Detecting Fraudulent Transactions in Online Payments using Supervised Learning

Valentina Bertei
Francesco Tarchi

Data Mining and Machine Learning Project
A.Y. 2024/2025

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

Problem Description

Problem Overview

Online transactions are increasingly exposed to fraudulent activities, posing risks to both consumers and businesses.
Credit card fraud detection is critical in preventing unauthorized access and minimizing financial losses.

Relevance of DMML Techniques

The challenge is inherently a **binary classification** task (*fraudulent* vs. *legitimate* transactions). This calls for robust **supervised** machine learning methods, advanced feature engineering, and imbalance handling strategies.

Proposed Approach

- Compare multiple classification algorithms to identify the most effective model for fraud detection.
- Candidate models:
  - K-NN
  - Naïve Bayes
  - Decision Tree
  - Random Forest
  - AdaBoost
  - XGBoost
- A set of evaluation metrics will be used to compare the classifiers and determine the best-performing model.

Dataset Description

Dataset Source

- Publicly available on Kaggle: IEEE-CIS Fraud Detection
- Originally provided by Vesta Corporation, a real-world e-commerce platform

Collection Details

The dataset contains historical online transaction data enriched by behavioral signals and device information.

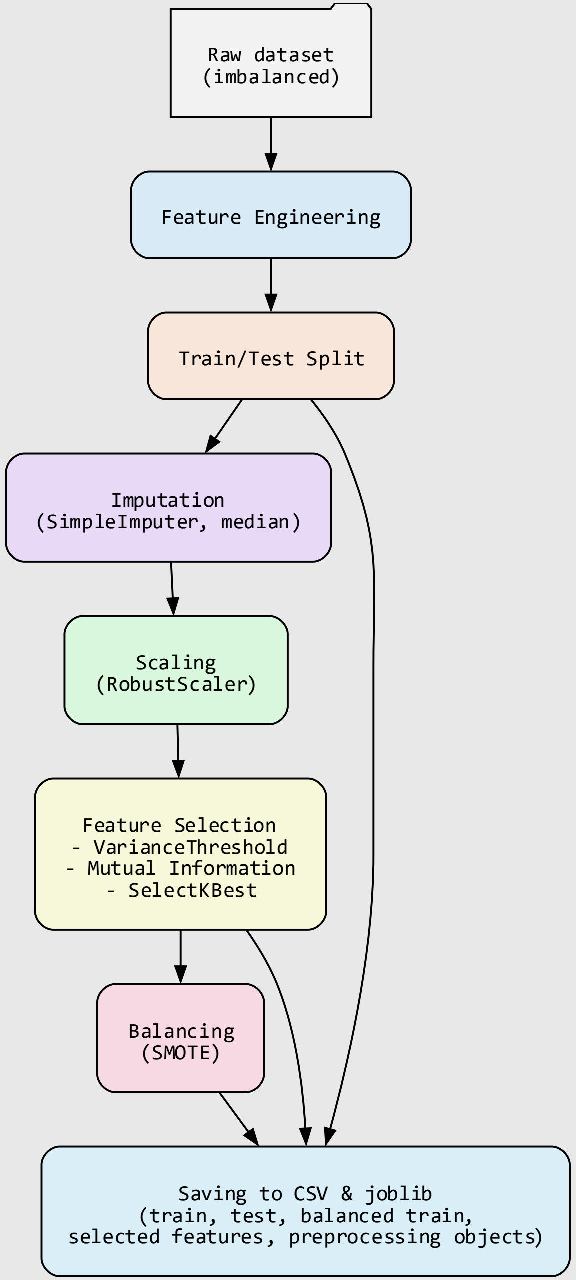
Dataset Properties:

- **Size**: 590,540 transactions (1.08 GB)
- **Fraudulent samples**: 20,663 (approx. 3.5%)
- **Columns**: 432 total columns including transaction details, card info, identity data, and 339 engineered Vesta features
- **Label**: `isFraud` (1 = fraudulent, 0 = normal)
- **Input/Output Format**:
    - Input: Multivariate features including TransactionAmt, card1–card6, addr1, D1, C1, M1, V1–V339, etc.
    - Output: Binary class label `isFraud` ∈ {0, 1}

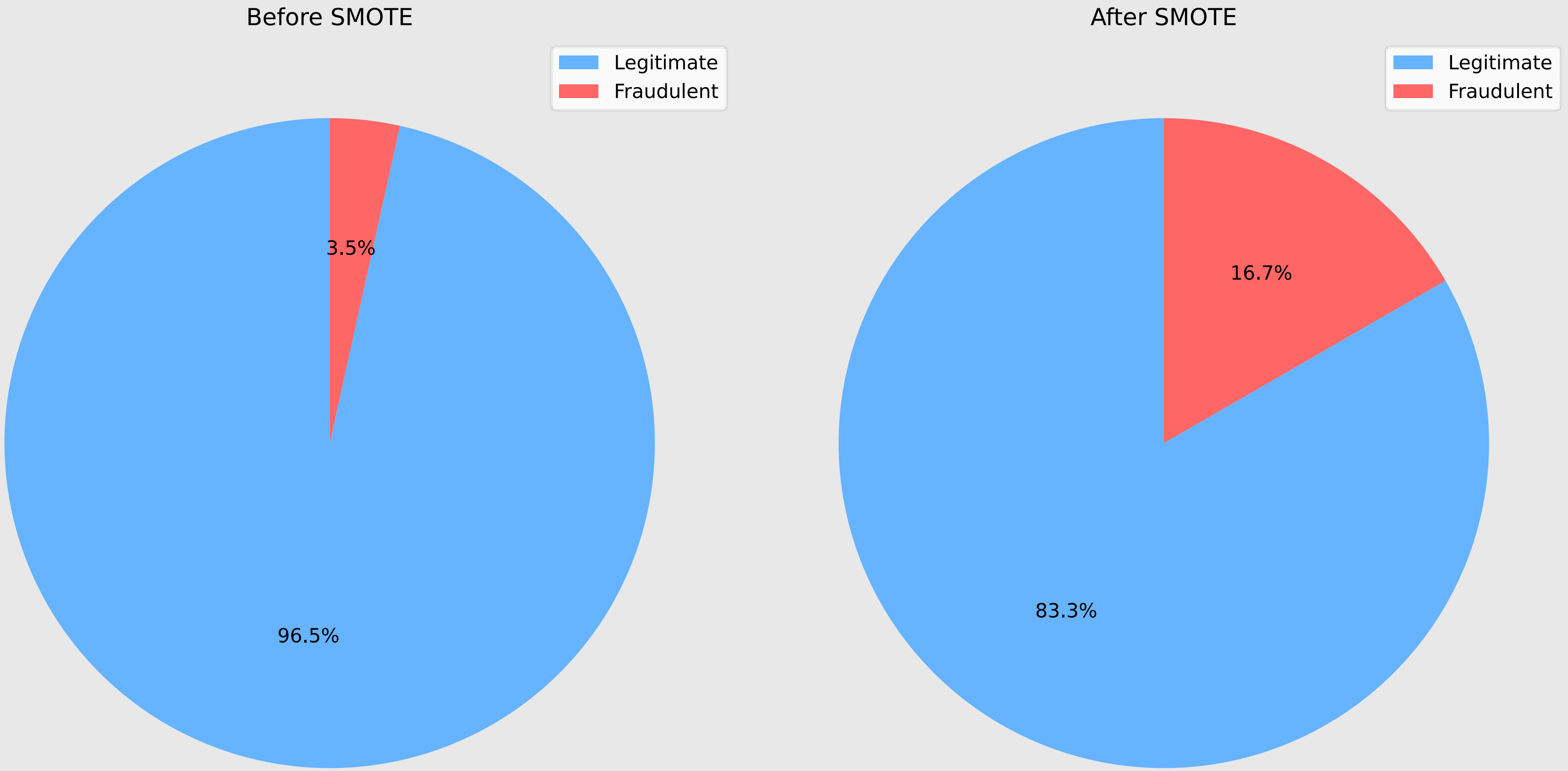
Preprocessing

Raw dataset cleaning

- **Feature engineering**: temporal features from `TransactionDT` → day, hour, weekday + cyclic encoding (sine/cosine)
- **Train/test split**: stratified sampling to preserve label distribution
- **Missing values**: median imputation
- **Scaling**: `RobustScaler` to reduce outlier impact
- **Feature selection**:
    - ✓ Variance Threshold → remove low-variability features
    - ✓ `SelectKBest` with Mutual Information → retain most informative features
- Class imbalance handling: **SMOTE** applied to training set only
    - ✓ Sampling strategy: 0.2 → fraud rate ↑ from 3.5% → 16.7%



Preprocessing



Comparison of the class distribution in the training set before (left) and after (right) the application of **SMOTE**.

Training and Testing

Classifiers evaluated

- KNN (K-Nearest Neighbors)
- NB (Naive Bayes)
- DT (DecisionTree)
- RF (RandomForest)
- ADA (AdaBoost)
- XGB (XGBoost)

Hyperparameter tuning

- **Grid Search** (`GridSearchCV`) with 5-fold CV
- Scoring metric: *f1* → balances *precision* & *recall* on imbalanced data
- Goal: find best hyperparameters for each classifier

Training & Testing

- Train on SMOTE-rebalanced train set
- 10-fold CV during training → robust validation metrics
- Test on imbalanced test set → evaluate predictions

Evaluation Metrics

Metrics used

- *Confusion Matrix* → base for other metrics (TN, FP, FN, TP)
- *Precision* & *Recall* → focus
- *f1 Score* → balances precision & recall: used in Grid Search
- *Accuracy* & *Balanced Accuracy* → quick overview, the 2^{nd} robust to imbalanced datasets
- Weighted versions → balancing the metrics between the 2 classes
- *ROC AUC* & *PR AUC* → compare overall classifier performance

Acceptable Level of Performance (ALP)

- Defined as $TPR \geq 0.8$ → correctly identify ≥80% of frauds
- **ALP_threshold** → decision threshold where ALP is reached
- **ALP_FPR** → FPR when ALP is reached
- Analysis via ROC curves → identify best trade-off between TPR and FPR

Model explanability

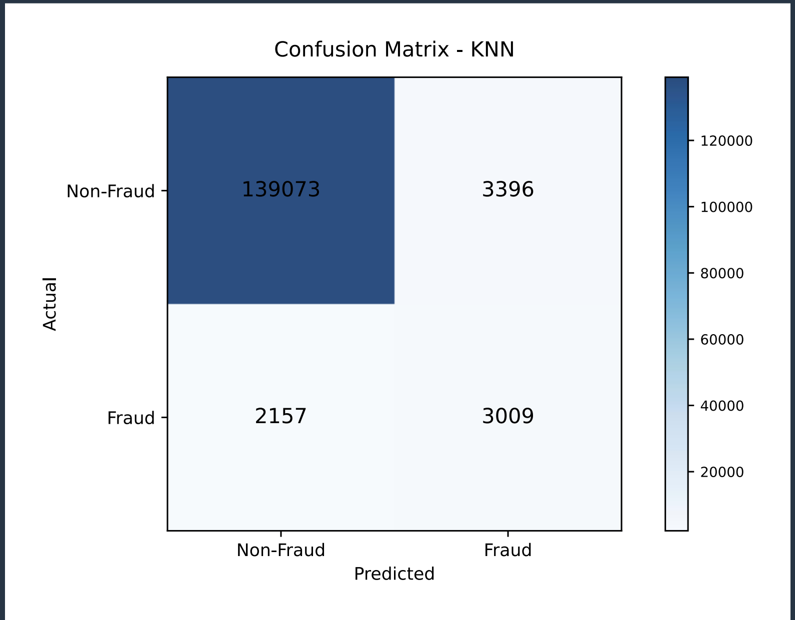
Goal

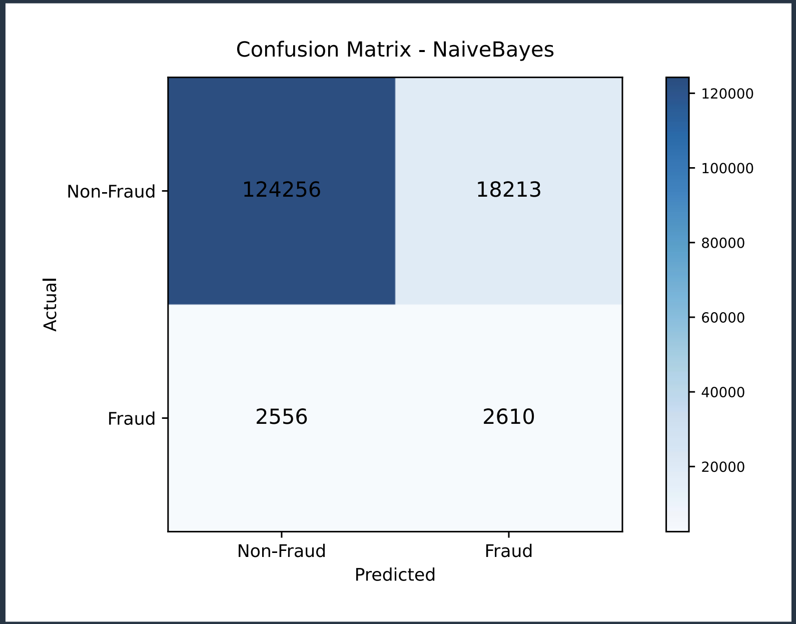
Understand predictions & identify influential features

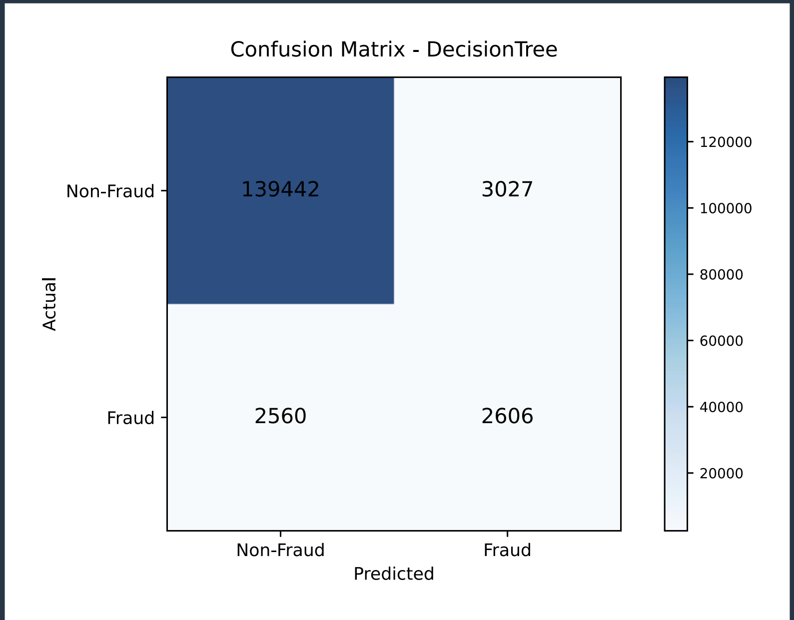
Techniques applied

- **Feature Importances**
  - For tree-based models (DT, RF, XGB)
  - Horizontal bar plots → top contributing features
- **SHAP Values**
  - Quantify feature contribution for individual predictions
  - Summary plots → global feature impact
  - Stratified sample of test set used
- **Permutation Feature Importance**
- **Extraction via Surrogate Models**
  - Surrogate decision trees (`depth=3`)
  - Rules saved as text files

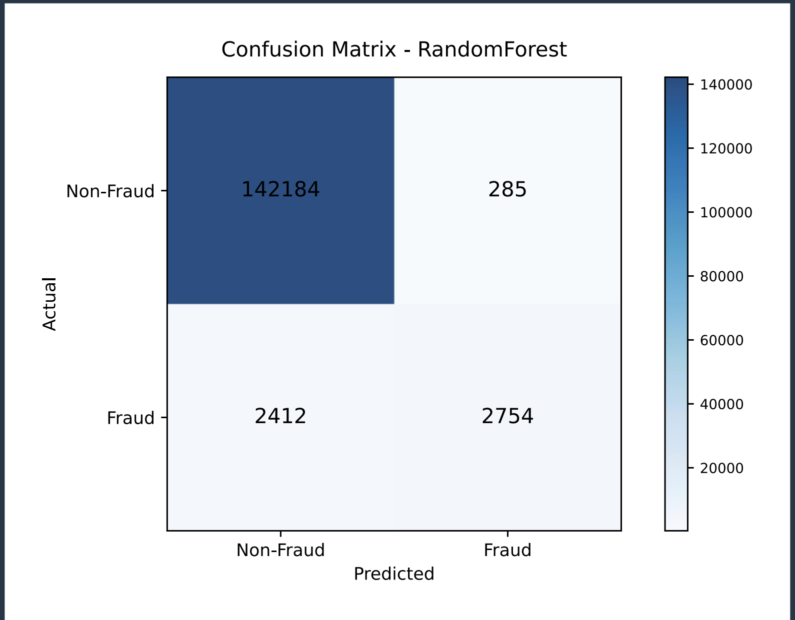
Individual results

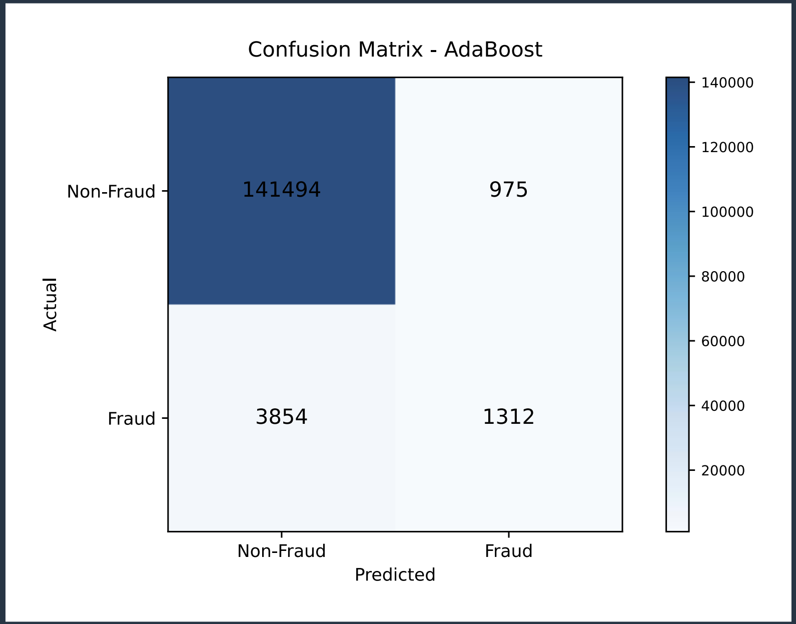


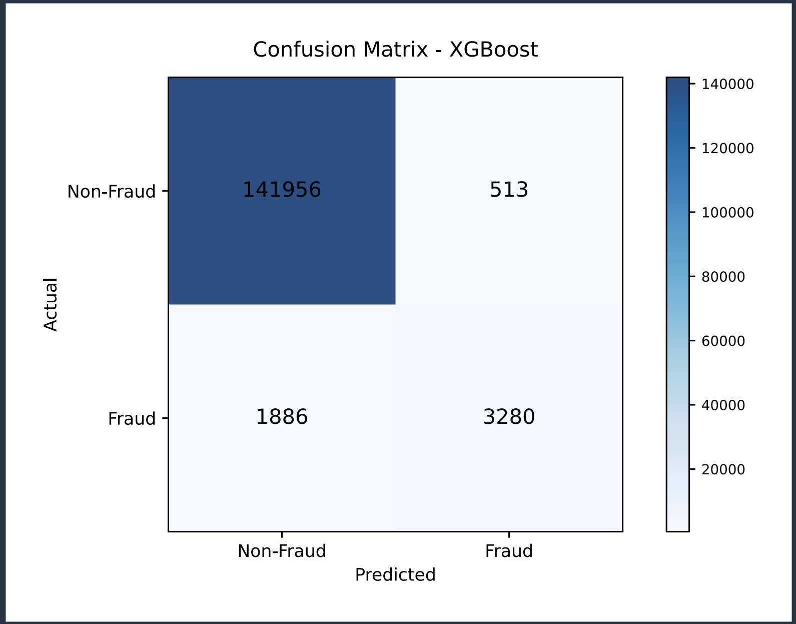




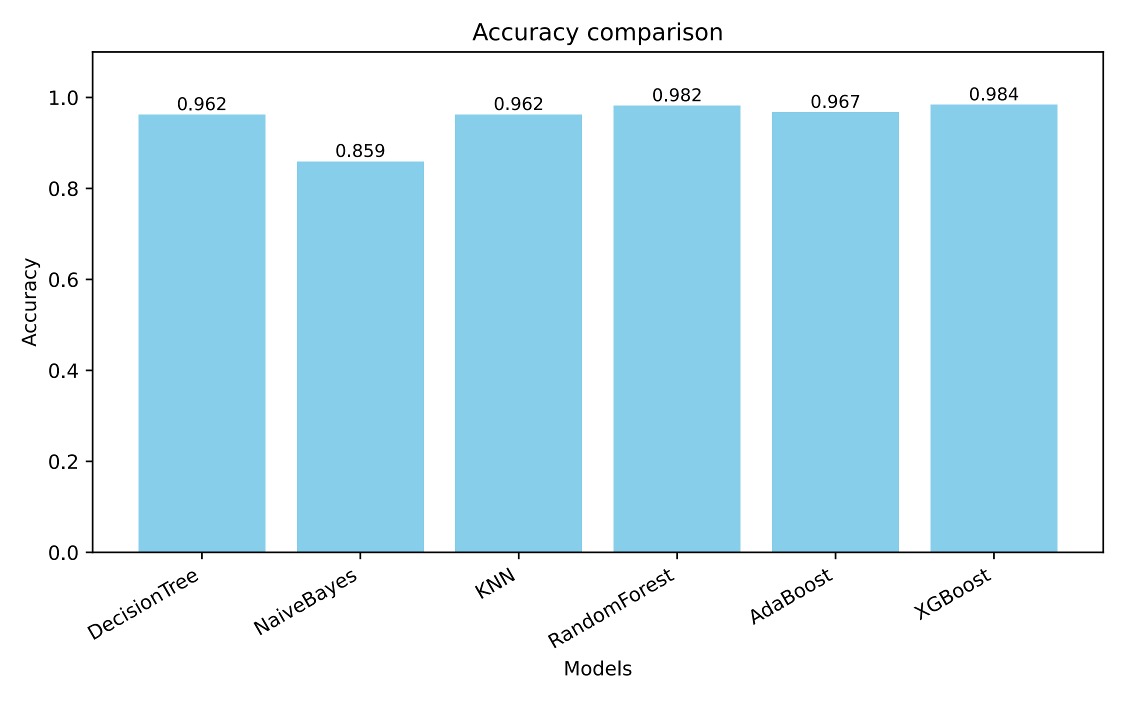
Individual results

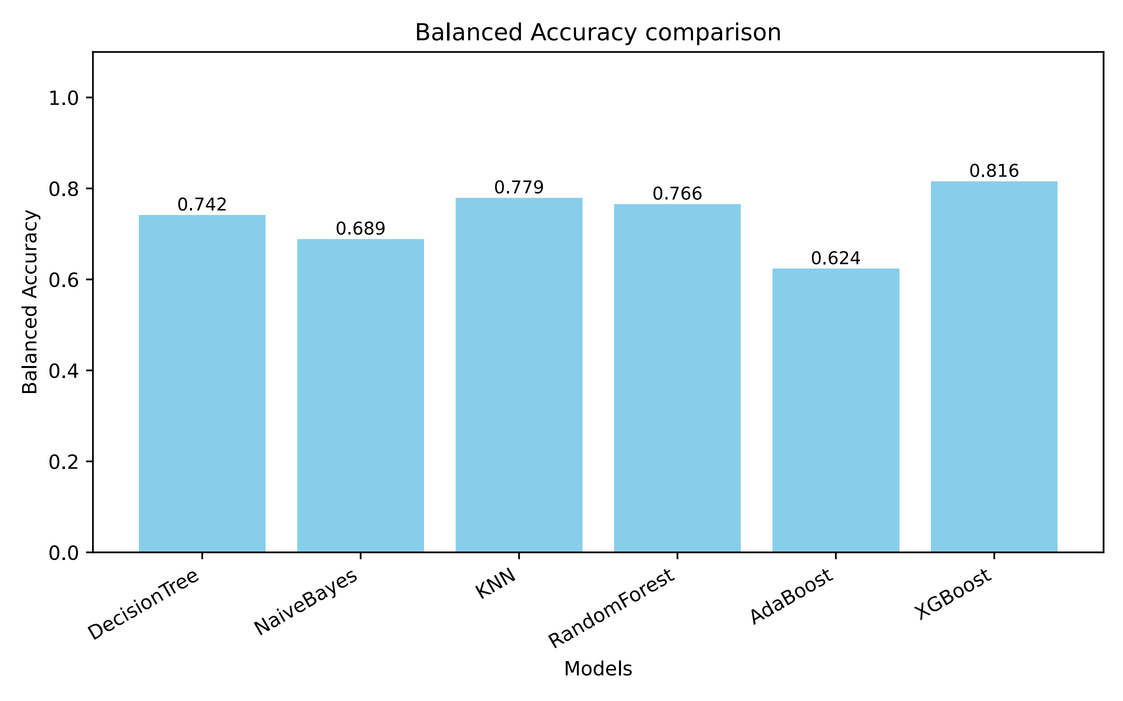




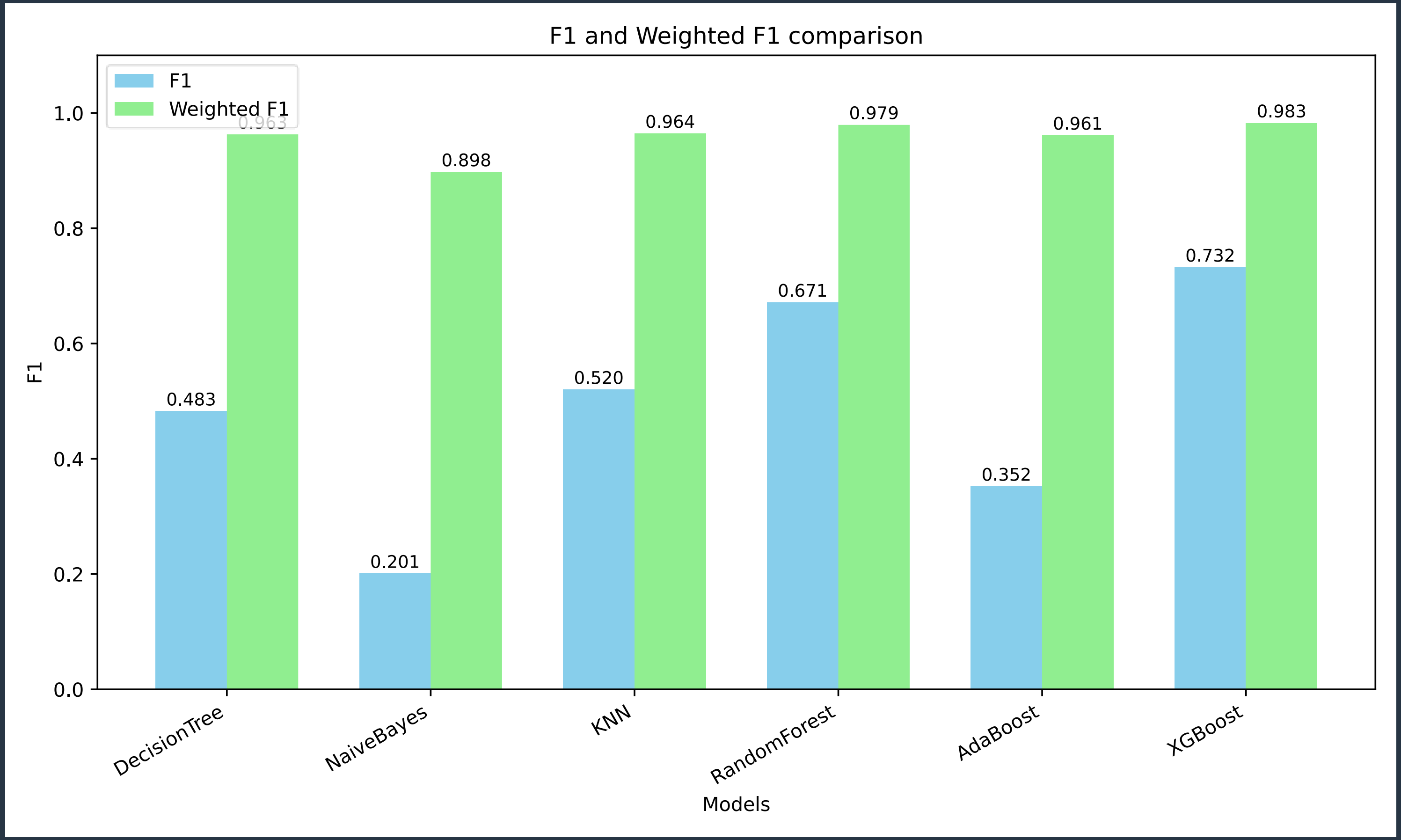


Comparison among models

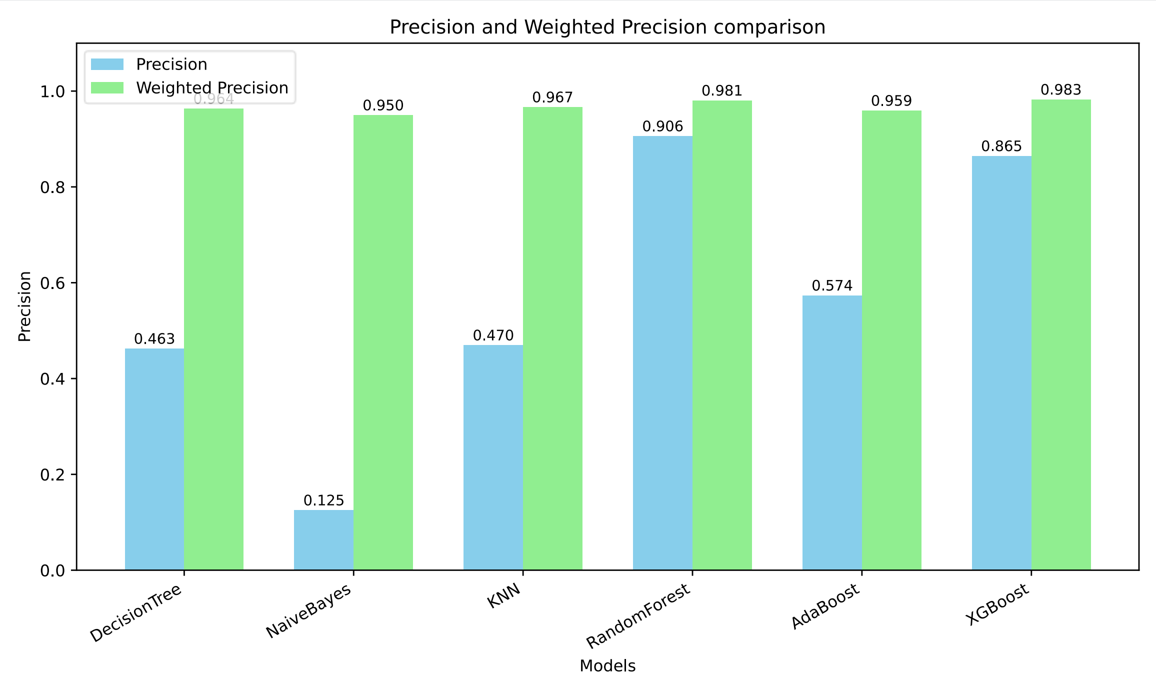


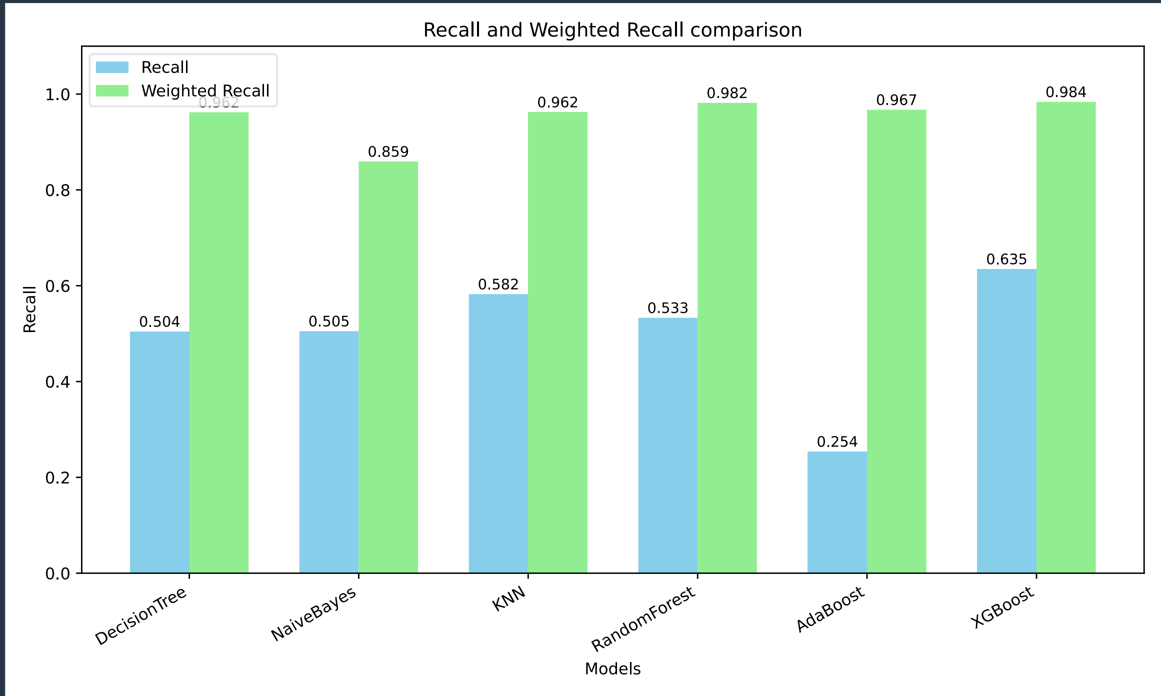


Comparison among models

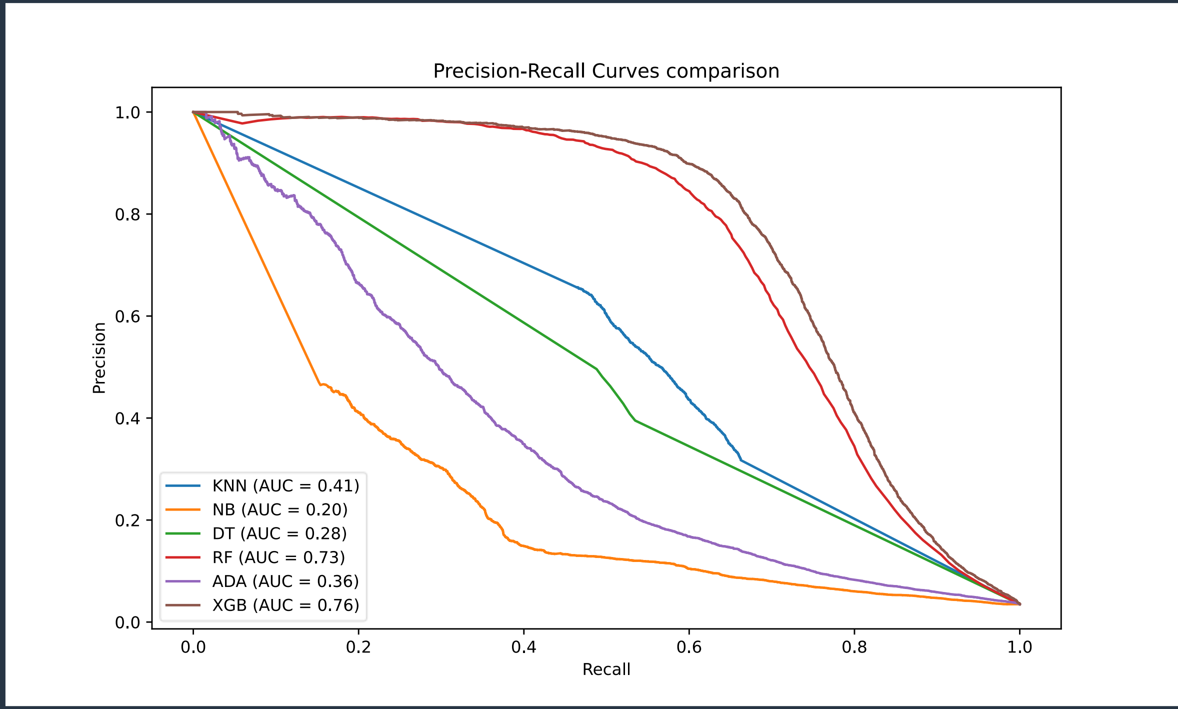


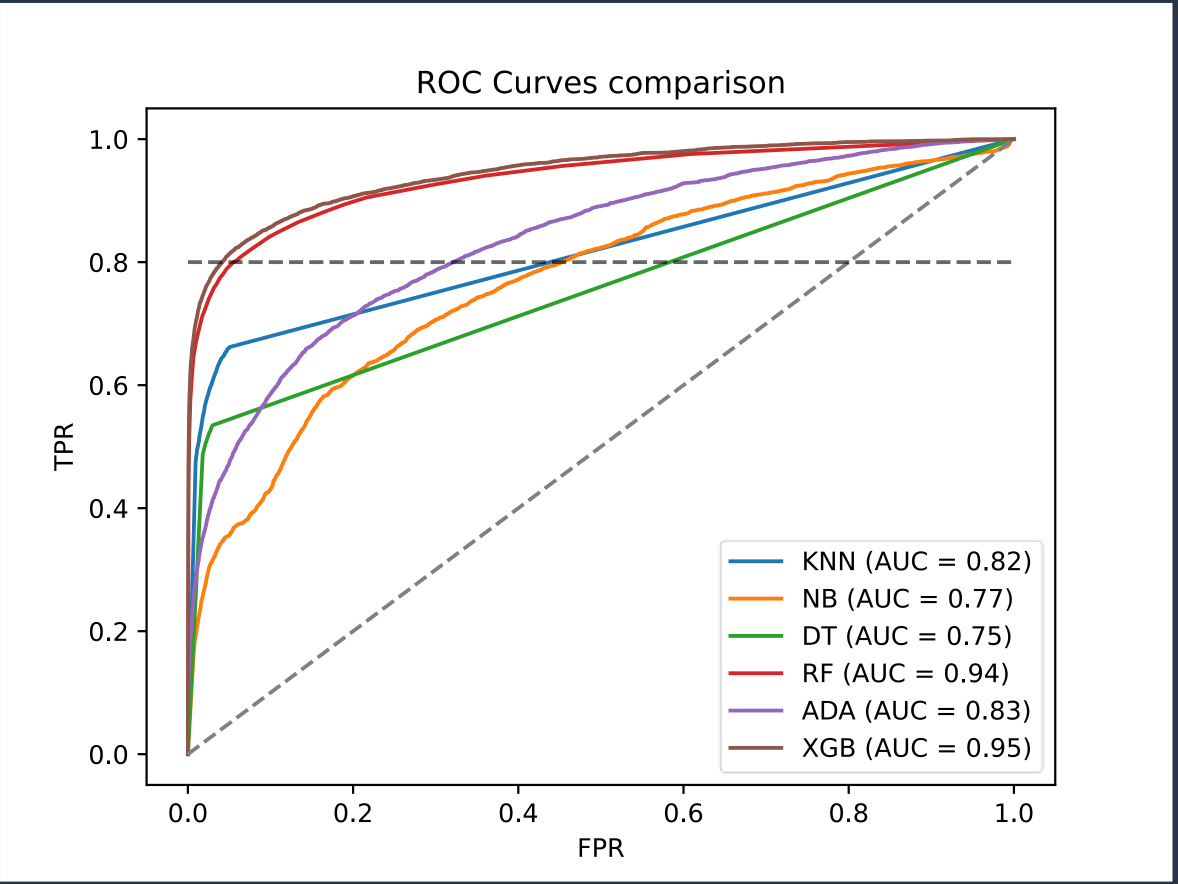
Comparison among models



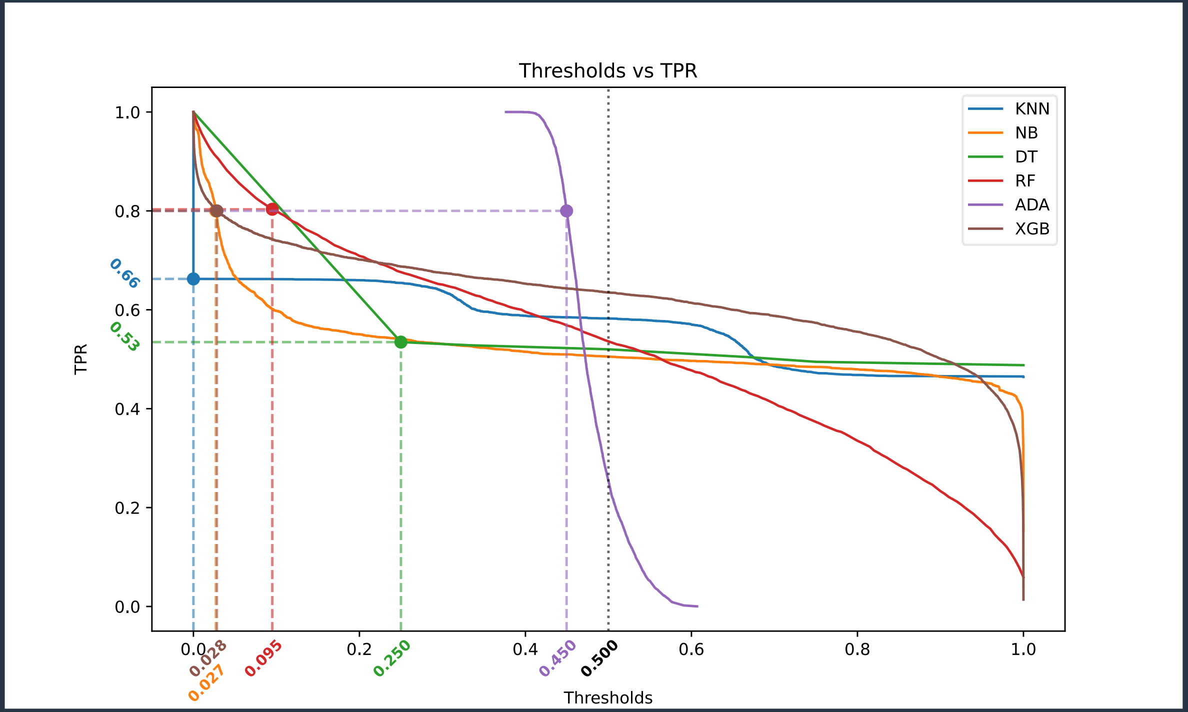


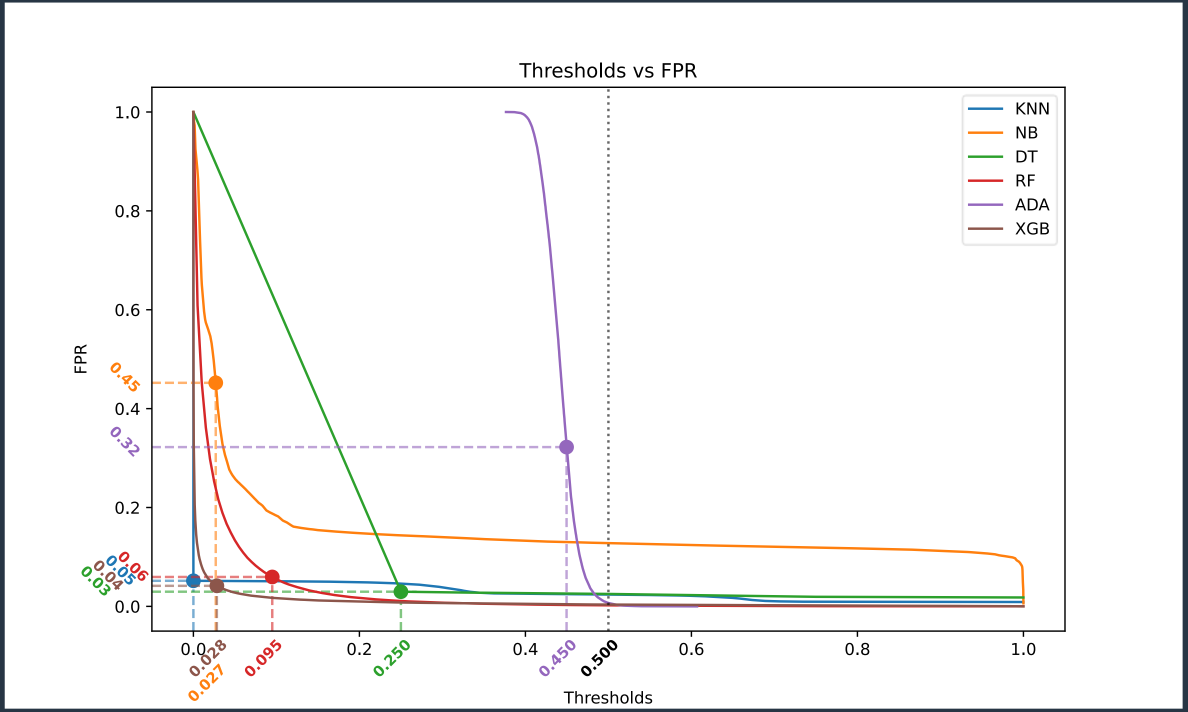
Comparison among models





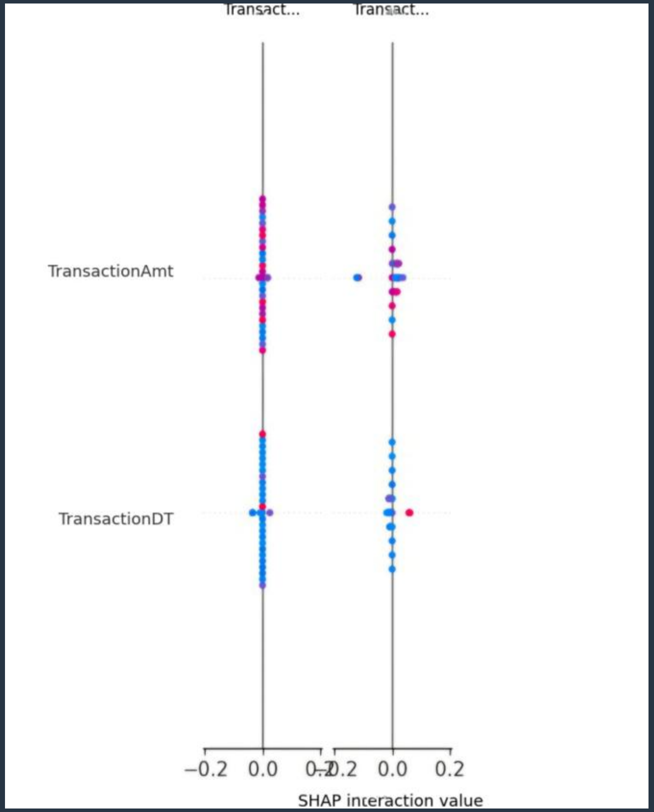
Comparison among models

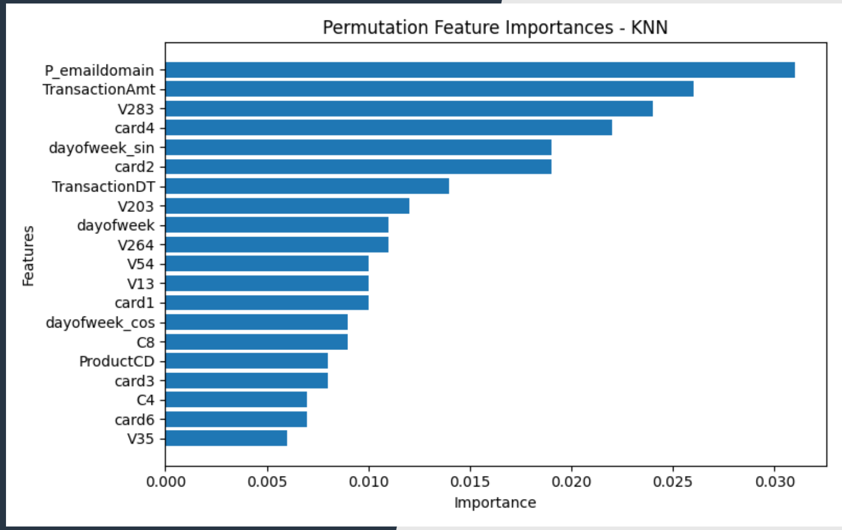


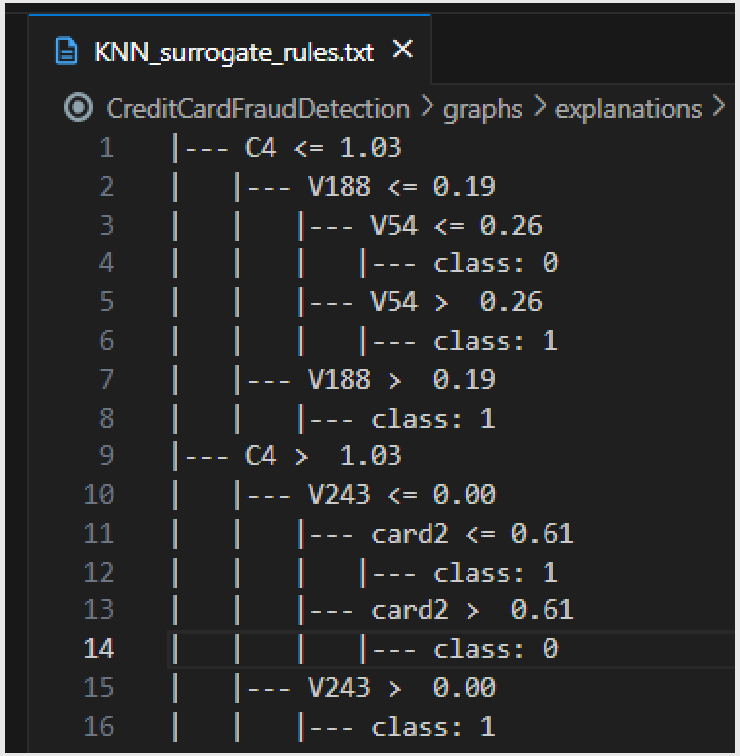


Explanations of the models

K-Nearest Neighbours





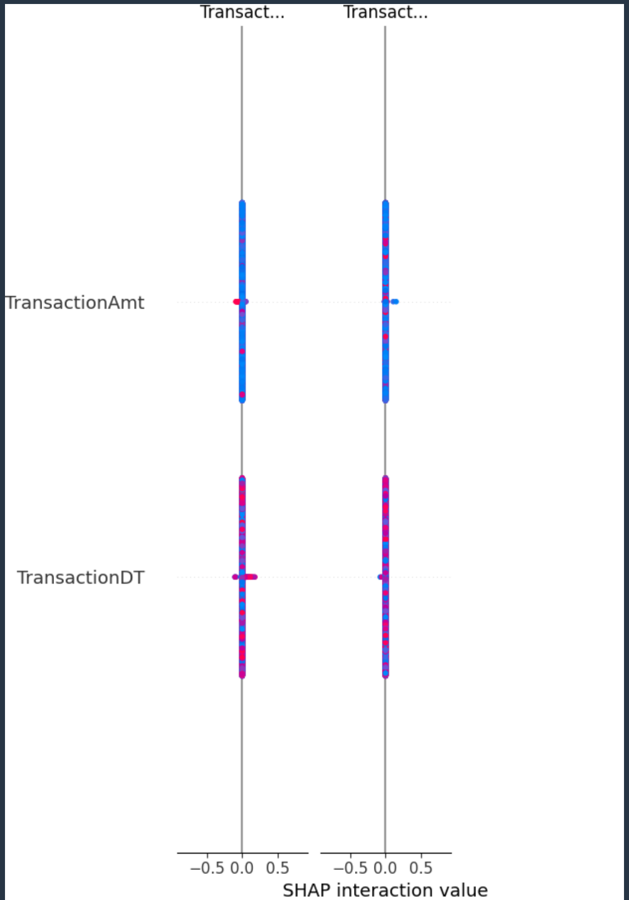

KNN_surrogate_rules.txt

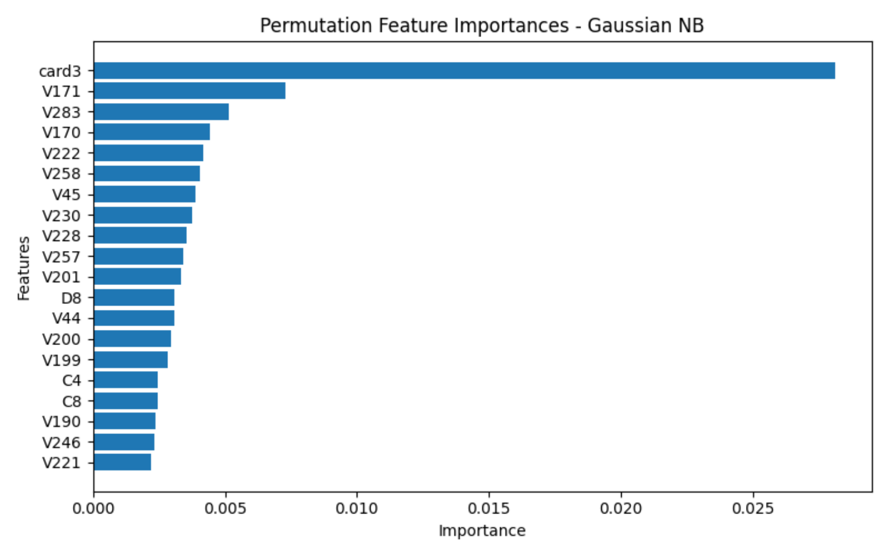
CreditCardFraudDetection > graphs > explanations >

```
|--- C4 <= 1.03
|   |--- V188 <= 0.19
|   |   |--- V54 <= 0.26
|   |   |   |--- class: 0
|   |   |--- V54 >  0.26
|   |   |   |--- class: 1
|   |--- V188 >  0.19
|   |   |--- class: 1
|--- C4 >  1.03
|   |--- V243 <= 0.00
|   |   |--- card2 <= 0.61
|   |   |   |--- class: 1
|   |   |--- card2 >  0.61
|   |   |   |--- class: 0
|   |--- V243 >  0.00
|   |   |--- class: 1
```

Explanations of the models

Naive Bayes





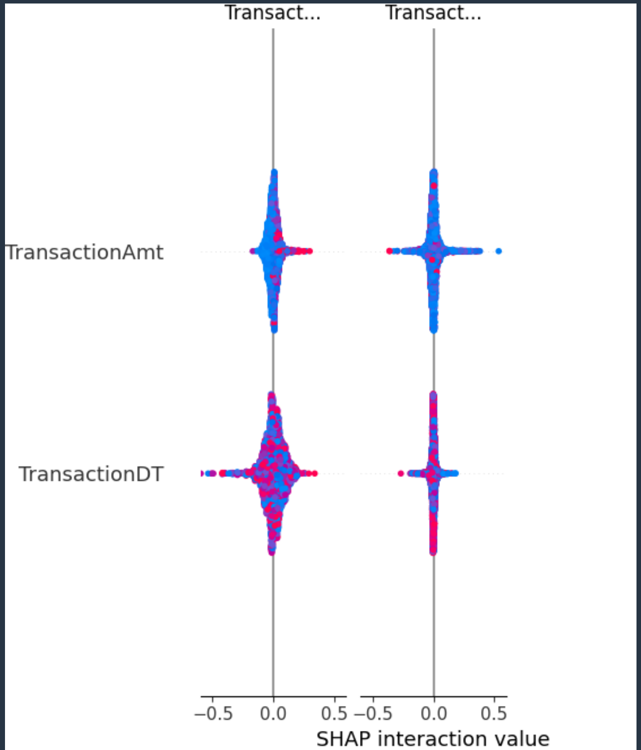
Gaussian NB_surrogate_rules.txt

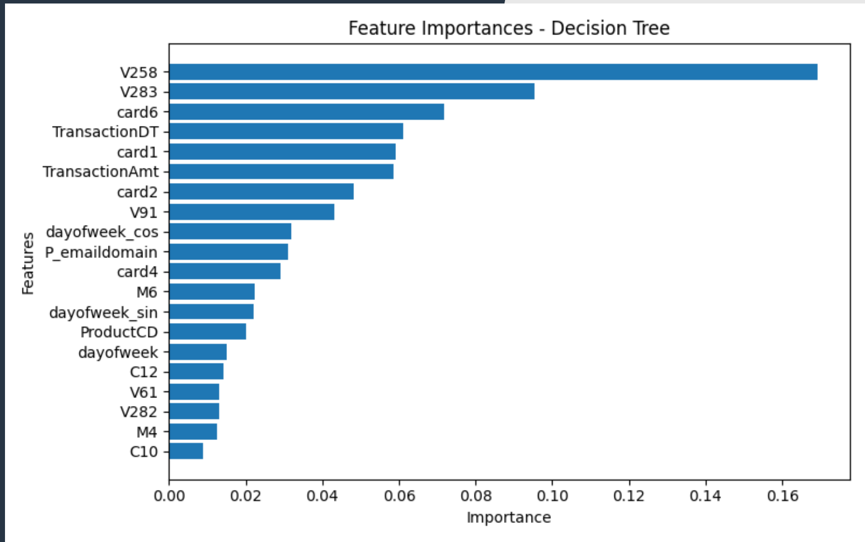
CreditCardFraudDetection > graphs > explanations >

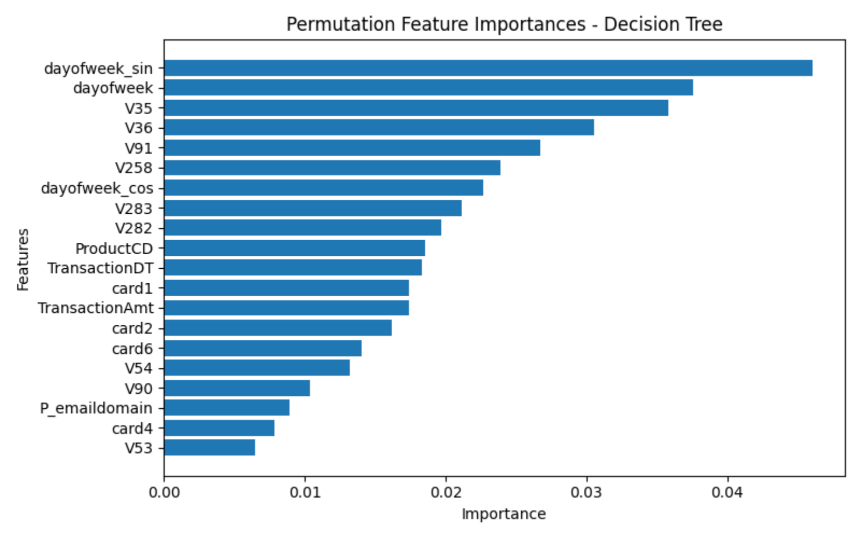
```
|--- ProductCD <= -3.01
|   |--- D8 <= 737.08
|   |   |--- V76 <= -0.13
|   |   |   |--- class: 1
|   |   |--- V76 >  -0.13
|   |   |   |--- class: 1
|   |--- D8 >  737.08
|   |   |--- V170 <= 1.50
|   |   |   |--- class: 0
|   |   |--- V170 >  1.50
|   |   |   |--- class: 1
|--- ProductCD >  -3.01
|   |--- V219 <= 0.00
|   |   |--- V283 <= 5.78
|   |   |   |--- class: 0
|   |   |--- V283 >  5.78
|   |   |   |--- class: 1
|   |--- V219 >  0.00
|   |   |--- V200 <= 0.36
|   |   |   |--- class: 0
|   |   |--- V200 >  0.36
|   |   |   |--- class: 1
```

Explanations of the models

DecisionTree







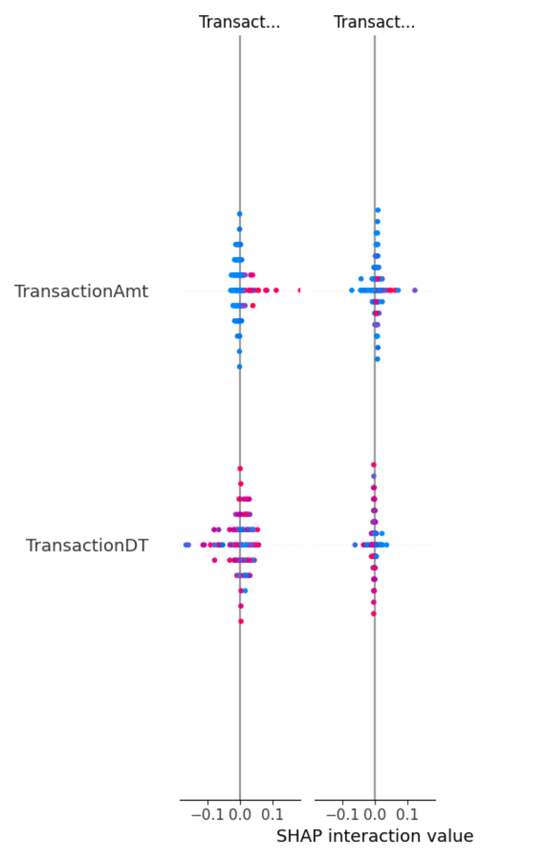
Decision Tree_surrogate_rules.txt

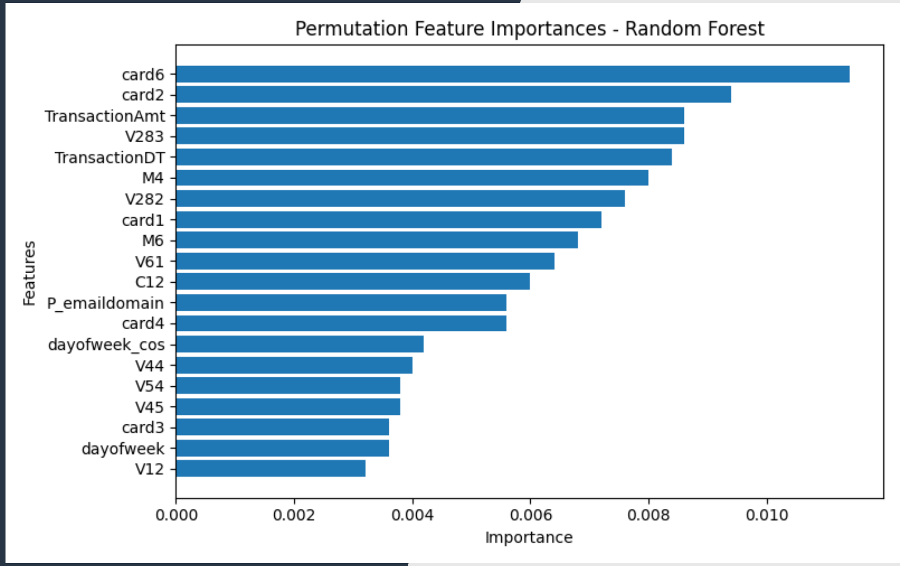
CreditCardFraudDetection > graphs > explanations >

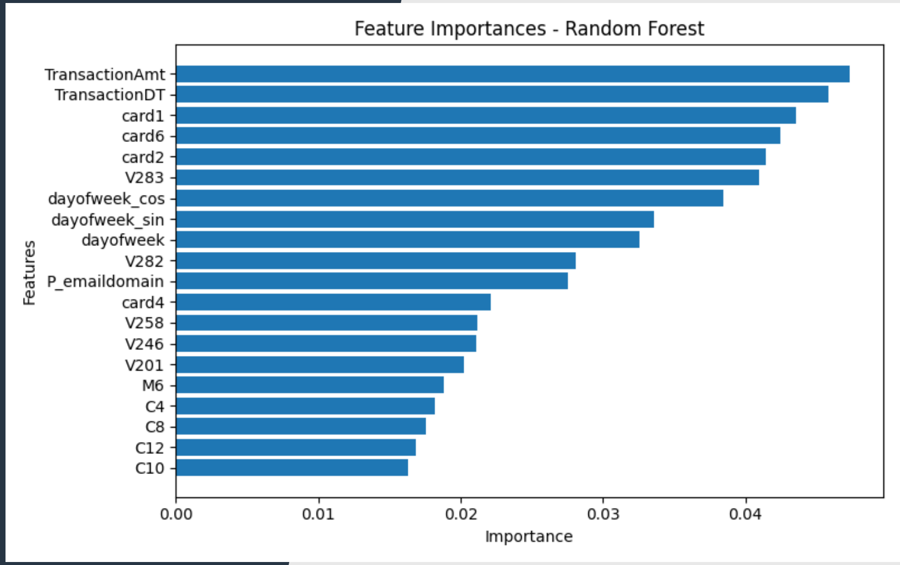
```
|--- V258 <= 0.00
|   |--- V283 <= 0.00
|   |   |--- C7 <= 0.00
|   |   |   |--- class: 0
|   |   |--- C7 >  0.00
|   |   |   |--- class: 0
|   |--- V283 >  0.00
|   |   |--- V283 <= 1.00
|   |   |   |--- class: 1
|   |   |--- V283 >  1.00
|   |   |   |--- class: 0
|--- V258 >  0.00
|   |--- V258 <= 1.00
|   |   |--- V258 <= 1.00
|   |   |   |--- class: 1
|   |   |--- V258 >  1.00
|   |   |   |--- class: 0
|   |--- V258 >  1.00
|   |   |--- V265 <= 5725.00
|   |   |   |--- class: 1
|   |   |--- V265 >  5725.00
|   |   |   |--- class: 0
```

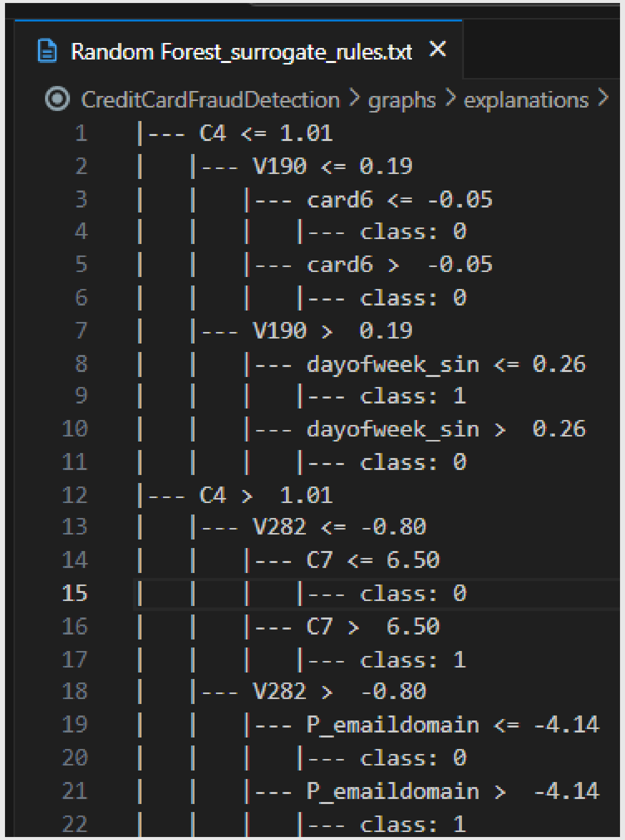
Explanations of the models

RandomForest





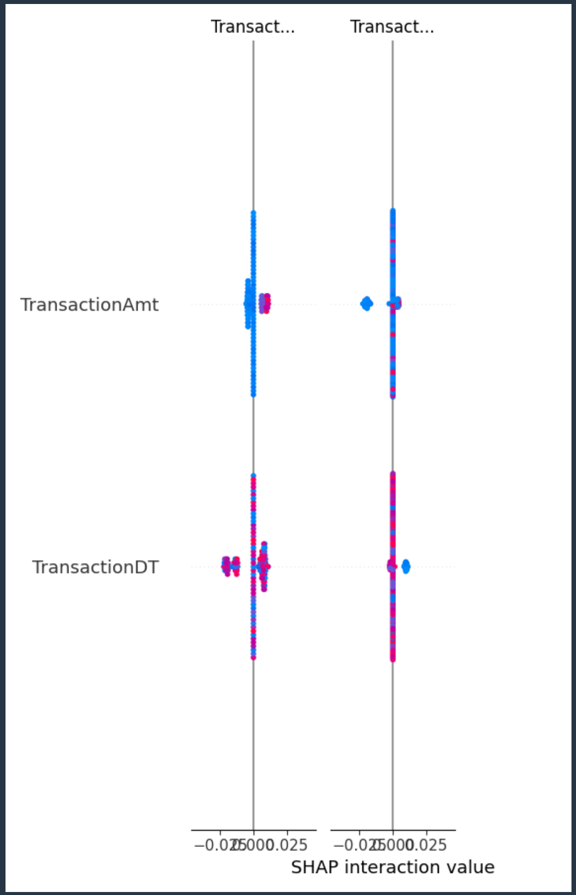


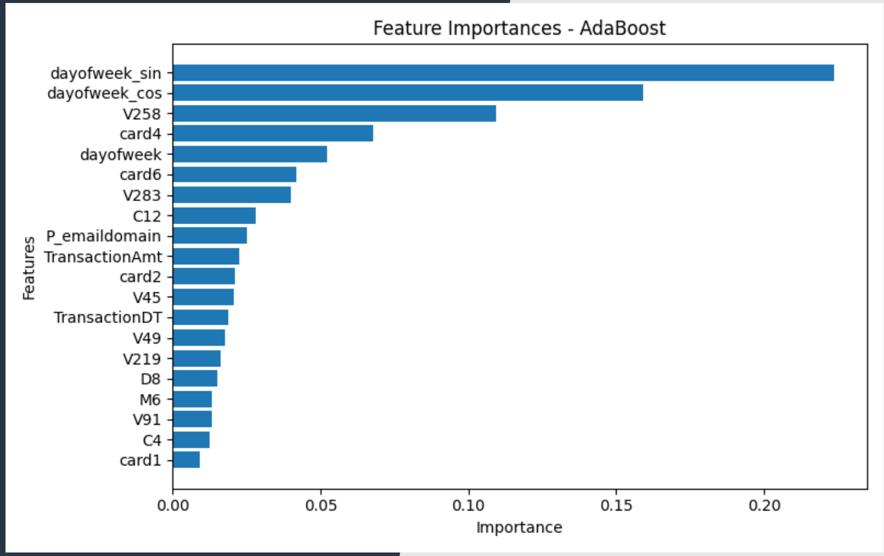


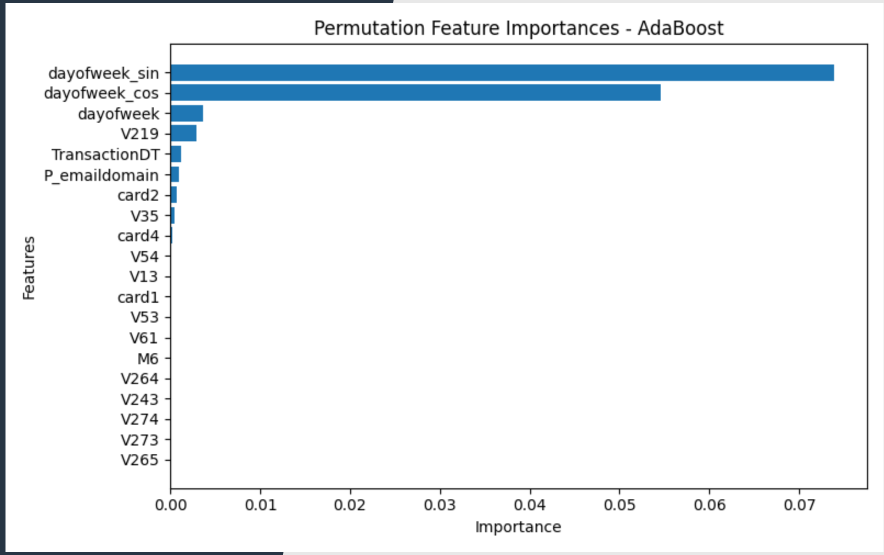

```
|--- C4 <= 1.01
|   |--- V190 <= 0.19
|   |   |--- card6 <= -0.05
|   |   |   |--- class: 0
|   |   |--- card6 >  -0.05
|   |   |   |--- class: 0
|   |--- V190 >  0.19
|   |   |--- dayofweek_sin <= 0.26
|   |   |   |--- class: 1
|   |   |--- dayofweek_sin >  0.26
|   |   |   |--- class: 0
|--- C4 >  1.01
|   |--- V282 <= -0.80
|   |   |--- C7 <= 6.50
|   |   |   |--- class: 0
|   |   |--- C7 >  6.50
|   |   |   |--- class: 1
|   |--- V282 >  -0.80
|   |   |--- P_emaildomain <= -4.14
|   |   |   |--- class: 0
|   |   |--- P_emaildomain >  -4.14
|   |   |   |--- class: 1
```

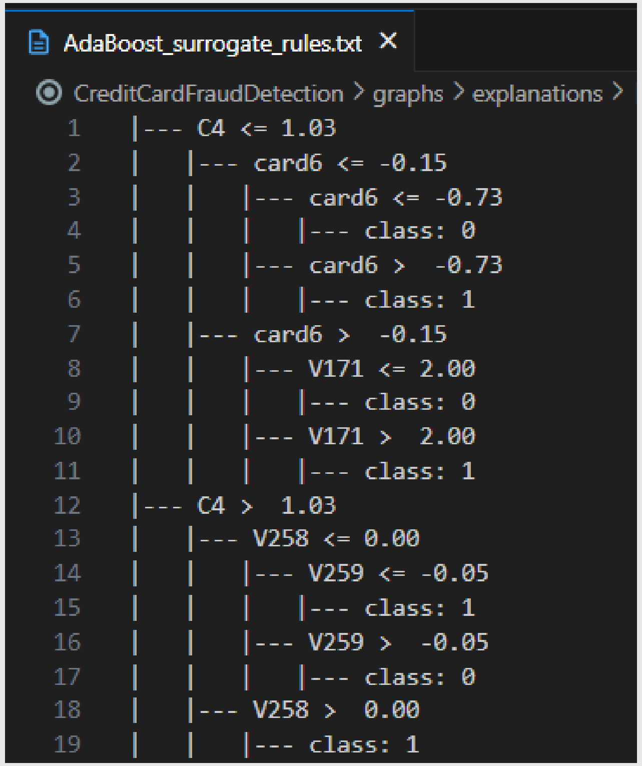
Explanations of the models

AdaBoost





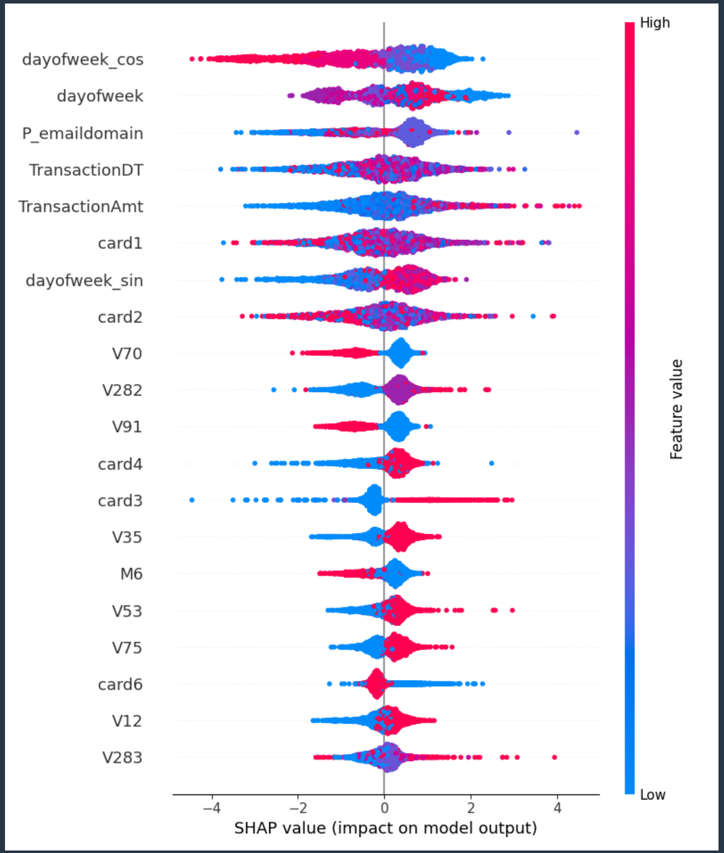


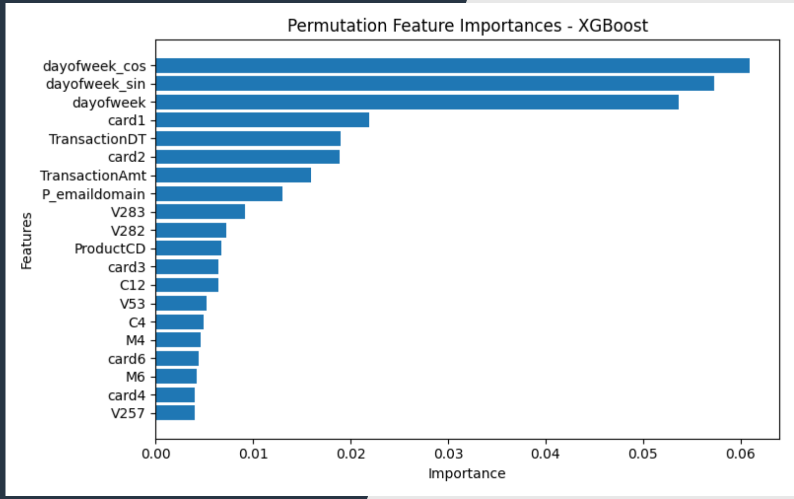


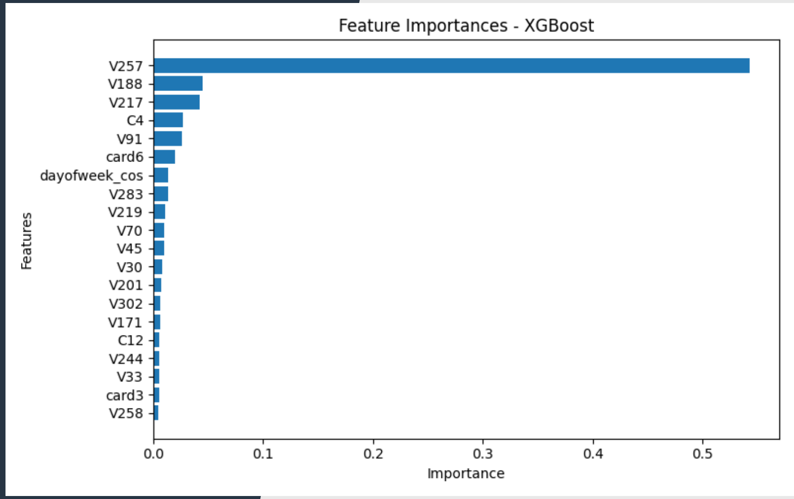

```
|--- C4 <= 1.03
|   |--- card6 <= -0.15
|   |   |--- card6 <= -0.73
|   |   |   |--- class: 0
|   |   |--- card6 >  -0.73
|   |   |   |--- class: 1
|   |--- card6 >  -0.15
|   |   |--- V171 <= 2.00
|   |   |   |--- class: 0
|   |   |--- V171 >  2.00
|   |   |   |--- class: 1
|--- C4 >  1.03
|   |--- V258 <= 0.00
|   |   |--- V259 <= -0.05
|   |   |   |--- class: 1
|   |   |--- V259 >  -0.05
|   |   |   |--- class: 0
|   |--- V258 >  0.00
|   |   |--- class: 1
```

Explanations of the models

XGBoost







XGBoost_surrogate_rules.txt

CreditCardFraudDetection > graphs > explanations >

```
|--- V258 <= 0.00
|   |--- V283 <= 0.00
|   |   |--- C7 <= 0.00
|   |   |   |--- class: 0
|   |   |--- C7 >  0.00
|   |   |   |--- class: 0
|   |--- V283 >  0.00
|   |   |--- V283 <= 1.00
|   |   |   |--- class: 1
|   |   |--- V283 >  1.00
|   |   |   |--- class: 0
|--- V258 >  0.00
|   |--- V258 <= 1.00
|   |   |--- V258 <= 1.00
|   |   |   |--- class: 1
|   |   |--- V258 >  1.00
|   |   |   |--- class: 0
|   |--- V258 >  1.00
|   |   |--- V265 <= 5725.00
|   |   |   |--- class: 1
|   |   |--- V265 >  5725.00
|   |   |   |--- class: 0
```

Real-world Application

Scenario

- Online payment systems → high risk of fraud
- Goal: real-time detection for banks, e-commerce, payment processors

Prototype

- Web interface built with *Streamlit*
- Users manually input transaction details

Pipeline

1. **Pre-processing**
   - Same as training: missing values, scaling, temporal features, feature selection
2. **Classification**
   - 6 classifiers: Decision Tree, Random Forest, Naive Bayes, KNN, AdaBoost, XGBoost
   - Output: Legitimate / Fraudulent
3. **Ensemble decision**
   - Majority voting for final prediction
4. **Explainability (XAI)**
   - Tree-based: SHAP values
   - AdaBoost: Kernel SHAP
   - Naive Bayes: posterior probabilities
   - KNN: nearest neighbors' examples

Real-world Application

Conclusions

Key Findings

- **Best performance**: Ensemble models → XGBoost > Random Forest (*precision*, *recall*, *f1-score*, *ROC AUC*)
- **Acceptable Level of Performance (ALP)**: some models can correctly identify ≥80% frauds with limited false positives
- **Threshold analysis**:
  - KNN & Decision Tree → cannot match top performers
  - KNN very slow
  - AdaBoost → “conservative” (small threshold deviation), but high false positive rate
  - Worst model: Naive Bayes

Overall takeaway

XGBoost is the best choice for fraud detection applications: high *TPR* with low *FPR*.

References

Related Work

- Cho Do Xuan, Dang Ngoc Phong, Nguyen Duy Phuong. *A new approach for detecting credit card fraud transaction*, International Journal of Nonlinear Analysis and Applications, Vol. 14 (2023), pp. 133–146.
  Available at:
  https://ijnaa.semnan.ac.ir/article_7623_b95b41b8707a1ba645b2ad938f3cd76f.pdf

Bibliography

- Kaggle Dataset: IEEE-CIS Fraud Detection
  Available at: https://www.kaggle.com/datasets/phambacong/ieee-cis-fraud-detection
- T. Chen and C. Guestrin, *XGBoost: A scalable tree boosting system* (2016)
  Available at: https://arxiv.org/abs/1603.02754