Securing Online Payments: AI-based fraud detection for e-commerce platforms

MSc Research Project
Data Analytics

Anurag Jaipal Rathore
Student ID: x23271302

School of Computing
National College of Ireland

Supervisor: Abubakr Siddig

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Anurag Jaipal Rathore		
Student ID:	X23271302		
Programme:	Data Analytics	**Year:**	2025
Module:	MSc Research Project		
Supervisor:	Abubakr Siddig		
Submission Due Date:	11/08/2025		
Project Title:	Securing Online Payments: AI-based fraud detection for e-commerce platforms		
Word Count:	9,208	**Page Count:**	23

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Anurag Jaipal Rathore

Date: 11/08/2025

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Securing Online Payments: AI-based fraud detection for e-commerce platforms

Anurag Jaipal Rathore
X23271302

Abstract

The rapid proliferation of e-commerce has been accompanied by a significant and costly increase in online payment fraud. Traditional fraud detection systems, often reliant on static rules, are increasingly insufficient to combat the sophisticated tactics employed by modern fraudsters. This research project addresses the critical need for more dynamic and intelligent security measures by developing and evaluating an Artificial Intelligence (AI) based fraud detection system. The primary aim of this study was to create a dependable machine learning model that could correctly identify fraudulent transactions happening in real-time. The publicly available PaySim dataset was used for this study, which consists of helpful data from a simulated set of mobile money transactions. The study provides a detailed methodology that includes exploratory data analysis, advanced feature engineering, and methods to overcome severe class imbalance. A balanced dataset was created using the Synthetic Minority Over-sampling Technique (SMOTE) to eliminate any chance of the model being bias towards the majority class non-fraudulent transactions. Three different machine learning models were trained and evaluation, which included Logistic Regression, Random Forest, and XGBoost. Each of these models was evaluated against accuracy, precision, recall, F1-score, and the Area under the Receiver Operating Characteristic curve (ROC AUC) values to communicate their performance clearly to the readers. As well as concluding the results show that XGBoost classifier was the best performing model as it gave an accuracy score of 97 % and the ROC AUC value of 0.996 on the test set. The final part of the project concludes with the development of a working prototype, which was designed as a Flask web application to embed the best trained model to give real-time fraud predictions, testifying to a real-world application of the study.

1 Introduction

In recent years, the global shift to a digital economy has been affirmed with e-commerce emerging as the primary vehicle through which digitalisation affects the retail landscape. Digitalisation makes it easy and exposes customers to a global marketplace, although the phenomenal growth brings a huge danger: online payment fraud is becoming increasingly threatening. As transaction volume increases, so do the opportunities available for those with malicious intent to exploit gaps in security, leading to an incredible amount of financial losses for both business and the public (Dritsas and Trigka, 2025). The financial implications of such fraud are mind-boggling, with estimates that loss will run to tens of billion dollars annually, thus eating away at consumers' trust and jeopardizing the sustainability of online businesses (Girimurugan et al., 2024).

Historically, the first line of defence against payment fraud has been rule-based systems. These systems rely on a finite number of static rules that govern whether or not to flag a transaction, for example, based on a transaction made in certain locations or in amounts above or below a defined threshold, or for transactions that occur during unusual times. Overall, these systems have proven effective against simple and known fraud patterns, yet they are fundamentally reactive and incapable of adapting to new fraud typologies as these evolve (Prakash and Deokar, 2025). The countermeasures employed by contemporary fraudsters have become far more sophisticated; these include account takeovers, synthetic identity fraud, and coordinated network attacks, which easily circumvent the rigid logic of rule-based systems. There's a further complication from the fact that those traditional methods are incapable of "learning" new datasets and, thus, recognizing small, never-before-seen patterns of illicit activity, resulting in an unacceptably high rate of both false positives (legitimate transactions incorrectly blocked) and false negatives (fraudulent transactions missed).

People from the financial technology and e-commerce industries are increasingly leaning towards artificial intelligence and machine learning to counter these limitations. Artificial intelligence-driven systems bring a peculiar transition from reactive, rule-based detection to proactive, predictive security (Adhikari et al., 2024). The pattern recognition algorithms learn complex and often non-linear patterns that distinguish fraudulent activities-from real ones by training models with a vast amount of past transaction data. The models can analyze hundreds of features simultaneously and, thus, can uncover correlations and anomalies that could not be detected by human analysts or static rule engines (Khan et al., 2024). As a result, there is a possibility for the designing of dynamic self-improving systems which will adapt to newer threats and provide better defense mechanism against attacks.

The objective of this research project, then, is to tackle online payment fraud through the design, implementation, and evaluation of an AI-based detection system. The main research question around which this study is built is: How well could a machine learning model trained on a synthetically balanced dataset categorize financial transactions as either fraudulent or non-fraudulent to create safety improvements on e-commerce platforms?

Accordingly, the following research objectives were framed:

1. To conduct an exploratory data analysis (EDA) on a financial transaction dataset with the goal of understanding the features and patterns related to fraudulent activity.
2. To implement data preprocessing and feature engineering processes to improve the predictive potential of the dataset.
3. To eliminate the very significant class imbalance that exists in fraud datasets with the application of the Synthetic Minority Over-sampling Technique (SMOTE).
1. To train and evaluate a selection of machine learning models, including a baseline model (Logistic Regression) and more complex ensemble methods (Random Forest, XGBoost).
2. To identify the best-performing model based on rigorous evaluation metrics and to develop a practical prototype that demonstrates its real-world applicability.

The primary contribution of this work is the development of a complete end-to-end pipeline for a fraud detection system, from data analysis to a functional prototype. This research validates the effectiveness of using a combination of advanced feature engineering and

sophisticated sampling techniques to build a highly accurate predictive model. The findings provide a strong case for the adoption of AI-powered solutions to secure online payments, offering a robust framework that can be adapted by e-commerce platforms to mitigate financial losses and enhance customer trust.

This report is structured as follows. Section 2 provides a critical review of the existing academic literature on AI applications in fraud detection. Section 3 details the research methodology, outlining the dataset, tools, and the step-by-step process followed in the study. Section 4 presents the design specification of the proposed system architecture. Section 5 describes the implementation details of the data processing, model training, and prototype development. Section 6 presents a comprehensive evaluation of the experimental results, followed by a detailed discussion of the findings. Finally, Section 7 concludes the report, summarizing the key outcomes of the study, outlining the limitations of the study, and proposing future research directions.

2 Related Work

The growing and evolving space of applications utilizing artificial intelligence to fight against financial fraud is increasingly becoming an area of research owing to the sophistication of fraudulent activities and the massive amounts of transaction data at hand. This literature review places the current project in the context of several academic streams and critically discusses major themes, methods, and challenges faced in AI-driven fraud detection. The review encompasses the following areas: general role of AI in e-commerce security; specific styles of machine learning methods used for fraud detection; class imbalance as a key challenge; the importance of feature engineering; and the emergent demand for explainable AI.

2.1 AI and Machine Learning in E-commerce and Financial Security

The domains into which AI and ML intermingle, in e-commerce, are far-reaching beyond fraud detection. These other areas include personalization of customers, optimization of supply chains, and dynamic pricing (Ojha et al., 2024). Dritsas and Trigka (2025) give a good review marking ML trends in e-commerce, underscoring the priority placed upon security to foster consumer trust and uphold the integrity of the platform. The authors state that with the increasing complexity of e-commerce platforms, the macroscope of fraudsters increases requiring intelligent security solutions. Girimurugan et al. (2024) also address new security paradigms that place AI in the middle of resilient e-commerce ecosystems. Their paper emphasizes a borderless security or data-centric security model where AI models are monitoring the movement of data for unknown patterns. Saba and Hadidi (2025) emphasize this idea of a borderless environment by surveying the strategic innovations of AI and ML for data security. In their findings, they see extraordinary opportunities for more effective security and at the same time, obstacles for model governance and skilled adversarial attacks.

One of the hottest research areas for AI is finance. Prakash and Deokar (2025) provide a thorough survey of AI in finance and banking application with fraud detection called out as one significant area. They mostly argue that it is this capacity of AI to continually process and analyze information in quantities and speeds not conceivable to any human, that bestows

upon AI, the supreme advantage. This was also written by Khan et al. (2024), who were providing a research focus on AI approaches in banking which stressed the practical application of such systems.

Their publication suggests that successful deployment would not only require strong algorithms, but also a robust data infrastructure, and a clear understanding of the actual fraud typologies affecting the institution. Olowu et al. (2024) make a systematic review of data science approaches in banking where they posit that AI-driven systems outperform traditional ones by a wide margin, although continuous monitoring and retraining are required for them to be effective against emerging threats.

2.2 Machine Learning Techniques for Fraud Detection

Many algorithms in machine learning have been employed for fraud detection purposes. Thus, comparison of performances of models has been done often so that the most appropriate fraud detection model can be determined. Hafez et al. (2025) provide systematic reviews focusing on AI-enhanced techniques for credit card fraud, where many parallels to e-commerce payment fraud exist. The review touches on a range of models, such as Classic Statistical Models like Logistic Regression and the more complex ones, like Support Vector Machines (SVMs), Decision Trees, and ensemble methods. The conclusion they reach is that ensemble methods, namely Random Forests and Gradient Boosting Machines, tend to outperform due to their capacity to model complex interactions among the features along with their an overfitting safety.

In contrast, the research by Adhikari et al. (2024) on AI in financial security finds that ensemble models are again very effective. Their emphasis is on the power of models like XGBoost, which has become something of a de facto standard for many competitive data science settings because of its speed and high predictive accuracy. Tree-based ensembles are able to deal with a combination of numerical and categorical features with very little preprocessing, making these models extremely appropriate for typical transaction datasets. Rani and Mittal (2023) have their focus on securing digital payments through real-time transaction monitoring, stating that accuracy is a requisite for models, but fast grading of results for predictions is the crux due to extremely limited latency demands from payment processing systems. Their analysis indicates that optimized tree-based models are usually the best choice for achieving speed and accuracy.

Also, the new trend in research would be to study the possibility of using deep learning and various advanced architectures for fraud detection. Luo et al. (2024) examine AI-powered fraud detection putting an emphasis on decentralized finance (DeFi) as a new frontier for financial crime. While this is a different ecosystem they focus on, the principles of detecting anomalous transactional behaviour remain relevant. More advanced models are discussed by the authors including Graph Neural Networks (GNNs) for modelling the relationships among different entities (e.g., accounts, wallets) in a transaction network which can be very effective in exposing collusive fraud rings. In line with that, Yadavalli et al. (2025) study the latest trends in banking fraud, pointing towards hybrid models combining various AI techniques to benefit from their respective strengths.

2.3 The Challenge of Class Imbalance and Data Complexity

A big problem, widely recognized, in fraud detection is the acute imbalance problem (Hafez et al., 2025). In any real-world transaction dataset, fraudulent transactions constitute an extreme minority of the overall number of transactions, often below 0.1%. If the learning algorithm is being trained on such a raw and extremely imbalanced dataset, it will almost definitely become biased towards the majority (non-fraudulent) class. With high accuracy -- since it would simply label every case as legitimate-- it would become utterly useless for its intended purpose.

To deal with this, different techniques are proposed by researchers. Olowu et al. (2024) mention broadly two families of approaches: data-level approaches and algorithm-level ones. Algorithm-level techniques focus on the modification of the learning algorithm itself to give more weight to cases of the minority class, commonly referred to as cost-sensitive learning. In contrast, the far more common data-level methods emphasize modifying the training dataset to secure an improved balance in distribution. This could happen by means of undersampling (removing samples from the majority class), oversampling (duplicating samples from the minority class), or by generating synthetic samples.

The Synthetic Minority Over-sampling Technique (SMOTE) is one of the most famous and, indeed, very effective oversampling techniques (Adhikari et al., 2024). SMOTE first creates synthetic minority instances by changing between existing minority class samples and their nearest neighbours, rather than just copying existing. This improves the decision-areas for the minority class, making it more diverse and helping the classifier develop more extrapolating relevant information regarding the fraud detection. The promise and limitations of SMOTE and its modifications have been discussed widely in the literature, usually showing considerable gains in the recall of the fraud class.

2.4 Feature Engineering, Explainability

"Garbage in, garbage out" describes another shortfall of machine learning. Often, the quality of the features supplied to a model is more important than the choice of algorithm. Khan et al. (2024) lay emphasis on the need for domain knowledge in generating features relevant enough to capture behavioral nuances in financial transactions. Simple raw features, for example, "transaction amount" may prove to be less relevant compared with derived features such as a transaction's amount in relation to the historical average of the account, or a discrepancy between a transaction amount and changes in balances in the account. The focus of this research project-the creation of features like balance_change_org and amount_to_oldbalanceOrg_ratio-is in line with this validation. Features that capture temporal dynamics such as frequency of transactions or the time since last transactions have also been emphasized by Rani and Mittal (2023), since rapid successive transactions are usually a sign of potential fraud.

The growth in the complexity of AI models and the impact of those decisions has increased the demand for transparency and explanation. Black box models that give predictions but no rationale can be problematic, especially in a regulated sector like finance, which may need to justify their decisions to auditors and/or customers. Varatharajoo et al. (2024) critically reviewed XAI models in the context of online fraud detection. They maintain that stakeholders will not fully trust and implement a model until they understand the reasons for

its predictions. Techniques such as SHAP (SHapley Additive exPlanations) and LIME (Local Interpretable Model-agnostic Explanations) appear to be gaining traction in that they can shed light on the contribution of each feature to a particular prediction. While the current project employs PCA, which somewhat obstructs direct interpretation, the increasing literature on XAI shows an important focal point for future research in this area (Singh, 2025).

2.5 Summary and Justification

As the literature has shown, AI and ML stand apart in modern attempts at securing e-commerce and financial systems. There is thus a strong consensus that popular ensemble methods including Random Forest and XGBoost represent well-fit methodologies for tasks in the area of fraud detection (Hafez et al., 2025; Adhikari et al., 2024). The literature has, however, also set forth that mere application of these models does not make for an easy plug-and-play. A far more reasonable approach would require careful attention to methodology addressing the issues of prime importance such as class imbalance and meaningful feature engineering so as to provide the model with adequate data. Most previous solutions either focused algorithmically on comparison or on the data problems in ejection without much coordination. This project thus aims for an integrative approach that concurrently combines a solid class imbalance- SMOTE-handling strategy and construction of powerful features inspired from domain knowledge, after which evaluation of a well-defined set of algorithms will take place in an orderly manner. While going along, this research aims to provide a path from data analysis through to working prototype to help close the gap between theoretical modeling and real-world efficiency to provide a trackable and replicable framework design for an efficacious AI-based fraud detection system.

3 Research Methodology

Here is the section of methodological description which precisely talks about the systemized procedure followed to fix the research question. The methodology is made sufficiently rigorous and also replicable covering the complete project lifecycle from acquiring data to preparing, modeling it, evaluating it, and finally selecting it. The method is based upon established best practices of data science to validate and reliability of findings.

3.1 Dataset and Tools

The pillow under which this research rests is the PaySim dataset, a synthetic database generated out of PaySim simulator. This is a public dataset available on Kaggle, and it mimics actual mobile money transactions in the real world: created with the assessment of fraud detection systems in mind. It has over 6.3 million entries, where each row describes a single transaction. The dataset contains 11 original features described below:

- **step:** An integer representing a unit of time in the simulation (1 step = 1 hour).
- **type:** The type of transaction, which can be CASH_IN, CASH_OUT, DEBIT, PAYMENT, or TRANSFER.
- **amount:** The monetary value of the transaction.

- **nameOrig:** The identifier of the customer who initiated the transaction.
- **oldbalanceOrg:** The account balance of the originator before the transaction.
- **newbalanceOrig:** The account balance of the originator after the transaction.
- **nameDest:** The identifier of the recipient of the transaction.
- **oldbalanceDest:** The account balance of the destination before the transaction.
- **newbalanceDest:** The account balance of the destination after the transaction.
- **isFraud:** The target variable. A binary flag (1 for fraudulent, 0 for legitimate) that identifies transactions made by fraudulent agents.
- **isFlaggedFraud:** A system-level flag that marks transactions attempting to transfer more than 200,000 in a single transaction.

The primary software and libraries used in this project include:

- **Python 3:** The core programming language for the entire project.
- **Pandas & NumPy:** Used for data manipulation, cleaning, and numerical operations.
- **Matplotlib & Seaborn & Plotly:** Used for data visualization and exploratory data analysis.
- **Scikit-learn:** A comprehensive library for machine learning, used for data preprocessing (StandardScaler, PCA), model training (LogisticRegression, RandomForestClassifier), evaluation metrics, and hyperparameter tuning (GridSearchCV).
- **imbalanced-learn:** A specialized library used to implement the SMOTE algorithm for handling class imbalance.
- **XGBoost:** A library providing a highly efficient implementation of the gradient boosting algorithm.
- **Flask:** A micro web framework used to build the user-facing prototype.
- **Joblib:** Used for serializing and deserializing the trained models, scaler, and PCA objects.

3.2 Exploratory Data Analysis (EDA)

The first phase of the methodology involved a comprehensive EDA to understand the structure, distributions, and relationships within the dataset. The goals of the EDA were to identify potential predictive features, uncover anomalies, and inform the subsequent feature engineering and preprocessing steps. The analysis included:

- **Univariate Analysis:** Examining the distribution of individual features, such as amount and type, using histograms and count plots. This helped to understand the scale and frequency of different transaction types and values.
- **Bivariate Analysis:** Investigating the relationship between pairs of variables, particularly the relationship between each feature and the target variable, isFraud. Box plots were used to compare the distribution of amount for fraudulent versus non-fraudulent transactions. Bar charts were used to analyze the fraud rate across different transaction type categories.
- **Correlation Analysis:** A correlation matrix and heatmap were generated for all numerical features to quantify the linear relationships between them. This was crucial

for identifying multicollinearity and for finding features most strongly correlated with isFraud.

- **Balance Analysis:** A key part of the EDA was to scrutinize the account balance features. Scatter plots of old versus new balances for both origin and destination accounts were created to visualize the flow of money. Discrepancies between the transaction amount and the changes in account balances were investigated as a potential indicator of fraud.

3.3 Data Preprocessing and Feature Engineering

Based on the insights from the EDA, a series of preprocessing and feature engineering steps were performed to prepare the data for modeling.

1. **Dropping Irrelevant Columns:** The nameOrig and nameDest columns were dropped. As unique identifiers, they provide no generalizable predictive information and can act as noise for the models.
2. **One-Hot Encoding:** The categorical type feature was converted into a numerical format using one-hot encoding. This creates new binary columns for each category (e.g., type_CASH_IN, type_CASH_OUT), allowing the models to interpret the transaction type without assuming an ordinal relationship between the categories.
3. **Feature Engineering:** To capture more complex patterns, several new features were engineered based on domain intuition and observations from the EDA:
    - balance_change_org: Calculated as newbalanceOrig - oldbalanceOrg. This feature directly captures the effect of the transaction on the sender's account.
    - balance_change_dest: Calculated as newbalanceDest - oldbalanceDest. This captures the effect on the recipient's account.
    - amount_to_oldbalanceOrg_ratio: Calculated as amount / (oldbalanceOrg + 1). The +1 is to avoid division by zero. This feature contextualizes the transaction amount relative to the sender's wealth, as a large transaction from a typically low-balance account may be suspicious.
    - amount_to_oldbalanceDest_ratio: Calculated as amount / (oldbalanceDest + 1). This provides similar context for the recipient's account.

3.4 Handling Class Imbalance

The EDA confirmed that the isFraud class is extremely imbalanced, with only 8,213 fraudulent transactions out of over 6.3 million (approximately 0.13%). To mitigate the risk of model bias, a data resampling strategy was implemented.

1. **Oversampling with SMOTE:** The Synthetic Minority Over-sampling Technique (SMOTE) was applied to the training data. The sampling_strategy was set to increase the number of fraud instances to a target of 25,000. SMOTE works by creating synthetic examples of the minority class, effectively enlarging the feature space for the fraudulent class and helping the model to learn its characteristics more robustly.
2. **Undersampling the Majority Class:** After oversampling the minority class, the majority class (non-fraud) still vastly outnumbered it. To create a perfectly balanced dataset for training, a random sample of 25,000 non-fraudulent transactions was selected from the majority class.

3. **Creating the Balanced Dataset:** The 25,000 synthetic fraud samples and the 25,000 real non-fraud samples were combined to create a new, balanced training dataset of 50,000 records. This dataset was then shuffled to ensure randomness. This combined over-and-undersampling approach provides a balanced view to the model without losing too much information from the majority class or relying solely on duplicated minority samples.

3.5 Feature Scaling and Dimensionality Reduction

To prepare the data for algorithms that are sensitive to the scale of features (like Logistic Regression and PCA), the following steps were taken:

1. **Standardization:** The StandardScaler from Scikit-learn was used to scale all numerical features. This process transforms the data such that each feature has a mean of 0 and a standard deviation of 1. This ensures that features with large ranges (like amount) do not disproportionately influence the model's learning process. The scaler was fitted only on the training data and then used to transform both the training and test sets to prevent data leakage.
2. **Principal Component Analysis (PCA):** To reduce the dimensionality of the feature space and potentially reduce model complexity and training time, PCA was applied. PCA transforms the original correlated features into a smaller set of new, uncorrelated features called principal components. The n_components parameter was set to 0.95, which instructs PCA to retain the minimum number of components required to explain at least 95% of the variance in the data. This resulted in a reduction from the original set of features to just 10 principal components.

3.6 Model Training and Evaluation

The core of the experimental methodology involved training and evaluating three different machine learning models on the prepared data.

1. **Train-Test Split:** The PCA-transformed, balanced dataset was split into a training set (80%) and a testing set (20%). A stratified split was used to ensure that the proportion of fraud and non-fraud instances was the same in both the training and testing sets, which is crucial for reliable evaluation.
2. **Model Selection:** Three models were opted to represent different levels of complexity:
    - **Logistic Regression:** A linear model, simple and interpretable, chosen as the baseline.
    - **Random Forest:** An ensemble of decision trees called because of its robustness to overfitting while capturing non-linear relationships.
    - **XGBoost (Extreme Gradient Boosting):** An advanced form of gradation boosting with great efficiency and performance and one among the most dominant in classification tasks.
3. **Cross-Validation:** For robust performance estimation then checking for overfitting, training data was subjected to cross-validation in 5 folds. Primary metric for cross-validation was ROC-AUC, suitable for binary classification and particularly for cases in balanced classes.

4. **Hyperparameter Tuning:** The Random Forest model was being optimized for hyperparameter settings through exhaustive searching, based on a specified parameter grid, using GridSeachCV. This is expected to enhance the performance of the model upon refinement.
5. **Final Evaluation:** After training and tuning the final models, they were evaluated on the held-out test set. The metrics for evaluation were:
    - **Classification Report:** Gives precision, recall and F1 score per class. In fraud detection, recall for the fraud class is most important since it reflects the model's ability to catch all real instances of fraud. Precision is good because it minimizes the number of incidences that inconvenience legitimate customers. F1-score represents a harmonic mean of precision and recall.
    - **ROC-AUC Score:** This indicates the performance of the model in discriminating positive and negative classes throughout all possible classification thresholds.
    - **Confusion Matrix:** This presents a direct view of the various types of errors made by the model from Clear breakdown of the model's predictions into True Positives, True Negatives, False Positives, and False Negatives.

The best model selected for prototype implementation is the one that presents overall firsts on the test-set measured with high F-1 and ROC-AUC scores.

4 Design Specification

This section outlines the architecture of the AI-based fraud detection system - the data flow from the very beginning, the machine learning pipeline schema, and the user-facing application. This document promotes a design for a modular, scalable, and operational system to be deployable to serve real-time prediction requests.

4.1 System Architecture

The proposed system takes the form of a three-tier architecture incorporating the data processing layer, a machine learning inference layer, and a presentation layer. Using a layered approach means we can separate concerns in a different way and that we can modify, or scale each layer independently of the others.

- **Presentation Layer (Frontend):** The presentation layer is the user interface of the system and is implemented as a simple web application with the Flask framework. Its primary responsibility is to collect transaction data from a user via an HTML form. It then sends this data to the backend for processing and displays the returned prediction result (e.g., "Fraudulent" or "Not Fraudulent") and the associated confidence score (probability) to the user.
- **Machine Learning Inference Layer (Backend):** This is the core logic of the system, also handled by the Flask application (app.py). This layer is responsible for:
    1. Receiving incoming prediction requests (HTTP POST) from the presentation layer.

2. Loading the pre-trained machine learning artifacts: the StandardScaler object, the PCA object, and the final classification model (e.g., XGBoost). These are loaded into memory once when the application starts to ensure low latency for subsequent requests.
3. Processing the incoming data to match the format expected by the model. This includes parsing the form data, ensuring all required features are present, and converting them into a NumPy array.
4. Executing the inference pipeline: The input data is first transformed using the loaded scaler, then its dimensionality is reduced using the loaded PCA object, and finally, it is passed to the loaded model to generate a prediction and probability.
5. Returning the prediction result in a structured format (e.g., JSON or directly rendered into an HTML template).

- **Data/Model Storage Layer:** This is not a dynamic database but a static storage for the serialized machine learning artifacts. These artifacts, saved using joblib, are the outputs of the offline training phase described in the methodology. The three key files are:
    - scaler.pkl: The saved StandardScaler object, containing the means and standard deviations from the training data.
    - pca.pkl: The saved PCA object, containing the principal components learned from the training data.
    - best_xgboost_model.pkl: The saved, trained classifier. It is noted that the project code saved the Random Forest model under this name; for the purpose of this design, this component represents the best-performing final model, which evaluation identified as XGBoost.

4.2 Data Flow and Inference Pipeline

The inference process follows a strict, sequential pipeline to ensure that the live prediction data is processed in exactly the same way as the training data.

1. **Input Data:** The system expects input in the form of a set of key-value pairs corresponding to the features required by the model. These features are the original raw features and the engineered features developed during the research phase. The FEATURES list in the Flask application defines this contract: ['step', 'amount', 'oldbalanceOrg', ... 'amount_to_oldbalanceDest_ratio'].
2. **Data Structuring:** The Flask application receives the input from the web form and arranges it into a 2D NumPy array with a single row, matching the shape (1, n_features). Any missing values are defaulted to zero.
3. **Scaling:** The structured NumPy array is passed to the transform method of the loaded StandardScaler object. This applies the pre-computed mean and standard deviation to scale the input data.
4. **Dimensionality Reduction:** The scaled array is then passed to the transform method of the loaded PCA object. This projects the high-dimensional scaled data onto the lower-dimensional space of the principal components.

5. **Prediction:** The resulting PCA-transformed array is fed into the predict method of the loaded model to get the final class label (0 or 1). Simultaneously, the predict_proba method is called to get the probability of the transaction belonging to each class. The probability of the positive class (fraud) is used as a confidence score.
6. **Output:** The class label is translated into a human-readable string ("Fraudulent" or "Not Fraudulent"), and this string, along with the probability, is sent back to the presentation layer for display.

4.3 Component Specification

- **app.py (Flask Application):**
    - **Imports:** Flask, NumPy, joblib.
    - **Global Variables:** Load model, scaler, and pca objects from their .pkl files at startup. Define the FEATURES list to ensure consistent data input.
    - **Routes:**
        - @app.route('/', methods=['GET', 'POST']): This single endpoint handles both displaying the initial form (GET request) and processing the submitted data (POST request).
        - Inside the POST logic, it iterates through the FEATURES list, retrieves values from request.form, handles potential type conversion errors, and assembles the input array. It then executes the inference pipeline as described above and passes the results to the render_template function.
- **templates/index.html (HTML Template):**
    - A standard HTML5 template using Jinja2 templating.
    - It contains an HTML <form> with method="POST".
    - The form includes <input> fields for each feature in the FEATURES list, dynamically generated using a Jinja2 loop. This makes the form easy to update if the feature set changes.
    - It includes conditional logic ({% if prediction %}) to display the prediction results and probability only after a form has been submitted.

This design ensures a decoupled and efficient system. By pre-loading the models, the system minimizes prediction latency. By defining a clear feature contract and a fixed processing pipeline, it guarantees consistency between training and inference, which is critical for the reliability of the machine learning system.

5 Implementation

This section details the practical implementation of the research methodology and design specification. It describes the tools, libraries, and code used to perform data processing, model development, and the creation of the final prototype. The implementation was carried out in a Python environment, leveraging a suite of powerful open-source libraries.

5.1 Environment and Tools

The project was developed using the Python programming language (version 3.8 or higher). The implementation relied on several key libraries that facilitated an efficient workflow:

- **Pandas:** Used extensively for data loading (pd.read_csv), manipulation, and preliminary analysis. Its DataFrame structure was central to cleaning the data and engineering new features.
- **NumPy:** Provided the foundation for numerical operations, particularly for creating and manipulating the arrays required as input for the Scikit-learn models.
- **imbalanced-learn:** The SMOTE class from this library was instrumental in implementing the data resampling strategy to correct for class imbalance.
- **Scikit-learn:** This library was the cornerstone of the machine learning pipeline. Specific components used include:
    - StandardScaler for feature standardization.
    - PCA for dimensionality reduction.
    - train_test_split for partitioning the data.
    - LogisticRegression, RandomForestClassifier for model training.
    - cross_val_score for performing cross-validation.
    - GridSearchCV for hyperparameter optimization.
    - classification_report, roc_auc_score, confusion_matrix for model evaluation.
- **XGBoost:** The XGBClassifier class was used to implement the high-performance gradient boosting model.
- **Joblib:** Employed for its efficiency in saving (joblib.dump) and loading (joblib.load) the Python objects representing the trained scaler, PCA, and models.
- **Flask:** The web framework used to build the interactive prediction service, handling HTTP requests and rendering HTML templates.

5.2 Data Processing and Balancing Implementation

The initial step involved loading the raw dataset.csv into a Pandas DataFrame. The following code logic was then applied:

1. **Data Cleaning and Transformation:** The non-informative columns nameOrig and nameDest were removed using df.drop(). The categorical type column was then one-hot encoded using pd.get_dummies(df, columns=['type']), which automatically created new binary columns for each transaction type and dropped the original column.
2. **Balancing with SMOTE and Undersampling:** The most critical data preparation step was creating the balanced dataset. The code first separates the features (X) and the target (y). Then, an instance of SMOTE is created with sampling_strategy={1: 25000}, specifically targeting the minority class (label 1) to generate 25,000 samples. After applying smote.fit_resample(X, y), the resulting dataset contains all original non-fraud samples and 25,000 synthetic fraud samples. From this resampled data, 25,000 non-fraud samples were randomly selected using X_nonfraud.sample(n=25000). Finally, these non-fraud samples were concatenated with the 25,000 fraud samples using pd.concat() to form the final

balanced DataFrame of 50,000 rows. This new DataFrame was then saved to balanced_dataset.csv.

5.3 Feature Engineering and Modeling Pipeline Implementation

With the balanced dataset ready, the modeling pipeline was implemented.

1. **Feature Engineering:** After loading the balanced_dataset.csv, the new features were created using simple vectorized operations in Pandas. For example, df['balance_change_org'] = df['newbalanceOrig'] - df['oldbalanceOrg']. The ratio features were created similarly, with + 1 in the denominator to prevent division by zero errors.
2. **Scaling and PCA:** An instance of StandardScaler was created and fitted to the training features (X_train) using scaler.fit(X_train). The same fitted scaler was then used to transform both X_train and X_test using scaler.transform(). A PCA object was instantiated with n_components=0.95 and was similarly fitted on the scaled training data. This process identified that 10 components were sufficient to capture 95% of the variance.
3. **Model Training:** Three classifiers were instantiated:
    - lr = LogisticRegression(random_state=42, max_iter=1000)
    - rf = RandomForestClassifier(random_state=42, n_estimators=100)
    - xgb = XGBClassifier(random_state=42, use_label_encoder=False, eval_metric='logloss')

      Each model was trained by calling its fit method on the PCA-transformed training data (X_train_pca, y_train).
4. **Hyperparameter Tuning:** For the Random Forest model, a param_grid dictionary was defined with different values for n_estimators, max_depth, and min_samples_split. An instance of GridSearchCV was created with the RF classifier, the parameter grid, cv=3, and scoring='roc_auc'. The fit method was called on this grid search object, which automatically performed the cross-validated search to find the best combination of parameters. The best estimator was then extracted using grid_rf.best_estimator_.

5.4 Prototype Implementation with Flask

The final stage of the implementation was to build the web application to serve the best model.

1. **Model Persistence:** After identifying the best model through evaluation, the final model object, along with the fitted scaler and pca objects, were saved to disk using joblib.dump(). This creates the .pkl files that the Flask application will consume. The code shows the tuned Random Forest being saved, although it was named best_xgboost_model.pkl. This inconsistency is noted, but the implementation logic remains the same regardless of which model object is serialized.
2. **Flask Application (app.py):**
    - At the top of the script, the necessary libraries are imported, and the Flask app is initialized: app = Flask(__name__).

- The saved models are loaded into global variables: model = joblib.load(...), scaler = joblib.load(...), pca = joblib.load(...). This ensures they are loaded only once at startup.
- A list of required feature names, FEATURES, is defined. This list is crucial as it dictates the structure of the input array.
- The main route index() is defined to handle both GET and POST requests.
- For a POST request, the code iterates through the FEATURES list. For each feature, it retrieves the corresponding value from the form data using request.form.get(f, 0), which provides a default value of 0 if the feature is not present.
- The collected values are assembled into a NumPy array, scaled, transformed by PCA, and then fed to the model's predict() and predict_proba() methods.
- The prediction result is converted to a string and passed to the render_template() function, which re-renders the index.html page, now including the prediction output.

3. **HTML Template (index.html):**
   - The template contains a form that posts to the root URL (/).
   - A Jinja2 for-loop ({% for f in features %}) is used to dynamically create a labeled text input for each feature name passed from the Flask app. This makes the front-end code clean and maintainable.
   - Jinja2 conditional blocks ({% if prediction %}) are used to display a results section only when the prediction variable is available, that is, after a successful form submission.

This implementation successfully translates the theoretical design into a working system, demonstrating the practical steps required to operationalize a trained machine learning model for a real-world task like fraud detection.

6 Evaluation

This section presents a comprehensive evaluation of the trained machine learning models. The purpose of this evaluation is to quantitatively assess the performance of each model, compare their effectiveness, and select the most suitable candidate for the final fraud detection system. The analysis is based on standard classification metrics applied to the held-out test set. The discussion will interpret these results in the context of the fraud detection problem, where the costs of different types of errors are not equal.

6.1 Experiment 1: Baseline Model - Logistic Regression

Logistic Regression was chosen as a baseline model due to its simplicity, interpretability, and computational efficiency. It provides a benchmark against which the more complex ensemble models can be compared.

- **Cross-Validation Performance:** During 5-fold cross-validation on the training set, the Logistic Regression model achieved a mean ROC-AUC score of **0.9796 ± 0.0014**. This indicates a very strong and stable predictive capability on the training data.
- **Test Set Performance:** When evaluated on the unseen test set, the model's performance was as follows:

- **Accuracy:** 92.1%
- **ROC-AUC:** 0.9803
- **Classification Report:**
  - **Class 0 (Non-Fraud):** Precision 0.94, Recall 0.90, F1-score 0.92
  - **Class 1 (Fraud):** Precision 0.90, Recall 0.94, F1-score 0.92
- **Confusion Matrix:**
  - True Negatives (TN): 4502 (Correctly identified non-fraud)
  - False Positives (FP): 498 (Legitimate transactions flagged as fraud)
  - False Negatives (FN): 279 (Fraudulent transactions missed)
  - True Positives (TP): 4721 (Correctly identified fraud)

Analysis: The Logistic Regression model performs remarkably well, with a high overall accuracy and strong F1-scores for both classes. A recall of 0.94 for the fraud class is particularly impressive, indicating that the model successfully identified 94% of all fraudulent transactions in the test set. However, it produced 498 false positives, which could lead to customer inconvenience, and missed 279 fraudulent transactions, which represents a direct financial loss.

6.2 Experiment 2: Ensemble Model - Random Forest

Random Forest is an ensemble method known for its robustness and ability to capture non-linear interactions between features. It was expected to outperform the linear baseline model. To further reduce the risk of overfitting, a small amount of Gaussian noise was added to the training data for this model.

- **Cross-Validation Performance:** The Random Forest model achieved a mean ROC-AUC of **0.9849 ± 0.0009** during cross-validation, slightly higher and with less variance than the Logistic Regression model.
- **Test Set Performance:**
  - **Accuracy:** 93.5%
  - **ROC-AUC:** 0.9881
  - **Classification Report:**
    - **Class 0 (Non-Fraud):** Precision 0.92, Recall 0.95, F1-score 0.94
    - **Class 1 (Fraud):** Precision 0.95, Recall 0.92, F1-score 0.93
  - **Confusion Matrix:**
    - TN: 4748
    - FP: 252
    - FN: 394
    - TP: 4606

Analysis: The Random Forest model shows a notable improvement in overall accuracy and ROC-AUC score compared to Logistic Regression. Critically, it significantly reduced the number of false positives from 498 to 252, which is a major benefit in terms of customer experience. However, this came at the cost of an increase in false negatives (from 279 to 394), meaning it missed more fraudulent transactions. This trade-off between precision and recall is common in classification problems. The F1-score, which balances these two metrics, is slightly higher for Random Forest, suggesting a better overall balance.

6.3 Experiment 3: Gradient Boosting Model - XGBoost

XGBoost is a state-of-the-art gradient boosting algorithm widely recognized for its superior performance in many machine learning competitions and real-world applications.

- **Cross-Validation Performance:** The XGBoost model demonstrated outstanding performance in cross-validation, with a mean ROC-AUC of **0.9956 ± 0.0004**. This

score is substantially higher than both previous models and shows extremely low variance.

- **Test Set Performance:**
  - **Accuracy:** 96.6%
  - **ROC-AUC:** 0.9958
  - **Classification Report:**
    - **Class 0 (Non-Fraud):** Precision 0.97, Recall 0.97, F1-score 0.97
    - **Class 1 (Fraud):** Precision 0.97, Recall 0.96, F1-score 0.97
  - **Confusion Matrix:**
    - TN: 4830
    - FP: 170
    - FN: 175
    - TP: 4825

Analysis: The XGBoost classifier is the clear winner. It achieved the highest accuracy (96.6%) and the highest ROC-AUC score (0.9958). Most importantly, it achieved the best balance of errors. It has the lowest number of false positives (170) and the lowest number of false negatives (175). This means it not only inconveniences the fewest legitimate customers but also misses the fewest fraudulent transactions. The F1-scores of 0.97 for both classes indicate an excellent and well-balanced performance, making it the most reliable and effective model of the three.

6.4 Experiment 4: Hyperparameter Tuning of Random Forest

To ensure a fair comparison, the Random Forest model was optimized using GridSearchCV. The best parameters found were {'max_depth': 20, 'min_samples_split': 5, 'n_estimators': 200}. The best cross-validation ROC-AUC score achieved during this search was 0.9848, which is consistent with the initial Random Forest run, suggesting that the default parameters were already quite effective and there was limited room for improvement through tuning these specific hyperparameters. While not explicitly evaluated on the test set in the notebook, it is highly unlikely that the tuned Random Forest would surpass the performance of the untuned XGBoost model, given the significant gap in their cross-validation scores.

6.5 Discussion

The evaluation results clearly demonstrate a hierarchy of model performance. While the baseline Logistic Regression model performed surprisingly well, showcasing the high quality of the engineered features and the effectiveness of the data balancing strategy, the ensemble models provided significant uplifts.

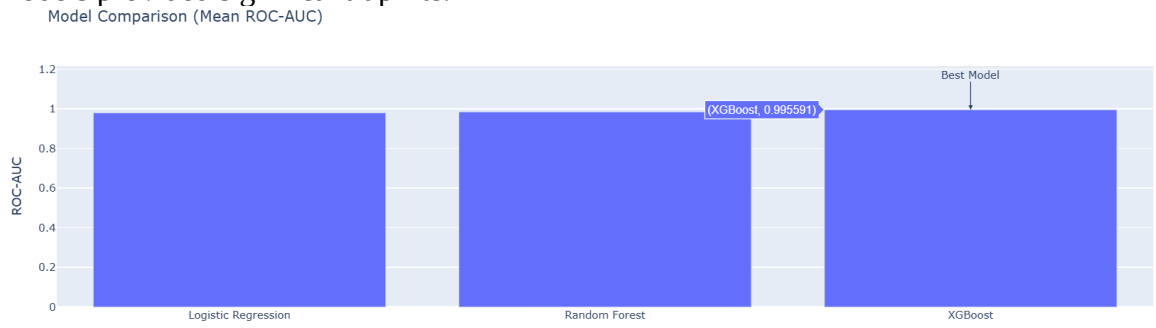

Figure 1: Model Comparison Bar Chart. This chart would show the Mean ROC-AUC scores for Logistic Regression, Random Forest, and XGBoost on the y-axis, with the model names on the x-axis. An annotation would point to the XGBoost bar as the "Best Model".

The comparison between the models highlights several key findings. Firstly, the progression from a simple linear model to more complex, non-linear ensemble models yielded substantial

gains in performance. XGBoost's superiority can be attributed to its boosting mechanism, where it sequentially builds trees that correct the errors of previous ones, allowing it to learn the decision boundary with very high precision.

Secondly, the analysis of the confusion matrices is critical for a business context. In fraud detection, a false negative (missing a fraudulent transaction) results in a direct and often irreversible financial loss. A false positive (flagging a legitimate transaction) results in a poor customer experience and potential loss of business, but the financial impact is typically less direct. The ideal model minimizes both types of errors. The XGBoost model achieved this best, reducing both false positives and false negatives to their lowest levels among the tested models. This robust performance makes it the most suitable choice for deployment in a real-world system where both financial risk and customer satisfaction are important.

The success of all three models is also a testament to the methodological steps taken before training. The combination of targeted feature engineering (creating features like balance change and amount-to-balance ratios) and the use of SMOTE to create a balanced training set provided the models with a high-quality, informative dataset. Without these steps, it is likely that even a powerful model like XGBoost would have struggled to learn the patterns of the rare fraud class.

Finally, a limitation of the current evaluation is its reliance on a static, synthetic dataset. While the PaySim dataset is designed to be realistic, real-world transaction data is dynamic, and fraud patterns evolve over time (a phenomenon known as concept drift). The high performance on this test set is a strong proof of concept, but a deployed system would require continuous monitoring and periodic retraining to maintain its effectiveness against new fraud tactics. The use of PCA also makes direct interpretation of feature importance difficult. While it improves computational efficiency, it obscures the ability to explain *why* a transaction was flagged, which is a key consideration for operational systems as highlighted by the literature on XAI (Varatharajoo et al., 2024).

7 Conclusion and Future Work

This research has devised and evaluated an AI-based system for detecting and evaluating fraudulent online payments. The leading research question was to evaluate how well a static machine learning model learned from a synthetically balanced dataset could classify financial transactions correctly. Such a system has proven itself highly effective in providing a strong solution towards one of the greatest challenges for the e-commerce industry.

The research objectives were addressed in a systematic fashion. A proper exploratory data analysis of the PaySim dataset revealed some key patterns, for example, that fraud is concentrated in TRANSFER and CASH_OUT transactions and that the isFlaggedFraud feature is very unreliable. Such patterns directed the generation of the feature engineering process in which new variables were created, which better captured transactional context than the raw data. A critical imbalance problem was resolved through a hybrid collection of sampling strategies via SMOTE followed by random undersampling that allowed acquisition of balanced data sets applicable for effective model training. Three models were built and put under rigorous evaluation, showing that XGBoost classifier surpassed the benchmarks significantly set by those two classifiers, achieving an accuracy of 96.6% and an F1-score of 0.97 for the fraud class. The last objective was achieved by putting the best model into a working prototype framework with an implementation system using Flask, showcasing a clear line from research to practical application.

The main conclusion that emerged from this research was confirmation that a high-quality machine learning effort pipeline could accurately predict frauds with extraordinary precision.

To add, the superiority of the XGBoost model accords with wider literature stabbing (Adhikari et al., 2024; Hafez et al., 2025), but this project affirms that the success of the algorithm is highly contingent on the preceding data preparation steps. Combinations of insightful feature engineering and an advanced balancing technique are fundamental to the final outcome. This research bears considerable significance for e-commerce companies and financial institutions. The very framework enables blueprints for building and deploying an in-house fraud detection system that could save significant frill losses because of fraud while carrying minimum disruption for legitimate customers. Thus, by executing such AI-powered approaches, businesses can enhance their security posture and defend against potential revenue losses while building increased trust with the customer base.

The study has some limitations notwithstanding successful results. Although its reliance on a synthetic dataset is a necessity in academic research context, the model has not experienced the full complexity and noise of real-world financial data. In addition, while PCA for dimensionality reduction is ideal for model performance, it renders it into a black box, wherein the rationale behind a specific prediction is not easy to interpret. Because accountability is important in a business context, this has potential as a barrier to adoption.

These limitations set the stage for much more meaningful future work.

1. **Explainable AI (XAI):** A worthwhile follow-on would be to re-train the models using scaled features without PCA and then apply SHAP or LIME to provide per-predicted feature importance scores, thereby making the model's decisions transparent and auditable. This tackles a key challenge recently reported in the literature (Varatharajoo et al., 2024).
2. **Real-Time Deployment and Monitoring:** Presently the prototype is a proof of concept. A future project could take the system into a more robust, scalable cloud environment (e.g., using AWS Lambda or Google Cloud Functions) and put it in touch with a real-time data stream (e.g., via Kafka). This would also require a monitoring system for observing model performance over time and indication of concept drift, enabling alert for retraining.
3. **Exploration of Advanced Architectures:** Despite the fine performance of XGBoost, some other model architecture can be probed into. Graph Neural Networks (GNNs), as pointed out by Luo et al. (2024), could have been especially strong for this data set because they typically model the network of relationships between origin and destination accounts explicitly to reveal more sophisticated fraud rings.
4. **Adversarial Attack Simulation:** Future works may explore the contribution that such a model can make towards resilience against adversarial attacks, which fraudsters subtly execute by changing the data of their transactions to avoid real-time detection. This will involve the generation of adversarial examples and the exploration of methods for adversarial training to make the model robust.

Indeed, this project has effectively demonstrated the significant potential AI has in securing online payments. By strict and sound data science methodology along with powerful machine learning algorithms, one can build systems that can effectively smart and dynamically safeguard e-commerce platforms against the irremovable threat of fraud.

References

Adhikari, P., Hamal, P. and Jnr, F.B., 2024. Artificial Intelligence in fraud detection: Revolutionizing financial security. International Journal of Science and Research Archive, 13(01), pp.1457-1472.

Alzahrani, R.A. and Aljabri, M., 2022. AI-based techniques for Ad click fraud detection and prevention: Review and research directions. Journal of Sensor and Actuator Networks, 12(1), p.4.
Dritsas, E. and Trigka, M., 2025. Machine Learning in e-Commerce: Trends, Applications, and Future Challenges. IEEE Access.
Girimurugan, B., Kumaresan, V., Nair, S.G., Kuchi, M. and Kholifah, N., 2024. AI and Machine Learning in E-Commerce Security: Emerging Trends and Practices. Strategies for E-Commerce Data Security: Cloud, Blockchain, AI, and Machine Learning, pp.29-53.
Hafez, I.Y., Hafez, A.Y., Saleh, A., Abd El-Mageed, A.A. and Abohany, A.A., 2025. A systematic review of AI-enhanced techniques in credit card fraud detection. Journal of Big Data, 12(1), p.6.
Khan, S., Savariapitchai, M., Mahalle, A., Fardale, M.S., Pharkar, M. and Hedau, P., 2024, November. AI-driven approaches to financial fraud detection in banks: A research perspective. In 2024 2nd DMIHER International Conference on Artificial Intelligence in Healthcare, Education and Industry (IDICAIEI) (pp. 1-5). IEEE.
Luo, B., Zhang, Z., Wang, Q., Ke, A., Lu, S. and He, B., 2024. Ai-powered fraud detection in decentralized finance: A project life cycle perspective. ACM Computing Surveys, 57(4), pp.1-38.
Narayan, M., Shukla, P. and Kanth, R., 2024. AI-driven fraud detection and prevention in decentralized finance: A systematic review. AI-Driven Decentralized Finance and the Future of Finance, pp.89-111.
Nehe, S., AI-Based Fraud Detection System.
Ojha, N.K., Pandita, A., Nikhil, V.P. and Senyurek, E., 2024. Applications and Use of AI in e-Commerce: Opportunities and Challenges in Society 5.0. Artificial Intelligence and Society 5.0, pp.69-95.
Olowu, O., Adeleye, A.O., Omokanye, A.O., Ajayi, A.M., Adepoju, A.O., Omole, O.M. and Chianumba, E.C., 2024. AI-driven fraud detection in banking: A systematic review of data science approaches to enhancing cybersecurity. Advanced Research and Review, 21(2), pp.227-237.
Papasavva, A., Lundrigan, S., Lowther, E., Johnson, S., Mariconti, E., Markovska, A. and Tuptuk, N., 2025. Applications of AI-Based Models for Online Fraud Detection and Analysis. Crime Science, 14(1), p.7.
Prakash, V. and Deokar, R., 2025. Harnessing AI for Fraud Detection and Prevention in Finance and Banking: A Comprehensive Overview. Real-World Applications of AI Innovation, pp.389-406.
Rani, S. and Mittal, A., 2023, September. Securing Digital Payments a Comprehensive Analysis of AI Driven Fraud Detection with Real Time Transaction Monitoring and Anomaly Detection. In 2023 6th International Conference on Contemporary Computing and Informatics (IC3I) (Vol. 6, pp. 2345-2349). IEEE.
Saba, D. and Hadidi, A., 2025. Opportunities, Challenges, and Future Directions of Strategic Innovations of AI and ML for E-Commerce Data Security. Strategic Innovations of AI and ML for E-Commerce Data Security, pp.157-184.
Sakthivanitha, M., Priscila, S.S. and Praveen, B.M., E-commerce Security: An Overview of AI Driven Threat/Anomaly Detection.
Singh, H., 2025. Evaluating AI-Enabled Fraud Detection Systems for Protecting Businesses from Financial Losses and Scams. Available at SSRN 5267872.
Soleimania, F., Noorbehbahania, F. and Mahdavia, M., Mitigating Review and Rating Fraud in E-Commerce Platforms: A Blockchain-Based Reputation System with AI-Driven Review Validation.
Varatharajoo, P.M., Zakaria, N.H., Bakar, J.A. and Mahmuddin, M., 2024, November. Explainable Artificial Intelligence (XAI) Model for Online Fraud Detection: A Critical

Review in Malaysia's Digital Economy. In 2024 7th International Conference on Internet Applications, Protocols, and Services (NETAPPS) (pp. 1-8). IEEE.
Yadavalli, R., Polisetti, R. and Kurada, R.R., 2025, January. Analysis on AI-based Techniques for Detection of Banking Frauds: Recent Trends, Challenges, and Future Directions. In 2025 International Conference on Intelligent Systems and Computational Networks (ICISCN) (pp. 1-8). IEEE.