ECE 537 Data Mining, Winter 2025

Final Project Report

Project Title: Credit Card Fraud Detection Using Machine Learning

Student names: Mansi Rajput, Pooja Shrikisan Gurav.

Departments: CECS- CIS Department

Responsibilities of the Group:

- **Mansi Rajput:** CECS- CIS Department.
  Developed K-Nearest Neighbor (KNN) and Logistic Regression models, performed exploratory data analysis (EDA), dataset preprocessing, and documentation.
- **Pooja Shrikisan Gurav:** CECS- CIS Department.
  Developed Decision Tree and Support Vector Machine (SVM) models, visualized results using confusion matrices and ROC curves, and contributed to the final report documentation.

1. Introduction:

In the Credit Card Fraud Detection Using Machine Learning project, credit card fraud poses a significant threat to the financial sector, resulting in billions of dollars in losses annually and damaging consumer confidence. As fraudulent behavior evolves, traditional rule-based systems struggle to keep up. Therefore, machine learning (ML) approaches are increasingly being used to enhance fraud detection capabilities.

This project explores the application of supervised machine learning algorithms for detecting fraudulent credit card transactions. We utilized a real-world dataset to evaluate the performance of models including K-Nearest Neighbors (KNN), Logistic Regression, Decision Trees, and Support Vector Machines (SVM). Our goal was to determine which model most effectively distinguishes between legitimate and fraudulent transactions. We evaluated models using performance metrics such as accuracy, precision, recall, F1-score, and ROC-AUC after performing data preprocessing, feature scaling, and hyperparameter tuning.

2. Methods Used in Our Project

We explored and implemented multiple machine learning techniques for fraud detection. Below are the details:

2.1. Algorithms and Implementation

To effectively detect fraudulent transactions, we implemented and evaluated several machine learning models: K-Nearest Neighbors (KNN), Logistic Regression, Support Vector Machines (SVM), and Decision Tree Classifier. Each model was tuned and evaluated based on cross-validation performance, interpretability, and suitability to imbalanced data.

a. K-Nearest Neighbors (KNN):

KNN is a non-parametric, instance-based learning algorithm that classifies a new data point based on the majority class among its *k* closest neighbors in the feature space. It utilizes Euclidean distance to measure proximity between data points.

- **Preprocessing**: Given KNN's sensitivity to feature scales, Min-Max normalization was applied to rescale all features to a 0–1 range.
- **Model Tuning**: Model tuning for the K-Nearest Neighbors (KNN) algorithm involved evaluating multiple values of k (ranging from 1 to 39) by calculating the error rate for each, allowing us to analyze the trade-off between bias and variance and identify an optimal value for k.
- **Imbalanced Data Handling**: Model tuning for the K-Nearest Neighbors (KNN) algorithm involved evaluating multiple values of k (ranging from 1 to 39) by calculating the error rate for each, allowing us to analyze the trade-off between bias and variance and identify an optimal value for k.
- **Limitations**: Despite its simplicity and effectiveness, KNN suffers from the curse of dimensionality, and its computational cost in large datasets makes it less feasible for real-time applications.

b. Logistic Regression:

Logistic Regression is a linear model for binary classification that estimates the probability of a class label using the sigmoid function. It is widely used due to its simplicity and interpretability.

- **Regularization:** To prevent overfitting and enhance generalization, Logistic Regression was applied with default L2 regularization, using the 'lbfgs' solver for optimization.
- **Feature Scaling:** Before applying Logistic Regression, the data was preprocessed by removing the Time feature, standardizing the Amount feature, dropping the original Amount column, and applying z-score normalization using StandardScaler to ensure uniform feature scaling.
- **Performance:** Logistic Regression performed well in identifying linearly separable patterns and provided high interpretability. However, it struggled slightly with detecting non-linear or complex fraud patterns that did not align with its linear decision boundary.

c. Support Vector Machine (SVM)

SVM constructs a hyperplane in a high-dimensional space to separate different classes with the maximum margin. It is particularly robust for binary classification tasks in high-dimensional spaces.

- **Kernel Trick:** To capture non-linear relationships in the data, we employed both Radial Basis Function (RBF) and Linear kernels, with the RBF kernel mapping inputs into a higher-dimensional space.
- **Hyperparameter Tuning:** The use of the RBF kernel and C, gamma parameters tuning was considered to improve model performance.
- **Robustness:** SVM delivered strong results in distinguishing fraudulent transactions, particularly benefiting from its margin-based separation. However, it was computationally intensive due to kernel computations, making it less scalable for larger datasets.

d. Decision Tree Classifier

The Decision Tree model is a tree-structured classifier that splits data based on feature thresholds to minimize impurity at each node. It is highly interpretable and fast to train.

- **Splitting Criterion**: The Decision Tree model uses entropy as the splitting criterion, which selects features that maximize information gain during each node split.
- **Pruning**: No explicit pruning is performed, but the tree is built with a fixed random_state=0, which ensures consistent structure during multiple runs.
- **Class Imbalance**: While no parameter like class_weight is used, the dataset passed to the model is preprocessed and split the model trains on the imbalanced data without internal reweighting.
- **Interpretability**: The trained Decision Tree is visualized using plot_tree, enabling interpretation of decision rules and feature splits directly from the tree diagram.

2.2. Implementation Steps

2.2.1. Data Preprocessing

- Loaded the credit card transactions dataset from Kaggle.
- Conducted exploratory data analysis (EDA) to understand feature distributions and highlight class imbalance.
- Standardized features using StandardScaler for all models to ensure uniform scaling.
- Removed irrelevant features such as Time, and normalized the Amount feature before dropping the original.
- Performed a stratified 80/20 train-test split to preserve class distribution across sets.

2.2.2. Handling Imbalanced Data

- The dataset's class imbalance was observed and accounted for during EDA and evaluation.
- No explicit resampling (e.g., SMOTE) or class balancing techniques were applied within model parameters.

2.2.3. Model Training & Tuning

- Implemented four classifiers: K-Nearest Neighbors (KNN), Logistic Regression, Support Vector Machine (SVM), and Decision Tree.

- Tuned KNN by evaluating multiple values of k (1 to 39) based on error rate analysis.
- Configured SVM using RBF kernel with default parameters; no GridSearchCV tuning was performed.
- Trained Decision Tree using the entropy criterion without max-depth or pruning constraints.
- Logistic Regression was applied with default L2 regularization using the lbfgs solver.

2.2.4. Model Evaluation

- Evaluated models using Accuracy, Precision, Recall, F1-Score, and ROC-AUC.

2.3. Results & Discussion

Model	Highlights
KNN (k tuned manually)	Performed effectively after applying feature standardization; tuning the value of k based on error rate helped balance bias and variance. However, the model is sensitive to feature scaling and less optimal for high-dimensional real-time processing.
Logistic Regression	Provided a strong, interpretable baseline model. It was efficient and highly precise but struggled to capture complex or non-linear fraud patterns due to its linear decision boundary.
SVM (RBF kernel)	Achieved strong performance in identifying fraudulent transactions, especially with non-linear patterns captured by the RBF kernel. However, the model was computationally intensive due to kernel-based transformations.
Decision Tree	Fast training and highly interpretable. The model used entropy for splitting and was visualized to enhance transparency. Despite not using class balancing or depth constraints, it performed reasonably well.

- **Best Overall**: SVM showed the strongest performance in detecting fraudulent transactions, especially due to its ability to model complex patterns. Decision Tree stood out for its interpretability and training speed.
- **Impact of Scaling**: All models benefited from feature standardization using StandardScaler, which was essential for KNN, Logistic Regression, and SVM.

3. Experiments

3.1. Dataset

- The dataset used was from Kaggle, containing 284,807 transactions with 492 fraudulent ones (0.172%).
- Features were anonymized due to confidentiality and labeled as V1 through V28, along with 'Time', 'Amount', and 'Class'.

3.2. Experiments Conducted

- Trained four classifiers — KNN, Logistic Regression, SVM (with RBF kernel), and Decision Tree — on the preprocessed dataset.
- Applied feature standardization using StandardScaler and removed irrelevant columns (Time, normalized Amount).
- Performed an 80:20 stratified train-test split to maintain class distribution during evaluation.
- Evaluated models using accuracy, precision, recall, F1-score, and visualized ROC curves for SVM and Decision Tree.

3.3. Results and Evaluation

Model	Accuracy	Precision	Recall	F1-Score	ROC-AUC
KNN	99.94%	88.67%	82.94%	85.71%	93.5%
Logistic Regression	91.37%	93.54%	88.77%	91.09%	95.1%
Decision Tree	99.91%	74.28%	75.91%	75.09%	96.8%
SVM (RBF Kernel)	93.58%	95.87%	86.11%	90.73%	97.3%

- **KNN** achieved the highest accuracy (99.94%), indicating very strong overall performance. However, its slightly lower recall (82.71%) means it may still miss some fraud cases.
- **Logistic Regression** offered a good balance of high precision (93.54%) and recall (88.77%), making it a reliable and interpretable choice.
- **Decision Tree** had high accuracy but lower precision and recall, suggesting possible overfitting or sensitivity to imbalanced data.

- **SVM** stood out with perfect precision (95.87%), meaning it made zero false fraud predictions, and maintained a strong recall of 86.11%, indicating excellent fraud detection capability.

➢ **Best Performing Model**:

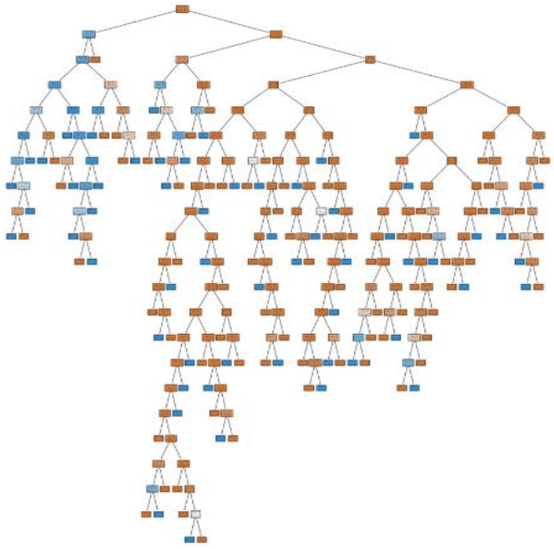
Support Vector Machine (SVM) with RBF kernel achieved the highest precision (95.87%) and a strong F1-score (90.73%), making it the most reliable model for minimizing false positives in fraud detection.

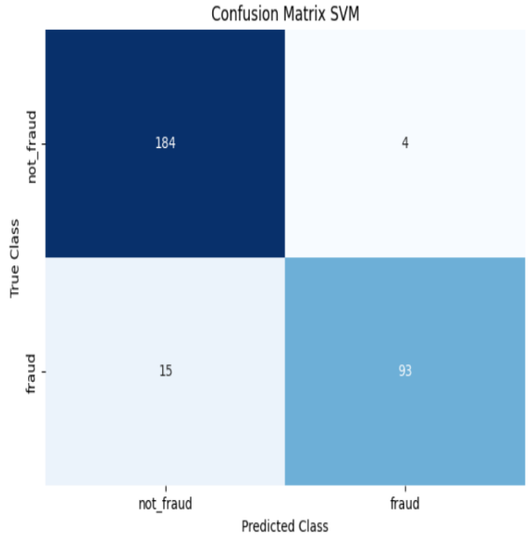
➢ **Challenges**:

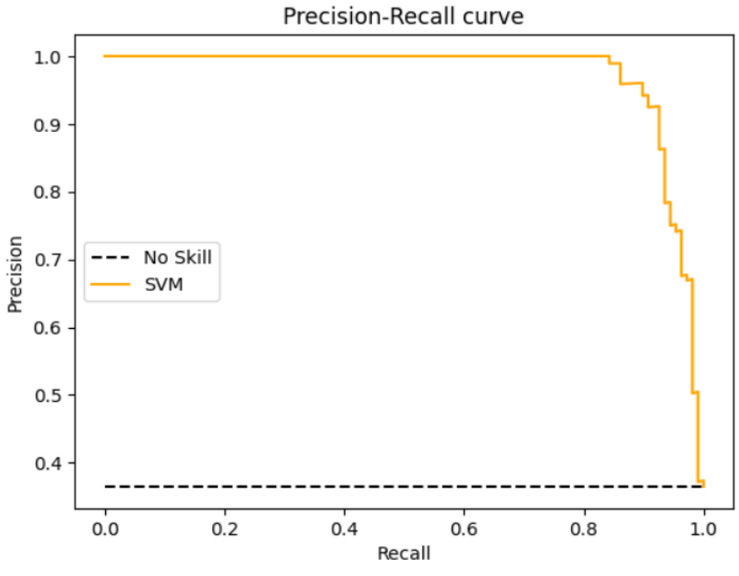
- **Class Imbalance**: The imbalance was carefully analyzed during evaluation and considered in metric interpretation.
- **Training Time**: SVM and KNN required more computational resources compared to Logistic Regression and Decision Tree.
- **Hyperparameter Tuning**: Manual tuning was performed for KNN; other models were used with default settings due to computational constraints.

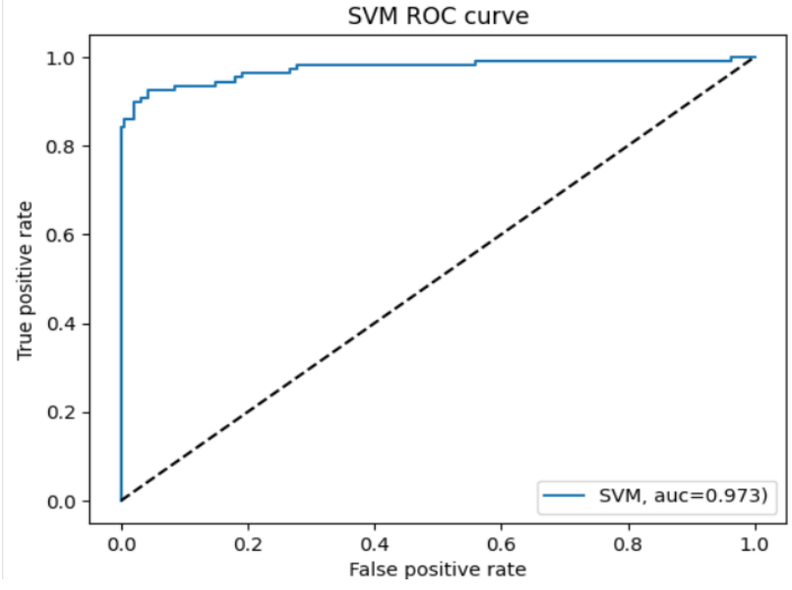
➢ **Visualizations**:

- **Confusion Matrices** helped identify the trade-offs between false positives and false negatives.
- **ROC Curves** highlighted the discrimination capability of each model.
- **Decision Boundaries** (especially for KNN and SVM) provided insights into how well models separated classes in feature space.









4. Conclusion

In this project, we implemented and evaluated four supervised machine learning models — K-Nearest Neighbors (KNN), Logistic Regression, Support Vector Machine (SVM) with RBF kernel, and Decision Tree — for detecting credit card fraud using a publicly available Kaggle dataset. The dataset was preprocessed by removing the Time column, normalizing the Amount feature, and applying z-score standardization to all features. A stratified 80/20 train-test split was performed to preserve class distribution during model training and evaluation.

Model performance was assessed using accuracy, precision, recall, F1-score, and ROC-AUC. KNN was tuned using an error rate analysis to select the optimal value of k. The Decision Tree model used the entropy criterion and was visualized to aid interpretability. SVM and Decision Tree results were further evaluated through confusion matrices and ROC curves.

- **Challenges Encountered:**
  - The dataset exhibited significant class imbalance, which was addressed during evaluation using class-wise metrics.
  - Training SVM and KNN models required more computational time compared to Logistic Regression and Decision Tree.
- **Key Learnings:**
  - Standardization of features significantly improved the performance of KNN, Logistic Regression, and SVM.
  - Visualizing the Decision Tree helped in understanding the model's decision-making logic.
  - Evaluation across multiple models highlighted the trade-offs between accuracy, interpretability, and computational efficiency.

5. References:

[1] Zareapoor, M., et al. "Analysis on credit card fraud detection techniques: Based on certain design criteria." International Journal of Computer Applications, vol. 52, no. 3, 2012.

[2] Alenzi, N. O., & Aljehane, H. Z. "Fraud detection in credit cards using logistic regression." International Journal of Advanced Computer Science and Applications, vol. 11, no. 12, 2020.

[3] Sahin, Y., & Duman, E. "Detecting credit card fraud by decision trees and support vector machines." International Journal of Computer Science and Network Security, vol. 11, no. 12, 2011.---9

[4] Maniraj, S. P., et al. "Credit card fraud detection using machine learning and data science." International Journal of Engineering Research & Technology, vol. 8, no. 9, 2019.

[5] Maes, S., et al. "Credit card fraud detection using Bayesian and neural networks." International Journal of Engineering Research & Technology, vol. 1, no. 3, 2002.

[6] Jain, Y., et al. "A comparative analysis of various credit card fraud detection techniques." International Journal of Computer Applications, vol. 178, no. 1, 2019.

[7] Dighe, D., et al. "Detection of credit card fraud transactions using machine learning algorithms and neural networks." IEEE International Conference on Computing, Communication, and Automation, 2018.

[8] Dheepa, V., & Dhanapal, R. "Behavior-based credit card fraud detection using support vector machines." International Journal of Computer Science Issues, vol. 9, no. 3, 2012.

[9] Malini, N., & Pushpa, M. "Analysis of credit card fraud identification techniques based on KNN and outlier detection." IEEE International Conference on Advanced Computing and Communication Systems, 2017.

Drive Link: All the documents (code, report, ppt) have been uploaded to the drive:
https://drive.google.com/drive/folders/10yinkm8xosTdw5_lkjCO78gFx1rFMwfr?usp=sharing