Indian Institute of Information Technology Surat

Project Report on
Customer Conversion Prediction using Machine Learning

Submitted by

HARSHIT PANJWANI (UI21CS21)

Course Faculty

Dr. Pradeep Kumar Roy

Mr. Vipul Kania

Department of Computer Science and Engineering
Indian Institute of Information Technology Surat
Gujarat-394190, India
Academic Year - 2024-25

INTRODUCTION

Businesses in the competitive insurance sector are always looking for new and creative methods to bring in new clients and keep existing ones while increasing conversion rates. Using machine learning algorithms to forecast consumer behaviour and enhance marketing initiatives is one such tactic. The purpose of this study is to investigate this area by developing a predictive model for customer conversion in the insurance industry.

The Customer Conversion Prediction project's main goal is to create a machine learning model that can predict, using marketing and demographic data, whether a customer would purchase an insurance policy. The research aims to help insurance businesses target their marketing efforts more efficiently and cut costs by correctly identifying customers with a high possibility of conversion.

In order to accomplish this, we have extensively trained and tested a number of machine learning models, thoroughly evaluating each to determine which is the most successful. To further ensure the model's accuracy and effectiveness, we have optimised its parameters using hyperparameter tuning.

We have trained our models to precisely forecast conversion outcomes by using historical sales data which is based on various demographic variables including age, occupation, marital status, education level, call type, day of the week, month, call duration, number of calls, and previous marketing outcomes, in order to find patterns and connections that help guide the creation of a strong predictive model.

Our extensive investigation has allowed us to pinpoint the critical elements influencing conversion success. To ensure the dependability and efficacy of our model, we will assess its performance using the ROC-AUC score.

In the end, this project's primary goal is to give insurance businesses an effective tool to raise sales conversion rates and lower marketing expenses. Businesses can make better decisions by using the insights from predictive analytics, which can help them increase their profits and convert customers efficiently.

BACKGROUND

Health, life, and property insurance plans are essential for reducing financial risks for both individuals and companies. Convincing customers to purchase insurance products is still very difficult. Developing successful marketing strategies can be simplified by having an understanding of the elements impacting consumer decisions. It has been believed that the variables including age, marital status, occupation, and education level are important indicators of consumer behaviour.

One of the fundamental steps towards better customer conversion is customer segmentation. This is where personalization starts, and proper segmentation will help you make decisions regarding targeting a specific age group, at a certain interval of month or calling a person for a certain number of times only.

Research has indicated that machine learning models are more effective in predicting customer behaviour than conventional statistical techniques. Algorithms such as gradient boosting, random forests, and decision trees, for example, have proven very accurate in spotting patterns and trends in consumer data.

There are many benefits to using machine learning for customer segmentation. First of all, it is faster than time-consuming manual segmentation procedures. Algorithms for machine learning accelerate the process of analysis, allowing critical time for decision-makers to concentrate on increasingly difficult problems that require innovative solutions.

Another important advantage of using machine learning models in production is scalability. These models are naturally scalable and can handle growing data quantities without the need for additional projects because they enable cloud infrastructure.

To put it simply, using machine learning to segment customers in the insurance sector improves accuracy and efficiency as well as enables businesses to grow and adapt in the dynamic market.

EXISTING SOLUTION

When it comes to customer conversion prediction in the insurance sector, existing systems are frequently simplified and only offer basic probabilities without going into the specifics of the variables in the dataset. Usually, these approaches use the basic statistical method of linear regression to calculate conversion probabilities. Even though linear regression can provide a rough estimate of conversion probability, it is not as accurate or detailed as other machine learning methods.

Linear Regression tries to model the relationship between the independent variables (such as demographic variable and marketing data) and target variable.When working with complicated datasets that have nonlinear interactions between variables, this method is inadequate. When it comes to forecasting client conversion in the insurance sector, linear regression is unable to sufficiently consider the interactions between variables and the differing importance of features among distinct customer segments or product offerings.

The model also misses out on patterns and insights that could be obtained from more complex algorithms because it only uses linear regression and lacks flexibility regarding dataset attributes. KNN, decision trees, random forests and gradient boosting models are examples of advanced machine learning algorithms that provide more flexibility and accuracy by identifying intricate correlations and nonlinearities.

There is also scope for feature engineering, hyper-parameter tuning and deployment of a web application for the discussed project. The model can be trained in such a manner that it is continuously fed with live data and the model gets trained on this live data but it is quite difficult to implement so.

The drawbacks of the current solution can be addressed by implementing these modern algorithms for machine learning and taking a more thorough approach that takes into account the complex relationships among dataset features.

Proper exploratory data analysis can help to know the relationship between features. It would also help in getting the value of the feature which is more probable for conversion and give the best idea to target customers based on that information. Getting perfect parameters for each model can help to achieve the most efficient model with a good ROC-AUC score.

PROPOSED IDEA

The proposed solution to the customer conversion prediction problem represents a significant improvement over the current methodology because it makes use of a range of machine learning methods and provides in-depth evaluations of the effects of each feature in the dataset. Insurance firms and their staff should be able to make better decisions in terms of decision-making abilities, predictive accuracy, and model comprehension with the help of this suggested solution.

Utilisation of Multiple Machine Learning Algorithms:

The use of several machine learning algorithms, such as KNN(K-Nearest Neighbours), Decision Trees, XG Boost and Random Forests, is one of the main features of the suggested solution. A more reliable and accurate prediction is made possible by choosing the best parameters for training of other models like depth of Decision tree and Random Forest, learning rates of XG Boost.

1.**Logistic Regression:** Logistic Regression, a specialized form of regression for binary classification, directly models the probability of an observation belonging to a certain class. Unlike linear regression, Logistic Regression doesn't rely on thresholding predicted probabilities for classification. By fitting the sigmoid function to the data, it learns to predict probabilities bounded between 0 and 1.

2.**K-Nearest Neighbors (KNN)**: KNN is a non-parametric algorithm that makes predictions based on the similarity of input data points. In the context of customer conversion prediction, KNN can identify customers with similar characteristics who have previously converted, thereby informing predictions for new customers based on their nearest neighbours' behaviour.

3.**Decision Trees:** Decision trees partition the feature space into distinct regions based on the input features, enabling the identification of nonlinear relationships and interactions among variables. Decision trees provide interpretable rules for predicting customer conversion and can uncover complex decision-making processes.

4.**XGBoost:** XGBoost (Extreme Gradient Boosting) is an ensemble learning algorithm known for its high performance and scalability. By combining the predictions of multiple weak learners (decision trees), XGBoost boosts prediction accuracy and generalisation capabilities. It can effectively handle large datasets and capture intricate relationships among features.

5.**Random Forest:** Random Forest is another ensemble learning technique that constructs multiple decision trees and aggregates their predictions to produce a final output. Random Forest mitigates overfitting and enhances prediction robustness by averaging predictions across multiple trees. It is particularly adept at capturing nonlinearities and interactions in complex datasets.

Impact of Each Feature in the Dataset:

The suggested approach not only makes use of various machine learning algorithms but also examines how each feature in the dataset affects admission predictions. Understanding which factors have a significant impact on admission decisions and how applicants can strategically improve their profiles to increase their chances of being accepted are made possible by this analysis.

1. **Age** : Age serves as a crucial demographic variable influencing customer behaviour and conversion likelihood. Different age groups may exhibit varying preferences, financial needs, and receptiveness to insurance products.

- Most Target : 30 to 40 years
- Least Target : below 20 and above 60

2. **Job** : The type of job held by the customer provides insights into their financial stability, risk tolerance, and insurance needs. Occupation categories such as blue-collar, white-collar, or self-employed may indicate different levels of income, lifestyle, and insurance requirements.

- Most Target : blue-collar and management
- Least Target : students and house maid

3. **Marital Status** : Married individuals may prioritise family protection and long-term financial planning, while single individuals may prioritise personal coverage and flexibility.

- Most Target : Married
- Least Target : Divorced

4. **Educational Qualification** : Higher levels of education may correlate with greater financial planning foresight and risk awareness, impacting insurance purchasing behaviour.

- Most Target : Secondary
- Least Target : Primary

5. **Call Type** : The method of communication utilised during the marketing outreach, such as telephone or cellular.

- Most Target : Cellular
- Least Target : Telephone

6. **Day** : The timing of customer outreach, represented by the day and month of the last contact, can influence response rates and conversion probabilities

- Most Target : Mid of the month
- Least Target : Beginning of the month

7. **Month** : Seasonal trends may impact customer attention to insurance offers at different times of the year. Timing marketing campaigns to coincide with peak engagement periods can maximize conversion opportunities.

- Most Target : May
- Least Target : December

8. **Duration** : The duration of the last contact with the customer provides insights into the depth of engagement and interest level. Longer contact durations indicates a more meaningful interaction and higher likelihood of conversion.

- Most Target : call last around 1750 second
- Least Target : call last around 100 to 200 second

9. **Number of Calls** : To maximise conversion rates, the amount of contacts must be balanced so that the customer does not get frustrated.

- Most Target : most people contacted one time
- Least Target : least people contacted 5 times

10. **Outcome of Previous Marketing Campaign**: The outcome of the previous marketing campaign, categorized as unknown, other, failure, or success, informs subsequent campaign strategies

- Most Target : unknown
- Least Target : success

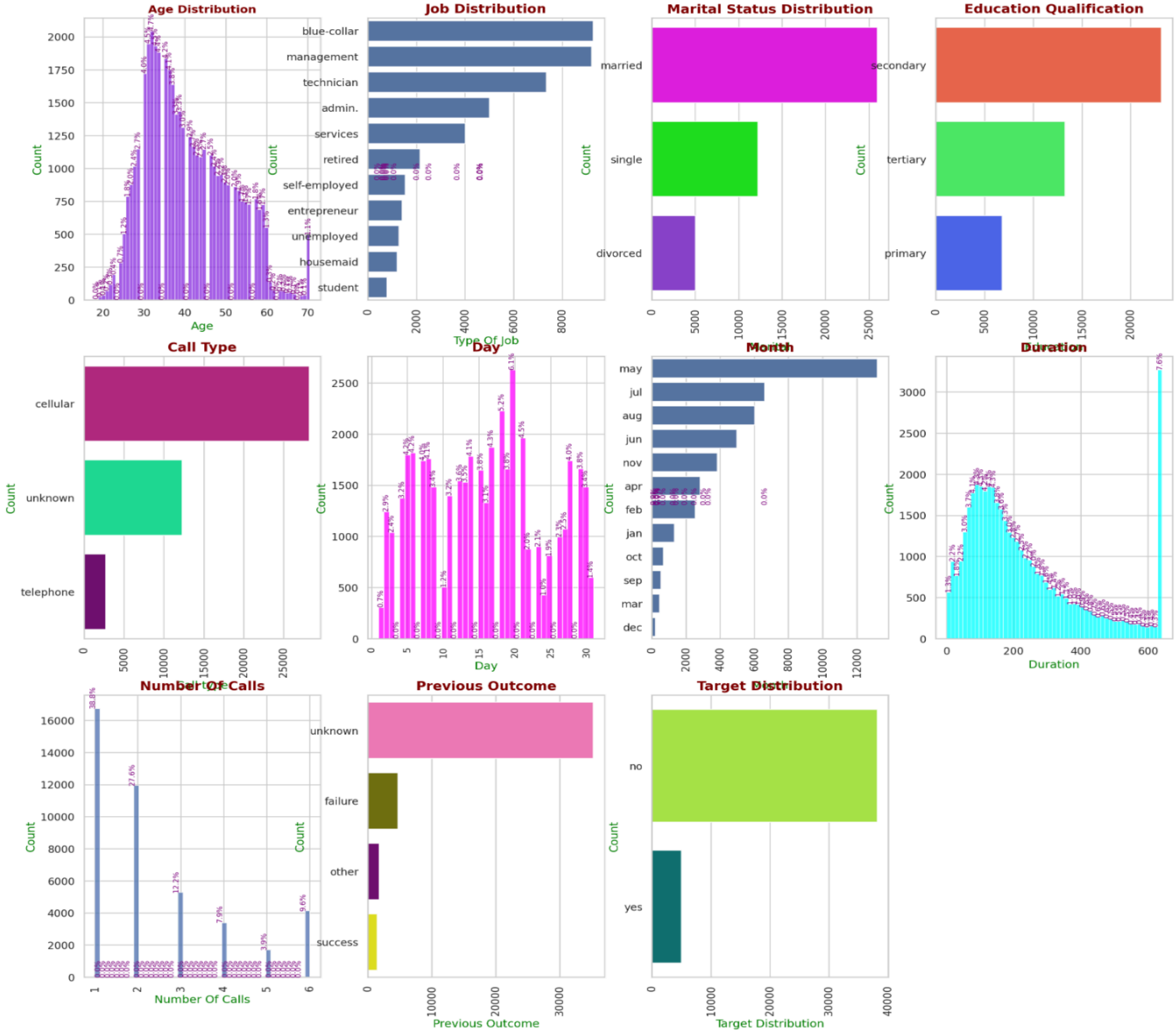


Figure 1: Feature vs Target graph

Libraries Used :

1. **pandas:** A Python library providing high-performance data structures and data analysis tools, particularly well-suited for working with tabular data and time series data.

```
import pandas as pd
```

2. **numpy (np)**: A fundamental package for scientific computing in Python, offering support for large, multi-dimensional arrays and matrices, along with a collection of mathematical functions to operate on these arrays efficiently.

```
import numpy as np
```

3. **matplotlib.pyplot (plt)**: A plotting library for creating static, interactive, and animated visualizations in Python, widely used for generating plots, histograms, and other types of charts to explore and communicate data.

```
import matplotlib.pyplot as plt
```

4. **seaborn (sns)**: A Python visualization library based on matplotlib, specializing in creating attractive and informative statistical graphics, facilitating the visualization of complex data patterns and relationships.

```
import seaborn as sns
```

5. **warnings:** A Python module providing utilities for handling warning messages, allowing users to control the display and behavior of warnings during program execution.

```
import warnings
```

6. **google.colab:** A library specific to Google Colab, enabling interaction with Google Drive and other Google services directly within the Colab environment, facilitating collaboration and data access.

```
from google.colab import drive
```

7. **StandardScaler**: A preprocessing module within scikit-learn (sklearn) offering utilities for feature scaling and normalization, particularly useful for standardizing features by removing the mean and scaling to unit variance.

```
from sklearn.preprocessing import StandardScaler
```

8. **imblearn.combine.SMOTEENN:** An imbalanced-learn (imblearn) module implementing the SMOTE-ENN algorithm, a combination of oversampling (SMOTE) and undersampling (ENN) techniques for addressing class imbalance in classification tasks.

```
from imblearn.combine import SMOTEENN
```

9. **train_test_split**: train_test_split is a function from scikit-learn used for splitting datasets into training and testing sets. It's essential for evaluating machine learning models' performance on unseen data and preventing overfitting.

```
from sklearn.model_selection import train_test_split
```

10.**sklearn.linear_model.LogisticRegression**: A module within scikit-learn (sklearn) implementing logistic regression, a linear model for binary classification tasks, estimating the probability of a binary outcome based on input features.

```
from sklearn.linear_model import LogisticRegression
```

11.**sklearn.metrics.roc_auc_score:** A module within scikit-learn (sklearn) computing the Receiver Operating Characteristic (ROC) Area Under the Curve (AUC) score, a performance metric for evaluating binary classification models based on their true positive rate and false positive rate.

```
from sklearn.metrics import roc_auc_score
```

12. **sklearn.neighbors.KNeighborsClassifier:** A module within scikit-learn (sklearn) implementing the k-nearest neighbors algorithm, a non-parametric method for classification tasks based on the similarity of feature vectors.

```
from sklearn.neighbors import KNeighborsClassifier
```

13. **sklearn.tree.DecisionTreeClassifier:** A module within scikit-learn (sklearn) constructing decision tree models for classification tasks, recursively partitioning the feature space to make predictions based on input features.

```
from sklearn.tree import DecisionTreeClassifier
```

14. **xgboost:** A Python library providing an optimized implementation of gradient boosting algorithms, known for their efficiency and accuracy in predictive modeling tasks, particularly well-suited for structured/tabular data.

```
import xgboost as xgb
```

15. **sklearn.ensemble.RandomForestClassifier:** A module within scikit-learn (sklearn) constructing random forest models, an ensemble learning method that builds multiple decision trees and aggregates their predictions for classification tasks.

```
from sklearn.ensemble import RandomForestClassifier
```

IMPLEMENTATION

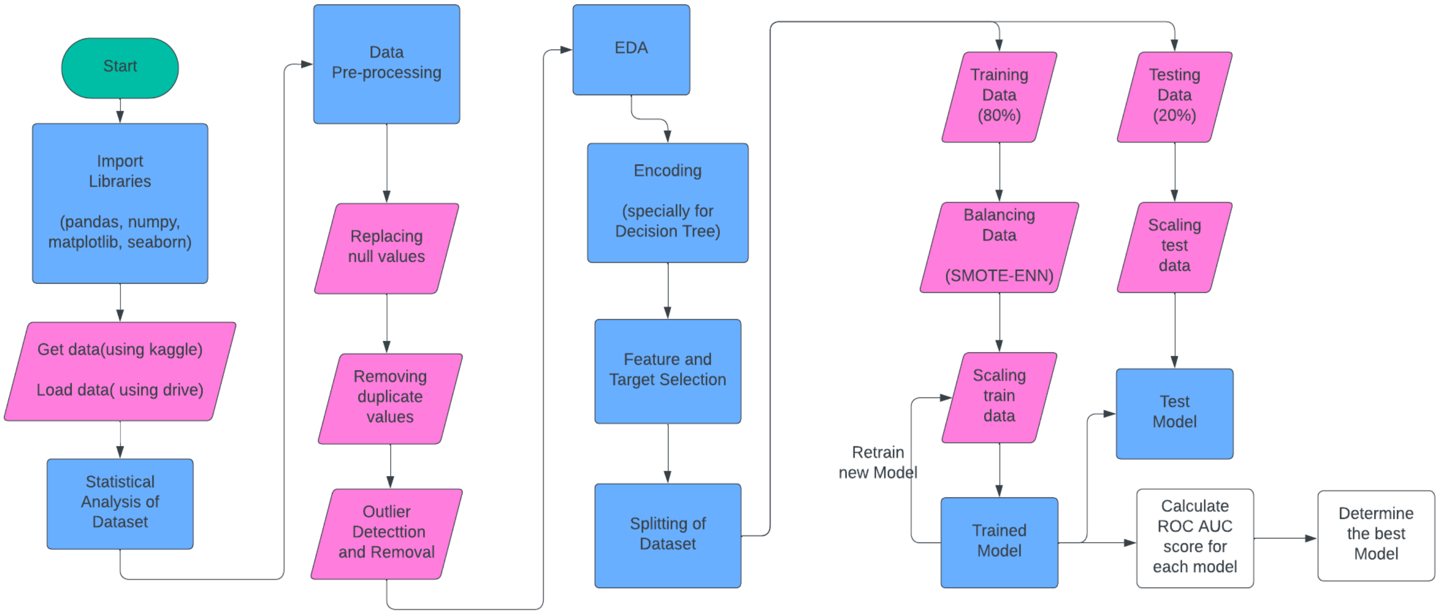


Figure 2 : Detailed flowchart of Project

Importing Libraries : In the initial step, we import essential libraries such as pandas, numpy, matplotlib, and seaborn, crucial for data manipulation and visualization tasks. These libraries collectively provide a robust foundation for efficient handling and analysis of data, as well as for creating insightful visualizations to aid in data exploration and interpretation.

Data Collection : Collecting data on customer conversion from various sources. This dataset comprises information regarding whether a group of customers subscribed to insurance or not, along with their demographic details and details related to the last contact made during the marketing campaign (e.g., day, month, duration). The data gathering process ensures the availability of a comprehensive dataset for modeling and analysis purposes.

Link to dataset

Data Preprocessing : In the dataset preprocessing phase, I focused on ensuring data cleanliness and readiness for analysis and modelling. Luckily there were no null values. Further I eliminated duplicate entries to maintain dataset integrity, and at last I detected and removed outliers to prevent them from skewing the analysis.

Exploratory Data Analysis : Analyzing a dataset includes looking into and understanding its properties. This includes looking at data distributions, feature correlations, possible outliers or anomalies, and descriptive statistics (mean, median, standard deviation). The objective is to learn more about the structure of the dataset and spot any patterns or trends that might have an impact on admission decisions.

Dataset Division : Separated the dataset into testing and training sets. The testing set is used for evaluating the performance of the machine learning models on untested data, whereas the training set is used to train the models. To guarantee proper training and evaluation, the dataset was split into 80% for the training set and 20% for the test.

Training Model : In order for the model to predict admission probabilities based on input features, it first needs to discover the basic trends and relationships in the data through training, Therefore I ran the model through multiple machine learning algorithms like Logistic Regression , KNN, Decision Trees, XGBoost and Random Forests.

Model Evaluation : Model evaluation uses the testing data to evaluate how well the trained models perform. Evaluation measures like model score, accuracy, cross val score and ROC-AUC score are employed to evaluate how well the models forecast admission results. The evaluation process helps to evaluate the model's efficiency and recognises areas in need of improvement or optimisation.

RESULT

In this project, we employed various machine learning algorithms to predict customer conversion for insurance companies. The algorithms used include Logistic Regression, KNN(K-Nearest Neighbours) Classification, Decision Tree Classification, XGBoost(Extreme Gradient Boosting) and Random Forest Classification. We evaluated the performance of each algorithm using key metrics such as accuracy, cross val score and ROC-AUC score.

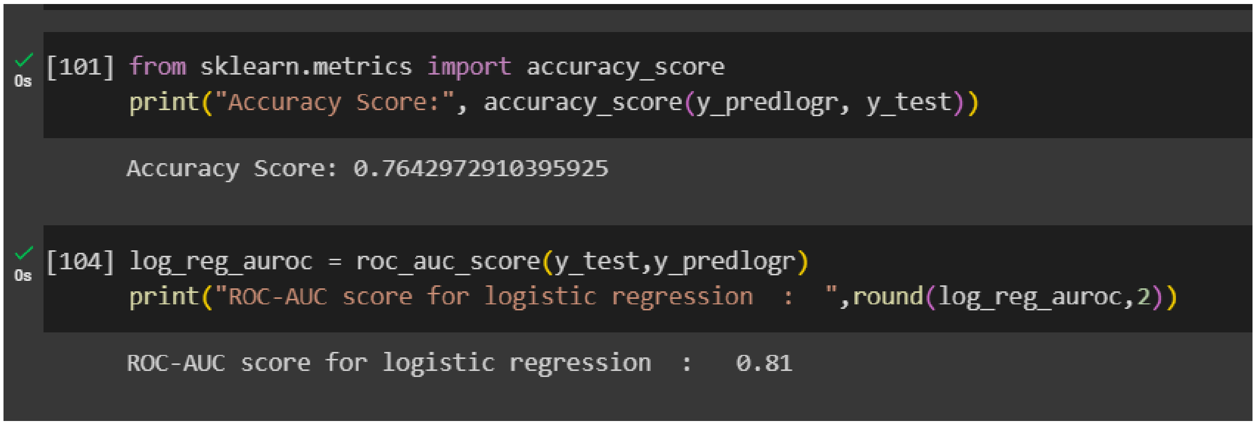
Data Preparation and Model Training: We began with a dataset containing necessary features for the data preparation and model training phase. These features included age, job, marital status, educational qualification, contact communication type, day, month, duration of the last contact, number of calls made during the campaign, previous campaign outcome, and the target variable, which indicated whether the customer had taken insurance or not. We divided the dataset into training and testing sets, allocating 80% of the data for training and 20% for testing, after completing preprocessing steps that included eliminating duplicate rows, addressing missing values, and identifying and eliminating outliers. The training set was used to teach all machine learning algorithms, such as Random Forest, XGBoost, Decision Tree, K-Nearest Neighbours, and Logistic Regression, to find patterns and relationships in the data.

Model Evaluation and Prediction :

Model evaluation means how accurately the machine learning algorithms predict admission outcomes (admitted or not admitted) based on features like age, day, month, duration, job, etc. Metrics such as accuracy, test-score and ROC-AUC score are used to measure the model's performance.

Algorithm Performance:

1. **Logistic Regression**: Logistic Regression, a specialised form of regression for binary classification, directly models the probability of an observation belonging to a certain class. The accuracy of the Logistic Regression model is 76% and the ROC-AUC score is 81% .



```
[101] from sklearn.metrics import accuracy_score
      print("Accuracy Score:", accuracy_score(y_predlogr, y_test))

      Accuracy Score: 0.7642972910395925

[104] log_reg_auroc = roc_auc_score(y_test,y_predlogr)
      print("ROC-AUC score for logistic regression  :  ",round(log_reg_auroc,2))

      ROC-AUC score for logistic regression  :   0.81
```

Fig 1.1 : Accuracy Score and ROC-AUC score for Logistic Regression

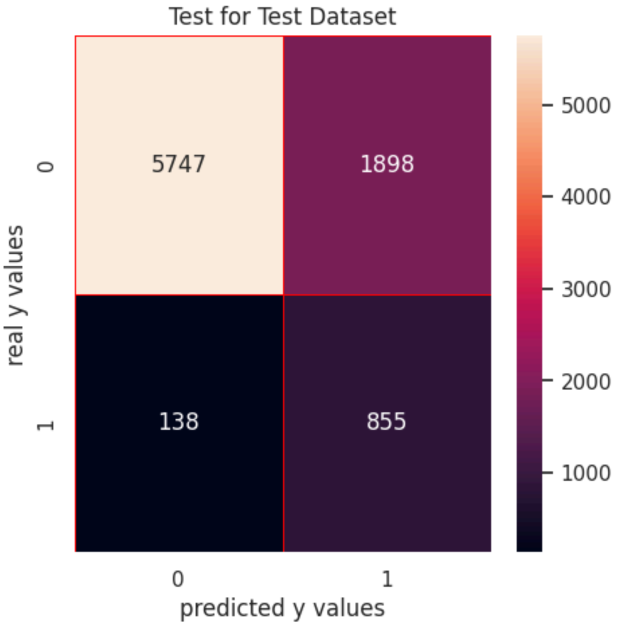


Fig 1.2 : Confusion matrix for Logistic Regression

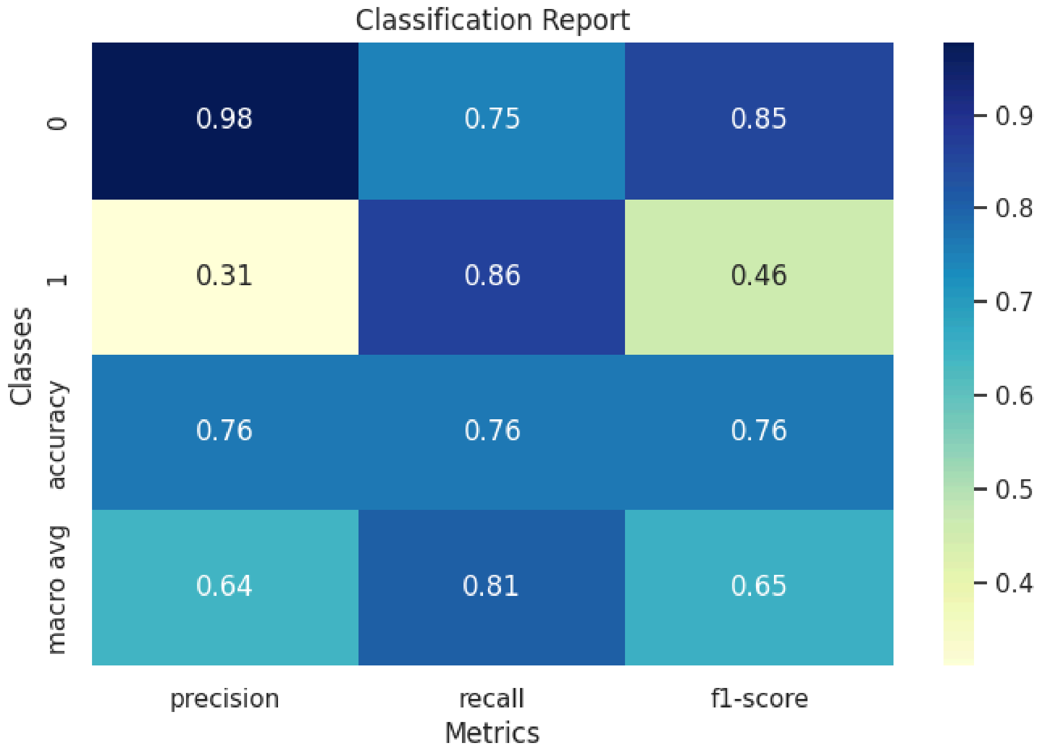
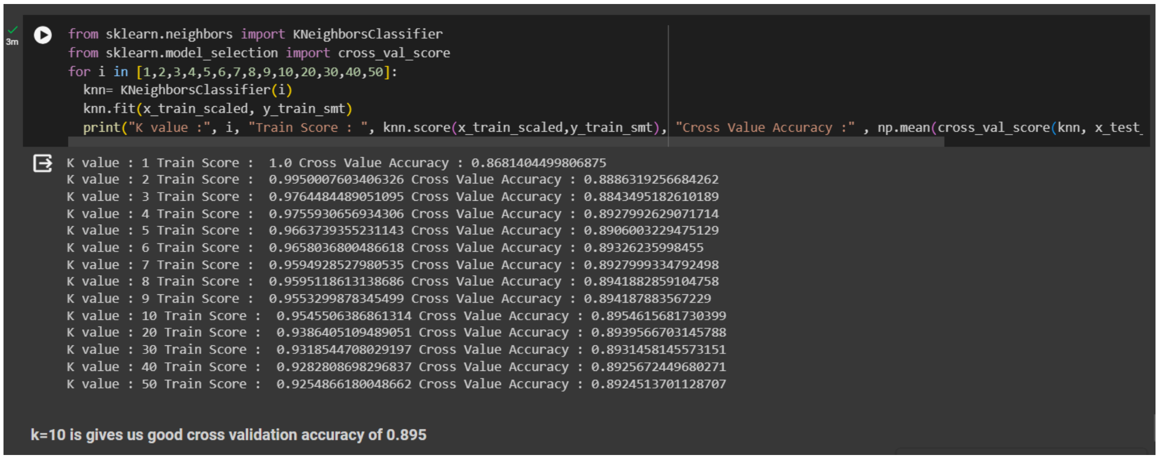


Fig 1.3 : Classification Report for Logistic Regression

2. **K-Nearest Neighbour (KNN)**: KNN classification achieved an accuracy of 89.5% (for K=10) and an ROC-AUC score of 55.5%.

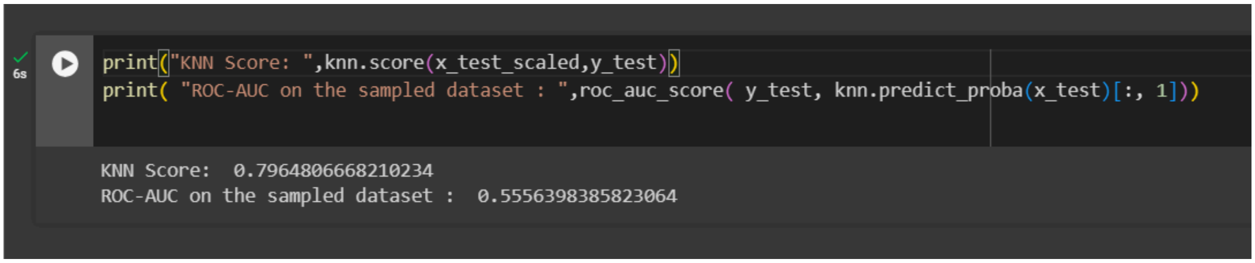


```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import cross_val_score
for i in [1,2,3,4,5,6,7,8,9,10,20,30,40,50]:
  knn= KNeighborsClassifier(i)
  knn.fit(x_train_scaled, y_train_smt)
  print("K value :", i, "Train Score : ", knn.score(x_train_scaled,y_train_smt), "Cross Value Accuracy :" , np.mean(cross_val_score(knn, x_test_
```

```
K value : 1 Train Score :  1.0 Cross Value Accuracy : 0.8681404499806875
K value : 2 Train Score :  0.9950007603406326 Cross Value Accuracy : 0.8886319256684262
K value : 3 Train Score :  0.9764484489051095 Cross Value Accuracy : 0.8843495182610189
K value : 4 Train Score :  0.9755930656934306 Cross Value Accuracy : 0.8927992629071714
K value : 5 Train Score :  0.9663739355231143 Cross Value Accuracy : 0.8906003229475129
K value : 6 Train Score :  0.9658036800486618 Cross Value Accuracy : 0.89326235998455
K value : 7 Train Score :  0.9594928527980535 Cross Value Accuracy : 0.8927999334792498
K value : 8 Train Score :  0.9595118613138686 Cross Value Accuracy : 0.8941882859104758
K value : 9 Train Score :  0.9553299878345499 Cross Value Accuracy : 0.894187883567229
K value : 10 Train Score :  0.9545506386861314 Cross Value Accuracy : 0.8954615681730399
K value : 20 Train Score :  0.9386405109489051 Cross Value Accuracy : 0.8939566703145788
K value : 30 Train Score :  0.9318544708029197 Cross Value Accuracy : 0.8931458145573151
K value : 40 Train Score :  0.9282808698296837 Cross Value Accuracy : 0.8925672449680271
K value : 50 Train Score :  0.9254866180048662 Cross Value Accuracy : 0.8924513701128707
```

k=10 is gives us good cross validation accuracy of 0.895

Fig 2.1: Accuracy Scores for KNN Classification for various values of K



```
print("KNN Score: ",knn.score(x_test_scaled,y_test))
print( "ROC-AUC on the sampled dataset : ",roc_auc_score( y_test, knn.predict_proba(x_test)[:, 1]))
```

```
KNN Score:  0.7964806668210234
ROC-AUC on the sampled dataset :  0.5556398385823064
```

Fig 2.2: Calculating ROC-AUC score for KNN Classification

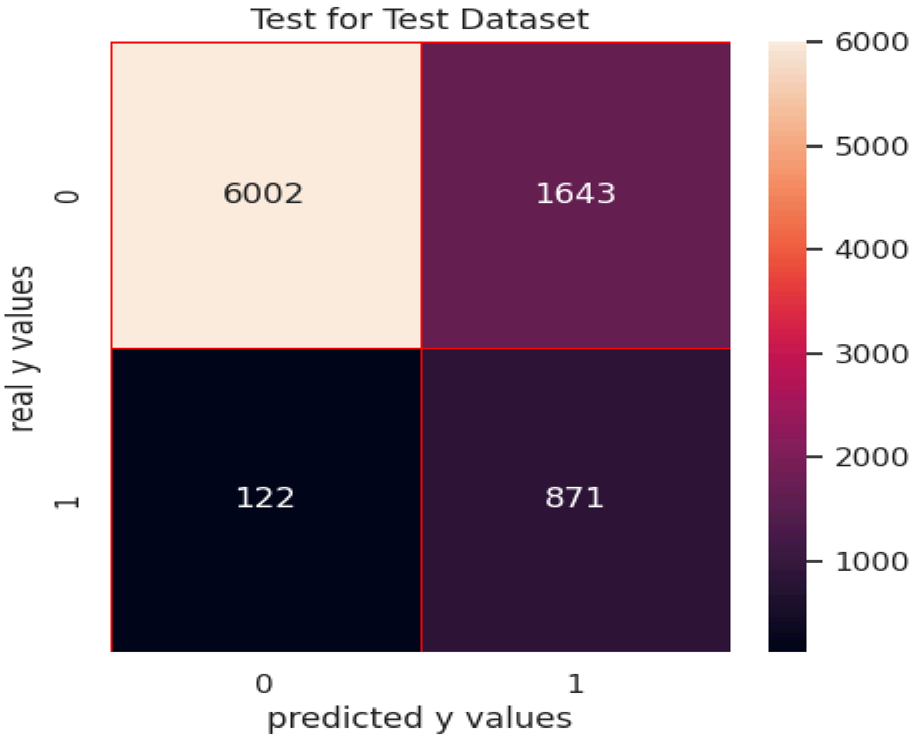


Fig 2.3: Confusion matrix for KNN algorithm

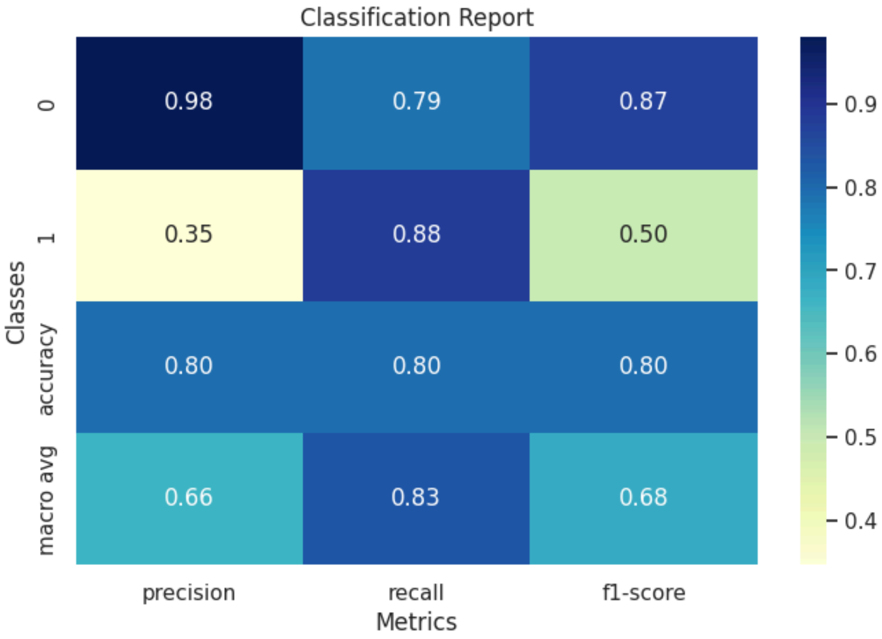
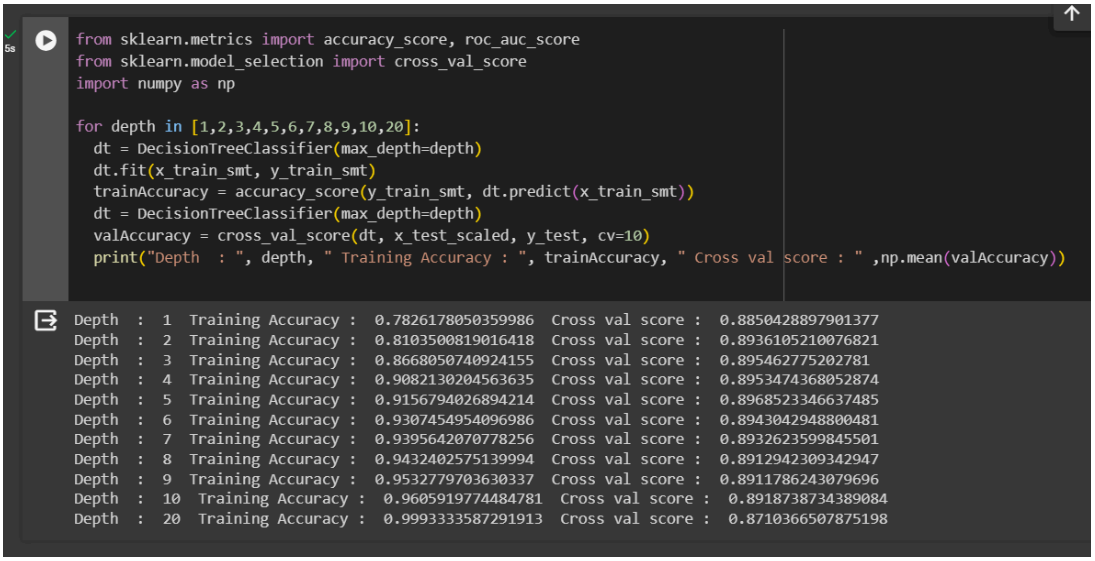


Fig 2.4: Classification Report for KNN algorithm

3. **Decision Tree Classification**: Decision Tree achieved an accuracy of 89.68%(when depth=5) and ROC-AUC score of 78.5%.

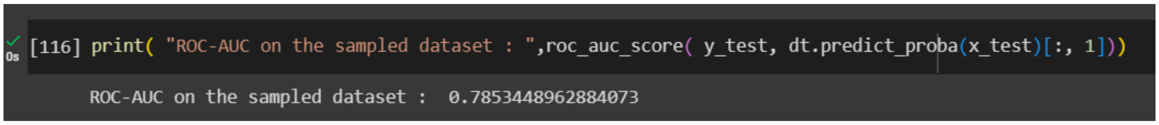


```
from sklearn.metrics import accuracy_score, roc_auc_score
from sklearn.model_selection import cross_val_score
import numpy as np

for depth in [1,2,3,4,5,6,7,8,9,10,20]:
  dt = DecisionTreeClassifier(max_depth=depth)
  dt.fit(x_train_smt, y_train_smt)
  trainAccuracy = accuracy_score(y_train_smt, dt.predict(x_train_smt))
  dt = DecisionTreeClassifier(max_depth=depth)
  valAccuracy = cross_val_score(dt, x_test_scaled, y_test, cv=10)
  print("Depth  : ", depth, " Training Accuracy : ", trainAccuracy, " Cross val score : " ,np.mean(valAccuracy))
```

```
Depth  :  1  Training Accuracy :  0.7826178050359986  Cross val score :  0.8850428897901377
Depth  :  2  Training Accuracy :  0.8103500819016418  Cross val score :  0.8936105210076821
Depth  :  3  Training Accuracy :  0.8668050740924155  Cross val score :  0.895462775202781
Depth  :  4  Training Accuracy :  0.9082130204563635  Cross val score :  0.8953474368052874
Depth  :  5  Training Accuracy :  0.9156794026894214  Cross val score :  0.8968523346637485
Depth  :  6  Training Accuracy :  0.9307454954096986  Cross val score :  0.8943042948800481
Depth  :  7  Training Accuracy :  0.9395642070778256  Cross val score :  0.8932623599845501
Depth  :  8  Training Accuracy :  0.9432402575139994  Cross val score :  0.8912942309342947
Depth  :  9  Training Accuracy :  0.9532779703630337  Cross val score :  0.8911786243079696
Depth  :  10  Training Accuracy :  0.9605919774484781  Cross val score :  0.8918738734389084
Depth  :  20  Training Accuracy :  0.9993333587291913  Cross val score :  0.8710366507875198
```

Fig 3.1: Calculating accuracy score for Decision Tree Classification



```
[116] print( "ROC-AUC on the sampled dataset : ",roc_auc_score( y_test, dt.predict_proba(x_test)[:, 1]))

ROC-AUC on the sampled dataset :  0.7853448962884073
```

Fig 3.2: Calculating ROC-AUC score for Decision Tree Classification

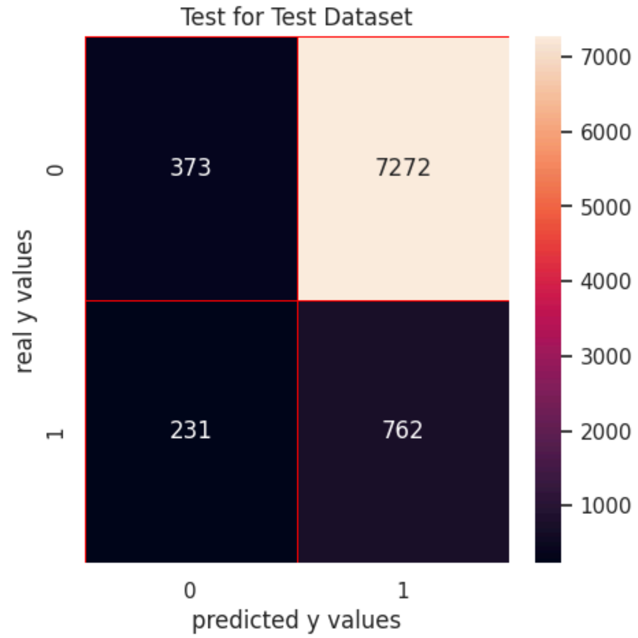


Fig 3.3: Confusion matrix for Decision Tree Classification

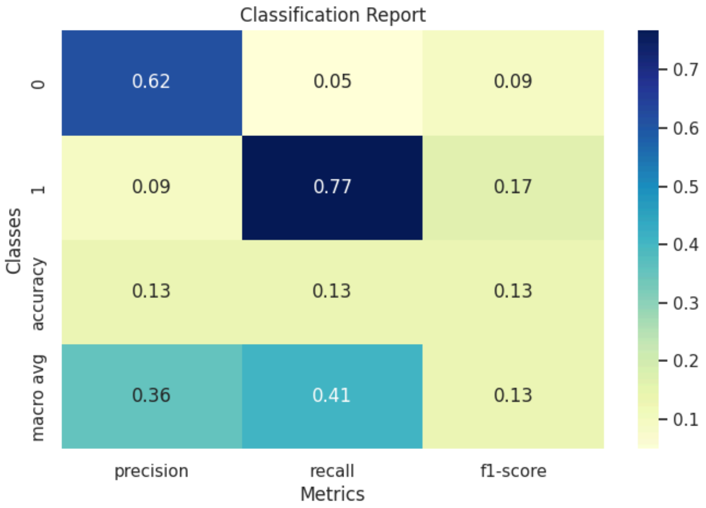
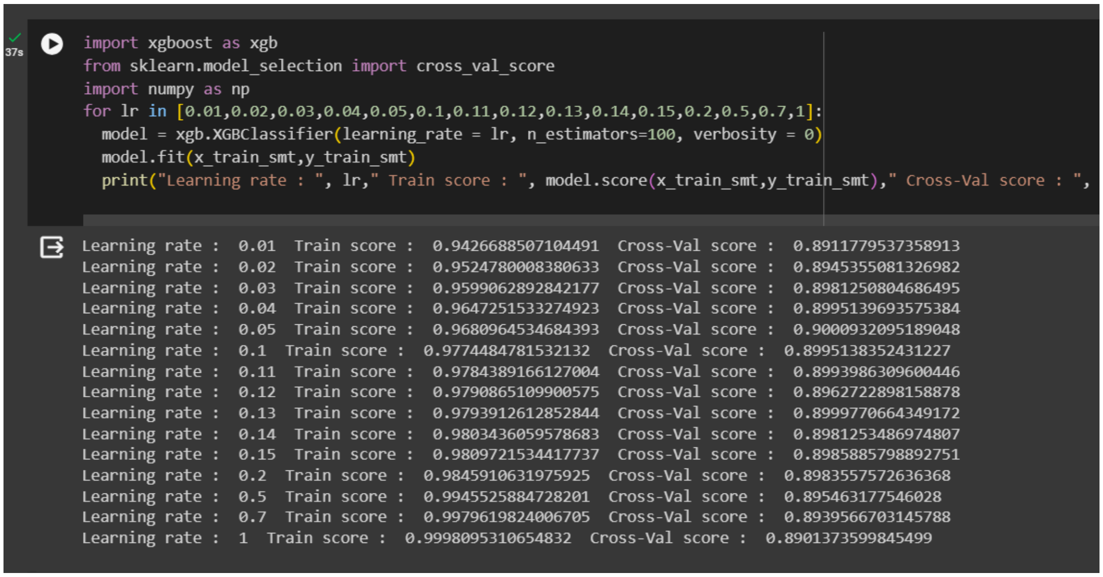


Fig 3.4: Classification Report for Decision Tree Classification

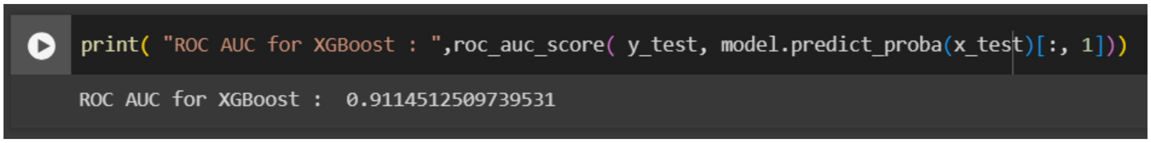
4. **XGBoost Classification**: XGBoost achieved an accuracy of 90% (when learning rate=0.05) and also achieved the highest ROC-AUC score of 91%



```
import xgboost as xgb
from sklearn.model_selection import cross_val_score
import numpy as np
for lr in [0.01,0.02,0.03,0.04,0.05,0.1,0.11,0.12,0.13,0.14,0.15,0.2,0.5,0.7,1]:
  model = xgb.XGBClassifier(learning_rate = lr, n_estimators=100, verbosity = 0)
  model.fit(x_train_smt,y_train_smt)
  print("Learning rate : ", lr," Train score : ", model.score(x_train_smt,y_train_smt)," Cross-Val score : ",
```

```
Learning rate :  0.01  Train score :  0.9426688507104491  Cross-Val score :  0.8911779537358913
Learning rate :  0.02  Train score :  0.9524780008380633  Cross-Val score :  0.8945355081326982
Learning rate :  0.03  Train score :  0.9599062892842177  Cross-Val score :  0.8981250804686495
Learning rate :  0.04  Train score :  0.9647251533274923  Cross-Val score :  0.8995139693575384
Learning rate :  0.05  Train score :  0.9680964534684393  Cross-Val score :  0.9000932095189048
Learning rate :  0.1  Train score :  0.9774484781532132  Cross-Val score :  0.8995138352431227
Learning rate :  0.11  Train score :  0.9784389166127004  Cross-Val score :  0.8993986309600446
Learning rate :  0.12  Train score :  0.9790865109900575  Cross-Val score :  0.8962722898158878
Learning rate :  0.13  Train score :  0.9793912612852844  Cross-Val score :  0.8999770664349172
Learning rate :  0.14  Train score :  0.9803436059578683  Cross-Val score :  0.8981253486974807
Learning rate :  0.15  Train score :  0.9809721534417737  Cross-Val score :  0.8985885798892751
Learning rate :  0.2  Train score :  0.9845910631975925  Cross-Val score :  0.8983557572636368
Learning rate :  0.5  Train score :  0.9945525884728201  Cross-Val score :  0.895463177546028
Learning rate :  0.7  Train score :  0.9979619824006705  Cross-Val score :  0.8939566703145788
Learning rate :  1  Train score :  0.9998095310654832  Cross-Val score :  0.8901373599845499
```

Fig 4.1: Calculating accuracy score for XGBoost Classification



```
print( "ROC AUC for XGBoost : ",roc_auc_score( y_test, model.predict_proba(x_test)[:, 1]))
```

```
ROC AUC for XGBoost :  0.9114512509739531
```

Fig 4.2: Calculating ROC-AUC score for XGBoost Classification

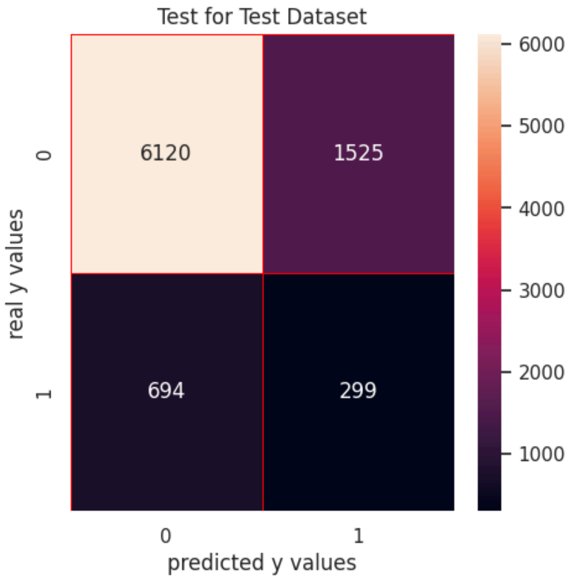


Fig 4.3: Confusion matrix for XGBoost Classification

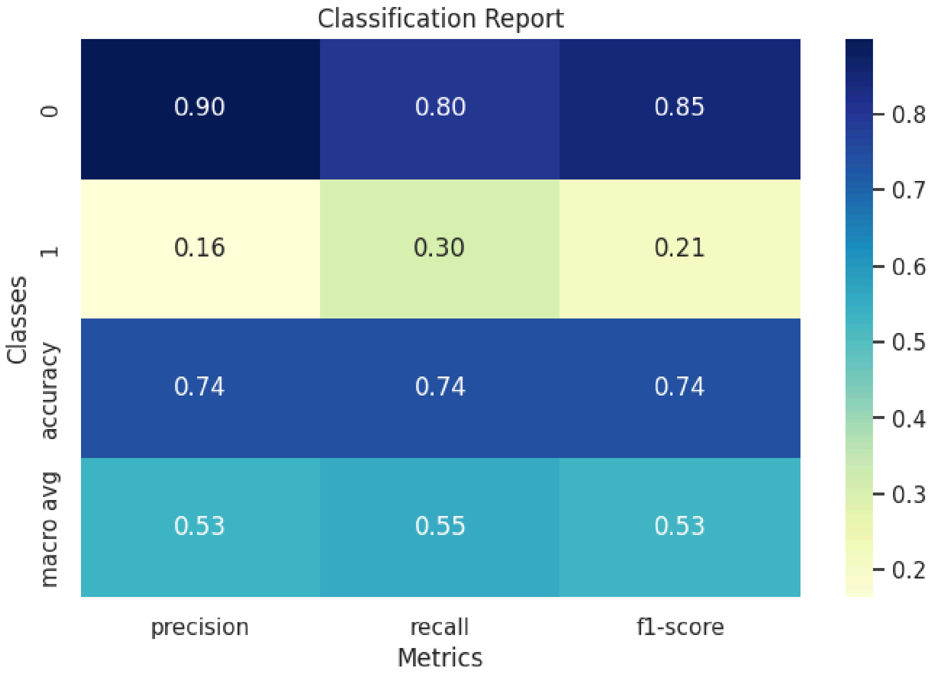
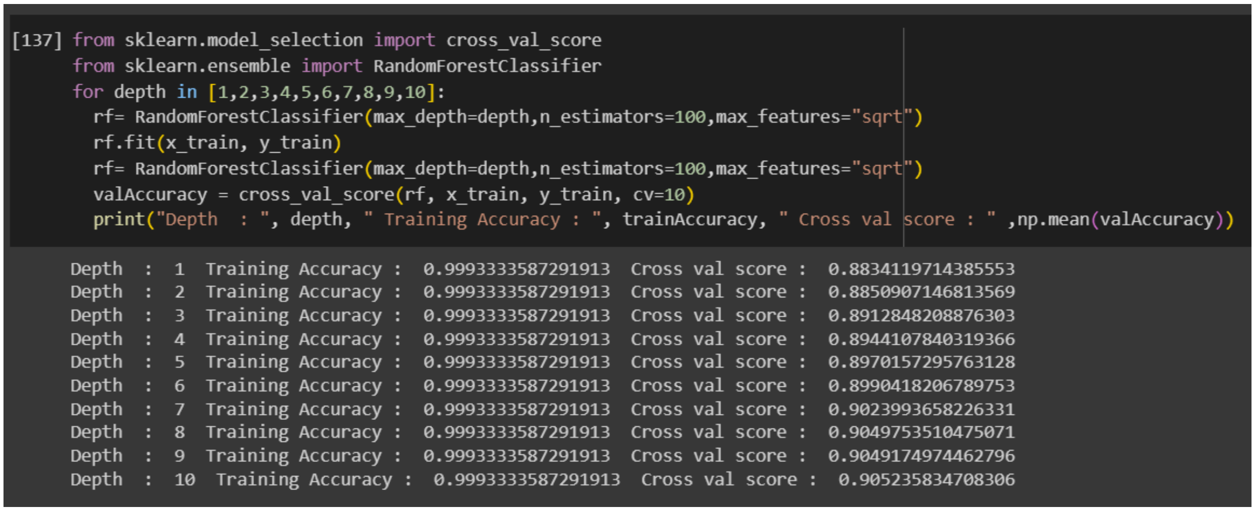


Fig 4.4: Classification Report for XGBoost Classification

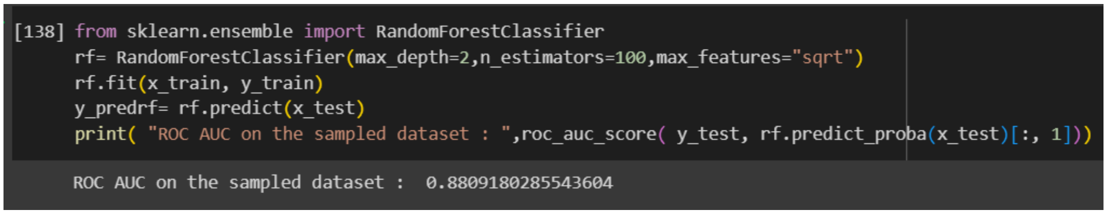
5. **Random Forest Classification**: Random forest classification delivered accuracy of 90.5% (at depth=10) and ROC-AUC score of 88% .



```
[137] from sklearn.model_selection import cross_val_score
      from sklearn.ensemble import RandomForestClassifier
      for depth in [1,2,3,4,5,6,7,8,9,10]:
        rf= RandomForestClassifier(max_depth=depth,n_estimators=100,max_features="sqrt")
        rf.fit(x_train, y_train)
        rf= RandomForestClassifier(max_depth=depth,n_estimators=100,max_features="sqrt")
        valAccuracy = cross_val_score(rf, x_train, y_train, cv=10)
        print("Depth  : ", depth, " Training Accuracy : ", trainAccuracy, " Cross val score : " ,np.mean(valAccuracy))

     Depth  :  1  Training Accuracy :  0.9993333587291913  Cross val score :  0.8834119714385553
     Depth  :  2  Training Accuracy :  0.9993333587291913  Cross val score :  0.8850907146813569
     Depth  :  3  Training Accuracy :  0.9993333587291913  Cross val score :  0.8912848208876303
     Depth  :  4  Training Accuracy :  0.9993333587291913  Cross val score :  0.8944107840319366
     Depth  :  5  Training Accuracy :  0.9993333587291913  Cross val score :  0.8970157295763128
     Depth  :  6  Training Accuracy :  0.9993333587291913  Cross val score :  0.8990418206789753
     Depth  :  7  Training Accuracy :  0.9993333587291913  Cross val score :  0.9023993658226331
     Depth  :  8  Training Accuracy :  0.9993333587291913  Cross val score :  0.9049753510475071
     Depth  :  9  Training Accuracy :  0.9993333587291913  Cross val score :  0.9049174974462796
     Depth  :  10  Training Accuracy :  0.9993333587291913  Cross val score :  0.905235834708306
```

Fig 5.1: Calculating Accuracy for Random Forest Classification



```
[138] from sklearn.ensemble import RandomForestClassifier
      rf= RandomForestClassifier(max_depth=2,n_estimators=100,max_features="sqrt")
      rf.fit(x_train, y_train)
      y_predrf= rf.predict(x_test)
      print( "ROC AUC on the sampled dataset : ",roc_auc_score( y_test, rf.predict_proba(x_test)[:, 1]))

      ROC AUC on the sampled dataset :  0.8809180285543604
```

Fig 5.2 : Calculating ROC-AUC score for Random Forest Classification

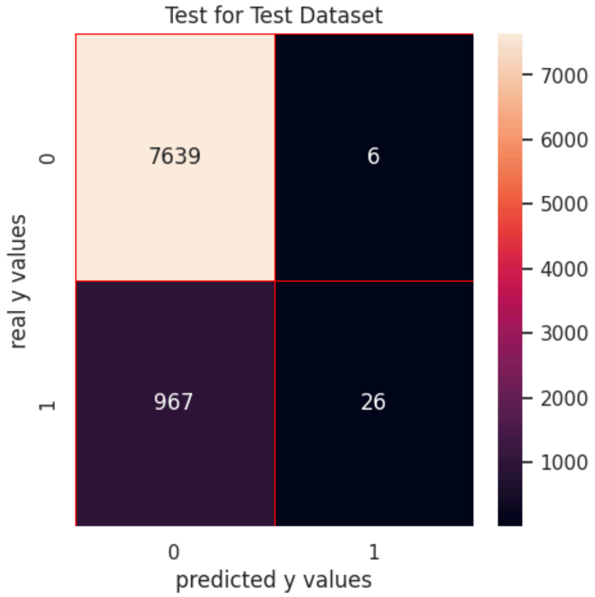


Fig 5.3 : Confusion Matrix for Random Forest Classification

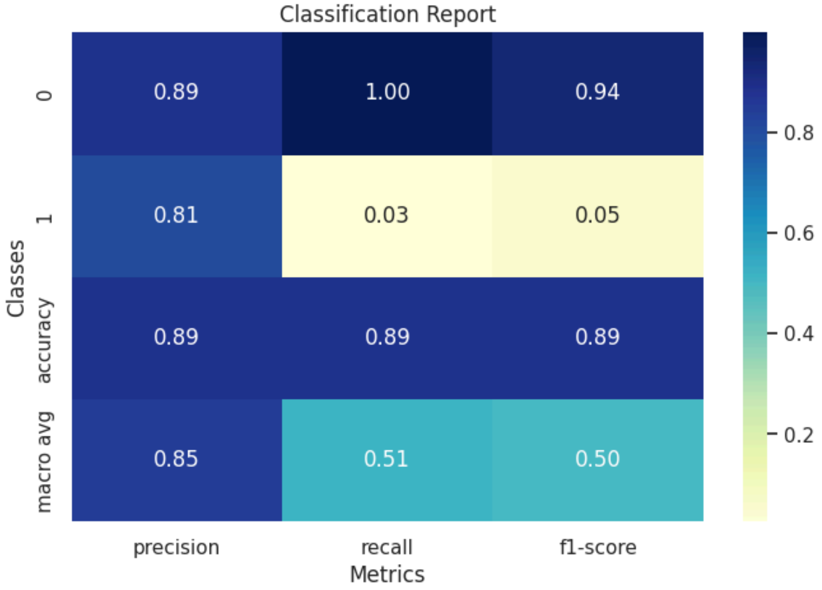


Fig 5.4 : Classification Report for Random Forest Classification

CONCLUSION

Through this customer conversion prediction study, we studied how well several machine learning algorithms performed in predicting outcomes based on characteristics of customers, including their age, job, marital status, educational qualification, call type, day, month, duration of call, number of calls and previous outcomes. Among the techniques put to the test were Logistic Regression, KNN Classification, Decision Tree, Random Forest and XGBoost.

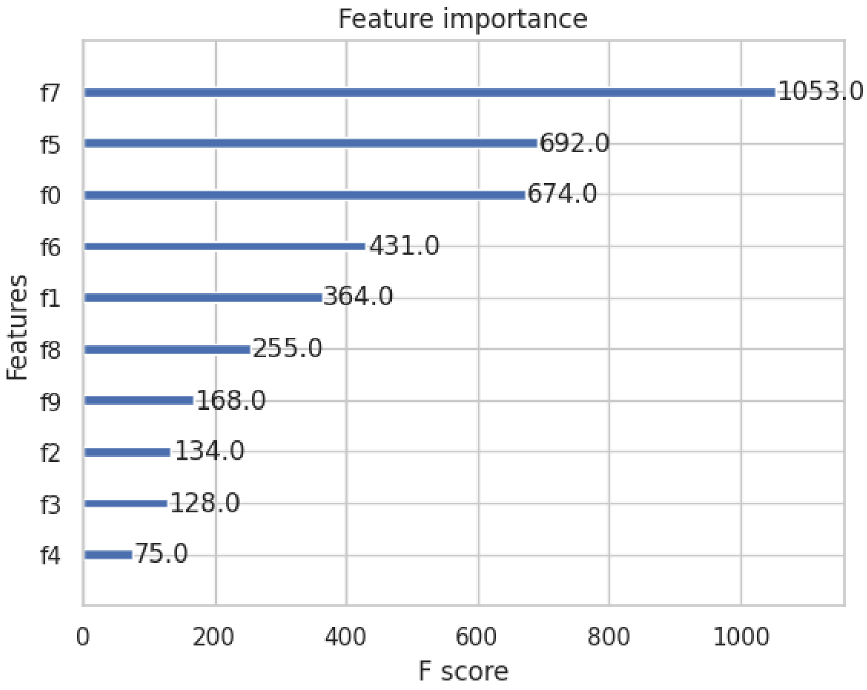
Findings and Limitations:

1. **Algorithm Performance**: Models are tested and below are the AUROC value of each model

Logistic Regression - ROC-AUC Score is 0.81
KNN - ROC-AUC Score is 0.55
Decision Tree - ROC-AUC Score is 0.785
XGBoost - ROC-AUC Score is 0.91
Random Forest - ROC-AUC Score is 0.88

Hence XGBoost is giving the good AUROC Score of 0.91, so XGBoost is the best model for customer conversion prediction.

2. **Feature Importance**: The importance of the features lies in the following order: duration(f7) > day(f5) > age(f0) > month(f6) > job(f1) > number of calls(f8) > previous outcome(f9) > marital status(f2) > educational qualification(f3) > call type(f4)



3. **Limitations**:

- Data Quality: The quality of the dataset significantly impacts model performance. Limited or biassed data may lead to optimal predictions.
- Feature Selection: The selection of features plays a crucial role in model accuracy. Predictive abilities may be improved by more feature importance analysis and improvement.
- Understanding of the Model: Although ensemble methods such as Random Forests yielded good results, their understanding may be lower than that of simpler models like Logistic Regression. This lead to difficulties in understanding the model's decision-making process.

Future Scope:

1. Feature Engineering: Exploring additional features or engineering existing ones could improve predictive power. Factors such as extracurricular activities, research experience, and personal statements could be considered.

2. Deployment: The current model could be deployed in a production environment, integrated with the company's systems, and used to target potential customers effectively.

3. Regular Maintenance: As the company's customer base grows and changes, the model's performance might degrade. Regular monitoring and maintenance of the model are necessary to ensure it continues to perform effectively.

In summary, although our project has shown promise in predicting customer conversion in the insurance industry, there are still opportunities for advancement and expansion. By addressing existing constraints and exploring potential areas for future enhancement, we can develop more accurate and reliable conversion prediction models. This continuous improvement endeavour will benefit insurance companies by optimising their marketing strategies and reducing costs, ultimately leading to improved conversion rates and enhanced customer acquisition.

REFERENCES

- https://www.kaggle.com/code/arun0309/customer-segment
- https://timesofindia.indiatimes.com/readersblog/janman/importance-of-insurance-51170/
- https://neptune.ai/blog/customer-segmentation-using-machine-learning
- https://www.sciencedirect.com/science/article/pii/S1877050918322385