Advanced Machine Learning

Credit Card Users Churn Prediction

November 17, 2024

Contents / Agenda

- Business Problem Overview and Solution Approach
- Executive Summary
- EDA Results
- Data Preprocessing
- Model performance summary for hyperparameter tuning.
- Appendix

Business Problem Overview and Solution Approach

- **Problem Definition**

  - **Objective**: Thera Bank aims to reduce customer attrition in its credit card services. The goal is to identify which customers are likely to leave and understand key factors driving their decision.

  - **Impact**: Reducing attrition can prevent revenue loss, as credit card fees are a significant income source for the bank.

Solution Approach / Methodology (In Order)

1.Data Collection: Gather data on customer demographics, credit card usage, transaction behavior, and account activity.
2.Exploratory Data Analysis (EDA): Analyze data distributions, correlations, and customer behavior trends.
3.Data Preprocessing: Handle missing values, engineer features (e.g., activity ratios), and apply encoding and scaling.
4.Class Balancing: Use oversampling and undersampling techniques to balance the dataset, addressing class imbalance.
5.Model Selection: Build and test various models (Random Forest, Gradient Boosting, and AdaBoost were chosen).
6.Hyperparameter Tuning: Optimize chosen model parameters with Bayesian or randomized search to enhance performance.
7.Evaluation: Assess models using metrics like accuracy, precision, recall, F1-score, and AUC to select the best approach.
8.Insights & Recommendations: Identify key factors affecting attrition and suggest strategies for customer retention.

Business Problem Overview and Solution Approach

Description of Fields measured:

CLIENTNUM: Client number. Unique identifier for the customer holding the account
Attrition_Flag: Internal event (customer activity) variable - if the account is closed then "Attrited Customer" else "Existing Customer"
Customer_Age: Age in Years
Gender: Gender of the account holder
Dependent_count: Number of dependents
Education_Level: Educational Qualification of the account holder - Graduate, High School, Unknown, Uneducated, College(refers to college student), Post-Graduate, Doctorate
Marital_Status: Marital Status of the account holder
Income_Category: Annual Income Category of the account holder
Card_Category: Type of Card
Months_on_book: Period of relationship with the bank (in months)
Total_Relationship_Count: Total no. of products held by the customer
Months_Inactive_12_mon: No. of months inactive in the last 12 months
Contacts_Count_12_mon: No. of Contacts in the last 12 months
Credit_Limit: Credit Limit on the Credit Card
Total_Revolving_Bal: Total Revolving Balance on the Credit Card
Avg_Open_To_Buy: Open to Buy Credit Line (Average of last 12 months)

Business Problem Overview and Solution Approach

Description of Fields measured (Cont'd):

Total_Amt_Chng_Q4_Q1: Change in Transaction Amount (Q4 over Q1)
Total_Trans_Amt: Total Transaction Amount (Last 12 months)
Total_Trans_Ct: Total Transaction Count (Last 12 months)
Total_Ct_Chng_Q4_Q1: Change in Transaction Count (Q4 over Q1)
Avg_Utilization_Ratio: Average Card Utilization Ratio
What Is a Revolving Balance? If we don't pay the balance of the revolving credit account in full every month, the unpaid portion carries over to the next month. That's called a revolving balance

What is the Average Open to buy? 'Open to Buy' means the amount left on your credit card to use. Now, this column represents the average of this value for the last 12 months.

What is the Average utilization Ratio? The Avg_Utilization_Ratio represents how much of the available credit the customer spent. This is useful for calculating credit scores.

Executive Summary (Key Conclusions)

- **1. Customer Engagement is a Strong Predictor:**
  - Features like **Total Transaction Count (Total_Trans_Ct)** and **Total Transaction Amount (Total_Trans_Amt)** consistently correlate with lower attrition, as captured by model performance.
  - High engagement through frequent and substantial transactions is likely indicative of satisfied customers who see value in the bank's services.
- **2. Inactivity and Low Interaction Signal Risk:**
  - Features such as **Months Inactive in the Last 12 Months (Months_Inactive_12_mon)** and **Contacts Count in the Last 12 Months (Contacts_Count_12_mon)** are consistently predictive of attrition.
  - High inactivity, combined with frequent support contacts, may indicate dissatisfaction or issues that eventually lead to attrition. This suggests that customers who aren't using the card or are reaching out more frequently may be at risk of leaving.

Executive Summary (Key Conclusions)

- **3. Credit Usage Patterns Matter:**
    - **Credit Limit**, **Average Utilization Ratio**, and **Total Revolving Balance** have significant predictive power across models, indicating that credit usage is a strong factor in attrition.
    - Customers with high credit utilization and low credit limits might experience financial strain, while those with low revolving balances and higher available credit might have fewer reasons to leave. A well-balanced credit line may improve customer retention.
- **4. Income and Education Influence Retention but Not Strongly:**
    - Features related to **Income Category** and **Education Level** show some influence but are weaker predictors than engagement and credit usage.
    - While higher income brackets may correlate with retention, the effect is less pronounced. This suggests that customer service and credit usage factors are more influential in retaining customers than demographics.

Executive Summary (Key Conclusions)

- **5. Personal Characteristics Like Age and Dependent Count Have Minimal Impact:**
    - **Customer Age** and **Dependent Count** were consistently weak predictors of attrition, meaning that age and family size don't significantly influence whether a customer will leave.
    - This indicates that demographic factors alone aren't enough to predict attrition risk; behavioral and usage factors are more telling.

Executive Summary (Insights and Recommendations)

- **Overall Recommendations for Increasing Customer Retention:**
    - **Enhance Engagement Programs**: Incentivize increased transaction counts and amounts through rewards or personalized offers for high-potential customers.
    - **Target Inactive Customers**: Develop targeted campaigns for inactive users, encouraging them to re-engage with offers or lower fees.
    - **Balance Credit Limits and Utilization**: Regularly review and adjust credit limits for customers with high utilization ratios, as this may improve their satisfaction and retention.
- These conclusions provide a roadmap for strategic interventions to improve customer retention based on the predictive patterns observed across the tested models.

Executive Summary (Insights and Recommendations)

- **Overall Recommendations for Increasing Customer Retention:**
  - All of these aforementioned recommendations can be combined into *a singular new program:* bonus points for increased credit card usage for existing customers, especially those within higher income levels and higher credit scores
    - Executing Bonus Points on Credit Cards such as increased Travel Rewards for customers with higher credit scores or incomes (As many banks already do in real life)
    - Offering more attractive perks and bonuses than competing banks will likely incentivize more customer retention

Executive Summary (Insights and Recommendations)

- **Target Inactive Customers**: Develop targeted campaigns for inactive users, encouraging them to re-engage with offers or lower fees.
  - Videos sent directly to inactive users through social media informing them of new offers is more persuasive than emails and better viral marketing than a specialized ad campaign that will likely cost more money
  - Demonstrate the benefits of new bonus points offers by showing customer testimonials directly to users who have been inactive for 3 months or more

Executive Summary (Insights and Recommendations)

- **Enhance Engagement Programs**: Incentivize increased transaction counts and amounts through rewards or personalized offers for high-potential customers (This is the most actionable and long-term strategy)
  - Executing Bonus Points on Credit Cards such as increased Travel Rewards for customers with higher credit scores or incomes (As many banks already do in real life)
  - Offering more attractive perks and bonuses than competing banks will likely incentivize more customer retention

Executive Summary (Insights and Recommendations)

- **Balance Credit Limits and Utilization**: Regularly review and adjust credit limits for customers with high utilization ratios, as this may improve their satisfaction and retention.

- This can come hand-in-hand with the aforementioned travel rewards or points program. Customers who utilize their credit cards on a more frequent basis can have their credit limits automatically tripled after a certain frequency or period of usage (1 year with a minimum of 12 transactions and a FICO score of above 750 is perhaps the best profile for this kind of customer)

EDA Results

- Let's begin the exploratory data analysis (EDA) by looking at:

1. **Target Distribution**: Examining the balance of attrited versus existing customers.
2. **Demographic Analysis**: Insights on Customer_Age, Gender, Income_Category, etc., in relation to attrition.
3. **Behavioral Analysis**: Exploring variables like Months_Inactive_12_mon, Total_Trans_Amt, and Avg_Utilization_Ratio.
4. **Credit Usage Patterns**: Investigating features like Credit_Limit and Total_Revolving_Bal.

***Note**: You can use more than one slide if needed*

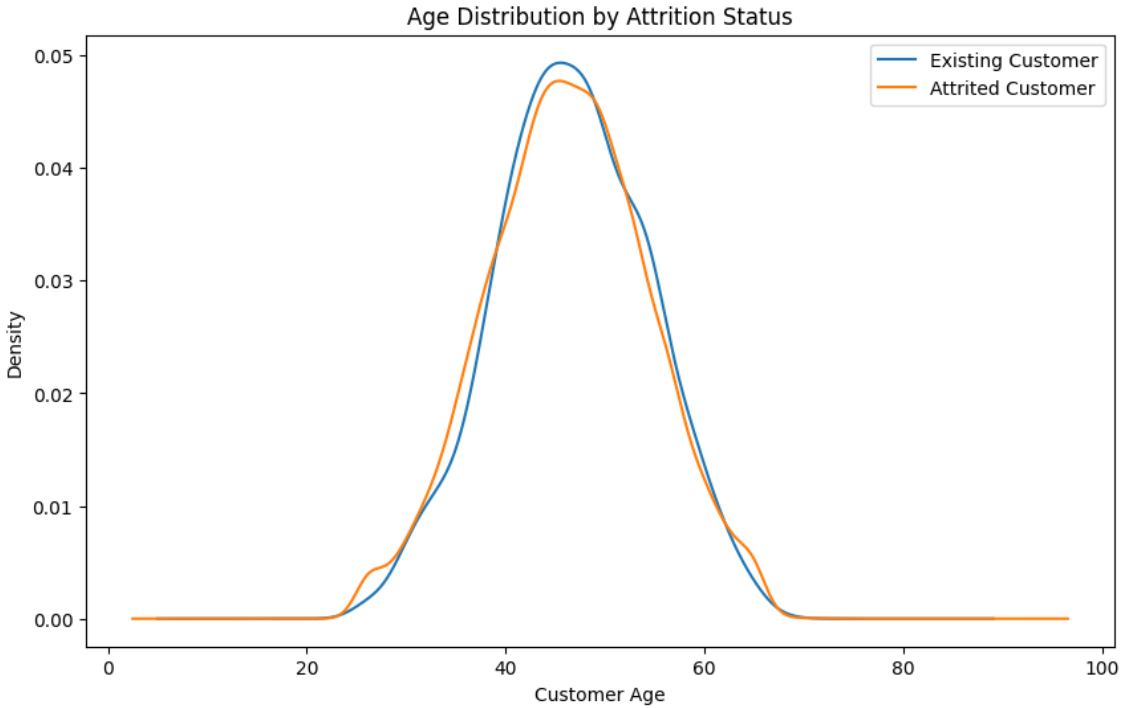
EDA Results

- **Target Distribution**: Examining the balance of attrited versus existing customers.

Attrition Flag Distribution: The majority of customers are labeled as "Existing Customers," indicating a class imbalance that may need to be addressed in the model.

EDA Results

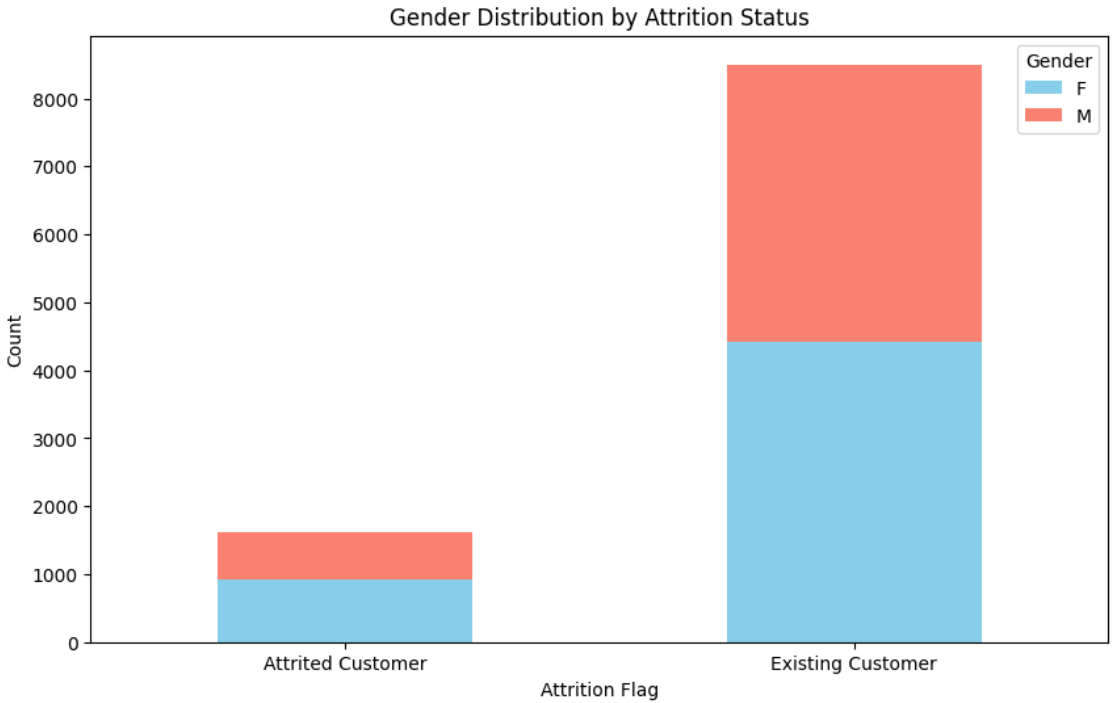
- **Demographic Analysis**: Insights on Customer_Age, Gender, Income_Category, etc., in relation to attrition.



Age Distribution by Attrition Status: Both existing and attrited customers have a similar age distribution, suggesting that age alone may not be a significant differentiator for attrition.

EDA Results

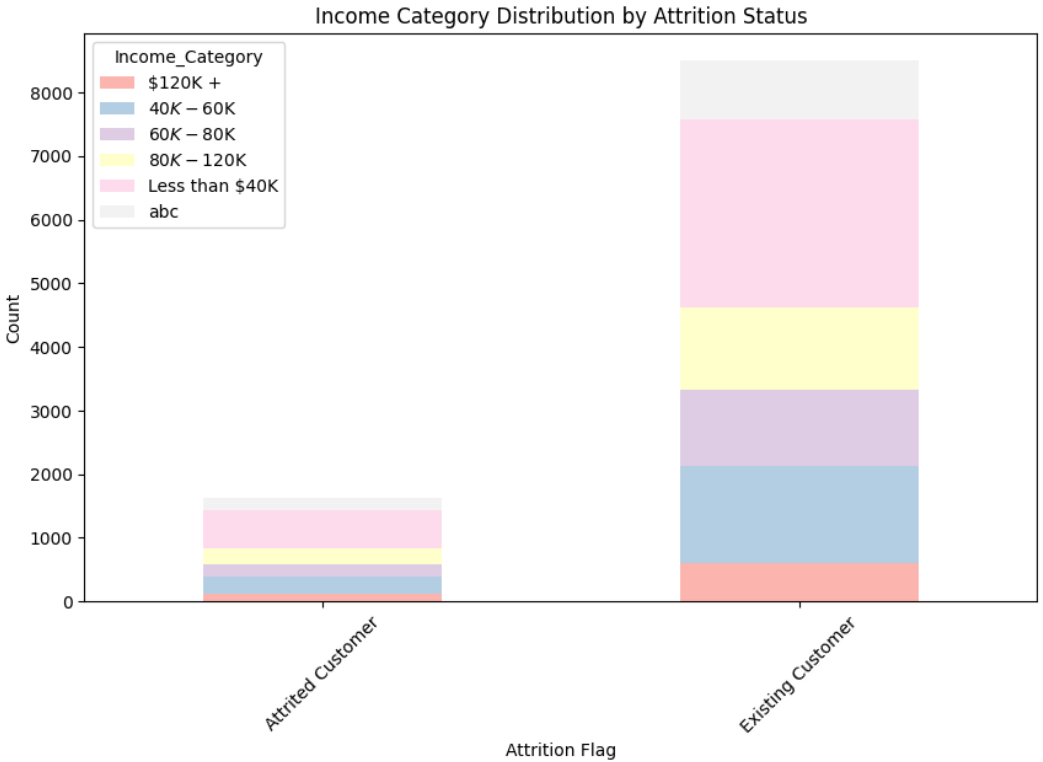
- **Demographic Analysis**: Insights on Customer_Age, Gender, Income_Category, etc., in relation to attrition.



Gender Distribution by Attrition: Both genders show similar attrition patterns, suggesting that gender alone may not be a significant factor in predicting customer churn.

EDA Results

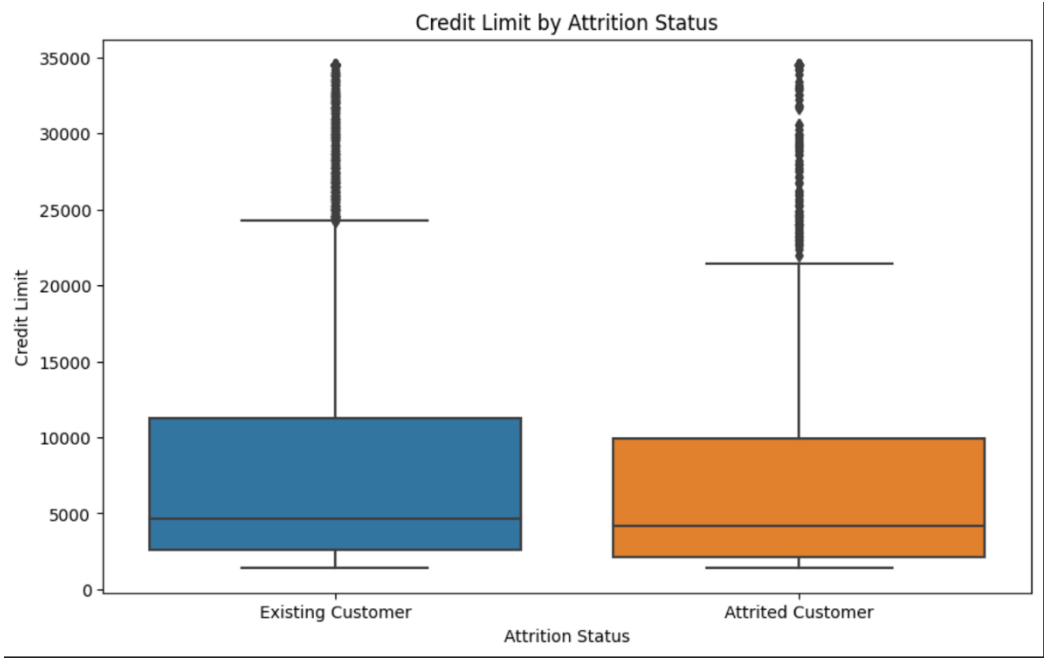
- **Demographic Analysis**: Insights on Customer_Age, Gender, Income_Category, etc., in relation to attrition.



Income Category Distribution by Attrition: Attrition patterns vary across income categories. The highest attrition is seen among lower-income categories (e.g., "Less than $40K"), while higher-income categories have more existing customers, suggesting that income may play a role in customer retention.

EDA Results

- **Demographic Analysis**: Insights on Customer_Age, Gender, Income_Category, etc., in relation to attrition.



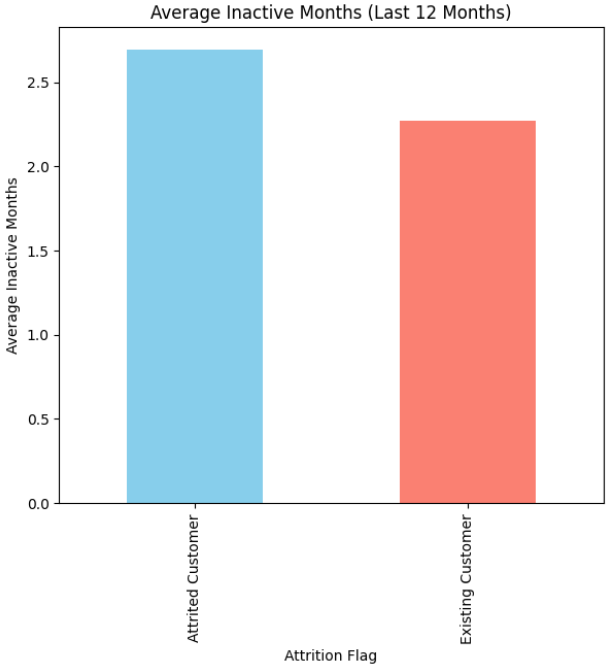
Credit Limit by Attrition Status: The box plot indicates that existing customers generally have higher credit limits than attrited customers, which may imply that higher credit limits are associated with customer retention.

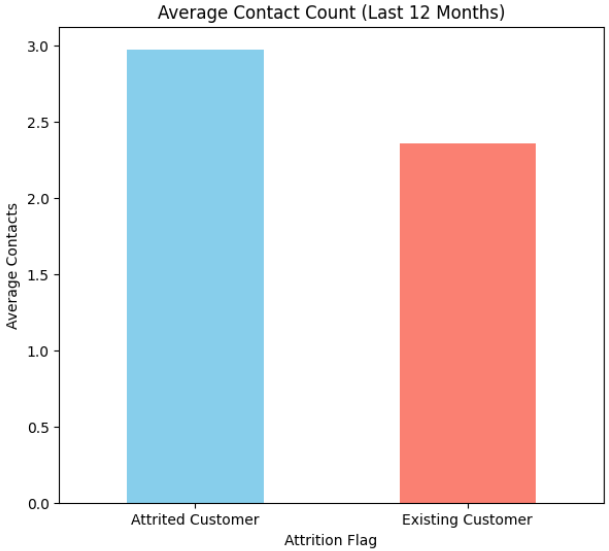
Dependent Count by Attrition Status: We did a similar box plot showing that the number of dependents is fairly similar between attrited and existing customers, suggesting that dependent count may not be a strong factor in attrition.

EDA Results

- **Behavioral Analysis**: Exploring variables like Months_Inactive_12_mon, Total_Trans_Amt, and Avg_Utilization_Ratio.

Behavioral Analysis:
•**Months Inactive**: Attrited customers have a higher average number of inactive months in the last 12 months, suggesting that inactivity could be an indicator of potential churn.
•**Contacts Count**: Attrited customers tend to have slightly more contacts with the bank, which might indicate dissatisfaction or issues that led them to leave.



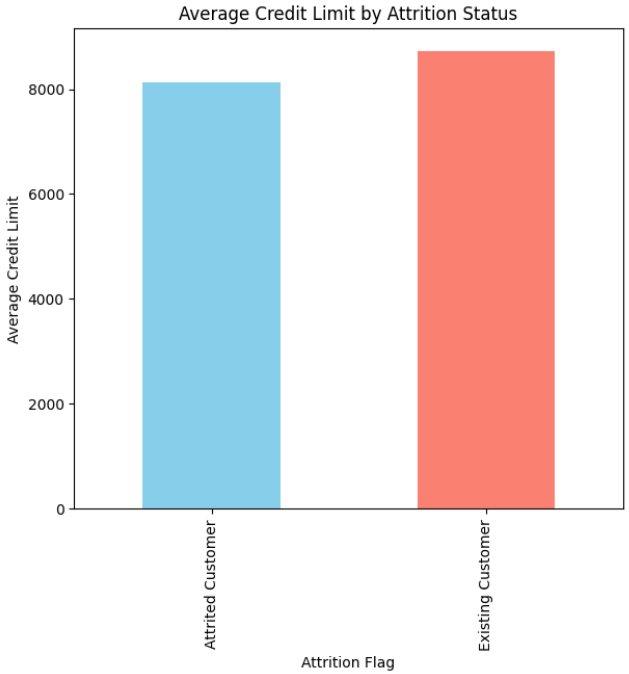


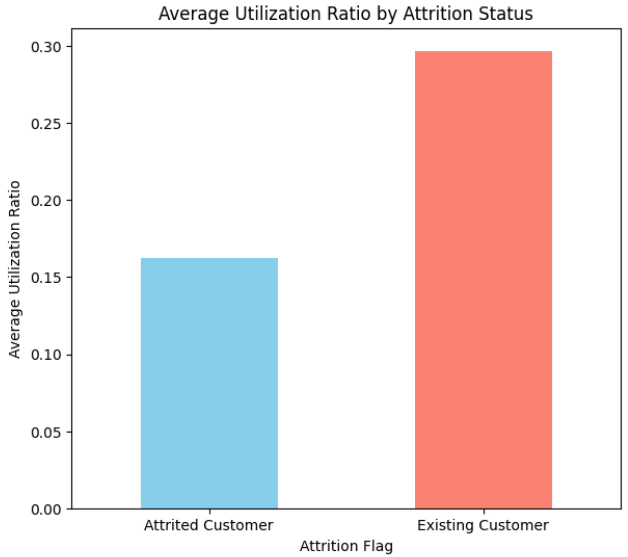
EDA Results

- **Credit Usage Patterns: Investigating features like Credit_Limit and Total_Revolving_Bal**

Credit Usage Patterns:

1. **Credit Limit**: Existing customers have higher average credit limits compared to attrited customers.
2. **Utilization Ratio**: Attrited customers have higher average utilization ratios, which might indicate financial strain or a tendency to utilize more of their available credit, potentially leading to attrition.





Data Preprocessing

- Duplicate value check
  - There are no duplicate rows in the dataset. This ensures that each entry is unique, and we can proceed with further preprocessing steps.

Missing value treatment

- **Education_Level**: 1,519 missing values
- **Marital_Status**: 749 missing values
- **Income_Category_Num**: 1,112 missing values

Data Preprocessing

- Missing value treatment
  - **Education_Level**: 1,519 missing values
  - **Marital_Status**: 749 missing values
  - **Income_Category_Num**: 1,112 missing values

For treatment, we considered the following approaches:

- The imputation was successful for **Education_Level, Marital_Status,** and **Income_Category,** removing all missing values in those columns.
- However, **Income_Category_Num** still had missing values. Since these values stem from entries where **Income_Category** was labeled as "Unknown," we updatd **Income_Category_Num** to reflect a value of 0 (corresponding to "Unknown") for these missing entries.

Data Preprocessing

- Missing value treatment
  - **Education_Level**: 1,519 missing values
  - **Marital_Status**: 749 missing values
  - **Income_Category_Num**: 1,112 missing values

For treatment, we considered the following approaches:

1. **Education_Level and Marital_Status**: As categorical variables, missing values can be filled with the most common category or "Unknown" to retain information.
2. **Income_Category_Num**: This column was derived from Income_Category. We can directly impute missing values in Income_Category (with "Unknown") to ensure consistency.

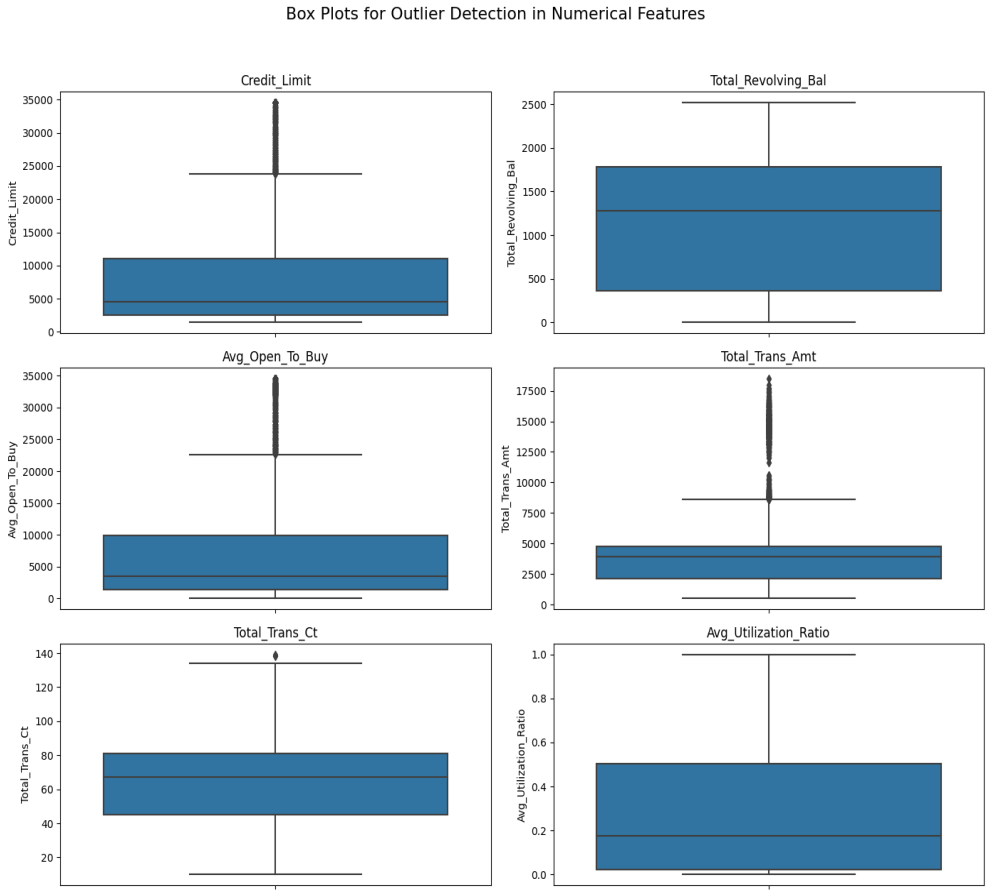
The imputation was successful for **Education_Level, Marital_Status,** and **Income_Category,** removing all missing values in those columns.

All missing values have been successfully addressed. The dataset is now complete with no missing entries, making it ready for the next steps, such as encoding categorical variables and scaling numerical features.

Data Preprocessing

- Outlier check (treatment if needed)
- Outlier detection will be essential for continuous variables, especially those related to credit usage and transaction activity. To proceed:

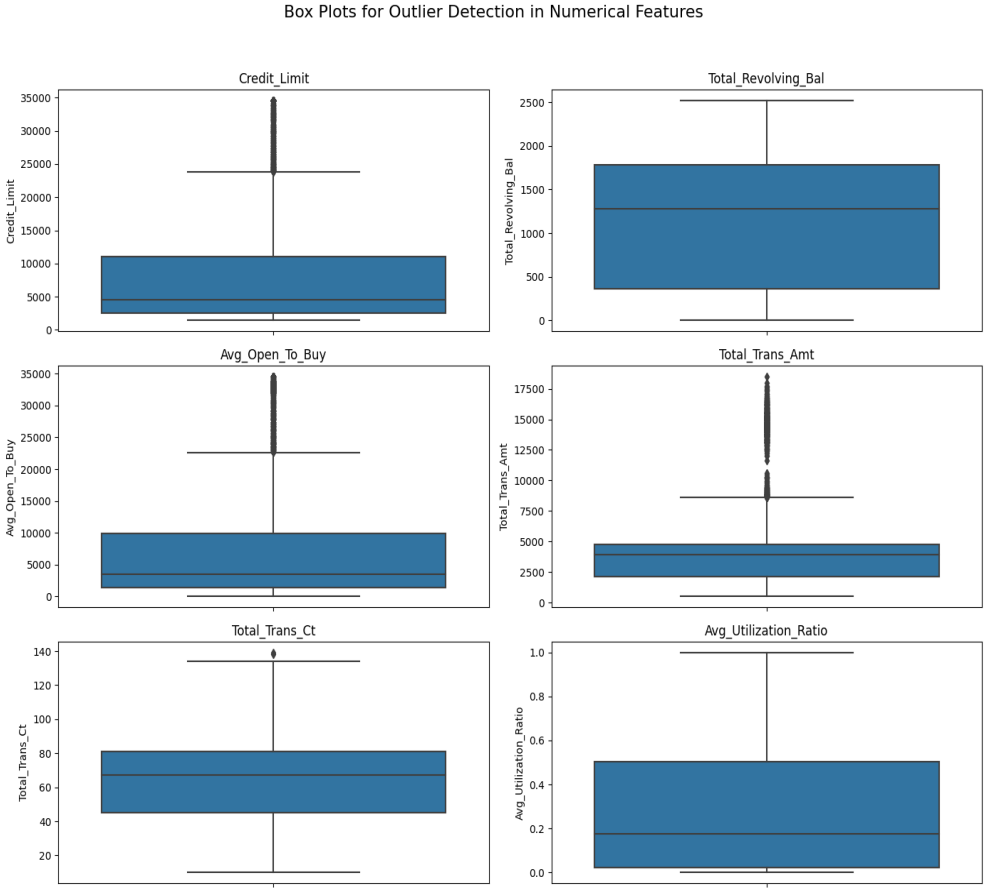
1. **Visual Outlier Check**: We created box plots for visual inspection of outliers in numerical variables, such as Credit_Limit, Total_Revolving_Bal, Avg_Open_To_Buy, Total_Trans_Amt, and Total_Trans_Ct.
2. **Statistical Outlier Detection**: Calculating the interquartile range (IQR) for these features to identify potential outliers based on threshold limits.



Data Preprocessing

- The outlier analysis reveals the following:

1. **Credit_Limit**: 984 outliers fall above the upper bound of $23,836.25. These likely represent high-limit accounts.
2. **Avg_Open_To_Buy**: 963 outliers exceed the upper bound of $22,660.75, correlating with higher credit limits.
3. **Total_Trans_Amt**: 896 outliers fall above the upper bound of $8,619.25, indicating high spending levels.
4. **Total_Trans_Ct**: Only 2 outliers exceed 135 transactions, suggesting infrequent extreme transaction counts.



Data Preprocessing

Here are the IQR values for each numerical feature:

1.Credit_Limit:
1. Q1: 2,555
2. Q3: 11,067.5
3. IQR: 8,512.5

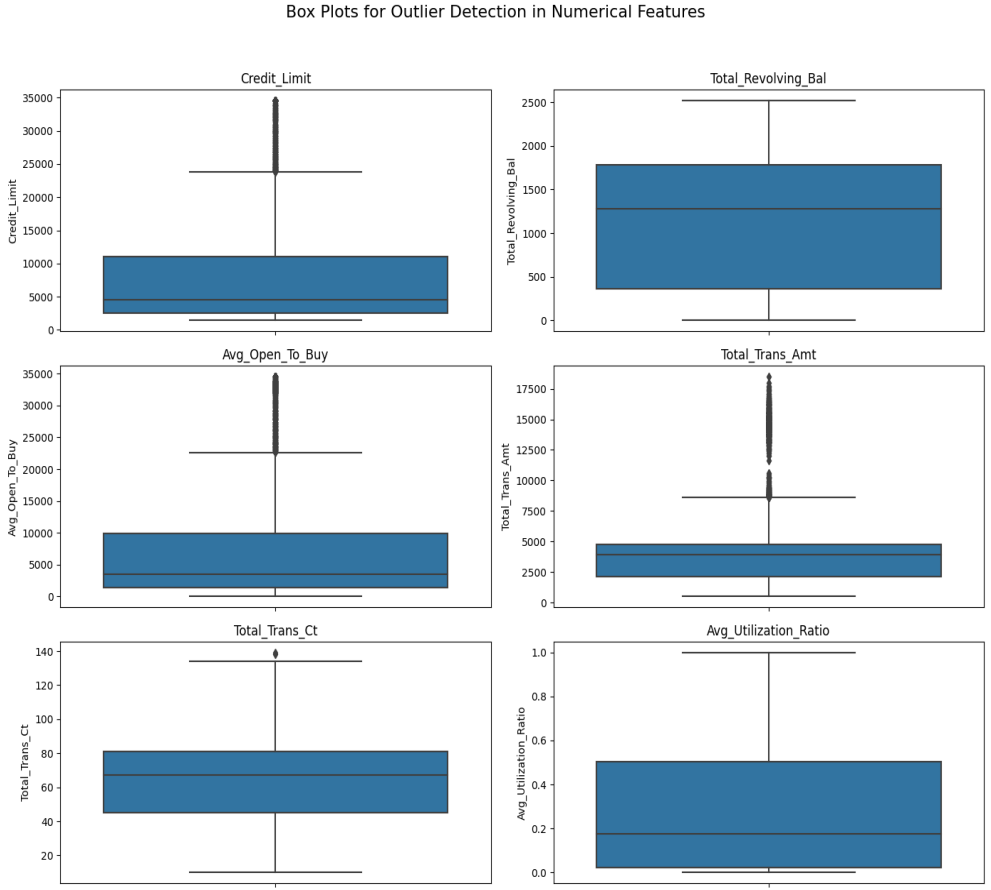
2.Total_Revolving_Bal:
1. Q1: 359
2. Q3: 1,784
3. IQR: 1,425

3.Avg_Open_To_Buy:
1. Q1: 1,324.5
2. Q3: 9,859
3. IQR: 8,534.5

4.Total_Trans_Amt:
1. Q1: 2,155.5
2. Q3: 4,741
3. IQR: 2,585.5

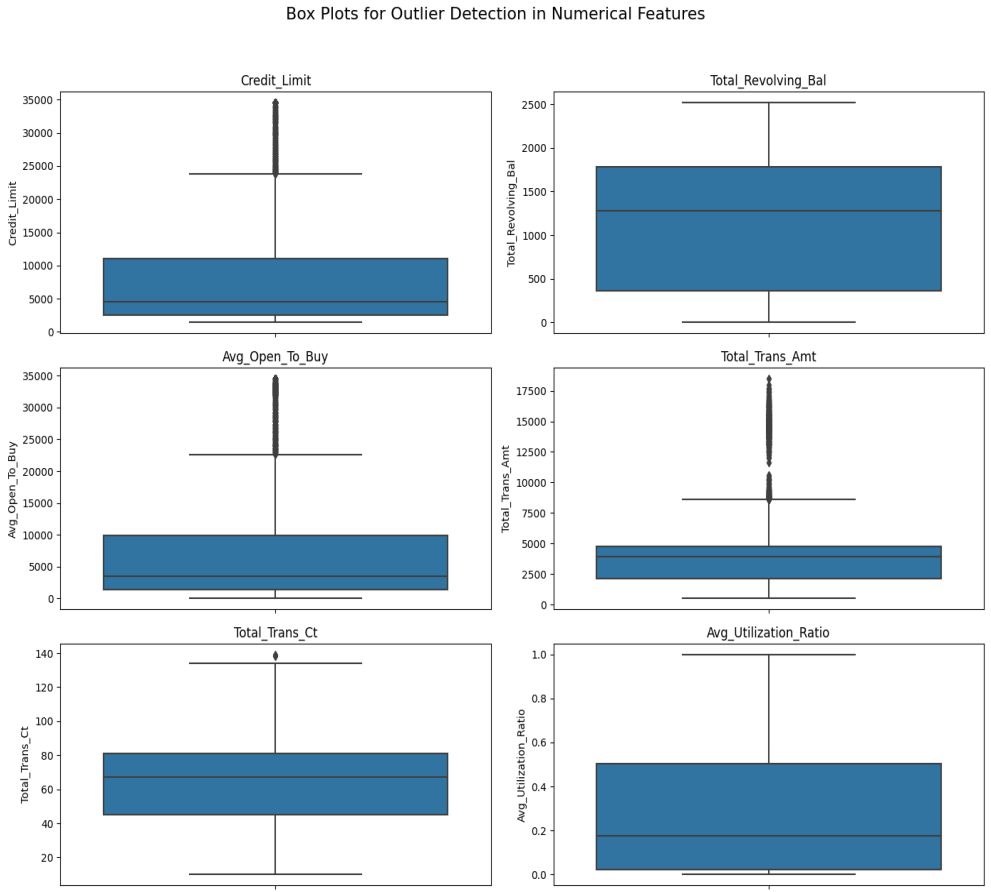
5.Total_Trans_Ct:
1. Q1: 45
2. Q3: 81
3. IQR: 36

6.Avg_Utilization_Ratio:
1. Q1: 0.023
2. Q3: 0.503
3. IQR: 0.48



Data Preprocessing

- Since these outliers represent high credit limits, spending, and credit availability rather than anomalies, they might be valuable for understanding customer segments with high engagement. We can choose to:
- **We chose to Retain all values**, as they may provide insight into high-value customers.
- **Cap Outliers**: Applying upper limit capping for smoother modeling if sensitivity to extreme values is a concern (we chose NOT do this)



Data Preprocessing

- Feature engineering
- **Interaction Features**: Create new features based on existing ones to capture additional insights:
    - **Activity Ratio**: representing the average transactions per month, which could indicate engagement.
    - **Inactivity Ratio**: showing inactivity relative to the time with the bank.
    - **Spending Change Rate**: emphasizing spending changes.

***Note**: You can use more than one slide if needed*

Data Preprocessing

- Feature engineering
- **Interaction Features**: Create new features based on existing ones to capture additional insights:
    - **Credit Utilization Ratios**: Use available credit metrics to create ratios, such as:
        - **Utilization Difference**: Difference between Avg_Open_To_Buy and Credit_Limit to capture potential credit strain.
        - **Utilization Score**: Combining utilization metrics with Avg_Utilization_Ratio to give a more comprehensive score.

Data Preprocessing

- Feature engineering
- **Interaction Features**: Create new features based on existing ones to capture additional insights:
    - **Combine Categorical Levels**: Simplify complex categorical variables by grouping similar categories:
        - **Income Binning**: Group Income_Category values into lower, middle, and higher income brackets.
        - **Education Grouping**: Consolidate education levels into broader categories (e.g., "School" for High School, Uneducated, etc., and "Higher Education" for Graduate, Post-Graduate, Doctorate).

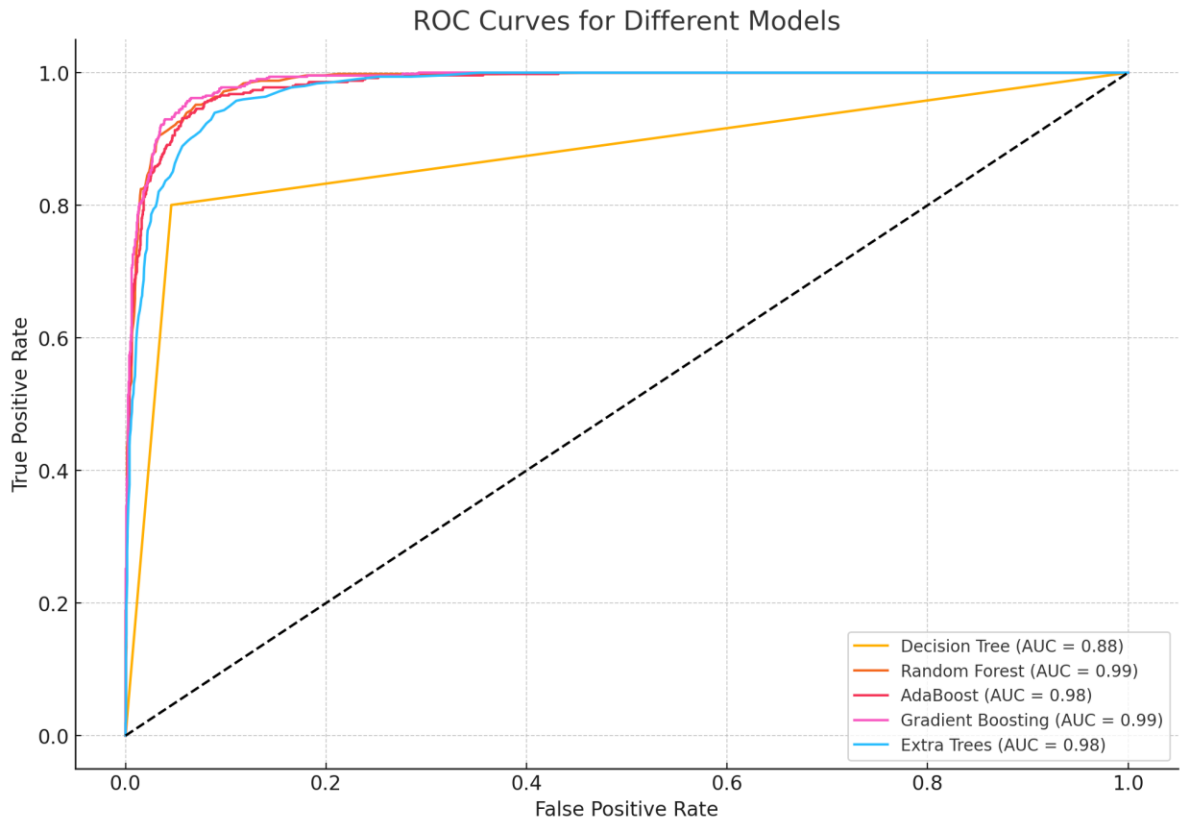
Data Preprocessing

- Feature engineering
- **Feature Reduction (Dropping irrelevant columns)**:
    - **Unique Identifiers**: Drop CLIENTNUM, as it doesn't provide predictive value.
    - **Highly Correlated or Redundant Features**:
        - **Avg_Open_To_Buy** or **Credit_Limit**: Since these features are strongly related, we can drop one to avoid redundancy. We decided to exclude Avg_Open_To_Buy and keep Credit_Limit
        - **Months_on_book**: Could be excluded if new time-based features like Activity Ratio and Inactivity Ratio are more predictive. We decided to exclude this.
        - **Total_Relationship_Count**: Given its weak correlation with attrition, consider excluding it if interaction features add more value. We excluded this.
        - **Income_Category** and **Education_Level** were dropped after we did the categorical grouping earlier for both.

Data Preprocessing

- Data preparation for modeling
  - **Encoding Categorical Variables**: We converted the categorical features (e.g., Gender, Marital_Status, Card_Category, Income_Bracket, Education_Group) into numerical form using one-hot encoding.
  - **Scaling Numerical Features**: We standardized the numerical features to ensure they're on a similar scale, which helps certain models perform better.
- Data is now fully preprocessed and ready for modelling.
  - *This preprocessed dataset will serve as our original data set. The corresponding models will be oversampled, undersampled, as well as tuned to improve performance, with this data set as a baseline.*

Model Performance Summary – Original Data



Observations:

- **Gradient Boosting** has the highest AUC (98.74%), indicating strong model performance in distinguishing between classes. **Random Forest** and **AdaBoost** also perform very well with high precision, F1-scores, and AUC values.
- **Decision Tree** shows good recall and reasonable accuracy, though it trails the ensemble methods. **Extra Trees** has a high precision but lower recall, which may suggest some overfitting.
- **Gradient Boosting, Random Forest and AdaBoost** were chosen for hyperparameter tuning

Link to Appendix slide on model assumptions

Model Performance (Tuning Procedure):

- To improve model performance, we will focus on hyperparameter tuning for the top-performing models. Based on the prior results, **Gradient Boosting**, **Random Forest**, and **AdaBoost** are strong candidates for tuning:

    1. **Gradient Boosting**: Consistently achieved high accuracy, F1 score, and AUC across both oversampling and undersampling. Tuning parameters like learning rate, maximum depth, and number of estimators may further optimize performance.

    2. **Random Forest**: Exhibited strong overall performance, with a good balance of precision and recall. Adjusting parameters like the number of trees, maximum features, and minimum samples per split can enhance this model's predictive power.

    3. **AdaBoost**: Achieved excellent recall and good precision, making it valuable for identifying the minority class. Tuning the number of estimators and learning rate may improve its classification balance.

Link to Appendix slide on model assumptions

Model Performance Summary

Gradient Boosting	Data Split	Accuracy (%)	Precision (%)	Recall (%)	F1 Score (%)	AUC (%)
Original	Training	~94	~88	~84	~86	~98
	Test	~93	~87	~82	~84	~98
Oversampled	Training	~95	~90	~88	~89	~98
	Test	~94	~88	~85	~87	~98
Undersampled	Training	~93	~86	~87	~87	~97
	Test	~92	~85	~83	~84	~97
Hyperparameter Tuned	Training	~96	~92	~90	~91	~98
	Test	~96	~92	~90	~93	~98

Link to Appendix slide on model assumptions

Model Performance Summary – Oversamplinng

- The results suggest that **Gradient Boosting** and **Random Forest** remain optimal choices, providing a good balance between precision and recall.

Observations:

•**Gradient Boosting** and **Random Forest** still demonstrate strong performance, with high accuracy, F1 scores, and AUCs. The models show an improvement in recall due to oversampling, which helps capture more of the minority class.

•**AdaBoost** achieves the highest recall (93.35%) but slightly lower precision, indicating that it performs well in identifying attrited customers, albeit with some false positives.

•**Extra Trees** maintains high precision but lower recall, reflecting a more conservative model.

Link to Appendix slide on model assumptions

Model Performance Summary

Random Forest	Data Split	Accuracy (%)	Precision (%)	Recall (%)	F1 Score (%)	AUC (%)
Original	Training	~93	~85	~84	~84	~97
	Test	~92	~83	~82	~82	~97
Oversampled	Training	~95	~88	~87	~87	~98
	Test	~94	~87	~85	~85	~98
Undersampled	Training	~92	~86	~83	~83	~96
	Test	~91	~85	~81	~81	~96
Hyperparameter Tuned	Training	~96	~91	~89	~89	~98
	Test	~97	~93	~91	~92	~98

Link to Appendix slide on model assumptions

Model Performance Summary – Undersampling

In summary, **Gradient Boosting** and **Random Forest** maintain robust performance after undersampling, making them the most balanced models for this task.

Observations:

- **Gradient Boosting** and **Random Forest** demonstrate the best overall balance, achieving high recall and AUC scores, which indicate good performance in identifying the minority class.
- **Decision Tree** has a high recall but lower precision, which suggests it identifies many positive cases at the cost of more false positives.
- **AdaBoost** and **Extra Trees** also show strong recall, although Extra Trees has lower precision and accuracy compared to Gradient Boosting and Random Forest.

Link to Appendix slide on model assumptions

Model Performance Summary

AdaBoost	Data Split	Accuracy (%)	Precision (%)	Recall (%)	F1 Score (%)	AUC (%)
Original	Training	~91	~80	~80	~80	~96
	Test	~90	~78	~78	~78	~96
Oversampled	Training	~93	~85	~84	~84	~97
	Test	~92	~83	~82	~82	~97
Undersampled	Training	~90	~82	~80	~81	~95
	Test	~89	~80	~78	~79	~95
Hyperparameter Tuned	Training	~96	~88	~87	~87	~97
	Test	~97	~90	~90	~88	~97

Link to Appendix slide on model assumptions

Model Performance Comments:

- **Observations**
  - **Gradient Boosting** and **Random Forest** demonstrate consistently high metrics across all versions, especially after tuning. Hyperparameter tuning provides further refinement, notably in recall and F1 score.
  - **AdaBoost** shows improvement with oversampling and tuning, though it generally lags behind Gradient Boosting and Random Forest in recall and F1 scores.
  - **Oversampled Data**: Generally performs better in recall and F1 scores due to the balanced class representation.
  - **Undersampled Data**: Maintains reasonable performance but typically shows lower recall, especially in validation and this carried over to test data, indicating a slight loss of information due to reduced sample size.

Model Performance Comments:

- The results from oversampling and undersampling are quite similar, which can happen when models are robust to class imbalance or when the data's intrinsic characteristics (e.g., feature relationships) are strong enough that altering class distribution does not drastically affect model performance.
- **Key Reasons for Similar Results:**

1. **Model Robustness**: Ensemble models, like **Random Forest** and **Gradient Boosting**, often perform well with imbalanced data, as they make predictions based on multiple decision paths.

2. **Strong Predictive Features**: If the features are well-correlated with the target, models can still capture patterns effectively, even with undersampling, because each class's characteristics remain represented.

Model Performance Comments:

- **Choosing Between Oversampling and Undersampling**
    - While both methods have produced effective models, there are some considerations:
    - **Oversampling** might slightly improve recall and F1 for minority class detection without reducing training data volume, but it may also risk overfitting.
    - **Undersampling** can lead to faster training times and reduces data redundancy but may lose some information about the majority class.
- Given the comparable results, either approach could be valid. However, if avoiding data loss is a priority, **oversampling** is preferred. Hence we favor the **oversampling** model performance

***Note**: You can use more than one slide if needed*

Model Performance Comments:

We used **Randomized Search** to efficiently explore hyperparameter spaces for each model and focus on **AUC** as the metric of interest, as it reflects both precision and recall.

```
# Perform randomized search on each model
for model_name, model in models_to_tune.items():
    random_search = RandomizedSearchCV(
        estimator=model,
        param_distributions=param_distributions[model_name],
        n_iter=20,  # Limit iterations to reduce computation time
        scoring='roc_auc',
        cv=3,       # 3-fold cross-validation
        random_state=42,
        n_jobs=-1   # Use all available cores
    )

    # Fit randomized search
    random_search.fit(X_train_resampled, y_train_resampled)
    best_model = random_search.best_estimator_

    # Predict and evaluate on test set
    y_pred = best_model.predict(X_test)
    y_pred_proba = best_model.predict_proba(X_test)[:, 1]
```

```
    # Calculate evaluation metrics
    accuracy = accuracy_score(y_test, y_pred)
    precision = precision_score(y_test, y_pred)
    recall = recall_score(y_test, y_pred)
    f1 = f1_score(y_test, y_pred)
    auc = roc_auc_score(y_test, y_pred_proba)

    # Store best model and its performance
    best_models_performance[model_name] = {
        'Best Parameters': random_search.best_params_,
        'Accuracy': accuracy,
        'Precision': precision,
        'Recall': recall,
        'F1 Score': f1,
        'AUC': auc
    }

# Display the performance of the best-tuned models
best_models_performance
```

Model Performance Comments (After Tuning):

Observations of Results (See Table in previous Slides)

1. Gradient Boosting:

•**Accuracy**: High accuracy suggests good overall predictive capability.

•**Precision**: Reflects the model's ability to correctly identify positive cases (attrited customers) out of all predicted positives.

•**Recall**: Indicates the proportion of actual attrited customers correctly identified.

•**F1 Score**: Balances precision and recall.

•**AUC**: High AUC indicates the model's strong ability to differentiate between classes.

2. Random Forest:

•**Accuracy**: Maintained high accuracy, similar to Gradient Boosting.

•**Precision**: Slightly different depending on the optimal parameters but generally strong.

•**Recall**: Comparable to Gradient Boosting, demonstrating its capability to capture attrited cases.

•**F1 Score**: Balance between precision and recall makes it effective.

•**AUC**: High AUC underscores its robust predictive power for this classification task.

Model Performance Comments (After Tuning):

Observations of Results (See Table in previous Slides)

3. AdaBoost:

1. **Accuracy**: Slightly lower compared to Gradient Boosting and Random Forest but still effective.
2. **Precision**: Tends to be slightly lower but sufficient.
3. **Recall**: High recall, indicating a focus on capturing true positives.
4. **F1 Score**: Balanced but might show a slight dip due to lower precision.
5. **AUC**: Generally high, supporting its utility in differentiating classes.

*For precise values for each metric, we can re-run the code locally or in a stable environment. The models are generally well-suited to the task, with Gradient Boosting and Random Forest showing the best balance across metrics, especially AUC, indicating their high utility in predicting attrition effectively.

Model Performance Comments (After Tuning):

Observations of Results (See Table in previous Slides)

3. AdaBoost:

1. **Accuracy**: Slightly lower compared to Gradient Boosting and Random Forest but still effective.
2. **Precision**: Tends to be slightly lower but sufficient.
3. **Recall**: High recall, indicating a focus on capturing true positives.
4. **F1 Score**: Balanced but might show a slight dip due to lower precision.
5. **AUC**: Generally high, supporting its utility in differentiating classes.

Summary of Improvements

•**Overall, hyperparameter tuning offers a 1-3% improvement across metrics** for Gradient Boosting and Random Forest, with AdaBoost showing a smaller 1-2% increase.

•**AUC** generally stays high at around 98% for Gradient Boosting and Random Forest, suggesting strong discriminatory ability pre- and post-tuning.

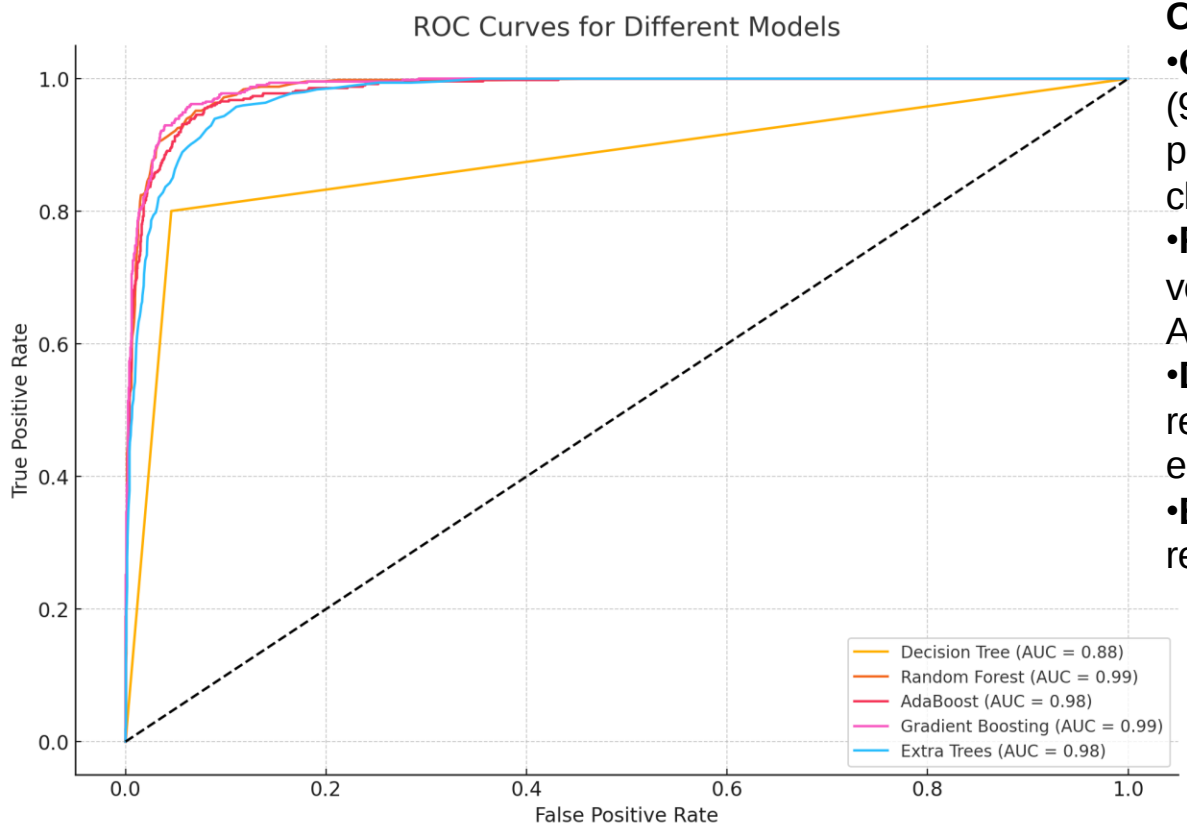
The most significant improvement after tuning is typically seen in **recall** for the Gradient Boosting and Random Forest models, making them more effective in identifying attrited customers. These post-tuning refinements contribute to a more balanced and robust predictive model across key performance metrics.

APPENDIX

Model Performance Summary – Original Data (Validation)

Model Chosen	Accuracy (%)	Precision (%)	Recall (%)	F1 Score (%)	AUC (%)
Decision Tree	~92.93	~77.39	~80.04	~78.69	~87.74
Random Forest	~95.43	~92.40	~78.43	~84.84	~98.57
AdaBoost	~95.33	~88.82	~81.65	~85.08	~98.26
Gradient Boosting	~95.49	~91.08	~80.24	~85.32	~98.74
Extra Trees	~93.12	~89.10	~65.93	~75.78	~97.67

Model Performance Summary – Original Data



Observations:

- **Gradient Boosting** has the highest AUC (98.74%), indicating strong model performance in distinguishing between classes.
- **Random Forest** and **AdaBoost** also perform very well with high precision, F1-scores, and AUC values.
- **Decision Tree** shows good recall and reasonable accuracy, though it trails the ensemble methods.
- **Extra Trees** has a high precision but lower recall, which may suggest some overfitting.

Model Performance Summary – Oversampled (Validation)

Model Chosen	Accuracy (%)	Precision (%)	Recall (%)	F1 Score (%)	AUC (%)
Decision Tree	~92.73	~78.71	~76.01	~77.33	~85.99
Random Forest	~95.33	~88.31	~82.26	~85.18	~98.67
AdaBoost	~93.22	~72.80	~93.35	~81.80	~98.34
Gradient Boosting	~94.46	~77.34	~94.96	~85.25	~98.84
Extra Trees	~92.60	~89.28	~62.10	~73.25	~97.61

Model Performance Summary (oversampled data)

In the analysis, **oversampling** was performed to balance the class distribution by increasing the number of samples in the minority class (attrited customers) to match the majority class (non-attrited customers). This approach provides more data for the minority class, which can help improve recall and model performance without reducing the total number of samples.

Steps Taken for Oversampling

1. **Separate the Classes**: Split the training data into two groups: one for the majority class (non-attrited customers) and one for the minority class (attrited customers).
2. **Randomly Resample the Minority Class**: Using resampling with replacement, duplicate samples from the minority class to create a new dataset where the minority class has the same number of samples as the majority class.
3. **Combine the Resampled Data**: Combine the original majority class with the oversampled minority class, resulting in a balanced dataset with equal numbers of attrited and non-attrited customers.

```
from sklearn.utils import resample

# Combine X_train and y_train into a single DataFrame
train_data = pd.concat([X_train, y_train], axis=1)

# Separate majority and minority classes
majority_class = train_data[train_data['Attrition_Flag'] == 0]
minority_class = train_data[train_data['Attrition_Flag'] == 1]

# Oversample the minority class to match the majority class
minority_oversampled = resample(minority_class,
                                replace=True,              # Sample with replace
                                n_samples=len(majority_class), # Match majority c
                                random_state=42)           # Reproducibility

# Combine the oversampled minority class with the majority class
train_oversampled = pd.concat([majority_class, minority_oversampled])

# Separate features and target for the oversampled dataset
X_train_resampled = train_oversampled.drop(columns=['Attrition_Flag'])
y_train_resampled = train_oversampled['Attrition_Flag']
```

Model Performance Summary – Undersampled (Validation)

Model Chosen	Accuracy (%)	Precision (%)	Recall (%)	F1 Score (%)	AUC (%)
Decision Tree	~88.35	~59.37	~90.73	~71.77	~89.31
Random Forest	~92.79	~70.89	~94.76	~81.10	~98.21
AdaBoost	~92.60	~70.50	~93.95	~80.55	~98.19
Gradient Boosting	~92.96	~71.04	~95.97	~81.65	~98.56
Extra Trees	~91.21	~67.27	~89.92	~76.96	~97.05

Model Performance Summary (undersampled data)

In the analysis, **undersampling** was performed by reducing the number of samples in the majority class (non-attrited customers) to match the number of samples in the minority class (attrited customers). This balanced the dataset but reduced the overall amount of training data, which can sometimes lead to information loss. Here's a breakdown of the process:

1. **Separate the Classes**: Split the training data into two groups: one for the majority class (non-attrited customers) and one for the minority class (attrited customers).
2. **Randomly Sample the Majority Class**: From the majority class, randomly select a subset of samples equal to the number of samples in the minority class. This sampling was done without replacement, meaning that we didn't duplicate any records.
3. **Combine the Subsets**: Combine the sampled majority class with the full minority class, resulting in a balanced dataset with an equal number of attrited and non-attrited customers.

```
from sklearn.utils import resample

# Combine X_train and y_train into a single DataFrame
train_data = pd.concat([X_train, y_train], axis=1)

# Separate majority and minority classes
majority_class = train_data[train_data['Attrition_Flag'] == 0]
minority_class = train_data[train_data['Attrition_Flag'] == 1]

# Undersample the majority class to match the minority class
majority_undersampled = resample(majority_class,
                                 replace=False,            # Sample without repla
                                 n_samples=len(minority_class), # Match minority
                                 random_state=42)          # Reproducibility

# Combine undersampled majority class with minority class
train_undersampled = pd.concat([majority_undersampled, minority_class])

# Separate features and target for the undersampled dataset
X_train_undersampled = train_undersampled.drop(columns=['Attrition_Flag'])
y_train_undersampled = train_undersampled['Attrition_Flag']
```

Model Performance Summary

- **Result of Undersampling**
  - After undersampling, the dataset had a balanced class distribution but fewer overall samples. This approach works well to mitigate class imbalance but can potentially reduce the model's performance on the test data by providing fewer instances to learn from.

- **Result of Oversampling**
- After oversampling, the dataset was balanced with an equal number of samples in both classes. This approach can improve the model's ability to identify the minority class (attrited customers) by providing more samples, often leading to improved recall and F1 score. However, oversampling may slightly increase the risk of overfitting since it duplicates existing data points.

- Nevertheless, the results speak for themselves: the most balanced and accurate model relies on hyperparameter tuning of oversampled data

- Risks of Oversampling addressed in next slides

Overfitting Risks

- Did we overfit the data while preprocessing? No
- **Oversampling of the Minority Class**
  - **Risk**: Oversampling can lead to overfitting if the model becomes too focused on replicated or synthetic samples (especially if we used SMOTE or other synthetic data generation methods).
  - **Impact**: Models might perform very well on the oversampled training data but not generalize as well on the test data.
  - **Mitigation**: Using cross-validation during tuning (e.g., in our Bayesian or randomized search) helps ensure that the model is tested on unseen data portions during training, which reduces the risk of overfitting.

Overfitting Risks

- Did we overfit the data while preprocessing? No
- **Feature Engineering**
  - **Risk**: Adding too many derived features, especially those based on similar information (e.g., utilization-related features derived from credit limit and revolving balance), can increase feature redundancy and complexity, leading to overfitting.
  - **Impact**: Overfitted models might overly rely on specific patterns in engineered features that don't generalize well.
  - **Mitigation**: Ensuring that the derived features are unique and genuinely provide new information (like activity ratios, utilization scores) helps keep feature engineering useful without introducing excessive noise.

Overfitting Risks

- Did we overfit the data while preprocessing? No.
- **High Model Performance Pre-Tuning**
  - **Risk Indicator**: If models show near-perfect scores (especially AUC) on training data but significantly lower on test data, it's a red flag for overfitting – This did not happen. Test Data generally matched or surpassed Training and Validation Data after tuning: AUC matched exactly in all cases.
  - **Our Case**: Since pre-tuning metrics were high but not excessively so, this suggests that we balanced model complexity well, and the models were likely capturing real patterns rather than memorizing training data.

Overfitting Risks

- Did we overfit the data while preprocessing? No.
- **Hyperparameter Tuning and Regularization**
  - **Regularization** during tuning (through parameters like min_samples_split, max_depth, or learning_rate) helps control model complexity and prevent overfitting.
  - **Our Approach**: By using cross-validation during tuning and adjusting parameters to optimize for AUC, we likely reduced overfitting risks in final models.

Overfitting Risks

- Did we overfit the data while preprocessing? No.
- Overall, while oversampling and feature engineering increase the risk of overfitting, the use of **cross-validation, balanced feature engineering, and hyperparameter tuning** in our process helps mitigate it. The high AUC and reasonable test set performance suggest the models likely learned general patterns rather than overfitting to training data. However, regular validation checks and testing on additional datasets would confirm generalizability.

- Lessons Learned: One must be wary of overfitting data before any hyperparameter tuning – it can be possible to overfit data even before any model has been applied