# Predicting Mobile Transaction Fraud Using XGBoost Ensemble Model

Created by: Kshitiz Badola, Sobnom Mostari, Jason Ortiz

# Motivation:

- As technology advances it works in favor of both those trying to use it to exploit others, and those using it to defend themselves.

- Machine Learning has allowed for transactions to be deemed as fraud or not at the moment at the transaction and potentially save people a lot of money.

- You can’t have a model that performs poorly, with regards to both accuracy and time.

# PaySim Synthetic Dataset

- Using the PaySim financial money simulator Kaggle dataset we are able to get data similar to otherwise highly confidential data.

- The dataset consists of 6,362,620 rows and 11 features (10 decision/1 target).

- Before any modeling can be done we need to understand the data that we are working with before any modeling can be done.

```
original_df.head()
```

| | step | type | amount | nameOrig | oldbalanceOrg | newbalanceOrig | nameDest | oldbalanceDest | newbalanceDest | isFraud | isFlaggedFraud |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 1 | PAYMENT | 9839.64 | C1231006815 | 170136.0 | 160296.36 | M1979787155 | 0.0 | 0.0 | 0 | 0 |
| **1** | 1 | PAYMENT | 1864.28 | C1666544295 | 21249.0 | 19384.72 | M2044282225 | 0.0 | 0.0 | 0 | 0 |
| **2** | 1 | TRANSFER | 181.00 | C1305486145 | 181.0 | 0.00 | C553264065 | 0.0 | 0.0 | 1 | 0 |
| **3** | 1 | CASH_OUT | 181.00 | C840083671 | 181.0 | 0.00 | C38997010 | 21182.0 | 0.0 | 1 | 0 |
| **4** | 1 | PAYMENT | 11668.14 | C2048537720 | 41554.0 | 29885.86 | M1230701703 | 0.0 | 0.0 | 0 | 0 |

```
original_df.shape
```

```
(6362620, 11)
```

```
original_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6362620 entries, 0 to 6362619
Data columns (total 11 columns):
 #   Column          Dtype  
---  ------          -----  
 0   step            int64  
 1   type            object 
 2   amount          float64
 3   nameOrig        object 
 4   oldbalanceOrg   float64
 5   newbalanceOrig  float64
 6   nameDest        object 
 7   oldbalanceDest  float64
 8   newbalanceDest  float64
 9   isFraud         int64  
 10  isFlaggedFraud  int64  
dtypes: float64(5), int64(3), object(3)
memory usage: 534.0+ MB
```

```
original_df.isnull().sum()
```

```
step              0
type              0
amount            0
nameOrig          0
oldbalanceOrg     0
newbalanceOrig    0
nameDest          0
oldbalanceDest    0
newbalanceDest    0
isFraud           0
isFlaggedFraud    0
dtype: int64
```

# Exploratory Data Analysis (EDA):

- The first feature that is important to get a good understanding of is the target feature: isFraud, of which there are only 8,213 instances in the entire dataset (6,000,000+ rows). This makes up only approximately 0.129% of the data, making the dataset very imbalanced.

- By using the types of transactions that actually contain fraud will help with the imbalance nature of the data, bringing down the total number of instances to 2,770,409 instances (down to 43.5% of the original data) while still having all 8,213 fraud transactions. This brings the fraud transactions to 0.296% of the dataset.

# Data Preprocessing

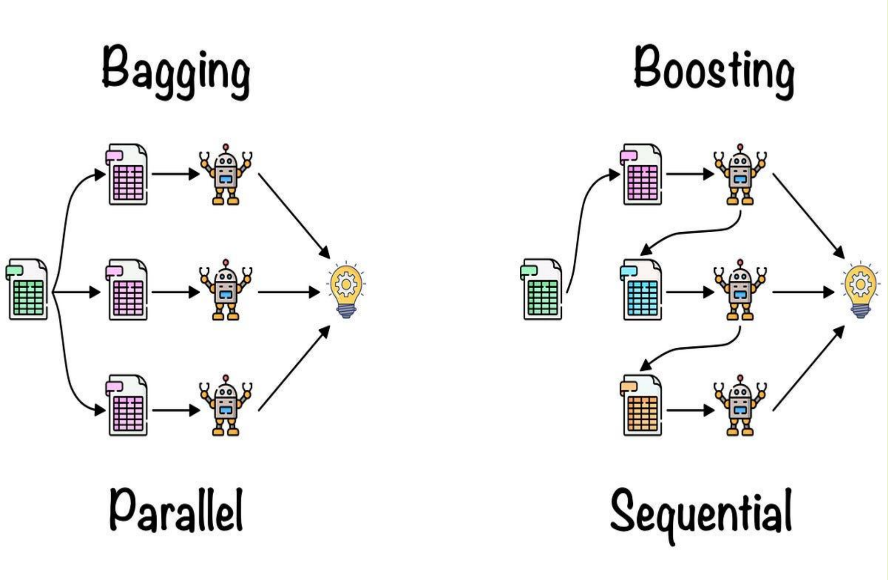
- Some features such as isFlaggedFraud were removed.
- Other features such as type and nameDest were encoded.
- Separate scaled data had to be created in preparation for some of the baseline models

```
useful_data.isFlaggedFraud.value_counts()

0    2770393
1         16
Name: isFlaggedFraud, dtype: int64

useful_data = useful_data.drop('isFlaggedFraud', axis=1)

len(useful_data)/len(original_df)

0.4354195284332555
```

```
columns_to_encode = ['type', 'nameOrig']

useful_data['type']

2          TRANSFER
3          CASH_OUT
15         CASH_OUT
19         TRANSFER
24         TRANSFER
             ...
6362615    CASH_OUT
6362616    TRANSFER
6362617    CASH_OUT
6362618    TRANSFER
6362619    CASH_OUT
Name: type, Length: 2770409, dtype: object
```

# Ensemble Learning

- Combines multiple models to improve the overall performance and accuracy of a predictive model
- **Bagging**:
  - Decreases overall variance
  - Aggregates several sampling subsets of the original dataset to train different learners chosen randomly with replacement.



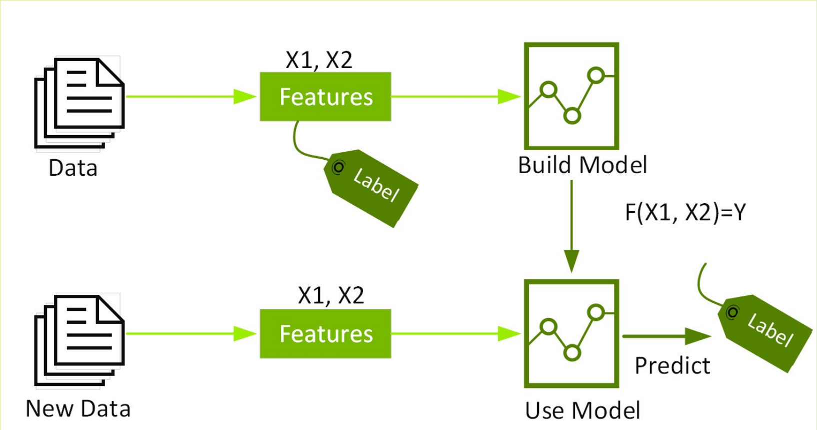
# Ensemble Learning

- **Boosting**:
    - Multiple models are trained sequentially to improve the performance and accuracy of a predictive model.
    - Models are trained iteratively
    - The idea is to reduce the bias and improve the accuracy of the model by focusing on the instances that were misclassified by the previous models.

| | Bagging | Boosting |
|---|---|---|
| Purpose | Reduce Variance | Reduce Bias |
| Base Learner Types | Homogeneous | Homogeneous |
| Base Learner Training | Parallel | Sequential |
| Aggregation | Max Voting, Averaging | Weighted Averaging |

# XGBoost

- Is a technique for building ensemble models
- Combines weak models, typically decision trees, into a stronger model
- Is able to handle large datasets with high-dimensional features
- Provides high prediction accuracies
- Used in various applications such as regression, classification, and ranking problems



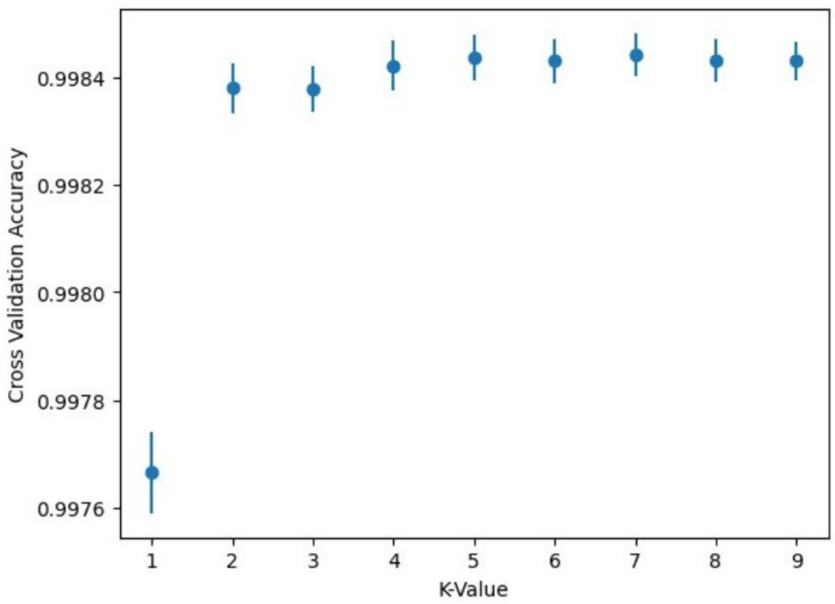
# Benefits of XGBoost

- **Auto Pruning:**
  - A feature that ensures that the trees do not grow beyond a limit.
  - Manages the tree structure and removes unnecessary parts in the classification
- **Cache Optimization:**
  - Used to store calculations and statistics in cache during training; by doing this, the algorithm can access this information quickly.
- **Parallelization:**
  - Utilizes all the cores of the CPU for training
- **Regularization:**
  - Technique that is used to calibrate machine learning models in order to minimize the adjusted loss function and prevent overfitting

# Baseline Models

- Before training our XGBoost model we first decided to train multiple other models on our data.
- We decided on training and testing some of the models learned about in this course: **K-Nearest-Neighbors, Random Forest, Logistic Regression/SGDClassifier,** as well as **Gaussian & Bernoulli Naive Bayes classifiers**.
- We tested each model using accuracy, confusion matrices, precision & recall.

# KNN



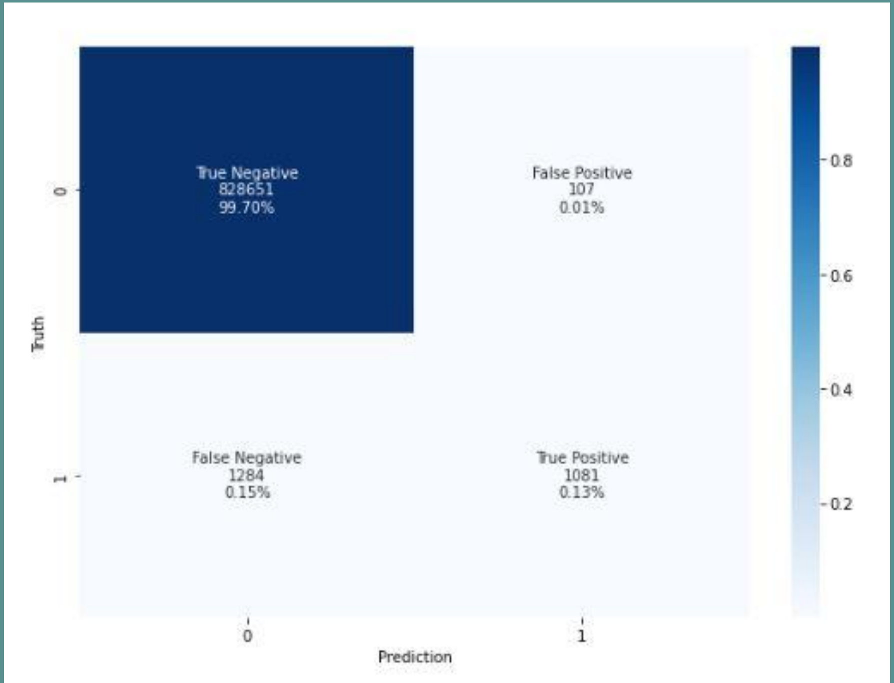

```
    Mean: 0.998441694489449
    Standard Deviation: 3.996351676252575e-05
The optimal K value = 7
```

# KNN

Classification Report

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 1.00 | 1.00 | 1.00 | 828758 |
| 1 | 0.91 | 0.47 | 0.62 | 2365 |
| accuracy | | | 1.00 | 831123 |
| macro avg | 0.95 | 0.74 | 0.81 | 831123 |
| weighted avg | 1.00 | 1.00 | 1.00 | 831123 |

Accuracy Score
0.9983588470057981

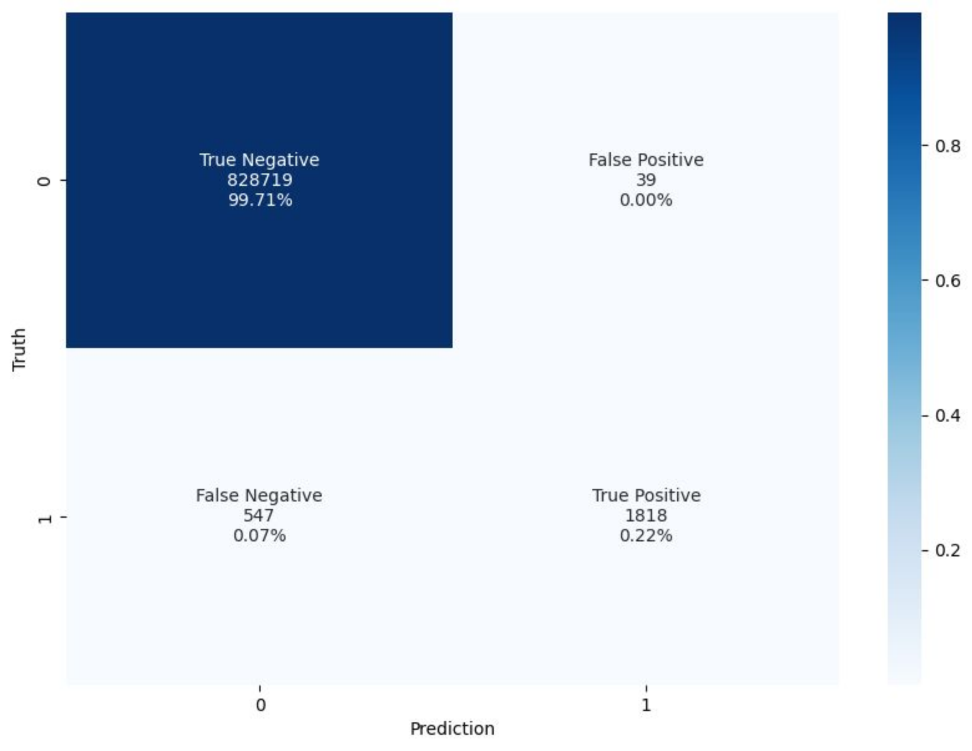


# Random Forest

Classification Report

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 1.00 | 1.00 | 1.00 | 828758 |
| 1 | 0.98 | 0.77 | 0.86 | 2365 |
| accuracy | | | 1.00 | 831123 |
| macro avg | 0.99 | 0.88 | 0.93 | 831123 |
| weighted avg | 1.00 | 1.00 | 1.00 | 831123 |

Accuracy Score
0.9992949298719924

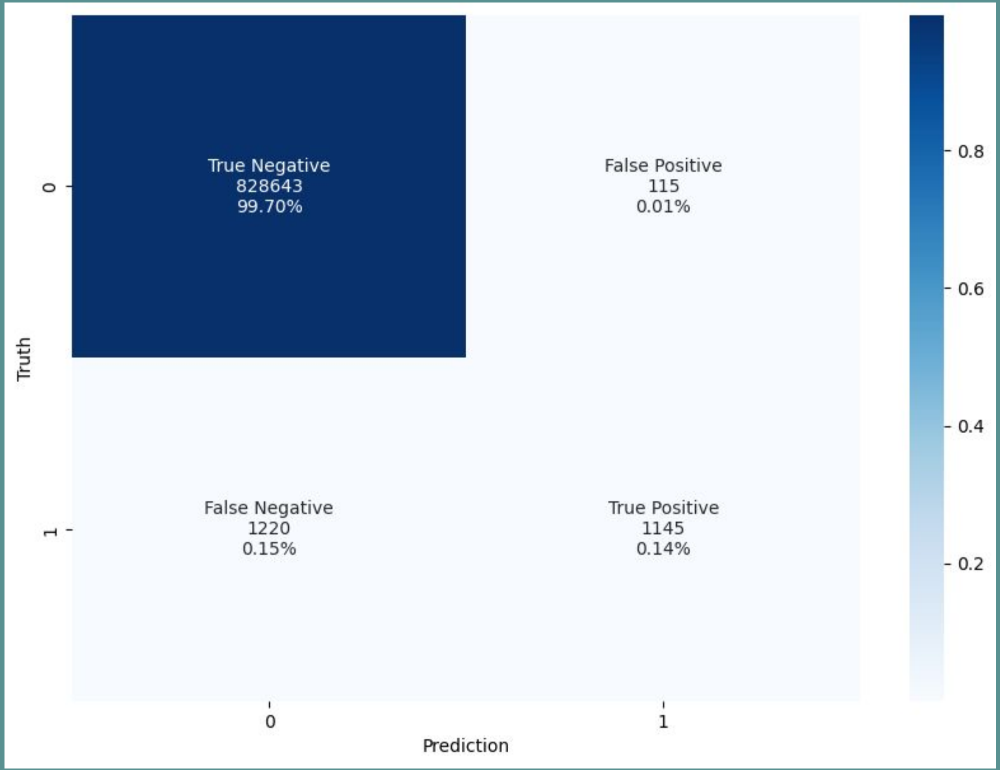


# Logistic Regression

Classification Report

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 1.00 | 1.00 | 1.00 | 828758 |
| 1 | 0.91 | 0.48 | 0.63 | 2365 |
| accuracy | | | 1.00 | 831123 |
| macro avg | 0.95 | 0.74 | 0.82 | 831123 |
| weighted avg | 1.00 | 1.00 | 1.00 | 831123 |

Accuracy Score
0.9983937395547952

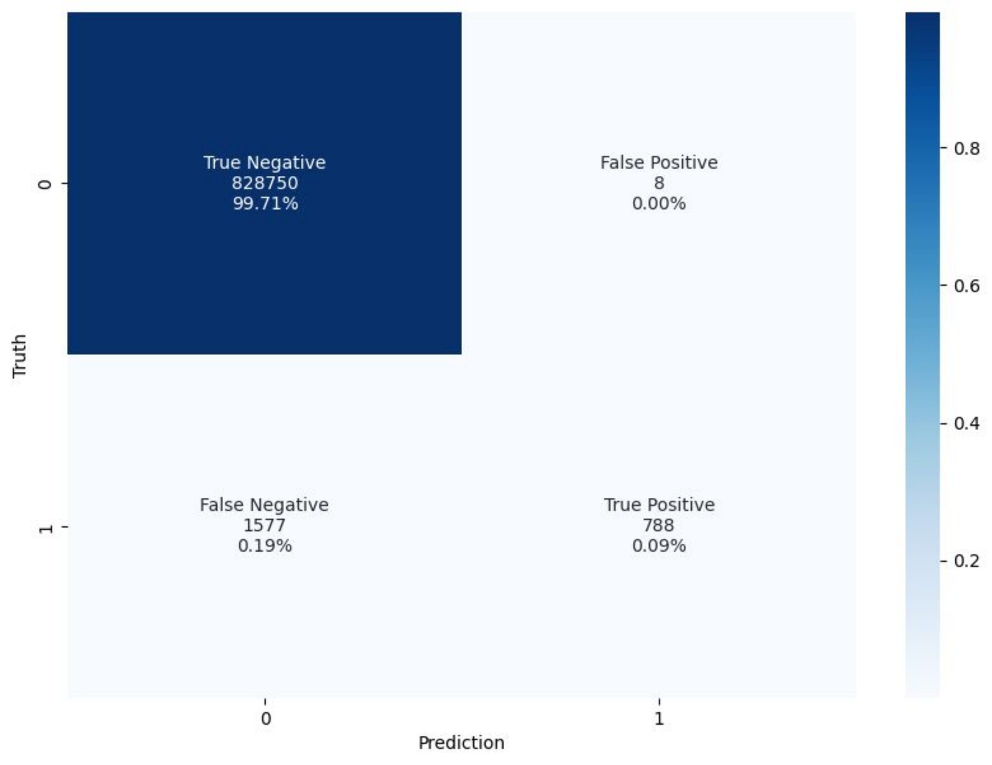


# SGD Classifier

Classification Report

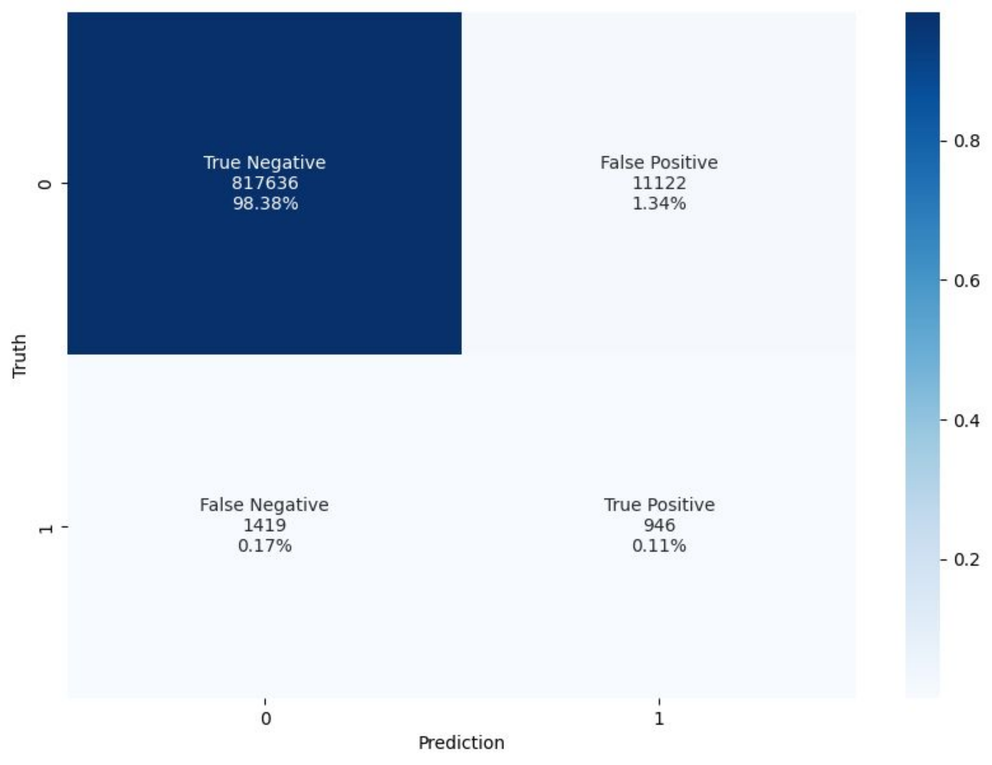
| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 1.00 | 1.00 | 1.00 | 828758 |
| 1 | 0.99 | 0.33 | 0.50 | 2365 |
| accuracy | | | 1.00 | 831123 |
| macro avg | 0.99 | 0.67 | 0.75 | 831123 |
| weighted avg | 1.00 | 1.00 | 1.00 | 831123 |

Accuracy Score
0.9980929417186144



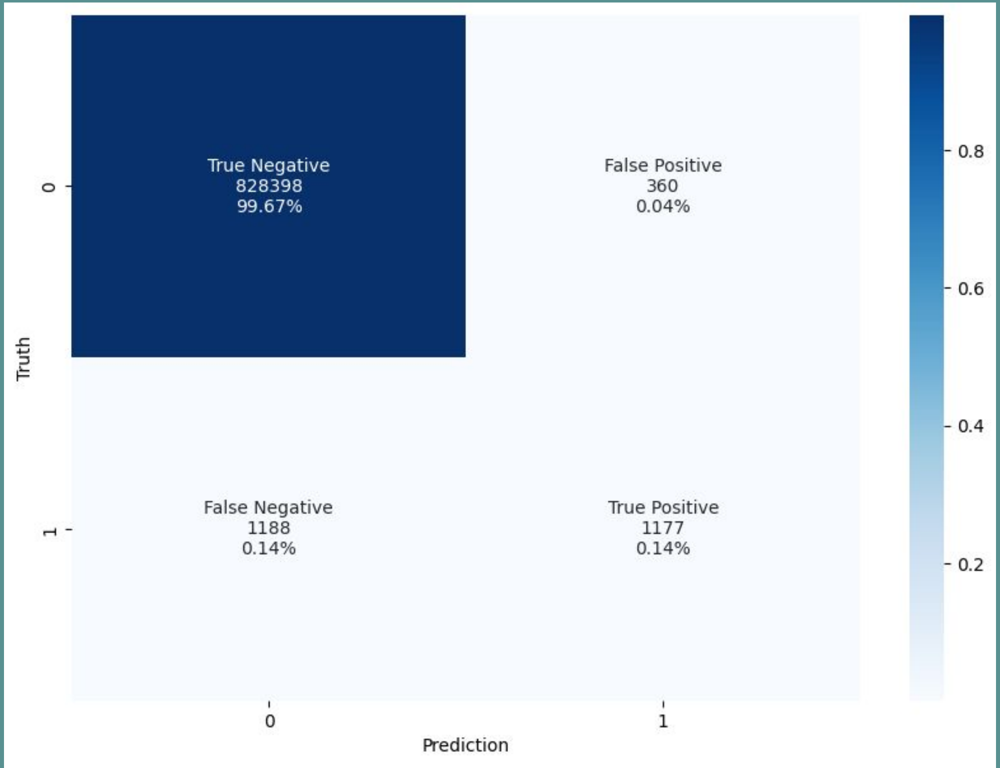
# Gaussian Naive Bayes

```
Classification Report
              precision    recall  f1-score   support

           0       1.00      0.99      0.99    828758
           1       0.08      0.40      0.13      2365

    accuracy                           0.98    831123
   macro avg       0.54      0.69      0.56    831123
weighted avg       1.00      0.98      0.99    831123

Accuracy Score
 0.9849107773458321
```



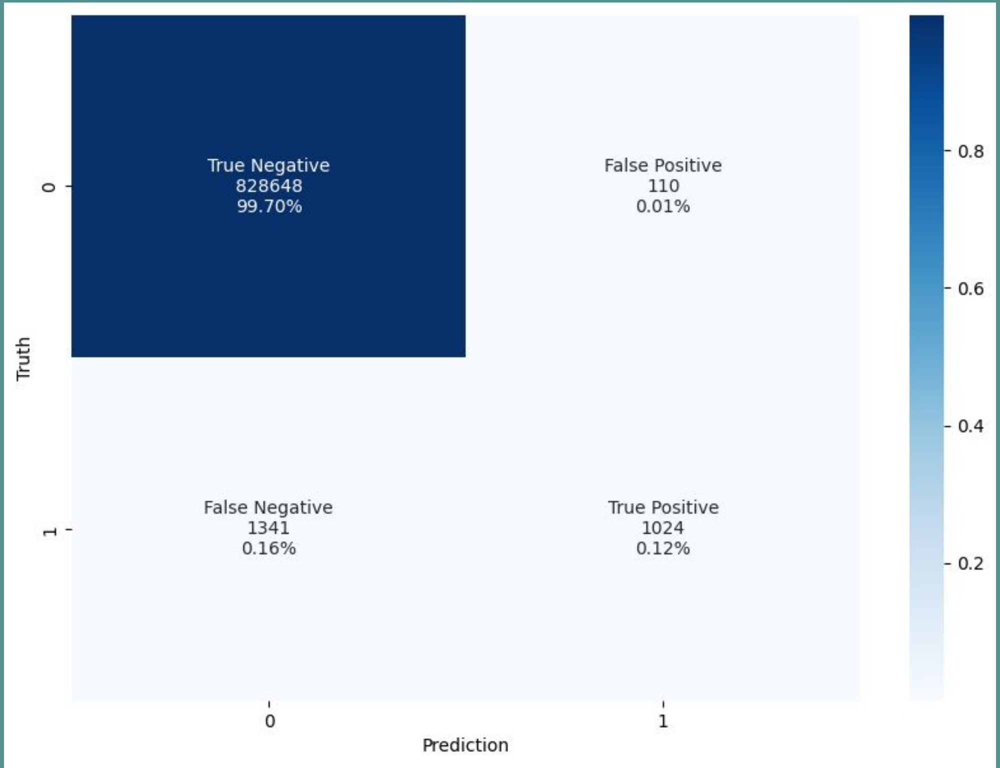
# Bernoulli Naive Bayes

```
Classification Report
              precision    recall  f1-score   support

           0       1.00      1.00      1.00    828758
           1       0.77      0.50      0.60      2365

    accuracy                           1.00    831123
   macro avg       0.88      0.75      0.80    831123
weighted avg       1.00      1.00      1.00    831123

Accuracy Score
 0.9981374597983692
```



# XGBoost

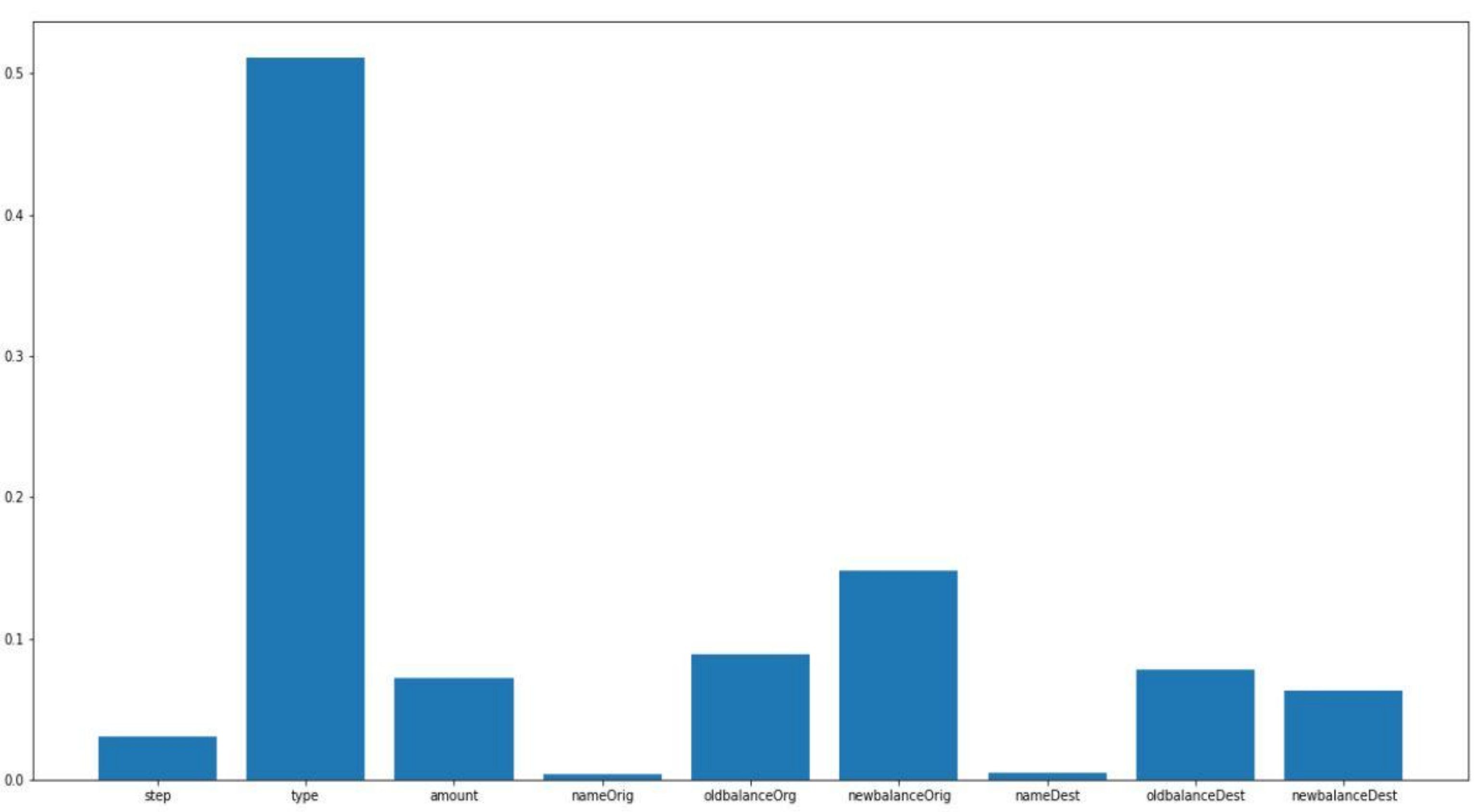
```
Classification Report
              precision    recall  f1-score   support

           0       1.00      1.00      1.00    828758
           1       0.90      0.43      0.59      2365

    accuracy                           1.00    831123
   macro avg       0.95      0.72      0.79    831123
weighted avg       1.00      1.00      1.00    831123

Accuracy Score
 0.9982541693588073
```



# Results

| \ Metric<br>Model \ | Accuracy | True Positive | True Negative | Precision | Recall |
|---|---|---|---|---|---|
| K-Nearest Neighbors (K=7) | 99.83% | 0.13% | 99.70% | 91% | 46% |
| Random Forest | **99.93%** | **0.22%** | **99.71%** | **98%** | **78%** |
| Logistic Regression | **99.84%** | 0.14% | 99.70% | 91% | 48% |
| SGDClassifier | 99.81% | 0.10% | **99.71%** | **99%** | 34% |
| GaussianNB | 98.49% | 0.11% | 98.38% | 8% | 40% |
| BernoulliNB | 99.81% | 0.14% | 99.67% | 77% | 50% |
| XGBoost(Tree-based) | **99.95%** | **0.24%** | **99.71%** | **98%** | **86%** |

# Results


0.5
0.4
0.3
0.2
0.1
0.0
step
type
amount
nameOrig
oldbalanceOrg
newbalanceOrig
nameDest
oldbalanceDest
newbalanceDest

# References:


- Seif, G. (2022, February 11). A beginner’s guide to xgboost. Medium. https://towardsdatascience.com/a-beginners-guide-to-xgboost-87f5d4c30ed7
- Gupta, P. (2017, November 16). Regularization in machine learning. Medium. https://towardsdatascience.com/regularization-in-machine-learning-76441ddcf99a
- Hachcham, A. (2023, April 25). XGBoost: Everything you need to know. neptune.ai. https://neptune.ai/blog/xgboost-everything-you-need-to-know
- Lopez-Rojas, E. (2017, April 3). Synthetic financial datasets for fraud detection. Kaggle. https://www.kaggle.com/datasets/ealaxi/paysim1