Lekh Sanata n Nayak
2023800068 A4

# Fraud Detection (HSBC HACKATHON 2023)

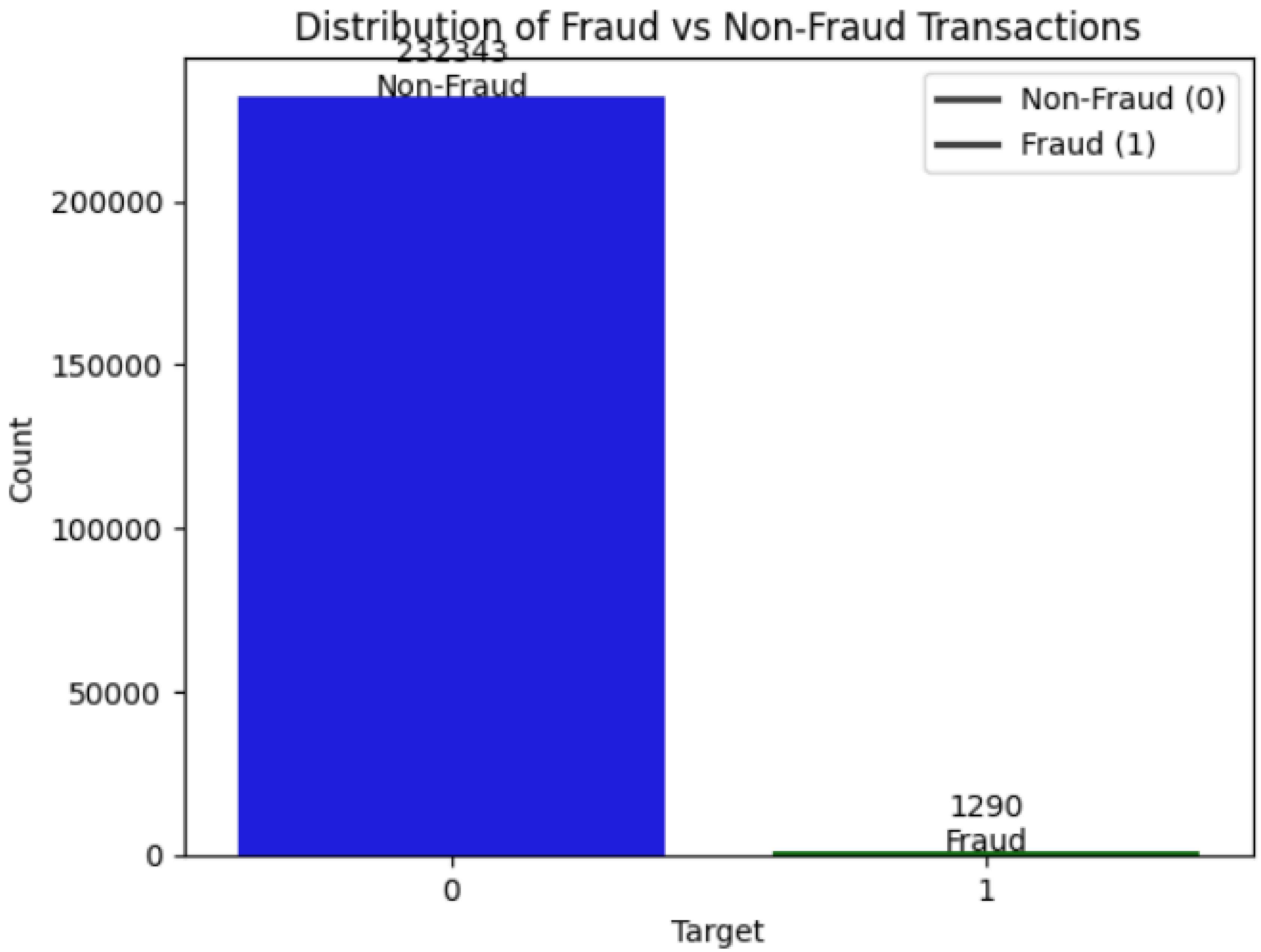
# Introduction

The dataset contains a set of banking transactions labeled as either genuine or fraudulent. This data has been provided as part of a real-time fraud detection challenge focused on the Indian digital payments ecosystem. Given India’s significant volume of real-time transactions, the risk of financial fraud has grown substantially. This dataset simulates real-world banking scenarios to help build a machine learning model that predicts whether a transaction was genuinely initiated by the customer.

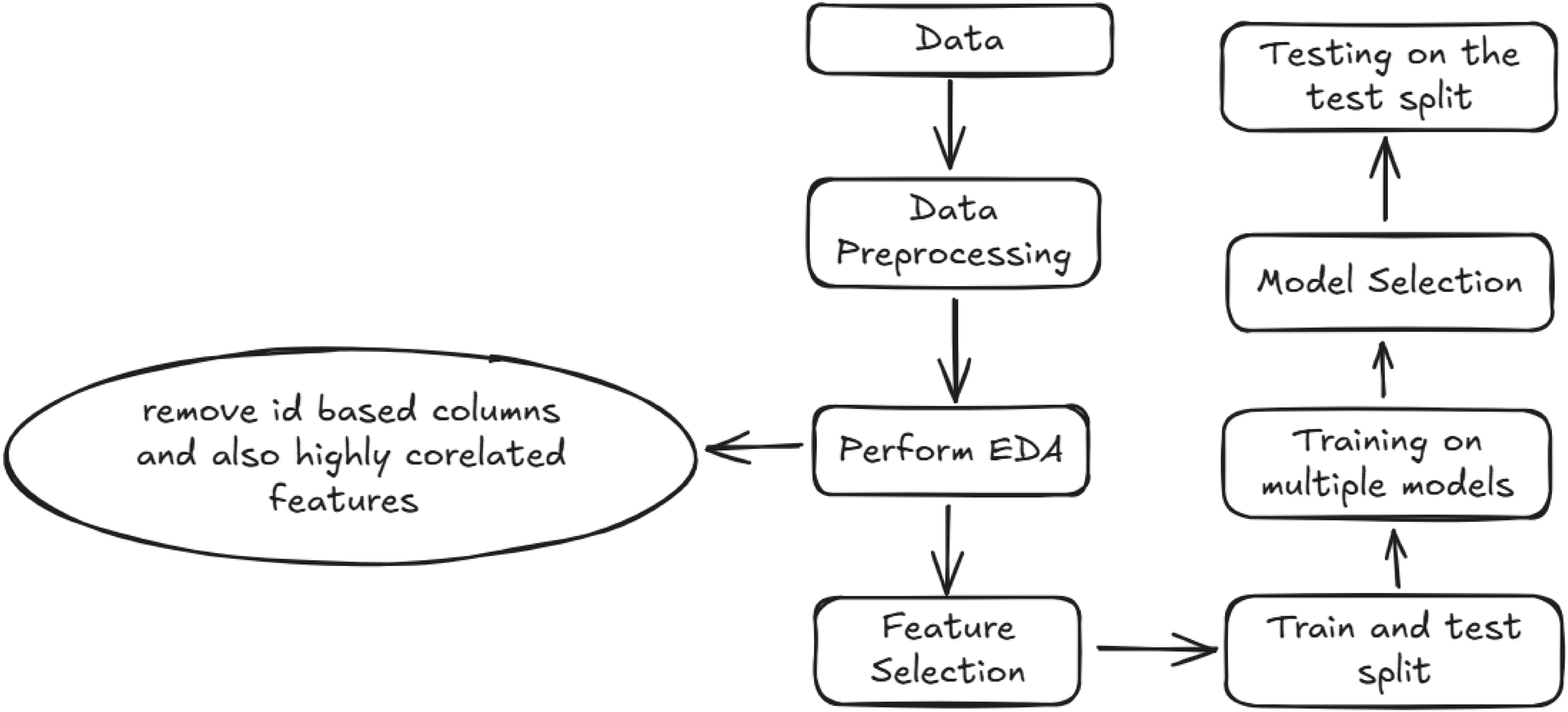
The dataset includes various features captured at the time of payment initiation, such as:

- **Transaction Metadata** (Timestamps, Transaction IDs, etc.)
- **User Behavior Patterns** (Amount, Frequency, Timing)
- **Device/Session Details** (IP Address, Device Fingerprint, Session ID)

The key challenge lies in the high class **imbalance**, where fraudulent transactions are much fewer than genuine ones, making it a suitable problem for applying advanced resampling techniques and robust classification models.



# Workflow of the Project



# Data Preprocessing

```
train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 233633 entries, 0 to 233632
Data columns (total 14 columns):
 #   Column   Non-Null Count   Dtype  
---  ------   --------------   -----  
 0   V1       233633 non-null  object 
 1   V2       233633 non-null  object 
 2   V3       233633 non-null  object 
 3   V4       233633 non-null  float64
 4   V5       233633 non-null  object 
 5   V6       233633 non-null  int64  
 6   V7       233633 non-null  object 
 7   V8       233633 non-null  object 
 8   V9       233633 non-null  object 
 9   V10      233633 non-null  object 
 10  V11      233633 non-null  object 
 11  V12      233633 non-null  object 
 12  Target   233633 non-null  int64  
 13  V13      231762 non-null  object 
dtypes: float64(1), int64(2), object(11)
memory usage: 25.0+ MB
```

```
train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 233633 entries, 0 to 233632
Data columns (total 19 columns):
 #   Column      Non-Null Count   Dtype         
---  ------      --------------   -----         
 0   V1          233633 non-null  int64         
 1   V2          233633 non-null  int64         
 2   V3          233633 non-null  int64         
 3   V4          233633 non-null  float64       
 4   V6          233633 non-null  int64         
 5   V7          233633 non-null  int32         
 6   V8          233633 non-null  int64         
 7   V9          233633 non-null  int64         
 8   V10         233633 non-null  int64         
 9   Target      233633 non-null  int64         
 10  V13         231762 non-null  datetime64[ns]
 11  V5_year     233633 non-null  int32         
 12  V5_month    233633 non-null  int32         
 13  V5_day      233633 non-null  int32         
 14  V5_hour     233633 non-null  int32         
 15  V5_min      233633 non-null  int32         
 16  V5_sec      233633 non-null  int32         
 17  is_weekend  233633 non-null  bool          
 18  season      233633 non-null  int32         
dtypes: bool(1), datetime64[ns](1), float64(1), int32(8), int64(8)
memory usage: 25.2 MB
```

# Top K features

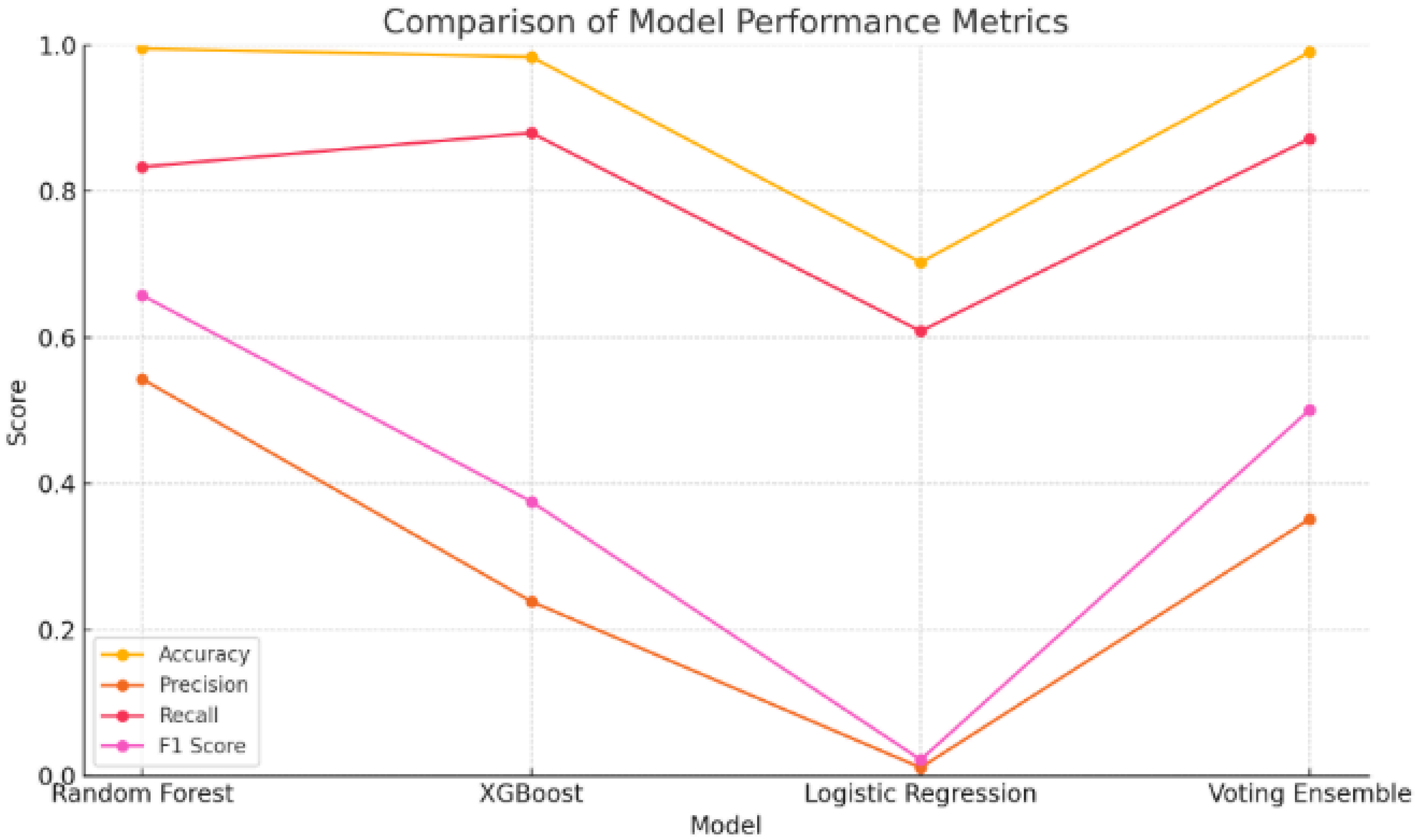
To ensure the machine learning model utilized only the most relevant and informative variables, multiple statistical feature selection techniques were employed, including **Chi-squared test**, **ANOVA F-test**, and **Mutual Information**. Each of these methods evaluates the strength of the relationship between independent features and the target variable, helping to identify features with the highest predictive potential.

**Top 4 features :**

- **V20 – Highest ANOVA F-score (454.07), high Chi2 score**
- **V17_IP_B – Strong ANOVA F-score (39.28), decent Chi2**
- **V16_month – Good ANOVA F-score (32.70), moderate Chi2**
- **V14 – High ANOVA F-score (30.55), good Chi2**
- **V19 – Moderate scores across all metrics**

| | Feature | Chi2 Score | ANOVA F-Score | Mutual Info |
|---|---|---|---|---|
| 3 | V18 | 0.004279 | 0.372849 | 0.059737 |
| 1 | V14 | 4.337522 | 30.548200 | 0.032716 |
| 7 | V16_month | 0.642108 | 32.695330 | 0.025536 |
| 0 | V1 | 0.405543 | 2.369698 | 0.022680 |
| 8 | V16_day | 0.052067 | 5.117973 | 0.014765 |
| 18 | V5_month | 1.001473 | 5.909114 | 0.008626 |
| 5 | V20 | 20.698691 | 454.065385 | 0.007598 |
| 4 | V19 | 1.781216 | 9.297205 | 0.007322 |
| 13 | V17_IP_B | 3.826857 | 39.279270 | 0.006894 |
| 14 | V17_IP_C | 0.262705 | 4.079803 | 0.006788 |

# Performance Metrics of All the graphs



# Hyperparameter Tuning

The objective of hyperparameter tuning in this project was to enhance model performance by optimizing critical parameters of the XGBoost classifier. Effective tuning ensures that the model achieves a balance between bias and variance, accelerates training, and improves generalization to new, unseen data

```
best_xgb = XGBClassifier(
    learning_rate=0.3,
    max_depth=6,
    n_estimators=200,
    scale_pos_weight=15,
    eval_metric='logloss',
    use_label_encoder=False,
    random_state=42
)
```

Best Combination of Hyperparameters

To identify the optimal set of hyperparameters, GridSearchCV was employed. Grid Search is an exhaustive search technique that evaluates all possible combinations of specified hyperparameter values based on cross-validation performance

# Final Analysis

Final Accuracy - 0.9982

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 1.00 | 1.00 | 1.00 | 46436 |
| 1 | 0.86 | 0.81 | 0.84 | 258 |

After applying hyperparameter tuning using Grid Search, the XGBoost classifier achieved excellent performance metrics. For the majority class (label 0), the model reached a perfect precision, recall, and F1-score of 1.00, indicating that it correctly classified all instances without any false positives or false negatives. For the minority class (label 1), which often presents a challenge due to its underrepresentation, the model still performed robustly with a precision of 0.86, recall of 0.81, and F1-score of 0.84. These results highlight the model's improved ability to correctly identify minority class instances after tuning, likely aided by the scale_pos_weight parameter that helped address class imbalance. The high F1-score for class 1 signifies a strong balance between precision and recall, making this model suitable for deployment in real-world scenarios where identifying minority class instances is critical

Lekh Nayak
2023800068 A4

# Thank You

**References**: **Training Dataset** :
https://www.kaggle.com/datasets/ashisparida/hsbc-ml-hackathon-2023