Fraud Detection in Credit Card Transactions

MTH443: Group Project

Zehaan Naik, Aditya V, Viral Chitlangia

April 2025

Abstract

Credit card fraud detection presents a challenging problem due to the extreme class imbalance and adaptive strategies used by fraudsters. Traditional supervised methods struggle with the sparsity of fraudulent cases, motivating the need for unsupervised and semi-supervised approaches. In this project, we analyse the Kaggle credit card transaction dataset using data science techniques. Anomaly detection methods such as Isolation Forests, One-Class SVMs, and DBSCAN identify outliers in high-dimensional spaces. Hidden Markov Models capture sequential patterns in user behaviour, while graph-based methods with NetworkX reveal structural anomalies in transaction networks. Our results highlight the power of combining diverse methods to detect complex fraud patterns in real-world financial data effectively.

1 Introduction

Credit card fraud detection is a critical challenge in financial security, characterised by the rarity and sophistication of fraudulent transactions. The highly imbalanced nature of fraud data, where fraudulent events constitute a fraction of the total transaction volume, complicates traditional modelling approaches based on supervised learning and optimisation. This sparsity and the evolving tactics fraudsters employ necessitate using robust, adaptive, and often unsupervised data science techniques for effective detection.

In this project, we leverage the publicly available credit card transaction dataset from Kaggle, which contains anonymised features and labels for fraudulent and legitimate transactions. We explore multiple paradigms of anomaly detection to uncover hidden patterns indicative of fraud. Isolation Forests and One-Class SVMs are utilised for their strength in high-dimensional anomaly detection without relying on labeled data. DBSCAN is employed to detect density-based anomalies that deviate from normal transaction clusters. To capture temporal fraud patterns, we implement Hidden Markov Models (HMMs), which model the sequential nature of user behaviour and identify deviations that suggest fraudulent activity. Further, we construct transaction graphs using

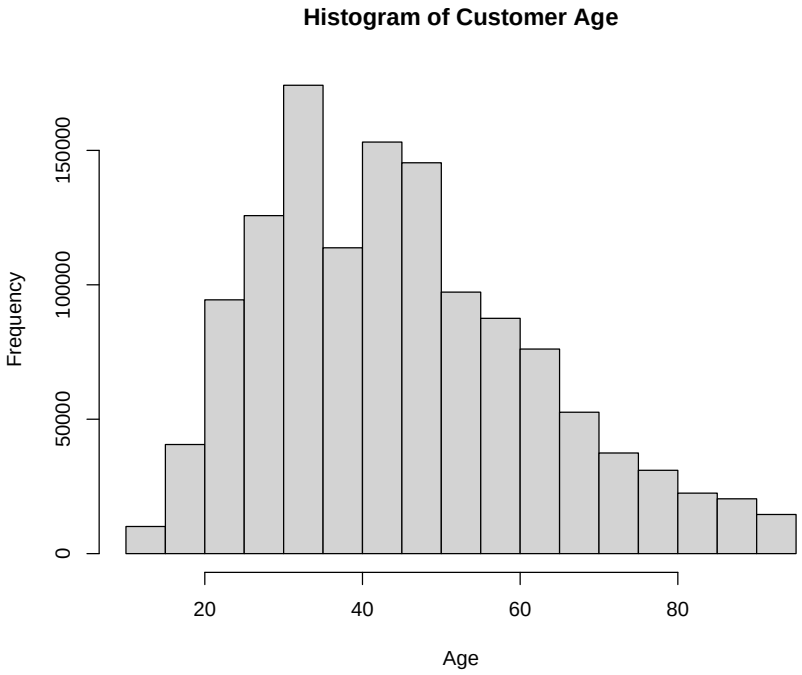

Figure 1: Histogram of Age of Customers

NetworkX to exploit relational structures between entities such as users, devices, and transaction channels. We highlight how fraud forms distinct topological signatures through graph-based anomaly scoring and structural analysis.

Overall, our work underscores the utility of combining multiple data science techniques across statistical, machine learning, and network analysis paradigms to address the complex and evolving landscape of credit card fraud.

2 Data

The Data, as mentioned in Section 1, came from Kaggle. This simulated credit card transaction dataset contains legitimate and fraudulent transactions from 1st Jan 2019 - 31st Dec 2020. It covers the credit cards of 973 customers doing transactions with a pool of 800 merchants.

2.1 Exploratory Data Analysis

We got the mean transactions observed as \$70.35. 45% of the Customers were Males. The mean age of the customers was 45.50 years(Subsection 2.2). Figure 1 shows the histogram of customers' ages. The minimum age observed was 13 years, and the maximum is 95 years. The histogram of the hour of a day of the transaction is as shown in Figure 2. We visualise the sparsity of frauds in our data in Fig. 3.

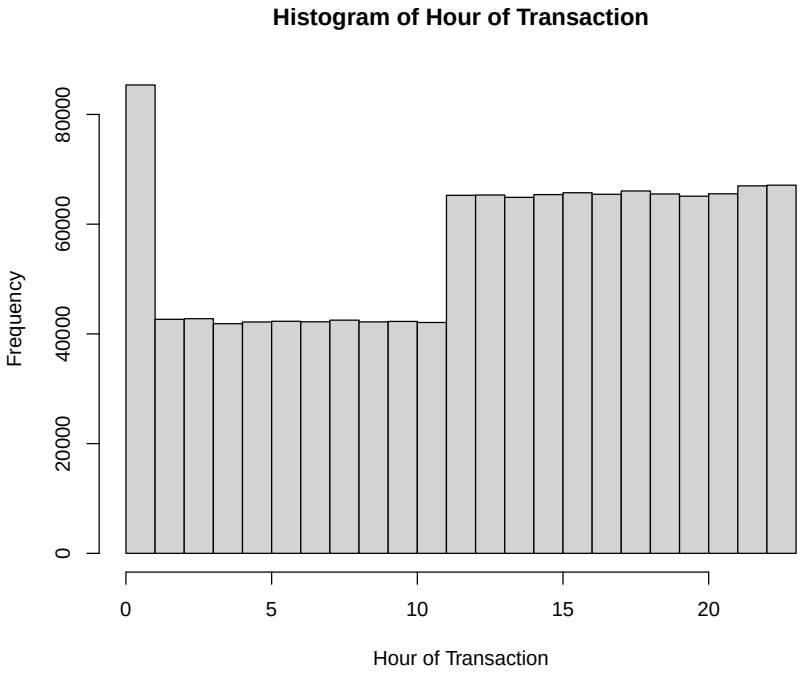

Figure 2: Histogram of Hour of Day of Transaction

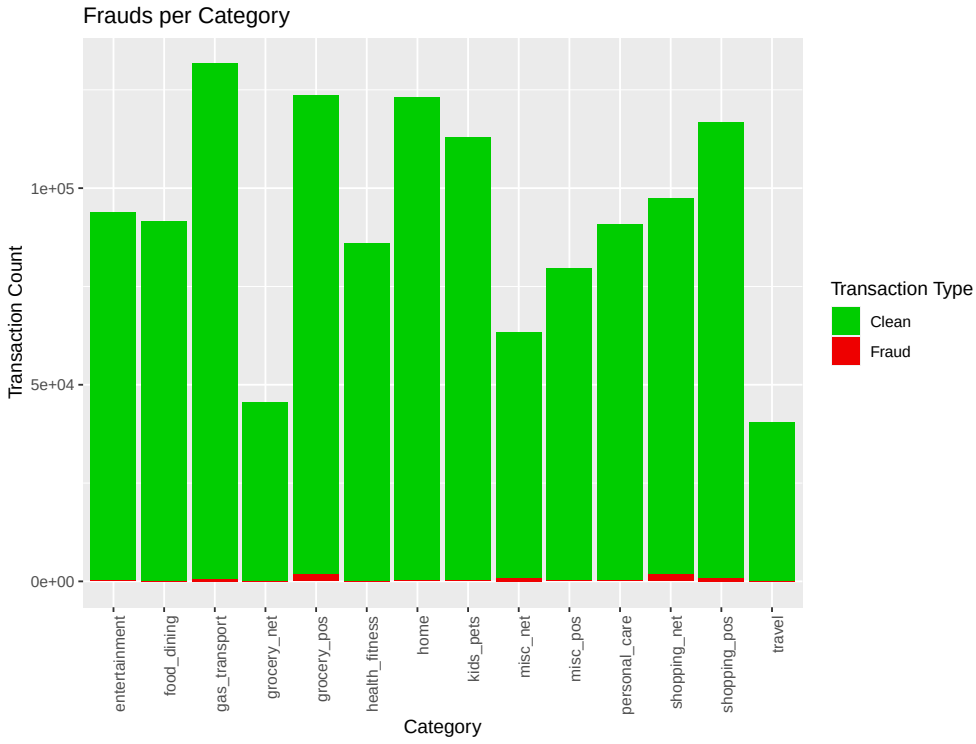

Figure 3: Sparse fraud cases in the data set

2.2 Cleaning Data

We take a random sample to work with for the modelling. Out of a total of 12,96,675 transactions, we take 10,000 samples. We first convert the dates to a R usable format from the data given. We use the date of birth and transaction dates to find the person's age while making the transaction. We used the Latitude and Longitude for the payer and payee to calculate the distance between them. We used the following features: transaction amount, age, city population, transaction hour, and distance. We scaled the features for analysis.

3 Method

We try many approaches, some unsupervised, some probabilistic modelling-based, and some graph-based, to identify a subset of methods that perform remarkably well on the data. By benchmarking and comparing them, we can identify a suitable combination of individual methodologies that perform well on fraud detection, allowing us to develop an appropriate pipeline to identify Credit Card fraud reliably.

The main approaches we tried were :

1. Isolation Forest
2. One Class SVM
3. DBSCAN
4. HMM
5. Tree-based detection

3.1 Isolation Forest

Isolation Forest is an Unsupervised Algorithm for Anomaly Detection. It works based on partitioning points from the clusters. The easier it is to partition a particle, the likelier it is to be an anomaly. This algorithm has linear time complexity and requires very low memory usage.
The algorithm builds random binary trees (isolation trees). It isolates observations by randomly selecting a feature and then selecting a split value between its minimum and maximum values. Since anomalies are distinct and located in sparse data regions, they typically require fewer splits to isolate, resulting in shorter average path lengths in the trees.
We tried to use Isolation Forest on the features to do Binary Classification, as in Figure 4. That did not give a very tangible result.

Performing a train-test split of 70-30, we get the accuracy values on the classification as mentioned in Table. 1.

Thus we see that while the accuracy values are good, its only able to identify 50% of fraud activities, while also falsely having 87% of its detected frauds being falsely accused.

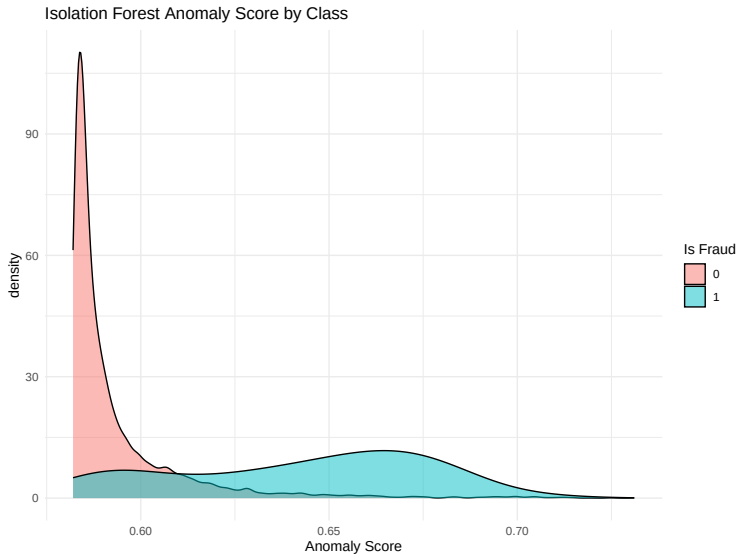

Figure 4: Isolation Forest Anomaly Scores

Accuracy	0.97
Precision	0.13
Recall	0.5
F1 Score	0.2
AUC	0.74

Table 1: Performance of Isolation Forest

3.2 One Class SVM

One Class SVMs are a variation of SVMs built to deal with real world data where the majority class outweighs the minority class. This is also the case in our data of interest, since instances of no fraud significantly outnumber the number of fraud cases.

One Class SVMs assume the data in the majority class to be normally distributed, following the general assumption of most real-world data being generally normal in nature. This form of modelling trains exclusively on the majority class, treating the minority class as outliers in the data.

The model uses a ν hyperparameter and effectively identifies Support vectors and error margins to identify "outliers" outside the normal space.

Moreover, since SVMs are linear models and the data we are dealing with is likely to have non-linearity concerning its classification, we use a Radial Basis Function kernel (RBF kernel) to transform the data, before training the SVMs, effectively transforming the classification model to a non-linear one.

Since SVMs would deal with distances between points for similarity, the Gaussian kernel transforms these distances as a result of the transformation of the datapoints as :

Accuracy	0.0073
Precision	0.0073
Recall	1
F1 Score	0.015
AUC	0.5

Table 2: Performance of One-Classification SVM

$$K(x_i, x_j) = \exp\left(-\gamma \|x_i - x_j\|^2\right)$$

However, SVMs struggle significantly in dealing with the data, since it performs very poorly, across a reasonably range of attempted hyperparameters. The approximate values obtained for performance metrics are in Table.2 Thus, SVMs seem to heavily identify data as all outliers, indicating that the Gaussian fit on this data is perhaps not a reasonable assumption.

3.3 DBSCAN

DBSCAN also identifies fraud cases as outliers in the credit card data. It is, in essence, a clustering algorithm that makes no assumptions about the shape or size of clusters. Instead, it assumes a cluster to be regions of high density surrounded by regions of low density.

It takes two hyperparameters :

- **eps**: The radius of the neighbourhood around a point.
- **minPts**: The minimum number of points within the radius to form a dense region.

It defines three types of points:

1. **core points**, which have a sufficient number of neighbours within a specified radius (epsilon)
2. **border points**, which are near core points but lack enough neighbours to be core points themselves
3. **noise points**, which do not belong to any cluster.

The algorithm works as follows:

1. **Identify Core Points**: For each point in the dataset, count the number of points within its eps neighbourhood. If the count meets or exceeds MinPts, mark the point as a core point.
2. **Form Clusters**: Create a new cluster for each core point that is not already assigned to a cluster. Recursively find all density-connected points (points within the eps radius of the core point) and add them to the cluster.

Accuracy	0.99
Precision	0.37
Recall	0.53
F1 Score	0.43
AUC	0.76

Table 3: Performance of DBSCAN

3. **Density Connectivity**: Two points, a and b, are density-connected if there exists a chain of points where each point is within the eps radius of the next, and at least one point in the chain is a core point. This chaining process ensures that all points in a cluster are connected through a series of dense regions.

4. **Label Noise Points**: After processing all points, any point that does not belong to a cluster is labelled noise.

In the case of our specific task, we treat noise points as cases of fraud. We analyse the algorithms' performance in Fig. 5. Fig. 6 compares the ROC curves of the Isolation Forest and One-Class SVM models. The Isolation Forest demonstrates superior discriminatory power, as evidenced by a higher curve closer to the top-left corner. This suggests it better distinguishes between fraudulent and legitimate transactions across thresholds. The One-Class SVM performs comparably but slightly worse, especially at lower false favourable rates. A diagonal line would indicate random performance; both models exceed this baseline, indicating effectiveness.

Performing hyperparameter tuning on a validation set, we identify optimal values as eps = 1.7 and minPts = 15.

The performance of DBSCAN on these hyperparameter values is

Thus we see that DBSCAN outperforms the other two approaches despite being unsupervised in nature. It's identification of fraud instances as outliers seem to be reasonably justified. Although it too, only identifies about 50% of fraud instances, it's precision is a lot better than isolation forests with a value of 0.37.

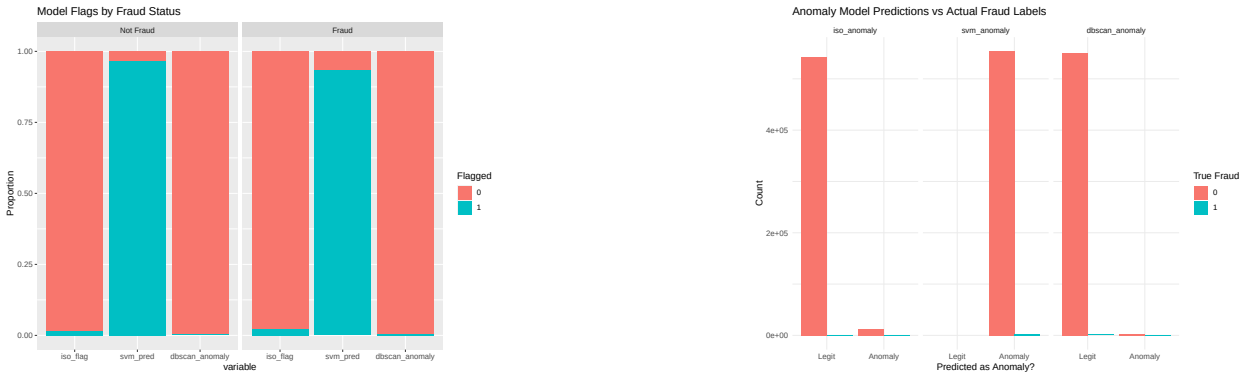

Figure 5: Comparing all the prediction models.

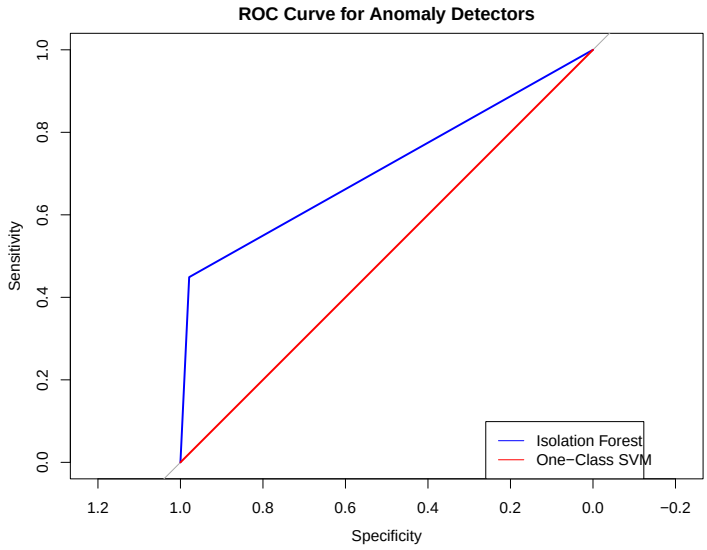

Figure 6: Receiver Operating Characteristic curve for Isolation Forest and SVMs

3.4 HMM

We use HMMS for risk profiling. i.e without using our labels, we attempt to profile hidden state spaces corresponding to "high-risk" behaviour, which potentially has implications on whether the data corresponds to a case of fraud.

Hidden Markov Models (HMMs) are statistical models used to represent systems that transition between hidden states over time, with each state producing observable outputs. A Hidden Markov Model (HMM) is a generative probabilistic model for sequential data. It consists of:

- Hidden states: Latent variables (not directly observed) that evolve following a Markov process.
- Emissions: Observed data generated from state-specific distributions.

We consider two states:

1 : Possibly "Low-risk"

2 : Possibly "High-risk"

We then build a 2-state HMM based on the transaction amount and hour. This HMM models the two hidden states as Gaussian and assigns the observations to each hidden state based on the posterior probability. It assumes an underlying sequence of hidden states $\{s_t\}_{t=1}^{T}$, each generating an observed vector x_t. The HMM consists of:

- A finite set of N hidden states, $s_t \in \{1, 2, \ldots, N\}$.
- An initial state distribution:

$$\pi_i = \Pr(s_1 = i), \quad i = 1, \ldots, N$$

- A state transition matrix:

$$A_{ij} = \Pr(s_t = j \mid s_{t-1} = i), \quad i, j = 1, \ldots, N$$

- Emission distributions:

$$p(x_t \mid s_t = j) = f_j(x_t), \quad j = 1, \ldots, N$$

In this fraud detection pipeline, we model each observation $x_t = (\texttt{amt}_t, \texttt{age}_t, \texttt{city\_pop}_t, \texttt{hour}_t, \texttt{distance}_t)$ as being emitted from a state-specific multivariate Gaussian:

$$p(x_t \mid s_t = j) \sim \mathcal{N}(\mu_j, \Sigma_j)$$

where μ_j and Σ_j denote the mean vector and covariance matrix for state j.

The full joint probability of the sequence $(x_{1:T}, s_{1:T})$ under the model is:

$$\Pr(x_{1:T}, s_{1:T}) = \pi_{s_1} \cdot p(x_1 \mid s_1) \cdot \prod_{t=2}^{T} A_{s_{t-1}, s_t} \cdot p(x_t \mid s_t)$$

We visualise the results of this model in Fig. 7.

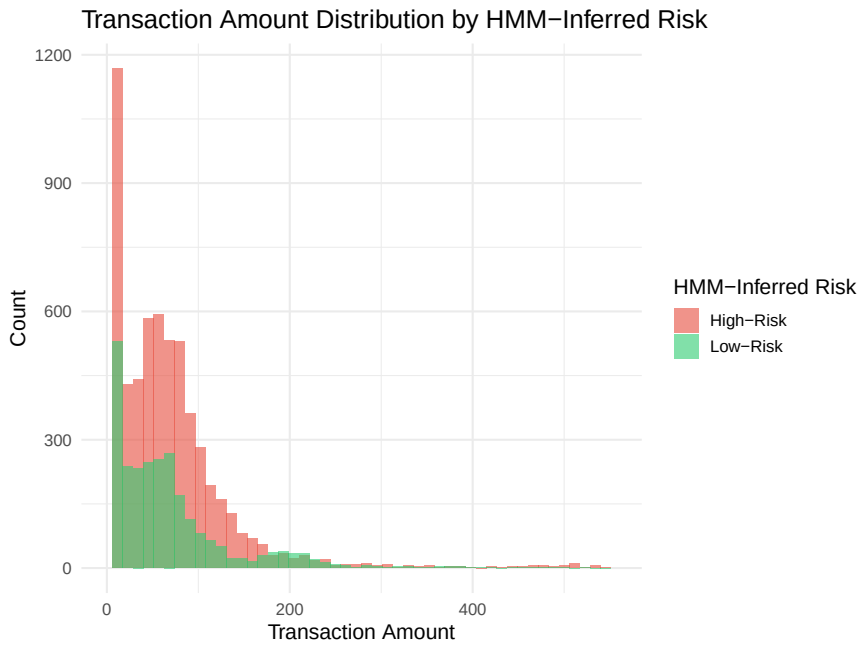

Figure 7: HMM inferred risk analysis

Given observations $x_{1:T}$, inference is performed using the forward-backwards algorithm to compute the posterior distribution over the hidden states:

$$\gamma_t(i) = \Pr(s_t = i \mid x_{1:T})$$

The most likely sequence of hidden states can be obtained via the Viterbi algorithm:

$$s^*_{1:T} = \arg\max_{s_{1:T}} \Pr(s_{1:T} \mid x_{1:T})$$

We then check each state's mean transaction amount, labelling them as high risk and low risk. In this way, we can classify the data into "high risk" and "low risk" cases of fraud detection.

3.5 Tree-based detection

We then tried to identify potential networks of fraudulent activity that might be occurring. For this purpose, we first create a graph. The nodes of this graph are unique credit cards, and the edges are constituted by cards that have transacted with the same merchant. This graph, therefore, represents networks of connected financial activity. By colouring them according to whether the card is associated with fraud or not, we can see networks of potential fraudulent activity being formed. The key idea is to model transactions as a network of shared merchant interactions among cardholders.

3.6 Graph Construction

Let $G = (V, E)$ be an undirected graph where:

- Each vertex $v \in V$ corresponds to a unique credit card number (`cc_num`).
- An edge $(v_i, v_j) \in E$ exists if both card numbers v_i and v_j have transacted with the same merchant.

Formally, for each merchant m with a set of associated cardholders $C_m = \{c_1, c_2, \ldots, c_k\}$, we define the set of induced edges:

$$E_m = \{(c_i, c_j) \mid c_i, c_j \in C_m,\ i < j\}$$

The union constructs the whole graph:

$$E = \bigcup_{m \in \mathcal{M}} E_m$$

where $\mathcal{M}$ is the set of all merchants.

3.6.1 Fraud Labelling and Node Attributes

Each node $v \in V$ is assigned a binary label:

$$f(v) = \begin{cases} 1 & \text{if card number } v \text{ has at least one known fraudulent transaction,} \\ \\ 0 & \text{otherwise.} \end{cases}$$

We use the Louvain algorithm to partition G into communities $\{C_1, C_2, \ldots, C_k\}$ by maximizing the *modularity* score:

$$Q = \frac{1}{2m} \sum_{i,j} \left[A_{ij} - \frac{k_i k_j}{2m} \right] \delta(c_i, c_j)$$

where:

- A_{ij} is the adjacency matrix of G,
- k_i is the degree of node i,
- m is the total number of edges in the graph,
- $\delta(c_i, c_j) = 1$ if nodes i and j are in the same community, 0 otherwise.

Higher modularity indicates stronger intra-community connections compared to a null model.

3.6.2 Visualization and Interpretation

We use a Fruchterman-Reingold layout to visualise the graph. Nodes are colored according to fraud status:

- **Red nodes**: cardholders with known fraudulent transactions.
- **Blue nodes**: legitimate cardholders.

Dense clusters of red nodes may indicate coordinated fraud or merchant compromise. This approach complements feature-based models by highlighting relational anomalies.

We provide this graph in Fig. 8.

This means that apart from previously discussed ways to identify fraud, checking if a card is associated with these networks of fraudulent activity identified via the graph, we can locate data at higher risk of being associated with fraudulent activity.

4 Conclusion

In this project, we explored a comprehensive set of anomaly detection techniques to tackle the complex challenge of credit card fraud detection. Given the rarity and unpredictability of fraudulent transactions, we focused primarily on unsupervised and semi-supervised methods that do not rely

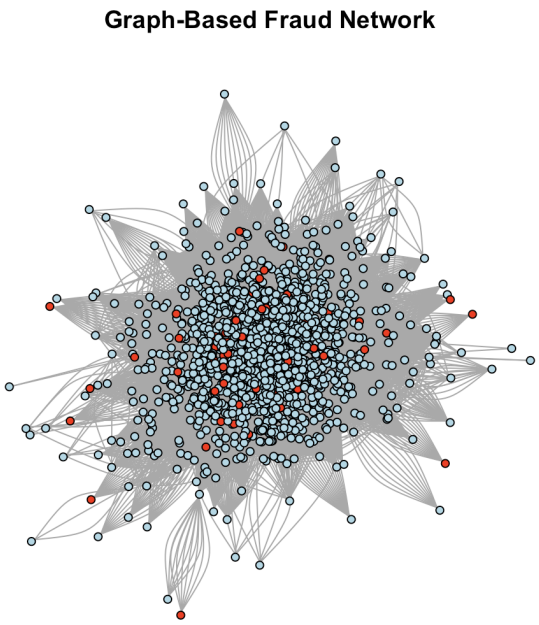

Figure 8: Graph-based analysis of fraudulent transactions

heavily on labelled data. Our pipeline incorporated statistical anomaly detection (Isolation Forest, One-Class SVM, DBSCAN), probabilistic modelling (Hidden Markov Models), and structural analysis (graph-based detection using NetworkX).

Each method brought unique strengths: Isolation Forest demonstrated strong general performance across features, while HMMs offered interpretable temporal insights through latent state transitions. Graph-based methods revealed communities of potentially collusive behaviour that would not be captured in feature space alone. Combining these approaches yielded a richer, more nuanced view of fraud than any single method could achieve in isolation.

Our findings underscore the value of integrating diverse data science methodologies—spanning statistics, machine learning, and network theory—for real-world fraud detection. Future extensions may include supervised learning ensembles, temporal graph modelling, or incorporating Bayesian inference for uncertainty quantification. This project highlights how principled modelling can illuminate latent structure in financial data and enhance fraud mitigation strategies.