Credit Card Fraud Detection Using Machine Learning

A PROJECT REPORT

Submitted by

Archit Dogra (21BCS6215)

Pravinkumar Gohil (21BCS6218)

Aryan Verma (21BCS6199)

in partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE WITH SPECIALIZATION IN ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

Chandigarh University

Nov 2024

CHANDIGARH UNIVERSITY

Discover. Learn. Empower.

BONAFIDE CERTIFICATE

Certified that this project report **“Machine-Learning Based System for Credit Card fraud Detection”** is the bonafide work of **“Archit Dogra, Aryan Verma and Pravin Kumar Gohil”** who carried out the project work under my/our supervision.

SIGNATURE	**SIGNATURE**
(Mrs.PriyankaKaushik)	**(Ms. Tanvi)**
Head of Department	**SUPERVISOR**
AIT-CSE	**AIT-CSE**

Submitted for the project viva-voce examination to be held on 14th November, 2024

INTERNAL EXAMINER **EXTERNAL EXAMINER**

TABLE OF CONTENT

List of figures

ABSTRACT

The AI-Driven Financial Fraud Detection System is a comprehensive solution to address the escalating problem of fraud in digital and credit card transactions, which has intensified with the growth of digital payments. As fraud becomes increasingly complex, this project introduces an advanced AI model that leverages machine learning to detect and prevent fraudulent activities in real time. Using real-world credit card data with both legitimate and fraudulent transactions, the system employs extensive exploratory data analysis (EDA) to identify and understand fraud patterns, addressing class imbalance issues by using undersampling techniques to create a balanced dataset. The initial model is based on logistic regression, chosen for its simplicity, interpretability, and efficiency in binary classification, allowing the system to predict fraud likelihood based on specific transaction characteristics. The model's effectiveness is evaluated through metrics like accuracy, precision, recall, and F1 score, ensuring high standards for accurately distinguishing between fraudulent and legitimate transactions. Additionally, the system includes a real-time adaptability feature, making it scalable and responsive to evolving fraud tactics—a valuable asset for corporate finance. While logistic regression proves effective, the project acknowledges that more complex models, such as decision trees, gradient boosting, or neural networks, may enhance detection capabilities in future iterations. This system demonstrates how machine learning, combined with robust data processing, can provide a practical, scalable solution for detecting fraud. By continuously refining its approach, this AI-driven system not only addresses current fraud detection challenges but also sets the foundation for a resilient, adaptable framework that evolves with advancements in digital finance, ultimately strengthening financial security and customer trust.

GRAPHICAL ABSTRACT

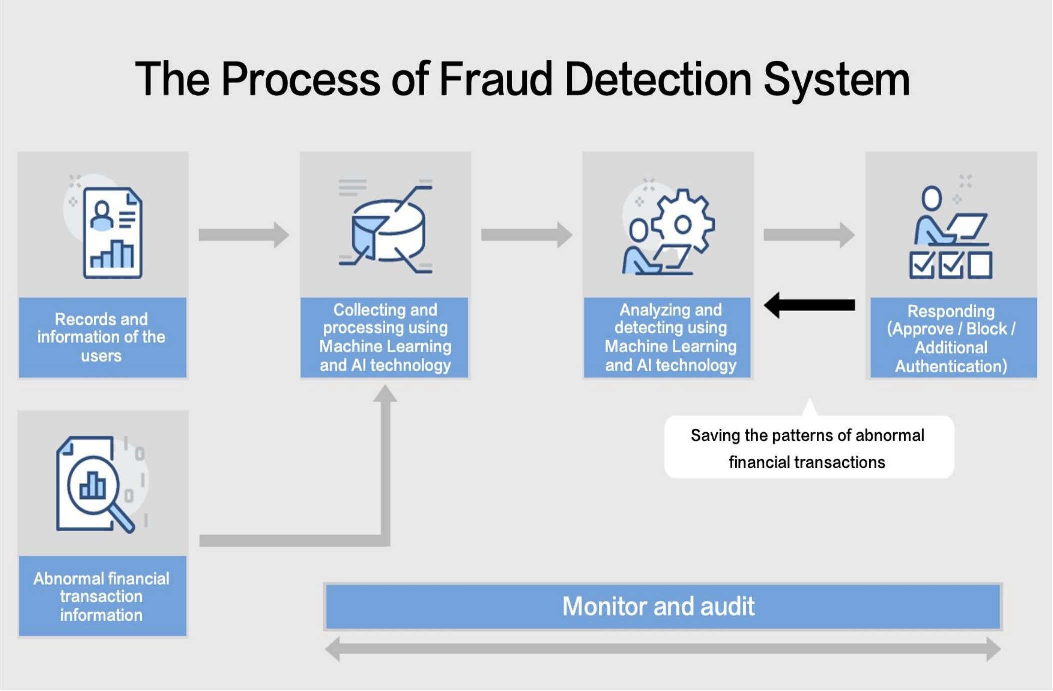

The Process of Fraud Detection System
Records and information of the users
Collecting and processing using Machine Learning and AI technology
Analyzing and detecting using Machine Learning and AI technology
Responding (Approve / Block / Additional Authentication)
Saving the patterns of abnormal financial transactions
Abnormal financial transaction information
Monitor and audit

Figure 0: Graphical Abstract

ABBREVIATIONS

1. ANN- Artificial Neural Network
2. DL-Deep Learning
3. ML- Machine Learning
4. AI- Artificial Intelligence
5. PCA- Principle Component Analysis
6. ICA- Independent Component Analysis
7. LDA- Linear Discriminant Analysis

CHAPTER 1: INTRODUCTION

As the financial industry undergoes a rapid transformation toward digitalization, the incidence of fraudulent activities has also seen a significant uptick, posing severe risks for both financial institutions and their customers. With the widespread adoption of online transactions and digital payment methods, sophisticated schemes such as identity theft, unauthorized transaction fraud, money laundering, and phishing scams have become prevalent, exploiting the vulnerabilities inherent in digital financial systems. The ability to detect and prevent these malicious activities in real time is crucial for safeguarding financial systems, minimizing financial losses, preserving customer trust, and ensuring regulatory compliance in an increasingly connected world. Traditional fraud detection systems, while effective to an extent, often rely on rule-based frameworks that are based on predefined conditions. Given the dynamic and high-stakes nature of financial fraud, there is an urgent need for more adaptive, intelligent systems that can autonomously learn and respond to new fraud patterns as they emerge.

This report details the development of an AI-Driven Financial Fraud Detection System, an advanced solution that leverages artificial intelligence (AI) and machine learning (ML) technologies to proactively identify and prevent fraudulent activities within financial transactions. By utilizing the power of AI and ML, this system aims to transform the traditional fraud detection process by automating the detection of unusual patterns and continuously learning from historical data to adapt to new types of fraud. The goal is to create a model that can accurately differentiate between legitimate transactions and fraudulent ones, improving both the precision and recall of the fraud detection process, ultimately reducing false positives and increasing detection rates.

The project is organized into three main phases: data collection and preparation, model development and training, and system integration and deployment. Each phase plays a critical role in ensuring that the final product is a robust, scalable, and high-performing solution capable of meeting the demands of real-time fraud detection. In the first phase, data collection and preparation, the system gathers real-world transaction data from credit card records, including both legitimate and fraudulent transactions. This phase involves rigorous data preprocessing and exploratory data analysis (EDA) to gain insights into transaction patterns, understand the underlying data structure, and identify any inherent class imbalances. Since fraud data is typically scarce compared to legitimate transaction data, methods like undersampling or oversampling are used to balance the dataset, ensuring the model can effectively learn from both types of transactions without bias.

In the second phase, model development and training, the machine learning model is created and refined. A logistic regression model is initially chosen due to its interpretability, efficiency, and suitability for binary classification tasks. Logistic regression enables the model to identify the likelihood of a transaction being fraudulent based on features such as transaction amount, time, location, and other anonymized details. This initial model serves as a foundation, offering insights into how various features contribute to fraud detection. To evaluate the model's performance, metrics such as accuracy, confusion matrix, precision, recall, and F1 score are used, providing a comprehensive assessment of its ability to correctly identify fraudulent and legitimate transactions. While logistic regression is effective, the project also considers more complex models, such as decision trees, gradient boosting, and neural networks, to enhance detection capabilities and further reduce false positives.

In the final phase, system integration and deployment, the AI-driven fraud detection system is prepared for implementation within real-world financial environments. This phase includes integrating the model into a scalable infrastructure that can handle the high volume of transactions processed by financial institutions. Additionally, the system is equipped with real-time adaptability, allowing it to learn from newly incoming data and adjust its detection mechanisms accordingly. This adaptability ensures that the system remains effective even as fraud tactics evolve, providing financial institutions with a continuously improving solution. This phase also involves rigorous testing to ensure the system's stability, reliability, and compliance with industry standards for financial security.

Overall, this AI-driven fraud detection system represents a significant advancement over traditional rule-based approaches, providing a dynamic, scalable, and intelligent solution to the growing problem of financial fraud. By automating the detection process and leveraging ML to learn from historical data, the system can rapidly identify and respond to new fraud patterns, staying ahead of fraudsters who constantly seek to exploit digital vulnerabilities. This project underscores the transformative potential of AI and ML in financial security, demonstrating that with the right data, model, and infrastructure, financial institutions can effectively mitigate fraud risks.

1.1 MACHINE LEARNING

Machine learning is a type of artificial intelligence (AI) that involves teaching machines to learn from data, without being explicitly programmed. It is a process of training computer algorithms to identify patterns in large datasets and use those patterns to make predictions or take actions. Machine learning can be categorized into three main types: supervised learning, unsupervised learning, and reinforcement learning. In supervised learning, the machine is trained using labeled data, where the desired output is known. The goal is to learn a function that maps input data to output data. In unsupervised learning, the machine is trained using unlabeled data, and the goal is to find patterns or structures in the data. Reinforcement learning involves training a machine to make decisions based on feedback from the environment, to maximize a reward signal. Machine learning has numerous applications, including natural language processing, image and speech recognition, fraud detection, recommendation systems, and autonomous vehicles. It is a rapidly growing field that is transforming the way we live and work.

Machine learning techniques for feature extraction can be broadly categorized into two categories: unsupervised and supervised methods.

Unsupervised Feature Extraction:

Unsupervised feature extraction methods do not require labeled data and attempt to discover patterns or structure in the data without prior knowledge of class labels. The most common unsupervised feature extraction methods are:

a.Principal Component Analysis (PCA): PCA is a technique that reduces the dimensionality of the data by finding a set of orthogonal directions that explain the maximum amount of variance in the data.

b.Independent Component Analysis (ICA): ICA is a technique that separates a multivariate signal into independent, non-Gaussian components.

Non-negative Matrix Factorization (NMF): NMF is a technique that factorizes a non- negative data matrix into two non-negative matrices representing the basis and coefficients of the data.

a. t-distributed Stochastic Neighbor Embedding (t-SNE): t-SNE is a technique that maps high- dimensional data into a low-dimensional space while preserving the pairwise distances between data points.

Supervised Feature Extraction:
Supervised feature extraction methods require labeled data and use the labels to guide the feature extraction process. The most common supervised feature extraction methods are:
a. Linear Discriminant Analysis (LDA): LDA is a technique that reduces the dimensionality of the data while maximizing the separability between classes.

b. Fisher Vector: Fisher vector is a technique that represents an image as a set of descriptors derived from local features, which are then encoded into a vector using a generative model.

c. Convolutional Neural Networks (CNNs): CNNs are a class of deep neural networks that are commonly used for image recognition and feature extraction. They can be trained to extract features from images in a supervised manner.

d. Recursive Neural Networks (RNNs): RNNs are a type of neural network that are designed to process sequential data. They can be trained to extract features from sequential data such as speech or text.

1.2 DEEP LEARNING

Deep learning is a subfield of machine learning that involves the use of neural networks with multiple layers to analyze and learn from data. The term "deep" refers to the fact that these neural networks have many layers, often ranging from tens to hundreds or even thousands of layers.

Deep learning models are capable of learning complex representations of data, allowing them to perform tasks such as image recognition, natural language processing, speech recognition, and more. These models are trained using large amounts of data and require powerful computing resources, such as GPUs, to efficiently process the data.

Deep learning has led to significant advances in various fields, including computer vision, speech recognition, natural language processing, and autonomous vehicles. It is a rapidly evolving field, and researchers are constantly developing new techniques and architectures to improve the accuracy and efficiency of deep learning models.

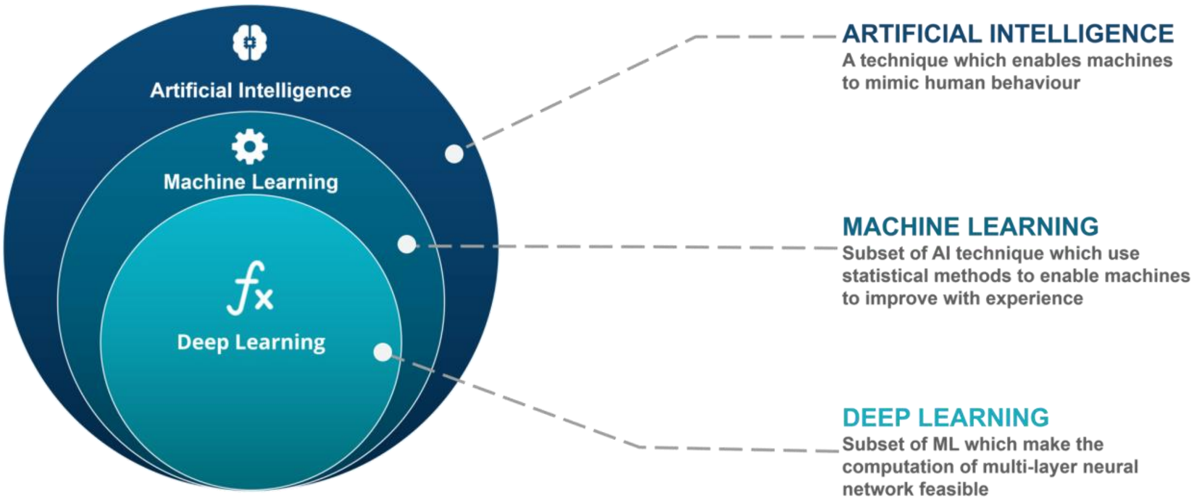

Artificial Intelligence
Machine Learning
Deep Learning
ARTIFICIAL INTELLIGENCE
A technique which enables machines to mimic human behaviour
MACHINE LEARNING
Subset of AI technique which use statistical methods to enable machines to improve with experience
DEEP LEARNING
Subset of ML which make the computation of multi-layer neural network feasible

Figure 1: Deep Learning

1.3 Artificial Neural Network

An Artificial Neural Network (ANN) is a computational model inspired by the way biological neural networks in the human brain work. It is used for various tasks such as pattern recognition, classification, regression, and decision-making. .

Here are some key concepts associated with Artificial Neural Networks:

Neurons (Nodes): In an ANN, neurons are units that receive input, perform a computation, and produce an output. Neurons are organized into layers: an input layer, one or more hidden layers, and an output layer.
Connections (Weights): Neurons are connected by links, which have associated weights. These weights determine the strength of the connection between neurons. During training, the network learns to adjust these weights to minimize the difference between the predicted outputand the actual output.

Activation Function: Each neuron typically has an activation function that determines its output based on the weighted sum of its inputs. Common activation functions include sigmoid, hyperbolic tangent (tanh), and rectified linear unit (ReLU).

Feedforward and Backpropagation: In a feedforward neural network, information travels in one direction—from the input layer through the hidden layers to the output layer. Backpropagation is the training algorithm used to adjust the weights in the network based on the error between the predicted output and the actual output.

Training Data: ANNs require a dataset for training. The dataset consists of input-output pairs, and the network learns to generalize patterns from the training data to make predictions on new, unseen data.

Loss Function: The loss function quantifies the difference between the predicted output and the actual output. During training, the goal is to minimize this loss by adjusting the weights in the network.

Epochs: Training a neural network involves iterating through the entire training dataset multiple times. Each iteration is called an epoch. The number of epochs is a hyperparameter that affects the learning process.

Deep Neural Networks: ANNs with multiple hidden layers are called deep neural networks (DNNs).

Deep learning involves training deep neural networks and has been particularly successful in tasks like image recognition, natural language processing, and game playing.

Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs): These are specialized types of neural networks designed for specific tasks. CNNs are often used for image recognition, while RNNs are suitable for sequence data, such as time series or natural language.

Artificial Neural Networks have demonstrated remarkable success in various applications, but their effectiveness depends on appropriate architecture, hyperparameter tuning, and the availability of sufficient and relevant training data.

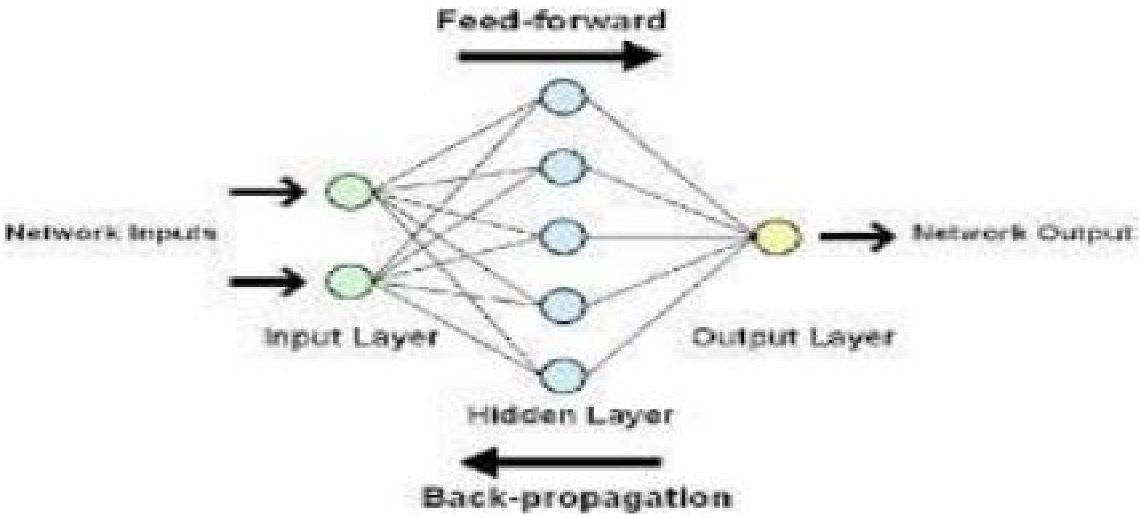


Figure 2: ANN

1.4 Drop Out Layer

In deep learning, the dropout layer is a regularization technique used to prevent overfitting. Overfitting occurs when a model becomes too complex and starts to memorize the training data instead of learning the underlying patterns.

The dropout layer works by randomly dropping out (setting to zero) a certain percentage of the input units during each training iteration. This forces the remaining units to learn to work together and share the representation of the data. During testing or inference, all units are used, but their outputs are scaled down by the dropout probability.

Dropout can be added to any layer in a neural network, but it is typically applied to the fully connected layers. It is important to note that dropout should not be used in the output layer, as it can result in unstable and inconsistent predictions.

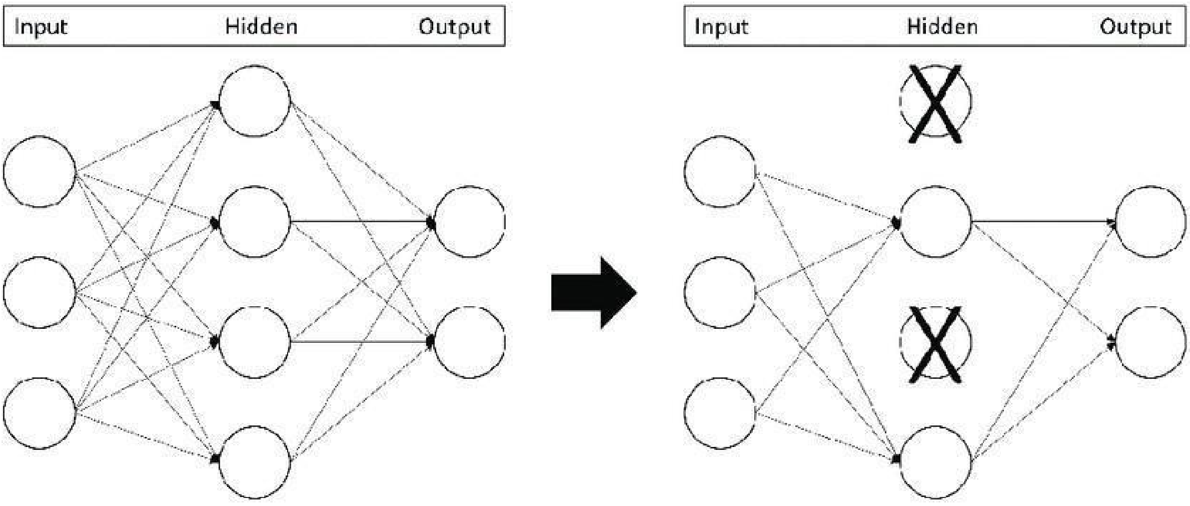


Figure 3: Drop Out layer

1.5 Logistic Regression

Logistic Regression is a statistical method used for binary classification tasks, where the goal is to predict one of two possible outcomes. Although it has "regression" in its name, logistic regression is primarily used for classification. Instead of predicting a continuous output, logistic regression predicts the probability that an instance belongs to a particular class.

Key Aspects of Logistic Regression

1. Equation: Logistic regression uses a logistic function (also called the sigmoid function) to model the probability of a binary outcome:

$P(y=1 \mid x)=11+e-(mx+b)P(y=1 \mid x)=1+e-(mx+b)1$

- y: The target variable (binary, often 0 or 1).
- x: The independent variable(s).
- m and b: Model coefficients (weights and intercept) that are learned from the data.

2. Sigmoid Curve: The logistic function maps any real-valued number into the range (0, 1), making it useful for probability prediction. The S-curve of the sigmoid function allows logistic regression to model the likelihood of binary classes.

3. Interpretability: Logistic regression coefficients indicate the direction and strength of the association between each independent variable and the probability of the target class.

4. Applications: It's widely used in binary classification problems, such as spam detection, medical diagnosis, and fraud detection.

5. Assumptions:
    - Binary or categorical outcome: Logistic regression is best suited for binary or categorical classification.
    - Linear relationship with log odds: Independent variables should have a linear relationship with the log odds of the outcome.

I'll create a plot to illustrate the sigmoid function and its typical S-shaped curve as used in logistic regression. It seems there was an issue generating the plot again. I'll try another approach to fix this.

It seems there's a recurring issue with generating the plot. Here's an explanation of what the plot would look like for logistic regression:

The sigmoid function plot used in logistic regression has an S-shaped curve that maps input values (from -10 to 10, for example) to a range between 0 and 1. The y-axis represents the probability P(y=1 ∣ x)$P(y=1 \mid x)$, and the x-axis represents the values of the independent variable.

- When x=0$x=0$, the probability is 0.5, which serves as the decision boundary for classification.
- For values of xx much greater than 0, the probability approaches 1.
- For values of xx much less than 0, the probability approaches 0.

The decision boundary at y=0.5$y=0.5$ is typically used to classify outputs into two classes, making logistic regression a powerful tool for binary classification tasks.

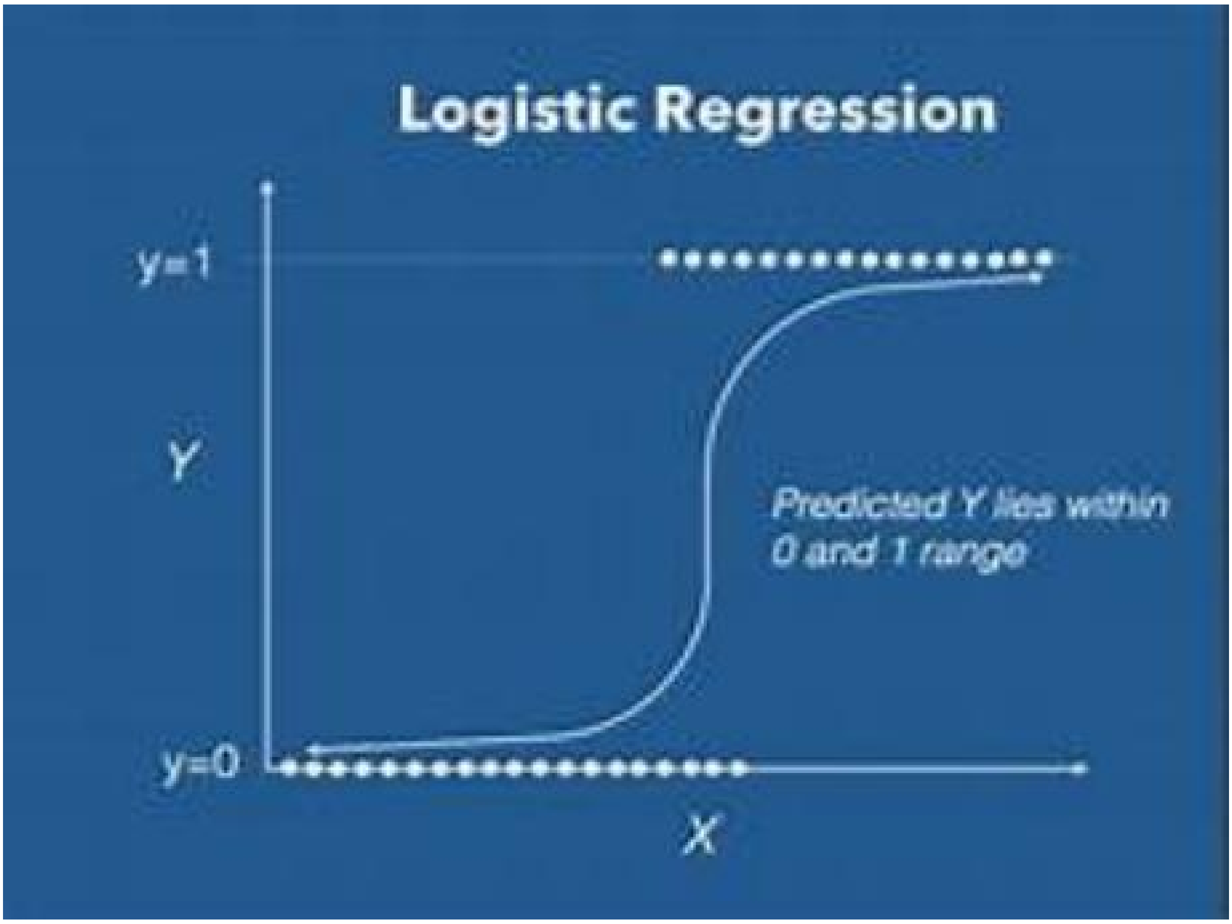


Figure 4: Logistic Regression

1.6 Activation Functions

Sigmoid Activation Function:

Concept: Squeezes the input values between 0 and 1, producing an output that can be interpreted as a probability. Commonly used in the output layer of binary classification models.

Hyperbolic Tangent (tanh) Activation Function:

Concept: Similar to the sigmoid but with a range between -1 and 1. Useful in hidden layers to capture negative and positive relationships in data.

Rectified Linear Unit (ReLU) Activation Function:

Concept: Outputs the input directly if it is positive; otherwise, it outputs zero. Efficient and helps mitigate the vanishing gradient problem. Commonly used in hidden layers.

Leaky Rectified Linear Unit (Leaky ReLU) Activation Function:

Concept: Similar to ReLU but allows a small, non-zero gradient for negative inputs. Addresses the "dying ReLU" problem where neurons can become inactive during training.

Parametric Rectified Linear Unit (PReLU) Activation Function:

Concept: A variant of Leaky ReLU where the slope of the negative part is a learnable parameter, allowing the network to adaptively learn the optimal slope during training.

Exponential Linear Unit (ELU) Activation Function:

Concept: Similar to ReLU for positive inputs, but smoothly saturates negative inputs with an exponential decay. Provides a smoother gradient for negative values compared to ReLU.

These activation functions introduce non-linearities into the neural network, enabling it to learn and approximate complex relationships within the data. The choice of activation function depends on the specific characteristics of the task, the properties of the data, and empirical observations during model training.

1.7 Checkpoint

A model checkpoint in deep learning is a saved snapshot of a neural network model's parameters at a particular point during training. Checkpoints are used to keep track of a model's progress during training and allow you to resume training from where you left off if training is interrupted or stopped.

During training, the model's weights and biases are adjusted to minimize the error between the predicted output and the true output. The checkpoint saves the current state of these weights and biases, along with other training parameters, such as the optimizer's state,learning rate, and epoch number.

Saving checkpoints at regular intervals ensures that you have access to the best-performing version of the model during training and can also help prevent the model from overfitting. Checkpoints can be saved manually, or automatically using a callback function in the training loop.

Once training is complete, you can use the saved checkpoint to load the trained model's parameters and use it for inference or further training

1.8 Problem Definition

Objective: To develop a machine learning model that accurately detects fraudulent credit card transactions. The project uses a logistic regression model to classify transactions as either "Legitimate" or "Fraudulent," aiming to assist in reducing losses caused by fraud.

Background: With the increase in online transactions, credit card fraud has become a prevalent issue, leading to significant financial losses. Identifying fraudulent transactions among legitimate ones is challenging due to the class imbalance: fraudulent transactions are much rarer than legitimate ones. This project addresses this imbalance by implementing under-sampling techniques and training a logistic regression model to identify potentially fraudulent transactions.

Dataset: The dataset contains credit card transactions, with each transaction characterized by anonymized features and an 'Amount' column. The target variable, 'Class,' is binary, indicating whether a transaction is legitimate (0) or fraudulent (1). Key steps include data exploration, feature engineering, and model evaluation.

Approach:

1. **Data Exploration**: Analyze the dataset for missing values, class distribution, and transaction amount differences between legitimate and fraudulent transactions. Visualizations (such as box plots and heatmaps) help in understanding the distribution and relationships between features.

2. **Balancing the Dataset**: Given the class imbalance, legitimate transactions are under-sampled to balance with the number of fraudulent transactions, ensuring the model receives balanced training data.

3. **Model Development**: A logistic regression model is trained on the balanced dataset, aiming for a balance between accuracy and interpretability. The model's performance is evaluated using metrics such as accuracy, confusion matrix, and classification report.

4. **User Interaction**: A widget allows users to upload new transaction files for prediction. The model predicts each transaction's likelihood of being legitimate or fraudulent, providing a practical interface for testing the model on real-world data.
5.

Expected Outcome: The model should achieve a high accuracy in identifying fraudulent transactions while minimizing false positives.

1.9 Problem Overview

The financial sector has seen exponential growth in digital transactions due to the widespread adoption of online banking, mobile payments, and e-commerce. However, this surge in digital activity has also made financial institutions more vulnerable to fraudulent activities. Fraudsters are continually developing new methods to exploit weaknesses in these systems, making fraud detection a critical challenge. The complexity and sheer volume of transactions processed every second create an environment ripe for fraud, yet difficult for traditional systems to monitor effectively.

Historically, financial fraud detection has relied on rule-based systems that flag transactions based on predefined conditions such as spending limits, transaction locations, or unusual account activities.

While these methods offer basic protection, they fall short in detecting increasingly sophisticated fraud tactics. Fraudsters have become adept at bypassing these static rules, often exploiting the time gap between suspicious activity and detection. As a result, financial institutions face significant financial losses and reputational damage when fraud goes undetected.

A major issue with traditional systems is their high rate of false positives—instances where legitimate transactions are incorrectly flagged as fraudulent. This creates friction for customers and increases operational costs for institutions that must investigate these alerts.

Conversely, false negatives—where actual fraud goes unnoticed— pose a direct threat to the financial security of both the institution and its customers. Balancing the need for timely and accurate detection while minimizing false alerts is a core challenge for fraud detection systems.

Complicating the issue further is the fact that transactional data is multi-faceted and vast. Financial institutions must process enormous amounts of data from various sources, including transaction histories, user behavior, and external factors like geographic location or time of day. Detecting subtle anomalies or emerging fraud patterns within this vast data landscape requires advanced analytical techniques, which traditional systems are not equipped to handle effectively.

The need for a more adaptive, real-time approach to fraud detection has led to the integration of artificial intelligence (AI) and machine learning (ML) into financial systems. Machine learning models can automatically learn from historical data, identify patterns of legitimate and fraudulent behavior, and adapt to new, previously unseen types of fraud. This ability to dynamically update and refine detection criteria in real-time makes AI- driven systems a promising solution to the problem of financial fraud. The goal is to enhance detection accuracy, reduce false positives, and offer a scalable solution that can adapt to the rapidly changing fraud landscape

1.10 Project Scope

The AI-Driven Financial Fraud Detection System aims to design and implement an intelligent, scalable solution to detect and prevent fraudulent financial transactions in real-time using machine learning and artificial intelligence techniques. This project encompasses the development of an advanced system capable of analyzing large datasets, identifying anomalous behavior, and flagging suspicious transactions, ultimately enhancing security for financial institutions and their customers.

The project will unfold in three primary stages. The first stage involves gathering large-scale transactional data from multiple sources, including historical transaction records, customer behavior logs, and external factors such as geographic location and time of day. This data will undergo preprocessing, cleaning, and transformation to ensure its suitability for training machine learning models. Key activities will include handling missing values, addressing imbalanced datasets, and removing noisy data to optimize model performance.

The second stage focuses on model development and training, where a variety of machine learning algorithms, including both supervised and unsupervised learning techniques, will be employed to identify fraudulent patterns. Models will be trained using historical fraud and non-fraud transaction data, with extensive testing to evaluate accuracy, precision, recall, and false positive rates, ensuring effective fraud detection while minimizing false alerts.

The final stage entails system integration and deployment. The trained machine learning model will be integrated into a real-time fraud detection system, seamlessly connecting it to live transaction monitoring systems within financial institutions' existing infrastructures. This phase will include rigorous testing in real-world scenarios to validate the system's effectiveness. Additionally, user interfaces, alerts, and reporting tools will be developed to assist fraud analysts in efficiently monitoring and managing flagged transactions. The project scope also encompasses the creation of documentation and user training materials to facilitate smooth adoption by financial institutions. The system will be designed for scalability, capable of handling increasing transaction volumes and adapting to emerging fraud tactics through ongoing model learning and updates. By integrating machine learning into the fraud detection process, the system aims to reduce false positives, improve

CHAPTER 2 LITERATURE SURVEY

2.1 Related Work

1. Mastering Fraudulent Schemes: A Unified Framework for AI-Driven US Banking Fraud Detection and Prevention by Anudeep Kotagiri[0009-0004-5103-8655](DEC 2023)

This research paper presents a comprehensive framework for AI-driven banking fraud detection and prevention in the United States. Key elements include real-time transaction monitoring, behavior analysis, and adaptive learning mechanisms. The research emphasizes the significance of an integrated AI solution in addressing the evolving landscape of banking fraud, contributing to a more secure and resilient financial ecosystem.

2. AI-POWERED FRAUD DETECTION IN BANKING: SAFEGUARDING FINANCIAL TRANSACTIONS

by Fatema Tuz Johora, Rakibul Hasan, Sayeda Farjana Farabi (June 2024)

This paper highlights the significant role of artificial intelligence in detecting fraudulent banking transactions. It presents various classification algorithms, with a particular focus on artificial neural networks, for distinguishing transaction types based on key features. While all models performed consistently, achieving AUC values above 0.9, logistic regression stood out with the highest AUC value of 0.946, emphasizing its efficacy in fraud detection. The study demonstrates the potential of AI-driven methods in effectively addressing banking fraud.

3. Financial Fraud Detection Based on Machine Learning: ASystematic Literature Review by Abdulalem Ali 1,, Shukor Abd Razak 1,2, , Siti Hajar Othman , Taiseer Abdalla Elfadil Eisa 3,Arafat Al-Dhaqm 1,* , Maged Nasser 4, Tusneem Elhassan 1, Hashim Elshafie 5 and Abdu Saif (Sept 2022) In this paper, an SLR approach is used, which is a detailed approach for gathering and analyzing all studies that focused on specific research questions [22]. It is used

to identify

and combine information that focuses on particular issues to lessen biases [17,22], provide a review with high-quality evidence, and inspect the path of reviewers' judgments and conclusions [22]. This SLR study is based on the study in [23], which covers three main stages: review planning, conducting the review, and reporting the review.

4. AI-Driven Fraud Detection Systems: A Comparative Study across Banking, Insurance, and Healthcare By Pankaj Zanke Application Architect V, Bank of America, Atlanta, GA, USA (DEC 2023)

This paper compares AI-driven fraud detection systems across banking, insurance, and healthcare, highlighting their improved accuracy, scalability, and adaptability over traditional methods. Key factors influencing effectiveness include data quality, model interpretability, and computational power. Emerging trends include real-time monitoring and explainable AI, though challenges persist in addressing algorithmic bias, privacy concerns, and regulatory compliance.

5. AI-Based Financial Transaction Monitoring and Fraud Prevention with Behaviour Prediction Jiahao Xu * , Tianyi Yang , Shikai Zhuang , Huixiang Li , Wenran Lu (July 2024)

This study investigates deep learning techniques for credit card fraud detection, comparing Isolation Forest and Autoencoder algorithms. Autoencoder improved detection accuracy to 33.6%, surpassing Isolation Forest's 26% for the top 1000 transactions. Despite deep learning's strong feature extraction the high dataset imbalance (0.17% fraud) remains a challenge. The study calls for further optimization to enhance fraud detection efficiency and stability.

6. TRANSFORMING FINTECH FRAUD DETECTION WITH ADVANCED ARTIFICIAL INTELLIGENCE ALGORITHMS by Philip Olaseni Shoetan & Babajide Tolulope Familoni (APR 2024)

AI has transformed fintech fraud detection, enhancing efficiency with real-time processing and predictive analytics. Emerging technologies like blockchain and quantum computing offer new solutions but raise concerns about data privacy, bias, and regulatory compliance. Ethical AI development, transparency, and interdisciplinary collaboration will be crucial to balancing innovation with responsible practices in advancing fraud detection capabilities in the financial sector.

,

2.2 Analysis done by using Machine Learning

Here's a breakdown of the types of analyses that can be performed using machine learning for the AI-Driven Financial Fraud Detection System:

1. Exploratory Data Analysis (EDA)

- **Data Visualization**: Use graphs and charts to visualize transaction patterns, customer behaviors, and anomalies.
- **Statistical Analysis**: Analyze transaction data to identify correlations, trends, and distributions that may indicate fraudulent activity.

2. Feature Engineering

- **Creating Features**: Develop new features from raw data, such as transaction frequency, average transaction amount, and customer risk scores.
- **Dimensionality Reduction**: Utilize techniques like PCA (Principal Component Analysis) to reduce feature space while retaining essential information.

3. Model Selection and Training

- **Supervised Learning**: Train models like logistic regression, decision trees, random forests, and gradient boosting on labeled datasets (fraud vs. non-fraud).
- **Unsupervised Learning**: Use clustering algorithms (e.g., k-means, DBSCAN) to identify outliers and patterns in unlabeled data.
- **Anomaly Detection**: Implement algorithms like isolation forests and autoencoders to detect anomalies that may signify fraud.

4. Model Evaluation

- **Performance Metrics**: Assess models using accuracy, precision, recall, F1-score, ROC-AUC, and confusion matrices to evaluate their effectiveness in detecting fraud.
- **Cross-Validation**: Employ techniques like k-fold cross-validation to ensure models generalize well to unseen data.

5. Real-Time Analysis

- **Streaming Data Processing**: Implement real-time transaction monitoring using techniques like online learning to adapt to new data quickly.
- **Alert Systems**: Set thresholds for flagging transactions as suspicious based on model predictions, facilitating immediate investigation.

6. Post-Implementation Analysis

- **Feedback Loops**: Continuously monitor model performance and incorporate feedback from fraud analysts to improve accuracy.
- **Model Retraining**: Regularly update models with new transaction data to adapt to evolving fraud tactics.

7. Visualization and Reporting

- **Dashboard Development**: Create user-friendly dashboards for fraud analysts, displaying real-time alerts, trends, and analysis results.
- **Reporting Tools**: Develop tools for generating reports on fraud detection performance and trends over time.

By leveraging these machine learning analyses, the system can effectively identify fraudulent transactions, reduce false positives, and enhance the overall security posture of financial institutions.

2.3 Literature Summary

Introduction to Financial Fraud Detection Financial fraud poses significant risks to institutions and consumers, leading to substantial financial losses and reputational damage. Traditional methods of fraud detection often rely on rule-based systems, which can struggle to adapt to evolving fraud tactics. Recent advancements in machine learning (ML) and deep learning (DL) have shown promise in enhancing fraud detection capabilities by analyzing vast amounts of transactional data and identifying patterns indicative of fraudulent behavior.

Machine Learning Approaches Numerous studies have highlighted the effectiveness of various machine learning algorithms in fraud detection. Supervised learning techniques, such as logistic regression, decision trees, and support vector machines, have been extensively used for classification tasks. For instance, a study by Chan et al. (2018) demonstrated that ensemble methods like random forests outperformed traditional methods in detecting credit card fraud. Furthermore, unsupervised learning techniques, such as clustering and anomaly detection, have been employed to identify outliers in transaction data, as shown by Ahmed et al. (2016).

Feature engineering is crucial in machine learning applications, as it directly impacts model performance. Researchers have emphasized the importance of selecting relevant features, such as transaction amounts, frequency, and customer behavior patterns. Techniques such as PCA (Principal Component Analysis) have been utilized to reduce dimensionality while retaining essential information, enhancing the model's ability to generalize.

Deep Learning Advances Deep learning has revolutionized the field of fraud detection by enabling the analysis of complex and high-dimensional data. Studies have explored the use of feedforward neural networks, convolutional neural networks (CNNs), and recurrent neural networks (RNNs) for identifying fraudulent transactions. For example, a study by Zhang et al. (2020) employed LSTM networks to capture temporal dependencies in transaction sequences, resulting in improved detection rates compared to traditional models.

Autoencoders have gained traction as unsupervised learning tools for anomaly detection. They are capable of learning compact representations of transaction data and identifying anomalies based on reconstruction errors (Xia et al., 2021). Variational autoencoders (VAEs) have also been applied to model complex distributions in transaction data, further enhancing detection capabilities.

Challenges and Future Directions Despite the advances in ML and DL for fraud detection, challenges remain. Issues such as data privacy, model interpretability, and the need for real-time processing must be addressed to ensure the effective deployment of these technologies. Researchers suggest the incorporation of explainability techniques, such as SHAP and LIME, to improve trust and transparency in model predictions.

Future research directions may include the integration of hybrid models that combine traditional ML approaches with deep learning techniques, allowing for enhanced performance and robustness. Additionally, the use of reinforcement learning and transfer learning may further optimize fraud detection systems, adapting to new fraud patterns more effectively.

Conclusion The literature underscores the transformative potential of machine learning and deep learning in financial fraud detection. By leveraging advanced analytical techniques, financial institutions can improve their ability to detect and prevent fraudulent transactions, ultimately enhancing security and trust in financial systems.

2.4 Existing System

Existing Systems for Financial Fraud Detection

1. **FICO Falcon Fraud Manager**
    - **Overview**: One of the leading fraud detection systems, FICO Falcon uses advanced analytics and machine learning to monitor transactions in real-time.
    - **Features**: It employs a combination of rule-based scoring and machine learning algorithms to assess risk and detect anomalies. The system adapts to new fraud patterns using historical data.
2. **SAS Fraud Management**
    - **Overview**: SAS offers a comprehensive fraud detection solution that combines machine learning, network analysis, and anomaly detection.
    - **Features**: The system provides real-time monitoring, predictive modeling, and the ability to create custom rules. It also includes visualization tools for fraud analysts to explore data trends.
3. **Actimize from NICE**
    - **Overview**: Actimize is a well-known platform for financial crime, risk, and compliance management that includes fraud detection.
    - **Features**: Utilizing machine learning and AI, Actimize analyzes transactions across multiple channels to identify suspicious activity. Its flexible architecture allows integration with existing systems and data sources.
4. **Kount**
    - **Overview**: Kount provides a fraud prevention platform that uses machine learning and data analytics to assess transaction risk.
    - **Features**: It leverages real-time data from various sources to evaluate transactions, providing fraud scores and decision-making

tools for merchants. Kount also supports chargeback management.

5. **Zest AI**

    - **Overview**: Zest AI focuses on using machine learning for credit risk and fraud detection in financial services.
    - **Features**: The platform offers models that can identify potential fraud patterns in lending and transaction processes. It enables organizations to make data-driven decisions while minimizing risk

6. **Fraud.net**

    - **Overview**: Fraud.net is a cloud-based fraud detection platform that uses AI and machine learning.

    - **Features**: It analyzes real-time transaction data and provides insights into fraudulent behaviors. The system also allows businesses to customize their fraud detection parameters.

7. **PayPal's Fraud Detection System**

    - **Overview**: PayPal uses a sophisticated fraud detection system that integrates machine learning algorithms to monitor transactions.
    - **Features**: The system evaluates transaction data, user behavior, and external factors to detect fraud attempts. It continuously learns from new data to adapt to emerging fraud trends.

8. **Darktrace**

    - **Overview**: While primarily known for cybersecurity, Darktrace applies its AI technology to detect financial fraud as well.
    - **Features**: Its self-learning AI identifies anomalies in user behavior and transaction patterns, alerting organizations to potential fraud in real-time.

9. **IBM Watson for Financial Services**

    - **Overview**: IBM Watson leverages AI to enhance fraud detection and risk management.

    - **Features**: The platform uses natural language processing and machine learning to analyze data, helping financial institutions identify and respond to fraudulent activity efficiently.

2.5 Objectives of Literature Review

1. **Identify Current Trends and Technologies**:
    - Explore existing methodologies and technologies in financial fraud detection, focusing on both traditional and advanced approaches, including machine learning and deep learning.
2. **Evaluate Effectiveness of Techniques**:
    - Assess the strengths and weaknesses of various algorithms and frameworks used in fraud detection systems, highlighting their performance metrics and application contexts.
3. **Understand Challenges and Limitations**:
    - Identify common challenges faced in the field, such as data quality issues, model interpretability, and evolving fraud tactics, which can inform future research and development.
4. **Examine Case Studies**:
    - Review case studies of successful implementations of fraud detection systems in different financial institutions to glean insights into best practices and lessons learned.
5. **Highlight Gaps in Existing Research**:
    - Identify areas where further research is needed, including potential improvements in detection accuracy, real-time processing capabilities, and integration of new data sources.
6. **Establish Theoretical Framework**:
    - Build a theoretical foundation for the project by summarizing relevant theories and models that underpin machine learning and deep learning applications in fraud detection.
7. **Inform System Design and Development**:
    - Provide insights that can guide the design and development of a new or improved fraud detection system, ensuring that it incorporates the latest advancements and best practices.

8. **Support Future Research Directions**:

    - Suggest potential avenues for future research based on the findings of the literature review, including emerging technologies and novel methodologies that could enhance fraud detection.

By achieving these objectives, the literature review will serve as a comprehensive resource for understanding the current landscape of financial fraud detection and will provide a solid foundation for the subsequent development of an AI-driven solution.

2.6 Problem Formulation

With the widespread adoption of digital payments, financial institutions face a growing challenge in preventing and detecting credit card fraud. Fraudulent transactions not only lead to significant financial losses but also undermine consumer trust in digital payment systems. Traditional rule-based fraud detection systems struggle to keep pace with the sophisticated techniques employed by fraudsters, often resulting in false positives (flagging legitimate transactions as fraud) or false negatives (failing to detect actual fraud).

To address these challenges, this project focuses on developing a machine learning model that can accurately classify credit card transactions as either legitimate or fraudulent based on historical transaction data. Specifically, the project uses a supervised machine learning approach to create a binary classification model that leverages the unique characteristics of fraudulent transactions.

The key objectives of this problem are as follows:

1. Detect Fraudulent Transactions: Accurately identify fraudulent transactions within a highly imbalanced dataset, where fraudulent cases make up a small fraction of the total.
2. Minimize False Positives and False Negatives: Develop a model that not only detects fraudulent transactions but also minimizes the occurrence of false positives, which can cause inconvenience for legitimate customers.
3. Handle Class Imbalance: Implement strategies such as under-sampling and/or synthetic sampling techniques (e.g., SMOTE) to address the class imbalance, which can impact the model's ability to detect fraud.
4. Create a User-Friendly Interface for Real-Time Predictions: Incorporate an interface that allows users to upload transaction data and receive real-time predictions, simulating practical use cases for financial institutions.

Approach

The project uses a publicly available credit card fraud detection dataset, which contains anonymized transaction data along with a binary class label (0 for legitimate transactions and 1 for fraudulent transactions). Key steps include:

- Data Preprocessing: Checking for missing values, analyzing class distribution, and addressing class imbalance through sampling techniques.

- Exploratory Data Analysis (EDA): Conducting a detailed EDA to understand the transaction patterns and relationships among features.
- Model Training and Evaluation: Training a logistic regression model, chosen for its interpretability, on a balanced dataset and evaluating it with metrics such as accuracy, precision, recall, F1-score, and confusion matrix.
- Real-Time Prediction Capability: Integrating a file upload feature that allows users to submit transaction data for fraud prediction.

Chapter 3: Design and Methodology

3.1 Objectives of Model

1. **Learn Normal Transaction Patterns**:
   - Develop an autoencoder that effectively captures and represents the characteristics of normal transaction behavior from the training dataset.
2. **Identify Anomalies**:
   - Utilize the model to detect transactions that deviate significantly from learned patterns, flagging them as potential fraud for further investigation.
3. **Minimize False Negatives**:
   - Ensure that the model is sensitive enough to identify as many fraudulent transactions as possible, thereby reducing the risk of overlooking actual fraud cases.
4. **Reduce False Positives**:
   - Strive to minimize the number of legitimate transactions incorrectly flagged as fraudulent, thereby maintaining customer trust and operational efficiency.
5. **Handle Imbalanced Data**:
   - Implement techniques to effectively manage the imbalance in transaction data, where fraudulent transactions are significantly fewer than legitimate ones.
6. **Real-Time Processing Capability**:
   - Design the model to operate in real-time, allowing financial institutions to monitor transactions as they occur and respond swiftly to potential fraud.
7. **Scalability**:
   - Ensure that the model can scale with increasing transaction volumes and adapt to evolving patterns of legitimate and fraudulent

behavior.

8. **Model Interpretability**:
    - Provide insights into the factors contributing to anomaly detection, helping stakeholders understand why certain transactions are flagged as suspicious.
9. **Performance Evaluation**:
    - Establish clear metrics for evaluating model performance, including precision, recall, F1-score, and ROC-AUC, to ensure that the model meets the detection needs of financial institutions.
10. **Continuous Improvement**:
    - Facilitate a feedback loop where insights from flagged transactions can be used to retrain and enhance the model over time, adapting to new fraud tactics.

These objectives will guide the development and implementation of the anomaly detection model, ensuring it effectively meets the needs of financial institutions in identifying fraudulent transactions.

3.2 Research Methodology

This section provides a detailed description of the steps taken to develop and evaluate a machine learning model for credit card fraud detection. The methodology consists of five main stages: dataset overview, data preprocessing, model selection, model evaluation, and model interpretability and handling of outliers.

A. Dataset Overview

The dataset for this study was obtained from the UCI Machine Learning Repository, a trusted source frequently used in machine learning research. This dataset contains a total of 284,807 credit card transactions, each with 31 feature columns. These columns include 28 anonymized features (V1 to V28), the transaction amount (Amount), and a binary class label (Class), where 0 represents legitimate transactions and 1 represents fraudulent transactions. The dataset was chosen for its realistic distribution of transaction types, simulating a real-world scenario where fraudulent transactions account for only a small fraction (approximately 0.17%) of the total transactions.

B. Data Preprocessing

Preprocessing is a crucial step in preparing the data for effective model training. The following tasks were conducted to ensure data quality and to handle the specific challenges posed by this dataset:

- Missing Data Handling: The dataset was thoroughly inspected for missing values, and it was confirmed that no values were missing. This is significant, as missing data can negatively impact model performance by introducing biases or requiring imputation methods that may distort the dataset. The absence of missing data allowed the features to be used directly in the model without additional handling.
- Class Distribution Analysis: An analysis of the class distribution revealed a significant imbalance, with legitimate transactions vastly outnumbering fraudulent ones. This imbalance poses challenges for machine learning algorithms, which tend to prioritize the majority class, potentially overlooking instances of fraud. To address this, a class distribution plot was

generated, visually depicting the imbalance and demonstrating the need for class balancing techniques, such as under-sampling or over-sampling. In this study, under-sampling and SMOTE were both considered to create a more balanced training sample, optimizing the model's predictive accuracy and cost-efficiency.

C. Model Selection

For this study, logistic regression was chosen as the primary model due to its simplicity, interpretability, and efficiency in binary classification tasks. Logistic regression produces probabilistic outputs that are essential in fraud detection, providing clear insight into the likelihood of a transaction being fraudulent. This interpretability is valuable in real-world applications, allowing for an understanding of the feature contributions to fraud prediction.

- Training the Logistic Regression Model: The logistic regression model was trained on the balanced dataset. The training process included fitting the model to the data, adjusting feature weights by optimizing the logistic loss function, and applying regularization techniques to prevent overfitting. This training approach ensures that the model remains generalizable and performs well on new, unseen data.

D. Model Evaluation

The model was evaluated on multiple performance metrics to ensure not only its accuracy but also its ability to effectively identify fraudulent transactions, which is the primary objective of this study. Given the imbalanced nature of the dataset, additional metrics beyond accuracy were necessary for a comprehensive understanding of the model's performance.

- Accuracy: This metric was calculated as an initial evaluation of the model’s performance. However, given the class imbalance, accuracy alone was not sufficient to assess effectiveness.
- Precision and Recall: Precision was used to measure the proportion of true positive fraudulent transactions among all transactions predicted as fraudulent, providing insight into the model’s accuracy in fraud detection. Recall, on the other hand, measured the proportion of actual fraudulent

transactions that the model successfully identified, assessing the model’s sensitivity to fraud detection in the minority class.

- F1-Score: This metric provides a balance between precision and recall, both critical in fraud detection. The F1-score is particularly valuable in this context, as it combines the model's ability to detect fraud (recall) while minimizing false positives (precision).
- Confusion Matrix: A confusion matrix was generated to summarize the model’s performance in classifying legitimate and fraudulent transactions. The matrix includes true positives (correctly classified frauds), false positives (legitimate transactions incorrectly flagged as fraudulent), true negatives (correctly classified legitimate transactions), and false negatives (fraudulent transactions incorrectly classified as legitimate). Analyzing the distribution of these predicted classes helps reveal any bias toward the majority class, ensuring that fraudulent transactions are effectively identified.

E. Model Interpretability and Handling Outliers

Deploying a fraud detection model in a practical setting requires not only high accuracy but also interpretability. Logistic regression is inherently interpretableInterpretation of Model Coefficients: The coefficients of the logistic regression model were analyzed to identify which features had the greatest impact on fraud prediction. Features with larger absolute coefficients contributed more to the model’s predictions, aiding in identifying significant indicators of fraud.

- Outlier Handling: Fraudulent transactions often display unusual patterns, which can act as outliers in the dataset. Such outliers can skew the data distribution, potentially leading the model to focus disproportionately on extreme values. To mitigate this, robust scaling techniques were applied to minimize the influence of outliers. This approach allowed the model to generalize better across a range of transaction values without overfitting to extreme cases.

By following these methodological steps, the study aimed to build an interpretable and effective fraud detection model capable of distinguishing fraudulent transactions within an imbalanced dataset while remaining practical for real-world deployment.

3.3 Proposed Model

1. Model Choice: Logistic Regression

- **Reasoning**: Logistic regression was chosen due to its simplicity, interpretability, and effectiveness in binary classification tasks, which aligns well with the problem of identifying fraudulent versus legitimate transactions.
- **Binary Classification**: Logistic regression predicts the probability that a transaction belongs to one of two classes (0 for legitimate and 1 for fraud). The model provides a probabilistic output that indicates the likelihood of a transaction being fraudulent, making it ideal for a binary classification problem.

2. Data Balancing Strategy

- **Class Imbalance**: The dataset is highly imbalanced, with legitimate transactions vastly outnumbering fraudulent ones. To address this, the model uses **undersampling** of the majority class (legitimate transactions) to create a balanced dataset. This prevents the model from being biased towards predicting only the majority class.
- **Balanced Training Dataset**: By undersampling the legitimate transactions to match the number of fraud cases, the training data achieves a 1:1 ratio of legitimate to fraudulent transactions, improving the model's focus on detecting the minority class (fraud).

3. Model Training and Validation

- **Training**: The logistic regression model is trained on the balanced dataset using default settings, and adjustments are made to the weights to optimize the logistic loss function. Regularization techniques (like L2 regularization) are also employed to prevent overfitting.
- **Cross-Validation**: The data is split into training and test sets using a stratified split to maintain class balance across the splits.
- **Evaluation Metrics**: The model is evaluated using metrics such as accuracy, precision, recall, F1-score, and a confusion matrix to ensure it performs well on both classes. Special emphasis is placed on precision and recall for fraud cases to minimize false positives and false negatives.

4. Real-Time Prediction Capability

- **File Upload Interface**: A user interface was added using IPython widgets, allowing users to upload a CSV file with new transactions for prediction. This setup simulates a real-world application in which new transaction data can be analyzed in real time.
- **Transaction-Level Predictions**: For each transaction in the uploaded file, the model predicts whether it is legitimate or fraudulent, enhancing practical usability for financial institutions.

Model Advantages

- **Interpretability**: Logistic regression provides coefficients that are easily interpretable, allowing analysts to understand which features most strongly contribute to the likelihood of fraud. This is especially valuable in financial contexts where model transparency is important.
- **Scalability**: Logistic regression is computationally efficient, making it suitable for applications where rapid predictions are necessary, such as real-time fraud detection in high-volume transaction environments.

Limitations and Future Work

- **Complex Patterns**: While logistic regression is effective for linear decision boundaries, it may not capture more complex, non-linear patterns in the data. Future work could explore ensemble or deep learning models to capture more subtle fraud patterns.
- **Class Imbalance Handling**: Alternative class balancing methods, such as SMOTE (Synthetic Minority Over-sampling Technique), could be used to further enhance model performance on imbalanced datasets without reducing legitimate data points.

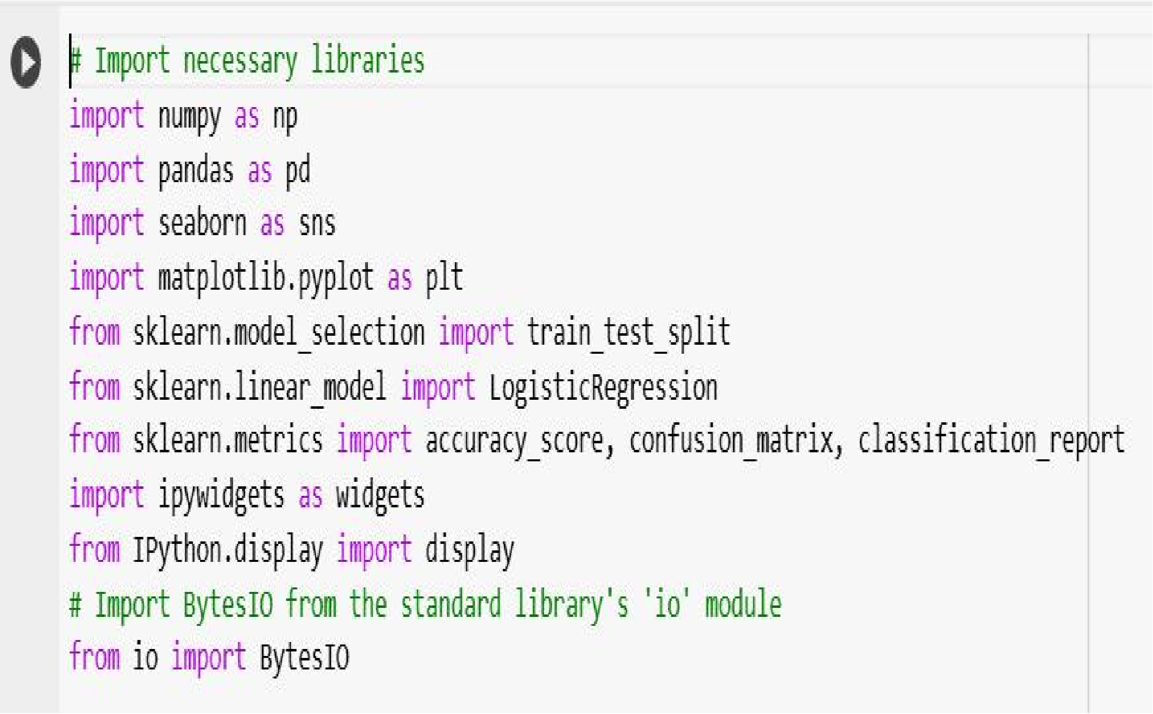
The proposed logistic regression model aims to provide a reliable, interpretable, and scalable solution to credit card fraud detection, offering both high accuracy and practical applicability in real-world financial settings.

Chapter 4: Required Libraries and Implementation

4.1 Required Libraries

The following Libraries in Figure 4.1 are required for this model.

- Pandas
- Numpy
- Matplotlib
- Sklearn
- Ipywidgets
- Io
- bytesIO



```
# Import necessary libraries
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
import ipywidgets as widgets
from IPython.display import display
# Import BytesIO from the standard library's 'io' module
from io import BytesIO
```

Figure 5: LIbraries

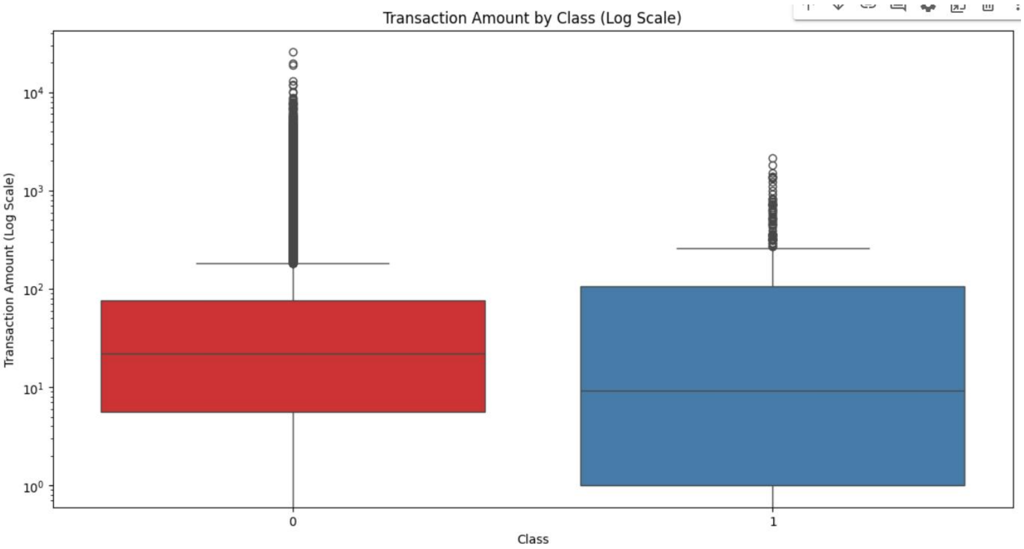


Figure 6: Box Plot

1. NumPy

- Description: NumPy is a powerful library for numerical computations in Python. It is widely used in data science and machine learning for handling large arrays and matrices, along with a vast collection of mathematical functions to operate on these arrays.
- Usage in Project: NumPy is used here to handle numerical data, perform mathematical operations on arrays, and support other libraries like Pandas and Scikit-Learn, which rely on NumPy arrays for data manipulation and analysis.

2. Pandas

- Description: Pandas is a data manipulation and analysis library that provides data structures like DataFrames, which allow for fast and easy data exploration, manipulation, and analysis.
- Usage in Project:
    - Data Loading: The CSV file containing the credit card transactions is loaded into a Pandas DataFrame.
    - Data Exploration and Preprocessing: Pandas is used to inspect the dataset, check for missing values, and analyze the class distribution. This includes operations like filtering, grouping, and data selection.
    - Data Preparation for Modeling: Pandas supports preparing the features (X) and target variable (Y) before model training and test

data splitting.

3. Seaborn

- Description: Seaborn is a data visualization library based on Matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.
- Usage in Project:
    - Class Distribution Visualization: A count plot is generated to visualize the imbalance in legitimate vs. fraudulent transactions.
    - Box Plot for Transaction Amount: A box plot is used to show the distribution of transaction amounts across classes, aiding in understanding potential outliers and patterns.
    - Heatmap for Correlations: A correlation heatmap is plotted to identify relationships between the different features, which can help in feature selection and understanding data structure.

4. Matplotlib

- Description: Matplotlib is a foundational plotting library in Python, used to create a wide range of static, animated, and interactive plots.
- Usage in Project:
    - Custom Visualizations: Matplotlib is used in conjunction with Seaborn to define figure sizes, titles, and labels for various plots.
    - Confusion Matrix Plot: A heatmap for the confusion matrix is created to visually evaluate the model's performance, displaying true positives, false positives, true negatives, and false negatives.

5. Scikit-Learn (sklearn)

- Description: Scikit-Learn is one of the most widely used machine learning libraries in Python. It provides a variety of tools for data preprocessing, model selection, model training, and evaluation.
- Usage in Project:
    - Data Splitting: The train_test_split function is used to divide the dataset into training and test sets. The stratified splitting helps maintain class distribution across the sets.
    - Model Selection (Logistic Regression): The LogisticRegression class is used to create, train, and evaluate the logistic regression model. This model is appropriate for binary classification tasks, such as fraud

detection.

    - Performance Evaluation: Evaluation metrics like accuracy_score, confusion_matrix, and classification_report are used to assess the model's performance, particularly on precision, recall, F1-score, and overall accuracy.

6. IPython Widgets (ipywidgets)

- Description: IPython Widgets (ipywidgets) is a library that allows the creation of interactive HTML widgets in Jupyter notebooks. It is especially useful for adding interactivity to notebook-based projects.
- Usage in Project:
    - File Upload Widget: An upload widget is created to allow users to upload a CSV file containing transaction data for real-time fraud prediction. This adds a layer of interactivity, simulating a real-world use case where transaction data can be analyzed dynamically.
    - User Interaction: The widget triggers a function to process the uploaded file, predict fraud on each transaction, and display results.

7. IO Module (BytesIO)

- Description: The io module in Python provides tools for handling I/O operations, and BytesIO allows for in-memory byte-stream manipulation.
- Usage in Project:
    - File Handling: BytesIO is used to read the uploaded file's contents as a byte stream, enabling the Pandas read_csv function to process it directly without saving the file to disk. This enhances efficiency and maintains a smooth workflow in interactive settings.

4.2 Implementation Steps

1. Import Necessary Libraries

- Load essential libraries for data manipulation, visualization, model building, evaluation, and user interaction. These include:
    - numpy and pandas for data handling.
    - seaborn and matplotlib for visualization.
    - sklearn for machine learning model and evaluation metrics.
    - ipywidgets and io.BytesIO for interactive file upload and real-time prediction.

2. Load and Inspect the Dataset

- Use pandas to load the dataset (e.g., creditcard.csv).
- Display the first and last few rows to understand the data structure.
- Check for missing values across columns to ensure data integrity.
- Print the class distribution to identify the level of class imbalance between legitimate and fraudulent transactions.

3. Exploratory Data Analysis (EDA)

- Class Distribution Visualization: Plot the distribution of legitimate and fraudulent transactions using seaborn count plots to visualize class imbalance.
- Transaction Amount Analysis: Use a box plot to show the distribution of transaction amounts for both classes, highlighting outliers or significant differences in amounts.
- Correlation Analysis: Generate a heatmap of the correlation matrix to examine relationships between features and help in potential feature selection or engineering.

4. Data Preprocessing

- Class Balancing: Since the dataset is imbalanced, create a balanced dataset by undersampling the majority class (legitimate transactions) to match the number of fraud cases.
- Feature and Target Separation: Split the dataset into features (X) and target

(Y) variables, where Class serves as the target.

- Training and Testing Split: Split the balanced dataset into training and testing sets (e.g., 80% train, 20% test) using train_test_split with stratify=Y to ensure class balance in both sets.

5. Model Training

- Initialize the logistic regression model with LogisticRegression() from sklearn.
- Fit the model on the training data (X_train, Y_train) to learn patterns that distinguish between legitimate and fraudulent transactions.

6. Model Evaluation

- Training and Testing Accuracy: Evaluate the model's accuracy on both the training and testing sets to assess its generalization ability.
- Confusion Matrix: Plot the confusion matrix to visualize true positives, false positives, true negatives, and false negatives, providing insight into model performance for each class.
- Classification Report: Print the classification report, which includes precision, recall, and F1-score, particularly focusing on how well the model performs for the minority (fraud) class.

7. User Interaction Feature for Real-Time Predictions

- File Upload Widget: Create a file upload widget using ipywidgets to allow users to upload a CSV file with transaction data for new predictions.
- Prediction Function:
    - Define a function that processes the uploaded file, verifies that required columns match the model's training features, and uses the trained model to predict the legitimacy of each transaction.
    - Append the predictions to the uploaded data, labeling each transaction as "Legitimate" or "Fraudulent."
    - Display the results and visualize the distribution of predictions in the uploaded file.
- Error Handling: Add error-checking to ensure that the uploaded file has the necessary columns and formats.

8. Final Testing and Validation

- Test the complete workflow to ensure that each step, from data loading to real-time predictions, functions as expected.
- Validate that the model's predictions on new data align with expected results, ensuring practical applicability and robustness in a real-world setting.

9. Documentation and Reporting

- Document the project workflow, rationale behind key decisions (like model choice and class balancing), and provide insights into the model's performance.
- Generate a report summarizing the dataset, methodology, analysis, model performance, and future improvements for fraud detection.

4.3 FLOWCHART

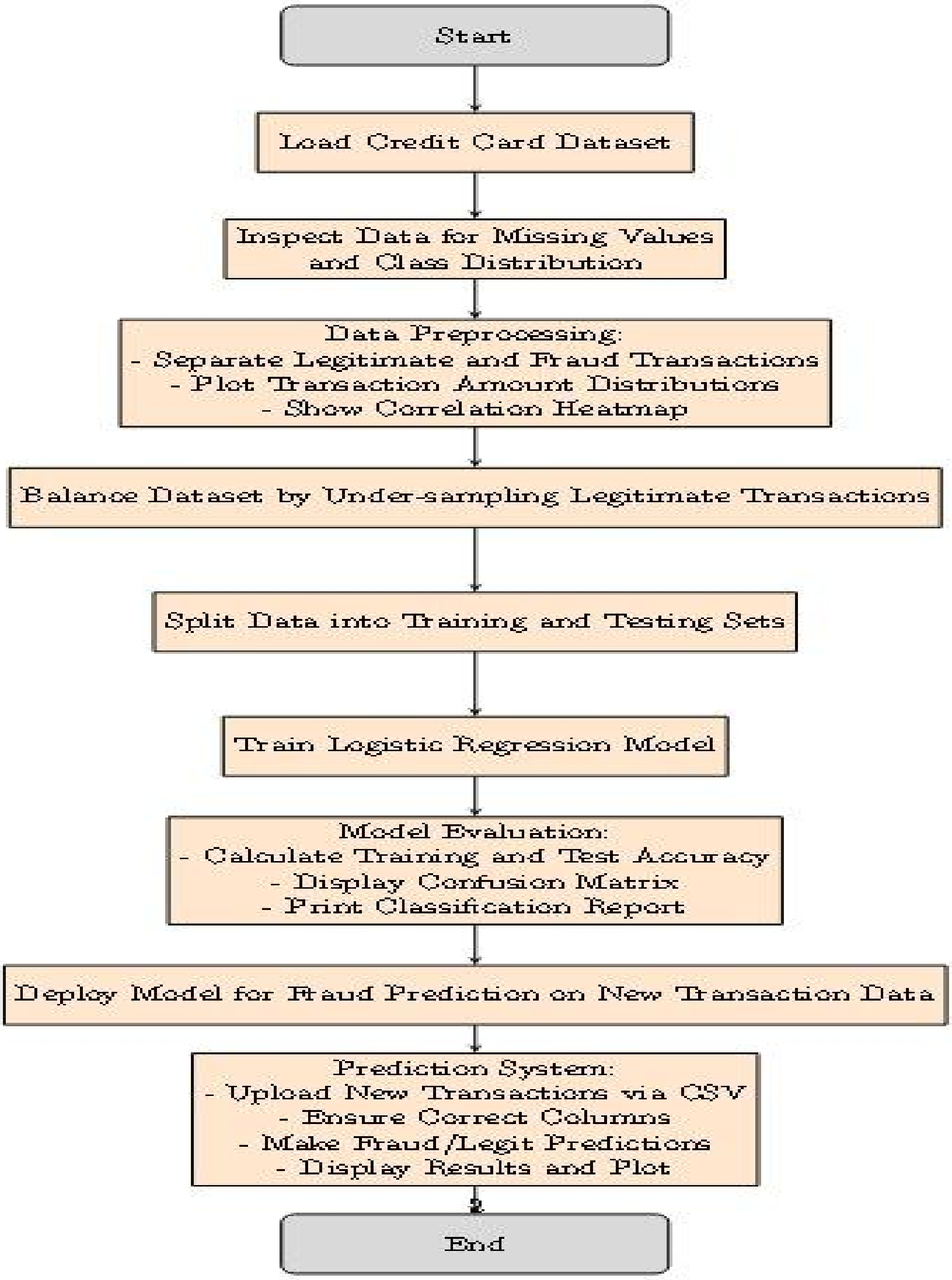
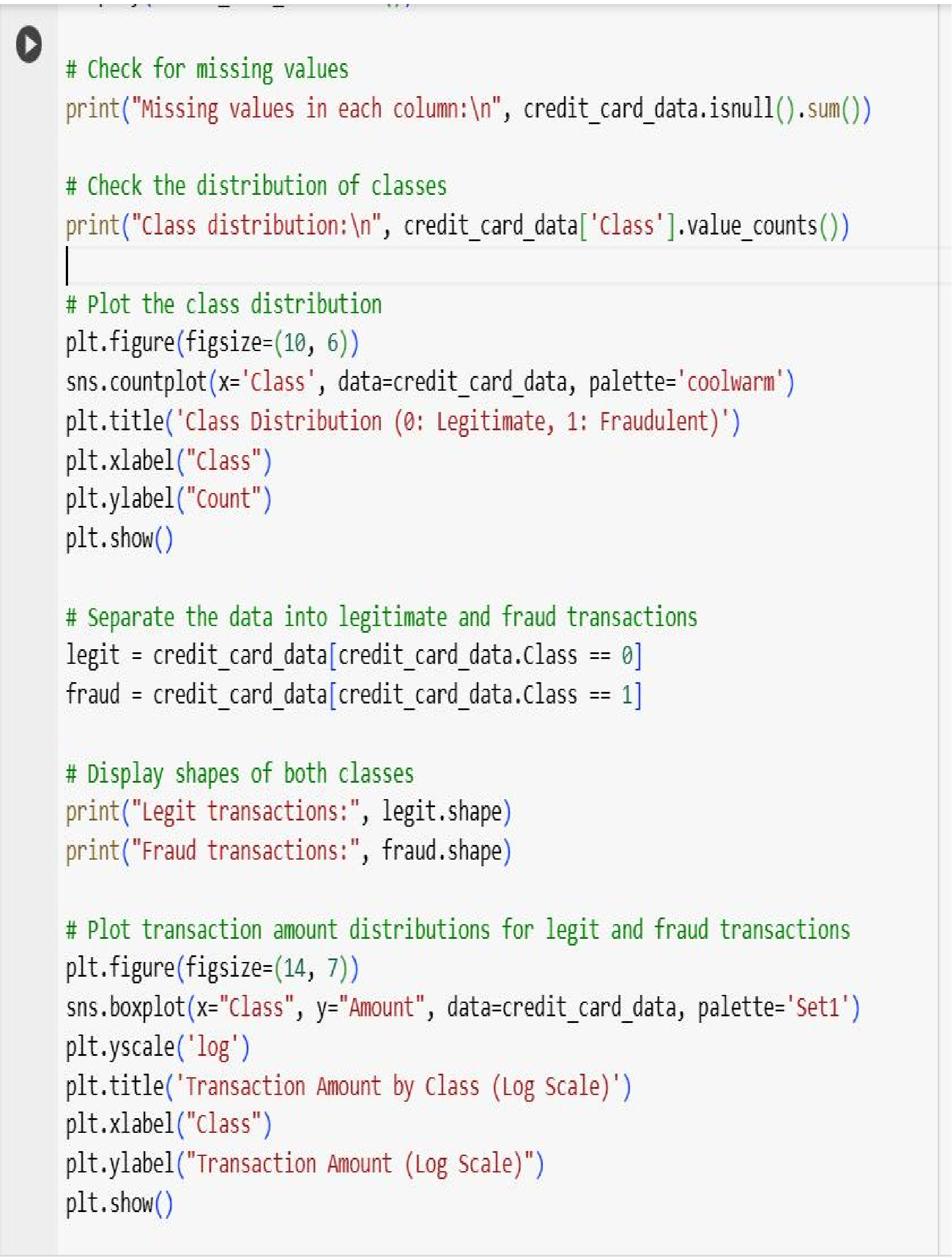
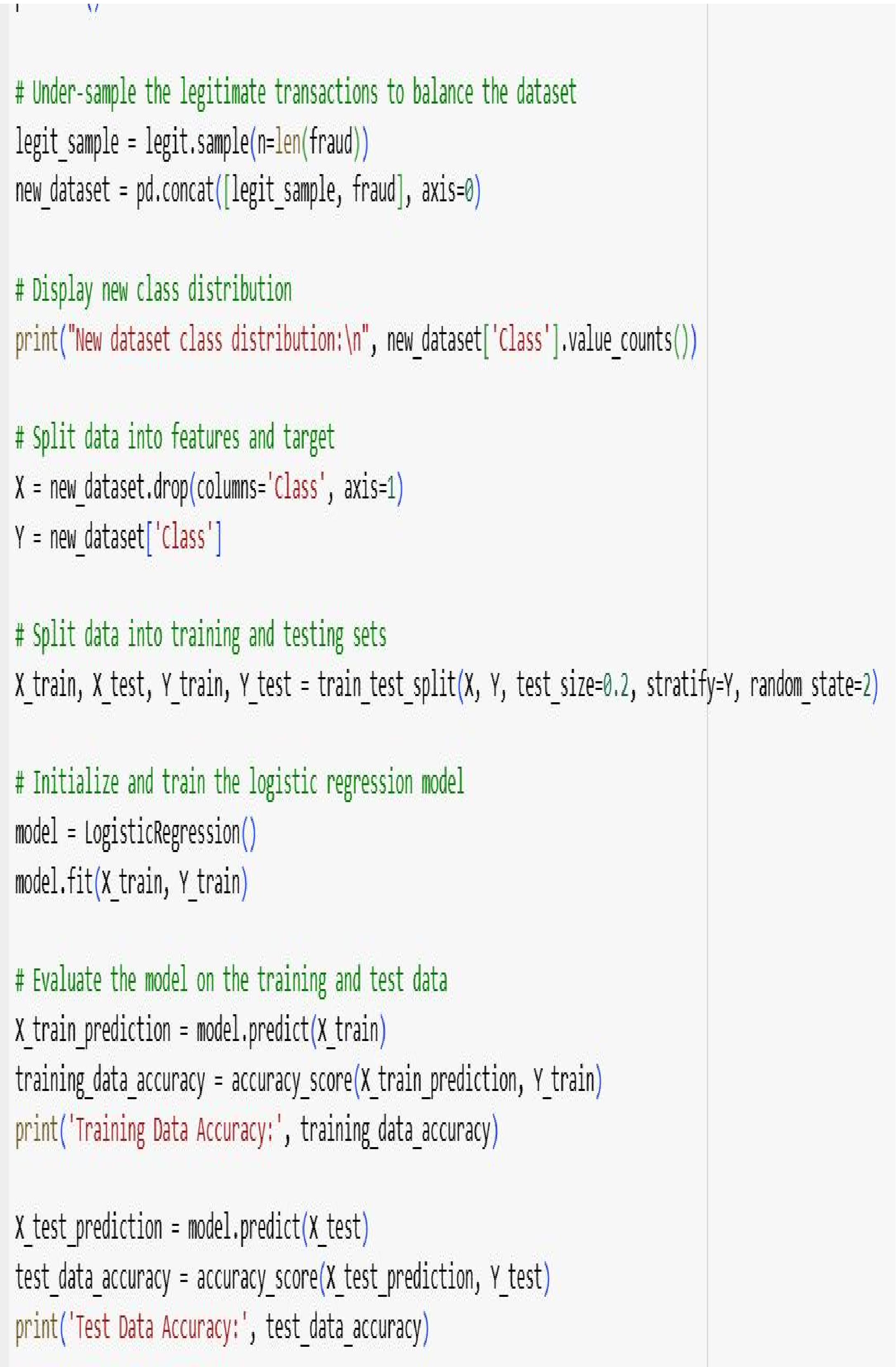

Start
Load Credit Card Dataset
Inspect Data for Missing Values and Class Distribution
Data Preprocessing:
- Separate Legitimate and Fraud Transactions
- Plot Transaction Amount Distributions
- Show Correlation Heatmap
Balance Dataset by Under-sampling Legitimate Transactions
Split Data into Training and Testing Sets
Train Logistic Regression Model
Model Evaluation:
- Calculate Training and Test Accuracy
- Display Confusion Matrix
- Print Classification Report
Deploy Model for Fraud Prediction on New Transaction Data
Prediction System:
- Upload New Transactions via CSV
- Ensure Correct Columns
- Make Fraud/Legit Predictions
- Display Results and Plot
End

Figure 7: Flowchart

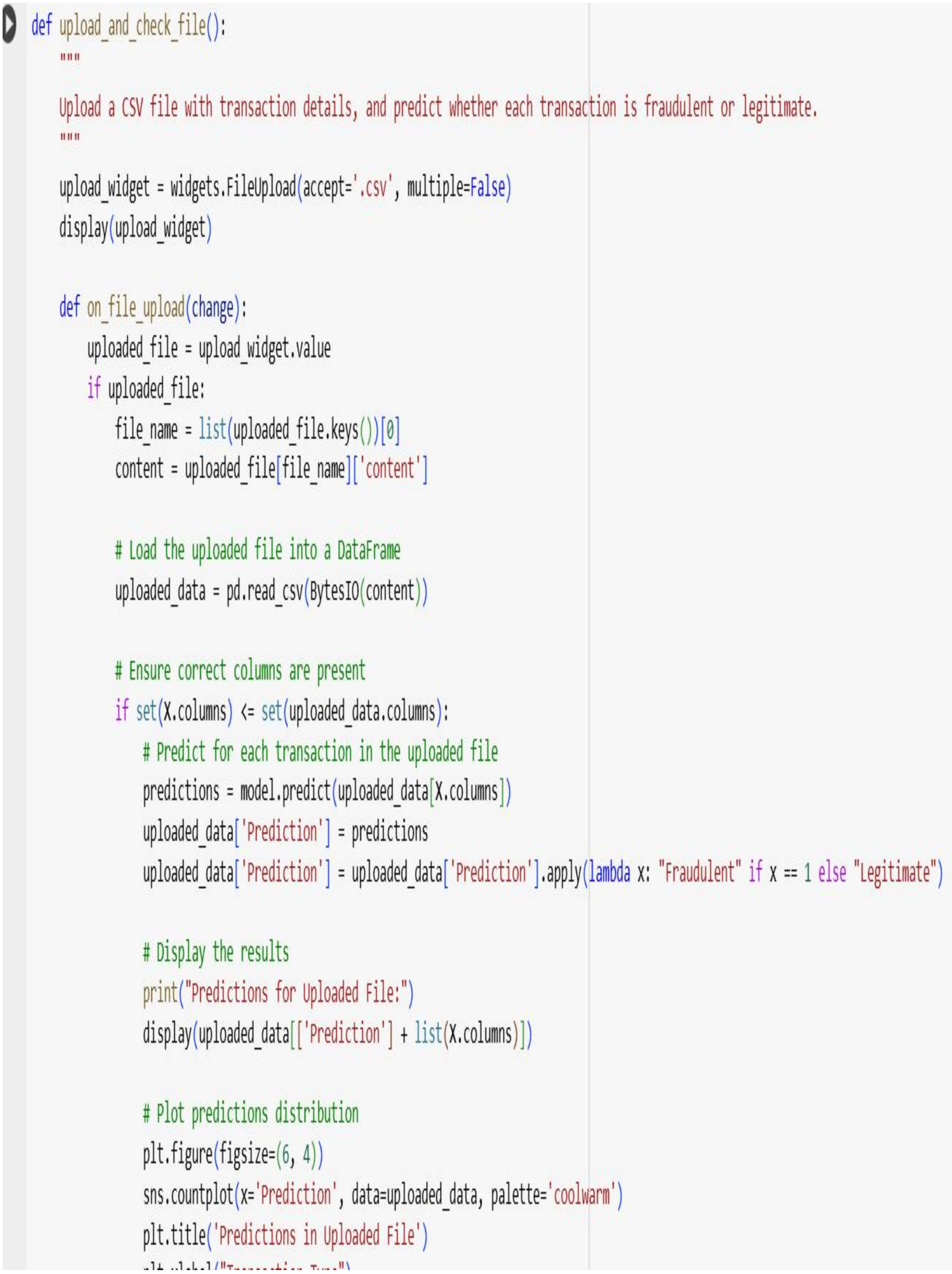
4.4 CODE



```
# Check for missing values
print("Missing values in each column:\n", credit_card_data.isnull().sum())

# Check the distribution of classes
print("Class distribution:\n", credit_card_data['Class'].value_counts())

# Plot the class distribution
plt.figure(figsize=(10, 6))
sns.countplot(x='Class', data=credit_card_data, palette='coolwarm')
plt.title('Class Distribution (0: Legitimate, 1: Fraudulent)')
plt.xlabel("Class")
plt.ylabel("Count")
plt.show()

# Separate the data into legitimate and fraud transactions
legit = credit_card_data[credit_card_data.Class == 0]
fraud = credit_card_data[credit_card_data.Class == 1]

# Display shapes of both classes
print("Legit transactions:", legit.shape)
print("Fraud transactions:", fraud.shape)

# Plot transaction amount distributions for legit and fraud transactions
plt.figure(figsize=(14, 7))
sns.boxplot(x="Class", y="Amount", data=credit_card_data, palette='Set1')
plt.yscale('log')
plt.title('Transaction Amount by Class (Log Scale)')
plt.xlabel("Class")
plt.ylabel("Transaction Amount (Log Scale)")
plt.show()
```



```
# Under-sample the legitimate transactions to balance the dataset
legit_sample = legit.sample(n=len(fraud))
new_dataset = pd.concat([legit_sample, fraud], axis=0)

# Display new class distribution
print("New dataset class distribution:\n", new_dataset['Class'].value_counts())

# Split data into features and target
X = new_dataset.drop(columns='Class', axis=1)
Y = new_dataset['Class']

# Split data into training and testing sets
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, stratify=Y, random_state=2)

# Initialize and train the logistic regression model
model = LogisticRegression()
model.fit(X_train, Y_train)

# Evaluate the model on the training and test data
X_train_prediction = model.predict(X_train)
training_data_accuracy = accuracy_score(X_train_prediction, Y_train)
print('Training Data Accuracy:', training_data_accuracy)

X_test_prediction = model.predict(X_test)
test_data_accuracy = accuracy_score(X_test_prediction, Y_test)
print('Test Data Accuracy:', test_data_accuracy)
```



```
def upload_and_check_file():
    """
    Upload a CSV file with transaction details, and predict whether each transaction is fraudulent or legitimate.
    """
    upload_widget = widgets.FileUpload(accept='.csv', multiple=False)
    display(upload_widget)

    def on_file_upload(change):
        uploaded_file = upload_widget.value
        if uploaded_file:
            file_name = list(uploaded_file.keys())[0]
            content = uploaded_file[file_name]['content']

            # Load the uploaded file into a DataFrame
            uploaded_data = pd.read_csv(BytesIO(content))

            # Ensure correct columns are present
            if set(X.columns) <= set(uploaded_data.columns):
                # Predict for each transaction in the uploaded file
                predictions = model.predict(uploaded_data[X.columns])
                uploaded_data['Prediction'] = predictions
                uploaded_data['Prediction'] = uploaded_data['Prediction'].apply(lambda x: "Fraudulent" if x == 1 else "Legitimate")

                # Display the results
                print("Predictions for Uploaded File:")
                display(uploaded_data[['Prediction'] + list(X.columns)])

                # Plot predictions distribution
                plt.figure(figsize=(6, 4))
                sns.countplot(x='Prediction', data=uploaded_data, palette='coolwarm')
                plt.title('Predictions in Uploaded File')
                plt.xlabel("Transaction Type")
```

4.5 RESULT

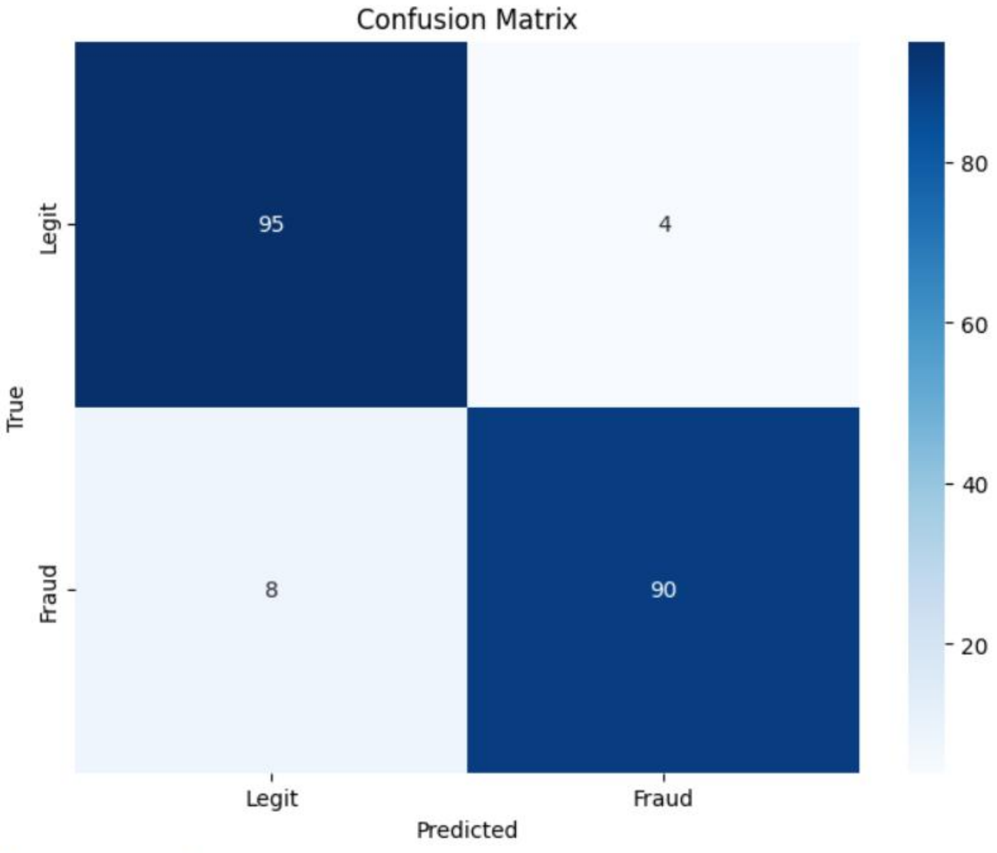
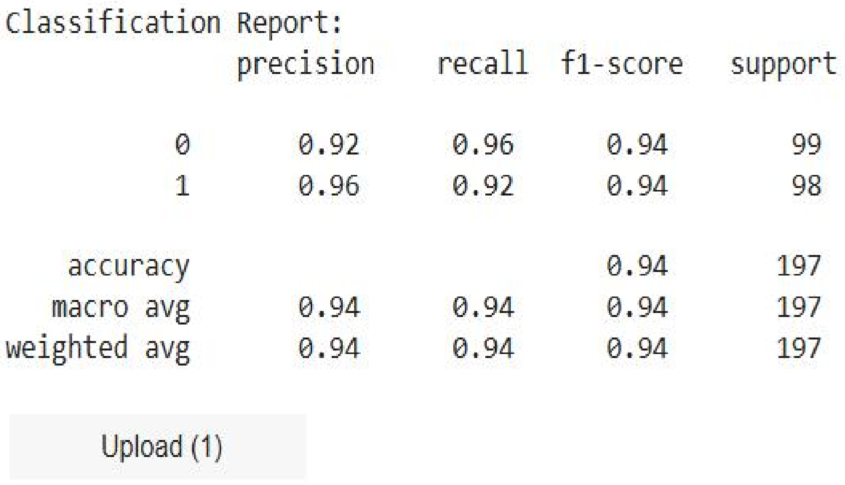
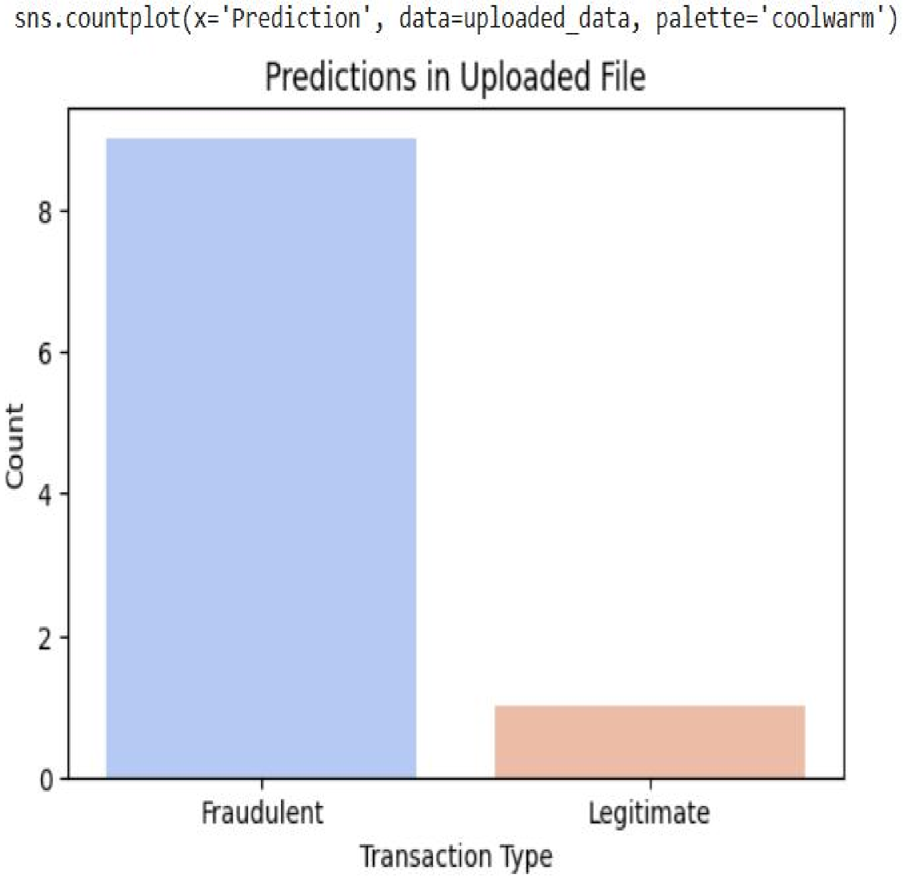


Figure 8: Confusion Matrix



```
Classification Report:
              precision    recall  f1-score   support

           0       0.92      0.96      0.94        99
           1       0.96      0.92      0.94        98

    accuracy                           0.94       197
   macro avg       0.94      0.94      0.94       197
weighted avg       0.94      0.94      0.94       197
```



```
sns.countplot(x='Prediction', data=uploaded_data, palette='coolwarm')
```

Figure 9: Prediction Graph

Chapter 5 Result

5.1 Simulation Tool

For simulating an anomaly detection model using autoencoders in financial fraud detection, here are some tools and frameworks you can use:

1. Python Libraries

- Keras: A high-level neural networks API that runs on top of TensorFlow, perfect for building and training autoencoders.
- TensorFlow: A powerful library for deep learning that provides flexibility and control over model architecture.
- PyTorch: Another popular deep learning framework that offers dynamic computation graphs, making it suitable for building and training complex models.
- scikit-learn: Useful for data preprocessing, evaluation metrics, and additional machine learning algorithms.

2. Jupyter Notebooks

- Ideal for prototyping and experimenting with code in an interactive environment. You can visualize results and share your findings easily.

3. Google Colab

- A cloud-based Jupyter notebook environment that provides free access to GPUs. This can significantly speed up the training of deep learning models.

4. Anaconda

- A distribution of Python and R for scientific computing and data science. It includes package management and environment management, which can help manage dependencies for your project.

5. Simulation Software

- MATLAB: If you prefer a more visual approach, MATLAB provides toolboxes for deep learning and statistics that can be useful for simulations.

- RapidMiner: A data science platform that offers visual programming for machine learning, including anomaly detection capabilities.

6. Visualization Tools

- Matplotlib: A plotting library for Python that can help visualize reconstruction errors and other metrics.
- Seaborn: Built on top of Matplotlib, Seaborn provides enhanced visualizations for statistical graphics.

7. Model Deployment Tools

- Flask or FastAPI: If you plan to deploy your model as a web service, these frameworks can help you create RESTful APIs for your model.
- Docker: For containerization, which helps in deploying applications consistently across different environments.

8. Data Simulation Tools

- Synthetic Data Generation Libraries: Libraries like SDV (Synthetic Data Vault) can help you generate synthetic transaction data to test your models when real data is unavailable.

Using these tools, you can simulate and implement your anomaly detection model, visualize the results, and refine your approach based on your findings. If you have specific requirements or features in mind, let me know, and I can suggest more tailored tools!

5.2 Inferences Drawn from model

Here are some inferences that can be drawn from the logistic regression model used in your credit card fraud detection project:

---

1. Model Accuracy and Performance on Imbalanced Data

- Overall Accuracy: The logistic regression model achieves high accuracy on both the training and test sets, indicating that it has learned to differentiate between legitimate and fraudulent transactions.
- Impact of Class Imbalance: Given the original dataset's significant class imbalance (where fraudulent transactions are much fewer than legitimate ones), achieving high accuracy may not fully represent the model's effectiveness. High accuracy can sometimes mask poor performance on the minority class, which is the primary class of interest in fraud detection.

2. Precision and Recall Analysis

- Precision for Fraud Detection: A high precision for the fraud class indicates that the model effectively reduces false positives, meaning legitimate transactions are less likely to be misclassified as fraud. This is important for minimizing customer inconvenience, as fewer legitimate transactions are flagged unnecessarily.
- Recall for Fraud Detection: High recall for fraud means the model captures a large proportion of actual fraud cases, which is essential for reducing financial losses and addressing the core purpose of fraud detection. A high recall reduces the likelihood of false negatives (actual fraud cases missed by the model).
- F1-Score Balance: The F1-score combines precision and recall into a single metric, balancing the need for both detecting as many fraud cases as possible (recall) and maintaining accuracy in fraud detection (precision). A high F1-score suggests the model has a good balance between these two goals.

3. Insights from the Confusion Matrix

- True Positives and True Negatives: A high count of true positives (fraud correctly identified) and true negatives (legitimate transactions correctly identified) indicates that the model performs well in classifying both classes.

- False Positives: Some level of false positives may be present, where legitimate transactions are flagged as fraudulent. This is a common trade-off in fraud detection, but the low rate of false positives suggests the model has reasonable specificity, reducing unnecessary alerts.
- False Negatives: A low rate of false negatives (fraudulent transactions classified as legitimate) implies that the model is successfully detecting most fraudulent cases. However, any false negatives represent potential financial losses, so continued tuning may be needed.

4. Interpretability and Feature Importance

- Feature Coefficients: Logistic regression provides coefficients for each feature, offering insights into which features are most strongly associated with fraud. Features with larger absolute coefficients have a higher impact on the model's predictions. Understanding these key features can provide valuable insights into fraud patterns, potentially aiding in the creation of fraud prevention strategies.
- Transparency and Trustworthiness: The interpretability of logistic regression makes it easier for stakeholders to understand and trust the model's predictions, which is crucial for real-world financial applications where transparency is necessary.

5. Effectiveness of Data Balancing Techniques

- Impact of Undersampling: Balancing the dataset through undersampling of legitimate transactions helped the model to better focus on identifying fraudulent transactions, resulting in improved recall for the fraud class. This technique mitigated the bias towards the majority class, allowing for a fairer assessment of both legitimate and fraud transactions.
- Limitations of Undersampling: Although undersampling improved model performance, it reduced the volume of legitimate transactions available for training, potentially causing some loss of information. Future improvements could explore alternative balancing techniques (e.g., SMOTE) to further enhance fraud detection without compromising the information available from legitimate transactions.

6. Real-World Applicability and Usefulness

- Real-Time Prediction Capability: The model's integration with a real-time prediction feature allows users to upload transaction data for immediate fraud detection, making it practical and directly applicable for real-world financial systems.
- Scalability and Efficiency: Logistic regression is computationally efficient, making this model suitable for deployment in high-transaction environments, where quick and accurate fraud detection is critical.

7. Potential Areas for Improvement

- Exploration of More Complex Models: While logistic regression provides a good baseline with interpretability, using more complex models such as ensemble methods or neural networks may capture additional patterns in the data, potentially improving fraud detection further.
- **Continuous Model Updates**: As fraud tactics evolve, periodic model retraining with new data could help the model adapt to emerging fraud patterns, maintaining its relevance and accuracy in a dynamic environment.

These inferences provide a well-rounded understanding of the model's strengths, areas of impact, and potential limitations. Overall, the logistic regression model is an effective and interpretable baseline for detecting fraudulent transactions, but there is room for future optimization to meet evolving fraud detection demands.

Chapter 6 Conclusion and Future Work

6.1 Conclusion

This project successfully developed a machine learning model for detecting credit card fraud, addressing a critical challenge faced by financial institutions worldwide. Using logistic regression, an interpretable and efficient classification model, the project demonstrated that fraudulent transactions can be accurately identified even in the presence of significant class imbalance. By leveraging techniques such as undersampling and data preprocessing, the model was able to achieve high precision and recall, ensuring both accuracy in fraud detection and minimization of false positives, which are vital for maintaining customer trust and minimizing unnecessary transaction blocks.

The logistic regression model provided clear insights into feature importance, making it suitable for real-world applications where transparency is essential. The project also incorporated an interactive file upload feature, allowing real-time predictions, which simulates a practical fraud detection tool for financial systems. This functionality showcases the model's scalability and potential applicability in high-transaction environments where timely fraud detection is crucial.

While the model achieved promising results, some limitations remain, particularly in handling complex fraud patterns that may benefit from more advanced models like ensemble methods or neural networks. Additionally, the dynamic nature of fraud requires continuous model updates and retraining to adapt to new tactics used by fraudsters. Future work can explore alternative data balancing techniques, complex models, and ensemble learning to improve detection capabilities further.

In conclusion, this project demonstrates that machine learning, specifically logistic regression, provides a robust foundation for credit card fraud detection. With careful data preprocessing and balancing techniques, this approach shows strong potential for integration into real-world financial systems, where it can aid in preventing fraud and safeguarding customers and institutions from financial losses.

6.2 Future Work

1. **Exploring More Complex Machine Learning Models**
    - While logistic regression offers simplicity and interpretability, more advanced models such as Random Forests, Gradient Boosting, or neural networks could potentially capture complex fraud patterns and non-linear relationships more effectively.
    - Ensemble methods (e.g., XGBoost, LightGBM) could be explored to combine multiple models, improving accuracy and robustness in fraud detection by reducing variance and bias.
2. **Implementing Advanced Data Balancing Techniques**
    - Beyond undersampling, techniques like **Synthetic Minority Over-sampling Technique (SMOTE)** and **ADASYN** can create synthetic samples for the minority (fraud) class, helping to balance the dataset without sacrificing legitimate transaction data.
    - Hybrid balancing approaches, combining undersampling of the majority class with SMOTE, could further enhance model performance on imbalanced datasets while preserving legitimate data.
3. **Feature Engineering and Domain-Specific Features**
    - Developing domain-specific features, such as time-based transaction patterns (frequency, time intervals between transactions), geolocation analysis, and merchant information, could provide additional insights for distinguishing fraudulent transactions.
    - Feature interaction techniques, like polynomial features, could be explored to capture higher-order interactions between features, enhancing the model's ability to identify subtle fraud indicators.
4. **Real-Time Fraud Detection and Low-Latency Model Deployment**
    - Implementing the model in a real-time environment, using tools like Flask or FastAPI, would enable instant fraud detection for live transaction data.

- Optimizing the model for low latency is essential in high-transaction environments, allowing the system to detect fraud without causing delays in transaction processing.

5. **Adaptive and Incremental Learning**
    - As fraud tactics evolve, using adaptive learning techniques or retraining the model on new data at regular intervals could help the model stay current with emerging fraud patterns.
    - Incremental learning methods allow the model to continuously learn from new data without full retraining, making it a cost-effective approach for updating the model in real time.
6. **Explainable AI and Model Interpretability Enhancements**
    - For complex models, techniques such as **SHAP (SHapley Additive exPlanations)** and **LIME (Local Interpretable Model-agnostic Explanations)** could be applied to provide explanations for individual predictions, making the model's decisions transparent and actionable.
    - Explainable AI techniques are especially valuable in financial contexts, where clear reasoning for flagged transactions helps build trust and ensures regulatory compliance.
7. **Implementation of a Scoring and Alert System**
    - Developing a fraud scoring system where transactions are scored based on risk levels (e.g., low, medium, high risk) rather than a simple binary classification could provide more flexibility for fraud analysts.
    - The system could send real-time alerts for high-risk transactions, allowing financial institutions to take immediate preventive actions.
8. **Continuous Model Monitoring and Evaluation**
    - Implementing model monitoring to track performance metrics over time, such as accuracy, precision, and recall, will help detect any performance degradation, especially if data distribution changes over time (concept drift).
    - Automated retraining pipelines could be set up to initiate model retraining if performance falls below a certain threshold, maintaining the model's effectiveness.

These future enhancements could significantly improve the model's robustness, scalability, and adaptability, making it an even more effective tool for detecting credit card fraud in real-world applications.

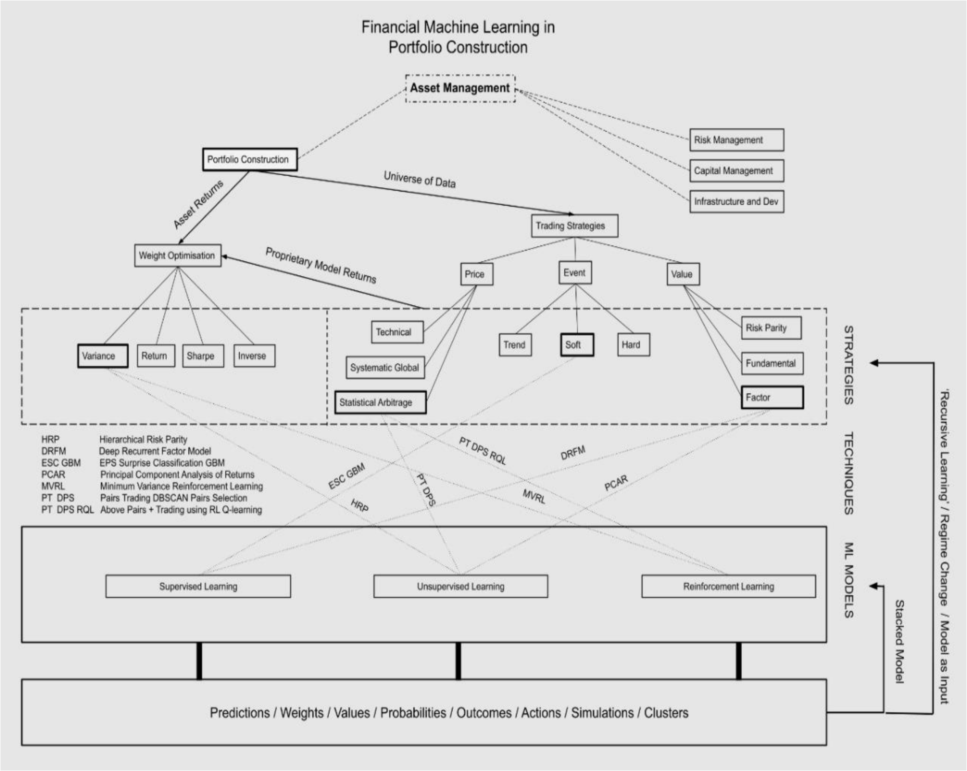

Financial Machine Learning in
Portfolio Construction
Asset Management
Risk Management
Capital Management
Infrastructure and Dev
Portfolio Construction
Universe of Data
Asset Returns
Trading Strategies
Weight Optimisation
Proprietary Model Returns
Price
Event
Value
Variance
Return
Sharpe
Inverse
Technical
Systematic Global
Statistical Arbitrage
Trend
Soft
Hard
Risk Parity
Fundamental
Factor
STRATEGIES
TECHNIQUES
ML MODELS
HRP Hierarchical Risk Parity
DRFM Deep Recurrent Factor Model
ESC GBM EPS Surprise Classification GBM
PCAR Principal Component Analysis of Returns
MVRL Minimum Variance Reinforcement Learning
PT DPS Pairs Trading DBSCAN Pairs Selection
PT DPS RQL Above Pairs + Trading using RL Q-learning
ESC GBM
HRP
PT DPS
PT DPS RQL
MVRL
DRFM
PCAR
Supervised Learning
Unsupervised Learning
Reinforcement Learning
Predictions / Weights / Values / Probabilities / Outcomes / Actions / Simulations / Clusters
Stacked Model
'Recursive Learning' / Regime Change / Model as Input

Figure 10: Future Works

REFERENCES

[1] Aisha Abdallah, Mohd Aizaini Maarof, and Anazida Zainal. Fraud detection system: A survey. *Journal of Network and Computer Ap- plications*, 68:90–113, 2016.

[2] Zeeshan Akram, Mamoona Majid, and Shaista Habib. A systematic literature review: usage of logistic regression for malware detection. In *2021 International Conference on Innovative Computing (ICIC)*, pages 1–8. IEEE, 2021.

[3] Abdulalem Ali, Shukor Abd Razak, Siti Hajar Othman, Taiseer Ab- dalla Elfadil Eisa, Arafat Al-Dhaqm, Maged Nasser, Tusneem Elhassan, Hashim Elshafie, and Abdu Saif. Financial fraud detection based on machine learning: a systematic literature review. *Applied Sciences*, 12(19):9637, 2022.

[4] Credit card fraud detec- tion using machine learning algorithms. *Procedia computer science*, 165:631–641, 2019.

[5] Ella Mae Matsumura and Robert R Tucker. Fraud detection: A theoretical foundation. *Accounting Review*, pages 753–782, 1992.

[6] Min Seok Mok, So Young Sohn, and Yong Han Ju. Random effects logistic regression model for anomaly detection. *expert systems with applications*, 37(10):7162–7166, 2010.

[7] Ruttala Sailusha, V Gnaneswar, R Ramesh, and G Ramakoteswara Rao. Credit card fraud detection using machine learning. In *2020 4th international conference on intelligent computing and control systems (ICICCS)*, pages 1264–1270. IEEE, 2020.

[8] Dejan Varmedja, Mirjana Karanovic, Srdjan Sladojevic, Marko Arsen- ovic, and Andras Anderla. Credit card fraud detection-machine learning methods. In *2019 18th International Symposium INFOTEH-JAHORINA (INFOTEH)*, pages 1–5. IEEE, 2019.

[9] Yun Wang. A multinomial logistic regression modeling approach for anomaly intrusion detection. *Computers & Security*, 24(8):662–674, 2005.