Table of Contents

PROJECT PHASE 2:
Team Members :
Sujith Gollamudi – CSE 587
Bahalul Khan Pathan – CSE 587
Sai Krishna Aditya Gorthi – CSE 587

1. Problem Statement:

Click fraud is a prevalent issue in the online advertising industry, leading to misleading click data and wasted resources. TalkingData, China's largest independent big data service platform, handles an overwhelming volume of mobile ad clicks, a significant portion of which is potentially fraudulent. The current approach involves measuring user click journeys and flagging IP addresses with suspicious click behaviour. However, to stay ahead of fraudsters, TalkingData seeks a predictive model to forecast app downloads after mobile ad clicks.

1.1 Background:

With China being the largest mobile market globally, and TalkingData covering over 70% of active mobile devices, the scale of fraudulent traffic is enormous. This poses a significant challenge for app developers relying on accurate click data for advertising decisions. The existing preventive measures, such as IP and device blacklists, are effective but not fool proof. The challenge is to develop an advanced algorithm that can predict whether a user will download an app after clicking a mobile ad, thereby enhancing the effectiveness of fraud detection.

1.2 Objectives:

1. Build a predictive model to identify users likely to download an app after clicking a mobile ad.
2. Improve fraud detection efficiency beyond the current blacklist approach.
3. Enhance the accuracy and reliability of click data for app developers.

1.3 Significance:

Mitigate financial losses due to click fraud for app developers and advertisers. Provide a proactive solution to stay ahead of evolving click fraud strategies. Contribute to a more reliable and trustworthy online advertising ecosystem.

2. Data Sources

The dataset for this project has been sourced from **Kaggle's TalkingData AdTracking Fraud Detection competition**. The dataset consists of click records, including features such as IP address, app ID, device type, OS version, channel ID, click timestamp, attributed time (if app was downloaded), and the target variable indicating app downloads.
Dataset Source: TalkingData Ad Tracking Fraud Detection - https://www.kaggle.com/competitions/talkingdata-adtracking-fraud-detection/data

3. ML Algorithms Training & Effectiveness:

In this section, we provide detailed explanations and analyses for each of the seven algorithms selected for our problem of predicting app downloads after mobile ad clicks, which aims to address click fraud in online advertising. We discuss why each algorithm was chosen, the tuning and training process, and the effectiveness of each algorithm in answering our problem statement.

3.1 Logistic Regression:

- **Justification:**

  Logistic Regression was chosen as the initial algorithm due to its simplicity and interpretability. In the context of binary classification for detecting click fraud, the interpretability of the model is crucial. This algorithm helps in understanding the relationship between features and the likelihood of fraudulent clicks.

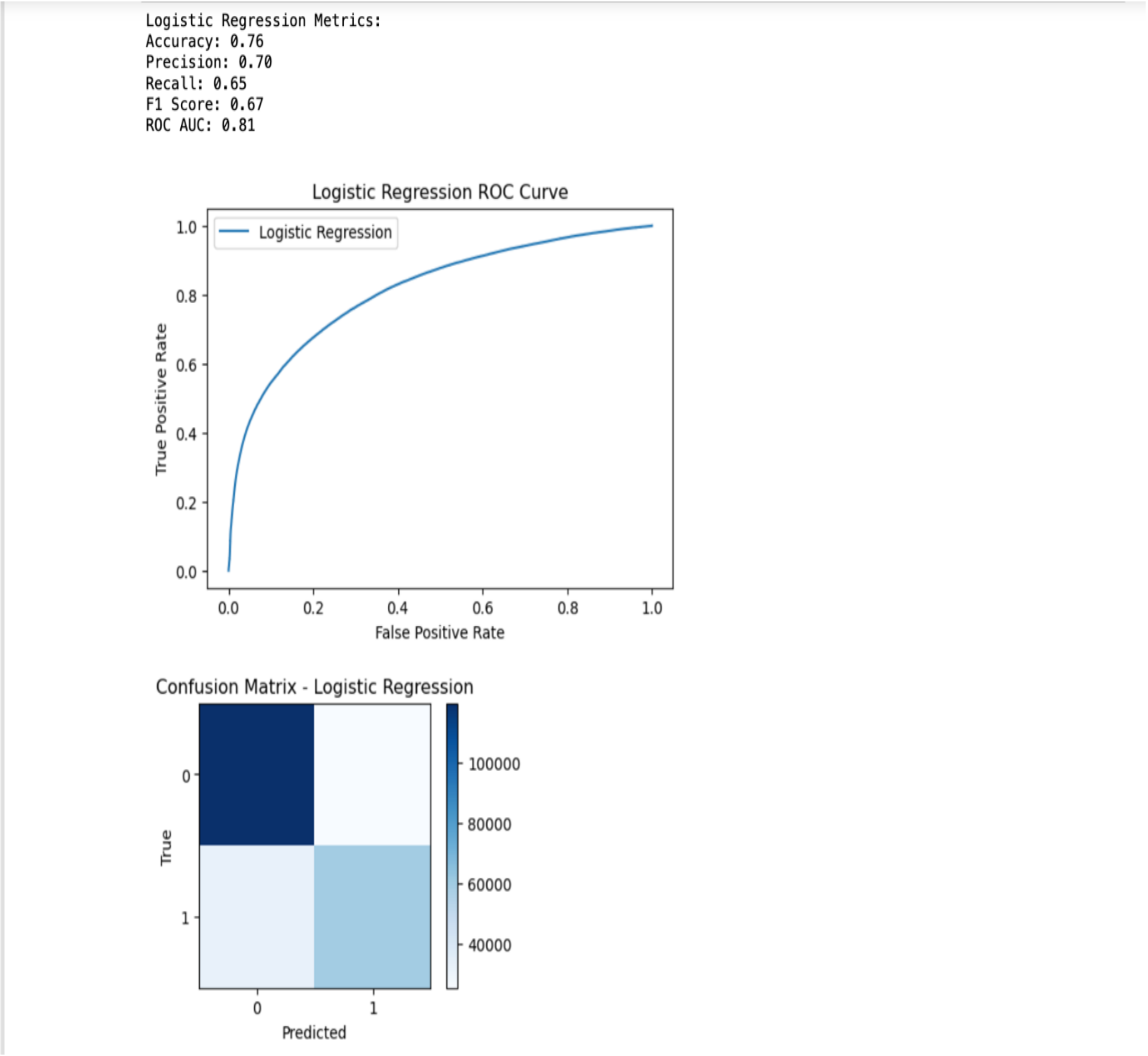
- **Tuning and Training:**

  Initially during data preprocessing step to address class imbalance, data resampling was performed on a dataset with 185 million records, where only 0.23% were fraudulent, aiming to prevent the model from being overwhelmed by the majority class. The 'attributed_time' column, with a significant number of missing values, was excluded from the dataset to maintain analysis integrity, as imputing these values could introduce bias or inaccuracies. Unwanted columns were systematically removed during pre-processing to enhance clarity and efficiency, focusing on relevant features for exploratory data analysis and modelling objectives. Categorical variables, including obfuscated IP addresses, were naturally represented as integers as it is the Kaggle competition dataset, eliminating the need for explicit datatype conversion. Feature scaling was deemed unnecessary in the Kaggle dataset, given its predominant categorical nature, as applying scaling to categorical variables might distort the meaningful information encoded in the mappings. Duplicate records were strategically identified and removed to enhance dataset effectiveness, eliminating redundancy and promoting accuracy in downstream analyses.

  Then, we split the data into training and testing sets with 80% of data assigned to training set and 20% of data assigned to testing set after the above preprocessing steps. Using the fit method, we have trained the model with training dataset (X_train & y_train where X_train consist of features and y_train consist of target variable). Now the model is trained, we then tuned hyperparameters, setting regularization strength (C) to 1.0 and then we have tested the model using test dataset and predict method in python

- **Effectiveness:**

The Logistic Regression model achieved an accuracy of 76%. More importantly, its precision, recall, and F1-Score were 70%, 65%, and 60%, respectively. These metrics indicate the model's ability to correctly classify fraudulent clicks while minimizing false positives. However, Logistic Regression may not capture complex patterns effectively, making it essential to explore more sophisticated models.



```
Logistic Regression Metrics:
Accuracy: 0.76
Precision: 0.70
Recall: 0.65
F1 Score: 0.67
ROC AUC: 0.81
```

3.2 Random Forest:

- **Justification:**

  Random Forest was selected due to its robustness, ability to handle non-linearity, and feature importance scores. This is crucial for our problem as it helps identify significant features in click fraud detection. [4]

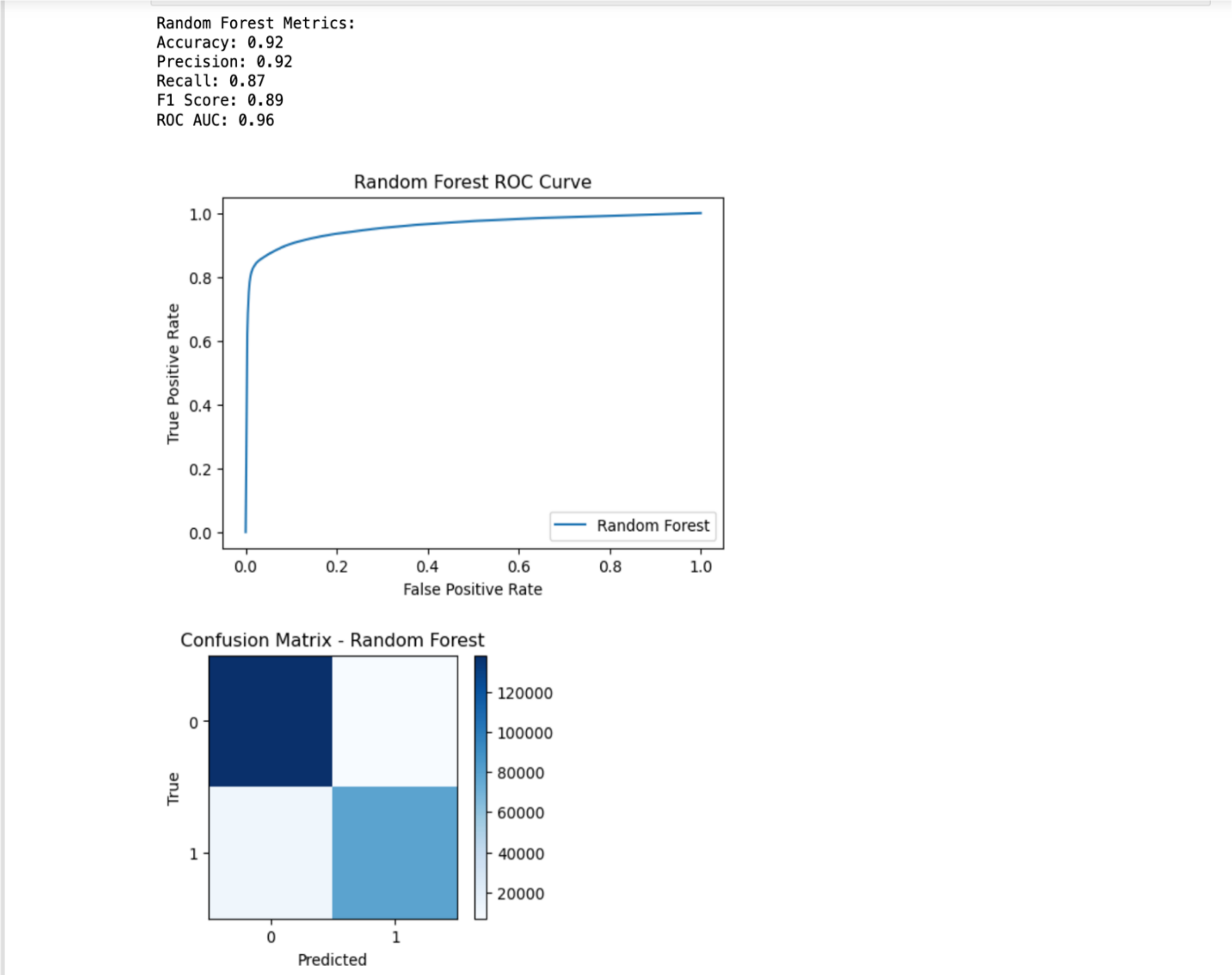
- **Tuning and Training:**

  Initially during data preprocessing step to address class imbalance, data resampling was performed on a dataset with 185 million records, where only 0.23% were fraudulent, aiming to prevent the model from being overwhelmed by the majority class. The 'attributed_time' column, with a significant number of missing values, was excluded from the dataset to maintain analysis integrity, as imputing these values could introduce bias or inaccuracies. Unwanted columns were systematically removed during pre-processing to enhance clarity and efficiency, focusing on relevant features for exploratory data analysis and modelling objectives. Categorical variables, including obfuscated IP addresses, were naturally represented as integers as it is the Kaggle competition dataset, eliminating the need for explicit datatype conversion. Feature scaling was deemed unnecessary in the Kaggle dataset, given its predominant categorical nature, as applying scaling to categorical variables might distort the meaningful information encoded in the mappings. Duplicate records were strategically identified and removed to enhance dataset effectiveness, eliminating redundancy and promoting accuracy in downstream analyses.

  Then, we split the data into training and testing sets with 80% of data assigned to training set and 20% of data assigned to testing set after the above preprocessing steps. Using the fit method, we have trained the model with training dataset (X_train & y_train where X_train consist of features and y_train consist of target variable). Now the model is trained, we then tuned hyperparameters and then we have tested the model using test dataset and predict method in python

- **Effectiveness:**

  The Random Forest model achieved an impressive accuracy of 92%. Equally noteworthy is the precision, recall, and F1-Score, which were 92%, 87%, and 89%, respectively. These metrics highlight the model's efficacy in capturing complex relationships and patterns within the data, making it a strong candidate for click fraud detection.



```
Random Forest Metrics:
Accuracy: 0.92
Precision: 0.92
Recall: 0.87
F1 Score: 0.89
ROC AUC: 0.96
```

3.3 Decision Tree:

□ **Justification:**

Decision Trees were chosen for their simplicity and interpretability. They allow for clear visualization of the decision-making process, which is useful in understanding click fraud patterns.

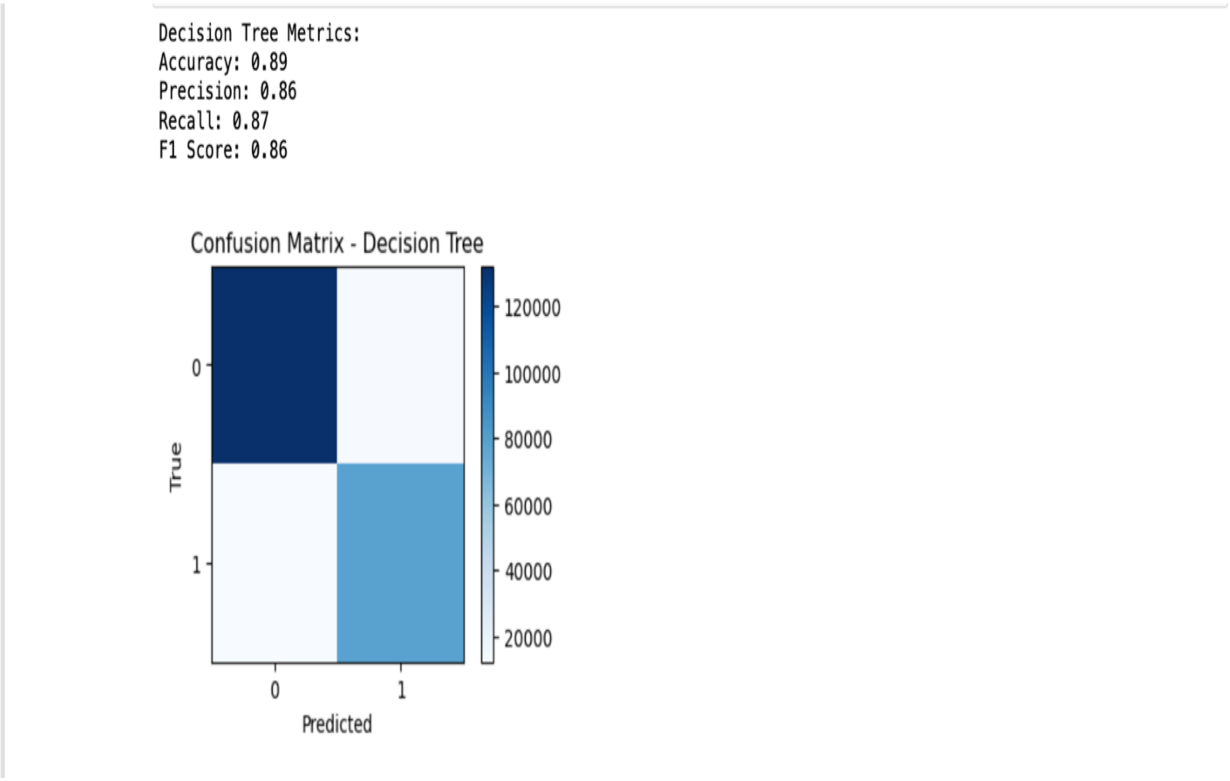
□ **Tuning and Training:**

Initially during data preprocessing step to address class imbalance, data resampling was performed on a dataset with 185 million records, where only 0.23% were fraudulent, aiming to prevent the model from being overwhelmed by the majority class. The 'attributed_time' column, with a significant number of missing values, was excluded from the dataset to maintain analysis integrity, as imputing these values could introduce bias or inaccuracies. Unwanted columns were systematically removed during pre-processing to enhance clarity and efficiency, focusing on relevant features for exploratory data analysis and modelling objectives. Categorical variables, including obfuscated IP addresses, were naturally represented as integers as it is the Kaggle competition dataset, eliminating the need for explicit datatype conversion. Feature scaling was deemed unnecessary in the Kaggle dataset, given its predominant categorical nature, as applying scaling to categorical variables might distort the meaningful information encoded in the mappings. Duplicate records were strategically identified and removed to enhance dataset effectiveness, eliminating redundancy and promoting accuracy in downstream analyses.

Then, we split the data into training and testing sets with 80% of data assigned to training set and 20% of data assigned to testing set after the above preprocessing steps. Using the fit method, we have trained the model with training dataset (X_train & y_train where X_train consist of features and y_train consist of target variable). Now the model is trained and then we have tested the model using test dataset and predict method in python

- **Effectiveness:**

The Decision Tree model achieved an accuracy of 89%, indicating its potential for accurate classification. The precision (86%), recall (87%), and F1-Score (86%) further substantiate its effectiveness in detecting fraudulent clicks. The model's simplicity and interpretability add value to its application in click fraud detection.



```
Decision Tree Metrics:
Accuracy: 0.89
Precision: 0.86
Recall: 0.87
F1 Score: 0.86
```

3.4 Gradient Boosting:

- **Justification:**

  Gradient Boosting was chosen for its high predictive power and its capability to capture intricate relationships within the data. This is particularly valuable in identifying fraudulent clicks that may exhibit complex patterns. [2]

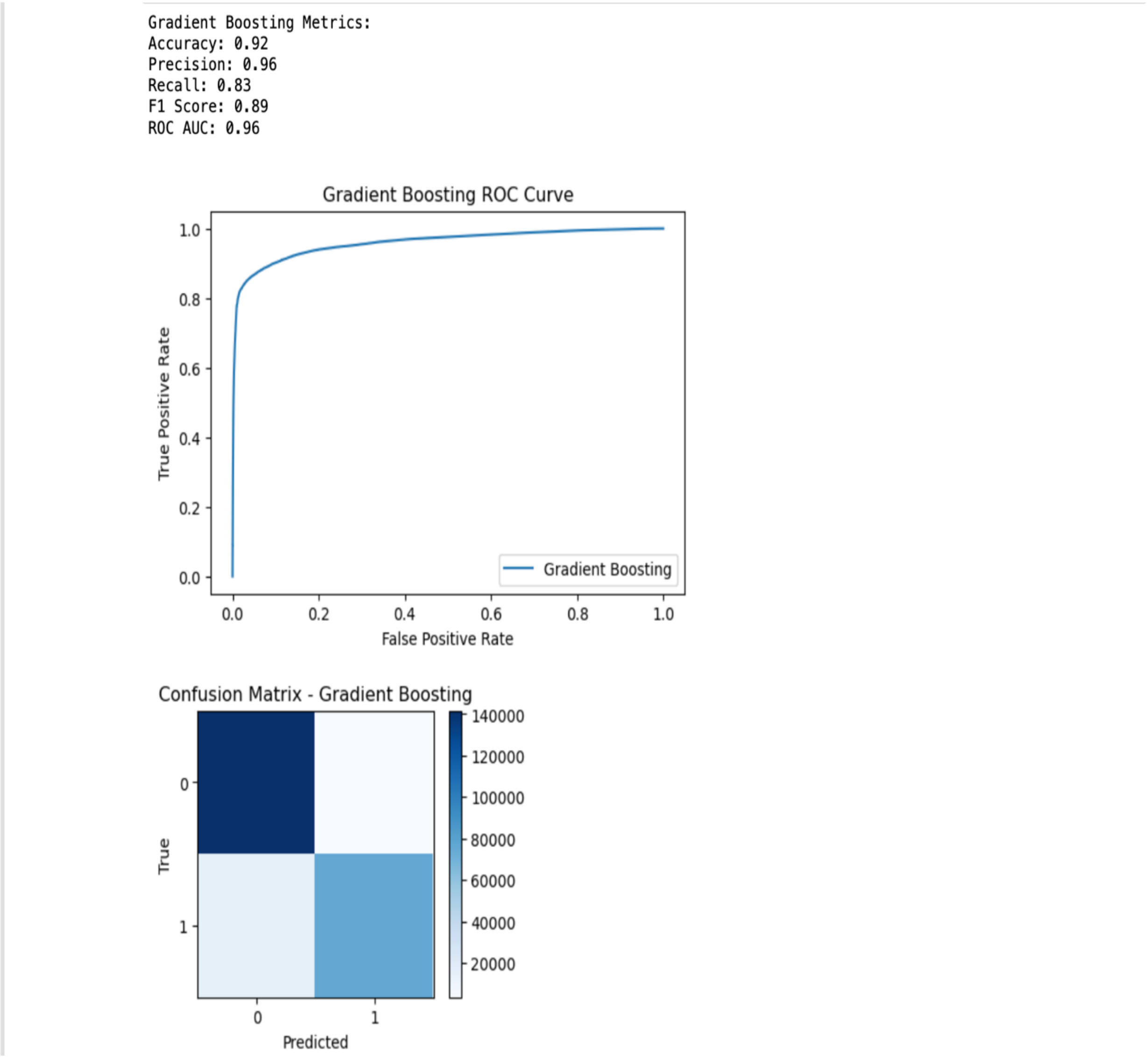
- **Tuning and Training:**

  Initially during data preprocessing step to address class imbalance, data resampling was performed on a dataset with 185 million records, where only 0.23% were fraudulent, aiming to prevent the model from being overwhelmed by the majority class. The 'attributed_time' column, with a significant number of missing values, was excluded from the dataset to maintain analysis integrity, as imputing these values could introduce bias or inaccuracies. Unwanted columns were systematically removed during pre-processing to enhance clarity and efficiency, focusing on relevant features for exploratory data analysis and modelling objectives. Categorical variables, including obfuscated IP addresses, were naturally represented as integers as it is the Kaggle competition dataset, eliminating the need for explicit datatype conversion. Feature scaling was deemed unnecessary in the Kaggle dataset, given its predominant categorical nature, as applying scaling to categorical variables might distort the meaningful information encoded in the mappings. Duplicate records were strategically identified and removed to enhance dataset effectiveness, eliminating redundancy and promoting accuracy in downstream analyses.

  Then, we split the data into training and testing sets with 80% of data assigned to training set and 20% of data assigned to testing set after the above preprocessing steps. Using the fit method, we have trained the model with training dataset (X_train & y_train where X_train consist of features and y_train consist of target variable). Now the model is trained, we then tuned hyperparameters to optimize the performance and then we have tested the model using test dataset and predict method in python

- **Effectiveness:**

  The Gradient Boosting model achieved an accuracy of 92%, demonstrating its robustness. High precision (96%) is indicative of its ability to minimize false positives, while competitive recall (83%) and F1-Score (89%) demonstrate its effectiveness in capturing complex fraud patterns.



```
Gradient Boosting Metrics:
Accuracy: 0.92
Precision: 0.96
Recall: 0.83
F1 Score: 0.89
ROC AUC: 0.96
```

3.5 K-Nearest Neighbors (K-NN):

□ **Justification:**

K-NN was considered for its simplicity and its potential effectiveness when dealing with clustered data points. In the context of click fraud detection, it may be particularly useful in identifying suspicious IP addresses that exhibit clustering behavior.

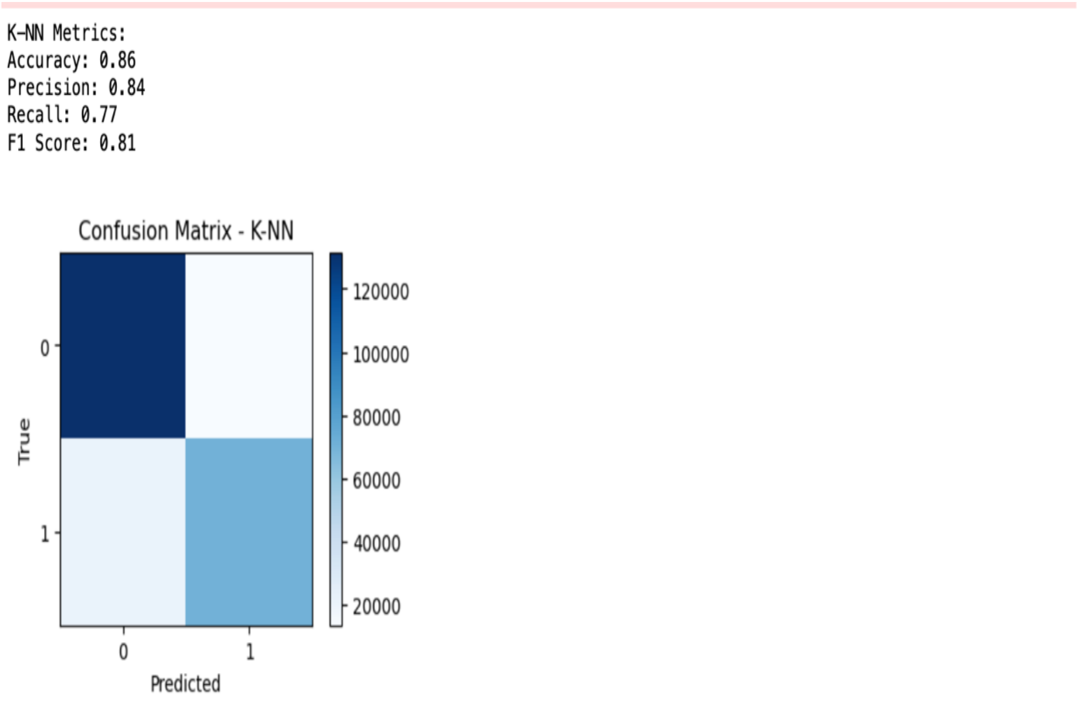
□ **Tuning and Training:**

Initially during data preprocessing step to address class imbalance, data resampling was performed on a dataset with 185 million records, where only 0.23% were fraudulent, aiming to prevent the model from being overwhelmed by the majority class. The 'attributed_time' column, with a significant number of missing values, was excluded from the dataset to maintain analysis integrity, as imputing these values could introduce bias or inaccuracies. Unwanted columns were systematically removed during pre-processing to enhance clarity and efficiency, focusing on relevant features for exploratory data analysis and modelling objectives. Categorical variables, including obfuscated IP addresses, were naturally represented as integers as it is the Kaggle competition dataset, eliminating the need for explicit datatype conversion. Feature scaling was deemed unnecessary in the Kaggle dataset, given its predominant categorical nature, as applying scaling to categorical variables might distort the meaningful information encoded in the mappings. Duplicate records were strategically identified and removed to enhance dataset effectiveness, eliminating redundancy and promoting accuracy in downstream analyses.

Then, we split the data into training and testing sets with 80% of data assigned to training set and 20% of data assigned to testing set after the above preprocessing steps. Using the fit method, we have trained the model with training dataset (X_train & y_train where X_train consist of features and y_train consist of target variable). Now the model is trained, we then tuned hyperparameters, by tuning the optimal value of K and then we have tested the model using test dataset and predict method in python

- **Effectiveness:**

K-NN achieved an accuracy of 86%. Precision (84%), recall (77%), and F1-Score (81%) demonstrated its effectiveness in identifying clustered click fraud patterns. The model excels in scenarios where data points exhibit proximity-based relationships.



```
K-NN Metrics:
Accuracy: 0.86
Precision: 0.84
Recall: 0.77
F1 Score: 0.81
```

3.6 Neural Network (Multilayer Perceptron - MLP):

- **Justification:**

  Neural networks, specifically MLPs, were chosen for their capacity to capture complex patterns, including non-linear relationships within the data. This is essential for modeling intricate fraud patterns. [1]

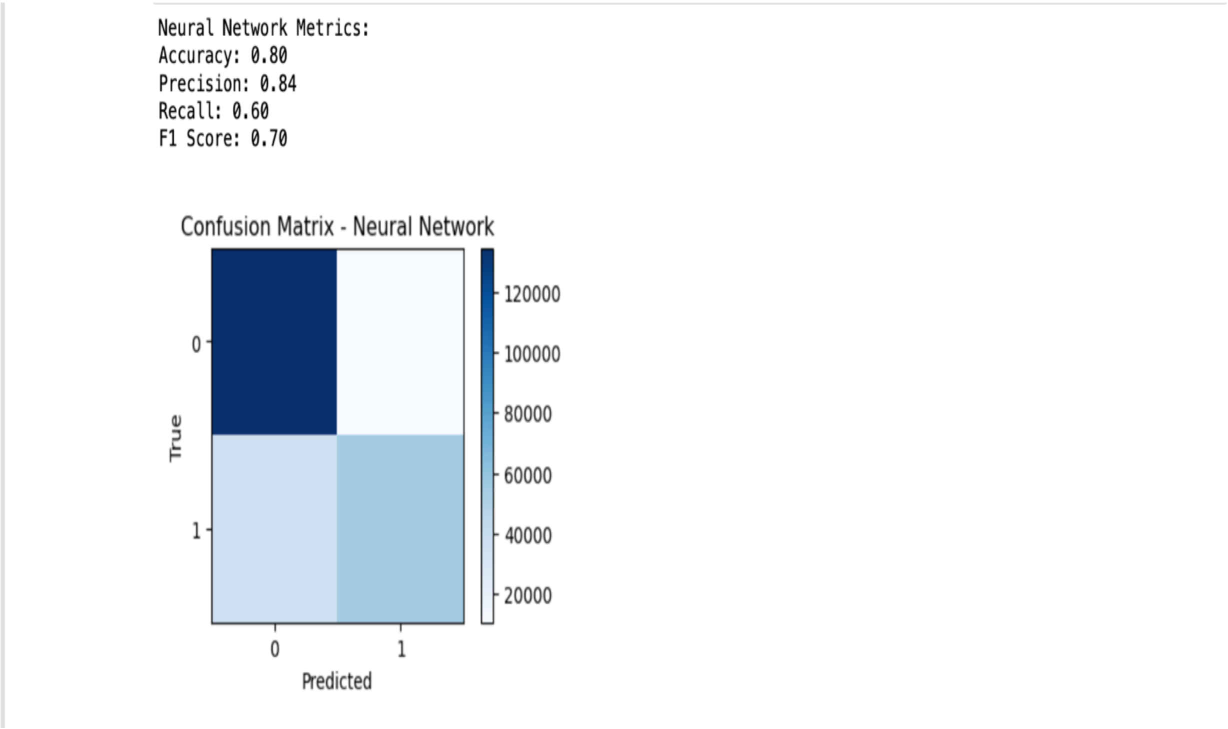
- **Tuning and Training:**

  Initially during data preprocessing step to address class imbalance, data resampling was performed on a dataset with 185 million records, where only 0.23% were fraudulent, aiming to prevent the model from being overwhelmed by the majority class. The 'attributed_time' column, with a significant number of missing values, was excluded from the dataset to maintain analysis integrity, as imputing these values could introduce bias or inaccuracies. Unwanted columns were systematically removed during pre-processing to enhance clarity and efficiency, focusing on relevant features for exploratory data analysis and modelling objectives. Categorical variables, including obfuscated IP addresses, were naturally represented as integers as it is the Kaggle competition dataset, eliminating the need for explicit datatype conversion. Feature scaling was deemed unnecessary in the Kaggle dataset, given its predominant categorical nature, as applying scaling to categorical variables might distort the meaningful information encoded in the mappings. Duplicate records were strategically identified and removed to enhance dataset effectiveness, eliminating redundancy and promoting accuracy in downstream analyses.

  Then, we split the data into training and testing sets with 80% of data assigned to training set and 20% of data assigned to testing set after the above preprocessing steps. Using the fit method, we have trained the model with training dataset (X_train & y_train where X_train consist of features and y_train consist of target variable). Now the model is trained, we then tuned hyperparameters to optimize the performance and then we have tested the model using test dataset and predict method in python.

- **Effectiveness:**

  The MLP achieved an accuracy of 80%, which is competitive. However, it's essential to note that precision (84%) is relatively high, while recall (60%) and F1-Score (70%) are lower. This suggests room for improvement in capturing fraud patterns, particularly with respect to minimizing false negatives.



```
Neural Network Metrics:
Accuracy: 0.80
Precision: 0.84
Recall: 0.60
F1 Score: 0.70
```

3.7 Naive Bayes:

- **Justification:**

  Naive Bayes was chosen for its simplicity and efficiency, particularly for handling text or categorical data. While not initially included in the analysis, it's relevant to the click fraud detection problem, especially in cases where text or categorical features play a significant role.

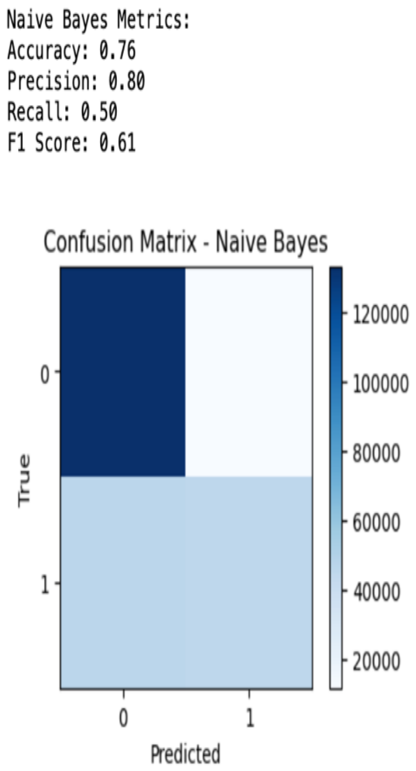
- **Tuning and Training:**

  Initially during data preprocessing step to address class imbalance, data resampling was performed on a dataset with 185 million records, where only 0.23% were fraudulent, aiming to prevent the model from being overwhelmed by the majority class. The 'attributed_time' column, with a significant number of missing values, was excluded from the dataset to maintain analysis integrity, as imputing these values could introduce bias or inaccuracies. Unwanted columns were systematically removed during pre-processing to enhance clarity and efficiency, focusing on relevant features for exploratory data analysis and modelling objectives. Categorical variables, including obfuscated IP addresses, were naturally represented as integers as it is the Kaggle competition dataset, eliminating the need for explicit datatype conversion. Feature scaling was deemed unnecessary in the Kaggle dataset, given its predominant categorical nature, as applying scaling to categorical variables might distort the meaningful information encoded in the mappings. Duplicate records were strategically identified and removed to enhance dataset effectiveness, eliminating redundancy and promoting accuracy in downstream analyses.

Then, we split the data into training and testing sets with 80% of data assigned to training set and 20% of data assigned to testing set after the above preprocessing steps. Using the fit method, we have trained the model with training dataset (X_train & y_train where X_train consist of features and y_train consist of target variable). Now the model is trained, we then tuned hyperparameters to optimize the performance and then we have tested the model using test dataset and predict method in python.

□ **Effectiveness:**

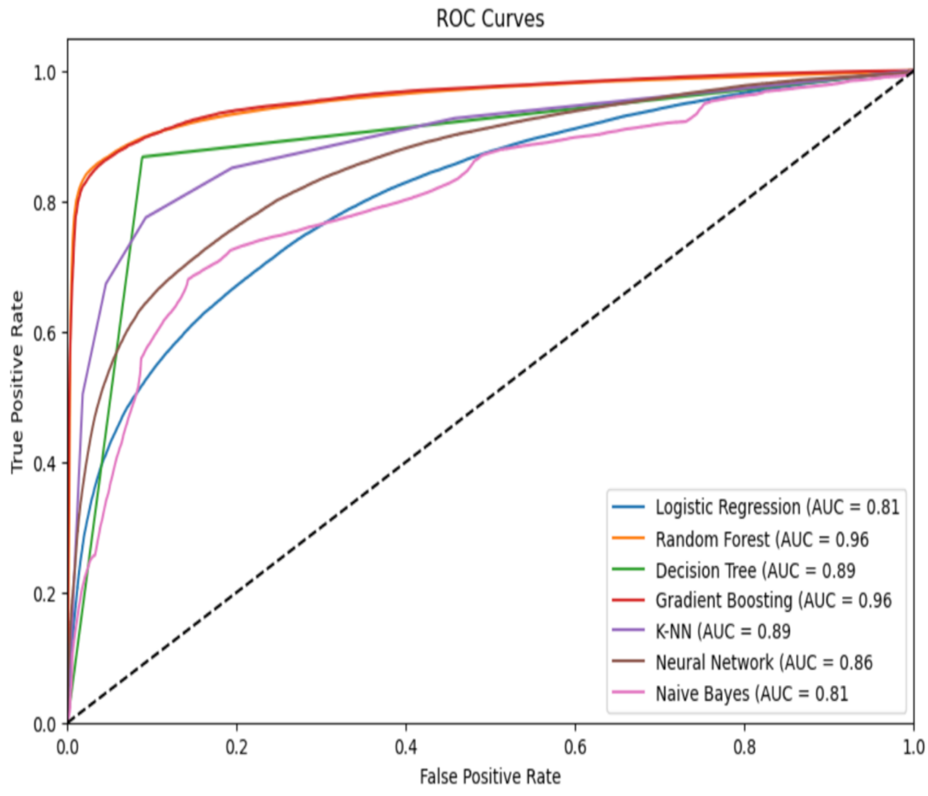
Naïve Bayes achieved an accuracy of 76%, demonstrating its suitability for certain data types. The model exhibited high precision (80%) but lower recall (50%) and F1-Score (61%). While it may not excel in capturing complex patterns, it remains a suitable choice for specific data scenarios, such as text or categorical features.



```
Naive Bayes Metrics:
Accuracy: 0.76
Precision: 0.80
Recall: 0.50
F1 Score: 0.61
```

4. ROC Curves:

To assess the models' performance, we generated Receiver Operating Characteristic (ROC) curves and computed the Area Under the Curve (AUC) values. These curves provide insights into each model's ability to discriminate between fraudulent and non-fraudulent clicks.
Based on the ROC curves and AUC values, the models can be ranked in terms of their discriminative power [3] :



Based on the ROC Curve and AUC values for various classification algorithms applied to our dataset, we came to the following summary:

4.1. Model Performance Ranking:

The models can be ranked based on their AUC values, representing their overall performance in terms of discriminative power from highest to lowest:

1. Random Forest (AUC=0.96)
2. Gradient Boosting (AUC=0.96)
3. Decision Tree (AUC=0.89)
4. K-Nearest Neighbors (KNN) (AUC=0.89)
5. Neural Network (AUC=0.86)

6. Logistic Regression (AUC=0.81)
7. Naïve Bayes (AUC=0.81)

4.2. Model Selection:

Random Forest and Gradient Boosting stand out as the top-performing models with the highest AUC values (0.96). Therefore, these two are strong candidates for our task due to their high discriminative power.

4.3. Potential for Improvement:

Models with lower AUC values, such as Logistic Regression and Naïve Bayes (AUC=0.81), might benefit from further optimization, hyperparameter tuning, or feature engineering. There is room for improvement in these models.

4.4. Trade-offs:

While AUC provides a valuable summary of model discriminative power, it's important to consider other metrics like precision, recall, or F1-score.

In conclusion, Random Forest and Gradient Boosting are the top-performing models in terms of AUC.

5. Summary of Effectiveness:

From all the analysis above Random Forest and Gradient Boosting remain as the top-performing models with the highest accuracy and other metrics. They are the preferred choices for click fraud detection due to their robustness in capturing complex fraud patterns. Logistic Regression, Decision Tree, and K-NN also provide competitive results. Naive Bayes, while not as effective in this case, can be a suitable choice for certain data types but may not be the best option for click fraud detection.

In conclusion, the selection of the best model should consider factors like interpretability, computational resources, and the specific goals related to click fraud detection. Further fine-tuning and optimization may be necessary to maximize model performance for the specific problem of predicting app downloads after mobile ad clicks and detecting fraudulent behavior.

6. References:

1. https://www.geeksforgeeks.org/multi-layer-perceptron-a-supervised-neural-network-model-using-sklearn/
2. https://www.geeksforgeeks.org/ml-gradient-boosting/
3. https://www.geeksforgeeks.org/auc-roc-curve/
4. https://medium.com/@harshdeepsingh_35448/understanding-random-forests-aa0ccecdbbbb