



Credit Card Fraud Detection Using Adaptive Machine Learning and Behavioral Analytics

Prepared By

Name: Ahmad Mostafa Abutalib

ID No.: 243002097

Submitted in fulfillment of the requirements for the degree Master of
Science in Artificial Intelligence

At

Collage of Informatics

Midocean University

UAE

Supervisor Name: Dr.Amjad Qatish

Collage of Informatics, Midocean University, Comoros

Acknowledgements

First and foremost, I would like to express my deepest gratitude to Allah Almighty for granting me the strength, patience, and determination to complete this work successfully.

I am profoundly thankful to my mother, whose constant prayers and unwavering support have been my greatest source of motivation, and to my wife for her supporting, understanding, and patience throughout this journey. Without their encouragement, this achievement would not have been possible.

I would also like to extend my sincere appreciation to Dr. Amjad Qatish, whose invaluable guidance and continuous support were instrumental in shaping this research. This work would not have reached its current quality without his advice, encouragement, and expertise.

Finally, I would like to acknowledge my colleagues and friends for their helpful feedback and moral support during different phases of this project.

Abstract

Credit card fraud remains a critical challenge in today's increasingly digital economy, requiring intelligent and adaptive detection mechanisms. This thesis investigates the application of machine learning techniques for real-time fraud detection using a real-world credit card transaction dataset. The study explores multiple models, including Logistic Regression, Random Forest, and Extreme Gradient Boosting (XGBoost), addressing key challenges such as class imbalance and the need for low-latency prediction. The proposed framework integrates adaptive learning to enhance detection performance over time. Results demonstrate that machine learning models—particularly ensemble-based approaches—achieve high accuracy and AUC scores while minimizing false positives, indicating their strong potential for practical deployment in fraud prevention systems.

Contents

Acknowledgements	2
Abstract	3
List of Abbreviations.....	5
Chapter 1: Introduction.....	6
1.1 Problem Definition	6
1.2 Thesis Objectives	6
1.2 Thesis Contribution	6
1.4 Thesis Organization.....	7
Chapter 2: Overview	8
2.1 Introduction to Credit Card Fraud Credit card fraud.....	8
2.2 Types of Credit Card Fraud Fraudulent activities	8
2.3 Challenges in Traditional Fraud Detection	8
2.4 Role of Machine Learning in Fraud Detection.....	9
2.5 Machine Learning Algorithms	9
2.6 Summary	10
Chapter 3: Related Work	10
3.1 Introduction	10
3.2 Summary of Key Studies.....	10
3.3 Analysis of Limitations and Gaps	11
3.4 Conclusion.....	12
Chapter 4: Proposed Approach	12
4.1 Introduction	12
4.2 Research Problem, Questions, and Objectives	13
4.3 How will we prepare the dataset?	13
4.4 Why did we choose these models?.....	15
4.5 How will the models be evaluated?.....	Error! Bookmark not defined.
4.6 Why include Adaptive Learning?	Error! Bookmark not defined.
4.7 Action Plan	25
References	26

List of Abbreviations

- **ML: Machine Learning**
- **DL: Deep Learning**
- **RF: Random Forest**
- **LR: Logistic Regression**
- **XGB: Extreme Gradient Boosting (XGBoost)**
- **AUC: Area Under the Curve**
- **DNN: Deep Neural Network**
- **LSTM: Long Short-Term Memory**
- **SMOTE: Synthetic Minority Oversampling Technique**
- **F1-Score: Harmonic Mean of Precision and Recall**

Chapter 1: Introduction

1.1 Problem Definition

As digital transactions continue to increase, so does the threat of credit card fraud. Fraudulent transactions often mimic legitimate behavior, making them difficult to detect using traditional rule-based systems. These systems are static, produce a high number of false positives, and are often unable to adapt to new fraud tactics. This thesis addresses the need for an intelligent, data-driven approach that can learn from complex patterns in user behavior and transaction characteristics, offering real-time detection and adaptability to evolving fraud techniques.

1.2 Thesis Objectives

The main objective of this thesis is to design and develop a machine learning–based system that can detect credit card fraud in real-time with high accuracy and minimal false positives. To achieve this, the research will focus on the following sub-objectives:

- Investigate the applicability of various supervised ML and DL models in fraud detection tasks.**
- Address the class imbalance problem inherent in fraud datasets using resampling and cost-sensitive techniques.**
- Develop and test a comprehensive pipeline that includes data preprocessing, feature engineering, and model training.**
- Evaluate and compare model performance using standard metrics such as precision, recall, F1-score, and AUC.**
- Propose a scalable and adaptive fraud detection framework that can be enhanced over time with new data.**

1.2 Thesis Contribution

This research offers the following key contributions to the field of AI-driven fraud detection:

- **A comparative evaluation of machine learning algorithms for credit card fraud detection, highlighting their strengths and limitations under real-world constraints.**
- **A practical strategy to handle class imbalance, one of the most challenging issues in fraud data modeling.**
- **The development of a modular, adaptable pipeline capable of learning from new patterns and continuously improving detection accuracy.**
- **A framework that supports integration into real-time systems with low latency and high sensitivity to emerging fraud trends.**

1.4 Thesis Organization

This thesis is structured into six chapters:

- **Chapter 1 – Introduction:** Outlines the research problem, objectives, motivation, and contributions of the study.
- **Chapter 2 – Overview:** Explores the fundamentals of credit card fraud, common fraud types, challenges with traditional detection systems, and the emerging role of machine learning.
- **Chapter 3 – Related Work:** Reviews existing academic and industrial research in the domain of credit card fraud detection using AI, highlighting key methodologies, datasets, and performance gaps.
- **Chapter 4 – Proposed Approach:** Details the methodology adopted in this thesis, including dataset preparation, feature engineering, model development, and evaluation strategies.
- **Chapter 5 – Experimental Results and Discussion:** Presents the results from the machine learning models, compares their effectiveness, and discusses insights drawn from the outcomes.
- **Chapter 6 – Conclusion and Future Work:** Summarizes the thesis findings and suggests future directions for enhancing fraud detection frameworks.

Chapter 2: Overview

2.1 Introduction to Credit Card Fraud Credit card fraud

Is one of the most prevalent types of financial crime in the digital age. With the increase in online shopping, mobile banking, and digital payments, the number of fraudulent transactions has risen significantly. Fraud can be committed in various forms such as identity theft, counterfeit cards, account takeovers, and card-not-present (CNP) fraud. These activities lead to severe financial losses for institutions and customers and pose a growing threat to digital ecosystems.

2.2 Types of Credit Card Fraud Fraudulent activities

Can be broadly categorized into:

- **Card-Present Fraud:** Occurs when a physical card is used for a transaction (e.g., skimming, card theft).
- **Card-Not-Present (CNP) Fraud:** Involves online or phone transactions where the physical card is not required.
- **Application Fraud:** Involves using false or stolen information to open a credit card account.
- **Account Takeover:** A legitimate account is hijacked by an unauthorized party.
- **Synthetic Identity Fraud:** Combining real and fake information to create a new identity for fraudulent use.

2.3 Challenges in Traditional Fraud Detection

Traditional fraud detection methods rely on predefined rules or manual review systems.

While they can catch known fraud patterns, they fall short when dealing with:

- Evolving fraud tactics.
- Large-scale data streams.
- High false positive rates, leading to disruptions for legitimate users.
- Lack of adaptability to unseen fraud behaviors.

In addition to their limited adaptability, traditional fraud detection systems often suffer from delayed response times, which allows fraudulent transactions to be completed before

they are flagged. Moreover, these systems are particularly ineffective when dealing with imbalanced datasets—where fraudulent cases make up only a small fraction of total transactions—leading to poor performance in real-world settings. As fraud techniques evolve rapidly, static rule-based approaches are unable to keep up, creating a critical need for more dynamic and intelligent solutions.

2.4 Role of Machine Learning in Fraud Detection

Machine learning offers a significant advantage by learning patterns from data and adapting to new types of fraud. These systems are capable of processing large volumes of transactions in real-time and identifying anomalies based on behavioral patterns and transaction features. Benefits of ML-based fraud detection include:

- High accuracy and adaptability.
- Reduced manual intervention.
- Ability to uncover hidden patterns not captured by static rules.

In addition to pattern recognition, machine learning techniques enable real-time fraud detection through fast and scalable inference processes. This is crucial in high-throughput systems where immediate action is needed to block suspicious transactions. Furthermore, modern ML models support incremental learning, which allows systems to adapt to emerging fraud patterns over time without requiring full retraining. This flexibility ensures that fraud detection remains effective even as fraud strategies continue to evolve.

2.5 Machine Learning Algorithms

Several ML algorithms are commonly used in credit card fraud detection:

- **Logistic Regression (LR):** A baseline binary classifier, effective for initial modeling.
- **Random Forest (RF):** An ensemble method that handles non-linear relationships and reduces overfitting.
- **XGBoost (XGB):** A gradient boosting technique known for high performance and efficiency.
- **Neural Networks:** Especially LSTM for sequential pattern recognition in time-series data.

2.6 Summary

This chapter provides a general understanding of credit card fraud, its types, and the shortcomings of traditional detection methods. It also introduces how ML techniques can significantly improve fraud detection by leveraging large datasets and behavioral patterns. The next chapter will examine previous research efforts in this field to identify gaps and inform the proposed methodology.

Chapter 3: Related Work

3.1 Introduction

As credit card fraud becomes more sophisticated and widespread, academic and industrial research has increasingly focused on developing robust detection techniques. This chapter presents a review of recent studies that have addressed credit card fraud detection using various machine learning and deep learning techniques. The goal is to identify the strengths, limitations, and methodologies used in these works to inform and guide the proposed research.

3.2 Summary of Key Studies

The following table summarizes recent and high-impact studies in the domain of credit card fraud detection. Each study is evaluated based on the dataset used, methodology, model or algorithm implemented, results achieved, limitations encountered, key contributions, and source information.

No.	Study Title & Source	Methodology & Model	Results	Key Contribution	Source Link
1	An AI-driven fraud detection framework (Elsevier, Q1)	Hybrid feature extraction + XGBoost with Voting Classifier	97%	Integration of behavioral and transactional features	Link
2	Graph neural networks for fraud detection (IEEE, Q1)	Graph learning with GCN & GAT	92%	Captures relational fraud patterns in networked data	Link

No.	Study Title & Source	Methodology & Model	Results	Key Contribution	Source Link
3	Fraud Detection using Deep Neural Networks (arXiv, 2024)	Deep Learning (DNN)	98%	Strong DNN performance on synthetic data	Link
4	Credit Card Fraud Detection Using HMM (IEEE)	Sequence modeling with Hidden Markov Model	86%	First use of HMM for fraud detection	Link
5	Data Mining Approach for Fraud Detection (IEEE)	K-Means Clustering + Decision Tree	90%	Early adoption of hybrid data mining techniques	Link

3.3 Analysis of Limitations and Gaps

Although many of these studies report high AUC scores, several common limitations persist:

- Over-reliance on specific or outdated datasets that limit generalizability.
- Limited interpretability in ensemble and deep learning models.
- Persistent class imbalance leading to false positives.
- Lack of adaptability to evolving fraud patterns in real time.

In addition, most existing models focus on either transactional data or user behavior in isolation, without integrating multi-source data for contextual fraud detection. This thesis seeks to address these gaps by proposing a real-time fraud detection framework that combines transactional, behavioral, and device-level information using adaptive learning strategies.

To enhance traditional fraud detection practices, the methodology adopted in this study introduces a unified pipeline that integrates real-time behavioral analytics and transaction monitoring with device-level signals (such as location or device fingerprinting). The proposed system will utilize ensemble machine learning techniques and deep learning models to support dynamic learning, anomaly detection, and feature importance analysis. This integrated approach is designed to overcome the major weaknesses observed in

previous works—particularly the inability to adapt to new fraud tactics and the high rates of false positives in imbalanced datasets.

3.4 Conclusion

This chapter has provided a review of recent research studies on credit card fraud detection. It reveals the progress and limitations in current methodologies, providing a foundation for the proposed approach discussed in the following chapter.

Chapter 4: Proposed Approach

4.1 Introduction

What is the purpose of this chapter?

This chapter explains the methodology for building a **real-time credit card fraud detection system**. The dataset is based on the **Kaggle Credit Card Fraud Dataset**, streamed in batches to simulate live transactions. The framework is designed for real-world use in **banks, payment gateways, e-commerce, and telecom sectors**, with a focus on **accuracy, real-time detection, and adaptability**.

Why is our approach better than previous studies?

- Previous studies focused on **offline detection** and **static models**.
- Our framework supports:
 - **Real-time processing** using simulated streaming.
 - **Adaptive learning** for continuous updates.
 - **Better imbalance handling** (SMOTE + cost-sensitive learning).
 - **Deployment readiness** for real-world systems.

4.2 Research Problem, Questions, and Objectives

Research Problem	Research Questions	Research Objective
Traditional fraud detection systems are static, slow to adapt, and generate high false positives.	Q1: How can machine learning improve detection accuracy compared to rule-based systems?	Objective 1: Build a real-time fraud detection framework using ML/DL.
	Q2: Which algorithm provides the best trade-off between accuracy and speed?	
	Q3: How can adaptive learning help maintain system accuracy over time?	Objective 2: Implement adaptive learning for continuous improvement.

Summary of Objectives:

1. **Objective 1:** Develop a real-time fraud detection framework using ML/DL models.
2. **Objective 2:** Apply adaptive learning to maintain accuracy over time.

4.3 How will we prepare the dataset?

We use the **Kaggle Credit Card Fraud Dataset**, which includes anonymized transactions:

- Features: Transaction amount, time, and 28 anonymized variables.
- Fraud cases: Only **0.172%**, creating strong class imbalance.

1.Preprocessing steps:

1. Remove Missing or Invalid Entries

The dataset was first examined for missing or invalid values using `df.isnull().sum()`.

Result:

- All columns (Time, V1–V28, Amount, Class) contained **0 missing values**.
- The dataset was complete and required no removal or correction.

Summary:

- No missing or corrupted records were found.
- No cleaning actions were necessary.

2. *Normalize Continuous Variables*

The raw Time and Amount columns operate on different numeric scales, which can distort distance-based algorithms and model training.

To ensure consistent feature contribution, both columns were normalized using StandardScaler.

Summary:

- Two standardized columns were created: norm_time and norm_amount.
- The original columns were removed after normalization.
- All features now operate on a consistent numerical scale.

3. Dataset Split (70% / 15% / 15%)

Purpose:

To ensure fair training while avoiding information leakage.

Method:

- Stratified split into:
 - **70% training,**
 - **15% validation,**
 - **15% testing.**
- **SMOTE was applied only to the training set,** keeping validation and test sets untouched.

Why?

- Validation and test sets must keep their **real-world imbalance** for unbiased evaluation.
- Only the training set should be balanced artificially.

4.4 *Handle Class Imbalance Using SMOTE*

The original data was highly imbalanced (only **0.172% fraud cases**).

To correct this, SMOTE (Synthetic Minority Oversampling Technique) was applied to the training set only.

Result:

After SMOTE, the training data became perfectly balanced, with a **1:1 ratio** between fraud and non-fraud cases.

Summary:

- SMOTE generated synthetic minority samples.
- The training set is now balanced.
- This prevents the model from leaning toward non-fraud predictions.

Final Preprocessing Summary

- **No missing values** found in the dataset.
- **Time and Amount normalized**, ensuring all features share a common scale.
- **Stratified 70/15/15 split** completed.
- **SMOTE applied only on training data**, balancing fraud vs non-fraud.

- Dataset is now **clean, normalized, balanced**, and ready for machine learning & deep learning models.

4.4 Why did we choose these models?

We selected multiple models to ensure robustness across different learning behaviors and data characteristics.

The study uses a publicly available *Kaggle credit card fraud dataset* (static, not real-time).

1. Logistic Regression (LR)

A simple and interpretable baseline model.

Useful for benchmarking more advanced models.

Purpose:

Serves as a simple baseline model to evaluate how well linear separation can detect fraudulent transactions.

Results Summary:

Metric	Value	Interpretation
Accuracy	0.977	The model correctly classified about 97.7% of all transactions overall.
ROC-AUC	0.966	High discrimination ability between fraud and non-fraud cases.
Precision (Fraud = 1)	0.06	Only 6% of the predicted frauds were actually fraud — high false positives.
Recall (Fraud = 1)	0.88	The model successfully detected 88% of real fraud cases — strong sensitivity.
F1-score (Fraud = 1)	0.12	The imbalance between precision and recall led to a low F1-score.

Confusion Matrix Interpretation:

- True Negatives (41676)** → Legitimate transactions correctly identified.
- False Positives (972)** → Normal transactions flagged incorrectly as fraud.
- False Negatives (9)** → Real frauds missed by the model.
- True Positives (65)** → Actual frauds correctly detected.

Overall Insight:

- The model achieves high overall accuracy due to the large number of non-fraud cases.
- However, the **precision for fraud detection is low**, meaning it raises many false alarms.
- Despite that, its **high recall (0.88)** makes it useful for minimizing missed frauds in a first-stage detection system.

2. Random Forest (RF)

An ensemble of decision trees that captures non-linear patterns.

Reduces overfitting and provides feature importance.

Purpose:

Use an ensemble of decision trees to capture non-linear patterns and improve fraud detection performance compared to linear models.

Results Summary

Metric	Value	Interpretation
Accuracy	0.99946	Extremely high overall correctness—almost all transactions classified correctly.
ROC-AUC	0.961	Strong ability to separate fraud from normal transactions.
Precision (Fraud = 1)	0.88	88% of predicted frauds were correct → far fewer false alarms.
Recall (Fraud = 1)	0.80	The model detected 80% of actual fraud cases.
F1-score (Fraud = 1)	0.84	Good balance between precision and recall.

Confusion Matrix Interpretation

Prediction	Count
True Negatives (42640)	Normal transactions correctly identified.
False Positives (8)	Normal transactions flagged wrongly as fraud → extremely low.
False Negatives (15)	Fraud cases missed by the model.
True Positives (59)	Fraud cases correctly detected.

Overall Insight

- Random Forest significantly outperforms Logistic Regression.
- **Very low false positives (only 8)** → excellent for user-experience and operational stability.
- **Better precision, recall, and F1** → more balanced detection quality.

- Slightly lower recall than Logistic Regression, but far superior precision → fewer false alarms.

Conclusion:

Random Forest provides a powerful, stable, and highly accurate fraud detection model for tabular financial data.

3. **XGBoost**

A highly efficient boosting algorithm.

Known for strong performance in fraud detection and structured tabular datasets.

Purpose:

A boosting-based ensemble designed to iteratively correct the errors of weaker trees, achieving higher stability and precision.

Results Summary

Metric	Value	Interpretation
Accuracy	0.99853	Almost perfect overall correctness.
ROC-AUC	0.968	Best discrimination score among ML models.
Precision (Fraud = 1)	0.55	55 % of predicted frauds were true frauds.
Recall (Fraud = 1)	0.82	Detected most actual frauds (strong sensitivity).
F1-score (Fraud = 1)	0.66	Moderate harmonic balance between precision and recall.

Confusion Matrix Interpretation

- **True Negatives (42598)** → Almost all normal transactions correct.
- **False Positives (50)** → Slightly more false alarms than RF.
- **False Negatives (13)** → Few missed frauds.
- **True Positives (61)** → Detected majority of actual frauds.

Overall Insight

- XGBoost reached **the highest ROC-AUC (0.968)**, showing superb separation ability.
- It trades some precision for better recall — useful when detecting every possible fraud is more important than avoiding false alarms.
- **Conclusion:** Best overall learner among traditional ML models before moving to Deep Learning.

4. LSTM

Long Short-Term Memory networks capture sequential transaction behavior.
Ideal when the temporal order of transactions matters.

LSTM Model — Architecture, Training, and Evaluation

- **LSTM Architecture**

To capture temporal dependencies in transaction behavior, an LSTM model was implemented.

The input was reshaped into a 3-dimensional structure:

(samples, timesteps = 1, features = 30).

Model Summary

- **LSTM (32 units):** Extracts sequential patterns.
- **Dropout (0.3):** Reduces overfitting.
- **Dense (16 units, ReLU):** Learns intermediate representations.
- **Dense (1 unit, Sigmoid):** Outputs fraud probability.

Total parameters: 8,609

This architecture balances complexity and efficiency, making it suitable for large tabular fraud datasets.

- **Training Performance**

The model was trained for **10 epochs** using:

- Optimizer: **Adam**
- Loss function: **Binary Cross-Entropy**
- Batch size: **32**

Training vs Validation Performance

- Rapid improvement during early epochs.
- Validation accuracy consistently above **0.99**.
- Very low validation loss (~0.015–0.02) indicating stable learning.
- No signs of significant overfitting.

This confirms the model successfully learned sequential patterns without degrading on unseen validation data.

- **Final Evaluation on Test Set**

- **Test Metrics**
- **Accuracy: 0.9991**
- **ROC-AUC: 0.9665**

Classification Report

Class	Precision	Recall	F1-score	Support
0 (Non-Fraud)	1.00	1.00	1.00	42648
1 (Fraud)	0.71	0.82	0.76	74

Interpretation

- The model correctly identified **82% of frauds** (high recall).
- Precision for fraud is **0.71**, meaning some false positives exist—acceptable in fraud detection.
- Overall performance remains strong, especially considering extreme class imbalance.

- Confusion Matrix

	Predicted 0	Predicted 1
Actual 0	42623	25
Actual 1	13	61

Insights

- Only **25 non-fraud** transactions were flagged incorrectly.
- **61 out of 74** fraud cases were detected.
- False negatives (13 cases) remain the main risk area, but significantly reduced compared to simpler models.

Final Summary (LSTM)

- Successfully captures sequential behavior in credit card transactions.
- Achieved **~99.9% accuracy** and **ROC-AUC ~0.97**.
- Outperforms Logistic Regression and Random Forest in fraud recall.
- Slightly below XGBoost in overall precision, but better balanced for sequence modeling.

5. DNN (Deep Neural Network)

Learns complex, high-level patterns through multiple dense layers.
Provides a flexible deep learning baseline.

Model 4 – Deep Neural Network (DNN)

Purpose

A fully connected neural network designed to learn complex, non-linear patterns in transaction data.

DNN acts as a deep learning baseline before moving to more sequential models like LSTM.

Training Performance (Epochs Summary)

Across 10 epochs:

- **Training accuracy** improved steadily from **88% → 99.8%**
- **Validation accuracy** increased from **98.2% → 99.84%**

- **Validation loss** decreased initially then stabilized
→ normal behavior indicating good generalization without strong overfitting.

Results Summary

Metric	Value	Interpretation
Accuracy	0.99869	Nearly perfect performance on the test set.
ROC-AUC	0.96279	Strong ability to separate fraud vs. non-fraud.
Precision (Fraud = 1)	0.59	59% of predicted fraud cases were correct (moderate).
Recall (Fraud = 1)	0.78	Detected 78% of real fraud cases — good sensitivity.
F1-score (Fraud = 1)	0.67	Balanced performance between precision and recall.

Confusion Matrix Interpretation

	Predicted 0	Predicted 1
Actual 0	42608	40
Actual 1	16	58

- True Negatives (42608) → Almost all legitimate transactions identified correctly
- False Positives (40) → Very few false alarms
- False Negatives (16) → Some fraud cases missed
- True Positives (58) → Majority of fraud cases detected

Overall Insight

- DNN performs as strong as XGBoost in overall accuracy.
- Provides better pattern learning than Logistic Regression.
- Recall improved compared to Random Forest in some runs.
- Precision is moderate, meaning it still produces some false alarms — common in fraud datasets.

Conclusion:

DNN is a solid deep learning baseline with excellent overall accuracy and competitive recall. It sets the foundation for more sequence-aware models like LSTM, which typically perform even better on fraud patterns over time.

Overall Model Comparison — Summary & Conclusion

After training five different machine-learning and deep-learning models on the preprocessed (cleaned, normalized, SMOTE-balanced) credit-card dataset, the following conclusions were reached:

Conclusion

All models achieved very high overall accuracy because the dataset, after SMOTE balancing, becomes easier for classifiers to separate.

However, fraud detection performance must be judged by Recall & Precision on the minority class (Class = 1) — not accuracy alone.

Best overall performer → LSTM

The LSTM model delivered the strongest recall for Class 1 while keeping precision reasonable. This means:

- It catches the highest number of fraud cases
- With fewer false alarms
- And learns temporal transaction patterns better than static ML methods

Ranking based on Class-1 Detection Quality

1. LSTM (Best fraud recall & strong F1)
2. DNN (Balanced, close to LSTM but slightly lower recall)
3. XGBoost (Stable & strong, but recall lower than deep models)
4. Random Forest (Excellent precision but recall weaker)
5. Logistic Regression (Baseline only — fails on fraud cases)

Comparison Table

Model	Accuracy	ROC-AUC	Precision (Fraud)	Recall (Fraud)	F1-Score (Fraud)	Notes
Logistic Regression	0.9770	0.9658	0.06	0.88	0.12	Very high false positives, weak baseline for fraud.
Random Forest	0.99946	0.9609	0.88	0.80	0.84	High precision, slightly weaker recall.
XGBoost	0.99852	0.9679	0.55	0.82	0.66	More balanced, but still fewer fraud cases detected.
DNN	0.99868	0.9628	0.59	0.78	0.67	Learns deeper patterns, good fraud recovery.
LSTM (Best)	0.99911	0.9665	0.71	0.82	0.76	Best overall detection quality; strong on sequential patterns.

Interpretation of Table

- Accuracy alone is misleading because non-fraud cases dominate.
- Recall on fraud cases is the key metric.
- LSTM gives the best combination of Precision, Recall, and F1, making it the most useful in a real fraud-detection scenario.

Conclusion

Among all tested models, the LSTM network demonstrated the strongest fraud-detection capability, achieving the highest recall and F1-score for the minority class. While traditional ML models such as Random Forest and XGBoost performed well overall, deep learning models—particularly LSTM—were better at identifying subtle fraud patterns. Therefore, LSTM is selected as the preferred model for the final fraud-detection framework.

4.5 Model Evaluation Framework

To ensure a fair and comprehensive assessment of all models, the evaluation relies on the following core metrics:

- AUC (Area Under Curve): Measures overall model discrimination quality.
- F1-Score: Balances precision and recall, crucial in imbalanced fraud datasets.
- Precision & Recall:
 - *Precision* shows how many flagged transactions are actually fraud.
 - *Recall* shows how many real fraud cases the model captures.
- Latency: Although the dataset is static (not real-time), latency is still measured to reflect expected performance in real-world deployment.

These metrics ensure that performance is not judged only by accuracy, which is misleading in imbalanced datasets.

4.6 Why Adaptive Learning Matters

Fraud patterns constantly evolve. A static model loses accuracy over time.

To maintain long-term reliability:

- Incremental Learning: Allows the model to update on new batches of data without full retraining.
- Concept Drift Monitoring: Detects changes in fraud behavior and adjusts the model accordingly.
- Sustained Performance: Ensures accuracy, precision, and recall remain high as fraud tactics change.

Adaptive learning strengthens the final system beyond a one-time training cycle.

Model Performance Summary

After training and evaluating the three advanced models (XGBoost, DNN, LSTM), the following results were obtained.

All metrics were computed using the untouched test set to ensure unbiased evaluation.

Comparison Table — XGBoost vs DNN vs LSTM

Model	Accuracy	ROC-AUC	Precision (Fraud)	Recall (Fraud)	F1-Score (Fraud)	False Positives	False Negatives
XGBoost	0.9985	0.9679	0.55	0.82	0.66	50	13
DNN	0.9987	0.9628	0.59	0.78	0.67	40	16
LSTM	0.9991	0.9665	0.71	0.82	0.76	25	13

Interpretation of Results

1. Overall Accuracy

All models reach exceptionally high accuracy (~99.8%–99.9%) because the dataset contains mostly legitimate transactions.

Thus, accuracy alone is not sufficient for model selection.

2. Fraud Detection Performance (Key Focus)

The critical indicators for fraud detection are:

- Recall → ability to catch fraud
- Precision → ability to avoid false alarms
- F1-Score → balance between both

XGBoost

- High recall (0.82) → catches most fraud cases.
- Moderate precision (0.55).
- Balanced performance but more false positives (50).

DNN

- Better precision than XGBoost.
- Lower recall → misses more fraud.
- Strong performance but not optimal for high-sensitivity environments.

LSTM Best Overall

- Highest precision (0.71).
- Equal recall to XGBoost (0.82).
- Top F1-Score (0.76).
- Lowest false positives (25).
- Temporal pattern learning offers superior fraud sensitivity.

Conclusion

- LSTM is the strongest model for fraud detection, providing the best balance between precision and recall with minimal false alarms.
- XGBoost remains an excellent and efficient model for production due to speed and solid recall.
- DNN performs well but is not as strong as LSTM in metrics that matter for fraud detection.

Recommended deployment strategy:

- Primary model: LSTM
- Secondary fallback: XGBoost

4.7 Action Plan

Weeks	Phase	Tasks
1–2	Data Collection & Preparation	Collect dataset, clean and normalize data, handle missing values, split into train/validation/test, apply SMOTE for imbalance.
3–4	Model Development	Build baseline models (LR, RF), train advanced models (XGBoost, LSTM, DNN), start hyperparameter tuning.
5–6	Optimization & Adaptive Learning	Complete tuning, evaluate models using AUC, F1-score, latency, implement adaptive learning and streaming simulation.
7–8	Final Evaluation & Documentation	Validate best model, compile evaluation results, prepare report and presentation.
9–10	Buffer & Revisions	Resolve issues, finalize adjustments, ensure readiness for submission.

References

1. [Zhang et al., 2024]
Zhang, Wei, Liu, Fang, & Chen, Hao. “An AI-driven fraud detection framework based on hybrid features and ensemble learning”. *Future Generation Computer Systems*, Elsevier, 2024.
Available at:
<https://www.sciencedirect.com/science/article/abs/pii/S0167739X24001997>
2. [Wang et al., 2024]
Wang, Jian, Li, Zhiqiang, & Zhao, Ming. “Graph neural networks for fraud detection in transaction networks”. *IEEE Access*, 2024.
Available at: <https://ieeexplore.ieee.org/abstract/document/10595068>
3. [Kumar et al., 2024]
Kumar, Ravi, & Singh, Arjun. “Fraud Detection in Credit Card Transactions Using Deep Neural Networks”. *arXiv preprint arXiv:2405.00026*, 2024.
Available at: <https://arxiv.org/abs/2405.00026>
4. [Bhattacharyya et al., 2011]
Bhattacharyya, Siddhartha, Jha, Sanjeev, Tharakunnel, Kumar, & Westland, J.C. “Credit Card Fraud Detection Using Hidden Markov Model”. *IEEE Transactions on Dependable and Secure Computing*, 2011.
Available at: <https://ieeexplore.ieee.org/abstract/document/5762457>
5. [Ghosh & Reilly, 2011]
Ghosh, S., & Reilly, D.L. “Data Mining Approach for Fraud Detection in Credit Card Transactions”. *IEEE Computational Intelligence Magazine*, 2011.
Available at: <https://ieeexplore.ieee.org/abstract/document/595879>



Credit Card Fraud Detection Using Adaptive Machine Learning and Behavioral Analytics

Prepared By

Name: Ahmad Mostafa Abutalib

ID No.: 243002097

Submitted in fulfillment of the requirements for the degree Master of
Science in Artificial Intelligence

At

Collage of Informatics

Midocean University

UAE

Supervisor Name: Dr.Amjad Qatish

Collage of Informatics, Midocean University, Comoros

Full Pipeline Overview — End-to-End Workflow

1. Data Loading

Load the Kaggle Credit Card Fraud dataset from Google Drive.

```
from google.colab import drive
drive.mount('/content/drive')

import pandas as pd
df = pd.read_csv('/content/drive/MyDrive/creditcard.csv')
```

2. Data Inspection

Check structure, types, and missing values.

```
df.info()
df.isnull().sum()
```

Description (1 line):

Dataset contains 31 features with *no missing values*, suitable for immediate processing.

3. Removing Missing or Invalid Entries

(If any existed — but this dataset has none.)

```
df = df.dropna()
```

Description:

Ensures data quality and prevents training issues caused by incomplete rows.

4. Feature Normalization

Normalize Time and Amount because they operate on different scales.

```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
df["norm_time"] = scaler.fit_transform(df[["Time"]])
df["norm_amount"] = scaler.fit_transform(df[["Amount"]])
df = df.drop(["Time", "Amount"], axis=1)
```

Description:

Creates consistent feature scaling to improve ML/DL model stability.

5. Splitting into Train / Validation / Test

70% train — 15% validation — 15% test (stratified).

```
from sklearn.model_selection import train_test_split

X = df.drop("Class", axis=1)
y = df["Class"]

# 70% train, 30% temp
X_train, X_temp, y_train, y_temp = train_test_split(
    X, y, test_size=0.30, random_state=42, stratify=y
)

# 15% val, 15% test
X_val, X_test, y_val, y_test = train_test_split(
    X_temp, y_temp, test_size=0.50, random_state=42, stratify=y_temp
)
```

Description:

Prevents data leakage and creates fair evaluation sets.

6. Handling Imbalance Using SMOTE

Apply oversampling *only* on training data.

```
from imblearn.over_sampling import SMOTE

sm = SMOTE(random_state=42)
X_train_resampled, y_train_resampled = sm.fit_resample(X_train, y_train)
```

Description:

Balances fraud vs. non-fraud transactions to prevent bias.

7. Machine Learning Models (ML Section)

7.1 Logistic Regression

```
from sklearn.linear_model import LogisticRegression

lr_model = LogisticRegression(max_iter=1000)
lr_model.fit(X_train_resampled, y_train_resampled)
```

7.2 Random Forest

```
from sklearn.ensemble import RandomForestClassifier

rf_model = RandomForestClassifier(
    n_estimators=200,
    max_depth=None,
    random_state=42
)
rf_model.fit(X_train_resampled, y_train_resampled)
```

7.3 XGBoost

```
from xgboost import XGBClassifier

xgb_model = XGBClassifier(
    n_estimators=300,
    learning_rate=0.05,
    max_depth=6,
    subsample=0.9,
    colsample_bytree=0.9,
    eval_metric='logloss'
)
xgb_model.fit(X_train_resampled, y_train_resampled)
```

Description for ML block:

These models learn decision boundaries based on tabular data; ideal for structured fraud detection.

8. Deep Learning Models (DL Section)

8.1 Deep Neural Network (DNN)

```
from keras.models import Sequential
from keras.layers import Dense, Dropout, Input

dnn_model = Sequential([
    Input(shape=(X_train_resampled.shape[1],)),
    Dense(32, activation='relu'),
    Dropout(0.3),
    Dense(16, activation='relu'),
    Dense(1, activation='sigmoid')
])

dnn_model.compile(optimizer='adam', loss='binary_crossentropy',
metrics=['accuracy'])
```

```
dnn_model.fit(
    X_train_resampled, y_train_resampled,
    validation_data=(X_val, y_val),
    epochs=10, batch_size=256
)
```

8.2 LSTM (Sequential Transactions Learning)

Reshape to 3D

```
import numpy as np

X_train_lstm = np.expand_dims(X_train_resampled.values, axis=1)
X_val_lstm   = np.expand_dims(X_val.values, axis=1)
X_test_lstm  = np.expand_dims(X_test.values, axis=1)

Model
from keras.layers import LSTM

lstm_model = Sequential([
    Input(shape=(1, X_train_resampled.shape[1])),
    LSTM(32, return_sequences=False),
    Dropout(0.3),
    Dense(16, activation='relu'),
    Dense(1, activation='sigmoid')
])

lstm_model.compile(optimizer='adam', loss='binary_crossentropy',
metrics=['accuracy'])

lstm_model.fit(
    X_train_lstm, y_train_resampled,
    validation_data=(X_val_lstm, y_val),
    epochs=10, batch_size=256
)
```

Description DL block:

DL models capture deeper patterns, with LSTM learning temporal transaction behavior.

9. Unified Evaluation Function

(Used for ML + DL)

```
from sklearn.metrics import accuracy_score, roc_auc_score,
classification_report, confusion_matrix
import time

def evaluate(model, X_test, y_test, name, proba=True):
```

```

start = time.time()

if proba:
    y_prob = model.predict(X_test)
    y_prob = y_prob.flatten() if hasattr(y_prob, "flatten") else y_prob
else:
    y_prob = model.predict(X_test).flatten()

y_pred = (y_prob > 0.5).astype(int)
latency = time.time() - start

return {
    "model": name,
    "accuracy": accuracy_score(y_test, y_pred),
    "auc": roc_auc_score(y_test, y_prob),
    "precision": classification_report(
        y_test, y_pred, output_dict=True
    )["1"]["precision"],
    "recall": classification_report(
        y_test, y_pred, output_dict=True
    )["1"]["recall"],
    "f1": classification_report(
        y_test, y_pred, output_dict=True
    )["1"]["f1-score"],
    "confusion": confusion_matrix(y_test, y_pred),
    "latency": latency
}

```

10. Model Comparison Table

```

import pandas as pd

comparison_table = pd.DataFrame([
    # Fill with your final values
    {"Model": "XGBoost", "Accuracy": 0.9985, "AUC": 0.9679, "Precision":
0.55, "Recall": 0.82, "F1": 0.66, "FP": 50, "FN": 13},
    {"Model": "DNN", "Accuracy": 0.9987, "AUC": 0.9628, "Precision":
0.59, "Recall": 0.78, "F1": 0.67, "FP": 40, "FN": 16},
    {"Model": "LSTM", "Accuracy": 0.9991, "AUC": 0.9665, "Precision":
0.71, "Recall": 0.82, "F1": 0.76, "FP": 25, "FN": 13},
])

```

comparison_table

11. Visualization (Latency, AUC, F1)

```
import matplotlib.pyplot as plt

def plot_latency(results):
    names = [r["model"] for r in results]
    latency = [r["latency"] for r in results]

    plt.figure(figsize=(8,4))
    plt.bar(names, latency)
    plt.title("Model Latency Comparison")
    plt.ylabel("Seconds")
    plt.show()

def plot_auc_f1(results):
    names = [r["model"] for r in results]
    aucs = [r["auc"] for r in results]
    f1s = [r["f1"] for r in results]

    plt.figure(figsize=(12,4))

    plt.subplot(1,2,1)
    plt.bar(names, aucs, color="steelblue")
    plt.title("ROC-AUC Comparison")

    plt.subplot(1,2,2)
    plt.bar(names, f1s, color="orange")
    plt.title("F1 Score Comparison")

    plt.show()
```

12. Adaptive Learning (Incremental Updates)

```
from sklearn.linear_model import SGDClassifier

adaptive_model = SGDClassifier(loss="log_loss")

# initial training
adaptive_model.partial_fit(X_train, y_train, classes=[0,1])

# incremental updates
for i in range(0, len(X_val), 5000):
    adaptive_model.partial_fit(X_val[i:i+5000], y_val[i:i+5000])
```

Description:

Allows model updates without full retraining to adapt to evolving fraud patterns.

