three algorithms shows an increase of performance across the board compared to using all of the attributes in the dataset. Which means that the attributes we have selected may be used for future references or used for other algorithms in credit card fraud detection tasks in order to increase performance and accuracy.

Room for improvement for our project includes using more data, using more complex algorithms for better results, as well as doing a more detailed EDA for the attributes for a more thorough analysis on all the attributes, and potentially finding other attributes that may contribute to determining potential fraudulent transactions

REFERENCES

[1] D. Kim, G. Antariksa, M. P. Handayani, S. Lee, and J. Lee, "Explainable Anomaly Detection Framework for Maritime Main Engine Sensor data," Sensors, vol. 21, no. 15, p. 5200, Jul. 2021, doi: 10.3390/s21155200.

[2] Y. Lim, "Unsupervised Outlier Detection with Isolation Forest," Medium, Mar. 22, 2022. [Online]. Available: https://medium.com/@limyenwee_19946/unsupervised-outlier-detection-with-isolation-forest-eab398c593b2

[3] M. Hachimi, G. Kaddoum, G. Gagnon, and P. Illy, "Multi-stage Jamming Attacks Detection using Deep Learning Combined with Kernelized Support Vector Machine in 5G Cloud Radio Access Networks," 2022 International Symposium on Networks, Computers and Communications (ISNCC), pp. 1–5, Oct. 2020, doi: 10.1109/isncc49221.2020.9297290.

[4] S. Bhattacharjee, P. Majumdar, and Y. J. Singh, An Effective Monitoring of Women Reproductive Organ Cancer using Mean based KPCA. 2018, pp. 87–92. doi: 10.1109/icrcicn.2018.8718730.

[5] K. Sultan, H. Ali, and Z. Zhang, "Call detail records driven anomaly detection and traffic prediction in mobile cellular networks," IEEE Access, vol. 6, pp. 41728–41737, Jan. 2018, doi: 10.1109/access.2018.2859756.

[6] Ibm, "K-Means Clustering," IBM, Dec. 19, 2024. https://www.ibm.com/think/topics/k-means-clustering

[7] "Anomaly Detection: Isolation Forest Algorithm:: My new Hugo site," Nov. 21, 2019. https://majdarbash.github.io/aws-cmls/2019-11-21-isolation-forest-algorithm/

[8] "IsolationForest," Scikit-learn. https://scikit-learn.org/1.5/modules/generated/sklearn.ensemble.IsolationForest.html

[9] "Major kernel functions in support Vector Machine - Javatpoint," www.javatpoint.com. https://www.javatpoint.com/major-kernel-functions-in-support-vector-machine

[10] GeeksforGeeks, "What is Isolation Forest?," GeeksforGeeks, Jul. 15, 2024. https://www.geeksforgeeks.org/what-is-isolation-forest/

[11] Selva Prabhakaran (ML+), "Isolation Forest: A Tree based approach for Outlier Detection (Clearly Explained)," YouTube. Aug. 16, 2023. [Online]. Available: https://www.youtube.com/watch?v=kqAxfOPlr1U

[12] GeeksforGeeks, "Support Vector Machine (SVM) algorithm," GeeksforGeeks, Oct. 10, 2024. https://www.geeksforgeeks.org/support-vector-machine-algorithm/

[13] H. Singh, "Support vector machines: from hard margin to soft margin," DEV Community, Aug. 12, 2024. https://dev.to/harsimranjit_singh_0133dc/support-vector-machines-from-hard-margin-to-soft-margin-1bj1

[14] GeeksforGeeks, "Kernel trick in support Vector classification," GeeksforGeeks, May 23, 2024. https://www.geeksforgeeks.org/kernel-trick-in-support-vector-classification/

[15] GeeksforGeeks, "Support Vector Machine (SVM) for anomaly detection," GeeksforGeeks, May 18, 2024. https://www.geeksforgeeks.org/support-vector-machine-svm-for-anomaly-detection/

[16] GeeksforGeeks, "K means Clustering Introduction," GeeksforGeeks, Aug. 29, 2024. https://www.geeksforgeeks.org/k-means-clustering-introduction/

[17] I. Dabbura, "K-Means Clustering: algorithm, applications, evaluation methods, and drawbacks," Medium, Sep. 27, 2022. [Online]. Available: https://towardsdatascience.com/k-means-clustering-algorithm-applications-evaluation-methods-and-drawbacks-aa03e644b48a

[18] T. Romani, "Harnessing the power of K-Means for anomaly detection," Medium, Aug. 10, 2023. [Online]. Available: https://medium.com/@tommaso.romani2000/harnessing-the-power-of-k-means-for-anomaly-detection-24dc71d260a8

[19] U. Riswanto, "K-Means Clustering for Anomaly Detection - Ujang Riswanto - Medium," Medium, Jan. 30, 2023. [Online]. Available: https://ujangriswanto08.medium.com/k-means-clustering-for-anomaly-detection-1bbbb0b20b52

[20] M. Rej and M. Rej, "Credit card fraud Statistics (2025)," Merchant Cost Consulting, Dec. 30, 2024. https://merchantcostconsulting.com/lower-credit-card-processing-fees/credit-card-fraud-statistics/: :text=Global%20credit%20card%20fraud%20will,and%20%2432.4%20billio

[21] S. Natha, "A Systematic Review of Anomaly detection using Machine and Deep Learning Techniques," Quaid-e-Awam University Research Journal of Engineering Science & Technology, vol. 20, no. 1, pp. 83–94, doi: 10.52584/qrj.2001.11.

[22] "Credit Card Transactions Dataset," Kaggle, Jul. 23, 2024. https://www.kaggle.com/datasets/priyamchoksi/credit-card-transactions-dataset/data

[23] "Credit Card Fraud data," Kaggle, Jul. 30, 2024. https://www.kaggle.com/datasets/neharoychoudhury/credit-card-fraud-data