Churn Prediciton System

A Executive Summary

Customer churn or the loss of customers is a significant concern for fast-moving consumer goods retailers like FoodCorp, where it is closely linked to customer loyalty and regularity. Predicting which customers are likely to churn can help FoodCorp to take proactive measures to retain their customers.

This report aimed to develop a predictive model for customer churn using transactional data. The first step was to define churn based on the initial analysis and statistics provided by ConsultingCorp. Using this definition, past transactional data of FoodCorp was analyzed to label customers as churned or non-churned on each recorded day. This was followed by an initial exploration of the dataset which revealed that the data was captured over a span of 609 days with 2,660 unique users. The task was then to build a Machine Learning (ML) model to predict customers who may churn. To capture the trends and patterns in customer behaviour over time, lagged input features were generated to be given to the ML model. These lagged features were created for each customer by aggregating their transactional data over different time periods, ranging from weekly and monthly intervals. On the other hand, an output feature of the ML model was created in which the customers labelled as churners were represented by 0 and non-churners were represented by 1.

The generated features were then used to train and evaluate different classification ML models, including Logistic Regression, Random Forest Classifier, and Light GBM Classifier. The performance of these models was evaluated using metrics such as accuracy, precision, recall and specificity. The analysis showed that Random Forest Classifier outperformed other models by achieving accuracy of 82% in predicting churn. The performance of the model was further improved by tuning hyperparameters and selecting important features. Finally, the model's performance was evaluated on test dataset, where the model achieved an unbiased accuracy of 81.9%.

Furthermore, the pen profiles of the customers who may churn or not which would aid in marketing. The customers with characteristics such as inconsistent in number of visits, less number of purchases over time, absence over two weeks after last purchase, were profiled as churners. Based on these profiles, the report identified several actionable insights for marketing that FoodCorp can use to prevent customer churn. For instance, the company should target customers who have not made a purchase in a while with personalized offers and promotions to incentivize them to make a purchase.

B Current levels of churn

ConsultingCorp proposes a methodology to define churn, which involves calculating the acceptable number of inactive days for a customer, assuming the knowledge of customer's return dates. Considering FoodCorp's decisions to ignore the time of day and multiple visits on the same day, ConsultingCorp formalizes a definition of churn as: "If the time elapsed between the date of the last purchase and the current date is greater than β (the period of inactivty); then the customer is considered to have churned. Otherwise, is not considered to have churned."

To determine the value of β, their report discusses two statistics: the distribution of times between visits and the Percentage of Customers who are Considered Churners (PCCC) for an increasing period of inactivity. After analyzing the plots of these two statistics, **this report recommends a churn definition with an inactivity period of 26 days**. This selection will target approximately 22% of active customers, thereby balancing the number of target customers and the company's resources spent for intervention. Notably, half of FoodCorp's customers have a median number of days between visits less than 26 days. Furthermore, it has been observed that as the interval of inactivity period decreases, the proportion of customers who will churn increases. Accordingly, the report summarises the levels of churn as follows:

- High level of churn: It can be described as selecting the acceptable inactivity period of less than 8 days, as more than 75% of total customers have inactivity period greater than this. Moreover it will target high number of active customers from 39.95% to 80.84%.
- Moderate level of churn: It can be described as selecting the acceptable inactivity period of less than 38 days, as more than 60% of total customers have inactivity period less than this. Hence, less than half customers fall in the category of churn. It will only target 18.18% to 22.22% of active customers which is a reasonable proportion of people to target.
- Low level of churn: It can be described as selecting the acceptable inactivity period of less than 71 days, as more than 75% of total customers have inactivity period less than this. It will only target 13.47% of active customers which is a low proportion of people to target.

C Churn Prediction System

To design an effective churn prediction system, an initial exploration of the data is necessary to gain insights. This report analysed the transactional data consisting of 31,738 purchases made by 2,660 customers. To account for multiple purchases made by a customer on the same day, these transactions were reduced to 29,774. The timeline for this data spans from 22 July, 2020 to 22 March, 2022, representing 608 days in total. The median time interval between consecutive purchases was found to be 5 days. Based on these findings, this section discusses the process of feature engineering, model selection, training, and evaluation to develop an accurate churn prediction system.

C.1 Features

The features required for churn prediction were created by merging the tables available in the provided SQL database, which included "customers", "products", "receipt lines", "receipts", and "stores". The relationship between these tables is depicted (in schema) in Figure A.1 of Appendix A. Feature engineering plays a crucial role in preparing time series data for analysis. The initial step involved checking for missing temporal data points. Since each row in the merged dataset was supposed to represent a customer on a specific day, all the missing temporal points for each customer were filled in to ensure that the time series data was complete. The resulting cleaned dataset was then used to create input and output features for the churn prediction system.

To create lagged features, aggregate functions such as sum, average, or variance were applied on the previous temporal data points. SQL window functions, such as Tumbling Window and Expanding Window, were used to achieve this, as shown in Figure 1. For the churn prediction, one output was created, which was the binary classification task. A list of generated input features along with the output feature is presented in table 3. It depicts the rationale behind creating them and details of the aggregate functions utilised.

C.2 Prediction Approach

The churn prediction approach involved several steps, starting with data pre-processing followed by an appropriate model selection and its training and evaluation to ensure accurate prediction of the output.

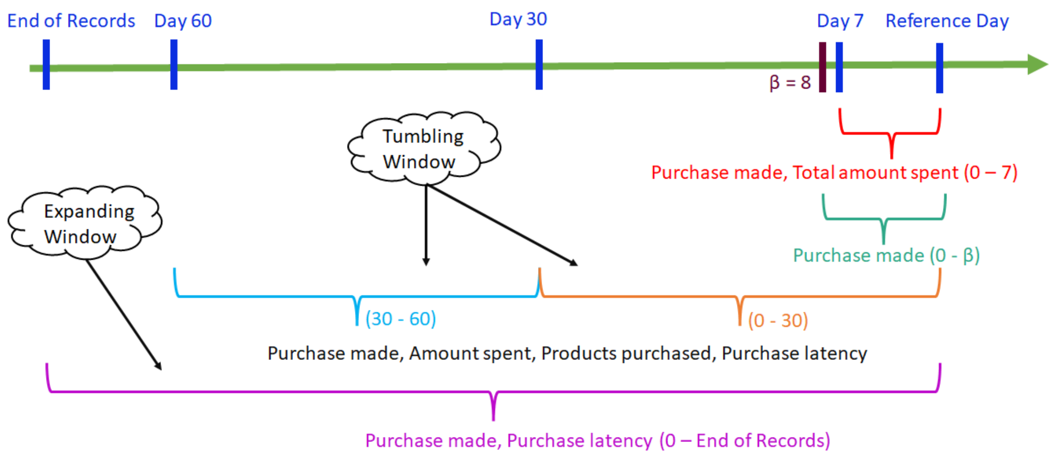

End of Records
Day 60
Day 30
Day 7
Reference Day
β = 8
Tumbling Window
Expanding Window
Purchase made, Total amount spent (0 – 7)
Purchase made (0 - β)
(30 - 60)
(0 - 30)
Purchase made, Amount spent, Products purchased, Purchase latency
Purchase made, Purchase latency (0 – End of Records)

Figure 1: Approach for creating Lagged Features

Table 1: Features created for churn prediction system

Feature	Rational	Details	No.
Customer churn	To predict if a customer on a particular reference day will churn or not.	Binary classification task. A data point is a customer on a particular day.	1 (Output)
Purchases made	Churners may have fewer number of purchases over the time period.	Number of purchases (sum) made in last x days with respect to the reference day, where x is 7, 30, 26 (β), time period between 30 to 60 and first purchase date.	5
Amount Spent	Churners may reduce their spending over the time period.	Sum, mean and variance of the amount spent in the last x days with respect to reference day, where x is 7, 30 and time period between 30 to 60. (Mean and variance not calculated for last 7 days as its a short time period)	7
Products purchased	Churners are more likely to buy fewer products over the time period.	Number of distinct products purchased (sum) in the last x days with respect to the reference day, where x is 30, time between 30 to 60	2
Purchase recency	An indication of regularity. Customers who haven't made a purchase in a while are likelier to churn.	Number of days (sum) since the last purchase made by a customer.	1
Customer lifetime	An indication of loyalty	Number of days (sum) since the first purchase made by a customer.	1
Purchase latency	An indication of how long a customer is active	Number of days customer was active (sum) in last x days with respect to reference day, where x is 30, time period between 30 to 60 and first purchase date.	3
Seasonality	An indication of customer behaviour with respect to time of the year.	Time of the year identified using reference date.	1

(i) Data preprocessing

The following data pre-processing steps were conducted to prepare the dataset for churn prediction:

- Removing Outliers: Extreme values, or outliers, were removed from the dataset to prevent them from adversely affecting the accuracy of certain ML models. The 3 sigma rule, a widely used method for detecting outliers, was employed. The rule states that for a normal distribution, approximately 68%, 95% and 99.7% of the data fall within 1, 2, and 3 standard deviations of the mean respectively. The features such as quantity and value of products in the "receipt lines" table, which have a distribution similar to normal distribution were cleaned by removing outliers.
- Removing Correlated features: To ensure that the lagged features created for the time series data were not redundant, correlation was calculated using the coefficient of correlation, and a heat map was plotted. The highly correlated features were then removed from the dataset.
- Feature Scaling: It is important to standardize all the input features to have the same scale to avoid any bias towards a particular feature. This was achieved by using StandardScaler() from the sklearn package, which normalizes the numerical values by scaling them to have a mean value of zero and a standard deviation of one.

(ii) Train, validation and test split

In time series prediction, the temporal ordering of the data plays a crucial role in creating accurate training, validation, and test sets. To ensure this, reference points for training, validation, and testing were defined. The validation reference day was defined as β units prior to the test reference day. The validation set was used to assess the performance of different models and to fine-tune the hyper-parameters of the selected model. Functions were developed to generate training, test and validation sets based on a designated reference day. The primary objective of the functions was to retrieve the indices of the created dataset from SQL to produce multiple sets for training and validation, which would be beneficial in repetitive testing scenarios. The two functions are as follows:

- The first function accepted a reference day or a list of reference days (list) as input and returned the indices of all data points/customers active on those days. If a single day was provided as input, it was transformed into a list with a single element. The function generated an SQL query to filter data points based on the days in the list. This function was used to generate the test set.
- The second function took a reference day and returned indices of both training and validation sets. It called the first function twice to get the indices for the validation and training sets separately. It generated the range of training reference days as a numpy array, which started from (reference day - beta days) and ended at the reference day, inclusive. It then passed this array of training reference days to the first function to get the indices of all data points present on those days for the training set.
- To generate multiple training and validation sets, the second function was called iteratively for five times, resulting in five different pairs of sets. At each iteration, the reference day was shifted back in time by a fixed number of beta days before obtaining the training and validation sets. These sets were then used for cross-validation to calculate the mean accuracy score for each model.

(iii) Model Selection and Cross-Validation

In order to accurately predict churn, a set of classification models were selected and evaluated. The chosen models and their respective benefits are as follows:

- Logistic Regression- It is a linear model and the simplest model of all, that also provides baseline accuracy for model selection. Moreover, the model is interpretable and can be trained quickly on large datasets.

- Random Forest classifier - It is a non-linear model based on ensemble-learning method that combines multiple decision tress to make predictions. It is known to provide high accuracy for classification tasks and is resilient to overfitting. The algorithms grow trees horizontally.
- Light GBM classifier - It is a gradient boosting framework that also uses a tree-based learning algorithm. It produces much more complex trees by following leaf wise split approach rather than a level-wise approach, thus providing higher accuracy. It is equally well with large datasets with a significant reduction in training time. The algorithm grows trees vertically.

To evaluate each model's performance, cross-validation was used to split the dataset into training and validation sets. This helped to detect overfitting and underfitting of the model. The models were trained on the training set and the performance of the model was evaluated on the validation set where the performance measure was accuracy. Based on this, the accuracy of the Logistic Regression, Random Forest Classifier and Light GBM classifier was 82.2%, 80.2%, and 81.7% respectively.

C.3 Model Evaluation

Model evaluation is a crucial step in the machine learning pipeline, as it helps assess the model's performance on new, unseen data. A confusion matrix is commonly used to summarize the predictions made by a classification model in terms of True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN). The chosen models were fit to the training data and their accuracy, precision, sensitivity, and specificity were calculated using the confusion matrix as follows:

$$\begin{array}{c|cccc} \text{Confusion Matrix} & \text{Baseline Classifier} & \text{Logistic Regression} & \text{Random Forest} & \text{LGBM Classifier} \\ \begin{bmatrix} TP & FN \\ FP & TN \end{bmatrix} & \begin{bmatrix} 0 & 235 \\ 0 & 264 \end{bmatrix} & \begin{bmatrix} 205 & 30 \\ 59 & 205 \end{bmatrix} & \begin{bmatrix} 198 & 37 \\ 51 & 213 \end{bmatrix} & \begin{bmatrix} 208 & 27 \\ 58 & 206 \end{bmatrix} \end{array}$$

In this classification problem, the business objective was to prevent customers from churning. When evaluating performance metrics for machine learning models, the report identified **the "churn" class as negative, while the "non-churn" class was identified as positive**. It was found that accuracy alone was not sufficient for measuring predictive performance. Therefore, **specificity** was identified as the appropriate metric for predicting churners (i.e., customers at risk) as it maximized true negatives and minimized false negatives. As can be observed from Table 2, the specificity of Random Forest Classifier is the highest among all the other models. Although, through cross-validation, the accuracy of Random Forest Classifier was the lowest, but based on high specificity, this report chose **Random Forest Classifier as the ML model to predict churn**.

Table 2: ML Model Evaluation using different Performance Measures

	Accuracy $\frac{(TP+TN)}{(TP+TN+FP+FN)}$	**Precision** $\frac{TP}{(TP+FP)}$	**Sensitivity** $\frac{TP}{(TP+FN)}$	**Specificity** $\frac{TN}{(TN+FP)}$
Point Predictor	52.9%	Undefined	0%	100%
Logistic Regression	82.16%	77.65%	87.23.3%	77.65%
Random Forest	82.36%	79.51%	84.25%	80.68%
Light GBM Classifier	82.96%	78.19%	88.51%	78.03%

C.4 Model Improvement

This section discusses methods such as hyperparameter tuning and feature selection to improve the performance of the selected ML model.

(i) Meta-Parameter Tuning

The performance of a model heavily depends on the values of its hyperparameters. Improper tuning of these hyperparameters can lead to overfitting or underfitting of the training data, resulting in poor performance when tested on unseen data. In this study, hyperparameter (meta-parameter) tuning was conducted using the GridSearchCV method for the Random Forest Classifier model. The maximum depth and maximum features were identified as the two hyperparameters to be considered. The former controls the model's complexity while the latter controls its randomness.

To conduct the tuning process, a dictionary named "grid" was created to include the hyperparameters and their corresponding values to be tried. The "max_depth" hyperparameter was tested with the values of [5, 8, 10, 15, 20, 25, 30, 35, and 40], while "max_features" was tested with the values of [1, 'log2', 'sqrt']. Then, different combinations of hyperparameters were tried and the model's performance was evaluated. The optimal combination of "max_depth" and "max_features" hyperparameters, were 8 and 'log2', respectively, and this increased the model's overall accuracy from 80.2% to 82.5%.

(ii) Feature Importance and Selection

To enhance the model's performance, the unnecessary variables were removed using variable importance to prevent the curse of dimensionality. Two methods were utilized to calculate feature importance: univariate variable importance and model-based importance. They are described as follows:

- Univariate variable importance: GenericUnivariateSelect method was utilised to evaluate each feature's relevance to the output variable. The top 10 features were selected based on their scores which were calculated using mutual information statistical test. This method is considered as a filter-based approach because it selects features independently of the ML algorithm used.
- Model-based feature importance: Random Forest Classifier calculates feature importance by measuring how much the inclusion of a given feature improves the predictive performance of the model. This report used the method, 'feature_importances_' associated with the trained Random Forest model, to obtain feature importance scores.
- Comparison of both methods: While the univariate method was computationally simple, it had limitations as it assumed that features were independent and did not account for interactions between them. Therefore, the report opted for **the model-based method to select features**, despite being computationally complex. The figures 2 and 3 display the top 10 features selected by both methods.

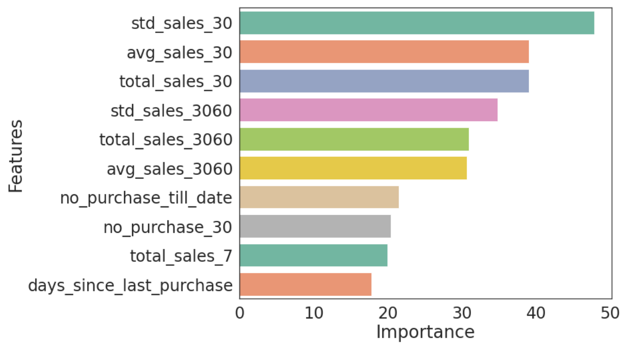


Figure 2: Variable Importance based on filter based method

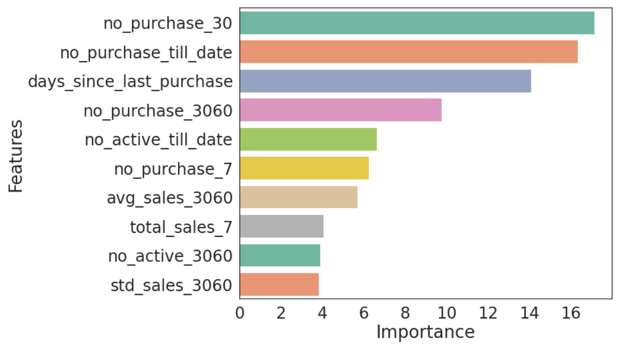


Figure 3: Variable Importance based on model based method

C.5 Summary

An efficient churn prediction system was developed by utilizing SQL and Python. Initially, the data was explored to reveal that the dataset captured 2,660 unique customers over a span of 608 days. To predict churn, 20 temporal features were generated by merging five SQL tables. Only two features were selected from the tables, while the others were discarded, as the main goal was to capture the time series of the dataset. To predict churn, a binary output feature was created, where 0 represented churned customers and 1 represented active customers. Next, a comprehensive prediction approach was adopted, starting with data preprocessing to remove outliers and correlated features, followed by scaling the features to the same scale. The data was then divided into training, validation, and test sets. Five pairs of training and validation sets were created for cross-validation, which would be used to evaluate the performance of the machine learning (ML) models. The report considered three classification models: Logistic Regression, Random Forest Classifier, and Light GBM Classifier. The performance of these models was evaluated using the confusion matrix, where the churn class was classified as negative, and the metric used to evaluate model performance was specificity. Based on the specificity scores, the Random Forest Classifier was chosen as the best ML model to predict churn. The performance of the selected model was further improved through metaparameter tuning. This resulted in a maximum depth of 8 and a maximum feature selection of 'log2' for the random forest. Additionally, the model-based feature importance analysis identified 10 features that had the most significant impact on the output.

D Insights

This section presents insights on churning and non-churning customers that informs the marketing strategies for the company.

D.1 Marketing Strategy

This section analyses and provides pen-portraits of the customers who may or may not churn, which will help to inform marketing strategy. Table 3 concisely represents the pen portraits of the churners and non-churners. FoodCorp can use these factual insights presented in Table 3 about the customers to develop targeted marketing campaigns to retain non-churning customers and re-engage with churning customers. For instance, since non-churning customers tend to be more active and frequent in their purchasing behaviour, FoodCorp can offer them exclusive deals and personalized offers to encourage them to continue engaging with the company. Loyalty programs can also be introduced to reward these customers for their loyalty.

On the other hand, since churning customers tend to be less active and less frequent in their purchasing behavior, FoodCorp can create campaigns that focus on re-engaging these customers by providing incentives that encourage them to make a purchase or revisit the company. These incentives could be in the form of discounts or special offers on products or services that the customer has shown an interest in or has previously purchased. Personalized emails or targeted advertisements can also be used to remind these customers of the value that FoodCorp can provide to them.

D.2 Deriving of Insights to inform marketing

To derive the insights presented in Section D.1, a data-driven approach was used where customers were segregated into churned and non-churned groups based on the output label. For both the categories, the important features were analysed using descriptive statistics such as mean, median, and standard deviation. This report utilised median values of the features as shown in Figure B1 of Appendix B. This analysis provided valuable insights into each feature, revealing past behavior and trends of both churners

Table 3: Pen profiles of churners and non-churners

Feature	Churner	Non-churner
Purchases made	Have made an inconsistent number of purchases. They made no (median) purchase over the last seven days, on average just one purchase over a month, and also no purchase between 30 to 60 days.	Have made a consistent number of purchases, at least one purchase per week.
Amount Spent and products purchased	Have spent less amount and bought less products in last week with respect to the reference date.	Spend significantly more per week and per month.
Purchase recency	Shown absence over two weeks after last purchase.	Active for four days even after last purchase.
Customer lifetime	Have shopped for 4 months since the day of first purchase.	Have shopped for more than 8 months since the day of first purchase.
Purchase latency	Churners were active initially (more number of active days compared to non-churners) and then gradually decreased their visits, not even once a week	Non-churners visit the store at least once a week

and non-churners. This information was then used to create pen-portraits, or detailed profiles, of these two customer categories.

In addition, Partial Dependence plots were generated for specific features to further comprehend the impact of varying values of these features on the model's predictions. As demonstrated in Figure 4, which shows that output value shifts from 0 (churners) to positive value (i.e. towards non-churners) with increase in the number purchase of total purchases throughout their customer life-cyle. Thus, these plots yielded significant insights into how distinct marketing interventions can potentially influence critical factors that determine whether a customer is likely to churn or not. This data-driven methodology facilitated the identification of fundamental factors that contribute to customer churn and furnished practical and actionable insights for devising marketing strategies that aim to retain customers.

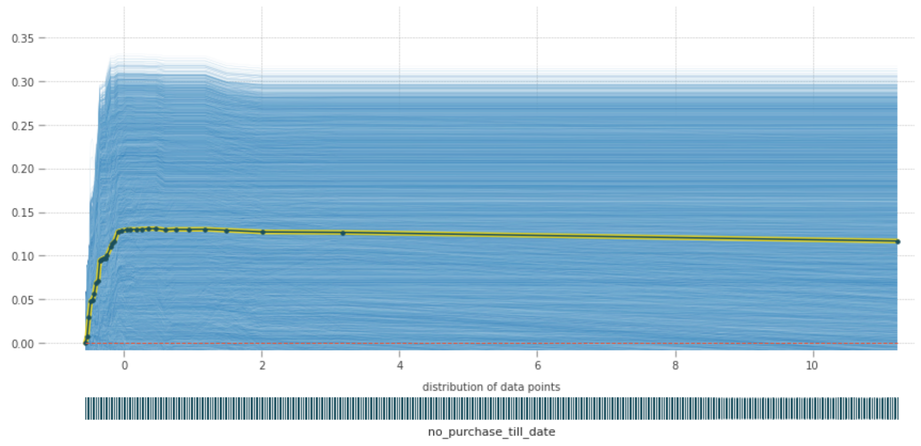


Figure 4: PDP for feature "no_purchase_till_date"

Appendix A

The following schema shows the relationship of the tables in SQL.

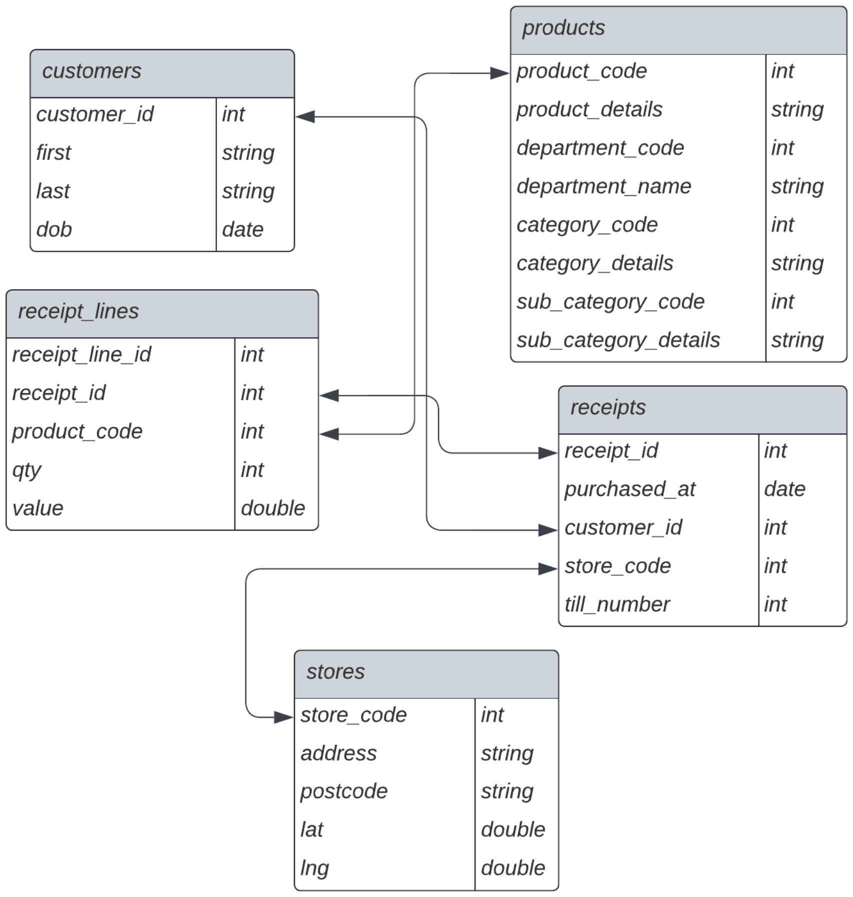
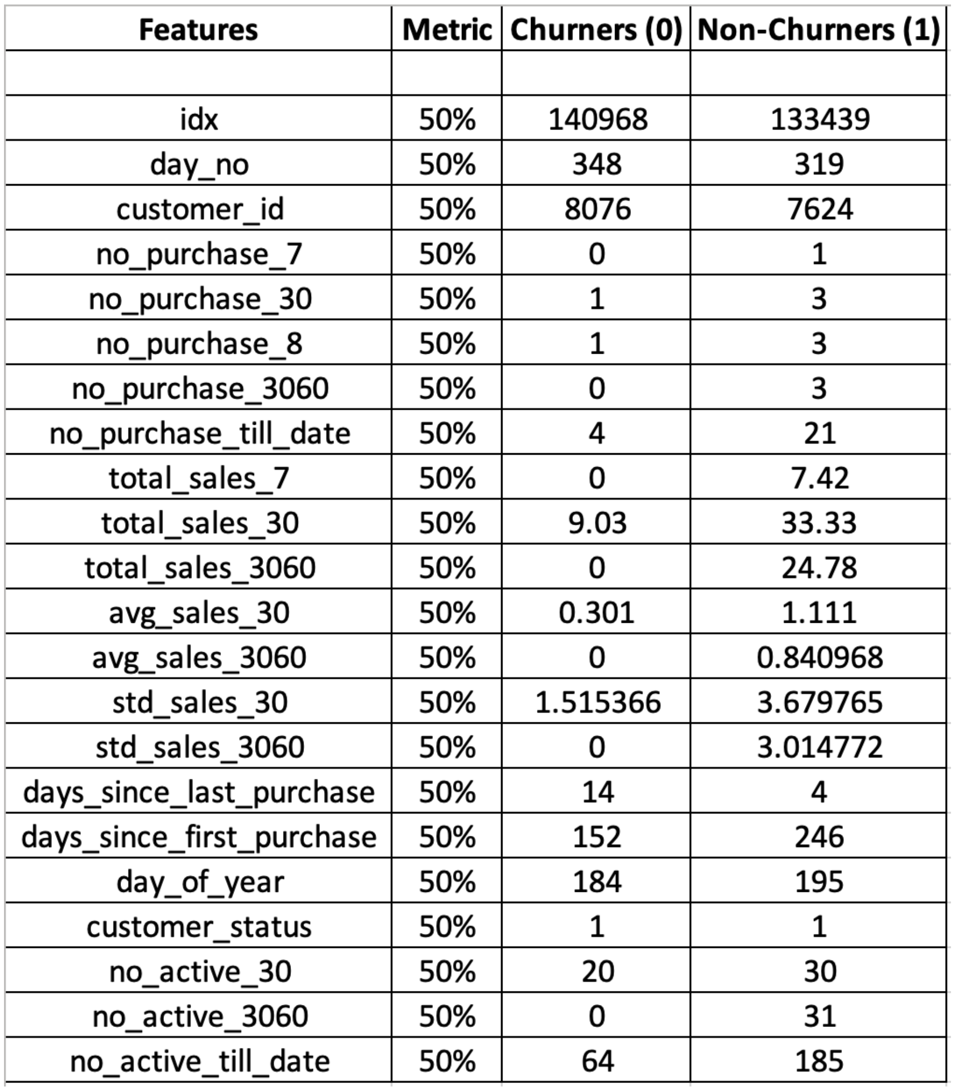


Figure A.1: SQL Database Schema showing relationship between SQL tables

Appendix B

The following figure shows the median values of the features as described for creating pen profiles:



Features	Metric	Churners (0)	Non-Churners (1)
idx	50%	140968	133439
day_no	50%	348	319
customer_id	50%	8076	7624
no_purchase_7	50%	0	1
no_purchase_30	50%	1	3
no_purchase_8	50%	1	3
no_purchase_3060	50%	0	3
no_purchase_till_date	50%	4	21
total_sales_7	50%	0	7.42
total_sales_30	50%	9.03	33.33
total_sales_3060	50%	0	24.78
avg_sales_30	50%	0.301	1.111
avg_sales_3060	50%	0	0.840968
std_sales_30	50%	1.515366	3.679765
std_sales_3060	50%	0	3.014772
days_since_last_purchase	50%	14	4
days_since_first_purchase	50%	152	246
day_of_year	50%	184	195
customer_status	50%	1	1
no_active_30	50%	20	30
no_active_3060	50%	0	31
no_active_till_date	50%	64	185

Figure B.1: Pen profile descriptive statistics

Supporting Documentation

Student ID: 20439708

Instructions for Running the Churn Model

To run the churn model, the following steps should be followed:

- Open the Databricks notebook named "Churn_Prediction_20439708.html" and run sll the cells sequentially till Section 6 -"Unbiased Accuracy of Model".
- Then run the next cell code of Section 7 - "Model Implementation" to save the file of trained model with the name "Model_final.pkl".
- Open Databricks notebook named "Predict_20439708.html" in Databricks and load the saved "Model_final.pkl" file in the Databricks environment.
- Load or connect to the SQL database, which has tables related to the transactional data by entering the url of the database in the "filepath" variable under first section called "Loading Database" section or load the SQL database in the environment. Ensure the database has same schema as shown in Appendix of the submitted report, to avoid any error in the execution of code.
- Run each cell sequentially under "Predict.html" Databricks notebook.
- The final output of the prediction will contain a CSV file containing the list of potential churners as predicted by the model.

Data Cleaning

No extra cleaning was required.

List of Figures

Figure Number	Figure Caption	Method used
1	Approach for creating Lagged Features	Powerpoint- shapes
2	Variable Importance based on filter based method	Python - seaborn library using bar plot function (Section: 5b in databricks code)
3	Variable Importance based on model based method	Python - seaborn library using bar plot function (Section: 5c in databricks code)
4	PDP for feature "no purchase till date"	Python - pdpbox library using pdp_plot function (Section 8 in databricks code)
A.1	SQL Database Schema showing relationship between SQL tables	Online tool- Lucid Chart
B.1	Pen profile descriptive statistics	Python - groupby and describe function (Section 9 in databricks code)