Bank Marketing (Campaign)

Data Science Virtual Internship

Name: Sainad Reddy Naini (Individual)
Email: nainisainad@gmail.com
Country: India
College: University of Bath (Graduate -UK)
Specialization: Data Science

Agenda

Problem Description

Approach

EDA Summary

Model Exploration

Model Selection

Summary

problem Description

ABC Bank is preparing to launch a new term deposit product and aims to maximize the effectiveness of its marketing efforts. To achieve this, they want to develop a machine learning (ML) model that can predict whether a customer will subscribe to the term deposit based on historical data from previous marketing campaigns. This predictive model will help the bank identify potential customers who are more likely to purchase the product, allowing them to focus their marketing efforts more effectively and efficiently.

Approach

1. **Data Preparation**
   - **Cleaning the Data**: Removing any unnecessary or irrelevant information to ensure the data is suitable for model building.
   - **Splitting the data:** Dividing the dataset into training and testing sets to evaluate model performance.
2. **Model exploration**
   - **Exploring Different Models**: Testing various models (Logistic Regression, Decision Tree, Random Forest, and XGBoost) and comparing their performances.
3. **Model selection**
   - **Fine-Tuning the Model:** Adjusting the chosen model's parameters to optimize its performance and ensure the best results.

EDA Summary

Key Findings:

- None of the variables are strongly connected to the target (whether someone subscribed).
- Retirees and single people are more likely to subscribe to term deposits.
- People tend to subscribe more when contacted by phone.
- The highest number of subscriptions happened in December, March, October, and September, mostly on Thursdays and Tuesdays.

Also, making more calls doesn't guarantee more subscriptions. But focusing on these patterns could help increase term deposits.

Model Exploration

From our previous analysis, we identified four models to test: **Logistic Regression, Decision Tree, Random Forest,** and **XGBoost.**

Before building these models, we first cleaned the data to make it usable for machine learning. This involved removing some unnecessary columns and converting categories (like job types and education) into numbers so the models can understand them.

Next, we split the data into **80% for training** and **20% for testing**—a standard approach to see how well the models perform. To address the imbalance in our data (since we have many more "No" cases than "Yes" cases), we used a technique called **SMOTE** to balance the classes.

Model Exploration
(Evaluation metrics)

Accuracy: The percentage of all predictions (both correct and incorrect) that the model got right.
Precision: The percentage of correct positive predictions made by the model out of all positive predictions it made.
Recall: The percentage of actual positives that were correctly identified by the model.
F1-Score:A balance between precision and recall, showing how well the model performs on positive cases.
ROC-AUC: A measure of how well the model distinguishes between positive and negative cases, with a score closer to 1 indicating better performance.
Confusion Matrix: A table that shows the counts of correct and incorrect predictions broken down by actual and predicted classes.

Model Exploration
(Logistic Regression)

We can see that all the values have increased when we use duration feature in the model training.

```
Logistic Regression with SMOTE:
Accuracy: 0.8438
Precision: 0.3448
Recall: 0.4563
F1-Score: 0.3928
ROC AUC: 0.6741
Confusion Matrix:
[[6469  783]
 [ 491  412]]
```

Fig:- Without duration feature

```
Logistic Regression with SMOTE(duration):
Accuracy: 0.8991
Precision: 0.5442
Recall: 0.5449
F1-Score: 0.5445
ROC AUC: 0.7440
Confusion Matrix:
[[6840  412]
 [ 411  492]]
```

Fig:- With duration feature

Model Exploration
(Decision Tree)

We can see that all the values have increased when we use duration feature in the model training.

```
Decision Tree with SMOTE:
Accuracy: 0.8362
Precision: 0.3034
Recall: 0.3699
F1-Score: 0.3333
ROC AUC: 0.6321
Confusion Matrix:
[[6485  767]
 [ 569  334]]
```

Fig:- Without duration feature

```
Decision Tree with SMOTE(duration):
Accuracy: 0.8917
Precision: 0.5100
Recall: 0.5626
F1-Score: 0.5350
ROC AUC: 0.7476
Confusion Matrix:
[[6764  488]
 [ 395  508]]
```

Fig:- With duration feature

Model Exploration
(Random Forest)

We can see that all the values have increased when we use duration feature in the model training.

```
Random Forest with Smote:
Accuracy: 0.8853
Precision: 0.4805
Recall: 0.4363
F1-Score: 0.4573
ROC AUC: 0.6888
Confusion Matrix:
[[6826  426]
 [ 509  394]]
```

Fig:- Without duration feature

```
Random Forest with Smote(duration):
Accuracy: 0.9113
Precision: 0.5976
Recall: 0.6102
F1-Score: 0.6038
ROC AUC: 0.7795
Confusion Matrix:
[[6881  371]
 [ 352  551]]
```

Fig:- With duration feature

Model Exploration
(XGBoost)

We can see that all the values have increased when we use duration feature in the model training.

```
XGBoost with SMOTE:
Accuracy: 0.8865
Precision: 0.4852
Recall: 0.4186
F1-Score: 0.4495
ROC AUC: 0.6817
Confusion Matrix:
[[6851  401]
 [ 525  378]]
```

Fig:- Without duration feature

```
XGBoost with SMOTE(duration):
Accuracy: 0.9127
Precision: 0.6044
Recall: 0.6124
F1-Score: 0.6084
ROC AUC: 0.7812
Confusion Matrix:
[[6890  362]
 [ 350  553]]
```

Fig:- With duration feature

Model Exploration
(Summary-Without duration)

- Both the Random Forest model and XGBoost performed well, correctly predicting around 88.5% of the cases.
- XGBoost was the best at correctly identifying positive cases, with about 48.5% of its positive predictions being accurate.
- The Random Forest model was better at finding actual positive cases, catching about 43.6% of them.
- Random Forest also balanced precision and recall the best, scoring 45.7% overall.
- Random Forest showed the best ability to distinguish between the two classes, with a score of 0.6888.

Conclusion: The Random Forest model seems to work the best among all the options we tried, providing a good balance of performance. The next step will be to fine-tune this model to make it even better.

Model Exploration
(Summary-With duration)

- Both the Random Forest model and XGBoost performed well, correctly predicting around 91% of the cases.
- XGBoost was the best at correctly identifying positive cases, with about 60.44% of its positive predictions being accurate.
- The XGBoost model was better at finding actual positive cases, catching about 61.24% of them.
- XGBoost also balanced precision and recall the best, scoring 60.84% overall.
- Random Forest and XGBoost showed the best ability to distinguish between the two classes, with a score of 0.7812 and 0.7795 respectively.

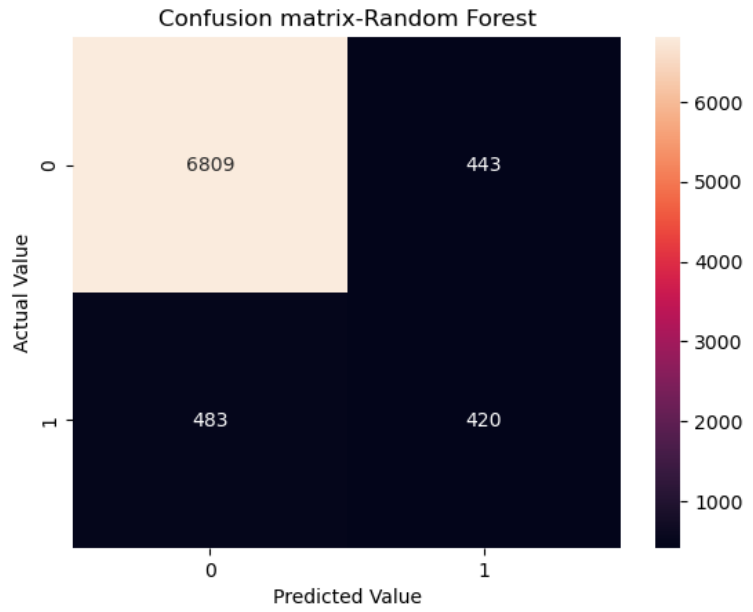
Conclusion: The XGBoost model seems to work the best among all the options we tried, providing a good balance of performance.

Model Selection
(Random Forest)

The model performs very well at predicting the negative class (0) but has some difficulty predicting the positive class (1), which might be due to the imbalance in the data. While the accuracy has increased slightly, it is important to consider other metrics like F1-score, recall, and precision, which have shown significant improvement after fine-tuning.

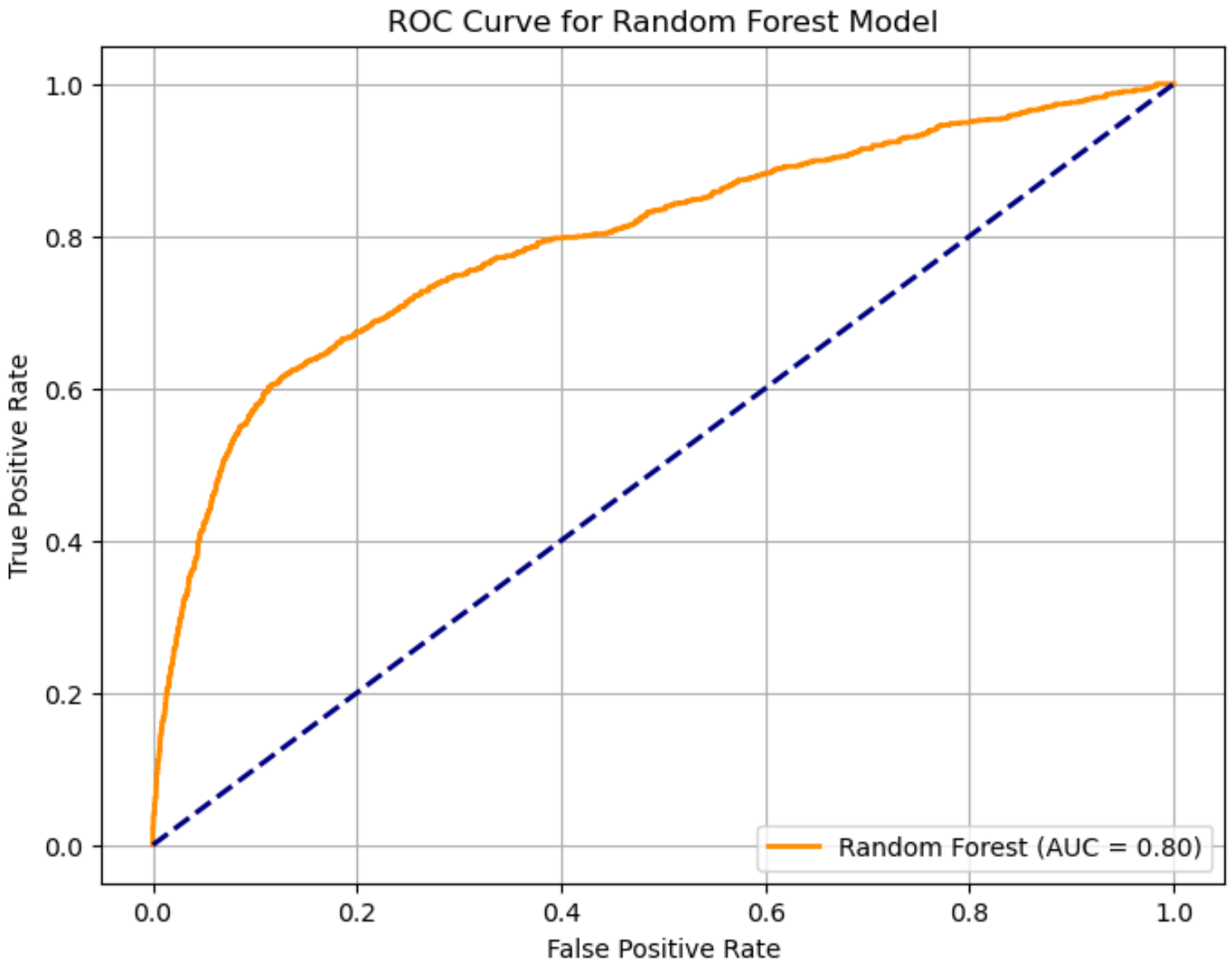
Although we built models both with and without the "duration" feature, our primary focus is on developing a model without using "duration". As a result, I have selected the **Random Forest Classifier** as our final model. After fine-tuning its parameters, we achieved the following results:

```
Accuracy: 0.8865
Precision: 0.4867
Recall: 0.4651
F1-Score: 0.4757
ROC AUC: 0.7020
Confusion Matrix:
[[6809  443]
 [ 483  420]]
```



Model Selection
(ROC Curve)

The ROC curve shows how well a model can distinguish between two classes by plotting the trade-off between correctly identifying positives and mistakenly identifying negatives. It helps to visualize and choose the best model by looking at how close the curve is to the top-left corner. We can say that the model performs well based on the graph. But need some more parameter tuning to make it a perfect model.



Thank You