Propensity to Lapse: Predictive Model and Business Implications

A data-driven approach for customer retention strategies

Monday July 31st, 2023
Athens

Presenter: Dimitrios Matsanganis, f2822212
Instructor: Mr. Papanikolaou Panagiotis
Course: Business Analytics Practicum II

Introduction

- ✓ Brief about the loyalty partner business.
- ✓ Define the concept of Lapse, Collections, and Redemptions.
- ✓ Importance of predicting customer churn for the business.

Introduction

- **Loyalty Partner Business Overview**: We are working with a successful business partner that operates a loyalty reward system. Customers accumulate points through various means, such as flights or purchases from affiliated partners like Tesco or Apple. These points can then be redeemed to book flights, hotels, or rent a car.

- **Collections Definition**: A collection refers to the addition of loyalty points to a customer's balance. This can happen when a customer makes a purchase from a partner store or participates in an affiliated program. Every interaction that leads to the accrual of points is considered a 'collection'.

- **Redemptions Definition**: Redemption involves the utilization of the loyalty points from a customer's balance. For example, a customer may redeem a certain number of points to book a flight. The redeemed points are then subtracted from the customer's balance.

- **Concept of Lapse**: A customer is deemed to have 'lapsed' if they have not collected or redeemed any points for a consecutive 12-month period. For instance, if a customer last collected points in April 2015 and then didn't engage in collection or redemption until May 2016, they are categorized as lapsed (Active=0, Lapsed=1).

- **Importance of Predicting Customer Churn**: Understanding and predicting customer churn is crucial for our business partner. It allows us to identify at-risk customers and implement strategies to improve their loyalty. By proactively addressing churn, we can increase customer retention, enhance the customer experience, and ultimately drive revenue growth.

Objective

✓ Description of the two-fold task - building a predictive model and creating a business presentation with findings and suggestions.

Objective

The goal in this project is two-fold, and it is geared towards enabling the marketing team to reduce customer churn effectively.

1. **Building a Predictive Model**: The first part of the task involves creating a machine learning model using the given dataset. This dataset contains various characteristics of customers, including their current lapsing status. Using these features, we aim to build a model that can successfully predict if a customer will lapse (become inactive) in the next 12 months. The model will aid the team in identifying customers at risk of churn, allowing us to implement preventive strategies in a timely manner.

2. **Collect and Present the Findings and Suggestions**: The second part of the task is about translating the findings from the predictive model into actionable insights. Then, present these findings in an accessible format for the marketing team, along with our suggestions for reducing churn rates. These recommendations will be data-driven, grounded in the results of our predictive model. The goal is to provide the marketing team with strategic direction based on rigorous data analysis, contributing to more efficient decision-making and potentially higher customer retention rates.

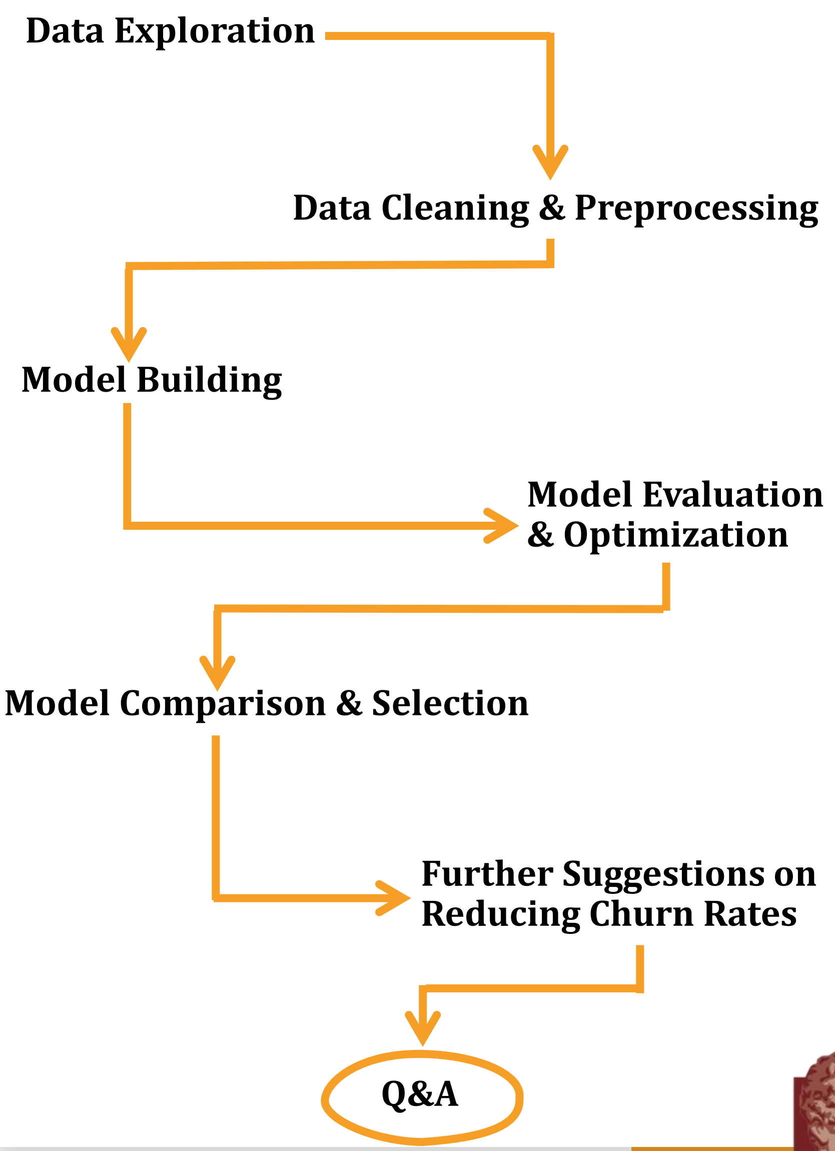
Agenda

✓ This presentation will follow a logical sequence of steps that make up the process of our churn prediction analysis.

Agenda

1. **Data Exploration**: We will begin with an overview of the given dataset. Understanding the nature of our data, exploring each feature, and deriving initial insights will form the base of our subsequent steps.
2. **Data Cleaning & Preprocessing**: This involves cleaning the data and making it suitable for model building. This step ensures that any missing or inconsistent data is handled appropriately, additional features are created if required, and categorical data is converted into numerical ones.
3. **Model Building**: Here, we will select a suitable machine learning algorithm to train ***some*** model on our dataset. The choice of algorithm will depend on the nature and distribution of our data.
4. **Model Evaluation & Optimization**: We will assess our initial models' performance and apply optimization techniques for better accuracy. This includes methods like cross-validation and hyperparameter tuning.
5. **Model Comparison & Selection**: In this stage, we will compare different models to select the most appropriate one for our case.
6. **Further Suggestions on Reducing Churn Rates**: Finally, based on our optimal model's outcomes, we will present the key findings and propose strategies that could help the marketing team devise ways to reduce churn rates.
7. **Q&A**: We will wrap up our presentation with a session for questions and answers, to address any queries or concerns you may have about our analysis or findings.

Roadmap of Our Churn Prediction Analysis



Data Exploration

Data Cleaning & Preprocessing

Model Building

Model Evaluation & Optimization

Model Comparison & Selection

Further Suggestions on Reducing Churn Rates

Q&A

Data Exploration

- ✓ Explanation of the dataset and its features.
- ✓ Data Overview
- ✓ Initial insights derived from the data.

Data Exploration

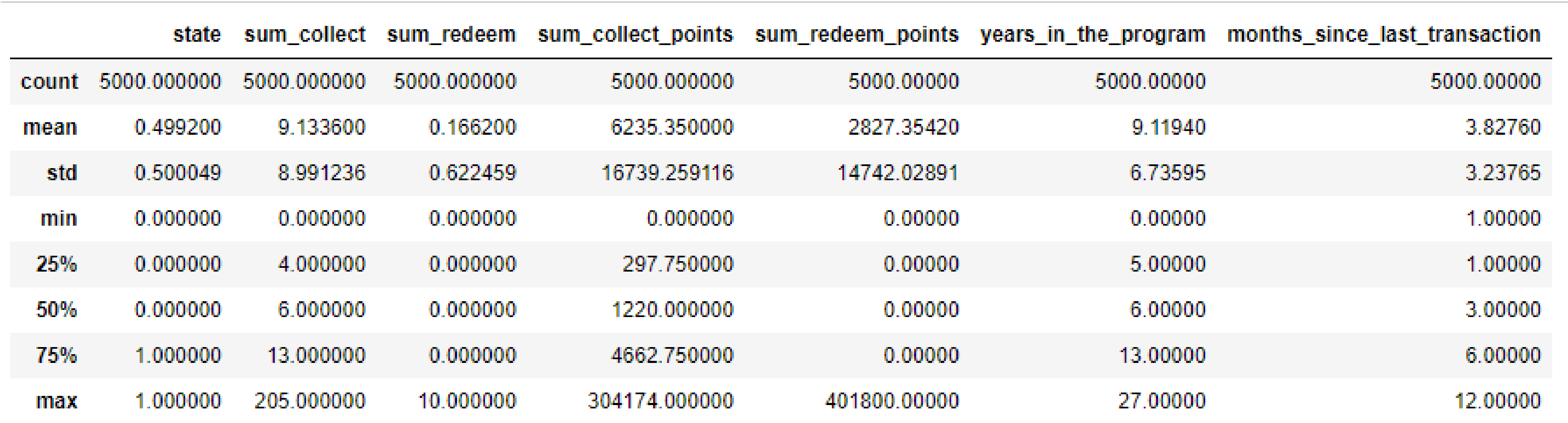
In this section, the dataset will be explored in depth. The features in the dataset include:

- **State**: This is our target variable indicating the lapsed status of a customer.
- **Sum_collect**: The total number of times a customer has collected points.
- **Sum_redeem**: The total number of times a customer has redeemed points.
- **Sum_collect_points**: The total points a customer has collected over their tenure.
- **Sum_redeem_points**: The total points a customer has redeemed.
- **Years_in_the_program**: The total number of years a customer has been registered in the loyalty program.
- **Months_since_last_transaction**: The number of months since a customer's last action (collection or redemption).

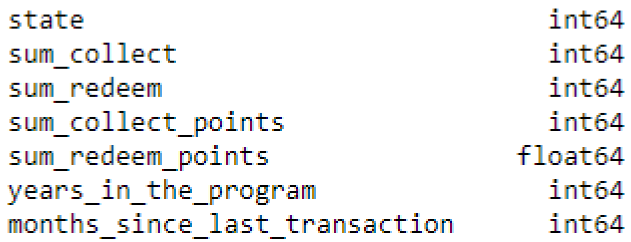
The dataset comprises **5000 observations**, each representing **a unique customer**.

Data Exploration - Data Overview

The dataset comprises **5000 observations and 7 columns**. All dataset's columns are **integers (int64),** except for the *sum_redeem_points* which is a **float type variable (float64)**. Below are presented the analysis results (summary statistics and variables types).



	state	sum_collect	sum_redeem	sum_collect_points	sum_redeem_points	years_in_the_program	months_since_last_transaction
count	5000.000000	5000.000000	5000.000000	5000.000000	5000.00000	5000.00000	5000.00000
mean	0.499200	9.133600	0.166200	6235.350000	2827.35420	9.11940	3.82760
std	0.500049	8.991236	0.622459	16739.259116	14742.02891	6.73595	3.23765
min	0.000000	0.000000	0.000000	0.000000	0.00000	0.00000	1.00000
25%	0.000000	4.000000	0.000000	297.750000	0.00000	5.00000	1.00000
50%	0.000000	6.000000	0.000000	1220.000000	0.00000	6.00000	3.00000
75%	1.000000	13.000000	0.000000	4662.750000	0.00000	13.00000	6.00000
max	1.000000	205.000000	10.000000	304174.000000	401800.00000	27.00000	12.00000



```
state                              int64
sum_collect                        int64
sum_redeem                         int64
sum_collect_points                 int64
sum_redeem_points                float64
years_in_the_program               int64
months_since_last_transaction      int64
```

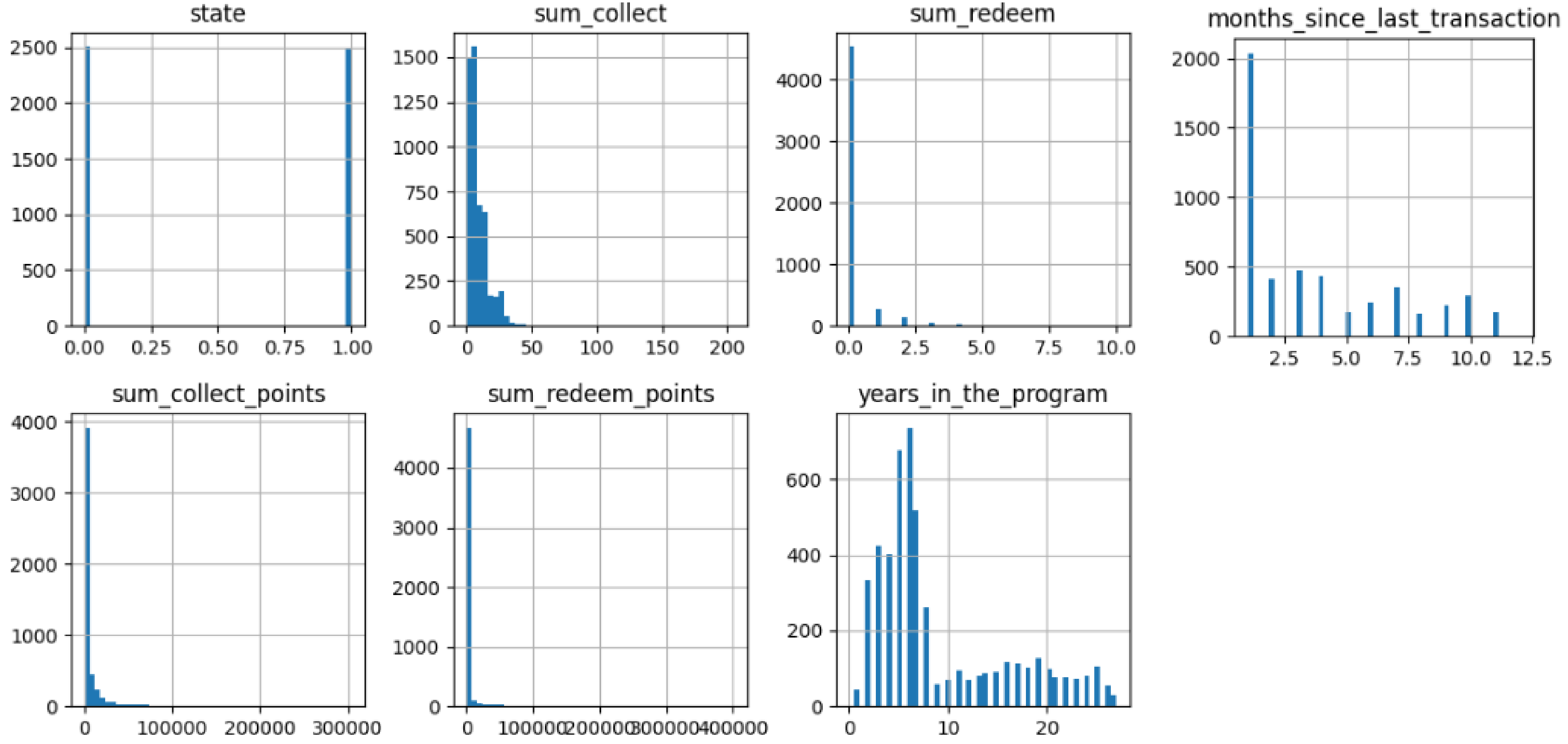
Data Exploration - Data Overview

Based on the previous analysis results, we can summarize the data as follows:

- **State**: This is our *target variable*. The dataset ***is well balanced with nearly equal proportions of active and lapsed customers.***
- **Sum_collect**: The average number of times a customer has collected points is around **9.1**, with significant variation in the data.
- **Sum_redeem**: Customers redeem points less frequently on average (mean around **0.166**).
- **Sum_collect_points**: On average, a customer collects about **6235 points**. However, there's a **high variation** in point collection.
- **Sum_redeem_points**: The average points redeemed by a customer are about **2827**, with **high variation**.
- **Years_in_the_program**: Customers have been in the program for an average of approximately **9.1 years**. The **loyalty program has a mix of new and long-term customers**.
- **Months_since_last_transaction**: On average, approximately **3.8 months have passed since a customer's last transaction**.

The dataset covers a broad range of customer behavior, from those who are new and relatively inactive to those who are highly engaged and have been in the program for many years.

Data Exploration - Univariate Analysis



Data Exploration - Univariate Analysis

Based on the previous slide, all variables, except the binary state variable, **follow a right-skewed distribution**.

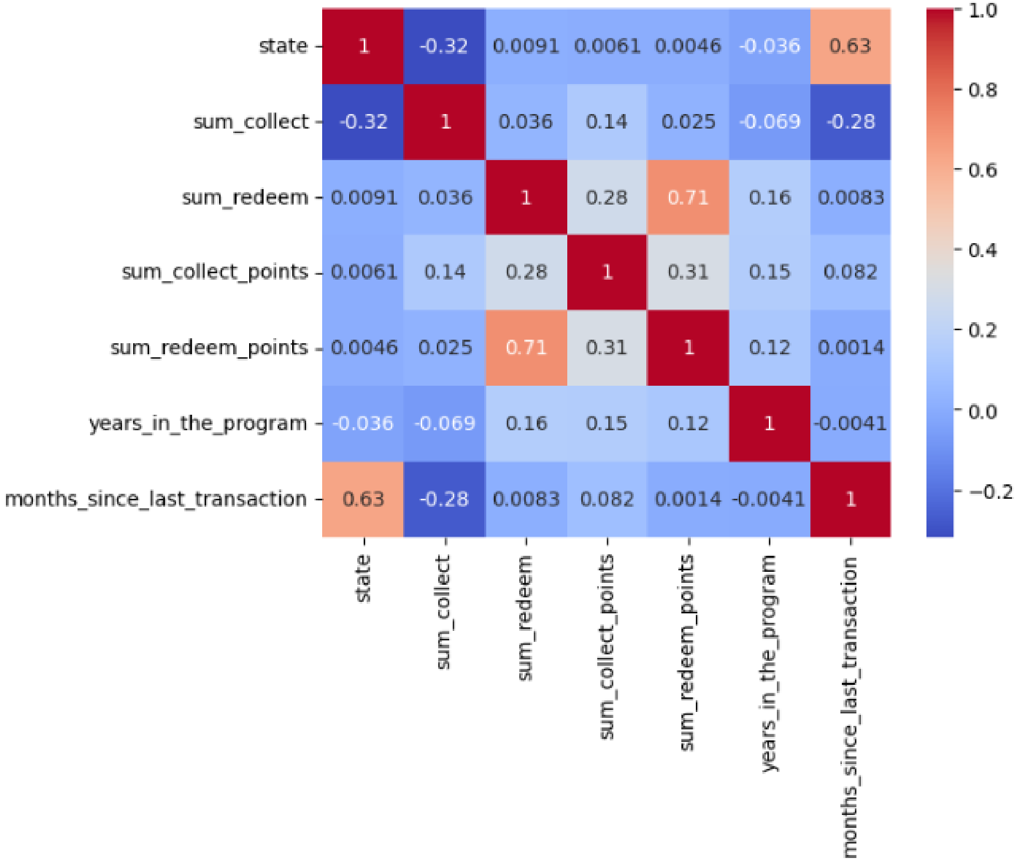
This skewness is reflected by the means being larger than the median values for Sum_collect, Sum_collect_points, Sum_redeem_points, Years_in_the_program, and Months_since_last_transaction. A large difference between the maximum value and the 75% percentile could indicate **a long tail to the right**.

For example, in **Sum_collect**, the mean is 9.13 while the median is 6, suggesting a right-skewed distribution with a tail of customers who collect points many more times than average.

Similarly, Sum_collect_points has a mean of 6235 and a median of 1220, indicating a right-skewed distribution with some customers collecting significantly more points than the average customer.

Data Exploration - Bivariate Analysis

For this analysis, a correlation heatmap was generated to understand the relationships between different features of the dataset.



	state	sum_collect	sum_redeem	sum_collect_points	sum_redeem_points	years_in_the_program	months_since_last_transaction
state	1	-0.32	0.0091	0.0061	0.0046	-0.036	0.63
sum_collect	-0.32	1	0.036	0.14	0.025	-0.069	-0.28
sum_redeem	0.0091	0.036	1	0.28	0.71	0.16	0.0083
sum_collect_points	0.0061	0.14	0.28	1	0.31	0.15	0.082
sum_redeem_points	0.0046	0.025	0.71	0.31	1	0.12	0.0014
years_in_the_program	-0.036	-0.069	0.16	0.15	0.12	1	-0.0041
months_since_last_transaction	0.63	-0.28	0.0083	0.082	0.0014	-0.0041	1

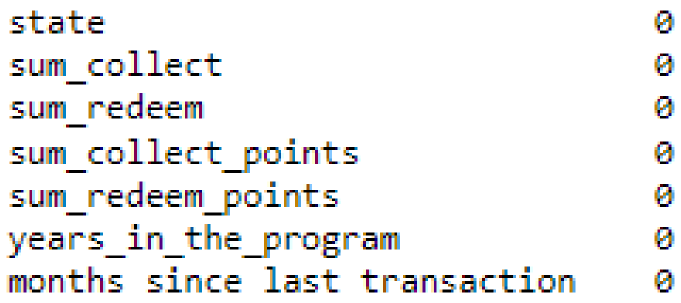
Data Exploration - Bivariate Analysis

The key insights from this correlation matrix and this analysis in general are:

- **State and Months_since_last_transaction**: A strong positive correlation suggests that *the more months that pass since the last transaction, the more likely it is that the customer will lapse*.
- **State and Sum_collect**: A moderate negative correlation implies that *customers who collect more points are less likely to lapse.*
- **Sum_redeem and Sum_redeem_points**: A strong positive correlation indicates that as *the number of times a customer redeems increases, the total number of points they redeem also tends to increase*.
- **Sum_redeem and Sum_collect_points**: A moderate positive correlation suggests that *customers who collect more points are likely to redeem more often.*
- **Years_in_the_program shows weak positive correlations with Sum_redeem, Sum_collect_points, and Sum_redeem_points**. This could mean that *the longer a customer is in the program, the more they interact with it, though these relationships are not strong.*
- **Months_since_last_transaction has a moderate negative correlation with Sum_collect**. This suggests that *customers who collect more frequently are likely to have more recent transactions*.

Data Exploration - Missing Value Treatment

For the latest part of the Data Exploration section, we aim to identify the number of missing values in each feature. Depending on the results, we can determine how to handle the missing values.



```
state                            0
sum_collect                      0
sum_redeem                       0
sum_collect_points               0
sum_redeem_points                0
years_in_the_program             0
months since last transaction    0
```

As we can see there are **no missing values** in our dataset and for these reason, we do not need to take any further actions.

These was our latest step in the Data Exploration analysis and next we will move forward to Data Cleaning and Preprocessing and Model Building steps.

Data Cleaning & Preprocessing

- ✓ Missing Values
- ✓ Inconsistent Data
- ✓ Feature Engineering & Insights
- ✓ Categorical Data Handling

Data Cleaning & Preprocessing

In this section, the data is cleaned and pre-processed to prepare it for model training. The steps involved are:

- **Missing Values:** The data was inspected for any missing values. **There were no missing values found in the dataset.**

- **Inconsistent Data**: Checked for any inconsistencies in the data such as unexpected data types or impossible values. **No such inconsistencies were found.**

- **Feature Engineering**: Created a new feature - ***the ratio of sum_collect_points to sum_redeem_points as an indicator of a customer's tendency to save points***.

- **Categorical Data Treatment**: There were **no categorical values in the dataset**.

Data Cleaning & Preprocessing

Feature Engineering

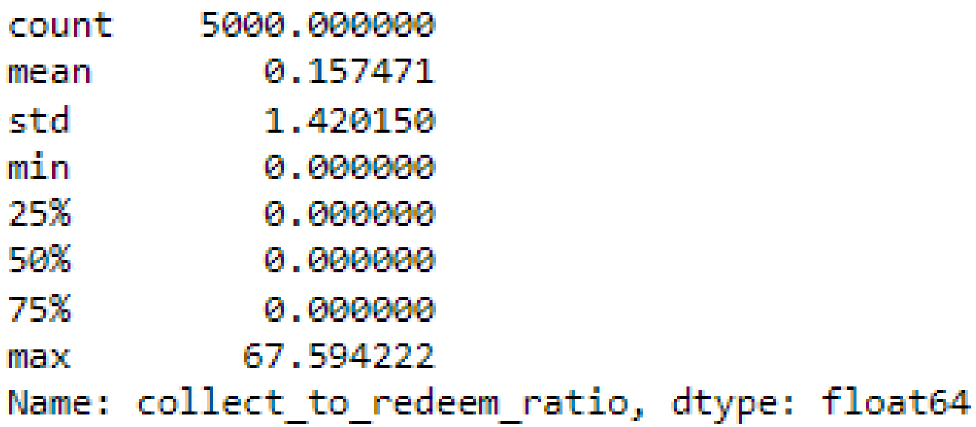
Feature Engineering is a critical step in the machine learning pipeline. It involves creating new features or modifying existing ones to improve the model's performance. In this analysis, a new feature was created to enhance the predictive power of the models:

Points_Ratio (collect_to_redeem_ratio): This feature is the *ratio of sum_collect_points (total points a customer has collected over their tenure) to sum_redeem_points (total points a customer has redeemed)*. It serves as an indicator of a customer's tendency to save or accumulate points. A higher ratio suggests a greater tendency to collect and save points, while a lower ratio may indicate a higher propensity to redeem points.

By introducing this new feature, the model could capture more nuanced behaviours that are not immediately apparent from the original features. This could lead to more accurate predictions of customer churn.

Data Cleaning & Preprocessing
Feature Engineering

The analysis statistical summary of the **collect_to_redeem_ratio** leads us to the following results.



```
count    5000.000000
mean        0.157471
std         1.420150
min         0.000000
25%         0.000000
50%         0.000000
75%         0.000000
max        67.594222
Name: collect_to_redeem_ratio, dtype: float64
```

Data Cleaning & Preprocessing
Feature Engineering

Based on the results of the **collect_to_redeem_ratio** we can note the following key findings:

On average, the ratio of collected points to redeemed points is **0.157**. This suggests that, on average, customers collect more points than they redeem. The standard deviation is quite high (**1.42**) relative to the mean, which indicates a wide spread of the ratio values across customers. The minimum ratio is **0**, which means there are some customers who have not collected any points, or those who have redeemed all of their collected points.

Regarding the quartiles we can notice the following. The first quartile (the 25th percentile) is also 0, meaning at least 25% of customers have a ratio of 0. The median is also 0, which means that at least 50% of customers have a ratio of 0. The third quartile (the 75th percentile) is also 0, indicating that at least 75% of customers have a ratio of 0. Finally, the maximum ratio is **67.59**, indicating there are some customers who have collected many more points than they have redeemed.

From this, we can infer that **a large proportion of customers either redeem points as soon as they collect them or haven't redeemed any points at all**, given the high percentage of zeros. However, there is a **small number of customers who have a very high collection to redemption ratio, suggesting they prefer to accumulate points rather than redeeming them**. These customers might be contributing to the high standard deviation.

Model Building

✓ Discuss & present the created models.

✓ Training procedure presentation.

Model Building

Model building is a key stage in the machine learning pipeline where we use the preprocessed and transformed data to train predictive models. The aim is to create a model that can learn from the patterns in the training data and accurately predict the outcome variable in unseen or new data. In this analysis, we'll be using three types of models:

- **Logistic Regression**: This is a statistical model used for binary classification problems. It estimates the probability that a certain event will occur based on one or more independent variables. It's simple to implement and provides a good baseline model.

- **Decision Tree**: This is a non-parametric supervised learning method used for classification and regression. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features.

- **Random Forest**: This is an ensemble learning method that operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes of the individual trees.

We will train each of these models using the training data, then use the test data to evaluate their performance.

Let's move to the next slide to start with the Logistic Regression model.

Model Building - Logistic Regression

In the first model, a **Logistic Regression** model was chosen **due to its simplicity and interpretability**. Here's a step-by-step training procedure:

Preparing the Data: The dataset was divided into training and testing sets. The training set, which constitutes 80% of the original data, is used to train the model. The remaining 20% is set aside as a test set to evaluate the model's performance.

Training the Model: A Logistic Regression model was initialized and then trained using the training data. The fit function is used to train the model, which involves learning the relationship between the independent variables (features) and the dependent variable (customer's state).

Making Predictions: Once the model was trained, it was used to make predictions on the test data. The predict function was used to predict the labels (active or lapsed) for the test data.

This concludes the Logistic Regression model building process. Let's move on to evaluating the performance of this model.

Model Evaluation & Optimization

✓ Evaluation metrics definition.

✓ Presentation of initial model performance.

✓ Demonstration of the model optimization process.

Introduction to Model Evaluation

Model evaluation is a critical step in the machine learning pipeline. It helps us understand how well our model is performing and whether it's ready to be used for prediction. Here, we define the metrics and techniques used for model evaluation in our analysis:

Accuracy: This is the ratio of the number of correct predictions to the total number of predictions. While it's a straightforward metric, it may not be useful if our classes are imbalanced.

Classification Report: This provides a more detailed performance analysis. It includes metrics such as **precision** (the ability of the classifier to avoid labeling a negative sample as positive), **recall** (the ability of the classifier to find all positive instances), and **F1-score** (the harmonic mean of precision and recall).

Confusion Matrix: This provides a tabular overview of the model's performance. It shows the true positives, true negatives, false positives, and false negatives. This can help us understand where the model is making mistakes.

ROC Curve and AUC: The **Receiver Operating Characteristic (ROC) curve plots** the true positive rate against the false positive rate for different thresholds. The **Area Under the Curve (AUC)** summarizes the ROC curve into a single value, which can serve as a performance measure for the model.

Introduction to Model Evaluation

Complimentary with the evaluation metrics presented on the previous slide we took into consideration also the followings:

Misclassification Rate: This calculates the proportion of instances that the model classified incorrectly. It can be useful when the costs of different types of errors vary significantly.

Feature Importance: For some models, we can calculate the importance of each feature in contributing to the predictions. This can help us understand which features are most relevant to our target variable (will be a cornerstone for our findings in the end).

During the model building and evaluation process, **we set a specific random state or 'seed'** for our models to *ensure the reproducibility of our results*. This means that anyone who runs our code will get the same results, which is important for verifying our findings.

In the following slides, we will apply these evaluation techniques to assess the performance of our Logistic Regression model initially and then for the other three models. Ultimately, these metrics will lead us to select the more appropriate model for our case.

Model Evaluation - Logistic Regression

The performance of the Logistic Regression model was evaluated using the previous discussed metrics:

Accuracy: The test accuracy of the Logistic Regression model was **82.6%**. This suggests that the model correctly *predicted whether a customer will lapse or not in approximately 83 out of 100 cases in the test set.*

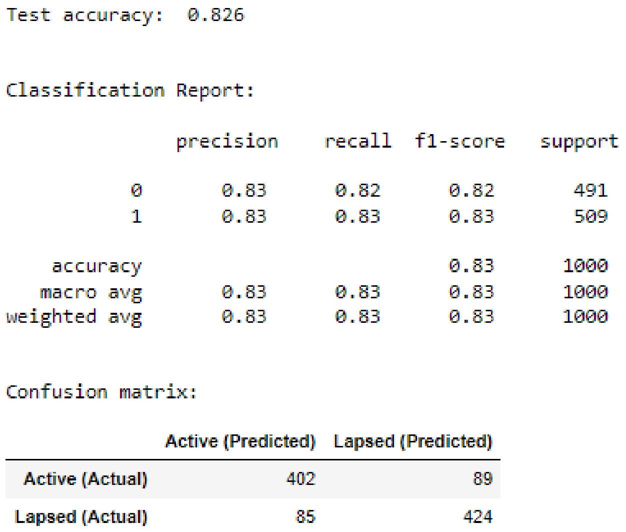
Classification Report: Precision **(83%),** Recall **(83%),** and F1-score**(83%)** were calculated for both 'Active' and 'Lapsed' classes.

Confusion Matrix: The confusion matrix provides a breakdown of the predictions by class. It gives a more granular view of the model's performance (see next slide).

ROC Curve and AUC: The Receiver Operating Characteristic (ROC) curve is a plot that shows the performance of a classification model at all classification thresholds. The Area Under the Curve (AUC) measures the entire two-dimensional area underneath the entire ROC curve and provides an aggregate measure of performance across all possible classification thresholds (**0.91**).

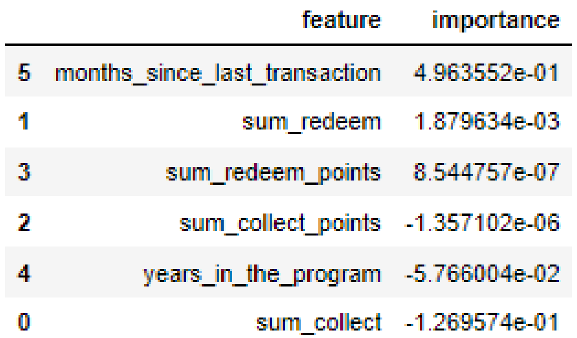
Misclassification Rate: The Logistic Regression model had a misclassification rate of **17.4%**. This means that in **approximately 17 out of 100 predictions**, the model classified the customer status **incorrectly**.

Model Evaluation - Logistic Regression

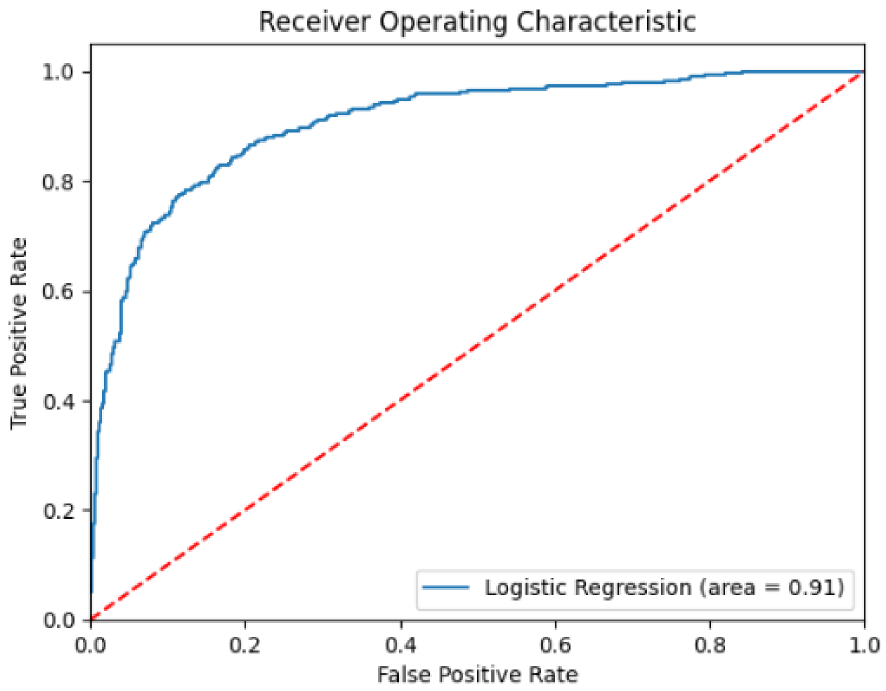


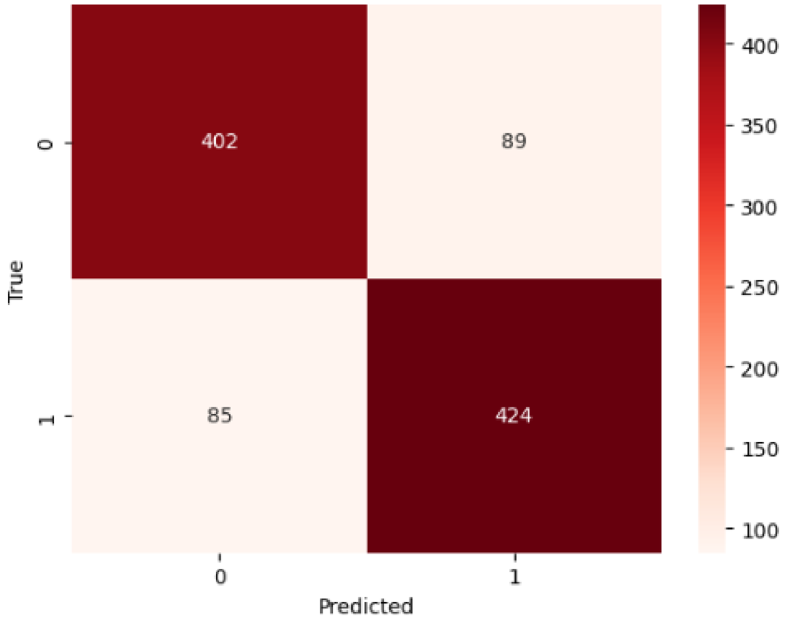
```
Test accuracy:  0.826

Classification Report:

              precision    recall  f1-score   support

           0       0.83      0.82      0.82       491
           1       0.83      0.83      0.83       509

    accuracy                           0.83      1000
   macro avg       0.83      0.83      0.83      1000
weighted avg       0.83      0.83      0.83      1000

Confusion matrix:
```

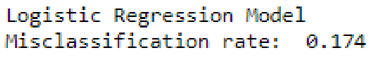
	Active (Predicted)	Lapsed (Predicted)
Active (Actual)	402	89
Lapsed (Actual)	85	424



	feature	importance
5	months_since_last_transaction	4.963552e-01
1	sum_redeem	1.879634e-03
3	sum_redeem_points	8.544757e-07
2	sum_collect_points	-1.357102e-06
4	years_in_the_program	-5.766004e-02
0	sum_collect	-1.269574e-01







```
Logistic Regression Model
Misclassification rate:  0.174
```

Model Building - Decision Tree

The next model we employed for this task is the **Decision Tree model**. The Decision Tree algorithm belongs to the family of supervised learning algorithms which can be used for both classification and regression tasks. Here's a step-by-step overview of the process:

Training the Model

A Decision Tree model was initialized and then trained using the training data. The fit function is used to train the model, which involves learning simple decision rules inferred from the training data.

Making Predictions

After the model was trained, it was used to make predictions on the test data. The predict function was used to predict the labels (active or lapsed) for the test data.

In the next slide, we will delve into how the performance of this Decision Tree model was evaluated.

Model Evaluation - Decision Tree

The performance of the Decision Tree model was evaluated using the previously mentioned metrics:

Accuracy: The test accuracy of the Decision Tree model was **78.6%**. This means that the model correctly predicted whether a customer will lapse or not in approximately **79 out of 100 cases in the test set**.

Classification Report: Precision**(79%)**, Recall**(79%)**, and F1-score**(79%)** were calculated for both classes.

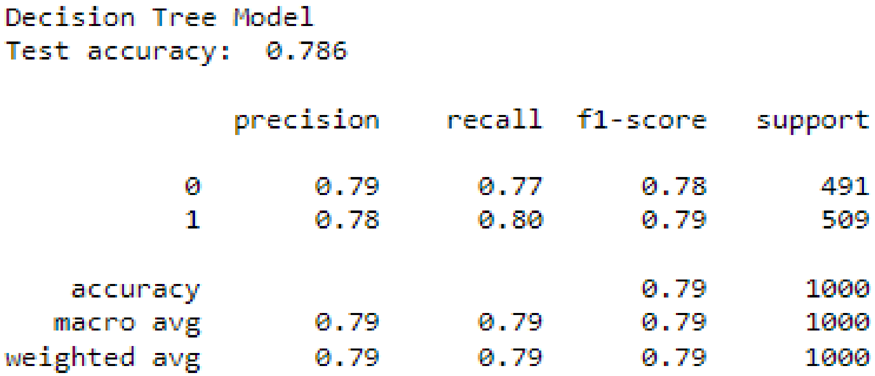
Confusion Matrix: The confusion matrix provides a breakdown of the predictions by class. It gives a more granular view of the model's performance (see next slide).

ROC Curve and AUC: The Receiver Operating Characteristic (ROC) curve was plotted, and the Area Under the Curve (AUC) was calculated (**0.78**).

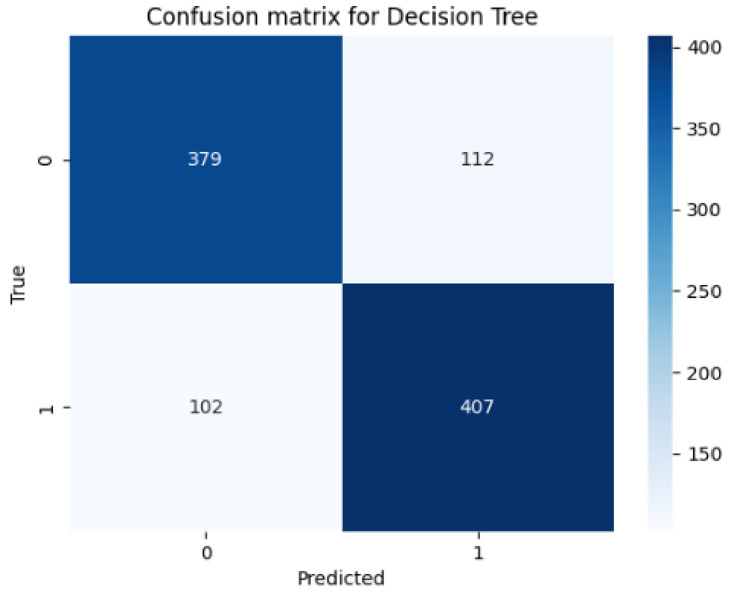
Misclassification Rate: The Decision Tree model had a misclassification rate of **21.4%**. This means that in approximately **21 out of 100 predictions, the model classified the customer status incorrectly**.

In summary, while the *Logistic Regression model showed slightly better performance in terms of accuracy and misclassification rate, the Decision Tree model may capture more complex relationships in the data*. However, given its propensity for overfitting, it's important to also consider more complex and robust models like the **Random Forest**, which we will discuss in the following slides.

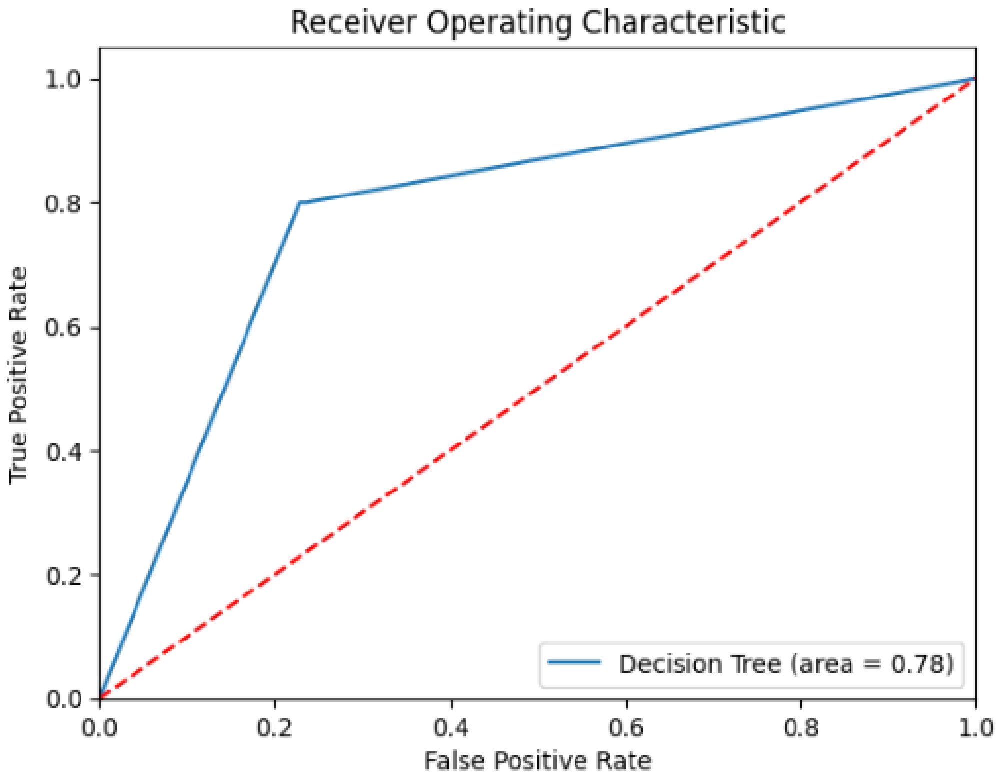
Model Evaluation - Decision Tree



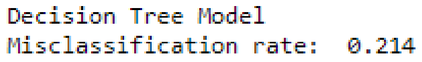
```
Decision Tree Model
Test accuracy:  0.786

              precision    recall  f1-score   support

           0       0.79      0.77      0.78       491
           1       0.78      0.80      0.79       509

    accuracy                           0.79      1000
   macro avg       0.79      0.79      0.79      1000
weighted avg       0.79      0.79      0.79      1000
```



Confusion matrix for Decision Tree



Receiver Operating Characteristic



```
Decision Tree Model
Misclassification rate:  0.214
```

Model Building - Random Forest

The **third model used for this task is the Random Forest model**. The Random Forest algorithm is a type of ensemble learning method, where a group of weak models combine to form a powerful model. Here's a step-by-step overview of the process:

Preparing the Data

The data preparation process is similar to the previous models. The data is divided into a training set and a test set.

Training the Model

A Random Forest model was initialized and then trained using the training data. The fit function is used to train the model, which involves learning the relationship between the independent variables (features) and the dependent variable (customer's state).

Making Predictions

After the model was trained, it was used to make predictions on the test data. The predict function was used to predict the labels (active or lapsed) for the test data.

In the following slides, we will delve into how the performance of this Random Forest model was evaluated.

Model Evaluation - Random Forest

The performance of the **Random Forest model** was evaluated using the evaluation metrics:

Accuracy: The test accuracy of the Random Forest model was **86%.** This suggests that the model correctly predicted whether a customer will lapse or not in approximately **86 out of 100 cases in the test set**.

Classification Report: Precision(**86%**), Recall(**86%**), and F1-score(**86%**) were calculated for both 'Active' and 'Lapsed' classes.

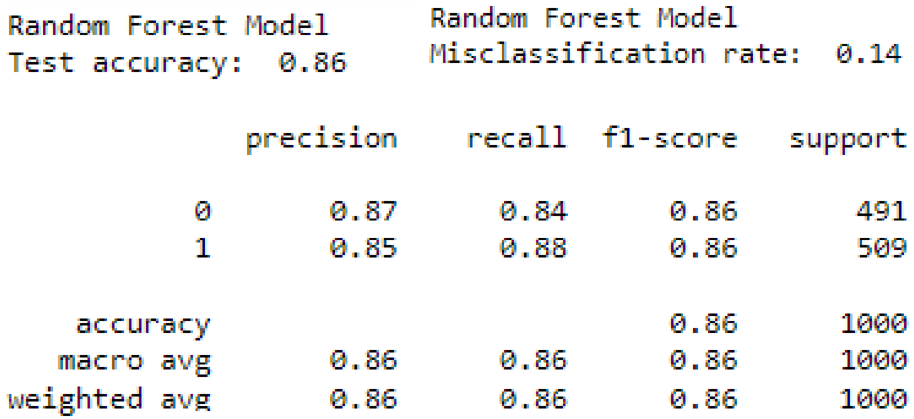
Confusion Matrix: The confusion matrix provides a breakdown of the predictions by class. It gives a more granular view of the model's performance.

ROC Curve and AUC: The Receiver Operating Characteristic (ROC) curve was plotted, and the Area Under the Curve (AUC) was calculated (**0.92**).

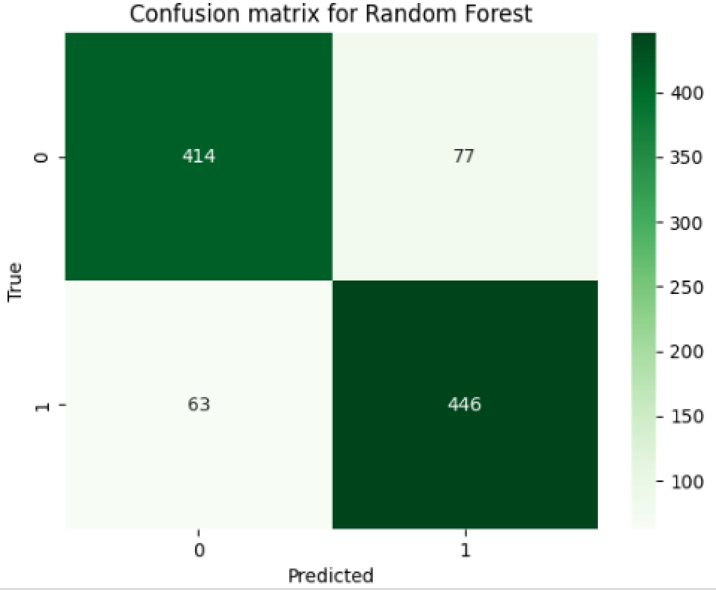
Misclassification b: The Random Forest model had a misclassification rate of **14%.** This means that in approximately **14 out of 100 predictions, the model classified the customer status incorrectly**.

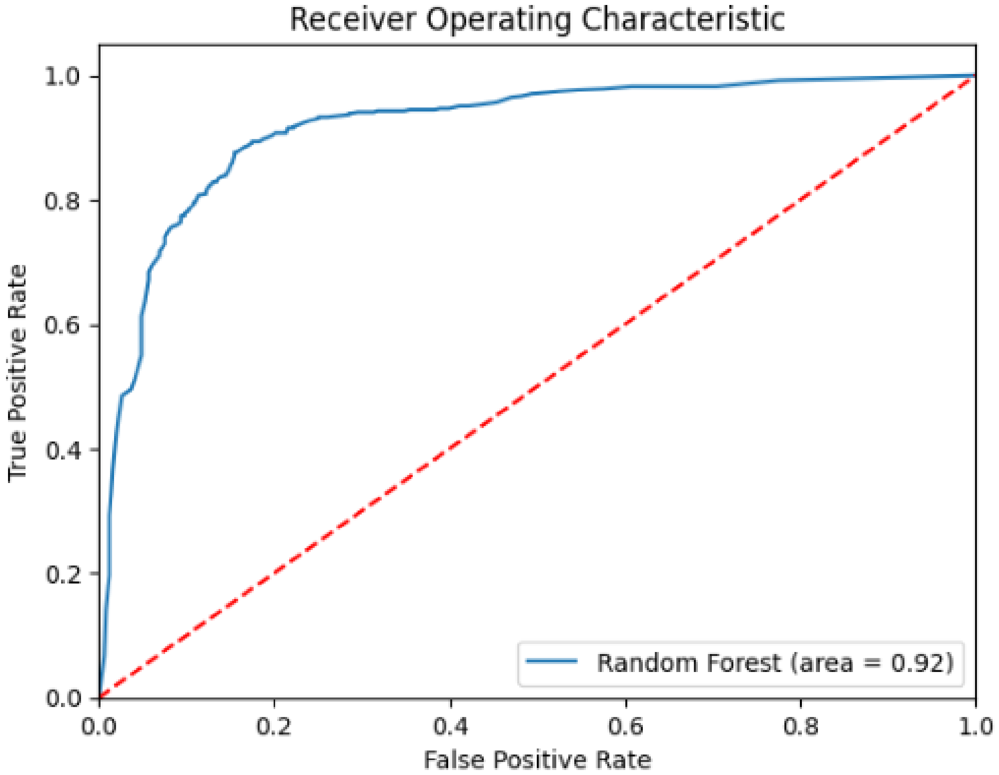
Based on these metrics, the Random Forest model showed better performance in terms of accuracy and misclassification rate compared to the Logistic Regression and Decision Tree models.

Model Evaluation - Random Forest



```
Random Forest Model                Random Forest Model
Test accuracy:  0.86               Misclassification rate:  0.14

              precision    recall  f1-score   support

           0       0.87      0.84      0.86       491
           1       0.85      0.88      0.86       509

    accuracy                           0.86      1000
   macro avg       0.86      0.86      0.86      1000
weighted avg       0.86      0.86      0.86      1000
```





Model Comparison - Logistic Regression, Decision Tree & Random Forest

Comparing the three models, we observe that:

The **Random Forest model achieved the highest accuracy of 86%,** followed by the Logistic Regression model at 82.6%, and the Decision Tree model at 78.6%.

The **Random Forest model also had the lowest misclassification rate of 14%,** compared to 17.4% for the Logistic Regression model and 21.4% for the Decision Tree model.

While the Decision Tree and Random Forest models can capture more complex relationships between features, they are more prone to overfitting. Logistic Regression, on the other hand, is simpler and easier to interpret but might not capture all the complex relationships.

In the next step, we will attempt to further optimize the Logistic Regression model through hyperparameter tuning.

The final decision on the best model to use will be made after this step, taking into consideration both the performance metrics and the business context.

Model Building - Optimized Logistic Regression

An optimized **Logistic Regression** model was built using a pipeline that includes StandardScaler and LogisticRegression. The model was trained with the training data. A cross-validation was conducted to validate the model's performance. GridSearchCV was then used to find the best parameter 'C' for the Logistic Regression model. The model was retrained using the best parameter found. The process can be expanded upon as follows:

Pipeline creation: The first step in the process is the creation of a pipeline that includes the **StandardScaler** and the **Logistic Regression model**. More specifically, the StandardScaler **standardizes features** *by removing the mean and scaling to unit variance.*

Model training: The pipeline is then used to train the model using the training data.

Cross-validation: To ensure that the model generalizes well to new data, cross-validation is performed. This involves splitting the training data into several subsets and training the model multiple times, each time using a different subset as a validation set, while the remaining data is used as training data. This gives a more robust estimate of how the model will perform on unseen data.

Hyperparameter tuning: **GridSearchCV** is used to find the optimal parameter for the Logistic Regression model. GridSearchCV is a method used to tune our model to achieve the best performance. It does this by performing a search over specified parameter values for an estimator. The parameter in question here is 'C' which is the inverse of regularization strength in Logistic Regression. Regularization is a method for preventing overfitting by penalizing high-valued coefficients. Inverse regularization strength 'C' must be a positive float, where smaller values specify stronger regularization.

Model Evaluation - Optimized Logistic Regression

The Optimized Logistic Regression model, which was created using a pipeline that includes StandardScaler and LogisticRegression, and GridSearchCV for optimizing the parameter 'C', demonstrated an improved performance. Here are the key metrics:

Accuracy: The Optimized Logistic Regression model achieved an accuracy of **0.831** on the test data, *correctly predicting the customer status for 83.1% of the cases*.

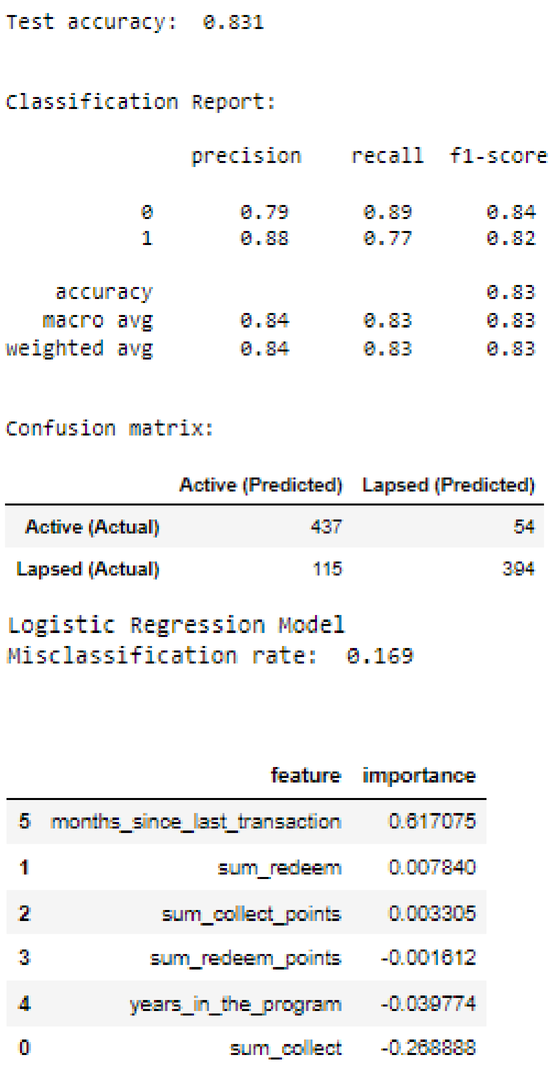
Precision, **Recall**, and **F1-Score**: For the 'Active' class, the model achieved a precision of **0.79**, recall of **0.89**, and F1-score of **0.84**. For the 'Lapsed' class, the precision was **0.88**, recall was **0.77**, and F1-score was **0.82**. The average of these measures across both classes was approximately **0.83**.

Confusion Matrix: The confusion matrix provides a breakdown of the predictions by class. It gives a more granular view of the model's performance.

ROC Curve and AUC: The Receiver Operating Characteristic (ROC) curve was plotted, and the Area Under the Curve (AUC) was calculated (**0.91**).

Misclassification Rate: The misclassification rate of the Optimized Logistic Regression model was **0.169**, indicating that the model incorrectly predicted the customer status for **16.9% of the cases in the test data**.

Model Evaluation - Optimized Logistic Regression

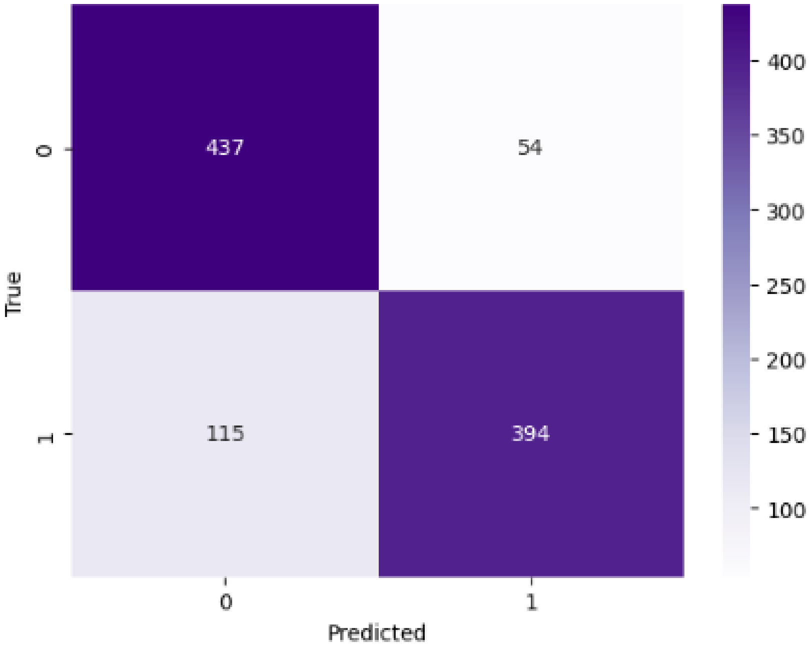


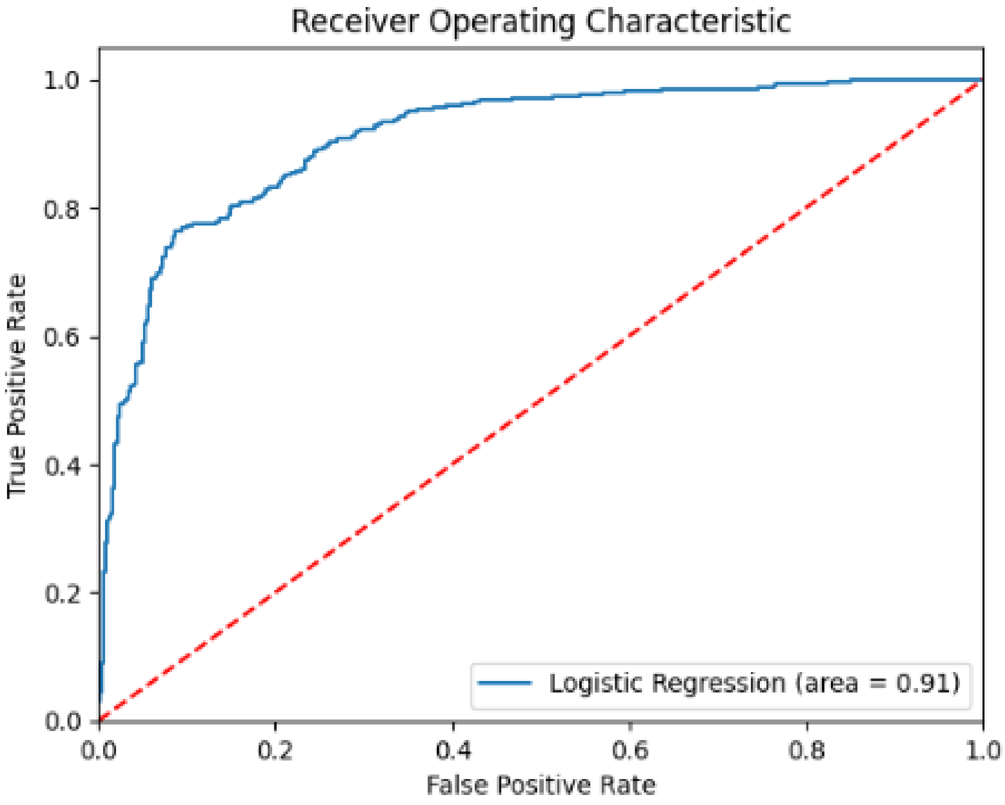
```
Test accuracy:  0.831


Classification Report:

              precision    recall  f1-score   support

           0       0.79      0.89      0.84       491
           1       0.88      0.77      0.82       509

    accuracy                           0.83      1000
   macro avg       0.84      0.83      0.83      1000
weighted avg       0.84      0.83      0.83      1000


Confusion matrix:
```

	Active (Predicted)	Lapsed (Predicted)
Active (Actual)	437	54
Lapsed (Actual)	115	394

```
Logistic Regression Model
Misclassification rate:  0.169
```

	feature	importance
5	months_since_last_transaction	0.617075
1	sum_redeem	0.007840
2	sum_collect_points	0.003305
3	sum_redeem_points	-0.001612
4	years_in_the_program	-0.039774
0	sum_collect	-0.268888





Model Comparison & Selection

✓ Showcase of the comparison of different models according to the evaluation metrics.

✓ Explanation regarding the final model selection.

Model Comparison

The performance of all models was compared using the same evaluation metrics. The optimized Logistic Regression Model performed slightly better compared to the original Logistic Regression Model in terms of accuracy (0.831 vs 0.826) and misclassification rate (0.169 vs 0.174).

Although the **Random Forest Model had the highest accuracy** among all (0.86), the optimized Logistic Regression model was easier to interpret and understand. The **Random Forest Model had a lower misclassification rate** compared to both the Logistic Regression (0.14 vs 0.169) and Decision Tree models (0.14 vs 0.214).

Model Selection

Based on the results and the current analysis task, the **Random Forest Model** was chosen as the best model for this task. The reasons for this choice are:

Accuracy and Misclassification Rate: The *Random Forest Model outperformed* the other models in terms of accuracy and misclassification rate, which means the model can correctly predict whether a customer will lapse in the next 12 months more often than the other models.

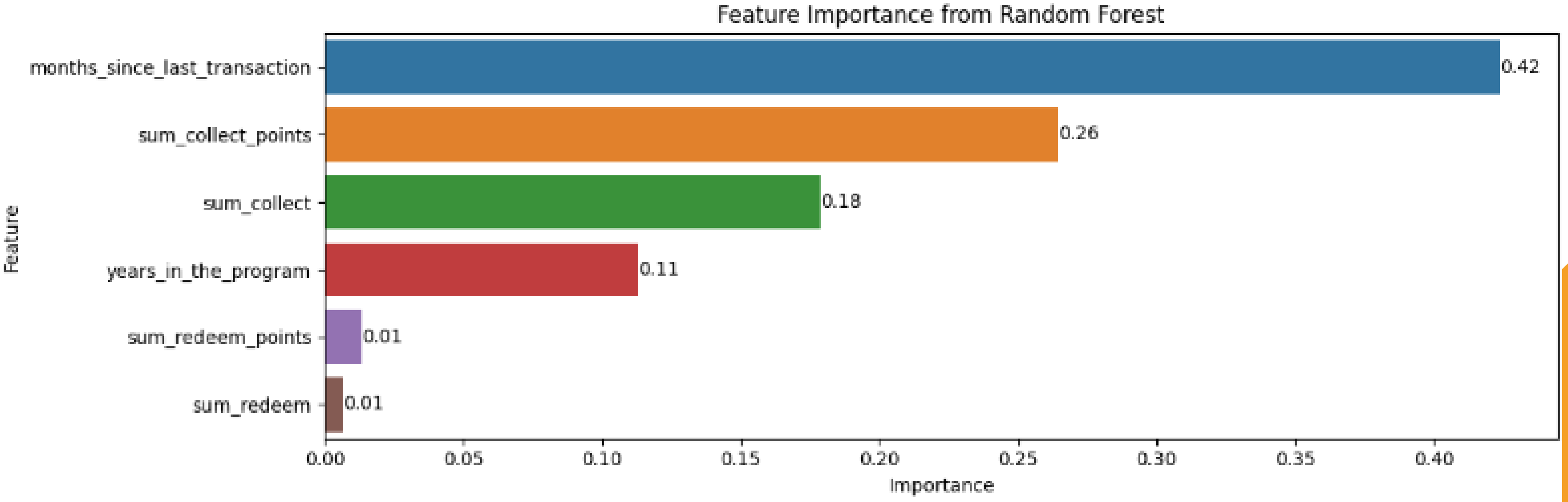
Feature Importance: Random Forest can handle a mix of binary and numerical data and provides a robust estimate of the feature importance. This provides valuable insights about what factors are influential in predicting customer churn and be a critical factor on a more depth future analysis that contains binary data/variables.

Handling Overfitting: Random Forest has an inherent ability to mitigate overfitting, a common problem with Decision Trees. This model performs well on unseen data.

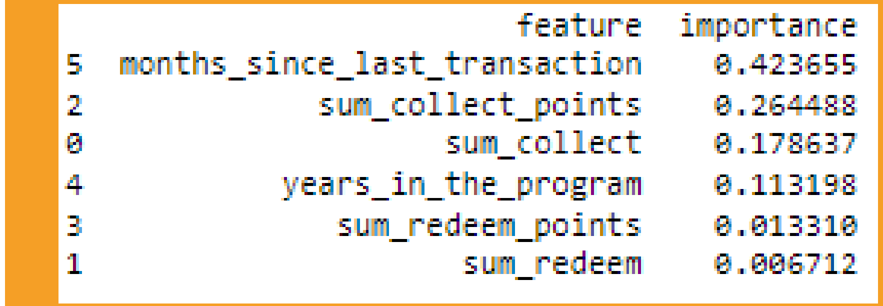
Model Findings

- ✓ Feature importance and their contribution to the selected model.
- ✓ Discussion regarding what those important features imply about customer behavior.

Model Key Findings



Model Key Findings



```
                    feature  importance
5  months_since_last_transaction    0.423655
2           sum_collect_points    0.264488
0                  sum_collect    0.178637
4         years_in_the_program    0.113198
3            sum_redeem_points    0.013310
1                   sum_redeem    0.006712
```

According to the Feature Importance graph we can point out the followings:

1. **Months Since Last Transaction**: With an importance score of **0.423655**, this feature is the **most significant predictor of customer churn**. This suggests that the length of time since a customer's last transaction is the strongest indicator of whether they will churn. This aligns with the definition of a "lapsed" customer, which is one who has not had any collections or redemptions for 12 consecutive months.

2. **Sum of Collected Points**: Coming in at a score of **0.264488**, the total number of points a customer has collected is the **second most important feature**. This suggests that customers who have collected more points are less likely to churn, possibly because they are more engaged with the rewards program.

3. **Sum Collect**: The total amount of money a customer has spent is the **third most important feature**, with a score of **0.178637**. This likely reflects that customers who spend more are more invested in the rewards program and therefore less likely to churn.

4. **Years in the Program**: The number of years a customer has been in the rewards program has an importance score of **0.113198**. This suggests that long-term customers are less likely to churn, likely due to a combination of habit and accumulated benefits.

5. **Sum of Redeemed Points**: With a score of **0.01331**, the total number of points a customer has redeemed is the fifth most important feature. This indicates that customers who actively redeem their points are less likely to churn, presumably because they are getting value from the rewards program.

6. **Sum Redeem**: The total amount of money a customer has spent redeeming points has the lowest importance score of **0.006712**. This suggests that the amount spent on redemptions is not a strong predictor of customer churn, perhaps because it is a less direct measure of engagement with the rewards program than the other features.

Further Suggestions on Reducing Churn Rates

✓ Based on the model's findings, provide suggestions on how to reduce customer churn.

✓ Propose potential marketing strategies and how they relate to the model's findings.

Further Suggestions on Reducing Churn Rates

Based on the feature importance generated by our **Random Forest Model**, we can derive several strategies to reduce customer churn:

✓ **Encourage Regular Activity**: The most significant feature in our model is **'*months_since_last_transaction*'**, which suggests that the more time a customer spends without making a transaction, the more likely they are to churn. To mitigate this, strategies to encourage regular transactions could be implemented. For example, consider:

- Sending personalized reminders
- Offering bonus points for transactions after a period of inactivity
- Running regular promotions to incentivize engagement.

✓ **Boost Point Collection**: The **'*sum_collect_points*'** is a significant feature in determining churn. Offering more opportunities for customers to collect points could, therefore, help reduce churn. For instance, consider:

- Collaborating with new partners
- Increasing point awards for certain transactions
- Hosting special events where extra points can be earned.

Further Suggestions on Reducing Churn Rates

- ✓ **Engage Long-term Customers**: The ***'years_in_the_program'*** feature indicates that customers who have been with the program longer are less likely to churn. Recognizing these customers with special benefits could further reduce their likelihood of churning. For instance, consider:
    - ➢ Exclusive offers
    - ➢ Early access to sales
    - ➢ Personalized rewards based on customer preferences.
- ✓ **Promote Point Redemption**: The ***'sum_redeem_points'*** feature suggests that customers who redeem their points are less likely to churn, likely because point redemption gives customers a sense of getting tangible value from the program. To promote point redemption, consider strategies like:
    - ➢ Simplifying redemption processes
    - ➢ Offering more redemption options
    - ➢ Running promotions where certain redemptions cost fewer points.
- ✓ Increase Spending: The ***'sum_collect'*** feature indicates that customers who spend more are less likely to churn. Strategies that encourage higher spending could be beneficial. For instance, consider implementing a tiered rewards system where customers earn more points for higher levels of spending.

Conclusion

✓ Summarize the key points from the presentation.

Conclusion - Modelling Logic

We built multiple machine learning models to predict customer churn. Our objective was to identify the characteristics that increase the likelihood of a customer lapsing. Our approach to predicting customer churn was a systematic process involving multiple stages. Here's a more detailed outline of our modelling logic:

1. Data Cleaning & Preprocessing: The first step involved cleaning and preprocessing our dataset of customer transactions. This included handling missing data, outliers, and incorrect entries. We also performed data transformations where necessary, such as converting categorical variables into a format suitable for machine learning algorithms.

2. Feature Engineering: Next, we engineered features from the existing data to extract more information that could be useful for our models. For example, we calculated the number of months since the last transaction for each customer, which could potentially indicate whether a customer is likely to lapse.

3. Model Building: We then selected three different machine learning models for this task, namely Logistic Regression, Decision Tree, and Random Forest. These models were chosen for their proven performance in classification tasks and their interpretability. Logistic Regression is a simple yet powerful model that can provide clear coefficients for each feature, making it easy to interpret. Decision Trees and Random Forests can capture complex patterns in the data and can also provide feature importance, which can give us insight into which features are most influential in predicting customer churn.

Conclusion - Modelling Logic

4. Model Training and Evaluation: Each model was trained using the training data and evaluated using both the training and test data. We used various metrics, including accuracy, precision, recall, F1-score, and AUC-ROC, to evaluate the performance of each model.

5. Model Optimization: For the Logistic Regression model, we also performed hyperparameter tuning using GridSearchCV. This allowed us to find the optimal parameters for the model that resulted in the best performance.

6. Model Selection and Comparisons: The selection of appropriate models is a crucial step in our churn prediction task. We started with three models: Logistic Regression, Decision Tree, and Random Forest, each having its strengths and weaknesses.

7. Feature Importance Analysis: After training and optimizing the models, we also analyzed the feature importance. This helped us understand which features were most influential in predicting whether a customer would lapse or remain active.

Conclusion - Model Key Findings

After evaluating the models based on their accuracy and misclassification rate, the **Random Forest model proved to be the most accurate**. The key findings from the model include:

- ✓ **Churn is closely related to customer activity**: The more time a customer spends without transacting, the more likely they are to churn. Regular engagement is crucial for customer retention.
- ✓ **Point collection matters**: The quantity of points customers collect is a significant factor in determining churn. Customers who frequently collect points are less likely to churn, suggesting that they see value in our loyalty program.
- ✓ **Loyal customers are less likely to churn**: Customers who have been part of our program for a longer duration are less likely to churn. These long-term customers are likely more engaged and see more value in our program.
- ✓ **Point redemption influences churn**: The quantity of points customers redeem also affects churn. Customers who frequently redeem points are less likely to churn, which makes sense as redeeming points gives customers a sense of receiving tangible benefits from the program.
- ✓ **Spending affects churn**: The more a customer spends, the less likely they are to churn. Higher spending could be a sign of higher engagement or satisfaction with the loyalty program.

Conclusion - Strategies for Reducing Churn

The proposed strategies in order to reduce churn after our analysis are the followings:

- **Customer Engagement Campaigns**: Develop campaigns to encourage regular transactions. This could involve personalized reminders, bonus points for transactions after a period of inactivity, or regular promotions to incentivize engagement.
- **Expand Point Collection Opportunities**: Collaborate with more partners, increase point awards for certain transactions, or host special events to offer more opportunities for point collection.
- **Reward Long-term Customers**: Recognize customers who have been with the program for a longer time with special benefits. This could involve exclusive offers, early access to sales, or personalized rewards based on their preferences.
- **Promote Point Redemption**: Make it easier and more attractive for customers to redeem their points. This could involve simplifying the redemption process, offering more redemption options, or running promotions where certain redemptions cost fewer points.
- **Encourage Higher Spending**: Implement strategies that encourage higher spending, such as a tiered rewards system where customers earn more points for higher levels of spending.

Q&A

Review the Jupyter Notebook Analysis Documentation

For a complete understanding of the analysis and findings, please review the Jupyter notebook "**propensity_to_lapse_model_building.ipynb**". This notebook includes:

1. **Detailed Code**: Every step of the analysis, including data preprocessing, model building, and evaluation, has been meticulously documented in the code cells.
2. **Explanations and Context**: The markdown cells provide context and explain what each section of the code does. They give insights into why certain choices were made during the analysis.
3. **Visualizations**: The notebook includes visualizations such as graphs and heatmaps that provide a visual representation of the data and the findings. They can make it easier to understand complex data patterns.
4. **Proof of Work**: The notebook serves as a proof of the work done in this analysis. It shows exactly how the results were obtained, ensuring transparency and reproducibility.

Therefore, to fully grasp the depth of this analysis and to validate the findings presented here, it's crucial to go through the notebook. The findings and recommendations in this presentation are based on the analysis done in the notebook. Reviewing the notebook will provide a deeper understanding of the data, the modeling approach, and the resulting insights.

You can find the analysis notebook here along with the needed datasets for the current analysis please have a look!

Thank you

Thank you

Thank you

Thank you

Thank you

Thank you