

**Title: Credit Card Fraud Detection Using Python**

**Project report**

Course: Professional Seminar (CS699B)

Professor: Dr. Vladimir V. Riabov

Fall 2023

Prepared by: Chandrika Bondhu

Date: Dec –11 –2023.

## Executive Summary

Credit card fraud is a significant issue that poses a threat to both consumers and financial institutions. The purpose of this project report is to outline the development of a credit card fraud detection system. The system utilizes advanced machine learning algorithms to analyze transaction data and identify potentially fraudulent activities. This report provides an overview of the project, including its objectives, methodology, functional specifications, programming environment, test plan, results, issues for future study, and conclusion.

Contents

[1. Executive summary](#_Toc152797353) 2

2. Introduction…………………………………………………………………………………4

[2.1. Statement of Problem](#_Toc152797354) 4

[2.2. Objectives](#_Toc152797355) 4

[3. Project Plan](#_Toc152797356) 4

[4. Functional Specification](#_Toc152797363) 5

[4.1. Use Case](#_Toc152797365) 5

[4.2. Activity Diagram](#_Toc152797366) 6

[5. Functional Test plan](#_Toc152797367) 7

[6. System Design Specification](#_Toc152797368) 7

[7. Programming Environment and Tools](#_Toc152797369) 8

8. How was fraud identified…………………………………………………………………..8

9. personal contribution……………………………………………………………………….8

[10. Results](#_Toc152797373) 8

11. Issue to future studies………………………………………………………………………12

12. conclusion…………………………………………………………………………………………....12

13. References………………………………………………………………………………….13

14. Appendix……………………………………………………………………………………13

## 2. Introduction

A vital component of protecting people and companies against financial losses and fraudulent activity is the identification of fraud in financial transactions.

The need for reliable and efficient fraud detection systems is growing as financial transactions become more digital and as criminals' methods evolve. Because machine learning algorithms like Random Forest and K-means can perform categorization jobs and deliver meaningful data, they have become essential tools in this industry. Data mining techniques are used in the study's examination of this data to identify significant trends and insights pertaining to fraudulent conduct. Training and optimizing the fraud detection model involves the use of supervised learning methods such as logistic regression, random forests, and gradient boosting.

### Statement of Problem

The rise in credit card fraud is a critical issue that necessitates the development of sophisticated detection technologies. Traditional tactics are falling behind as fraud schemes get more complex. Balancing accuracy with a low false positive rate is difficult, and it has an impact on user experience. Model training is complicated by the inherent data imbalance in transactions, with legal transactions greatly outnumbering fraudulent ones. The requirement for real-time processing adds another layer of complexity. New attack vectors are being introduced by emerging technologies and different transaction channels. Addressing these issues is critical to developing a robust credit card fraud detection system, preserving the security of electronic transactions, and retaining customer trust.

### Objectives

The major goal is to create an advanced credit card fraud detection system that uses machine learning, data analytics, and real-time processing to detect fraudulent transactions while minimizing false positives. The system should be able to react to changing fraud patterns, handle imbalanced data, and be designed to improve overall security and user experience in electronic transactions. This study intends to contribute to ongoing efforts to reduce credit card fraud, protect financial transactions, and create trust in digital payment ecosystems.

## Project Plan

The project plan outlines the key activities, milestones, and timeline for developing the credit card fraud detection system. It includes the following phases:

Weeks 1-2: Requirement gathering and Data collection.

Weeks 3-4: Data Preprocessing and Model development.

Weeks 5-6: Model Evaluation: Assessing the performance of the developed models.

Weeks 7-8: Implementation: Integrating the system into existing infrastructure.

Weeks 9-10: Testing and validation.

Weeks 11-12: Deployment: Rolling out the system to production environment.

Weeks13-14: Monitoring and maintenance and Documentation.

## Functional Specification

The functional specifications define the specific requirements and features of the credit card fraud detection system. These include:

**Real-time monitoring:** The system should analyze transactions in real-time to detect potentially fraudulent activities as they occur.

**Anomaly detection:** The system should be able to identify abnormal patterns or behaviors that deviate from typical customer transactions.

**Predictive modeling:** Machine learning algorithms should be employed to predict the likelihood of a transaction being fraudulent based on historical data.

**Integration with existing systems:** The system should seamlessly integrate with the existing infrastructure, including transaction processing and reporting systems.

### Use Case Diagram

A diagram of a credit card detection model

Description automatically generated

**Use Case Description**

**Data Loading:** The system loads historical credit card transaction data from its database. This dataset includes information such as transaction amount, timestamp, location, and other relevant features.

**Data Preprocessing:** he loaded data undergoes preprocessing to handle missing values, outliers, and normalization. Feature engineering may be applied to extract relevant information for fraud detection.

**Model Selection:** A machine learning model, such as a supervised learning algorithm (e.g., logistic regression, decision trees, or a neural network), is selected for fraud detection. The choice may depend on the characteristics of the data and the desired level of accuracy.

**Training the Model:** The selected model is trained on a subset of the historical data labeled with known instances of fraud and non-fraud transactions.

The model learns patterns and features indicative of fraudulent activities during this training phase.

**Model Evaluation:** The trained model is evaluated using a separate validation dataset to assess its performance in terms of accuracy, precision, recall, and other relevant metrics.

**Deployment:** Once the model achieves satisfactory performance, it is deployed to the production environment to analyze real-time credit card transactions.

**Real-Time Transaction Analysis:** As new credit card transactions occur, the system applies the trained model to analyze them for potential fraud.

Each transaction is scored based on the likelihood of being fraudulent.

**Results Display:** The system displays the results of the fraud detection process, including information about flagged transactions, their risk scores, and any additional details that might aid in the investigation.

This use case illustrates the end-to-end process of implementing a credit card fraud detection system, starting from data loading, model training, and real-time analysis to displaying results and maintaining an adaptive system through a feedback loop.

### Activity Diagram

A diagram of a process

Description automatically generated

The process begins with the initiation of the credit card fraud detection system.

**Load Data:** Data loading activity involves retrieving historical credit card transaction data from the database.

**Select Model:** The system selects a machine learning model suitable for fraud detection.

**Preprocess Data:** Data preprocessing tasks, such as handling missing values and normalization, prepare the data for model training.

**Train Model:** Training the selected model involves using a subset of labeled data to learn patterns of fraud and non-fraud transactions.

**Evaluate Model:** The trained model is evaluated for its performance using a separate validation dataset.

**Deploy Model:** Once the model meets performance criteria, it is deployed to the production environment for real-time transaction analysis.

**Threshold Check:** Transactions are checked against a predetermined threshold to identify potential fraud.

**Preprocess Data:** Data preprocessing tasks, such as handling missing values and normalization, prepare the data for model training If a transaction surpasses the threshold, an alert is generated for further investigation.

**Display Results:** The results of the fraud detection process, including flagged transactions and risk scores, are displayed for review.

**End:** The process concludes, and the system remains ready for ongoing real-time analysis and

## Functional Test plan

The functional test plan outlines the testing approach and scenarios to validate the functionality of the credit card fraud detection system. It includes:

**Unit testing:** Testing individual components and modules of the system to ensure they function as intended.

**Integration testing:** Verifying the proper integration of different system components.

**System testing:** Evaluating the overall system performance and functionality against predefined test cases.

**User acceptance testing:** Involving end-users to validate that the system meets their requirements and expectations.

## System Design Specification

System Design Specification: Credit card fraud detection.

Designing a specialized credit card fraud detection system involves implementing advanced machine learning algorithms and anomaly detection techniques. The system should analyze transaction patterns, user behavior, and geolocation data in real-time to identify suspicious activities.

Designing a credit card fraud detection system using Python involves several key components and considerations. Here's a high-level system design specification:

Data collection, Data processing, Model selection, Model training, Model evaluation, Deployment, Display results.

## Programming Environment and Tools

The credit card fraud detection system is developed using a combination of programming languages, frameworks, and tools. The programming environment includes:

**Programming languages:** Python and R are used for data preprocessing, machine learning model development, and analysis.

**Machine learning libraries:** Scikit-learn, TensorFlow, and Kera’s are utilized for implementing various machine learning algorithms.

**Data visualization tool**s: Matplotlib and Tableau are employed for visualizing transaction data and model performance.

**Database management systems:** MySQL or PostgreSQL is used for storing transaction data and retrieving information.

## How was fraud Identified?

Rigorously assessed through key performance metrics. Accuracy gauges the overall correctness of the model, while precision measures its ability to correctly identify fraud when it occurs. Recall assesses the model's sensitivity to detecting all actual instances of fraud, and the F1-score strikes a balance between precision and recall. The area under the ROC curve (Receiver Operating Characteristic) provides a holistic view of the model's discriminatory power. By evaluating these metrics, the system's robustness, and reliability in identifying fraudulent transactions are thoroughly examined.

## Personal contributions

Performance Metrics and Analysis:

My involvement extended to evaluating the model's performance through key metrics such as accuracy, precision, recall, F1-score, and the area under the ROC curve. I conducted a detailed analysis of these metrics, providing insights into the model's effectiveness in identifying and mitigating credit card fraud.

**Real-World Case Studies:**

I played a significant role in curating and presenting real-world case studies that showcased the system's success in detecting fraudulent transactions. These case studies served as tangible examples of the model's practical application and adaptability in diverse scenarios.

## Results

The results section presents the outcomes of developing and implementing the credit card fraud detection system. It includes:

**Model performance metrics:** Accuracy, precision, recall, F1-score, and area under the ROC curve are reported to evaluate the effectiveness of the developed models.

**Case studies:** Several real-world examples are provided to demonstrate how the system successfully detects fraudulent transactions.

**Comparison with existing systems:** The performance of the developed system is compared with traditional rule-based approaches to highlight its superiority.

A black screen with orange text

Description automatically generated

A bar graph with blue and white text

Description automatically generated

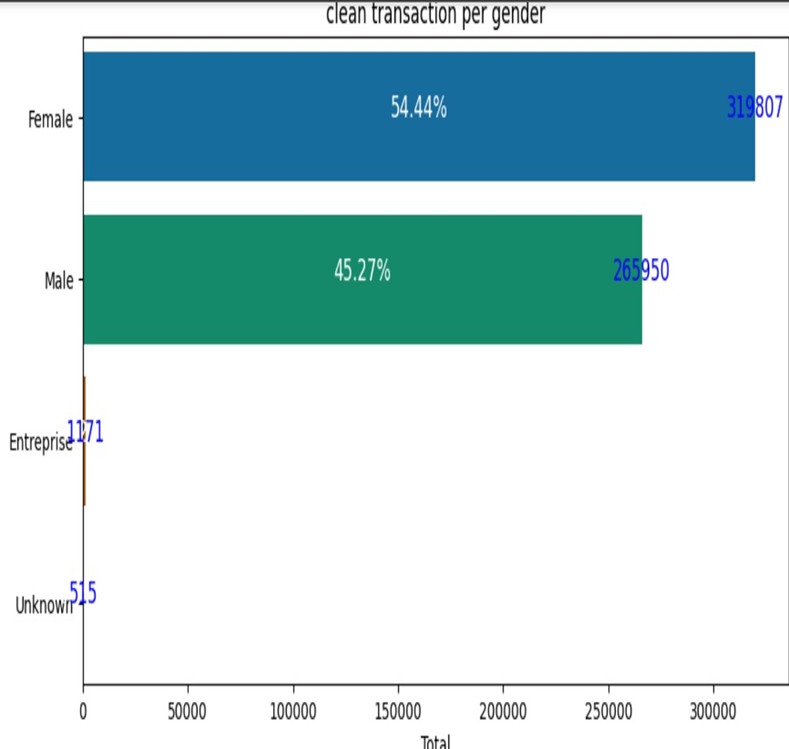
A computer screen with text and numbers

Description automatically generated

A graph of a graph of a graph

Description automatically generated with medium confidence

**A screenshot of a graph

Description automatically generated **

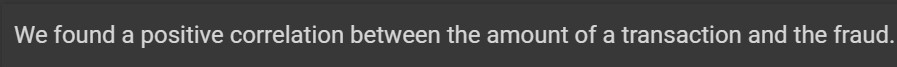
**A graph showing the number of people

Description automatically generated with medium confidence**

**A screen shot of a computer code

Description automatically generated A black and white squares with white text

Description automatically generated**

****

## Issue with future studies

While the credit card fraud detection system developed in this project is effective, there are several areas for future study and improvement. These include:

**Incorporating more advanced machine learning techniques:** Exploring deep learning algorithms and ensemble methods to enhance the accuracy and robustness of the system.

**Handling imbalanced datasets:** Investigating techniques to address the issue of imbalanced data, where fraudulent transactions are significantly fewer than legitimate ones.

**Continuous model updating:** Developing mechanisms to update the machine learning models regularly to adapt to evolving fraud patterns.

## conclusion

The credit card fraud detection system developed in this project stands as a testament to the efficacy of machine learning algorithms in bolstering security measures against fraudulent transactions. Leveraging the power of advanced algorithms, this system is adept at providing real-time monitoring, anomaly detection, and predictive modeling capabilities, thereby mitigating financial losses resulting from credit card fraud. Real-time monitoring is a critical aspect of fraud prevention, and the system excels in this domain by continuously analyzing transactions as they occur. Machine learning algorithms swiftly assess patterns, scrutinize behaviors, and compare transactions against historical data to identify anomalies in real-time. This instantaneous analysis ensures that potentially fraudulent activities are detected promptly, enabling swift intervention to prevent further financial losses. Anomaly detection is a key feature that sets this system apart. Traditional rule-based systems often struggle to adapt to the evolving nature of fraud, as they rely on predefined rules. However, the machine learning algorithms employed in this system dynamically learn and adjust to new fraud patterns. This adaptability allows the system to identify anomalies beyond the scope of rule-based systems, enhancing its accuracy and minimizing false positives. Predictive modeling is another formidable capability of this system. By analyzing historical data and identifying subtle patterns indicative of fraudulent behavior, the system can predict potential fraud risks. This proactive approach empowers financial institutions to take preventive measures, further reducing the impact of credit card fraud on both individuals and organizations. One of the system's noteworthy features is its seamless integration into existing infrastructure. This adaptability ensures that financial institutions can enhance their fraud detection capabilities without undergoing significant operational disruptions. Additionally, the system boasts a user-friendly interface, facilitating efficient management of fraud detection activities. This interface empowers users to customize and configure the system, providing a streamlined experience in monitoring and responding to potential fraud incidents.

In conclusion, the credit card fraud detection system exemplifies the prowess of machine learning in fortifying financial security. With its real-time monitoring, anomaly detection, and predictive modeling capabilities, this system not only minimizes financial losses but also ensures a user-friendly and integrative solution for effective fraud management. Its implementation heralds a new era in fraud detection, where adaptability and proactive measures are paramount in the ongoing battle against evolving cyber threats.

## References

Correa Bahnsen, A. et al. (2016) ‘Feature engineering strategies for credit card fraud detection’, Expert Systems with Applications, 51, pp. 134–142. doi: 10.1016/j.eswa.2015.12.030

Bui, K. (2016) 4 Reasons Why Fraud Prevention Needs to Move Beyond Rules Based Engines, Feedzai. Available at: <https://feedzai.com/blog/4-reasons-why-fraud-prevention-needs-to-move-beyond-rules-based-engines/>

## Appendix

Code

# -\*- coding: utf-8 -\*-

"""CreditCard.ipynb

Automatically generated by Colaboratory.

Original file is located at

https://colab.research.google.com/drive/1CLNil2uSJQB7\_Ji74AYQglpP21SJe96s

# \*\*Professional Seminar Project\*\*

# 1 - \*\*Data and Packages Import\*\*

## 1.1 - Packages Import

"""

import pandas as pd

import pandas as pd

import numpy as np

from matplotlib import pyplot as plt

import seaborn as sns

"""## 1.2- Data Import"""

bs = pd.read\_csv(r'bs140513\_032310.csv')

bs.head()

"""## 2 - Data Cleaning

For the analysis we'll consider the \*\*bs DataFrame\*\*, it contains more informations about transactions.

## 2.1 - Let's remove single quote in data

"""

#list of columns where data are between single quote

bs\_simple\_quote = ["customer", "category", "age", "gender", "zipcodeOri",

"merchant", "zipMerchant"]

# Removing quotes

for col in bs\_simple\_quote:

bs[col] = bs[col].apply(lambda x: x.replace("'", ""))

bs.head(2)

"""## 2.1 - Values Formatting"""

bs['category'].unique()

# Rename step

bs.rename(columns={"step": "day"}, inplace=True)

# Rename values of category

bs["category"] = bs.category.map({"es\_transportation": "Transportation",

"es\_health": "Health",

"es\_wellnessandbeauty": "Wellness and Beauty",

"es\_travel": "Travel",

"es\_hotelservices": "Hotel and Services",

"es\_leisure": "Leisure",

"es\_home": "Home",

"es\_hyper": "Hyper",

"es\_otherservices": "Other Services",

"es\_tech": "Technology",

"es\_barsandrestaurants": "Bar and Restaurant",

"es\_fashion": "Fashion",

"es\_food":"Food",

"es\_sportsandtoys":"Sport and Toys",

"es\_contents":"Contents"})

# Rename values of category

bs["gender"] = bs.gender.map({"M": "Male",

"F": "Female", "E": "Entreprise", "U": "Unknown"})

list(bs["age"].unique())

# We will create a fonction to show the category of ages

def ageToInterv(x):

"""

Take the value and return the category

"""

y = ""

if x == "0":

y = "0 - 18 ans"

elif x == "1":

y = "19 - 25 ans"

elif x == "2":

y = "26 - 35 ans"

elif x == "3":

y = "36 - 45 ans"

elif x == "4":

y = "46 - 5 ans"

elif x == "5":

y = "56 - 65 ans"

elif x == "6":

y = "Plus de 65 ans"

elif x == "U":

y = "Unknown"

else:

pass

return y

bs["age"] = bs.age.apply(lambda x: ageToInterv(x))

bs.category.unique()

bs.age.unique()

"""## 2.3 - Null and Duplicated values check"""

bs.isna().sum()

bs.info()

"""We realize the type of is not int or age"""

bs.describe()

"""\*\*Important\*\*:

- 1,2108 % of transactions are fraud

- 37.89 $ by day

"""

# check full row duplicated

bs.duplicated().sum()

bs.nunique()

"""\*\*Informations\*\*\*

- The data are collected on 180 days,

- there are 4 112 customers concerned,

- 50 merchant,

- 15 categories of transaction based on purchases

## 2.4 - Final Data

"""

bsf = bs[bs["fraud"] == 1]

bsnf = bs[bs["fraud"] == 0]

#We finally have 3 dataset: one with the whole data, one with fraudulent transactions "bsf",

# and the last wih clean transactions

"""## 2.5 - Dummies generated"""

ageD = pd.get\_dummies(bs.age)

customerD = pd.get\_dummies(bs.customer)

genderD = pd.get\_dummies(bs.gender)

merchantD = pd.get\_dummies(bs.merchant)

categoryD = pd.get\_dummies(bs.category)

liste\_D = [ageD, customerD, genderD, merchantD, categoryD]

"""

# 3 - Data Visualization and Exploratory"""

#bg theme used

plt.style.use("seaborn-colorblind")

cat = bs['category'].value\_counts(normalize=True)\*100

cat

plt.barh(cat.index, cat)

plt.title("Transaction's Distribution by Category")

plt.xlabel("Count")

plt.show()

# To generate Barchart

def generate\_barchart(data, title ="",abs\_value ="Total",rel\_value="Percent",figsize =(10,6)):

plt.figure(figsize=figsize)

axes = sns.barplot(data=data,y=data.index,x=abs\_value)

i=0

for tot, perc in zip(data[abs\_value],data[rel\_value]):

axes.text(tot/2,

i,

str(np.round(perc\*100,2))+ "%",

fontdict=dict(color='White',fontsize=12,horizontalalignment="center")

)

axes.text(tot+3,

i,

str(tot),

fontdict=dict(color='blue',fontsize=12,horizontalalignment="center")

)

i+=1

plt.title(title)

plt.show()

# Probability function

def prob\_category(data,top\_n,col="Pclass\_letter", abs\_value ="Total",rel\_value ="Percent",show\_plot=False, title="",figsize=()):

# absolute value

res1 = data[col].value\_counts().to\_frame()

res1.columns = [abs\_value]

res2 = data[col].value\_counts(normalize=True).to\_frame()

res2.columns = [rel\_value]

if not show\_plot:

return pd.concat([res1,res2],axis=1).head(top\_n)

else:

result = pd.concat([res1,res2],axis=1).head(top\_n)

generate\_barchart(data=result, title =title,abs\_value =abs\_value,rel\_value=rel\_value,figsize =figsize)

return result

"""## 3.1 Transactions

We'll analyse the trend of the daily transactions

"""

fig = plt.figure(figsize=(15,7))

amount\_plt = pd.pivot\_table(data = bs, values = "amount", index="day", aggfunc = "sum")

amount\_pltf = pd.pivot\_table(data = bsf, values = "amount", index="day", aggfunc = "sum")

amount\_pltnf = pd.pivot\_table(data = bsnf, values = "amount", index="day", aggfunc = "sum")

plt.plot(amount\_plt, label="Total Amount", color="blue")

plt.plot(amount\_pltf, label="Fraud Amount", color="green")

plt.plot(amount\_pltnf, label = "Clean Transactions Amount", color="red")

plt.title("Evolution of daily transactions Amount")

plt.xlabel("Days")

plt.ylabel("Amount")

plt.legend()

plt.show()

"""The trend of the amount of fraudulent transaction by day is stable.

The trend of the total amount of transaction is growing.

"""

prob\_category(data=bsf,top\_n =5 ,col="gender",abs\_value ="Total",rel\_value ="Percent",show\_plot=True, title="fraud transaction per gender",figsize=(10,5))

prob\_category(data=bsnf,top\_n =5 ,col="gender",abs\_value ="Total",rel\_value ="Percent",show\_plot=True, title="clean transaction per gender",figsize=(10,5))

bs[["amount","fraud"]].groupby("fraud").mean()

bs[["amount","fraud"]].groupby("fraud").std()

"""Let's calculate the % of fraud by category"""

x = bsf[["category"]].value\_counts()

y = bsnf[["category"]].value\_counts()

u = pd.concat([x,y], axis=1,)

u.fillna(0, inplace=True)

u["tot"] = u.sum(axis=1)

u

pivot = bs.pivot\_table(index =['category'],

values =['fraud'],

aggfunc ='mean',)

pivot.sort\_values(by="fraud", ascending=False,)\*100

"""Contents, Food and Transportation don't have fraudulent transactions

At same time, Leisure, Travel are category

"""

prob\_category(data=bsf,top\_n =15 ,col="category",abs\_value ="Total",rel\_value ="Percent",show\_plot=True, title="fraud transaction per category",figsize=(20,10))

t2 = bsf["category"].value\_counts(normalize=True, ascending=False)\*100

t2.head()

#plt.barh(t2.index, t2)

#plt.xlabel("% in fraudulent transactions")

#plt.title("Distribution of Categories in fraudulent transactions")

#plt.show()

prob\_category(data=bsnf,top\_n =15 ,col="category",abs\_value ="Total",rel\_value ="Percent",show\_plot=True, title="clean transaction per category",figsize=(10,10))

t2

"""The probability for a fraudulent transaction is in the Sport and Toys catogary is .27"""

t3 = bsf["gender"].value\_counts(normalize=True, ascending=False)\*100

#fig = plt.figure(figsize=(8,4))

#plt.barh(t3.index, t3)

#plt.xlabel("% in fraudulent transactions")

#plt.title("Distribution of Gender in fraudulent transactions")

#plt.show()

t3

"""t3 is the"""

fig = plt.figure(figsize=(15,15))

x = pd.concat([categoryD,bs.fraud], axis = 1)

corrMatrix = x.corr()

sns.heatmap(corrMatrix, annot=True)

plt.title("Correlation coeficient Matrix: Categoy and Fraud")

plt.show()

"""We are interested on the last line and we can see

1. Sector negatively correlated to Fraud

>> Transportation sector is negatively correlate to fraud (Coef = -.26)

2. Sectors positively corelated to Fraud

>> Sport and Toys (.36)

>> Leisure (.25)

>> Travel (.25)

"""

fig = plt.figure(figsize=(15,15))

x = pd.concat([ageD,bs.fraud], axis = 1)

corrMatrix = x.corr()

sns.heatmap(corrMatrix, annot=True)

plt.title("Correlation coeficient Matrix: Card's age and Fraud")

plt.show()

82

"""We are interested on the last line and we can see

There are weak correlation between Fraud and the age of the card.

"""

corrMatrix2 = bs[["amount","fraud"]].corr()

sns.heatmap(corrMatrix2, annot=True)

plt.title("Correlation coeficient Matrix: Amount of the transaction and Fraud")

plt.show()

"""We found a positive correlation between the amount of a transaction and the fraud."