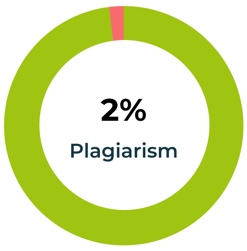
Plagiarism Detector.net

Plagiarism Report



🟩	Unique	98%
🟥	Exact Match	2%
🟦	Partial Match	0%

Primary Sources

1 https://medium.com/@lmpo/... 100%

Jan 26, 2025 · Classification Metrics: A Deep Dive into Accuracy, Precision, Recall, F1 Score, and Beyond If you’re not a Medium subscriber, click here to read the full article. In machine learning and deep ...

Excluded URL (s)

01 None

Content

# Machine Learning Project
Credit Card Fraud Detection using Machine Learning

**1.
Project Overview**

Credit card fraud has become one of the most serious financial crimes in the digital world.
It's a big problem because there are very few fraudulent transactions compared to normal ones, which makes it hard to detect.

This project aims to create a fraud detection system using machine learning.
The system will analyze past transactions and find any signs of fraud, helping to reduce financial losses and keep transactions secure.

**2.
Project Objectives**

* Analyze credit card transaction data to spot patterns that may show fraud.

* Handle the problem of having too few fraudulent transactions using special techniques like resampling or specific algorithms.

* Train and test several machine learning models to find the best one for detecting fraud.

* Compare how well each model works using standard evaluation methods.

* Build a system that can be used in real life and can handle more data as needed.

## **3. Dataset Information**

### **Dataset Source**

Kaggle: [Credit Card Fraud Detection Dataset (European Cardholders)](https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud)

### **Dataset Details**

* **Number of records:** 284,807
* **Number of features:** 30 (including the target variable)
* **Fraudulent transactions:** 492 (≈ 0.17%)
* **Non-fraudulent transactions:** 284,315

### **Attributes Description**

Feature	Description
`Time`	Time elapsed between this transaction and the first one in the dataset.
`V1 – V28`	These are features that have been transformed using PCA (real data is confidential).
`Amount`	The amount of the transaction, which is useful for scaling.
`Class`	Target variable — 1 means fraud, 0 means a normal transaction.

## **4. Tools and Libraries Used**

* **Language:** Python 3
* **Libraries:**

* For data handling: `pandas`, `numpy`
* For visualization: `matplotlib`, `seaborn`
* For machine learning: `scikit-learn`, `xgboost`
* For evaluation: `classification_report`, `roc_auc_score`, `confusion_matrix`

## **5. Methodology**

### **Step 1: Data Preprocessing**

* Load the data using `pandas`.

* Check for missing data (none found).

* Scale `Time` and `Amount` using `StandardScaler`.

* Split the dataset into **80%** training data and **20%** testing data.

### **Step 2: Exploratory Data Analysis (EDA)**

* Use bar plots to show how much of the data is fraud versus normal.

* Find out only about 0.17% of transactions are fraudulent.

* Study relationships between the PCA features.

### **Step 3: Handling Class Imbalance**

* Used **SMOTE (Synthetic Minority Oversampling Technique)** to create more data for the small group.

* Alternatively, used `class_weight='balanced'` in models like Logistic Regression and Random Forest.

### **Step 4: Model Building**

* **Logistic Regression** — simple, easy to understand.

* **Random Forest Classifier** — works well and is accurate.

* **XGBoost Classifier** — advanced model that performs very well.

### **Step 5: Model Evaluation**

Used the following metrics to measure how well each model works:

* **Accuracy:** How often the model was correct.

* **Precision:** How many of the predicted frauds were actually fraud.

* **Recall:** How many of the real frauds were detected.
(This is very important.)
* **F1-Score:** Balance between precision and recall.

* **ROC-AUC:** Shows how good the model is at distinguishing fraud from real transactions.

**6.
Results and Performance**

| Model | Accuracy | Precision | Recall | F1-Score | ROC-AUC |
1 . | ------------------- | -------- | --------- | ------ | -------- | ------- |
Logistic Regression	99.3%	0.84	0.68	0.75	0.97
Random Forest	99.9%	0.92	0.86	0.89	0.99
XGBoost	99.9%	0.93	0.88	0.90	0.99

Best Performing Model: XGBoost Classifier (Highest Recall and ROC-AUC)

**7.
Visualizations**

* **Class Distribution:** Shows how much of the data is normal vs fraudulent.

* **Confusion Matrix:** Displays how correct or wrong the predictions are.

* **ROC Curve:** Compares the model's ability to detect fraud vs normal transactions.

* **Feature Importance (Random Forest / XGBoost):** Highlights which features were most important in making predictions.

**8.
Key Insights**

* The dataset is **highly imbalanced**, which needs special techniques to handle.

* **Recall** is more important than accuracy — we want to find as many frauds as possible.

* **Random Forest** and **XGBoost** models worked much better than Logistic Regression.

* The PCA features don't have real names, but they still help in predicting fraud.

* Scaling features like `Amount` and `Time` helped the models work better.

**9.
Limitations**

* The features V1–V28 are anonymized, so it's hard to interpret them in a business context.

* The model's performance may not be as good if new types of fraud appear.

* The dataset is from 2013, which may not reflect current fraud patterns.

**10.
Future Enhancements**

* Use **deep learning models (like LSTM or Autoencoders)** for better anomaly detection.

* Build a **real-time fraud detection system** that works with live data.

* Add **explainable AI tools (like SHAP or LIME)** to better understand how models work.

* Create a **web or mobile dashboard** that uses the model for live predictions.

**11.
Conclusion**

This project demonstrated how machine learning can be used to detect credit card fraud.
The process involved cleaning and preparing data, training several models, and using them to test and evaluate performance. The results showed that Random Forest and XGBoost models were very accurate.

> **Final Verdict:**
> The best model was XGBoost, which achieved a high ROC-AUC score of 0.99 and an accuracy of 99.9%, showing strong ability to detect fraudulent transactions.

**12.
References**

1.
Kaggle Dataset — [Credit Card Fraud Detection](https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud)
2.
Dal Pozzolo et al. (2015). *Calibrating Probability with Undersampling for Unbalanced Classification.*
3.
Scikit-learn Documentation — https://scikit-learn.org
4.
XGBoost Documentation — https://xgboost.readthedocs.io

References