**Team – 5**

**Team Members: Natnael Seyoum and Yonatan Embiza**

**Project Report: Bank Churn Prediction**

**Final Report and Documentation**

**1. Introduction**

**Customer Attrition Prediction in the Banking Sector**

Customer attrition is a critical issue for the banking industry. Retaining customers is significantly more cost-effective than acquiring new ones, making the ability to predict and mitigate customer churn a valuable asset. This project aims to develop robust machine learning models to predict customer attrition using the "BankChurners" dataset.

The project follows a structured workflow from data exploration and preprocessing to model training and evaluation. We employ various techniques to handle class imbalance and test multiple models, including advanced ensemble methods, to identify the best performing model. The comprehensive analysis and findings are documented to provide insights and recommendations.

**2. Data Exploration**

**Exploratory Data Analysis (EDA)**

The "BankChurners" dataset is loaded and examined to understand the underlying data structure and distributions.

Dataset: <https://www.kaggle.com/code/jessintha/credit-card-customer-churn-prediction>

Dataset description, dataset columns representations  
  
**CLIENTNUM**: A unique identifier for each customer

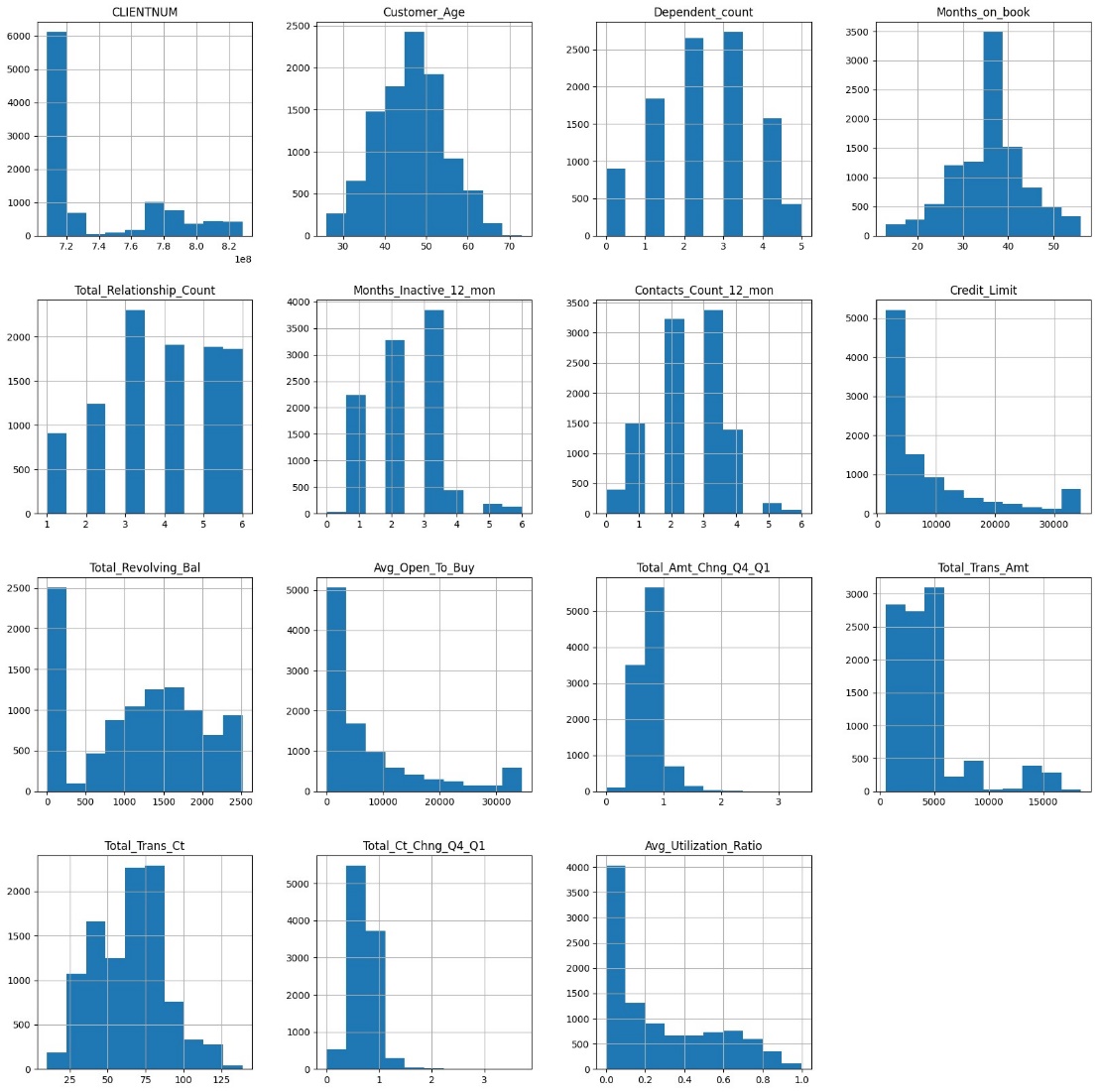
**Attrition\_Flag**: Indicates whether a customer has experienced attrition or not

**Customer\_Age**: Age of each customer  
**Gender**: Gender of each customer  
**Dependent\_count**: No of dependents (e.g., children, elderly relatives) that each customer has  
**Education\_Level**: Education level of each customer  
**Marital\_Status**: Marital status of each customer  
**Income\_Category**: Income category of each customer  
**Card\_Category**: Type of credit card held by each customer(“Blue", "Silver", "Gold", "Platinum")  
**Months\_on\_book**: Number of months that each customer has been a customer of the bank  
**Total\_Relationship\_Count**: Total number of services held by each customer with the bank  
**Months\_Inactive\_12\_mon**: Number of months in the last twelve months during which each customer has been inactive (i.e., not engaged with the bank's services)  
**Contacts\_Count\_12\_mon**: Number of times each customer has been contacted by the bank within the last twelve months  
**Credit\_Limit**: Credit limit assigned to each customer  
**Total\_Revolving\_Bal**: Total revolving balance (i.e., the outstanding balance on a credit card account) for each customer  
**Avg\_Open\_To\_Buy**: Average amount of credit available for each customer to use  
**Total\_Amt\_Chng\_Q4\_Q1**: Percentage change in the total transaction amount from the fourth quarter to the first quarter  
**Total\_Trans\_Amt**: Total transaction amount for each customer  
**Total\_Trans\_Ct**: Total number of transactions for each customer  
**Total\_Ct\_Chng\_Q4\_Q1**: Percentage change in the total number of transactions from the 4th quarter to the 1st quarter  
**Avg\_Utilization\_Ratio**: Average utilization ratio of each customer's credit card

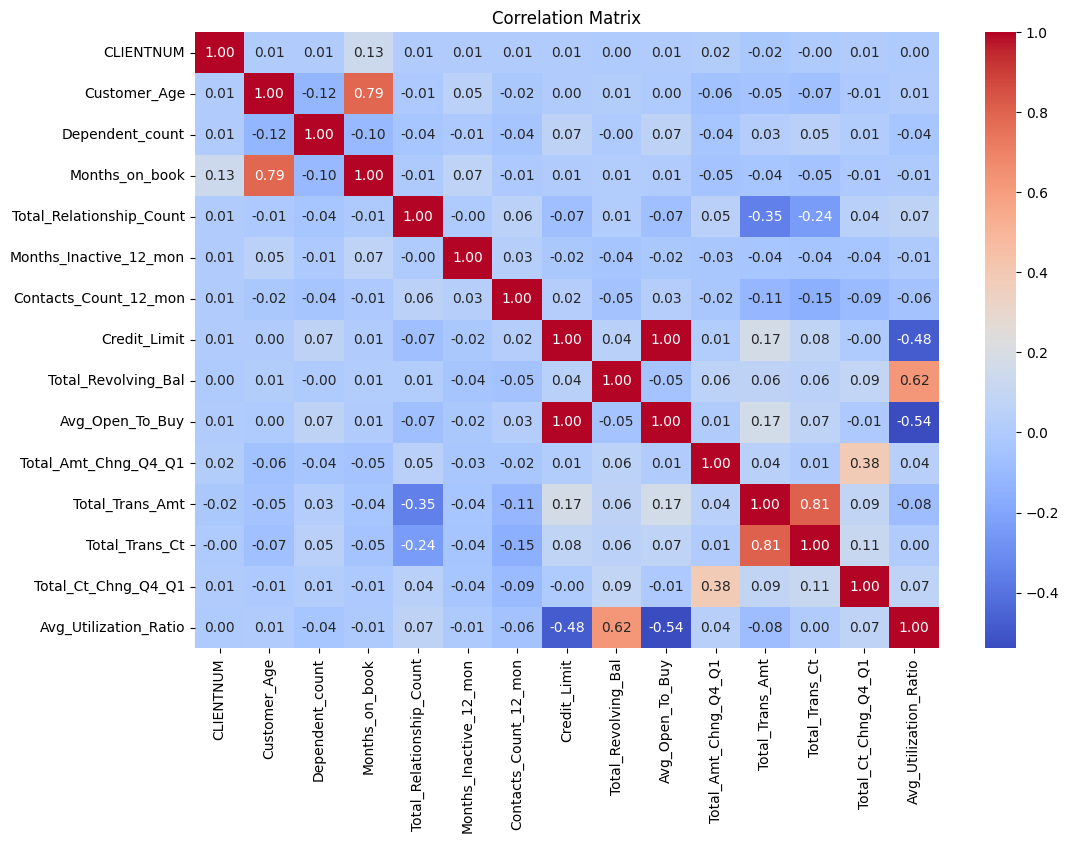
Key steps include:

* **Loading and Inspecting Data**: Initial inspection reveals the dataset contains 23 features and 1 target column ('Attrition\_Flag').
* **Descriptive Statistics**: Summary statistics provide insights into the central tendency, dispersion, and shape of the data distribution.
* **Missing Values**: The dataset is checked for missing or null values, ensuring data completeness.
* **Categorical Features**: The distribution of categorical variables is visualized using bar plots.
* **Numerical Features**: Histograms and box plots are used to explore numerical feature distributions and detect outliers.

**Data distribution in Numerical features before Data Preprocessing**



**Correlation Matrix before preprocessing**

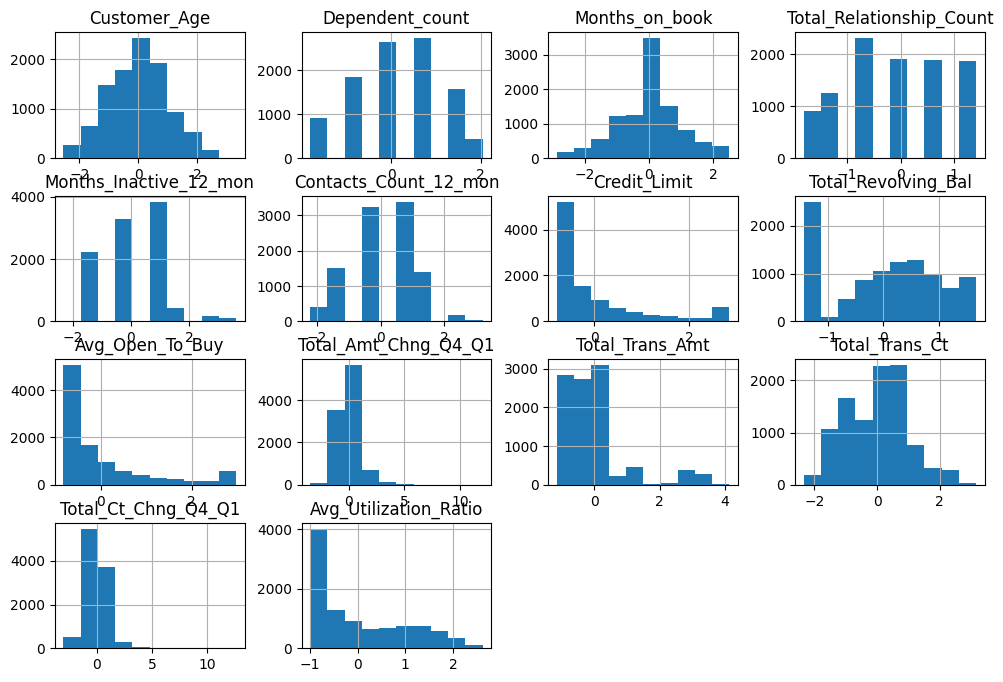


**3. Data Preprocessing**

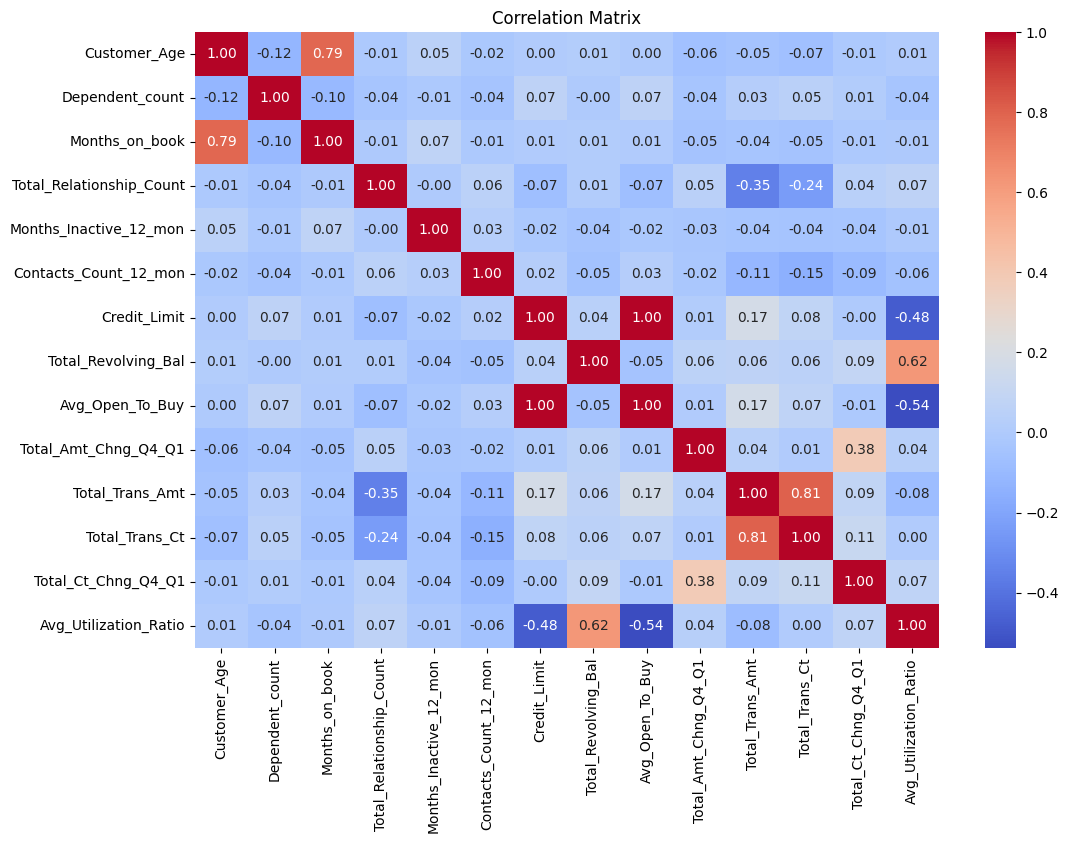
**Data Cleaning and Encoding**

* **Dropping Irrelevant Columns**: The first column, representing customer IDs, is dropped.
* **Encoding Categorical Variables**: Categorical variables are encoded using appropriate techniques. One-Hot Encoding is used for 'Gender' and 'Marital\_Status', while ordinal encoding is applied to 'Education\_Level', 'Income\_Category', and 'Card\_Category'.
* **Class Imbalance Handling**: Techniques such as SMOTE and random undersampling are employed to balance the class distribution.

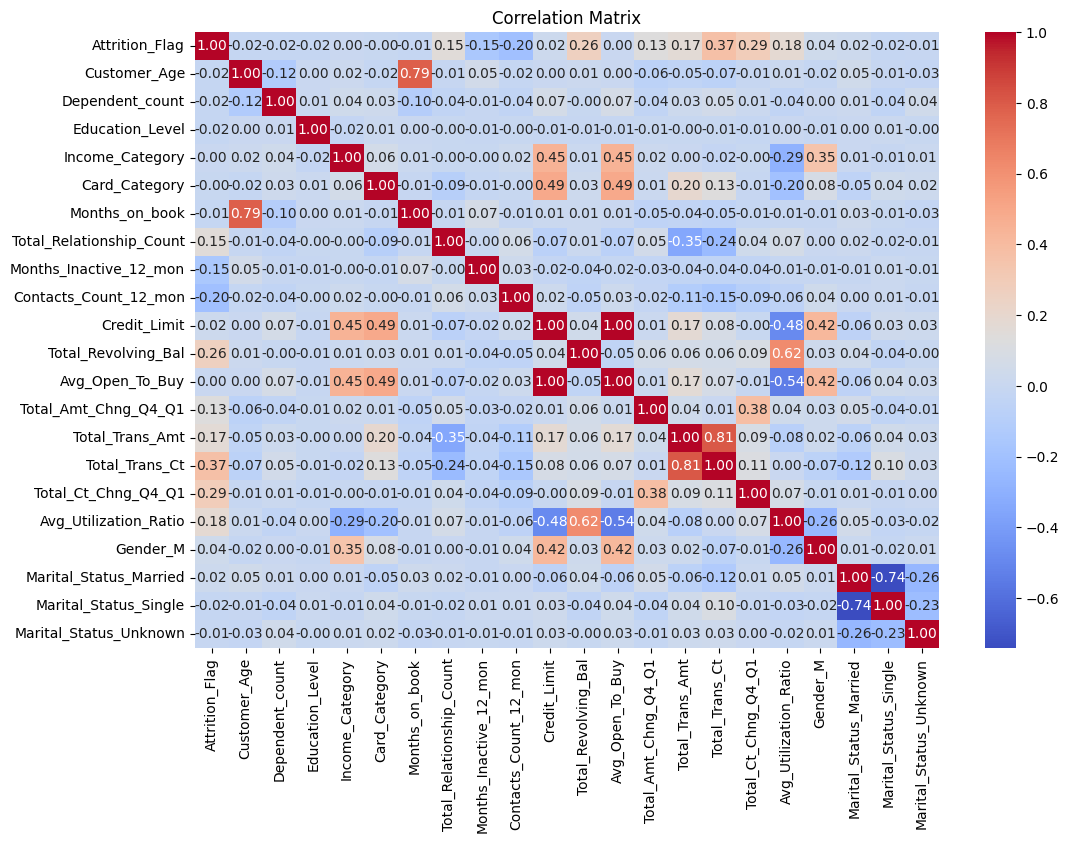
**Data distribution in Numerical features after Data Preprocessing**



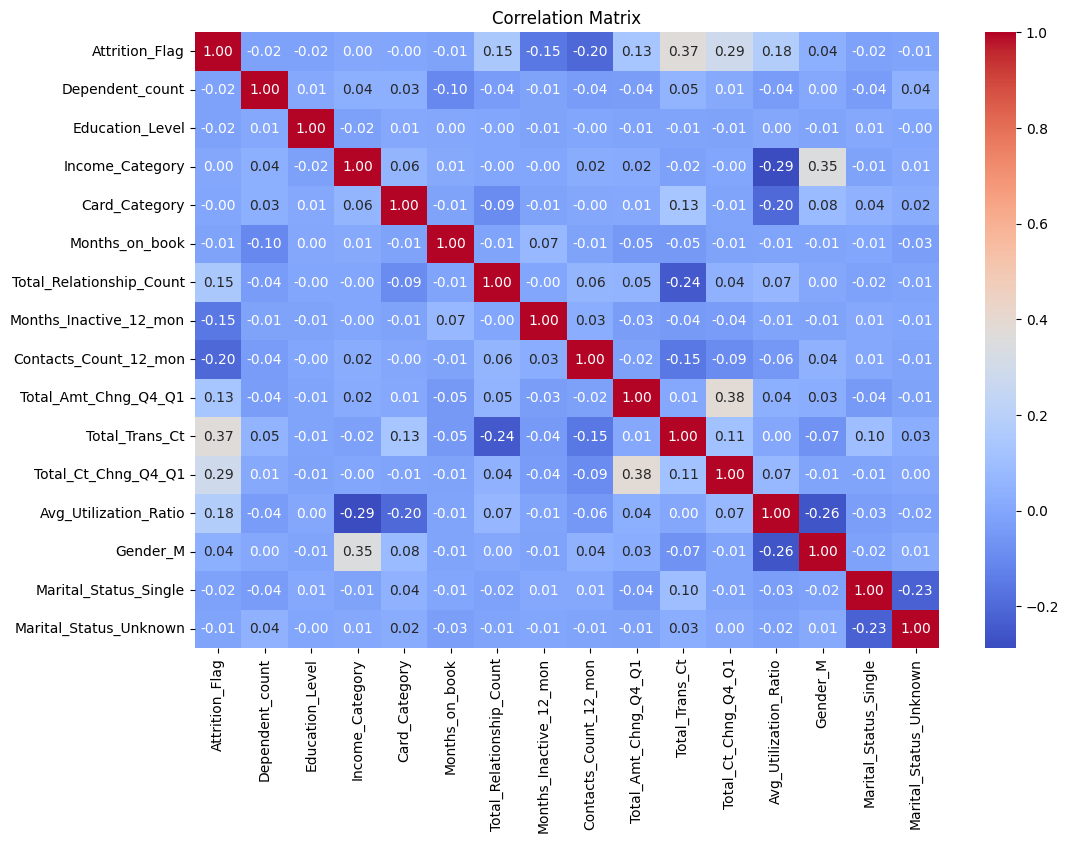
**Correlation Matrix after preprocessing**



**Correlation Matrix after one-hat encoding**



**Correlation Matrix after removing highly correlated features**

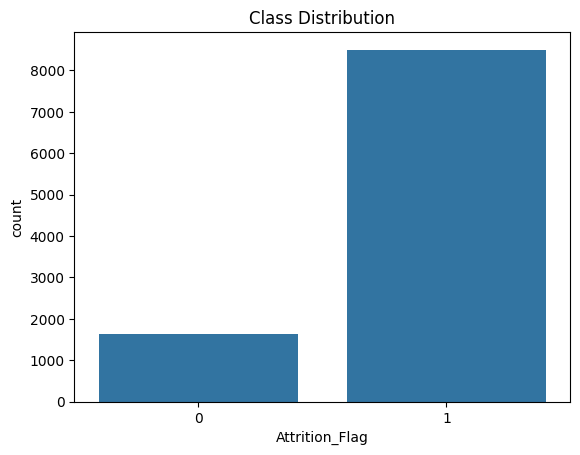


**4. Handling Class Imbalance**

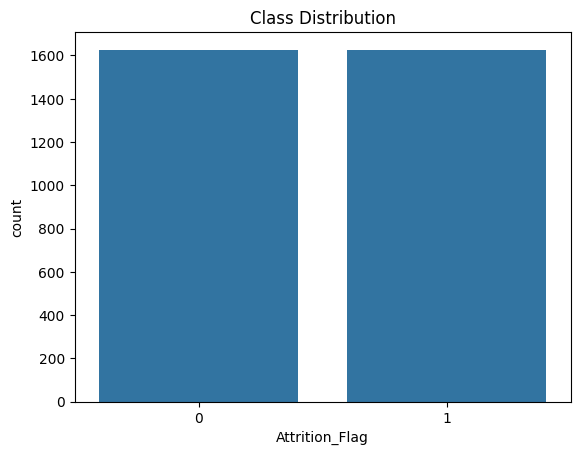
Class imbalance is addressed using various techniques:

* **SMOTE (Synthetic Minority Oversampling Technique)**: Generates synthetic samples for the minority class to balance the dataset.
* **Random Undersampling**: Reduces the majority class samples to balance the class distribution.

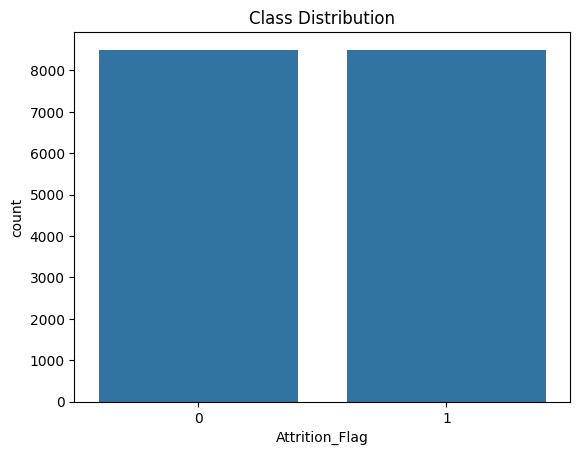
**Original Class Distribution**

****

**Class Distribution after Undersampling**



**Class Distribution after SMOTE**

****

**5. Feature Engineering and Selection**

**Enhancing the Dataset**

* **Feature Creation**: New features are derived from existing ones to provide additional predictive power.
* **Feature Selection**: Correlation analysis and feature importance techniques are used to select the most relevant features, removing highly correlated and redundant features.

**6. Modeling**

**Training and Evaluating Machine Learning Models**

Multiple models are trained and evaluated:

First we trained a baseline model which is the **Logistic Regression** and the following results were recorded:

Performance Metrics:

Baseline Model - Training Score: 0.89

Baseline Model - Testing Score: 0.89

Classification Report (Baseline Model):

precision recall f1-score support

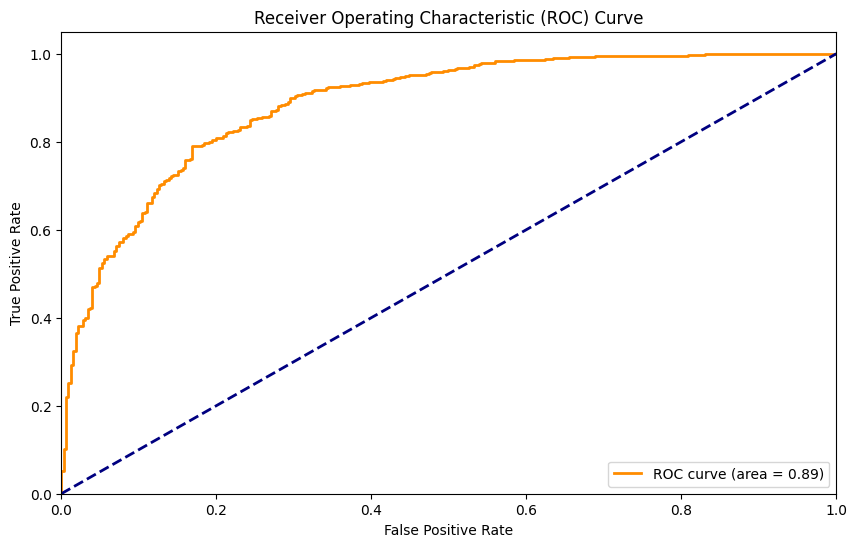
0 0.73 0.50 0.59 325

1 0.91 0.96 0.94 1701

accuracy 0.89 2026

macro avg 0.82 0.73 0.76 2026

weighted avg 0.88 0.89 0.88 2026



For classification, we tried to investigate different types of classifier including deep neural networks.

* **K-Nearest Neighbors**
* **Support Vector Classifier**
* **Naive Bayes**
* **Decision Tree**
* **Random Forest**
* **XGBoost**
* **AdaBoost**
* **Gradient Boosting**
* **CatBoost**
* **LightGBM**
* **Neural Networks**

### Description of Each Model Used

1. **Logistic Regression**
   * **Description**: Logistic Regression is a linear model used for binary classification. It predicts the probability that a given input belongs to a certain class by fitting a logistic function to the data.
   * **Use Case**: Often used as a baseline model due to its simplicity and interpretability.
2. **K-Nearest Neighbors (KNN)**
   * **Description**: KNN is a non-parametric, instance-based learning algorithm. It classifies a sample based on the majority class among its K-nearest neighbors.
   * **Use Case**: Useful for datasets with few attributes and for multi-class classification problems.
3. **Support Vector Classifier (SVC)**
   * **Description**: SVC is a supervised learning model that finds the hyperplane which best separates the classes in the feature space. It works well for both linear and non-linear data.
   * **Use Case**: Effective for high-dimensional spaces and where the number of dimensions exceeds the number of samples.
4. **Gaussian Naive Bayes**
   * **Description**: A probabilistic classifier based on Bayes' Theorem, assuming independence between features. Gaussian Naive Bayes assumes that the features follow a normal distribution.
   * **Use Case**: Suitable for high-dimensional data and when the assumption of feature independence holds.
5. **Decision Tree Classifier**
   * **Description**: A non-parametric model that splits the data into subsets based on the value of input features, forming a tree structure. Each node represents a decision based on a feature.
   * **Use Case**: Useful for capturing non-linear relationships and handling both numerical and categorical data.
6. **Random Forest Classifier**
   * **Description**: An ensemble method that builds multiple decision trees and merges them to get a more accurate and stable prediction. Each tree is trained on a random subset of the data.
   * **Use Case**: Effective for various types of data, including high-dimensional spaces.
7. **XGBoost (Extreme Gradient Boosting)**
   * **Description**: An advanced implementation of gradient boosting designed for performance and speed. It builds trees sequentially, with each tree correcting errors of the previous ones.
   * **Use Case**: Widely used in machine learning competitions and real-world applications.
8. **AdaBoost (Adaptive Boosting)**
   * **Description**: An ensemble method that combines multiple weak classifiers to create a strong classifier. It adjusts the weights of incorrectly classified instances, giving them more importance in subsequent classifiers.
   * **Use Case**: Effective for boosting the performance of weak classifiers.
9. **Gradient Boosting Classifier**
   * **Description**: A sequential ensemble method that builds trees to minimize the residual errors of the previous trees. Each tree in the sequence is fit on the residual errors of the previous trees.
   * **Use Case**: Suitable for various types of prediction problems.
10. **MLPClassifier (Multi-Layer Perceptron)**
    * **Description**: A type of feedforward artificial neural network that consists of multiple layers of nodes, with each node connected to the nodes in the next layer. It uses backpropagation for training.
    * **Use Case**: Suitable for complex datasets with non-linear relationships.
11. **CatBoost (Categorical Boosting)**
    * **Description**: A gradient boosting algorithm that handles categorical features automatically. It uses ordered boosting to deal with overfitting.
    * **Use Case**: Effective for datasets with many categorical features.
12. **LightGBM (Light Gradient Boosting Machine)**
    * **Description**: A gradient boosting framework that uses tree-based learning algorithms, designed for efficiency and scalability. It uses histogram-based algorithms to speed up training.
    * **Use Case**: Suitable for large datasets and high-dimensional data.
13. **Stacking Classifier**
    * **Description**: An ensemble method that combines multiple base classifiers via a meta-classifier. The base classifiers are trained on the original dataset, and the meta-classifier is trained on the predictions of the base classifiers.
    * **Use Case**: Effective when base models are diverse and have complementary strengths.

Model Comparison: Class imbalanced dataset

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **GNB** | **DTree** | **KNN** | **SVC** | **RForest** | **AdaB** | **GB** | **XGB** | **LGBM** | **CatB** |
| **Accuracy** | 0.88 | 0.89 | 0.89 | 0.91 | 0.93 | 0.92 | 0.93 | 0.93 | 0.93 | 0.94 |
| **Precision** | 0.91 | 0.94 | 0.91 | 0.92 | 0.94 | 0.94 | 0.94 | 0.95 | 0.95 | 0.95 |
| **Recall** | 0.95 | 0.93 | 0.96 | 0.98 | 0.98 | 0.98 | 0.98 | 0.97 | 0.97 | 0.98 |
| **F1 Score** | 0.93 | 0.93 | 0.94 | 0.95 | 0.96 | 0.96 | 0.96 | 0.96 | 0.96 | 0.96 |
| **ROC-AUC** | 0.88 | 0.81 | 0.87 | 0.92 | 0.96 | 0.94 | 0.95 | 0.97 | 0.97 | 0.97 |

Model Comparison: Class balanced dataset, using undersampling

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **GNB** | **DTree** | **KNN** | **SVC** | **RForest** | **AdaB** | **GB** | **XGB** | **LGBM** | **CatB** |
| **Accuracy** | 0.81 | 0.80 | 0.81 | 0.85 | 0.88 | 0.86 | 0.89 | 0.88 | 0.89 | 0.90 |
| **Precision** | 0.78 | 0.78 | 0.78 | 0.82 | 0.84 | 0.82 | 0.86 | 0.85 | 0.86 | 0.86 |
| **Recall** | 0.83 | 0.81 | 0.82 | 0.87 | 0.91 | 0.88 | 0.92 | 0.90 | 0.92 | 0.93 |
| **F1 Score** | 0.80 | 0.79 | 0.80 | 0.84 | 0.88 | 0.85 | 0.89 | 0.87 | 0.89 | 0.89 |
| **ROC-AUC** | 0.88 | 0.80 | 0.90 | 0.92 | 0.95 | 0.93 | 0.96 | 0.96 | 0.96 | 0.97 |

Model Comparison: Class balanced dataset, using SMOTE

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **GNB** | **DTree** | **KNN** | **SVC** | **RForest** | **AdaB** | **GB** | **XGB** | **LGBM** | **CatB** |
| **Accuracy** | 0.84 | 0.92 | 0.91 | 0.91 | 0.97 | 0.92 | 0.96 | 0.97 | 0.96 | 0.97 |
| **Precision** | 0.84 | 0.92 | 0.99 | 0.92 | 0.97 | 0.92 | 0.95 | 0.96 | 0.96 | 0.96 |
| **Recall** | 0.83 | 0.92 | 0.83 | 0.89 | 0.96 | 0.92 | 0.96 | 0.97 | 0.97 | 0.97 |
| **F1 Score** | 0.84 | 0.92 | 0.90 | 0.90 | 0.97 | 0.92 | 0.96 | 0.97 | 0.96 | 0.97 |
| **ROC-AUC** | 0.91 | 0.92 | 0.97 | 0.97 | 0.99 | 0.98 | 0.99 | 1.00 | 1.00 | 1.00 |

Each model is trained using the balanced dataset and evaluated on the test set. Key metrics such as accuracy, precision, recall, F1 score, and ROC-AUC are computed.

**7. Model Comparison**

**Performance Analysis**

Performance metrics for each model are collected and compared. The results are visualized using bar charts and ROC curves, highlighting the strengths and weaknesses of each model.

**8. Stacking Ensemble**

**Advanced Ensemble Method**

A stacking ensemble model is implemented, combining the predictions of multiple base models using a meta-classifier. The stacking model aims to leverage the strengths of each base model to improve overall performance.

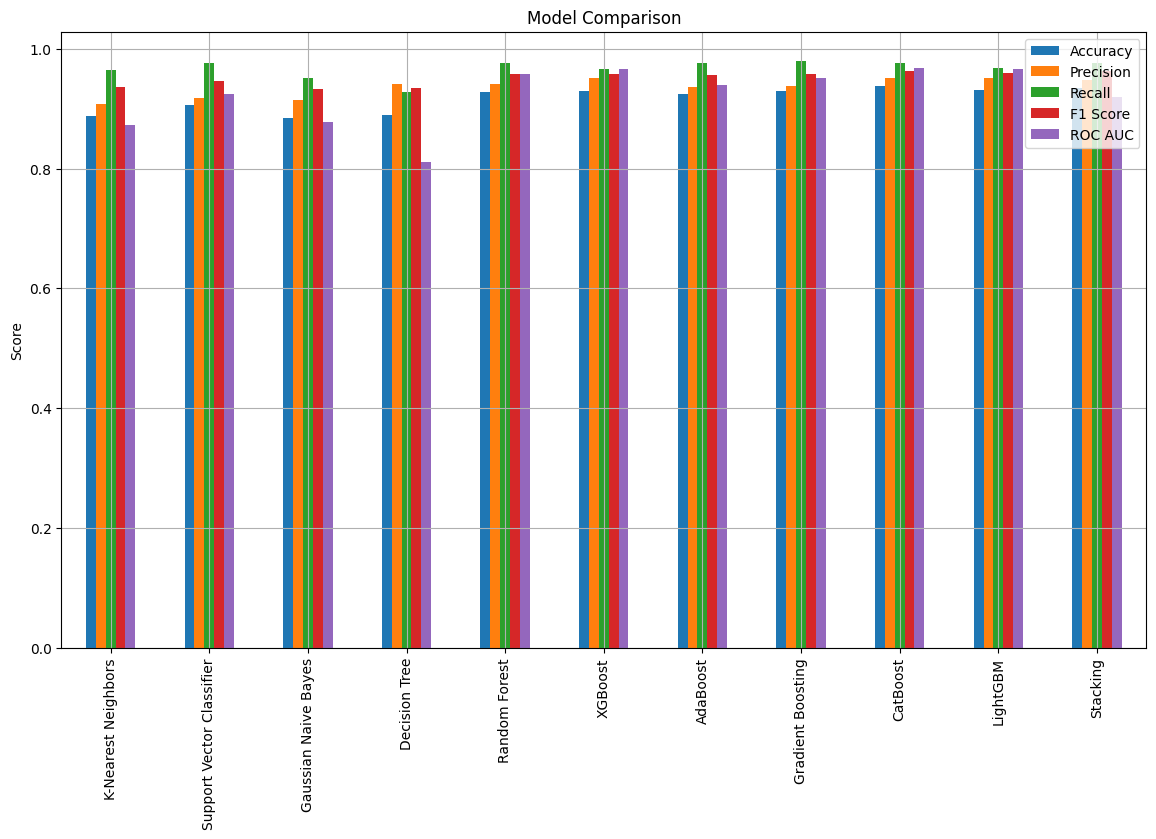
**9. Visualization of Performance Curves**

**Training and Testing Curves**

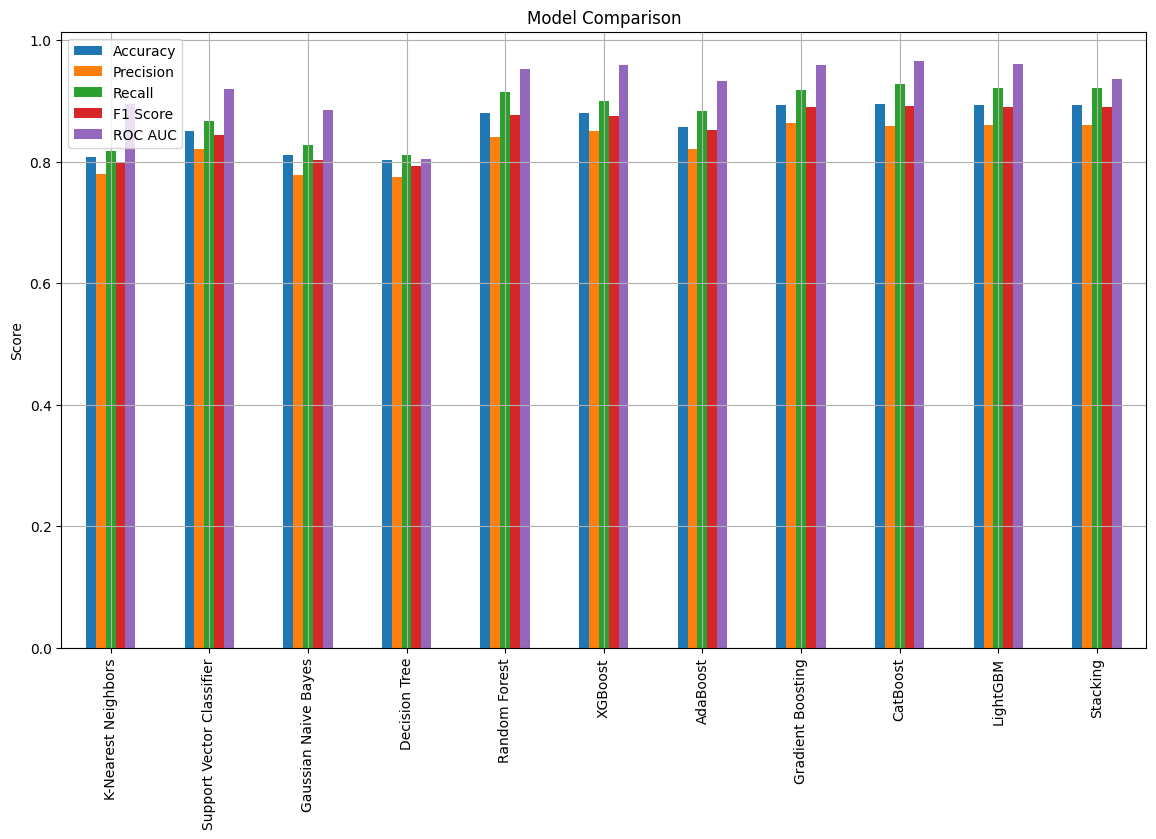
Training and testing accuracy curves, as well as ROC-AUC curves for all models, are plotted to visualize performance and generalization capabilities.

**Bar Chart**

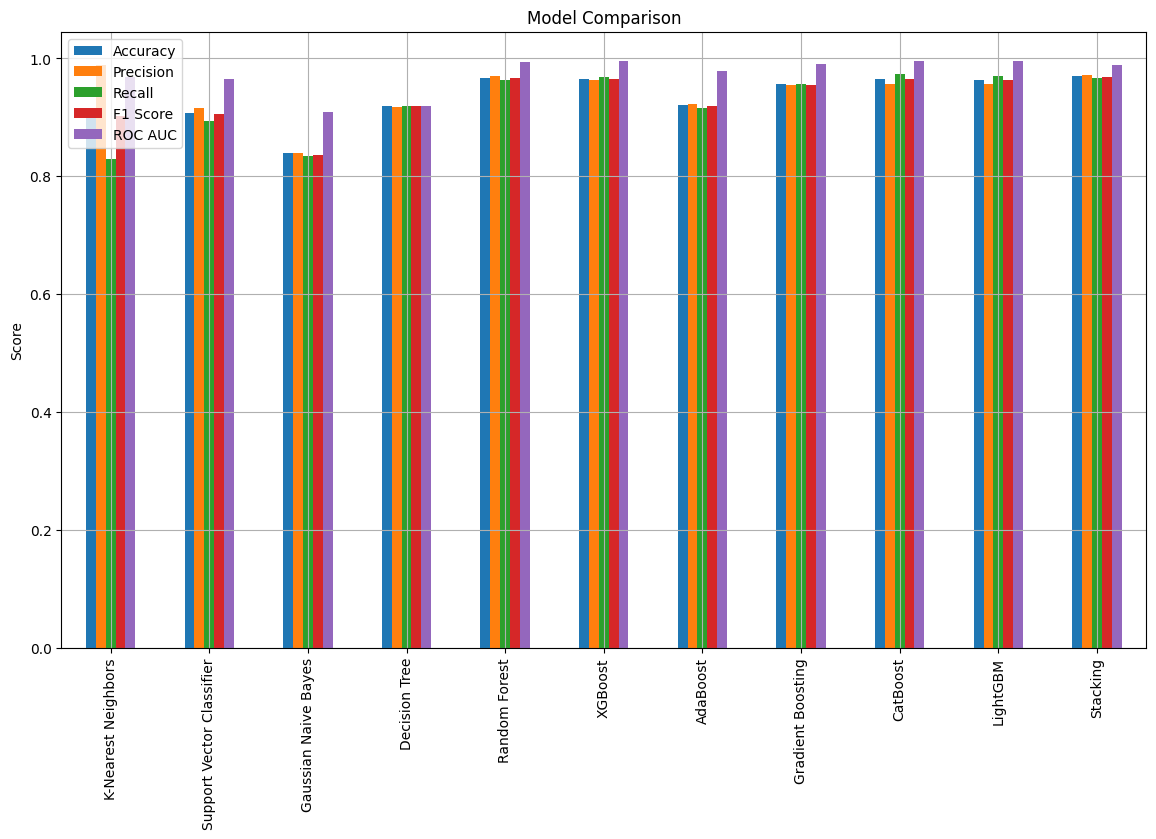
Without taking care the class imbalance



After applying under-sampling to handle the sever class imbalance present in the dataset

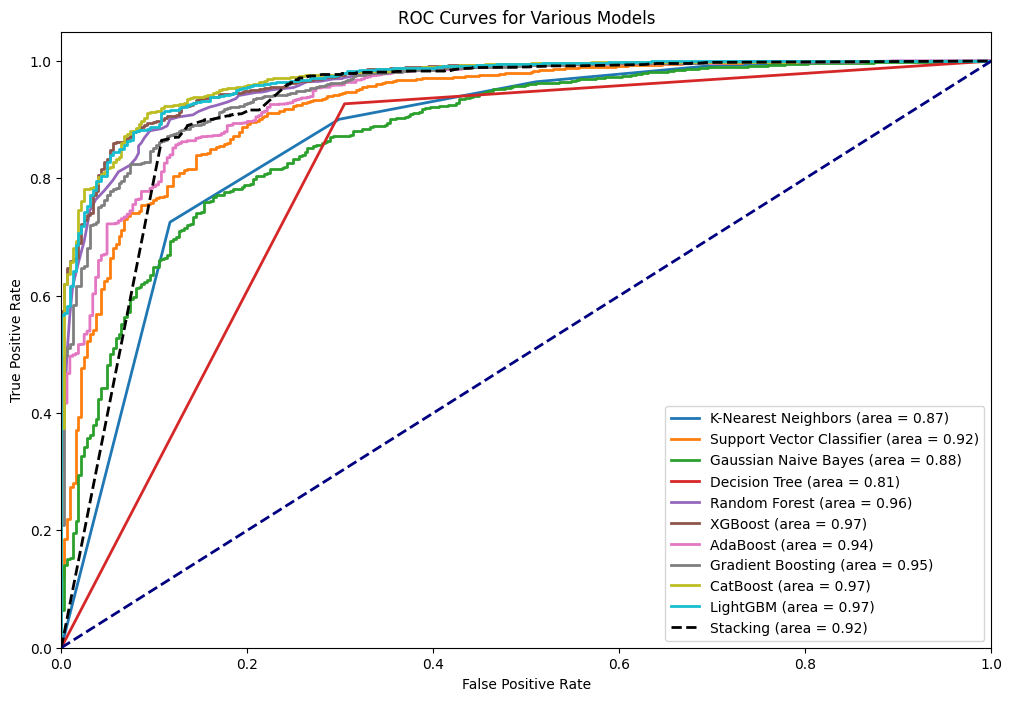


After applying SMOTE to handle the sever class imbalance present in the dataset

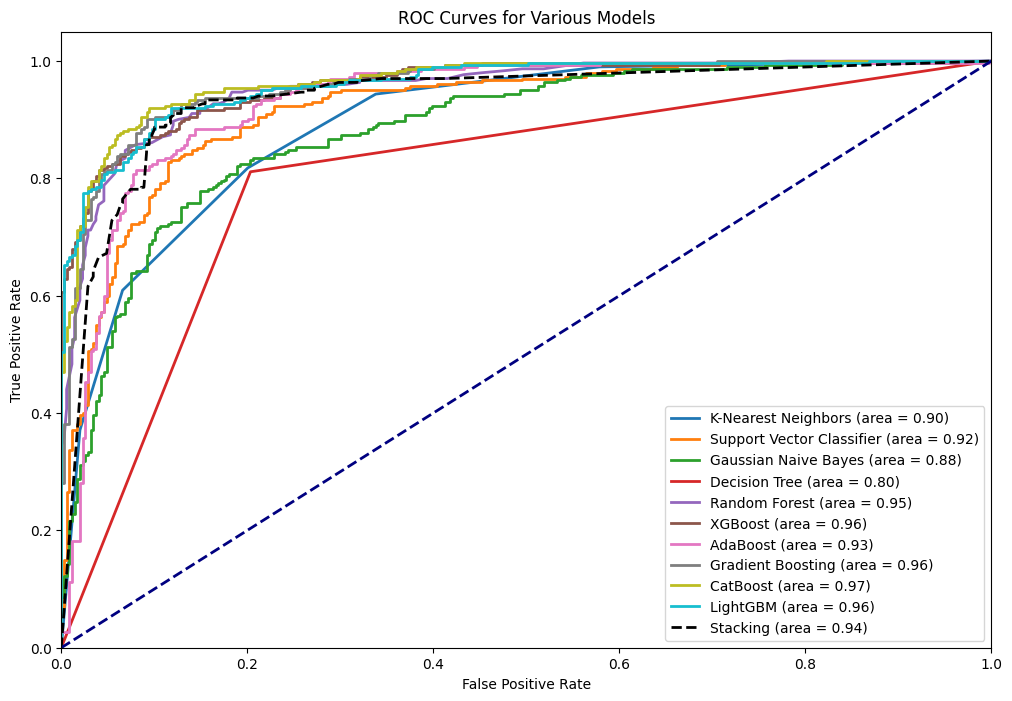


**ROC-AUC**

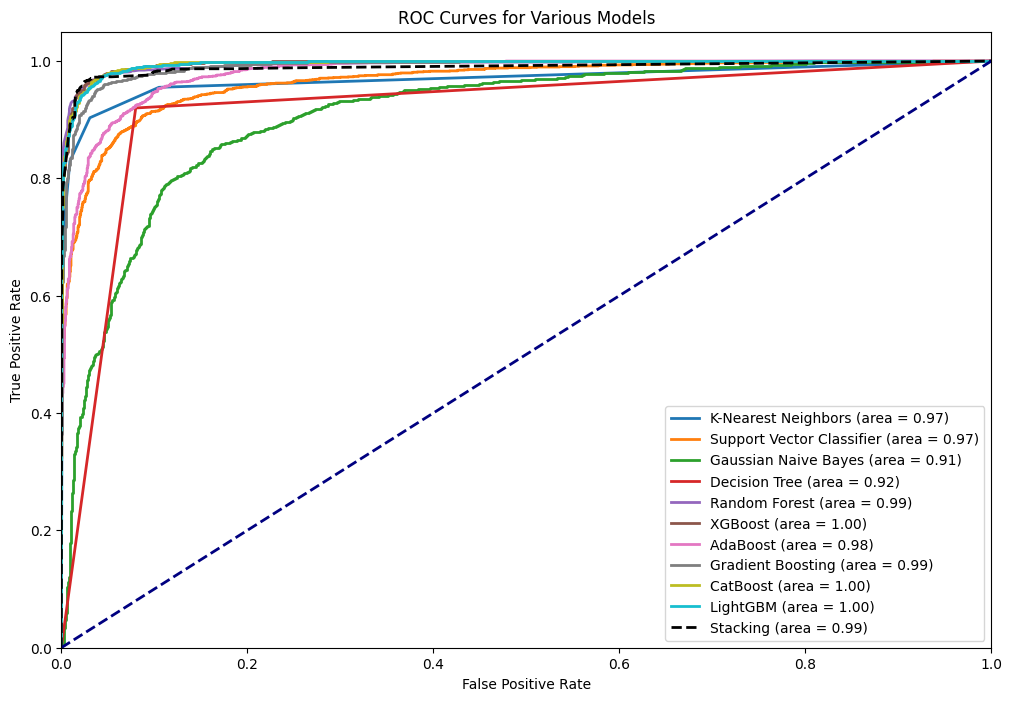
Without taking care of the class imbalance



After applying under-sampling to handle the sever class imbalance present in the dataset



After applying SMOTE to handle the sever class imbalance present in the dataset



**10. Conclusion**

**Summary and Recommendations**

The project provides a thorough analysis of customer attrition prediction using various machine learning models. The stacking ensemble model demonstrates superior performance, combining the strengths of individual models. Future work could explore more advanced techniques, including deep learning and additional data augmentation methods, to further enhance predictive accuracy.