**Final Project Report**

**Practical Machine Learning**

**MS\_DSP 422**

# **Northwestern University**

**GROUP 3**

**Krishna Karthik Reddy Chavva**

**Rohit Bharadwaj Balarama Somayajula**

**Swapnil Roy**

**Vignesh Sridhar**

**INDEX**

1. EXECUTIVE SUMMARY
2. PROBLEM STATEMENT / RESEARCH OBJECTIVES
3. EXPLORATORY DATA ANALYSIS
4. DATA PREPARATION & FEATURE ENGINEERING
5. METHODS
6. FINDINGS AND CONCLUSIONS
7. LESSONS LEARNED AND RECOMMENDATIONS
8. REFERENCES
9. **EXECUTIVE SUMMARY:**

**Enhancing Customer Engagement and Predictive Insights in Retail**

This executive summary presents the outcomes of a comprehensive analysis conducted on the Online Retail II dataset, spanning two years of genuine online retail transactions in the United Kingdom. This dataset provides a valuable perspective into the operational dynamics of a non-store, all-occasion gift-ware retailer, covering the period from December 1, 2009, to December 9, 2011. The primary objective of this analysis was to distill actionable insights from the dataset, thus empowering strategic business decisions. Furthermore, it aimed to enhance customer engagement and refine predictive marketing strategies for the non-store online retail enterprise. Through a synergy of in-store shopping behavior, transaction history, external variables, and online interactions, the analysis endeavors to furnish invaluable insights for optimizing business practices. Integral to the dataset are the following salient attributes:

InvoiceNo:An identifier assigned to each transaction, with 'c' signifying a cancellation.

StockCode: A code assigned to each product.

Description: The nomenclature of the product or item.

Quantity: The volume of each product per transaction.

InvoiceDate: The date and timestamp of transaction generation.

UnitPrice: The individual price of each product in British pounds (Â£).

CustomerID: A unique numerical identifier designated to each customer.

Country: The geographic location where the customer is domiciled.

The dataset bears substantive relevance for comprehending customer tendencies, product proclivities, and sales trajectories. While its immediate pertinence is evident for online retail operations, it also extends its utility to conventional enterprises aspiring to heighten their comprehension of customer preferences, thereby optimizing their operational methodologies.

The analytical journey encompassed the following sequential stages:

Data Preprocessing:

A rigorous data cleansing process rectified missing data, anomalies, and incongruities, ensuring data fidelity and establishing a robust foundation for subsequent analyses.

Exploratory Data Analysis (EDA):

EDA techniques were adeptly harnessed to uncover latent patterns, correlations, and distributions within the data. Data visualizations were judiciously employed to elucidate nuanced trends in sales dynamics, favored merchandise, and patron engagement.

Data Enrichment and Feature Engineering:

The data spectrum was enriched through the generation of synthetic data, introducing columns such as Date of Birth (DOB), Age (as per DOB), and Membership Flag. Key metrics, including maximum and minimum purchase dates, were computed. Customer segments were defined based on purchase behavior, and a test quarter was established to encompass the last 90 days of transactions.

Variable Transformations and Customer Segmentation:

Recency, Frequency, and Monetary Value/Revenue were key dimensions analyzed. Recency encapsulated purchase behavior based on the most recent transaction and customer inactivity period. Frequency captured customer purchase behavior, while Monetary Value/Revenue depicted customer spending patterns. These dimensions facilitated customer segmentation, leading to the creation of predictive features like NextPurchaseDayRange.

Methodology and Tools:

We used tools like Anaconda Jupyter Notebook, providing a robust analytical environment. Multiple ML and DL models were evaluated, with Random Forest, XGBoost, Neural Networks, and SVM Classifier offering insights into customer behavior. Model deployment strategies, including automation, and deployment in azure machine learning platform were explored.

Propensity Modeling:

Predictive algorithms were employed to create a customer propensity model, serving as a compass for tailored marketing strategies.

Classification Models:

A suite of classification algorithms, including Random Forest, XGBoost, Neural Networks, and SVM Classifier, were leveraged to forecast customer behavior and preferences.

Findings and Business Implications:

The models displayed promising performance in accurately predicting customer behavior and purchase patterns. The validation of assumptions and subsequent assessment of their potential impact yielded valuable insights, quantifying benefits in terms of cost and time savings. Notably, these findings underscore the practicality of integrating these insights into daily business operations and signal potential expansions into diverse domains. The resultant business implications are profound, emboldening strategic decisions. They encompass the strategic identification of high-value patrons, the finesse in molding marketing paradigms, and the strategic augmentation of customer engagement endeavors. In concert, these steps culminate in elevated business performance metrics and a stronger competitive edge.

Recommendations:

The analysis underscored the significance of fine-tuning model parameters for optimal accuracy. Ensembling methods and integration of third-party datasets were recommended for further enhancing predictive outcomes.

In conclusion, the analysis of the Online Retail II dataset yields actionable insights that can substantially enhance customer engagement and inform marketing strategies. These insights, coupled with the deployment of advanced classification models, hold the potential to revolutionize business operations, driving customer loyalty, and augmenting overall business performance.

**2. PROBLEM STATEMENT / RESEARCH OBJECTIVES:**

Build a machine learning model that predicts whether an online customer of a retail shop will make their next purchase 90 days from the day they made their last purchase. Next layer of model which is a multi-class classification which predicts the month for which a customer is likely to buy in case that customer is expected to buy in the next 90 days. This application shows the top ten similar users to the entered customer id and also the potential recommended products using collaborative filtering.

**3. EXPLORATORY DATA ANALYSIS:**

Data Type Conversion and Column Renaming:

In this step, we converted the data types of certain columns to better match their content. For instance, we converted 'Customer ID' to an object data type and 'InvoiceDate' to a datetime data type. We also renamed columns to improve their clarity, such as changing 'Invoice' to 'InvoiceNo', 'Customer ID' to 'CustomerID', 'Price' to 'UnitPrice', and 'StockCode' to 'ProductCode'.

Adding Additional Variables:

We introduced new columns to the dataset to enhance its information. For example, we calculated the 'total\_spend' by multiplying 'UnitPrice' and 'Quantity'. We also created a 'refund' column to identify transactions with canceled orders and refunds.

Handling Outliers and Anomalies:

This step involved identifying outliers and anomalies in the 'total\_spend' column using box plots and summary statistics. We removed transactions with negative total spend values and capped extremely high values at the 99th percentile.

Identifying and Handling Non-Transactional Entries:

Here, we located and eliminated non-transactional entries based on 'ProductCode' and 'Description', ensuring the dataset only contained valid transactions.

Analyzing Amazon Fee and Postage Fee:

We calculated and examined the total Amazon Fee cost and Postage Fee cost to gain insights into the financial aspects of the transactions.

Removing Chosen Values:

Specific descriptions like 'Sample' and 'Adjust bad debt' were removed from the dataset, as they likely represented entries that weren't relevant to the analysis.

Handling Missing Values:

We assessed the percentage of missing values in each column. For 'CustomerID', we decided to remove rows with missing values since the absence of 'CustomerID' would hinder customer-based analysis.

Data Filtering and Cleaning:

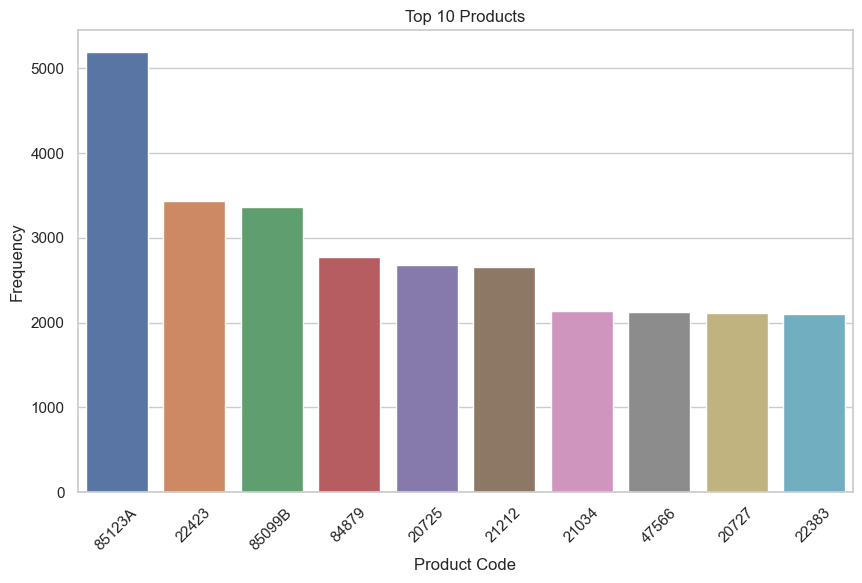
A new DataFrame was created by removing rows with missing values. We then checked the cleaned DataFrame's data types and information to ensure data quality.

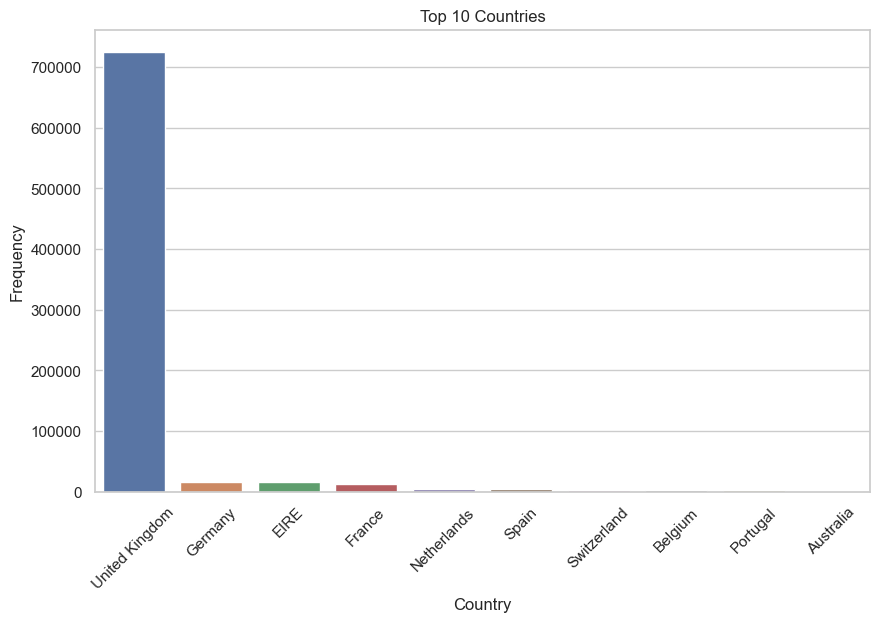
Distribution of Numerical Features:

Utilizing histograms and visualizations, we explored the distributions of 'Quantity', 'UnitPrice', and 'CustomerID'. These plots provided insights into the data's numerical characteristics.

Top Products and Countries:

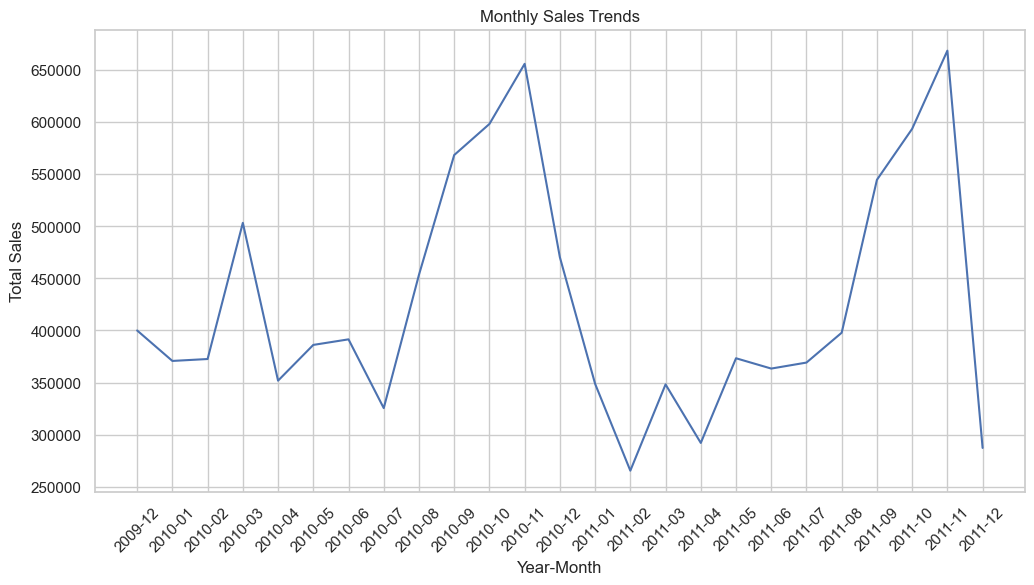
Through bar plots, we identified and presented the most frequent products and countries in the dataset, offering an overview of popular items and customer locations.





Monthly Sales Trends:

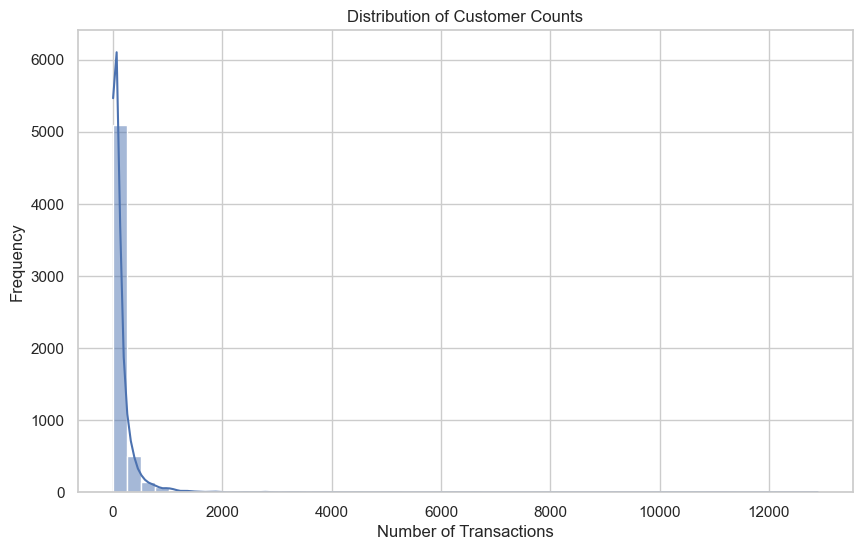
We transformed the data by extracting year and month information from 'InvoiceDate', enabling us to calculate total sales for each month. A line plot visualized the sales trends over time.



Despite observing a potential pattern of increased sales during the months of October and November, it is not possible to definitively establish seasonality with only two years' worth of data. A more extended timeframe of at least five years is necessary to draw conclusive insights regarding seasonal trends.

Data Distribution Visualization:

This step employed histograms and box plots to visually capture the age distribution of customers and compared the distribution among different membership types.



The data appears to be right-skewed.

Number of Online Customers and Countries of Origin:

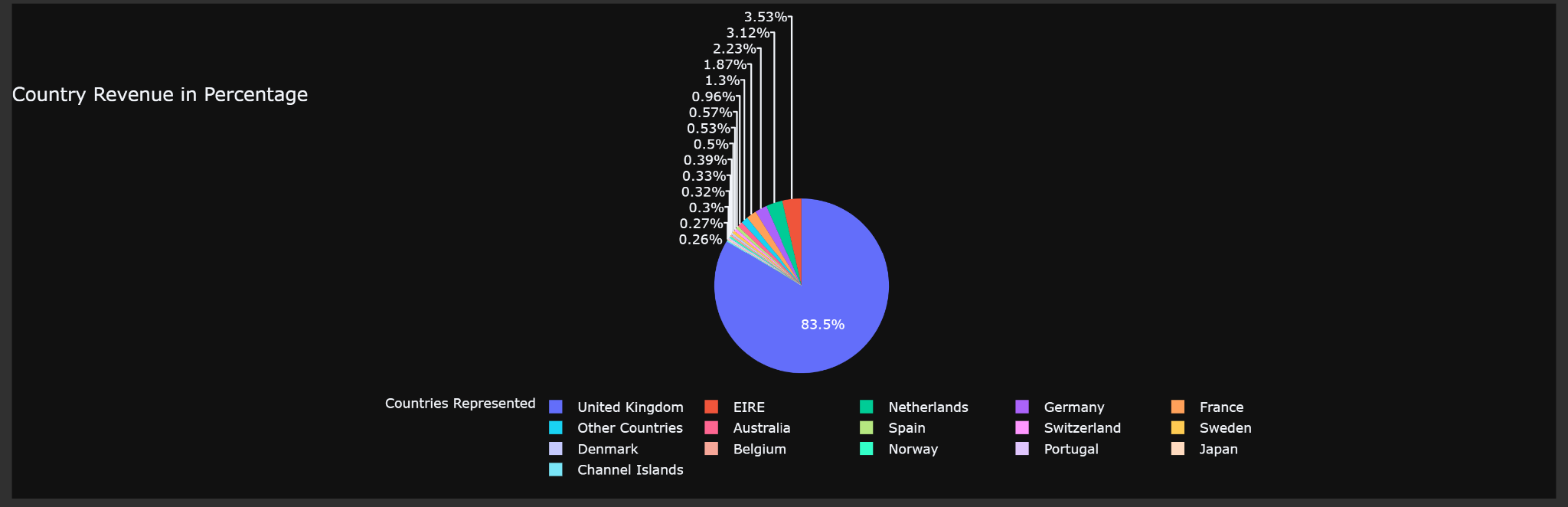
We determined the count of online customers and their distribution across various countries, displaying the information using a bar chart.

Monthly Revenue:

By calculating revenue for each month and creating a line plot, we could observe the revenue variations over the specified time period.

Country Revenue in Percentage:

Revenue by country was computed along with its percentage contribution to total revenue. A pie chart was used to visualize the distribution of revenue across different countries.



Generating Target Labels:

In preparation for predicting customer purchase behavior, we divided the dataset into distinct time segments, extracting relevant customer data for analysis. Opting for a 3-month window offered a balanced perspective, capturing noteworthy behavioral patterns while remaining pragmatic. This choice balances sensitivity to short-term changes and practicality.

To simulate testing conditions without a dedicated test dataset, we partitioned the data into two separate frames based on dates. This facilitated predicting whether customers would purchase in the upcoming quarter, allowing model evaluation in a controlled setting within the given timeframe.

The selection of a 90-day prediction horizon derives from its alignment with business dynamics and behavior shifts. This timeframe, representing a quarter, accommodates short-term variations and longer-term trends, harmonizing with business cycles and enabling comprehensive modeling of customer responses to marketing strategies and external factors.

**4. DATA PREPARATION & FEATURE ENGINEERING:**

4.1. Generating Synthetic Data:

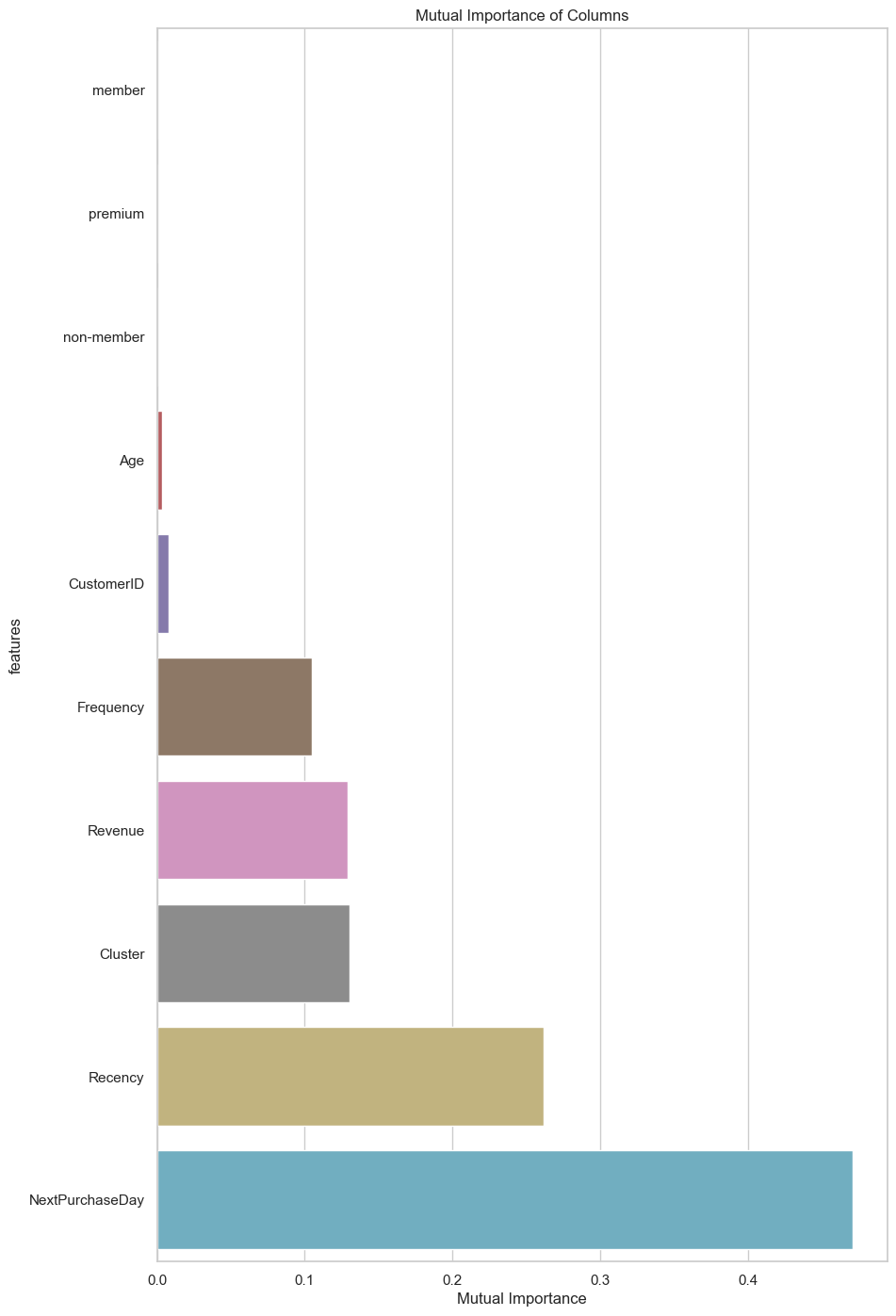
Synthetic data was generated to augment the original dataset. This process included creating customer-related columns, generating synthetic Date of Birth (DOB), calculating age, and assigning membership flags. The synthetic data was then exported and loaded back into the dataset.

This step is crucial as it helps in adding demographic data of customers and diversity, which is particularly valuable for training machine learning models. This data generation is done to check if the additional features helped in improving model performance.

4.2. Feature Importance:

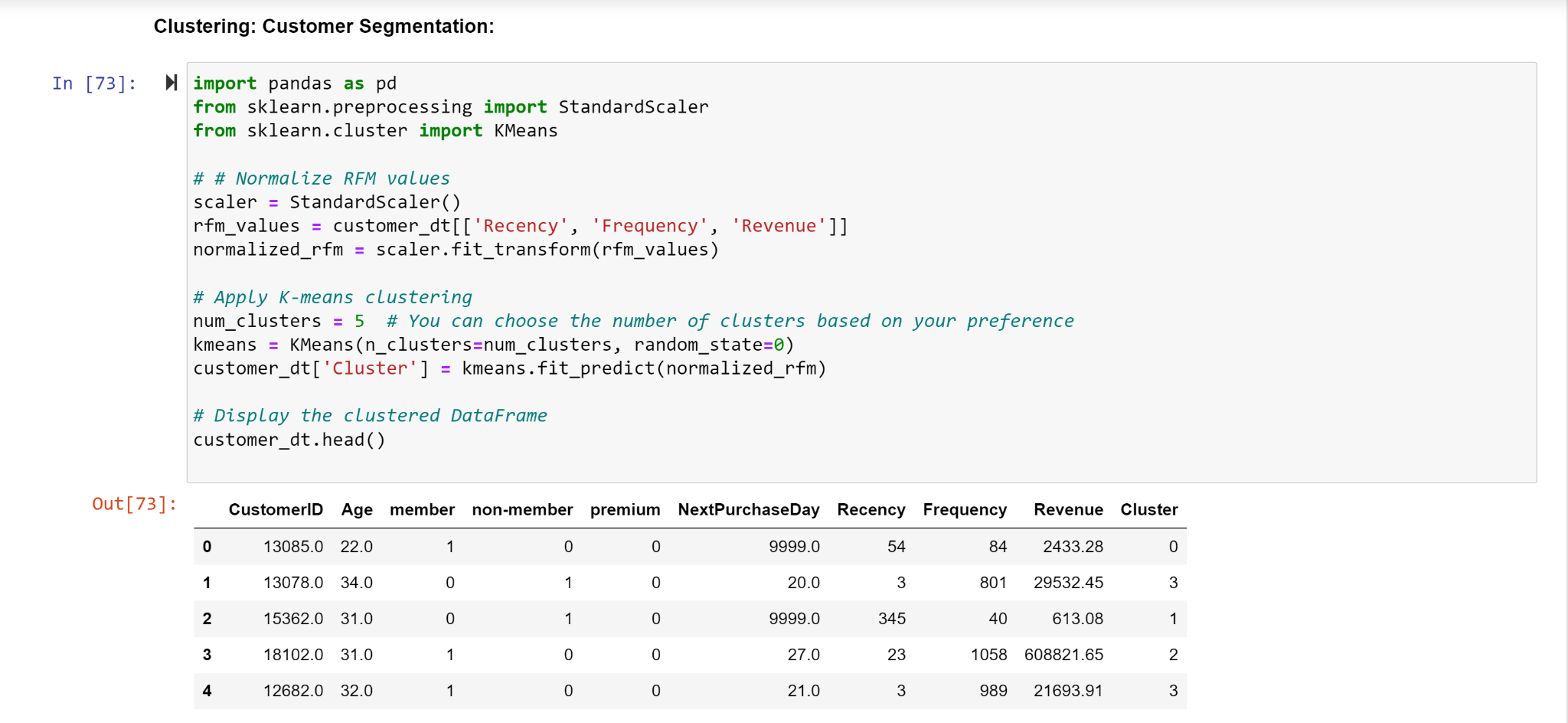
Automated feature importance ranking using the mutual\_info\_classif function was performed to identify important features that contribute to the target variable.





4.3. Variable Transformations and Customer Segmentation:

In the phase of Variable Transformations and Customer Segmentation, we focused on three essential dimensions: Recency, Frequency, and Monetary Value/Revenue. Recency was utilized to encapsulate purchasing behavior through the examination of the most recent transaction and the duration of customer inactivity. Frequency enabled us to grasp customer purchase patterns, while Monetary Value/Revenue portrayed their spending tendencies. By leveraging these dimensions, we conducted customer segmentation, a process that paved the way for the development of predictive attributes such as NextPurchaseDayRange.



The process of RFM (Recency, Frequency, Monetary Value) segmentation involves utilizing K-means clustering to group customers based on their purchasing behavior patterns.The goal of clustering is descriptive, that of classification is predictive (Veyssieres and Plant,1998). To ensure consistent scales for each RFM dimension, the RFM values are normalized using the StandardScaler. This normalization transforms the Recency, Frequency, and Monetary Value attributes into standardized representations for each customer. The subsequent step employs the K-means algorithm on these normalized RFM values. By specifying the desired number of clusters (k), K-means assigns each customer to a cluster based on their RFM attributes. The outcomes of this clustering procedure are stored in a new column labeled 'Cluster' within the customer dataset. This method enables businesses to discern distinct customer segments, permitting tailored marketing strategies aligned with their preferences and behaviors.

K-means Algorithm Formula:

X = {x1, x2, ..., xn} is the set of n data points to be clustered.

k is the desired number of clusters.

In the context of utilizing RFM for K-means clustering:

- The variable X symbolizes the normalized RFM values for each individual customer.

- The variable n signifies the number of customers.

- The variable k represents the chosen number of clusters.

- The algorithm's purpose is to cluster customers based on their RFM values, resulting in groups that share comparable behavior patterns.

Integrating these dimensions through the RFM formula yields a comprehensive profile of each customer's behavior. By applying the K-means clustering algorithm to these RFM values, the segmentation strategy is augmented. The ensuing clustering assigns customers to groups with shared behavior traits, forming the foundation for tailored strategies and predictive attributes like the innovative NextPurchaseDayRange. This fusion of techniques empowers the interpretation of historical behavior and the projection of future actions and preferences, fostering a targeted and effective approach to customer engagement.

**5. METHODS:**

**5.1. Model Selection and Cross-Validation:**

In this step, several machine learning models were selected to be trained and evaluated. These models included techniques such as Logistic Regression, Gaussian Naive Bayes, Random Forest, etc. The aim was to have a diverse set of algorithms to compare and choose the best-performing one.

1. Model Definition:

Each selected model was defined and configured with appropriate hyperparameters. For example, Logistic Regression and Random Forest were instantiated with specific settings that determine their behavior during training and prediction.

2. Cross-Validation:

To ensure robust model evaluation, a k-fold cross-validation approach was employed. The dataset was divided into k subsets (folds), and each fold was used as a validation set while the others were used for training. This process was repeated k times, with each fold serving as the validation set exactly once. This technique helps prevent overfitting and provides a more accurate estimation of the model's performance.

3. Splitting train and test data:

We have created a new DataFrame 'X' by dropping columns from the original dataset that we don't need for prediction. These columns are 'NextPurchaseDayRange', 'NextPurchaseDay', and 'CustomerID'. We have excluded the 'NextPurchaseDayRange' column because it will be our target variable, and 'NextPurchaseDay' and 'CustomerID' because they are identifiers and not predictive features.

We have created a new Series 'y' containing the values of the 'NextPurchaseDayRange' column. This column is our target variable that we want to predict.

We split our data into two sets: a training set and a testing set. This is important to evaluate how well our model generalizes to unseen data. Here's what each variable represents:

* X\_train: This will contain a subset of the data features that our model will use for training.
* X\_test: This will contain another subset of the data features that our model will use for testing and evaluation.
* y\_train: This corresponds to the target variable for the training set, 'NextPurchaseDayRange'.
* y\_test: This corresponds to the target variable for the testing set, 'NextPurchaseDayRange'.

The test\_size parameter is set to 0.2, which means 20% of the data will be reserved for testing, while 80% will be used for training. The random\_state parameter is set to None, meaning the data split will be random each time we run the code. The shuffle parameter is set to True, which shuffles the data before splitting to ensure randomness.

**5.2. Trivial Classifiers and Evaluation:**

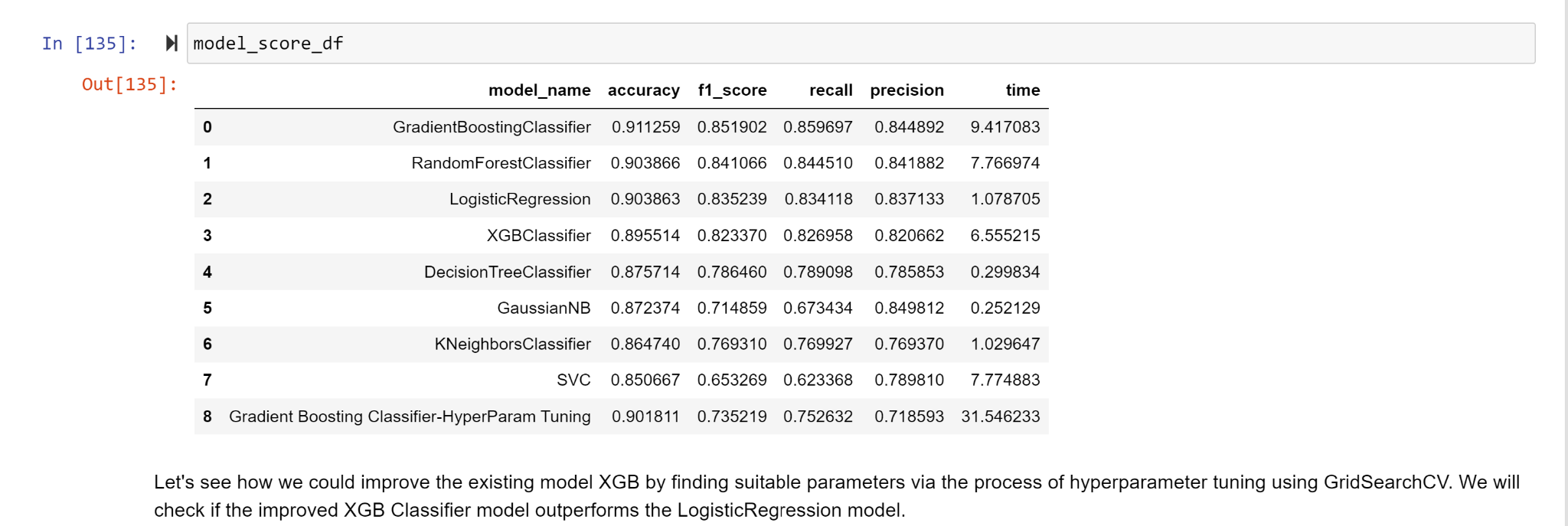
A comprehensive evaluation of various classification models is conducted using cross-validation techniques. An array of different machine learning models, including Logistic Regression, Gaussian Naive Bayes, Random Forest, Support Vector Classifier (SVC), Decision Tree, Gradient Boosting, XGBoost, and K-Nearest Neighbors, is created.

We performed an analysis on two of the basic classifiers, specifically focusing on XGBoost and Logistic Regression. "Logistic regression is among the discrete choice models, commonly employed for predicting and assessing probabilities. In scenarios requiring regression or classification solutions, the establishment of a cost function is necessary. Optimal model parameters are then derived through iterative optimization techniques. Subsequently, the solution model's merits and drawbacks are assessed through verification and testing" (Zhang 2021).

"XGBoost not only uses the first derivative but also the second derivative, so the loss is more accurate and the loss can be customized. This method considers the case where the training data is sparse values, and the default direction of the branch can be specified for missing values or specified values, which can greatly improve the efficiency of the algorithm" (Zhang 2021).

For each model, a dictionary records relevant metrics such as accuracy, F1 score, recall, precision, and execution time. The code employs K-fold cross-validation with five folds to assess each model's performance on the training data. Cross\_val\_score function from sklearn calculates the average scores across different folds for each metric.

The results are then aggregated into a DataFrame, sorted by accuracy, F1 score, and time taken for execution. This tabulated summary provides a comparative overview of the models' performance, helping to identify the most suitable model based on the specified evaluation metrics.



**5.3. Hyperparameter Tuning:**

Hyperparameters are configuration settings that are not learned from the data during model training but significantly impact a model's performance. Tuning hyperparameters is crucial to optimize a model's effectiveness. The successful deployment of the machine algorithms highly depends on the model parameters which are extremely difficult to tune. (Joy et al. 2016)

1. RandomizedSearchCV:

Hyperparameter tuning for the Gradient Boosting Classifier was performed using `RandomizedSearchCV`. This technique explores a randomized set of hyperparameter combinations within predefined ranges. By doing so, it efficiently narrows down the hyperparameter space, helping to find a better set of values that enhance the model's performance.

2. Best Hyperparameters:

The outcome of the hyperparameter tuning was the identification of the best combination of hyperparameters for the Gradient Boosting Classifier. This set of hyperparameters maximized the model's performance on the validation sets.

The hyperparameters used for tuning Gradient boosting classifier are listed below:

* n\_estimators: This parameter specifies the number of boosting stages (trees) to be built in the ensemble.

Values: It's a list of integers, such as [10, 50, 100, 200].

A higher number of estimators can improve the model's performance, but there's a trade-off with computation time.

* learning\_rate: The learning rate controls the contribution of each individual classifier (tree) to the final ensemble. It scales the impact of each tree's prediction.

Values: A list of floating-point values, often in the range [0.1, 0.01, 0.001].

Lower values make the model more robust by reducing the effect of each individual tree. However, smaller learning rates may require more estimators to achieve good performance.

* max\_depth: This parameter determines the maximum depth of each individual tree in the ensemble.

Values: A list of integers, like [3, 5, 7].

Deeper trees can capture complex relationships in the data, but they can also overfit. Shallower trees can lead to a more generalized model.

* max\_features: It controls the number of features considered at each split point during tree building.

Values: A list of strings, typically ['auto', 'sqrt', 'log2'], which determine how many features are used.

Different values influence the diversity of trees. 'auto' uses all features, while 'sqrt' and 'log2' consider a fraction of them.

* subsample: This parameter specifies the fraction of the training data to be used for fitting individual trees. It introduces stochasticity into the boosting process.

Values: A list of floating-point values, often [1.0, 0.8, 0.6].

Values less than 1.0 introduce randomness and can help prevent overfitting by not using the entire training dataset.

3. Test Set Evaluation:

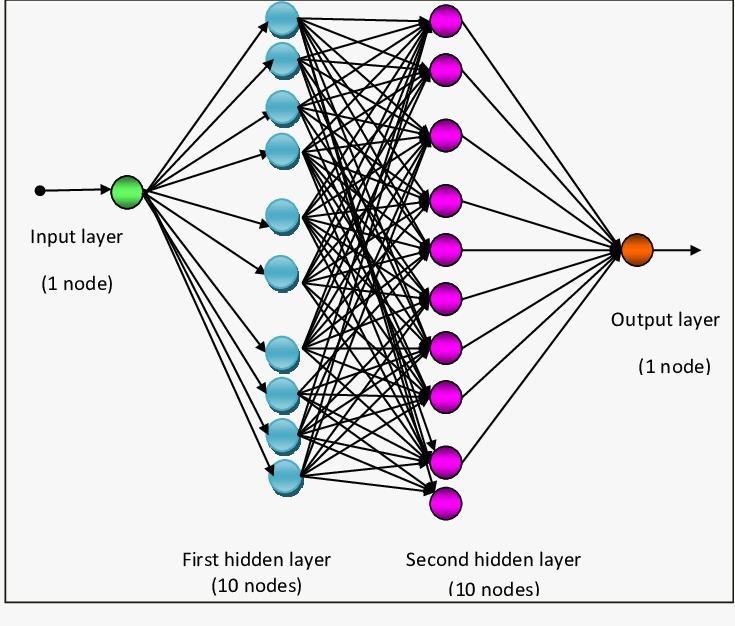
After obtaining the best hyperparameters, the tuned Gradient Boosting Classifier was evaluated on a separate test set that wasn't used during hyperparameter tuning. This allowed an unbiased assessment of the model's performance on unseen data.

**5.3. Neural Network Model:**

In addition to traditional machine learning models, a neural network model was also employed. Neural networks are a type of deep learning model capable of capturing complex patterns and relationships in data.

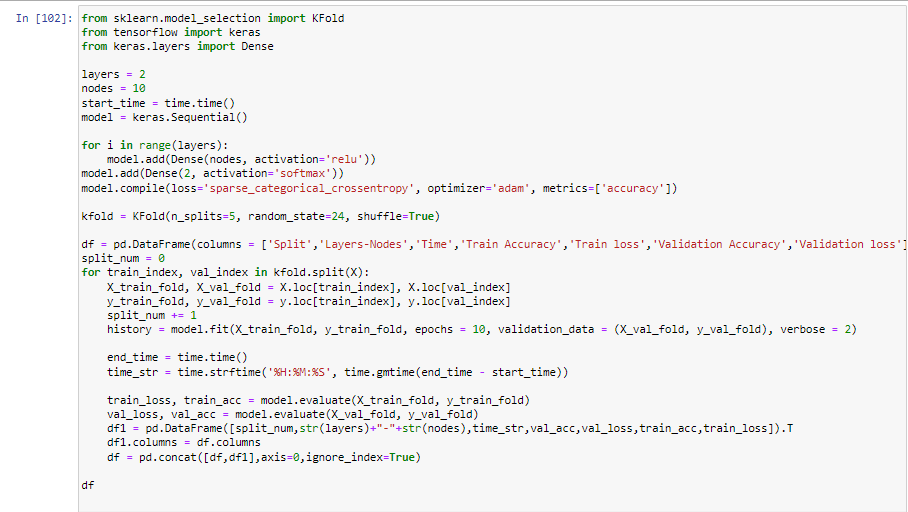
1. Model Architecture:

The neural network model's architecture was defined, specifying the number of layers and nodes in each layer. This architecture determines the model's capacity to learn and represent intricate relationships in the data.



2. K-Fold Cross-Validation:

Similar to other models, the neural network underwent k-fold cross-validation. The dataset was divided into k folds, and the model was trained and evaluated on each fold. This provided a comprehensive understanding of the model's performance across different data subsets.

****

3. Results Compilation:

The performance metrics from k-fold cross-validation were recorded and aggregated. This allowed a comparison of the neural network's performance against other models using the same evaluation criteria.

**5.4. Model Comparison and Evaluation:**

After training and evaluating all the different models, a comparative analysis was conducted to select the most suitable model for the task.

5.4.1. Tools and Packages;

numpy, pandas,scipy - These packages are used to perform basic Data wrangling and preparation.

Matplotlib,seaborn,plotly - These packages are used to visualize the outputs.

Sklearn - It is used for train and test split, modeling and evaluating the models.

Keras,tensorflow - Used to build Neural network models.

Pickle - Used to store the executable model for deployment.

Anaconda - Used for performing the whole exercise.

Mercury - Used to convert jupyter notebook into an App.

5.4.2. Performance Metrics:

Performance metrics such as accuracy, F1-score, precision, and recall were used to assess each model's strengths and weaknesses.

Accuracy measures the proportion of correctly classified instances out of the total instances in the dataset. It provides an overall view of how well the model performs.

* Accuracy- (True Positives + True Negatives) / Total Predictions

Precision is the ratio of correctly predicted positive instances to the total instances predicted as positive. It measures the model's ability to avoid falsely labeling negative instances as positive.

* Precision- True Positives / (True Positives + False Positives)

Recall is the ratio of correctly predicted positive instances to the total actual positive instances. It measures the model's ability to correctly identify all positive instances.

* Recall- True Positives / (True Positives + False Negatives)

The F1-Score is the harmonic mean of Precision and Recall. It balances the trade-off between Precision and Recall, providing a single metric that considers both false positives and false negatives. It's especially useful when classes are imbalanced.

* F1-score- [ 2 \* (Precision \* Recall) / (Precision + Recall)]

5.4.3. Confusion Matrices:

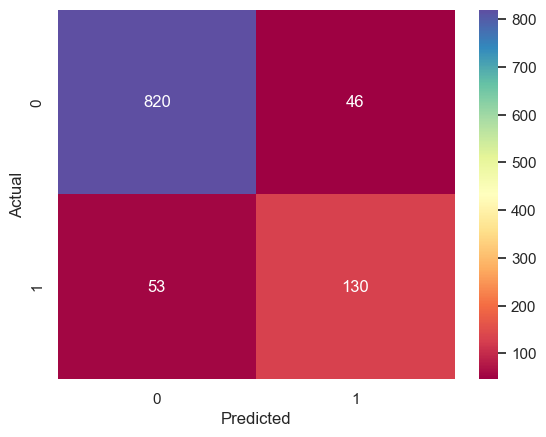
Confusion matrices were employed to visualize the models' classification performance. These matrices highlight true positives, true negatives, false positives, and false negatives, providing a comprehensive understanding of the model's classification decisions.

“A confusion matrix contains information about actual and predicted classifications done by a classification system. Performance of such systems is commonly evaluated using the data in the matrix” (Santra, Christy 2012).

* True Positives (TP): when the actual value is Positive and predicted is also Positive.
* True negatives (TN): when the actual value is Negative and prediction is also Negative.
* False positives (FP): When the actual is negative but prediction is Positive. Also known as the Type 1 error
* False negatives (FN): When the actual is Positive but the prediction is Negative. Also known as the Type 2 error

In our model we have derived a confusion matrix for the XGB classifier model and Logistic Regression model.

XGB classifier model:



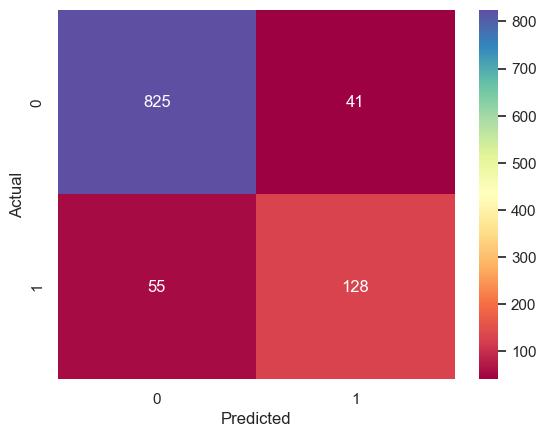
True Positives (TP): The value in the bottom-right corner (130) represents the number of instances where the actual class is positive (1) and the model correctly predicted it as positive (1).

False Positives (FP): The value in the top-right corner (46) indicates the number of instances where the actual class is negative (0), but the model incorrectly predicted it as positive (1).

False Negatives (FN): The value in the bottom-left corner (53) signifies the number of instances where the actual class is positive (1), but the model predicted it as negative (0).

True Negatives (TN): The value in the top-left corner (820) denotes the number of instances where the actual class is negative (0), and the model correctly predicted it as negative (0).

Logistic Regression:



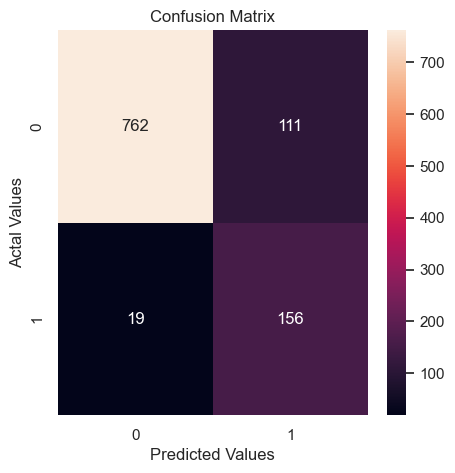
True Positives (TP): The value in the bottom-right corner (128) represents the number of instances where the actual class is positive (1) and the Logistic Regression model correctly predicted it as positive (1).

False Positives (FP): The value in the top-right corner (41) indicates the number of instances where the actual class is negative (0), but the Logistic Regression model incorrectly predicted it as positive (1).

False Negatives (FN): The value in the bottom-left corner (55) signifies the number of instances where the actual class is positive (1), but the Logistic Regression model predicted it as negative (0).

True Negatives (TN): The value in the top-left corner (825) denotes the number of instances where the actual class is negative (0), and the Logistic Regression model correctly predicted it as negative (0).

Neural Networks Model:



True Positives (TP): 156 instances were correctly predicted as positive (1) by the neural network model.

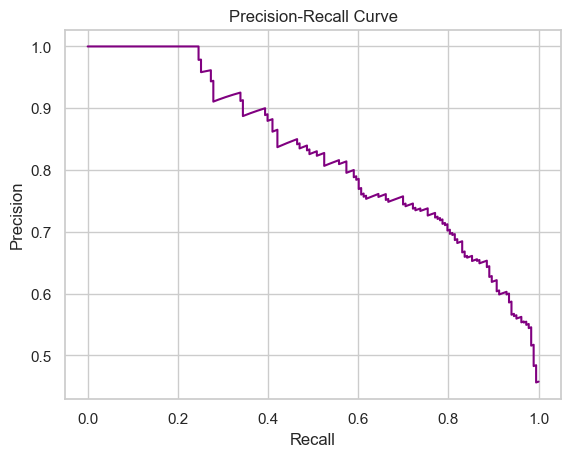
False Positives (FP): 47 instances that were actually negative (0) were incorrectly predicted as positive (1) by the model.

False Negatives (FN): 111 instances that were actually positive (1) were incorrectly predicted as negative (0) by the model.

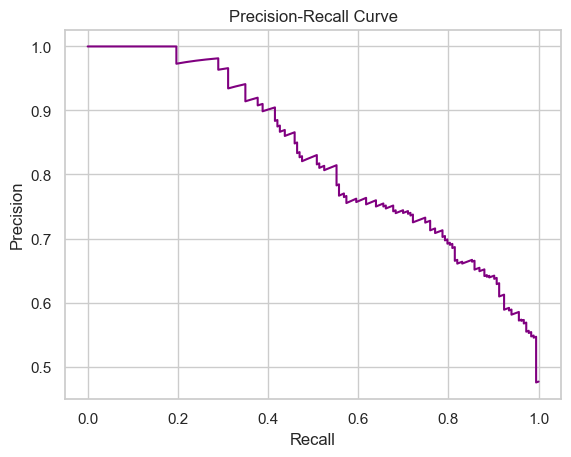
True Negatives (TN): 762 instances were correctly predicted as negative (0) by the neural network model.

5.4.4. Precision-Recall Curves:

Precision-recall curves were plotted to illustrate the trade-off between precision and recall for each model. These curves are particularly useful when dealing with imbalanced datasets.



Logistic Regression



XGBClassifier

As we can see, the improved XGB classifier model is more accurate than the LogisticRegression model.

4. ROC AUC curve:

* ROC:

The ROC curve is a graphical representation of the trade-off between the true positive rate (Sensitivity) and the false positive rate (1 - Specificity) as the discrimination threshold of a binary classifier is varied. The x-axis of the ROC curve represents the False Positive Rate (FPR), and the y-axis represents the True Positive Rate (TPR).

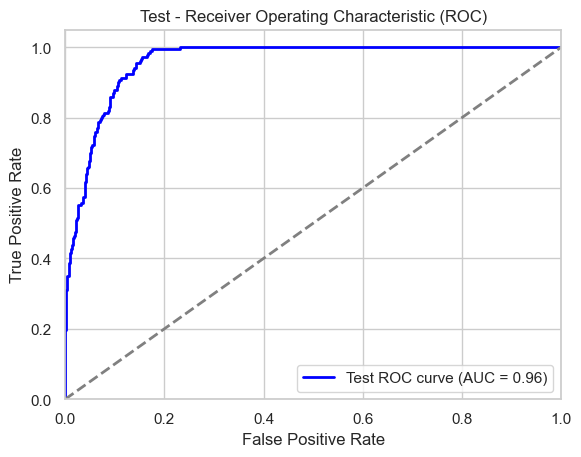
The TPR is also known as Sensitivity or Recall. It measures the ratio of correctly predicted positive instances to all actual positive instances. The FPR is the ratio of incorrectly predicted positive instances to all actual negative instances.

* AUC:

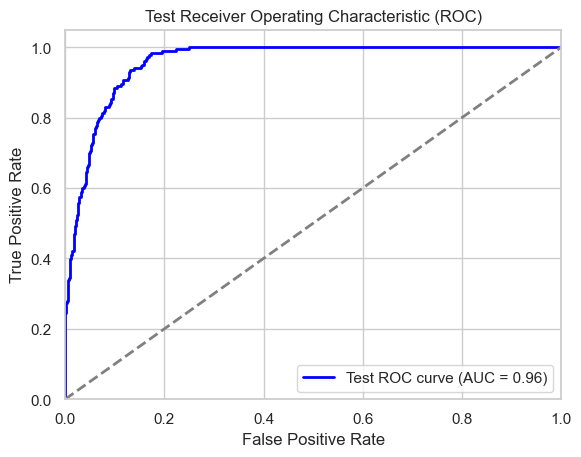
The AUC is a single scalar value that represents the area under the ROC curve. It quantifies the overall performance of a binary classification model. The AUC value ranges between 0 and 1, where a value of 0.5 indicates random guessing, and a value of 1 indicates perfect separation of the classes.

A higher AUC value indicates better discrimination between the classes and better model performance. The AUC can also be interpreted as the probability that a randomly chosen positive instance will have a higher predicted probability than a randomly chosen negative instance.

XGBClassifier ROC Curve:



Logistic Regression ROC Curve:



**5.5. Model Deployment Strategy:**

For the model deployment, our team developed a user-friendly application tailored to assess customer purchase patterns. When a specific customer is selected:

Purchase Probability: The app promptly determines if the chosen customer is likely to make a future purchase.

Purchase Timeline Distribution: If a positive purchasing probability is detected, the app presents a comprehensive bar graph detailing the likelihood distribution of the customer's next purchase. This distribution is categorized into three timeframes:

Within the next 30 days,

Between 31 to 60 days,

And between 61 to 90 days.

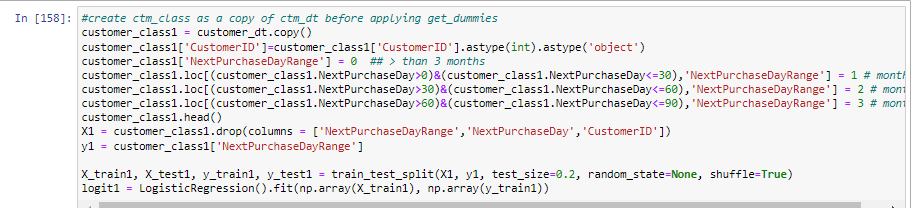
Historical Purchase Data: Users can gain insights from the customer's past transactional behavior. This historical data offers a nuanced understanding of the customer's purchasing habits over time.

Similar Customer Analytics: The app identifies customers with comparable purchasing patterns, facilitating a broader understanding of shared behaviors and tendencies within the customer base.

Product Recommendations: Independent of whether the customer is expected to make a purchase or not, the application furnishes a curated list of the top 10 product recommendations. These suggestions are data-driven, ensuring relevance and appeal to the targeted customer.

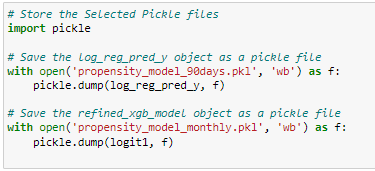
In summary, our application serves as a comprehensive tool, amalgamating predictive analytics with historical data, to drive actionable business strategies centered on customer behavior.

To deploy our application, we have used Mercury for which we have created the below Model which helps us create a model that uses customer information to predict when they might make their next purchase, and it checks how accurate those predictions are using training and testing data.



Below are the steps we have used to deploy Jupyter notebook into an APP:

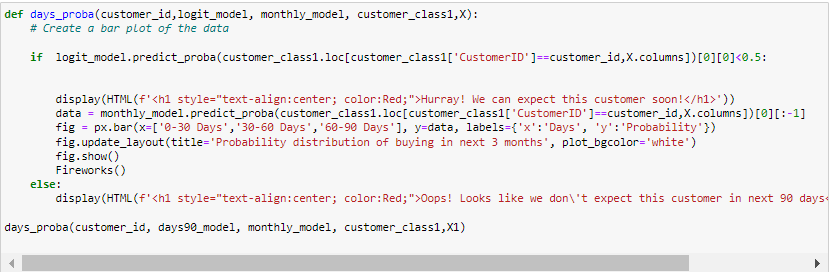
1. Train the model after performing the through EDA and data preprocessing.
2. After evaluating the various models we picked the best models based on goodness of fit measures.
3. We converted them into pickle files so we don’t have to retrain and perform modelling everytime.
4. We developed another program to read the pickle files and historic data of customers from a parquet file (Csv converted to optimize the size).
5. Transformed the jupyter notebook into an APP using mercury.
6. We can also reload the pickle files when we fine tune to model, but hosting on the web took a toll because of package and versioning issues.



The below code helps us use that tool for a specific customer. We need to give the customer's ID to the code, and it will use two different prediction models to tell us something interesting.

First, the code checks if the customer is likely to shop within the next few days. If the tool thinks the customer is likely to buy soon, it will show a message like "Hurray! We can expect this customer soon!" in big red letters. It will also make a graph that shows the probability of the customer shopping in different time ranges, like "0-30 days," "30-60 days," and "60-90 days." This graph helps us see when the customer might come back to shop.

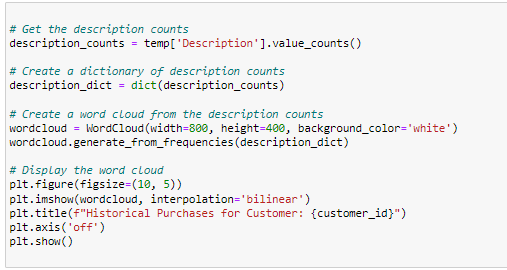
If the tool thinks the customer won't buy in the next 90 days, it will show a different message like "Oops! Looks like we don't expect this customer in the next 90 days." This helps us know that this customer might take some time before they shop again.



The code below helps us get a clear picture of how much money this specific customer spends on shopping each month. This can give us insights into their shopping habits and whether they spent more or less during certain periods.



The below code helps us visualize what items this specific customer liked to purchase the most, using a colorful and easy-to-understand picture.

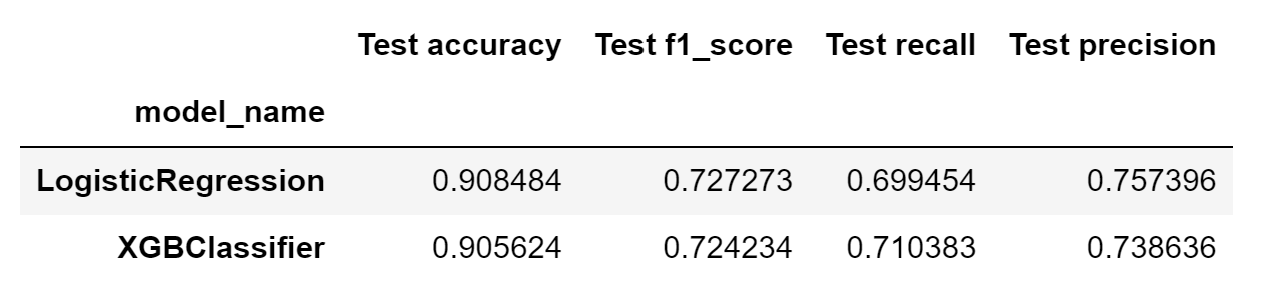


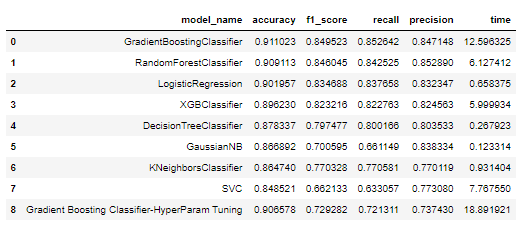
The code below helps us find customers who shop in a similar way, figures out what products they liked, and then suggests similar products to the customer we're interested in. This way, we can help the customer discover new things they might like based on what others have enjoyed.



**6. Findings and Conclusions**

6.1. Model Results, performance results, visualizations





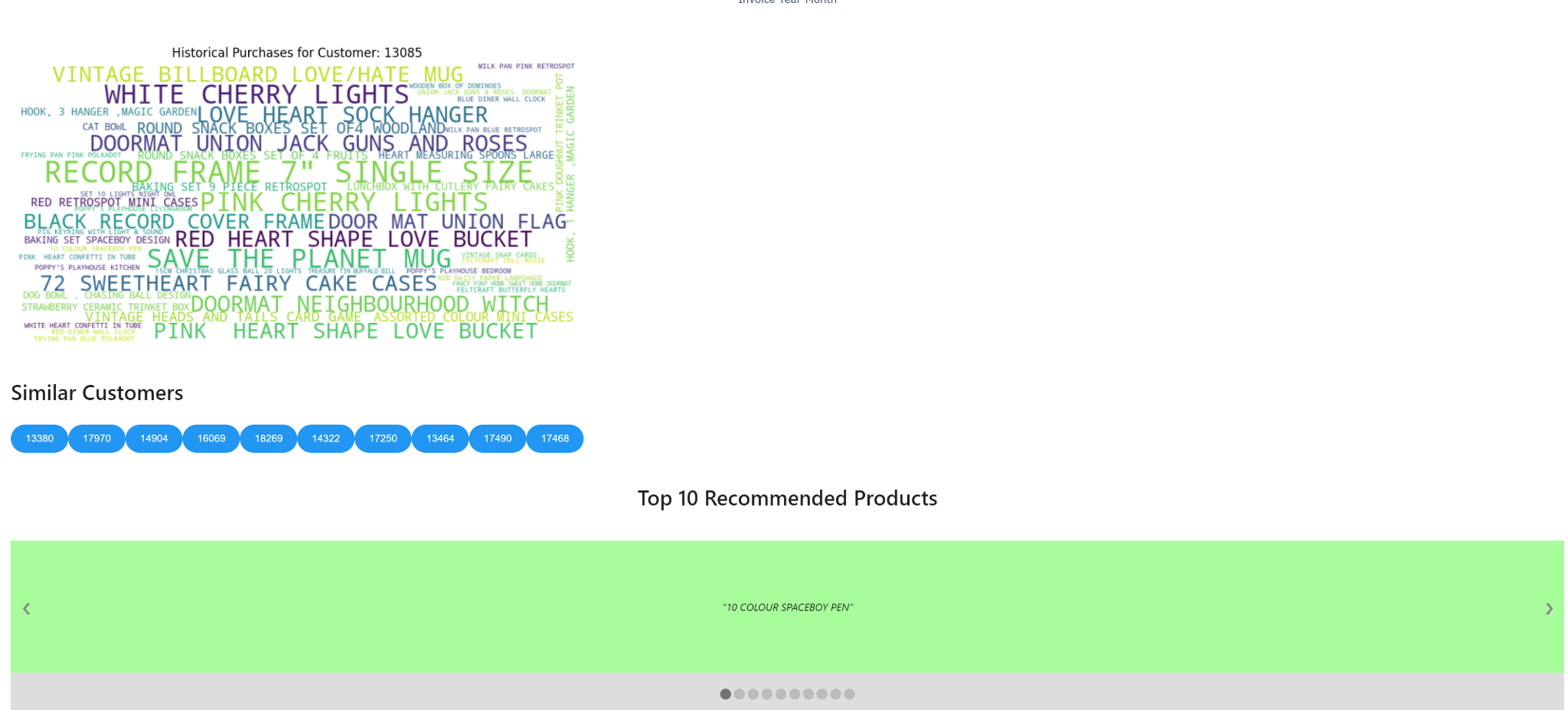
Case 1: Customer Prediction - MISS

The screenshots below are captured from the final converted APP.  
Customer ID - 13085

We can see that this Customer will not buy products in the next quarter, also we have given a list of products this customer has bought previously.

We have shown all the similar customers who have a similar trend of not buying the products for next quarter along with some recommendations of products.





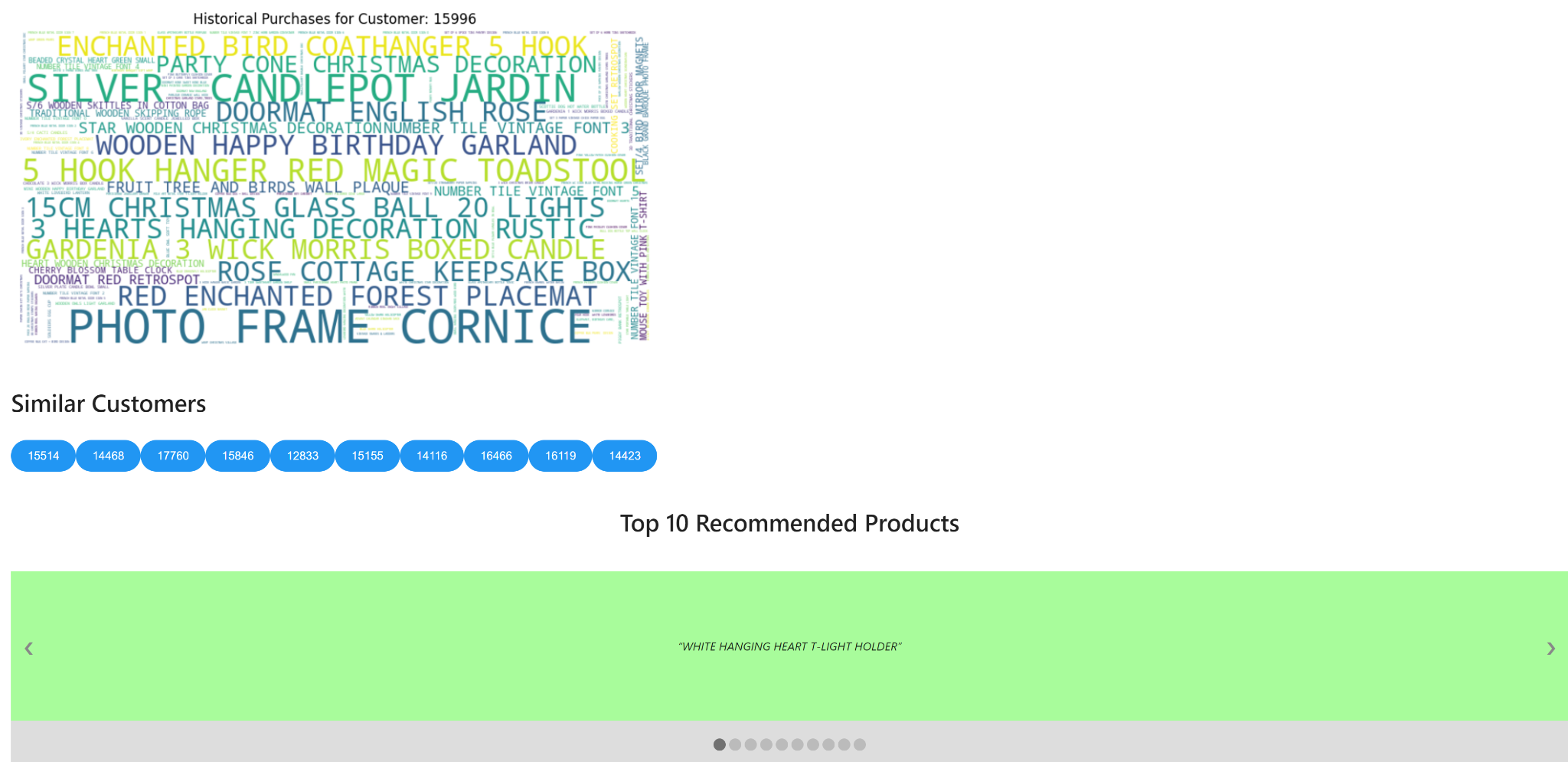
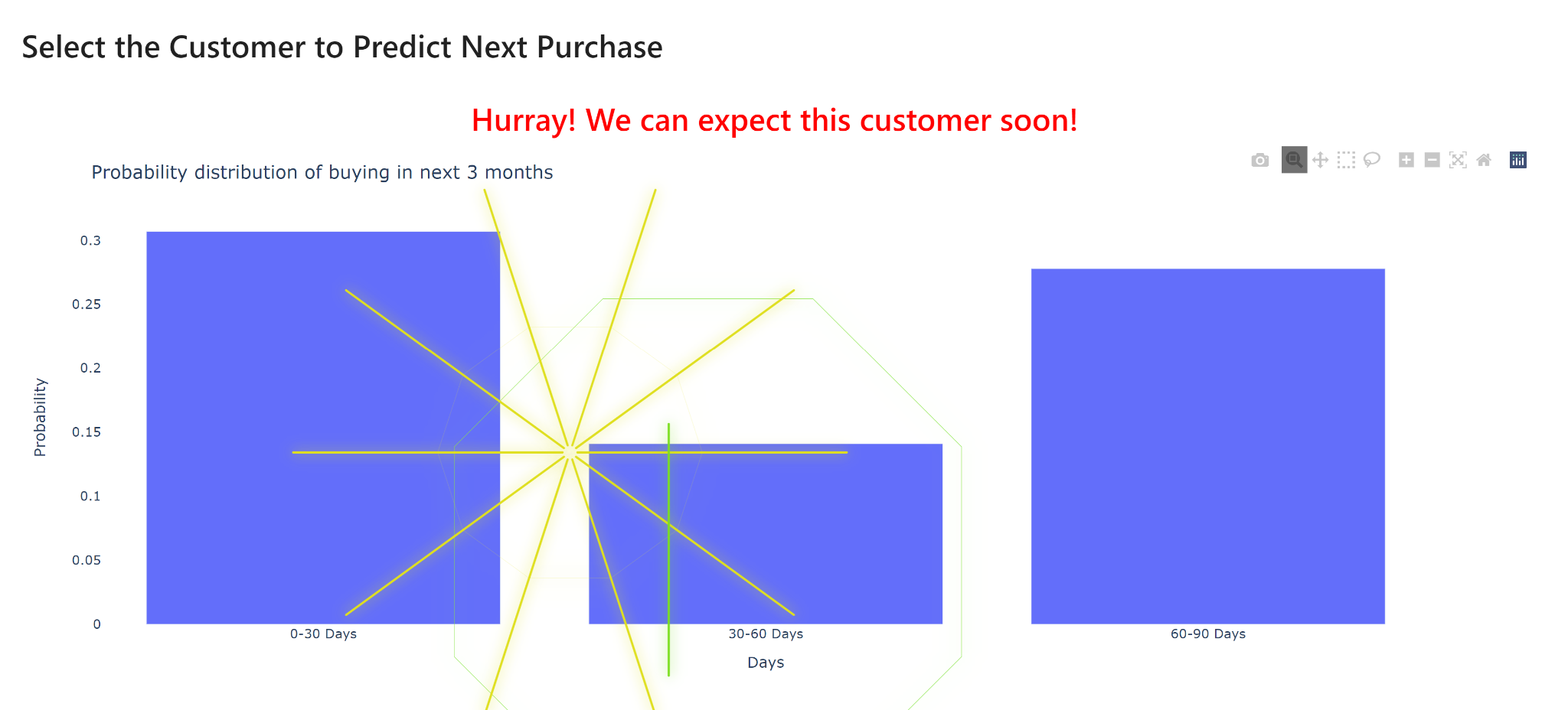
Case 2: Customer Prediction - HIT

The screenshots below are captured from the final converted APP.

Customer ID - 15996

We can see that this Customer has a probability to buy products in the next quarter, the probability of this customer buying in the next 30 days is around 30%, also we have given a list of products this customer has bought previously.

We have shown all the similar customers who have a similar trend of buying the products for next quarter along with some recommendations of products that this customer can buy.



6.2. Validating assumptions and impact ($/hrs.) based on the problem statement

This model can have a significant impact on a business’s bottom line. By predicting a customer’s next purchase, businesses can gain a competitive edge in the market and deliver value to their customers. This can help businesses improve their operational efficiency and better understand their customers. Additionally, by identifying high-value customers and creating strategic customer segments based on their potential value, businesses can run personalized campaigns with targeted sales, marketing, and support efforts. This can help guide product development by focusing on features that increase customer value and optimize sales or marketing strategy and allocate budget more accurately for customer outreach.

6.3. Practicality for the business and any possible extension to other areas.This analysis is not only relevant for online retailers, but also for traditional businesses that want to understand their customers better and improve their operational efficiency. By learning from this dataset, we can gain a competitive edge in the market and deliver value to our customers.

7. Lessons Learned and Recommendations

7.1. Next Steps along with additional methods/algorithms/models that can be used  
 Integration of Real-Time Model run and execution.

Integrating our current solution to Azure Services.

Model based approach for Product Recommendations.

Exploring features to enhance model performance.

**8. REFERENCES**

8.1. Literature Review:

Customer Propensity Model:

A customer propensity model is a predictive analytics technique used to estimate the likelihood of a customer taking a specific action, such as making a purchase, subscribing to a service, or clicking on an advertisement. It involves analyzing historical customer data to identify patterns and factors that influence customer behavior. This model helps businesses tailor their marketing and sales strategies by targeting customers who are most likely to engage in a desired action. By focusing efforts on high-propensity customers, businesses can optimize resource allocation and improve conversion rates.

RFM (Recency, Frequency, Monetary) Analysis:

RFM stands for Recency, Frequency, and Monetary Value, and it is a marketing and customer segmentation technique commonly used in the field of e-commerce and direct marketing

RFM analysis is used to categorize and target customers based on their past behaviors and interactions with a business, with the goal of optimizing marketing efforts and improving customer engagement.

1. Recency (R): This refers to how recently a customer has made a purchase or interacted with the business. Customers who have made a purchase more recently are often considered more valuable since their engagement is more current and relevant. Recency is typically measured in terms of days, weeks, or months since the last transaction.

2. Frequency (F): Frequency measures how often a customer makes purchases or engages with the business. Customers who make frequent purchases are often considered more loyal and engaged with the brand. Frequency can be measured by counting the number of transactions within a certain time period.

3. Monetary Value (M): This component measures the amount of money a customer has spent on purchases. Customers who have made larger purchases or spent more money overall are often considered more valuable to the business. Monetary value can be measured as the total monetary value of purchases over a specific time period.

RFM analysis involves assigning numerical values to each of these three components for each customer, and then grouping customers into segments based on their RFM scores. These segments can help businesses tailor their marketing strategies and offers to different customer groups. For example:

- High-RFM customers (recent, frequent, high spenders) might receive special offers or exclusive promotions to encourage repeat purchases.

- Medium-RFM customers might receive reminders or targeted discounts to re-engage them.

- Low-RFM customers might receive reactivation campaigns or incentives to increase their engagement.

By analyzing customer behavior through the RFM framework, businesses can better understand their customers' preferences and behaviors, leading to more effective and personalized marketing strategies.

RFM Formula:

The RFM formula involves the calculation of three pivotal values:

Recency (R):

Recency is calculated by subtracting the date of the customer's last transaction from the present date.A lower R value indicates recent engagement, while a higher value signifies prolonged inactivity.

R = Present Date - Last Transaction Date

Frequency (F):

Frequency is the count of transactions made by a customer within a specific timeframe.A higher F value suggests regular interactions, while a lower value implies infrequent activity.

F = Number of Transactions

Monetary Value (M):

Monetary Value represents the total amount spent by a customer on transactions.Higher M values correspond to higher spending habits, while lower values indicate more restrained purchasing.

M = Sum of Monetary Amounts

In summary, RFM encapsulates crucial customer behavior dimensions using mathematical expressions. It quantifies recency, frequency, and monetary influence, offering a multidimensional perspective that businesses can leverage for segmentation, targeted marketing, and predictive modeling.

K-means Clustering with RFM:

Central to our segmentation strategy was the utilization of the K-means clustering algorithm in conjunction with the RFM dimensions. This technique facilitated the categorization of customers into distinct clusters based on their similarities in Recency, Frequency, and Monetary Value/Revenue. By grouping customers with akin behavior, we paved the way for customized marketing strategies tailored to each cluster's preferences and habits.

Given:

- n: Number of data points (customers)

- k: Number of clusters

- X: Data matrix containing RFM values for each customer (n x 3 matrix)

Algorithm Steps:

1. Initialization:

Initialize k cluster centroids (μ) randomly or using a specific strategy.

μ^(0) = [μ\_1^(0), μ\_2^(0), ..., μ\_k^(0)]

2. Assignment:

For each data point i (1 ≤ i ≤ n), compute the distance to each cluster centroid and assign the data point to the cluster with the nearest centroid.

c^(t)(i) = arg min\_j ||X\_i - μ\_j^(t-1)||^2

Where c^(t)(i) is the cluster assignment of data point i in iteration t.

3. Update Centroids:

Update each cluster centroid by calculating the mean of all data points assigned to that cluster.

μ\_j^(t) = (1 / |S\_j^(t)|) ∑\_(i∈S\_j^(t)) X\_i

Where S\_j^(t) is the set of data points assigned to cluster j in iteration t.

4. Convergence Check:

Repeat steps 2 and 3 until the centroids converge (change minimally) or a maximum number of iterations is reached.

||μ\_j^(t) - μ\_j^(t-1)|| < ε

Where ε is a small threshold.

5. Output:

The final cluster assignments c^(t)(i) provide the grouping of data points into k clusters.

This algorithm iteratively refines cluster assignments and centroids to find a solution that minimizes the sum of squared distances between data points and their assigned centroids.

Clustering and Segmentation:

Clustering is a technique used to group similar customers together based on certain characteristics. We are using clustering to segment customers into distinct groups based on their RFM scores. This helps in understanding customer behavior patterns and tailoring strategies for each segment.

Classification Models:

1. Random Forest: A decision tree ensemble method that combines multiple decision trees to improve predictive accuracy and control overfitting.

2. XGBoost: An optimized gradient boosting algorithm that excels in handling complex relationships in data and minimizing errors.

3. Neural Networks: Deep learning models inspired by the human brain, capable of learning complex patterns from data.

4. SVM Classifier: Support Vector Machines find a hyperplane that best separates different classes, making them suitable for binary and multiclass classification tasks.

Model Evaluation Metrics:

Accuracy: Proportion of correctly classified instances.

Precision: Proportion of true positive predictions among all positive predictions.

Recall: Proportion of true positive predictions among all actual positives.

F1-Score: Harmonic mean of precision and recall, providing a balanced measure.

ROC Curve and AUC: Visualization of model performance across different classification thresholds.

Confusion Matrix: Tabulates true positive, true negative, false positive, and false negative counts.

Business Impact:

This use case will help the business in several ways:

1. Targeted Marketing: By identifying high-propensity customers, the business can direct marketing efforts towards those who are more likely to respond positively.

2. Resource Optimization: Resources can be allocated more efficiently by focusing efforts on customers with higher conversion probabilities.

3. Personalization: Tailoring marketing messages and offerings based on customer segments leads to improved engagement and customer satisfaction.

4. Revenue Increase: Improved targeting and personalized strategies often lead to higher conversion rates and increased revenue.

5. Customer Retention: Understanding customer behavior allows for proactive measures to retain existing customers by offering incentives or solutions before they churn.

Overall, the customer propensity model enhances the business's decision-making capabilities, resulting in improved customer engagement, increased efficiency, and higher revenue.

8.2. References:  
 Y. Zhang, "Prediction of Customer Propensity Based on Machine Learning," 2021 Asia-Pacific Conference on Communications Technology and Computer Science (ACCTCS), Shenyang, China, 2021, pp. 5-9, doi: 10.1109/ACCTCS52002.2021.00009.

<https://ieeexplore.ieee.org/abstract/document/9407585>

Mahboubeh Khajvand, Kiyana Zolfaghar, Sarah Ashoori, Somayeh Alizadeh,

Estimating customer lifetime value based on RFM analysis of customer purchase behavior: Case study, Procedia Computer Science,Volume 3, 2011, Pages 57-63, ISSN 1877-0509, <https://doi.org/10.1016/j.procs.2010.12.011>  
 Milligan, G. W., & Cooper, M. C. (1987). Methodology Review: Clustering Methods. Applied Psychological Measurement, 11(4), 329–354. <https://doi.org/10.1177/014662168701100401>  
 Rokach, L., Maimon, O. (2005). Clustering Methods. In: Maimon, O., Rokach, L. (eds) Data Mining and Knowledge Discovery Handbook. Springer, Boston, MA. <https://doi.org/10.1007/0-387-25465-X_15>

T. T. Joy, S. Rana, S. Gupta and S. Venkatesh, "Hyperparameter tuning for big data using Bayesian optimisation," 2016 23rd International Conference on Pattern Recognition (ICPR), Cancun, Mexico, 2016, pp. 2574-2579, doi: 10.1109/ICPR.2016.7900023.

<https://ieeexplore.ieee.org/document/7900023>

A. K. Santra and C. Josephine Christy, "Genetic Algorithm and Confusion Matrix for Document Clustering," CARE School of Computer Applications Dean, Trichy – 620 009, India, and Research Scholar, Bharathiar University, Coimbatore – 638401, India.