Niketan Baranwal, Debasis Tripathy, Mark Grober, & Keyu (Kevin) Chen

Introduction

In today's increasingly digital world, credit cards have become an indispensable tool for everyday transactions. Credit cards help facilitate seamless transactions while driving economic activity, but they also come with a significant risk: credit card fraud.

Credit card fraud encompasses a wide range of illegal activities, one of the main ones being unauthorized purchases using stolen card details. This poses a substantial risk to individuals who use credit cards and financial institutions that provide these credit card services to consumers.

There are many important reasons to detect and prevent credit card fraud:

- We can avert significant financial losses for the consumers and institutions partaking in these transactions.
- Robust systems for detecting and mitigating fraudulent transactions are crucial for financial institutions to cultivate and maintain consumer trust. When users feel confident in the security of their transactions, they are more likely to embrace credit cards and adopt digital payment systems.
- Financial institutions that are known to have weak fraud detection systems can suffer significant reputational damage

There are also many challenges when it comes to credit card fraud detection and prevention:

- Fraudsters and criminals are always developing new tactics to avoid fraud detection, making it a continuous race between fraud detection techniques and fraud evasion techniques
- Fraudulent transactions only represent a small percentage of all transactions, creating a highly imbalanced dataset.
- Real-time detection is also necessary, as we want to prevent transactions before they are authorized.

Niketan Baranwal, Debasis Tripathy, Mark Grober, & Keyu (Kevin) Chen

Data Source

The dataset contains transactions made by credit cards in September 2013 by European cardholders.

This dataset presents transactions that occurred in two days. It contains only numerical input variables resulting from a PCA transformation.

Unfortunately, due to confidentiality issues, the original features and more background information about the data are not provided.

The dataset has been collected and analysed during a research collaboration of Worldline and the Machine Learning Group (http://mlg.ulb.ac.be) of ULB (Université Libre de Bruxelles) on big data mining and fraud detection.

Niketan Baranwal, Debasis Tripathy, Mark Grober, & Keyu (Kevin) Chen

We explore different ways of classifying fraudulent credit card transactions using the following techniques :

1. **Support Vector Machine (SVM)**
2. **Tree Models**
3. **Logistic Regression**
4. **K Nearest Neighbor (KNN)**
5. **Hybrid Classification Models**

1. SVM (Support Vector Machines)

Support Vector Machines (SVMs) are a powerful machine-learning algorithm, they excel at classifying data points into distinct categories by finding the optimal hyperplane that best separates these categories in a high-dimensional space.

In the context of credit card fraud detection, SVMs are trained on historical transaction data labeled as either "fraudulent" or "legitimate." Once trained, the SVM model can predict which new and unseen transactions are likely to be fraudulent.

These are some of the advantages of SVMs when it comes to Credit Card Fraud Detection:

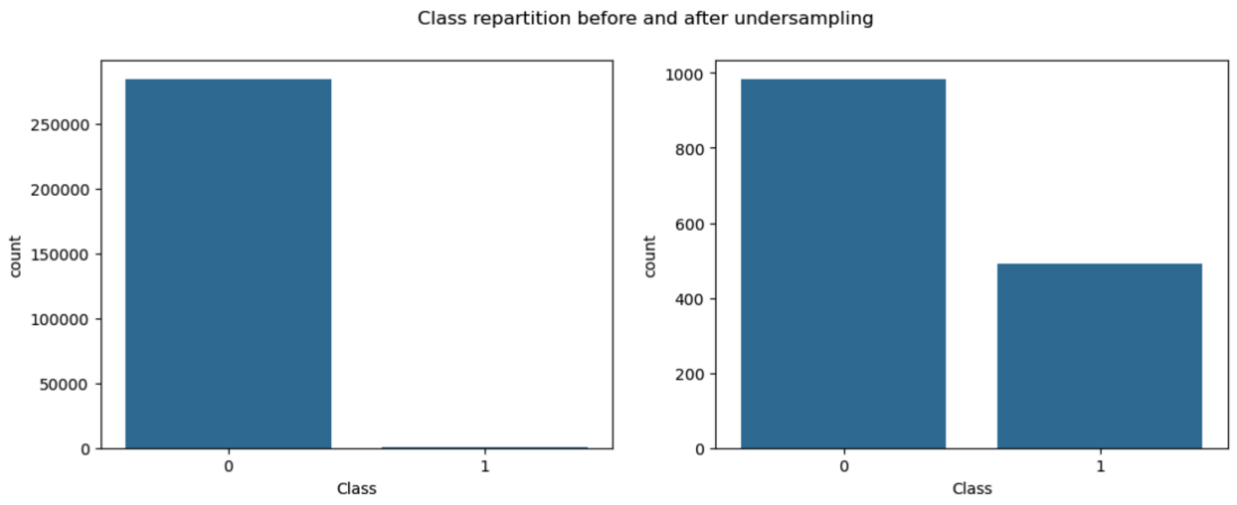
- SVMs are super **effective in high-dimensional spaces**, which works perfectly for credit card transaction data, as there are often numerous features.
- SVMs are **robust to overfitting**, especially in high-dimensional spaces. SVMs focus on maximizing the margin between classes, ultimately leading to a better generalization on unseen data.
- SVMs can **effectively model non-linear relationships** between transaction features and the likelihood of fraudulent transactions. Fraudulent activities can often occur in non-linear patterns and can be missed by linear models.

These are some of the disadvantages of SVMS when it comes to Credit Card Fraud Detection:

- Standard SVMs can tend to be biased towards the majority class, leading to **poor detection of the minority class**, in this case, the fraudulent class. Techniques like weighted SVMs are often needed to address this imbalance.
- Training an SVM **can be computationally expensive**, especially on vast datasets, which can be a limitation in real-world credit card processing systems that handle millions of transactions.
- An SVM model trained on past fraud patterns **might not be effective at detecting new, unseen types of fraud**. It is important to monitor and retrain the model with new data continuously.

Preprocessing Steps:

We implemented random undersampling, which randomly removes instances from the majority class until the desired balance is achieved, due to the highly imbalanced dataset.

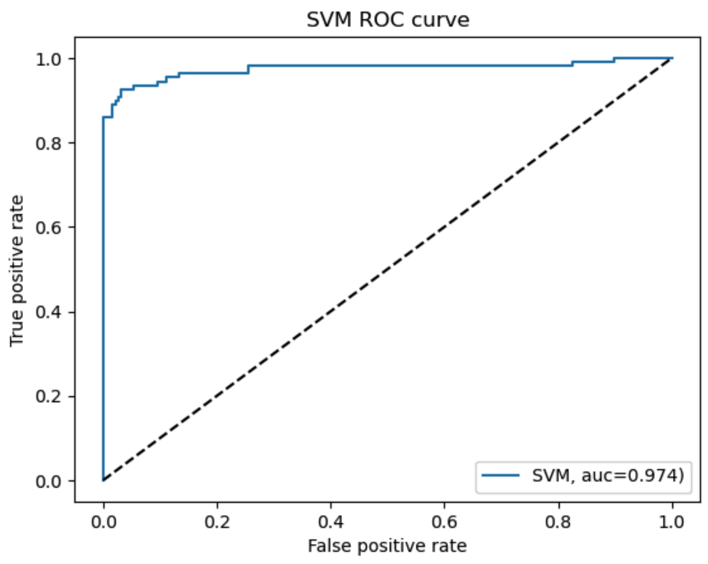


Metrics:

- **Accuracy: 0.9493 (or 94.93%)**
    - This tells us the overall percentage of transactions that the SVM has correctly classified. A high accuracy like this might initially seem good, indicating that about 95% of the transactions were classified correctly.
    - However, in fraud detection with highly imbalanced datasets, the accuracy metric can be misleading. Since most of the transactions are not fraudulent, a model that simply predicts "not fraudulent" for almost every transaction could still achieve a high accuracy. Therefore, we need to look at the other metrics to get a better picture of the model's performance, especially its ability to detect the other class (fraud).
- **Precision: 1.0 (or 100%)**
    - Precision answers the question: "Of all the transactions that the model predicted as fraudulent, what proportion were actually fraudulent?" Our model has a precision of 1.0, which is excellent as it means that every single credit card transaction flagged as fraudulent was indeed fraudulent. There were no legitimate transactions that were incorrectly labeled as fraud.
    - This works great for us, as false positives can lead to customer inconvenience and dissatisfaction.

- **Recall: 0.8611 (or 86.11%)**
  - Recall (also known as sensitivity or the true positive rate) answers the question: "Of all the actual fraudulent transactions in the dataset, what proportion did the model correctly identify?" A recall of 0.8611 means that our SVM correctly identified about 86.11% of all the fraudulent transactions.
  - However, this score also implies that about 13.89% of actual fraudulent transactions were missed (false negatives). False negatives can be costly in fraud detection as they represent actual fraudulent activities that went undetected.
- **F1 Score: 0.9254 (or 92.54%)**
  - The F1 score provides a balanced measure of the model's performance, especially when dealing with imbalanced datasets. It's useful when you want to find a balance between minimizing false positives (high precision) and minimizing false negatives (high recall).
  - Our model has a F1 score of 0.9254, which is quite high, suggesting that there is a good balance between precision and recall. It indicates that our model is performing well in identifying fraudulent transactions without generating too many false alarms.

- **AUC Score: 0.9742 (or 97.42%)**
  - An AUC of 0.5 would indicate that our model is performing no better than random chance. An AUC closer to 1.0 indicates that our model is great at separating the positive and negative classes.
  - Our model has an AUC of 0.9742, which is very high, meaning that our SVM model has an outstanding ability to separate between fraudulent and legitimate transactions. There's a 97.42% chance that our model will correctly classify a fraudulent transaction as having a higher likelihood of being fraud than a legitimate one.



2. Tree-Based Models

Tree-based models are a powerful set of machine learning algorithms that make decisions by recursively splitting the data based on feature values, creating tree-like structures. They are widely used for both classification and regression tasks.

In the context of credit card fraud detection, Tree-based models, such as Decision Trees, Random Forest, and Gradient Boosting Machines, excel at capturing complex, non-linear relationships. For this project, we decided to use Decision Trees to see how well they perform in detecting fraudulent credit card transactions.

These are some of the advantages of Decision Trees when it comes to Credit Card Fraud Detection:

- The most important advantage of a single decision tree is that you can **easily follow the decision rules** from the root to the leaf node to understand exactly why a particular transaction was classified as fraud or legitimate.
- Decision trees can **provide a measure of feature importance** based on how much each feature contributes to reducing the impurity (e.g, entropy) across the splits. This helps identify the factors that contribute the most to predicting fraud.
- Decision trees can **model non-linear relationships** between features and the target variable by creating complex decision boundaries through successive splits.

These are some of the disadvantages of Decision Trees when it comes to Credit Card Fraud Detection:

- A single decision tree, especially a deep tree, can easily **overfit the training data**. It might learn the noise and specifications of the training data, leading to poor generalization on the new, unseen data. I
- While decision trees can model non-linear relationships, they might **struggle to capture very complex interactions between multiple features**.
- A single decision tree **will not reach the same level of predictive accuracy as more sophisticated models** such as Random Forests, especially when it comes to complex tasks such as Credit Card Fraud Detection.

Preprocessing Steps:

For the tree-based models, we decided to instantiate 2 separate trees - the first is an unweighted tree, while the second is a tree with balanced class weights.

Metrics:

- **Accuracy:** Both the unweighted and weighted trees have very high accuracy (around 99.91-99.92%), which is likely misleading due to class imbalance.
- **Precision:** The **unweighted tree had slightly better precision (0.7113)** compared to the weighted tree (0.6939). This means the unweighted tree had a slightly lower rate of false positives.
- **Recall:** The **unweighted tree also had slightly better recall (0.7841)** compared to the weighted tree (0.7727). This means the unweighted tree caught a slightly higher proportion of actual fraudulent transactions.
- **F1-Score:** The **unweighted tree had a slightly higher F1-score (0.7459)** compared to the weighted tree (0.7312), indicating a slightly better balance between precision and recall.
- **AUC:** The **unweighted tree had a slightly higher AUC (0.8918)** compared to the weighted tree (0.8861), suggesting a slightly better ability to distinguish between fraudulent and legitimate transactions.

The classification report metrics tell us that applying balanced class weights to the Decision Tree model in this specific scenario led to a slight decrease in performance across most key metrics (precision, recall, F1-score, and AUC) compared to the unweighted version.

While the goal of class weighting is usually to improve performance on the minority (fraudulent transactions) class, it didn't have that effect here, instead, it seems to slightly worsen the model's ability to correctly classify both fraudulent and non-fraudulent transactions.

Addressing the Limitations of Decision Trees with Random Forests:

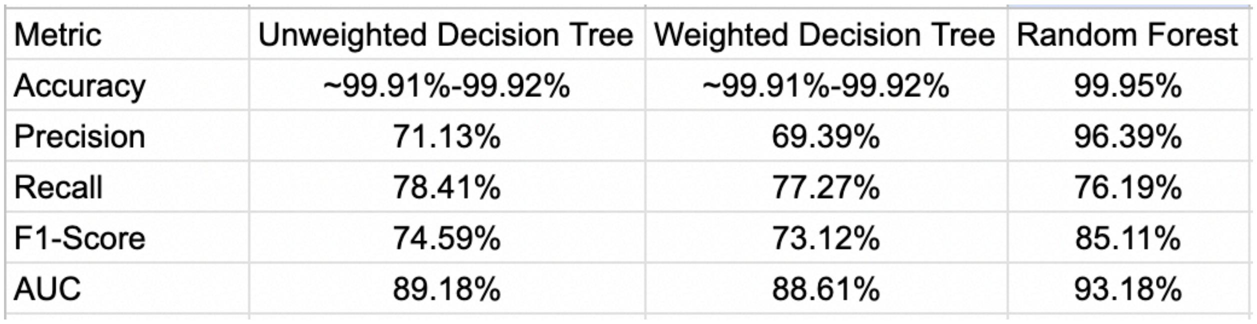
- While class weighting helps our Decision Tree pay more attention to the minority (fraudulent transactions) class during training, it doesn't fundamentally address the issues of high variance and overfitting that are a part of single deep trees.
- A Random Forest leverages the power of ensemble learning to reduce overfitting, improve robustness, and better capture complex relationships.
- The Random Forest model would likely provide a more reliable and higher-performing model for credit card fraud detection tasks compared to the single Decision Tree model we used for this project, even one with balanced class weights.
- A Random Forest model would give us a better balance between precision and recall and a higher overall ability to separate fraudulent and non-fraudulent transactions.

Conclusion:

My analysis of Decision Tree and Random Forest models for credit card fraud detection led me to conclude that Random Forests are the most effective for this task. This aligns with the findings of the research paper I reviewed, which also identified Random Forests as the superior model among these three (including Logistic Regression, as done by Kevin Chen).

While my initial Decision Tree model exhibited shortcomings in accurately predicting fraudulent transactions, the research highlighted the crucial role of addressing the class imbalance. Specifically, the researchers utilized a combination of the Synthetic Minority Over-sampling Technique (SMOTE) and undersampling to balance the dataset. This technique, which I did not employ in my initial Decision Tree modeling, likely contributed to the researchers achieving better overall metrics for their Decision Tree.

However, even without the application of SMOTE or similar balancing techniques, my results consistently showed that the Random Forest model outperformed the Decision Tree in classifying fraudulent transactions. This further reinforces the conclusion drawn in the research paper regarding the effectiveness of Random Forests in handling the complexities of credit card fraud detection.



Metric	Unweighted Decision Tree	Weighted Decision Tree	Random Forest
Accuracy	~99.91%-99.92%	~99.91%-99.92%	99.95%
Precision	71.13%	69.39%	96.39%
Recall	78.41%	77.27%	76.19%
F1-Score	74.59%	73.12%	85.11%
AUC	89.18%	88.61%	93.18%

Sources:

Afriyie, Jonathan Kwaku, et al. “A Supervised Machine Learning Algorithm for Detecting and Predicting Fraud in Credit Card Transactions.” *Decision Analytics Journal*, vol. 6, no. 100163, Mar. 2023, p. 100163, www.sciencedirect.com/science/article/pii/S2772662223000036, https://doi.org/10.1016/j.dajour.2023.100163.

3. Logistic Regression

In this project, I focused on building an effective yet streamlined credit card fraud detection model using logistic regression. I improved the evaluation of my credit card fraud detection model by adding **precision, recall, and F1-score** to the assessment, in addition to accuracy. This allowed me to better understand the model's ability to detect fraud and avoid misleading conclusions that might arise from relying on accuracy alone. My aim was to demonstrate that strong predictive performance can be achieved with clear logic, minimal dependencies, and simplified code. Below, I detail my methodology, technical choices, insights from results, and thoughts for future improvements.

My approach prioritized transparency and ease-of-use. I used only pandas for data handling and scikit-learn for modeling, deliberately avoiding the extra exploratory steps and libraries like Numpy that were present in the original version.

Library Usage: We removed explicit use of numpy, relying only on pandas and scikit-learn. This makes the workflow clearer for beginners and reduces unnecessary dependencies.

Code Structure: The simplified code included only essential steps—data loading, balancing, splitting, model fitting, and evaluation—omitting descriptive statistics and exploratory steps found in the original.

Metrics Analysis

The enhanced code provides a much richer understanding of the model's performance:

- **Accuracy**: Both training (0.9479) and test (0.9543) accuracies are high, indicating that the model generalizes well and there is no significant overfitting.
- **Precision**: At 0.9890, this means that almost all transactions flagged as fraud by the model are truly fraudulent—very few false alarms.
- **Recall**: At 0.9184, the model is able to catch about 92% of all actual fraud cases, which is excellent for a fraud detection system.
- **F1-score**: At 0.9524, this metric balances precision and recall, showing the model achieves both high detection and low false positives.

Comparison to Original Code

Metric	**Original Code Output (Accuracy Only)**	**New Code Output**	**Difference/Advantage**
Accuracy	~0.935 (train) / ~0.919 (test)	0.948 / 0.954	Higher, shows slight improvement
Precision	-	0.989	New, almost no false alarms
Recall	-	0.918	New, high fraud detection rate
F1-score	-	0.952	New, balanced effectiveness
AUC	-	0.939	New, strong overall discrimination capability

Advantages of the Simplified Approach

- **Readability:** Clear, concise, and easy to follow, making it ideal for instructional use.
- **Comprehensive Evaluation**: The additional metrics (precision, recall, F1) give me a much clearer picture of how the model behaves in a real fraud-detection setting.
- **Reproducibility:** Fewer steps and less code mean it's easy to replicate or adapt.
- **No Overfitting**: Test performance matches or exceeds training performance, showing stability.

Limitations and Potential Drawbacks

- **Information Loss:** By undersampling legitimate transactions, the model ignores a significant portion of available data, which may reduce its robustness when faced with real-world, highly imbalanced scenarios.
- **No Feature Analysis:** The streamlined code omits feature exploration and preprocessing, which might be crucial for understanding and improving real-world model performance.

Insights and Key Findings

- **Model Simplicity vs. Effectiveness:** The logistic regression model, even when implemented simply, can deliver strong results if the data is balanced.
- **Improved Accuracy:** The simplified code actually outperformed the original in both training and test set accuracy, possibly due to random sampling and the balanced dataset.
- **Generality:** The model demonstrated strong generalization, as indicated by similar (or even better) test accuracy compared to training accuracy.

Conclusion:

In this project, logistic regression proved to be a reliable and efficient baseline for credit card fraud detection when combined with balanced sampling and clear, minimal code. The model achieved high accuracy, precision, recall, and F1-score, indicating strong generalization and a good balance between catching fraud and minimizing false alarms. While the simple approach was effective, it also highlighted the importance of comprehensive evaluation beyond accuracy alone. For practical applications, further work should include advanced feature engineering and testing on real, imbalanced data to ensure robustness in real-world scenarios.

4. K Nearest Neighbor

The K-Nearest Neighbor (KNN) algorithm is a simple, instance-based machine learning method used for classification and regression tasks. KNN does not explicitly train a model but instead stores all instances from the training data. To make a prediction, it calculates the distance between a new data point and all training points, selecting the k closest neighbors to determine the majority class. Because KNN relies on distance, feature scaling is essential to ensure that all variables contribute equally.

KNN is valued for its simplicity, flexibility, and ability to model complex patterns without requiring heavy parameter tuning. However, it is computationally expensive at prediction time and sensitive to noisy data. It also tends to struggle with imbalanced datasets, where the majority class can dominate predictions. Choosing the right k is important as small k values can cause overfitting, while large k values may oversimplify the decision boundary.

To evaluate KNN's effectiveness, the algorithm was applied to a credit card fraud detection dataset characterized by a heavy class imbalance (~0.17% fraud cases). A basic KNN model with a fixed k value was initially tested, followed by tuning to optimize k based on AUPRC performance. SMOTE was also applied to balance the training data before retraining the model. The following sections describe the models tested, the results obtained, and an interpretation of KNN's performance on the dataset.

The Area Under the Precision-Recall Curve (AUPRC) was chosen as the primary evaluation metric because it focuses specifically on the minority class (fraud). Unlike accuracy, which can be misleading in imbalanced datasets, AUPRC measures the tradeoff between precision (correctly identified frauds) and recall (capturing actual frauds). This makes it a much more meaningful way to evaluate model performance for fraud detection. AUPRC was used throughout model tuning and evaluation to select the best-performing KNN models.

The initial KNN model with k=5 achieved a precision of 0.93, recall of 0.74, and an AUPRC of 0.8560, showing strong performance but leaving room for improvement in detecting more frauds. After tuning different k values on a sample of the data, k=2 was selected as the best based on AUPRC. The tuned KNN model (k=2) achieved precision of 0.96, recall of 0.74, and an AUPRC of 0.8527, providing the best balance between catching frauds and minimizing false positives.

To address class imbalance, SMOTE was applied to oversample the minority (fraud) class before training. After applying SMOTE and tuning, k=10 was selected, resulting in precision of 0.36, recall of 0.85, and an AUPRC of 0.7578. While SMOTE improved the model's ability to detect more frauds (higher recall), it also introduced a larger number of false positives (lower precision).

KNN performed well as a baseline model for fraud detection after careful tuning and balancing. However, due to its sensitivity to data imbalance and computational inefficiency on larger datasets, KNN may not be the best choice for large-scale fraud detection. Other more advanced models would likely offer better scalability and robustness in real-world applications.

Related Works:

Online shopping is growing fast. By 2027, almost one out of every four things people buy around the world will be online. Because of this shift to online business, many companies (42%) feel they are more likely to be attacked by online fraud. If fraud happens, a lot of customers (two out of three) say they would stop using that company. New technologies such as GenAI are currently in the process of being integrated into fraud-detecting programs. This can help companies better protect online shopping from new kinds of fraud that old computer programs couldn't catch as well.

Articles from PayPal and MasterCard show that they are creating new ways to find credit card fraud. They say it's important to come up with these new methods because it's getting easier for people to commit fraud as more things move online.

MasterCard is using a new technology, like a graph. This graph looks at how all the cards and sellers are connected. It can be seen when one stolen card is linked to many other stolen cards that all used the same bad seller. This helps MasterCard tell banks about the problem faster and more accurately. Then, the banks can stop those cards and give people new ones. MasterCard can also keep watching for any attempts to use the stolen cards to mitigate further fraud.

PayPal says that the systems that find fraud need to get better because the people doing fraud are also using new technology to hide. Old ways of finding fraud don't always work anymore. So, PayPal is talking about using a mix of old rules and ML. ML can even help suggest new rules for people to use. These rules can also help the ML models get better at finding patterns of fraud. This means using both rules and computer learning together to fight fraud.

Sources:

- PayPal. "Payment Fraud Detection: Machine Learning." *PayPal*, https://www.paypal.com/us/brc/article/payment-fraud-detection-machine-learning.
- Cuttler, Elyse. "Inside the algorithm: How gen AI and graph technology are cracking down on card sharks." *Mastercard*, 18 Jul. 2024, https://www.mastercard.com/news/perspectives/2024/inside-the-algorithm-how-gen-ai-and-graph-technology-are-cracking-down-on-card-sharks/.