Credit Card Fraud Detection Using Machine Learning

Shogan Tanya

DS20 Class, Bar Ilan University

Nov 2025

Project Overview:

This document outlines the protocol for an advanced machine learning project focused on credit card fraud detection, utilizing a dataset sourced from Kaggle. This project, undertaken as part of a data science course at `High-tech and Cyber School` of `Bar Ilan University` encompasses the entire data science lifecycle, from data preparation and exploratory data analysis (EDA) to model selection, fine-tuning, and evaluation. Each stage is briefly summarized, providing a roadmap for the project's execution.

Introduction to the Fraud Detection Project

The foundation of this project addresses the critical financial threat posed by **fraudulent transactions** : unauthorized or deceptive financial activities like using stolen credit card details or identity theft, intended to gain money or goods illegally. These fraudulent activities are characterized by being **outliers** compared to legitimate transactions, often exhibiting unusual patterns such as high amounts, odd locations, or rapid frequency.

Fraud detection is the essential process of identifying and preventing such transactions using data analysis and machine learning. This involves analyzing transactional data for anomalies and behaviors indicative of fraud, often employing techniques like advanced classification models (e.g., RandomForestClassifier), sophisticated resampling methods for imbalanced data (e.g., SMOTETomek), and real-time monitoring to flag or block suspicious activities. The ultimate goal is to minimize financial losses while ensuring legitimate transactions proceed smoothly.

Project Goal

The project's central aim is to determine the **best machine learning model** to accurately identify fraudulent transactions within a **highly imbalanced dataset** (where fraud accounts for only **0.56%** of all transactions).
This effort is focused on maximizing both **recall** (to catch most fraud cases) and **precision** (to minimize false alarms) for the fraud class.

Project Main Objective

Maximize recall and precision for the fraud class to improve detection reliability.
.

Data Preparation

1.1 Data Source Overview

The dataset used for this project was sourced from the **Kaggle repository**, titled *"Credit Card Fraud Mega Dataset"* by Karthik Gangula (2023).

It consists of two primary raw data files:

- **credit_card_fraud.csv** – the main transactional dataset containing approximately ~ 34 million records including transaction IDs, timestamps, merchant categories, geolocations, costumer identifiers and fraud indicators.

- **customers.csv** – a supplementary table with around ~20K customer records providing customer identifiers and additional metadata.

1.2 Data Loading and Optimization

Both files were loaded into Pandas Data Frames.
The initial CSV load required **6–12 minutes** due to the dataset size.
To improve efficiency, both tables were serialized into **Pickle (.pkl) format**, allowing near-instant reloads in later experiments narrowing the loading time to **~ 2 minutes total**.

Data Inspection and Pre-Merge Checks
Before any further manipulation, several integrity checks were performed:

- **Shape and Structure Review**

  transactions_df.shape → (~34 million rows × 26 columns)
  customers_df.shape → (~20000 rows × 16 columns)
  The schema, data types, and index columns were verified.

- **Pre-Merge Checks:**

  A structural and content-based comparison was conducted between **credit_card_fraud.csv** and **customers.csv** to evaluate whether integration was necessary.

  Column Name Match:
  Checked for shared columns between both datasets and confirmed that they have consistent naming conventions and capitalization.

- **Data Type Match:**
  Verified that the datatypes of common column fields were identical.

- **<u>Results:</u>**

  - There are **16 common features**, all with matching names and matching data types.
  - **No unique columns** found in **customers.csv,** hence merging the two tables appeared unnecessary.

```
============================================================
🔹 Common Columns with Data Types
  Column transactions dtype customers dtype
acct_num              int64           int64
  cc_num              int64           int64
    city             object          object
city_pop              int64           int64
     dob             object          object
   first             object          object
  gender             object          object
     job             object          object
    last             object          object
     lat            float64         float64
    long            float64         float64
 profile             object          object
     ssn             object          object
   state             object          object
  street             object          object
     zip              int64           int64

============================================================
🔹 Columns only in transactions:
['amt', 'category', 'is_fraud', 'merch_lat', 'merch_long', 'merchant', 'trans_date', 'trans_num', 'trans_time', 'unix_time']

============================================================
🔹 Columns only in customers:
No unique columns found.
============================================================
```

- **Value Overlap (Key Comparison):**

  Conducted a cross-dataset comparison of the cc_num values.

  - All `cc_num` entries present in customers.csv were found within credit_card_fraud.csv.
  - No unique or missing card numbers were detected in either direction.
  - This confirms that the **customer dataset is fully encompassed by the transaction dataset**.

- **Null Value Assessment:**

  - Both datasets were examined for missing values.
  - No nulls were found in the key or shared columns.

- **Duplicate Detection:**
    - The .duplicated().sum() method confirmed zero duplicate rows in both DataFrames.

1.3 Conclusion of Data Relationship

Since all customer_ids in customers.csv already existed in credit_card_fraud.csv, no join or merge was required.

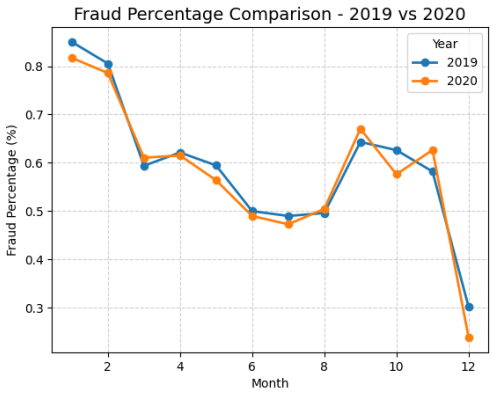
The customer table provided **no additional or distinct records**, and thus was excluded from further analysis.

Subsequent steps focus solely on preparing and cleaning the **credit_card_fraud.csv** file for analysis and modeling.
Date Range Confirmation:

To validate the temporal scope of the dataset, the minimum and maximum transaction dates were extracted from the trans_date field.
This inspection confirmed that the records cover a period **between 2019 and 2020.**

1.4 Initial Raw Data Reduction

To maintain the integrity and representativeness of the dataset while optimizing performance, the analysis period was limited to transactions records from **2020**.
This decision was based on both temporal relevance and observed fraud distribution patterns between the years 2019 to 2020.



Based on this temporal and seasonal consistency seen in above chart , transactions from **2020** were selected as a representative subset of the complete dataset.

Reducing our total raw data set from 37M to 1.7M transaction records for our flat file generation.

1.5 Flat file preparation:

	count	nulls	nulls%	cardinality	dtype
merch_long	17292422	0	0.0	13991489	float64
merch_lat	17292422	0	0.0	11063565	float64
unix_time	17292422	0	0.0	10472260	int64
amt	17292422	0	0.0	148733	float64
cc_num	17292422	0	0.0	18653	int64
acct_num	17292422	0	0.0	18653	int64
dob	17292422	0	0.0	13192	datetime64[ns]
zip	17292422	0	0.0	9847	int64
long	17292422	0	0.0	9706	float64
lat	17292422	0	0.0	9522	float64
city_pop	17292422	0	0.0	6023	int64
trans_date	17292422	0	0.0	366	datetime64[ns]
age	17292422	0	0.0	83	int64
trans_day	17292422	0	0.0	31	int32
trans_hour	17292422	0	0.0	24	int32
trans_time	17292422	0	0.0	24	int32
trans_month	17292422	0	0.0	12	int32
trans_dayofweek	17292422	0	0.0	7	int32
trans_quarter	17292422	0	0.0	4	int32
is_fraud	17292422	0	0.0	2	int64
gender_encoded	17292422	0	0.0	2	int8
trans_year	17292422	0	0.0	1	int32

Handle Date types

- Extracted components from trans_date: year, quarter, month, day, dayofweek.
- Customer age from dob.
- Added as separate columns for time-based feature engineering.

Handle Object Data Types

Methodological review of all the object types sored by cardinality :

	count	nulls	nulls%	cardinality	dtype
trans_num	17292422	0	0.0	17292422	object
ssn	17292422	0	0.0	18653	object
street	17292422	0	0.0	18653	object
city	17292422	0	0.0	5034	object
last	17292422	0	0.0	1000	object
merchant	17292422	0	0.0	693	object
first	17292422	0	0.0	686	object
job	17292422	0	0.0	639	object
state	17292422	0	0.0	51	object
category	17292422	0	0.0	14	object
profile	17292422	0	0.0	12	object
gender	17292422	0	0.0	2	object

❖ **Gender**

- Cleaned formatting (trimmed spaces, uppercased).
- Converted to category.
- Created new encoded numeric version (gender_encoded) with 0 = Female, 1 = Male.

❖ **Profile Parsing**

- Decomposed the profile field (a compound text field) into multiple components: Extracted age_group, gender, and location_profile.
- Redundancy verification :
    - Verified consistency between derived and existing gender columns.
    - Verified consistency between derived `age_group` to customer `age` derived from date of birth - 98% match 87% match confirm.

- The mismatch seemed to be related to rounding the age before classification (exp. 24 is grouped into the 25to50), the mistake however seems consistent, hence kept both for EDA.

❖ **Merchant Cleaning**

- Cleaned merchant names by removing “fraud_” prefix and punctuation.
- Converted to categorical and documented transformation.
- Medium cardinality.

❖ **Job Categorization**

- Normalized text to lowercase and stripped spaces.
- Reduction of Categorized occupations into high-level job groups using keyword matching.

❖ **State and Region Mapping**

- Mapped U.S. state abbreviations to geographic regions (e.g., West, Midwest).
- Added as a new categorical column for regional fraud analysis.

❖ **Category Column**

- Visualized transaction distribution by category using bar plots.
- Verified reasonable balance across purchase types.

❖ **City**

- Displayed top 10 most frequent cities.
- Converted all remaining object columns to categorical type.
- Saved cleaned dataset (df_2020_cat_clean.pkl).

Handle Non-Object Columns

	count	nulls	nulls%	cardinality	dtype
merch_long	17292422	0	0.0	13991489	float64
merch_lat	17292422	0	0.0	11063565	float64
unix_time	17292422	0	0.0	10472260	int64
amt	17292422	0	0.0	148733	float64
acct_num	17292422	0	0.0	18653	int64
cc_num	17292422	0	0.0	18653	int64
zip	17292422	0	0.0	9847	int64
long	17292422	0	0.0	9706	float64
lat	17292422	0	0.0	9522	float64
city_pop	17292422	0	0.0	6023	int64
trans_date	17292422	0	0.0	366	datetime64[ns]
age	17292422	0	0.0	83	int64
trans_day	17292422	0	0.0	31	int32
trans_time	17292422	0	0.0	24	int32
trans_hour	17292422	0	0.0	24	int32
trans_month	17292422	0	0.0	12	int32
trans_dayofweek	17292422	0	0.0	7	int32
trans_quarter	17292422	0	0.0	4	int32
is_fraud	17292422	0	0.0	2	int64
gender_encoded	17292422	0	0.0	2	int8

❖ **Latitude/Longitude Features**

- Calculated distance between customer and merchant locations.
- Prepared for potential spatial feature engineering.

1.6 Feature Drop

- **Dropped the following columns:**

- ['ssn', 'first', 'last', 'street', 'trans_num', 'profile', 'job', 'trans_year', 'dob', 'lat', 'long', 'merch_lat', 'merch_long']
- enshuring no loss of analytical information -only redundant, irrelevant, or sensitive attributes were removed.

- **Reasons for removal:**
  - **High cardinality or uniqueness:** ssn, trans_num, lat, long, merch_lat, merch_long
  - **Already processed or represented elsewhere:** profile, job, trans_year, dob
  - **Non-informative for modeling:** personal identifiers like first, last, street

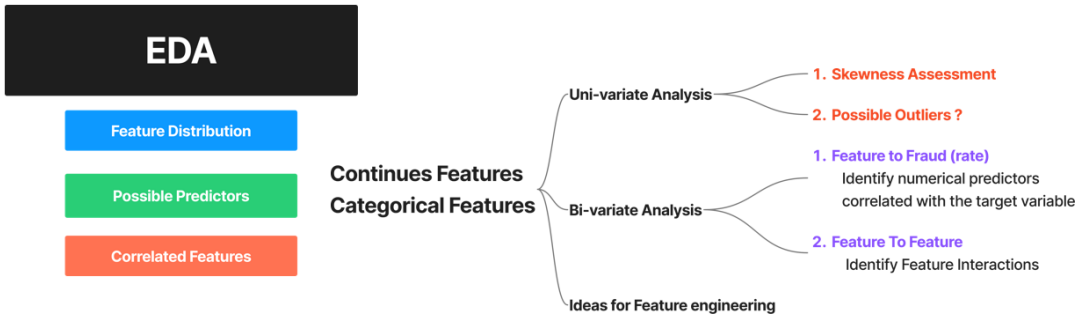
1.7 Clean Flat File

- Saved the resulting clean flat file in pkl format as the final prepared dataset for preprocessing and modeling.

Exploratory Data Analysis (EDA) Protocol Summary

The primary objective of the EDA phase was to systematically inspect the prepared dataset using statistical and visual analysis to inform feature selection, transformation, and model design. The methodology followed a five-step structure to investigate the relationship between all feature types and the target variable, as well as inter-feature redundancy.

Step	Check Type	Purpose
1	**Continuous → Target**	Assess the predictive power of numerical features against the target.
2	**Continuous ↔ Continuous**	Identify multicollinearity and feature redundancy among numerical variables.
3	**Categorical → Target**	Evaluate the influence of categorical variables on the target variable.
4	**Categorical ↔ Categorical**	Identify redundant or overlapping categorical variables.
5	**Continuous ↔ Categorical**	Explore feature interactions and demographic/time-based structure.



1. Data Preparation

The original flat file dataset (~17 million rows) was reduced via **Stratified Random Sampling** to create a manageable subset of **518,772 transactions** for the EDA phase. Stratification was performed on the binary target variable (is_fraud) to ensure the original fraud ratio was preserved in the sample.

Data Protocol & Quality Findings

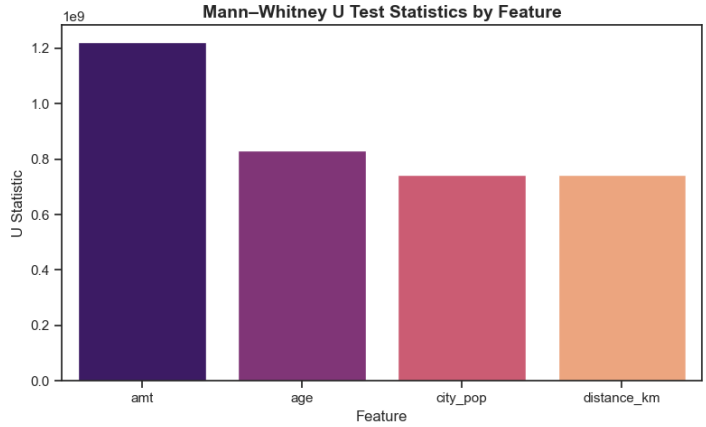
- **Missing Values:** No missing data was detected in any of the 25 features.
- **Target Imbalance:** The target column (is_fraud) shows a strong class imbalance, which will require specialized handling (e.g., SMOTE, down-sampling, or class weights) in the modeling phase.

2. Key Feature Insights

Continues Features and Interactions

1) Continuous Features to Target

Non-parametric testing (Mann-Whitney U test) confirmed a statistically significant difference in the distributions of all numerical features (amt, age, city_pop, distance_km) between the Fraud and Non-Fraud classes.



2) Continuous Feature Correlation

A Spearman correlation check revealed **no strong correlation** between the primary continuous predictors. The highest observed correlation was 0.12 (between age and amt). This confirms **no redundancy** (low multicollinearity) among these features, making them all safe to include in the model's initial feature set.

Categorical Features and Interactions

1) **Categorical-to-Target:**
   Chi-Squared tests with Cramer's V confirmed strong associations between categorical features and the target

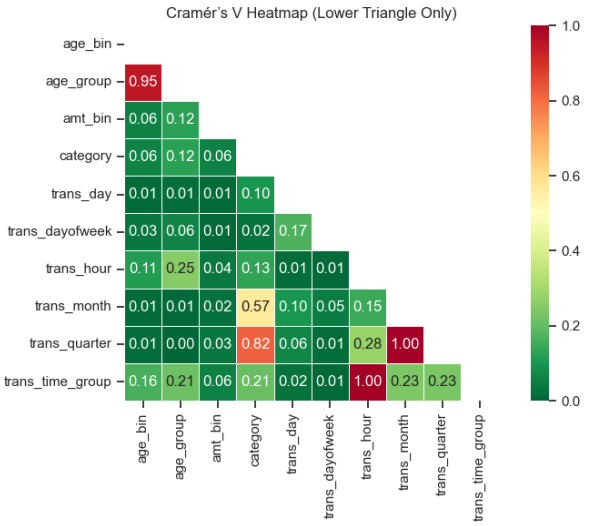
	Variable	Chi2 Statistic	p-value	Degrees of Freedom	Cramers V
2	category	375285.115881	0.000000e+00	13	0.850535
7	trans_hour	17272.882554	0.000000e+00	23	0.182471
13	amt_bin	14292.943155	0.000000e+00	9	0.165986
11	trans_time_group	8649.143825	0.000000e+00	3	0.129121
8	age_group	406.099161	6.556734e-89	2	0.027979
14	age_bin	349.163833	9.335424e-70	9	0.025943
3	trans_month	220.834414	3.460385e-41	11	0.020632
6	trans_dayofweek	132.462381	3.893741e-26	6	0.015979
4	trans_quarter	96.976907	6.940926e-21	3	0.013672
5	trans_day	55.561934	3.064529e-03	30	0.010349

2) **Categorical to Categorical :**

- **trans_quarter** was created directly from **trans_month**.
- **trans_time_group** (e.g., morning/night) is a binned version of **trans_hour**.
- The hourly feature is more precise (24 values), so we chose to **drop trans_time_group** and keep **trans_hour**.

- **Unexpected Strong Correlation:**

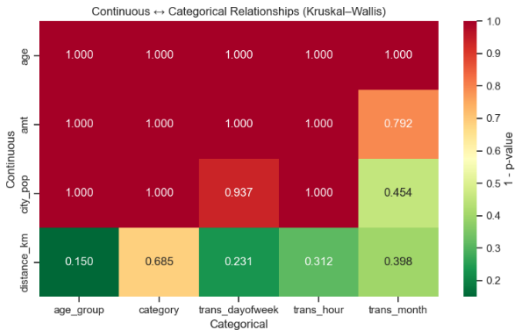
  An unexpectedly strong, non-engineered correlation was observed between **category** and **trans_quarter**. Since we are already keeping the more granular **trans_month**, and to avoid potential model bias from this strong association, we confirmed the decision to **drop trans_quarter**.



3) Continuous-Categorical Interaction:

Analysis showed that amt varies significantly across all categorical groupings This implies that transaction patterns are highly **group-driven**, suggesting that combining numerical features with categorical features (e.g., through group-based feature engineering) will be highly predictive.

The relationship between age and amt was found to be **non-linear and group-driven**, despite a low overall linear correlation.



Data Cleansing Protocol Summary

The Data Cleansing phase focused on ensuring the quality and stability of the numerical features prior to modeling by addressing missing values, correcting distributional skewness, and managing outliers.

1. Missing Data Verification

- **Protocol:** An initial check for null values was performed across all features.
- **Result: No missing values** were found in the dataset, meaning no imputation or deletion protocol was required.

2. Skewness and Transformation

- **Finding:** Two features were found to be highly **Right-Skewed**: amt (Skewness ≈ 47.6) and city_pop (Skewness ≈ 2.9).
- **Action:** A **log transformation** ($\log(1+x)$) was applied to both amt and city_pop.
- **Validation:** This step successfully normalized the distributions, reducing the skewness of amt to -0.01 and city_pop to -0.1.

3. Outlier Handling

- **Finding:** Outliers, especially in city_pop (≈16%) and amt (≈5.6%), were identified using the IQR method.
- **Action:** The log transformation applied to amt and city_pop was deemed sufficient to **mitigate the impact of these outliers** on the model.
- age outliers (representing high ages) were determined to be **valid data** points (not errors) and were **not treated**.

Feature Engineering and Selection

1. Feature Encoding and Scaling

- **Feature Encoding**
  - The protocol used **One-Hot Encoding** for the main categorical features after cleaning and grouping them, creating a binary representation for each category.

- **One-Hot Encoding**
  - was applied to features like category_clean (resulting in features such as category_clean_gas_transport, category_clean_grocery_pos, etc.).

- **Cyclical Encoding**
  - (using sine and cosine transformations) was applied to temporal features like trans_dayofweek and trans_month to capture their cyclical nature, resulting in features like trans_month_cos and trans_dayofweek_sin.

- **Label Encoding** (
  - binary 0/1) was used for gender (gender_encoded) and potentially other binary features.

Feature Scaling (Standardization)

The continuous numerical features were scaled using the **StandardScaler**.

Why Scaling is Done After the Split on the Train Set ?

- Scaling was performed **after** the data was split into training, development, and test sets (X_train_scaled, X_dev_scaled, X_test_scaled). This process is crucial to prevent **Data Leakage**:

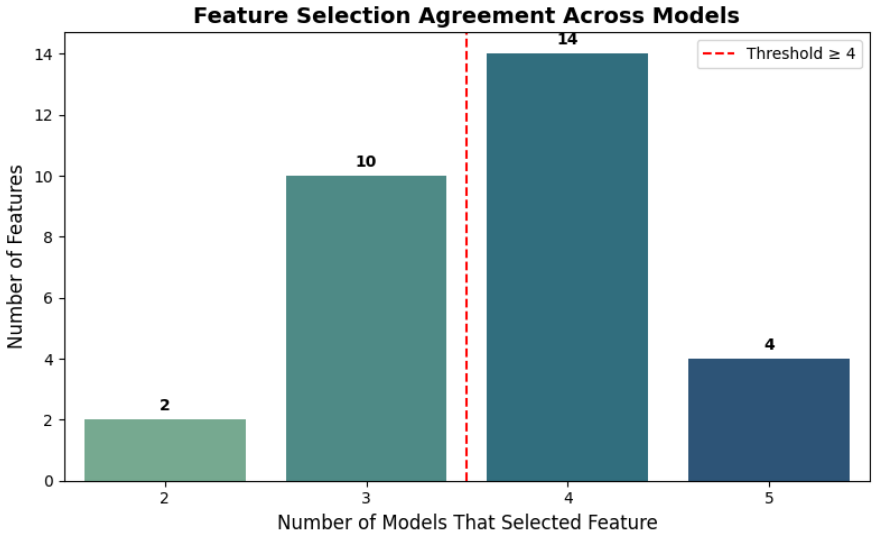
- The StandardScaler calculates the **mean and standard deviation** of the data it is fitted on. By fitting the scaler **only** on the **training data** (X_train), the model ensures that the scaling parameters do not contain any information (statistics) from the future-unseen validation or test sets.

- Once fitted on the training set, the scaler is used to **transform** all three datasets (train, dev, and test), maintaining the consistency of the data preparation while preserving the integrity of the test set as truly "unseen" data.

Ensemble and Linear Model Selection Results

Feature selection was conducted using five models to ensure the chosen features were robust across different model types (linear and tree-based/ensemble). A feature was considered "selected" if its coefficient (for linear models) or importance score (for ensemble models) was non-zero.

```
Reduced feature count: 30 (dropped 4)
✅ Feature selection complete:
                        Feature  LASSO  Ridge  SVM  GB  RF  LASSO_coef  \
0                     is_online      1      1    1   1   1    0.036105
1     category_clean_grocery_pos      1      1    1   1   1    0.023990
2   category_clean_gas_transport      1      1    1   1   1    0.011000
3        category_clean_misc_pos      1      1    1   1   1    0.002496
4                            amt      0      1    1   1   1    0.000000
5                       city_pop      0      1    1   1   1    0.000000
6            trans_dayofweek_sin      0      1    1   1   1    0.000000
7                    amt_per_age      0      1    1   1   1    0.000000
8                trans_month_cos      0      1    1   1   1    0.000000
9                trans_month_sin      0      1    1   1   1    0.000000
10                trans_hour_sin      0      1    1   1   1    0.000000
11                           age      0      1    1   1   1    0.000000
12                gender_encoded      0      1    1   1   1   -0.000000
13                    trans_time      0      1    1   1   1   -0.000000
14         category_clean_travel      0      1    1   1   1   -0.000000
```



✅ Selected 18 features:
['is_online', 'category_clean_grocery_pos', 'category_clean_gas_transport', 'category_clean_misc_pos', 'amt', 'city_pop', 'trans_dayofweek_sin', 'amt_per_age', 'trans_month_cos', 'trans_month_sin', 'trans_hour_sin', 'age', 'gender_encoded', 'trans_time', 'category_clean_travel', 'category_clean_shopping_pos', 'category_clean_home', 'trans_hour_cos']

Handling Data Imbalance

The training set exhibited a severe class imbalance, with the fraud class (is_fraud=1) representing only **0.54%** of the data.

- **Technique Used:** Several over- and under-sampling techniques were tested (Random OverSampling, Random UnderSampling, SMOTE, and SMOTETomek).

- **Final Choice: SMOTE** (Synthetic Minority Over-sampling Technique) was chosen to create a balanced training set. SMOTE synthesizes new minority samples via interpolation, bringing the fraud class proportion up to 50%.

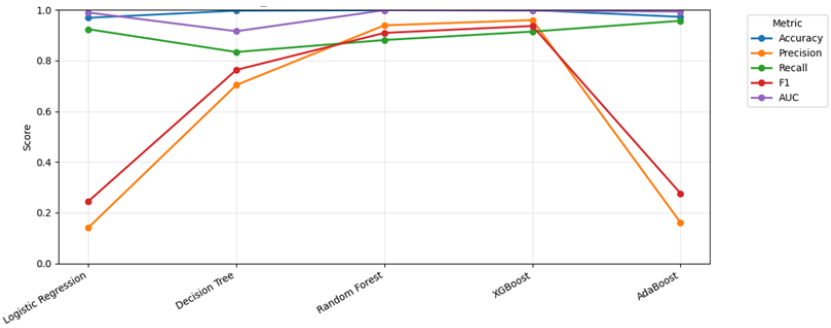
- **Justification:** While SMOTETomek and Random UnderSampling achieved slightly higher AUCs, SMOTE was selected for its balance of performance (AUC of 0.9954) and method reliability.

Initial Ensemble and Linear Model Selection Results

Five distinct models were evaluated on the **development (dev) set** using the balanced training data to select the most promising algorithm. Metrics used were **Precision, Recall, F1-score, and AUC**, as accuracy is misleading in highly imbalanced datasets.

- **Champion Model Selection:**
  - **XGBoost** was chosen as the champion model because it achieved the **best overall F1-score (0.937)** and a high AUC, demonstrating the strongest trade-off between identifying fraud (high Recall) and minimizing false alerts (high Precision).

- **Linear Model Performance:**
  - Linear models (Logistic Regression and AdaBoost) showed very high Recall but extremely low Precision (e.g., Logistic Regression Precision: 0.141), making them impractical for production fraud alerts due to excessive false positives.



	Model	Accuracy	Precision	Recall	F1	AUC
0	Logistic Regression	0.969081	0.141100	0.924171	0.244821	0.989894
1	Decision Tree	0.997199	0.704000	0.834123	0.763557	0.916105
2	Random Forest	0.999049	0.939394	0.881517	0.909535	0.999068
3	XGBoost	0.999332	0.960199	0.914692	0.936893	0.997272
4	AdaBoost	0.972846	0.161665	0.957346	0.276618	0.993542

Hyperparameter Tuning

The final stage involved optimizing the selected XGBoost model:

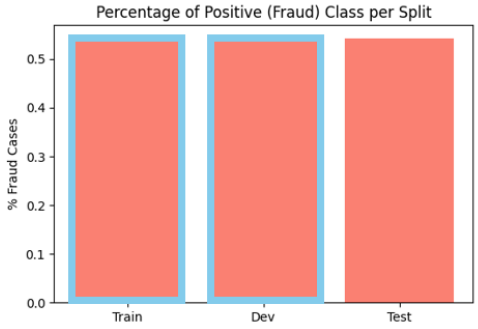
- **Tuning Method:**
  - **Randomized Search Cross-Validation (RandomizedSearchCV)** was used to efficiently search for the optimal combination of hyperparameters.

- **Best Parameters:**
  - The best parameters found after tuning were:
  - n_estimators: 300
  - max_depth: 6
  - learning_rate: 0.1
  - subsample: 1.0
  - colsample_bytree: 0.8
  - scale_pos_weight: 3

📊 XGBoost: Base vs. Tuned Performance Comparison

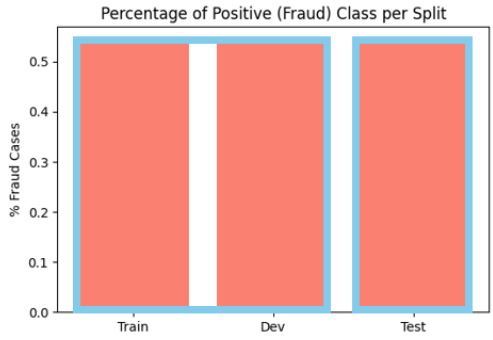
Metric	Base XGB	Best Tuned XGB	Change	Interpretation
Accuracy	0.9993	0.9991	−0.0002	Practically identical
Precision	0.9602	0.9147	↓	Slight increase in false positives
Recall	0.9147	0.9147	—	Unchanged
F1	0.9369	0.9147	↓	Minor drop (threshold trade-off)
AUC	0.9973	0.9979	↑	Slightly *better* ranking ability

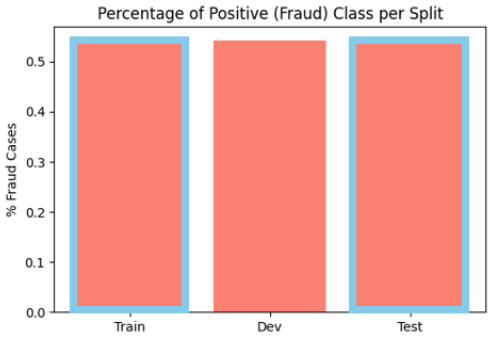
Continue with **Base XGB**



Final Model Retraining on Combined Train + Dev Sets

After the best model (XGBoost) and its optimal hyperparameters were identified using the **training set** and validated on the independent **development set**, the final model was re-trained on the combined **Train** + **Dev** data.





Model	Accuracy	Precision	Recall	F1 Score	AUC
Pre-Merge (Train only)	0.999062	0.930864	0.893365	0.911729	0.993962
Final (Train+Dev)	**0.999100**	**0.940000**	0.890995	**0.914842**	**0.994775**

1. Test Evaluation Results

The tuned XGBoost model was evaluated one final time on the independent test set (X_test_selected, y_test) to confirm its real-world generalization capability, achieving a very strong balance between identifying fraud and minimizing false alarms.

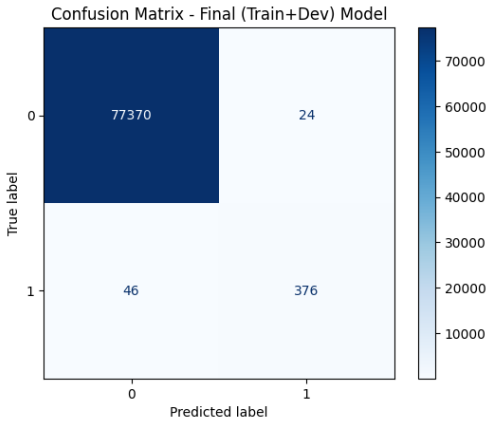
2. Feature Importance Analysis

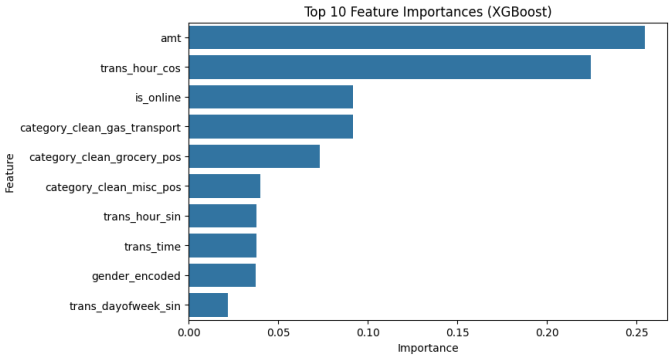
The analysis of feature importance confirmed the predictive power of the final selected variables:

- **Dominant Predictors:** The most important features were overwhelmingly related to **Time** (e.g., trans_hour, trans_dayofweek_sin/cos), suggesting fraud activity concentrates during specific temporal windows.

- **Core Fundamentals:** Fundamental risk factors like **amt** (transaction amount), **city_pop**, and **age** remained highly influential.

- **High-Risk Categories:** Specific transactional categories such as gas_transport and misc_pos also ranked high, reinforcing their intuitive risk profile.


Confusion Matrix - Final (Train+Dev) Model
77370
24
46
376
0
1
True label
Predicted label
70000
60000
50000
40000
30000
20000
10000


Top 10 Feature Importances (XGBoost)
amt
trans_hour_cos
is_online
category_clean_gas_transport
category_clean_grocery_pos
category_clean_misc_pos
trans_hour_sin
trans_time
gender_encoded
trans_dayofweek_sin
Feature
0.00
0.05
0.10
0.15
0.20
0.25
Importance

3. Final Retraining on Combined Train + Dev

A critical final step involved retraining the model on the combined **Training and Development** datasets:

- **The Process:**

The final optimized hyperparameters (found using the Development set) were locked in. The final XGBoost model was then re-trained using all data from the Training set **plus** the Development set (X_train_selected + X_dev_selected).

- **The Rationale:**

This strategy maximizes the amount of data the final model learns from, potentially increasing its robustness and generalization ability across the known data distributions.

- **Preventing Data Leakage:**

This step is only performed *after* the Development set has fulfilled its purpose (hyperparameter tuning). The **Test set** remains untouched and truly unseen until the final performance metrics were calculated, ensuring the reported results are an unbiased measure of the final model's effectiveness in a real-world setting.

Metric	Result	Interpretation
F1-Score	**0.91**	Represents the balance between Precision and Recall. A high score confirms the model's overall effectiveness.
AUC	**0.9942**	Indicates excellent discriminative power, meaning the model is highly effective at separating fraud from non-fraud cases.
Precision	**0.94**	Of all transactions flagged as fraud, **94% were correctly identified** (minimizing false positive alerts).
Recall	**0.89**	The model successfully identified **89% of all actual fraud cases** (minimizing missed fraud).