Bank Customer Churn

March 6, 2025

By Tevin Parathattal

1 Factors Affecting Bank Customer Churn

1.1 Purpose and Questions

In today's financial landscape, customer retention has become a critical challenge for banks and financial institutions. With the rise of digital banking and increased competition, understanding why customers leave—also known as churn—is more important than ever. Many factors may contribute to a customer's decision to close their account, including financial stability, customer service quality, account fees, and available banking features. Some argue that demographics, income levels, or transaction behaviors play a significant role, while others believe that customer loyalty programs and personalized banking experiences can reduce churn rates. Despite extensive research, the debate continues. Through this analysis, I aim to uncover key factors influencing bank customer churn, explore potential predictors, and determine whether correlation implies causation or if there are deeper insights driving customer behavior. ### Questions to Answer

- Does financial stability influence a bank customer's likelihood to churn?
- Do premium banking services, often accessible to higher-income customers, reduce churn rates?
- How do customer engagement and personalized banking experiences impact retention?

I will be using a dataset from Kaggle. You can access the dataset with this link https://www.kaggle.com/datasets/saurabhbadole/bank-customer-churn-prediction-dataset.

[5]:
```
#first, let's load the dataset, we'll need pandas
import pandas as pd

df = pd.read_csv('Churn_Modelling.csv')
```

[6]:
```
df.shape #rows x columns
print(df.head())
```

```
   RowNumber  CustomerId   Surname  CreditScore Geography  Gender  Age  \
0          1    15634602  Hargrave          619    France  Female   42
1          2    15647311      Hill          608     Spain  Female   41
2          3    15619304      Onio          502    France  Female   42
3          4    15701354      Boni          699    France  Female   39
4          5    15737888  Mitchell          850     Spain  Female   43
```

```
   Tenure    Balance  NumOfProducts  HasCrCard  IsActiveMember  \
0       2       0.00              1          1               1
1       1   83807.86              1          0               1
2       8  159660.80              3          1               0
3       1       0.00              2          0               0
4       2  125510.82              1          1               1

   EstimatedSalary  Exited
0        101348.88       1
1        112542.58       0
2        113931.57       1
3         93826.63       0
4         79084.10       0
```

1.2 Preprocessing

Before we start visualizing data, we should evaluate the dataset and determine whether the dataset needs any cleaning done.

1.2.1 Check Nulls

[9]:
```
print(df.isnull().sum())   #check for nulls in every column
print(df.isnull().sum().sum())
print(df.dtypes)
```

```
RowNumber          0
CustomerId         0
Surname            0
CreditScore        0
Geography          0
Gender             0
Age                0
Tenure             0
Balance            0
NumOfProducts      0
HasCrCard          0
IsActiveMember     0
EstimatedSalary    0
Exited             0
dtype: int64
0
RowNumber           int64
CustomerId          int64
Surname            object
CreditScore         int64
Geography          object
Gender             object
Age                 int64
Tenure              int64
```

```
Balance            float64
NumOfProducts        int64
HasCrCard            int64
IsActiveMember       int64
EstimatedSalary    float64
Exited               int64
dtype: object
```

1.2.2 Check Duplicates

[11]:
```
print(df.duplicated().sum())  # Count duplicate rows
```

```
0
```

1.2.3 Encoding Categorical Variables

In the following code snippet, I transformed categorical variables into numeric form so the machine learning model can interpret them. Specifically, I applied label encoding to the "Gender" column, converting the categories "Male" and "Female" into numeric values (1 and 0 respectively). For the "Geography" column, which has multiple categories (countries like Germany, Spain, etc.), I used one-hot encoding, creating new binary columns for each category while dropping one to avoid redundancy. These transformations make categorical data suitable for modeling and ensure accurate predictions by clearly representing these variables numerically.

[14]:
```
import pandas as pd
from sklearn.preprocessing import LabelEncoder

# label Encoding for Gender (Binary)
df['Gender'] = LabelEncoder().fit_transform(df['Gender'])  # Male: 1, Female: 0

# One-Hot Encoding for Geography (Multiclass)
df = pd.get_dummies(df, columns=['Geography'], drop_first=True)

print(df.head())
```

```
   RowNumber  CustomerId   Surname  CreditScore  Gender  Age  Tenure  \
0          1    15634602  Hargrave          619       0   42       2
1          2    15647311      Hill          608       0   41       1
2          3    15619304      Onio          502       0   42       8
3          4    15701354      Boni          699       0   39       1
4          5    15737888  Mitchell          850       0   43       2

     Balance  NumOfProducts  HasCrCard  IsActiveMember  EstimatedSalary  \
0       0.00              1          1               1        101348.88
1   83807.86              1          0               1        112542.58
2  159660.80              3          1               0        113931.57
3       0.00              2          0               0         93826.63
4  125510.82              1          1               1         79084.10
```

```
   Exited  Geography_Germany  Geography_Spain
0       1              False            False
1       0              False             True
2       1              False            False
3       0              False            False
4       0              False             True
```

So far, we have checked for duplicates, encoded categorical variables, and ensured that the dataset is clean and well-structured. Since there were no missing values, no imputation was necessary, allowing us to retain the original integrity of the data. I also converted categorical variables into numerical representations to make them compatible with machine learning models. These preprocessing steps ensure that the dataset is properly formatted and ready for analysis. With a clean and structured dataset, we can now proceed confidently with further exploration and modeling.

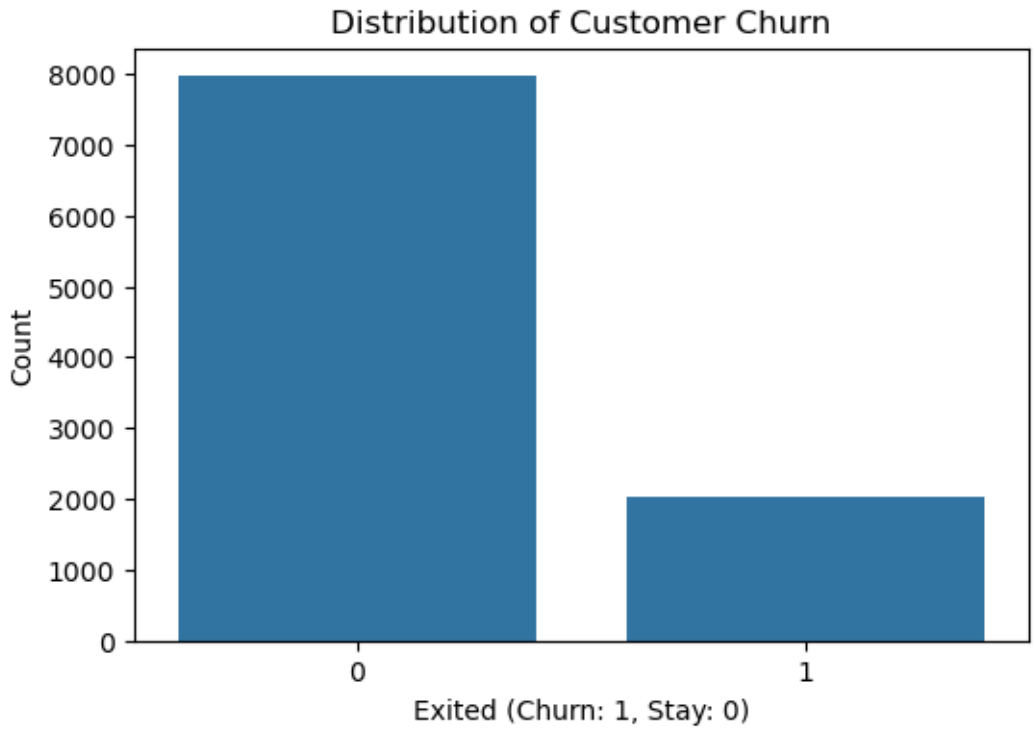
1.3 Data Visualization and Understanding

Before we start setting up our model, we can often learn a lot from data visualization and exploring our dataset first. Through this section, we'll visualize the distribution of the target variable, age, and account baance. Additionally, we'll set up a box plot evaluating balance vs churn status.

1.3.1 Analyzing Distribution of Churn

In the below bar chart, I am visualizing the distribution of the "Exited" variable to quickly assess how many customers have churned (Exited = 1) versus those who remain (Exited = 0). This initial exploration is crucial because it highlights whether the dataset is imbalanced—a common occurrence in churn scenarios where most customers typically stay. Recognizing such imbalances informs my modeling strategy, guiding decisions about which performance metrics (e.g., recall, precision, or F1-score) might best reflect how effectively a model identifies customers at risk of leaving. By creating this count-based visualization, I ensure a clear understanding of the churn landscape before proceeding with deeper analysis and using modeling.

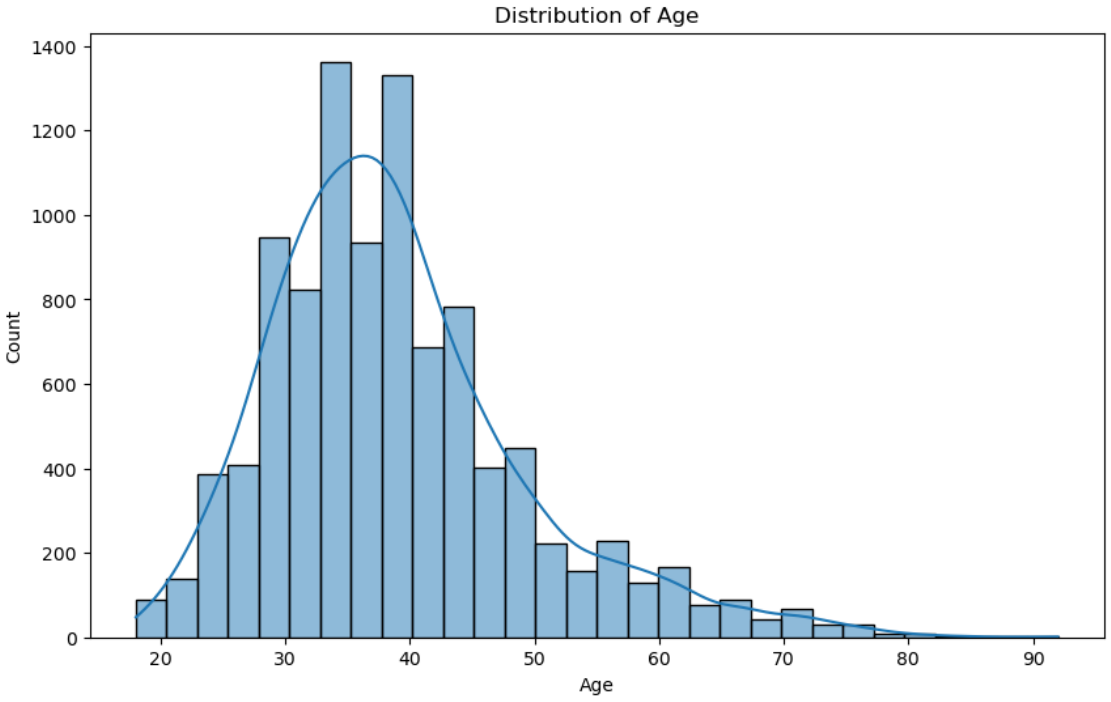
[18]:
```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

plt.figure(figsize=(6,4))
sns.countplot(data=df, x='Exited')
plt.title("Distribution of Customer Churn")
plt.xlabel("Exited (Churn: 1, Stay: 0)")
plt.ylabel("Count")
plt.show()
```



1.3.2 Distribution of Ages in Dataset

In this histogram, I'm looking at the distribution of customer ages in our dataset. This helps me quickly see the most common age groups and how ages vary among customers. Understanding this distribution is important because age might be a key factor influencing whether customers leave or stay with the bank. With this visualization, I can easily spot patterns—such as age ranges where churn might be more frequent—which helps guide my decisions later when building a predictive model.

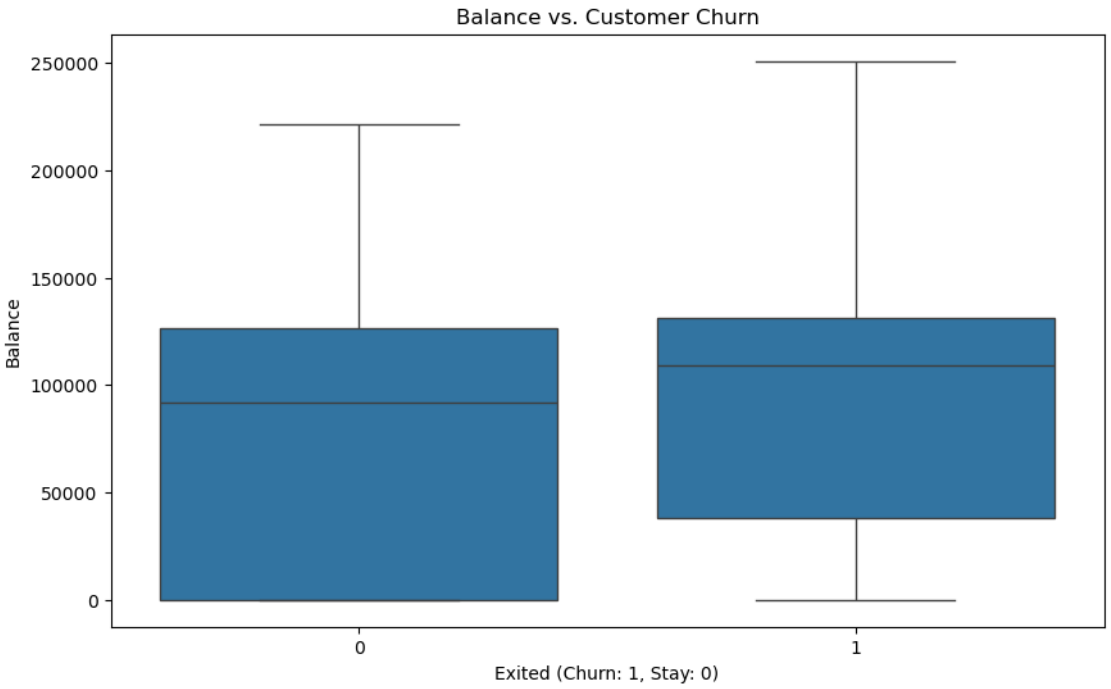
[20]:
```
plt.figure(figsize=(10,6))
sns.histplot(df['Age'], bins=30, kde=True)
plt.title("Distribution of Age")
plt.xlabel("Age")
plt.show()
```



1.3.3 Account Balance vs Customer Churn

In this histogram, I'm visualizing the distribution of customers' account balances. This helps me quickly understand how much money customers typically have in their accounts and identify common balance ranges. Seeing this distribution is useful because account balance could be an important factor affecting whether customers decide to leave or stay with the bank. By clearly visualizing these patterns early, I'm better prepared to build a model that accurately predicts customer churn based on financial factors like account balance.

[22]:
```
plt.figure(figsize=(10,6))
sns.boxplot(x='Exited', y='Balance', data=df)
plt.title("Balance vs. Customer Churn")
plt.xlabel("Exited (Churn: 1, Stay: 0)")
plt.ylabel("Balance")
plt.show()
```



1.3.4 Data Visualization Conclusion

Based on the visualizations, I've found that our dataset has a clear imbalance: most customers remain with the bank, while a smaller group chooses to leave (churn). Additionally, the distributions for age and account balance indicate that these factors vary significantly among customers and might be key predictors for churn. Recognizing these trends early on helps ensure we choose the right techniques and metrics for modeling and evaluation. Moving forward, I'll use methods suited for imbalanced data, ensuring that our model accurately identifies customers most likely to churn.

1.4 Establishing Method and Setting Up Model - Random Forest

I decided to use Random Forest for this churn prediction problem because it balances accuracy, interpretability, and robustness. Since churn prediction involves a mix of numerical and categorical features, Random Forest handles both effectively without needing heavy preprocessing. Unlike a single decision tree, which can easily overfit, Random Forest builds multiple trees and averages their predictions, making it more stable and reliable. Another big reason I chose it is that it provides feature importance rankings, so I can see which factors—like Credit Score, Balance, or IsActiveMember—actually influence churn the most. Plus, it works well even with imbalanced data, which is common in churn problems. Overall, it gives me a solid mix of performance and explainability, making it a great choice for this dataset.

Random Forest models excel at handling mixed data types, reducing overfitting through ensemble learning, and providing feature importance. However, their ensemble nature can make the model less interpretable, as it's challenging to trace individual predictions. Additionally, they can be computationally intensive, especially on large datasets, and may require extra techniques to effectively manage class imbalances.

1.4.1 Explanation of Steps

In this code snippet, I prepared the data by dropping columns that aren't useful for predicting customer churn, like identifiers (RowNumber, CustomerId, Surname). Then I separated the dataset into two groups: features (X) and the target variable (Exited).

I split the dataset into a training set and a testing set. The training set helps the model learn patterns, while the test set checks how well the model performs on new data.

Next, I built a Random Forest model, specifying class_weight='balanced'. This helps the model handle the fact that the dataset has more customers who stay than those who leave, ensuring the model pays attention to both groups.

Finally, I calculated and displayed the feature importance scores. These scores show which features, like Age, Balance, or EstimatedSalary, most strongly influence a customer's decision to leave or stay. This information is valuable because it helps us understand customer behavior and focus on the factors that really matter.

[25]:
```
# Prepare the dataset for modeling by dropping columns that are not used for␣
 ↪prediction
df_model = df.drop(['RowNumber', 'CustomerId', 'Surname'], axis=1)


# Separate features and target variable
X = df_model.drop('Exited', axis=1)
y = df_model['Exited']

# Split the dataset into training and testing sets
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)

# Build the Random Forest model
from sklearn.ensemble import RandomForestClassifier

rf_model = RandomForestClassifier(n_estimators=100, random_state=42,␣
 ↪class_weight = 'balanced') #addresses imbalance w class weight
rf_model.fit(X_train, y_train)

# Display feature importances
importances = rf_model.feature_importances_
feature_names = X.columns
feat_importances = pd.Series(importances, index=feature_names)
print("\nFeature Importances:")
print(feat_importances.sort_values(ascending=False))
```

```
Feature Importances:
Age                0.252308
Balance            0.140746
```

```
EstimatedSalary      0.138442
CreditScore          0.134498
NumOfProducts        0.133740
Tenure               0.079454
IsActiveMember       0.037735
Geography_Germany    0.029164
Gender               0.021121
HasCrCard            0.018646
Geography_Spain      0.014145
dtype: float64
```

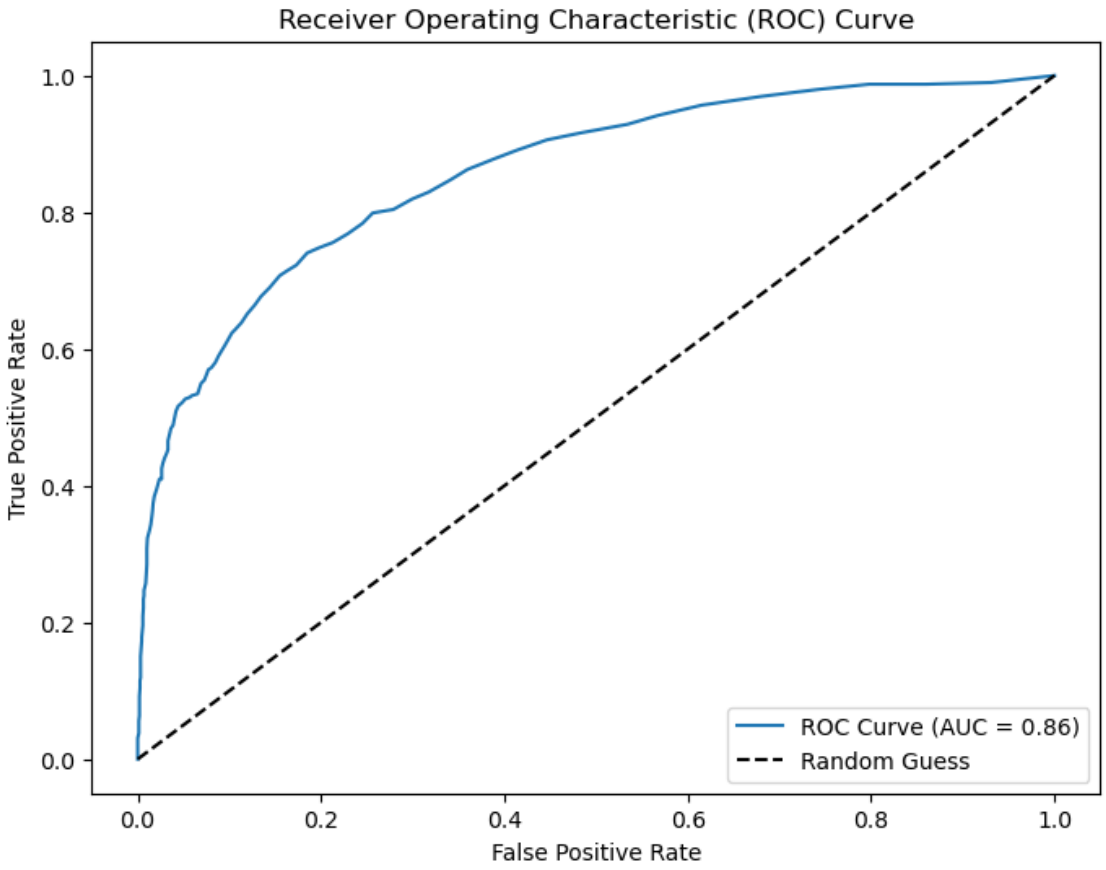
1.4.2 Evaluation of Model - Steps

In this section, I evaluated the model by first predicting churn on the test data and then calculating its accuracy, which tells me how often the model predicted correctly. I also created a detailed classification report, providing insights into how precisely the model identifies churned customers versus those who stay. Additionally, the confusion matrix helps visualize where the model correctly or incorrectly classified customers. Finally, I plotted a ROC curve and calculated the AUC score to measure how effectively the model distinguishes between customers likely to churn and those likely to stay. These evaluation metrics give a clear, detailed picture of my model's performance beyond just simple accuracy.

[27]:
```
from sklearn.metrics import accuracy_score, classification_report,␣
 ↪confusion_matrix, roc_auc_score, roc_curve

# Make predictions on the test set
y_pred = rf_model.predict(X_test)

# classification report + accuracy
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
print("\nClassification Report:")
print(classification_report(y_test, y_pred))

# confusion matrix
conf_matrix = confusion_matrix(y_test, y_pred)
print("Confusion Matrix:\n", conf_matrix)

# roc curve
y_pred_proba = rf_model.predict_proba(X_test)[:, 1]
fpr, tpr, thresholds = roc_curve(y_test, y_pred_proba)
auc = roc_auc_score(y_test, y_pred_proba)

plt.figure(figsize=(8,6))
plt.plot(fpr, tpr, label=f'ROC Curve (AUC = {auc:.2f})')
plt.plot([0, 1], [0, 1], 'k--', label='Random Guess')
plt.xlabel("False Positive Rate")
plt.ylabel("True Positive Rate")
```

```
plt.title("Receiver Operating Characteristic (ROC) Curve")
plt.legend(loc='lower right')
plt.show()
```

```
Accuracy: 0.8685

Classification Report:
              precision    recall  f1-score   support

           0       0.88      0.97      0.92      1607
           1       0.78      0.47      0.58       393

    accuracy                           0.87      2000
   macro avg       0.83      0.72      0.75      2000
weighted avg       0.86      0.87      0.86      2000

Confusion Matrix:
 [[1554   53]
 [ 210  183]]
```



1.4.3 Analysis of Results

Based on these results, the Random Forest model achieved an overall accuracy of approximately 86.85%, indicating strong overall predictive capability. However, the detailed classification report reveals some weaknesses. The precision for identifying churned customers (label 1) is relatively good at 78%, meaning when the model predicts churn, it's usually correct. However, the recall is only 47%, meaning the model identifies less than half of actual churn cases, missing many customers who might leave.

The confusion matrix confirms this, showing the model misclassified 210 actual churn cases as staying customers. The ROC Curve, with an AUC score of 0.86, demonstrates the model is quite effective at distinguishing between churned and retained customers overall, significantly better than random guessing. However, improving the recall for churned customers should be prioritized to better capture at-risk customers and reduce customer loss.

1.5 Summary

When I started exploring this dataset, I knew banks struggled with retaining customers, especially in the competitive digital landscape. Initially, it seemed straightforward: customers either stay or leave. But as I dug deeper into the numbers, a clearer story began to emerge. Customers didn't just randomly leave; specific traits, especially their age, financial balances, and how actively they engaged with the bank, played a crucial role. Using a Random Forest model, I learned that older customers or those with higher balances often behaved differently than younger customers or those with fewer financial resources. My model could confidently identify many customers likely to stay, but it sometimes struggled to spot customers on the verge of leaving. This insight showed me the importance of looking beyond just overall accuracy. Real-world predictions demand sensitivity toward those few customers who quietly slip away, as they're the key to a bank's long-term success.

1.5.1 Impact

This project carries important implications, both socially and ethically. Banks using models like this can proactively identify and retain at-risk customers, improving financial stability for individuals who might be unaware of their financial vulnerability. However, there's a risk of unintended bias—such as favoring wealthier customers who are less likely to churn, while potentially neglecting younger or lower-income customers. This could unintentionally reinforce existing inequalities in banking access and financial security. Thus, it's crucial for institutions to approach predictive modeling responsibly, ensuring transparency and fairness, and being mindful of potential ethical consequences, especially when decisions based on these predictions significantly impact individuals' financial well-being.

1.6 References

Badole, Saurabh . "Banking Customer Churn Prediction Dataset." Kaggle.com, 2024, www.kaggle.com/datasets/saurabhbadole/bank-customer-churn-prediction-dataset.