Customer Churn Prediction

Authors:

Muhammad Taimoor

U61736034

Muhammad Faadil

87311082

CS 699 – Data Mining

Semester Project

1st November 2022

List of Figures

List of Tables

Abstract

Customer churn, also referred to attrition, is a stage in the consumer pipeline when the customer chooses to stop using the company's products or services. Customer churn is measured using customer churn rate, which is defined as the number of people who chose to stop being a customer in a specified time frame. Customer churn is inevitable, the idea that a customer will keep on using a company's service forever is not realistic. However, if a company is experiencing a high churn rate, that may be an indication to access the factors associated with the customer and the product. There are usually several avenues to consider when it comes to customer data, by looking at the customer's journey and the experiences they have had, analyst access features to measure the key experiences, and quantify them in order to predict which customer is likely to churn.

To detect early signs of potential churn, one must first develop a holistic view of the customers and their interactions across numerous channels, including store/branch visits, product purchase histories, customer service calls, Web-based transactions, and social media interactions, to mention a few.

As a result, by addressing churn, these businesses may not only preserve their market position, but also grow and thrive. The more customers they have in their network, the lower the cost of initiation and the larger the profit. As a result, the company's key focus for success is reducing client attrition and implementing an effective retention strategy.

Statement

This dataset contains information about customers that factors unique and proprietary predictions of how long a user will remain a customer. Every user is assigned a prediction value that estimates their state of churn at any given time. This value is based on:

1) User demographic information

2) Browsing behavior

3) Historical purchase data among other information

Our data mining goal is to analyze the data using multiple data mining tools and apply different classification algorithms to accurately classify whether a customer is at risk of churning or not. Our goal is to find an algorithm which can give us a good percentage of true positives and minimize the number of false negatives.

Dataset Description

The dataset used for this project will be acquired from Kaggle. The dataset uses a .csv format and holds several features that can be used to show churning customers. There are a total of 24 attributes including the class attribute “churn risk score” and 36992 tuples.

Column Name	Description
age	Represents the age of a customer
gender	Represents the gender of a customer
security_no	Represents a unique security number that is used to identify a person
region_no	Represents the region that a customer belongs to
membership category	Represents the category of the membership that a customer is using
joining date	Represents the date when a customer became a member
joined through referral	Represents whether a customer joined using any referral code or ID
referral id	Represents a referral ID
preferred offer types	Represents the type of offer that a customer prefers
medium of operation	Represents the medium of operation that a customer uses for transactions
internet option	Represents the type of internet service a customer uses
last visit time	Represents the last time a customer visited the website
days since last login	Represents the no. of days since a customer last logged into the website
avg time spent	Represents the average time spent by a customer on the website
avg transaction value	Represents the average transaction value of a customer
avg frequency login days	Represents the no. of times a customer has logged in to the website
points in wallet	Represents the points awarded to a customer on each transaction
used special discount	Represents whether a customer uses special discounts offered
offer application preference	Represents whether a customer prefers offers
past complaint	Represents whether a customer has raised any complaints
complaint status	Represents whether the complaints raised by a customer was resolved
feedback	Represents the feedback provided by a customer
churn risk score	Represents the churn risk score that 0 or 1

Table 1: Description of dataset

Column Name	Type	Min	1st Qu	Median	Mean	3rd Qu	Max
age	int64	10	23	37	37	51	64
gender	object	-	-	-	-	-	-
security_no	object	-	-	-	-	-	-
region_no	object	-	-	-	-	-	-
membership category	object	-	-	-	-	-	-
joining date	object	-	-	-	-	-	-
joined through referral	object	-	-	-	-	-	-
referral id	object	-	-	-	-	-	-
preferred offer types	object	-	-	-	-	-	-
medium of operation	object	-	-	-	-	-	-
internet option	object	-	-	-	-	-	-
last visit time	object	-	-	-	-	-	-
days since last login	int64	-999	8	12	-42	16	26
avg time spent	float64	-2814	60	162	243	357	3236
avg transaction value	float64	800	14178	27554	29271	40855	99914
avg frequency login days	object	-	-	-	-	-	-
points in wallet	float64	-761	616	698	687	764	2069
used special discount	object	-	-	-	-	-	-
offer application preference	object	-	-	-	-	-	-
past complaint	object	-	-	-	-	-	-
complaint status	object	-	-	-	-	-	-
feedback	object	-	-	-	-	-	-
churn risk score	int64	0	0	1	0.5	1	1

Table 2: Summary of the dataset representing column type, followed by the six number summary if applicable

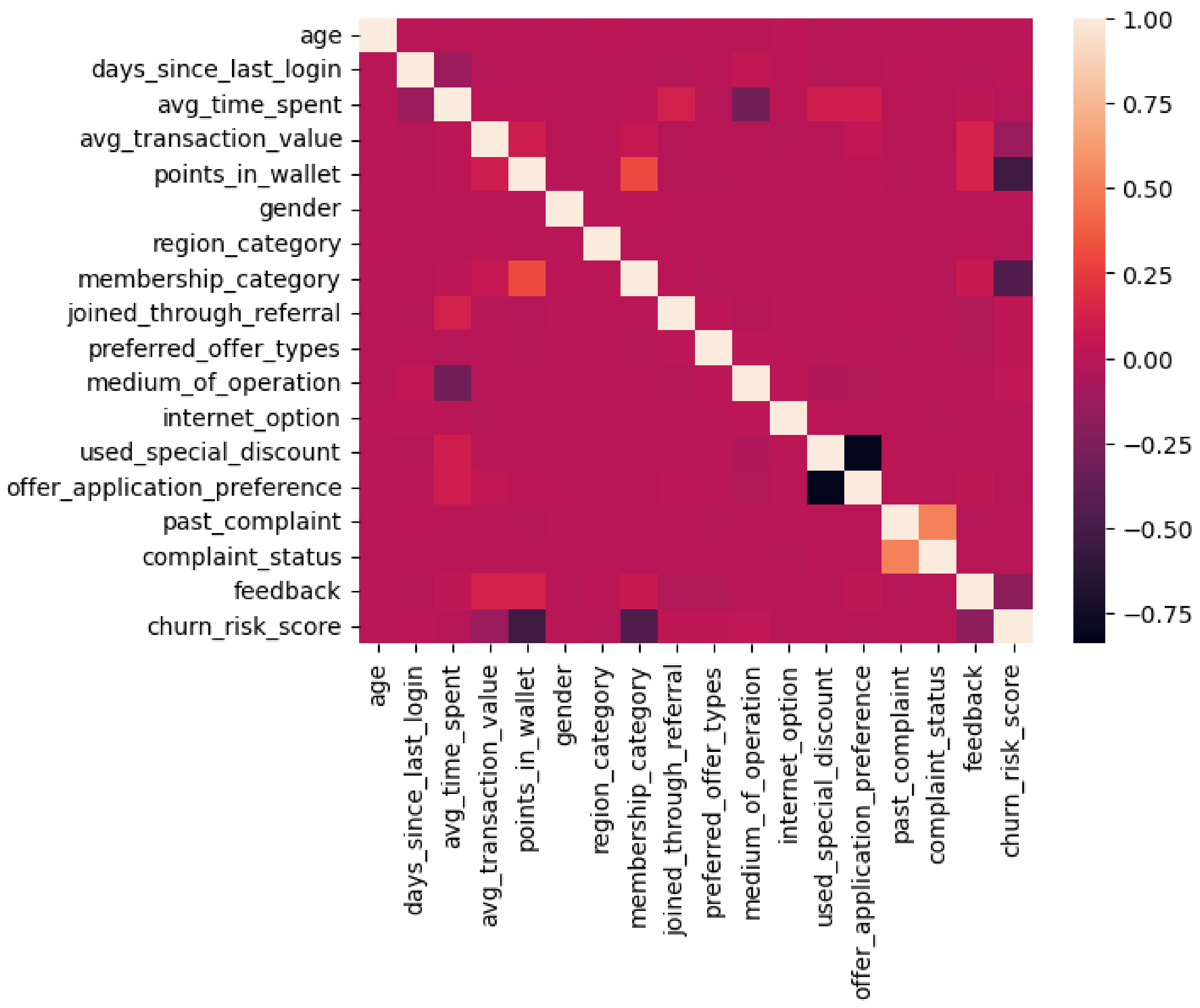


Fig 1: Heatmap representing correlation between each column in the dataset

We can observe that the correlation between the variables is not high.

Since the variables are positively correlated, this indicates they are independent from each other, hence, we can use decision trees, random trees and random forests as our classifiers

1. Introduction

Customer churn prediction is regarded as detecting which customers are likely to leave the company's product or services based on certain experiences or features. It is a critical prediction because for any company, acquiring new customers is more costly as compared to retaining the present ones. Time and resources can instead be utilized to analyze data and identify customers who are at a risk of churning. Once identified, preventive measures can be opted to maximize the chances that the customer will not leave. For this task, the dataset labelled Customer Churn was taken from Kaggle. Data mining tools used to process, analyze, and classify customers into groups labelled (1,0) were Python and Weka. If the classification algorithm predicted an outcome of 1, that would categorize the customer as safe, and an outcome of 0 would categorize the customer at risk of churning. We will not discuss in detail the aspects used to implement the project. Consisting of:

1) Data Mining Tools
2) Classification Algorithms
3) Attribute Selection Methods

1.1 Data Mining Tools

Data mining tools are used to turn raw data into useful information. Using software's to look for patterns in large datasets, analysts can develop effective strategies regarding the task at hand. For this task, the tools used were Python and Weka.

1.1.1 Python

Python is a high-level general purpose programming language. It is not specialized for any specific problem and can be used for a wide variety of functions and tasks. Due to its easy interpretation, English-like syntax, and its vast library support, it allows users to perform almost any possible task. NumPy, Pandas, Matplotlib, Scikit-Learn, etc., offer support to perform data manipulation, data preparation, data visualization, statistical analysis, classification and regression, and much more.

While there are several platforms to use Python on, Jupyter notebooks was used to perform Python related tasks in this project. It offers easy implementation of Python and complies all aspects of the data project into one place.

1.1.2 Weka

Weka supports several standard data mining tasks, more specifically, data preprocessing, clustering, classification, regression, visualization, and feature selection. Input to Weka is expected to be formatted according to the Attribute-Relational File Format and with the filename bearing the .arff extension. All of Weka's techniques are predicated on the assumption that the data is available as one flat file or relation, where each data point is described by a fixed number of attributes (normally, numeric, or nominal attributes, but some other attribute types are also supported). Weka provides access to SQL databases using Java Database Connectivity and can process the result returned by a database query. Weka provides access to deep learning with Deeplearning4j. It is not capable of multi-relational data mining, but there is separate software for converting a collection of linked database tables into a single table that is suitable for processing using Weka.[5] Another key area that is currently not covered by the algorithms included in the Weka distribution is sequence modeling.

1.2 Classification Algorithms

Machine learning algorithms are broadly categorized as regression and classification algorithms. In regression models, the machine learning algorithms are trained to predict continuous values and in classification models, the algorithms are trained to predict the output class. The classification algorithm is a supervised learning technique, it learns from the given dataset and predicts the class of a new observation based on that information. The algorithm that implements classification is called the classifier, and for this project, we applied five different classifiers to our dataset. This allows to better understand the classification style that can optimize results. The five classification algorithms are as follows.

1.2.1 Logistic Regression

Logistic Regression is a statistical model that uses a logistic function to model a binary dependent variable along with many more complex extensions. In logistic regression classification, the output can take only discrete values for a given set of inputs. Setting a threshold value is an important aspect of logistic regression and is dependent on the classification problem itself. The threshold value is mainly affected by the values of precision and recall. In applications where we want to reduce the number of false negatives without necessarily reducing the number of false positives, we choose a threshold value with low precision and high recall. In applications where we want to reduce the number of false positives without necessarily reducing number of false negatives, we choose threshold value with high precision and low recall. Logistic regression can be classified as binomial, multinomial and ordinal.

1.2.2 Naïve Bayes

Naive Bayes classifier, based on Bayes theorem, falls in the category of eager learning classifiers. Eager Learners develop a classification model based on a training dataset before receiving a test dataset. Eager Learner takes more time in learning, and less time in prediction. There are three types of Naive Bayes classifiers, Multinomial, Bernoulli, and Gaussian, and they all work of finding the probability of an outcome happening, given the input features. While they are fast and easy to implement, they all work on the assumption that all input features are independent of each other, which in real-life is not always the case.

1.2.3 Random Tree

The Random Trees node can be used with data in a distributed environment. In this node, you build an ensemble model that consists of multiple decision trees. The Random Trees node is a tree-based classification and prediction method that is built on Classification and Regression Tree method. As with CR Tree, this prediction method uses recursive partitioning to split the training records into segments with similar output field values. The node starts by examining the input fields available to it to find the best split, which is measured by the reduction in an impurity index that results from the split. The split defines two subgroups, each of which is then split into two more subgroups, and so on, until one of the stopping criteria is triggered. All splits are binary (only two subgroups). The Random Trees node uses bootstrap sampling with replacement to generate sample data. The sample data is used to grow a tree model. During tree growth, Random Trees will not sample the data again. Instead, it randomly selects part of the predictors and uses the best one to split a tree node. This process is repeated when splitting each tree node.

1.2.4 Random Forest

Random forest is a machine learning algorithm which combines the output of multiple decision trees to get a single output. It handles both classification and regression problems. It is easy to use and flexible. Random forest builds multiple decision trees using bagging method and merges them together to get single, accurate and stable prediction. The bagging method is a combination of learning models to increase the overall result. The goal of random forest is to reduce variance by averaging multiple decision trees, trained on various parts of the same training data set. Random forest gives output like K-fold cross validation.

1.2.5 J48

J48 is like a flow chart in which the internal node represents a test on an attribute. Each branch represents an outcome of a test, and each leaf node represents a class label. Paths from root to leaf represent classification rules. One of the key features of the decision tree classifier is the capability of capturing descriptive decision-making knowledge from given data. The first node of the decision tree is called a root node. We add a node to the tree every time a question is asked. The result of questions splits the dataset based on the value of a feature and creates new nodes. If there are no further questions or we decide to stop the process after a split, the last nodes are created, and they are called leaf.

1.3 Attribute Selection Model

1.3.1 Classifier Subset Evaluator

Classifier subset evaluator. Uses a classifier to estimate the 'merit' of a set of attributes. Evaluates attribute subsets on training data. The subsets which have less inter-correlation but highly correlated to the target class are preferred

1.3.2 Correlation Attribute Evaluation

CA evaluates the attributes with respect to the target class. Pearson's correlation method is used to measure the correlation between each attribute and target class attribute. It considers nominal attributes in value basis and each value acts as an indicator.

1.3.3 Wrapper Subset Evaluation

Evaluates attribute sets by using a learning scheme. Cross validation is used to estimate the accuracy of the learning scheme for a set of attributes. In wrapper methods, the feature selection process is based on a specific machine learning algorithm that we are trying to fit on a given dataset. It follows a greedy search approach by evaluating all the possible combinations of features against the evaluation criterion. In our project wrapper method is based on Random Tree algorithm.

1.3.4 Classifier Eval Attribute

The last two attribute selection methods are both Classifier Eval Attribute, but we have used them with ZaroR and Random Forest. This selection method Evaluates the worth of an attribute by using a user-specified classifier.

2. Methodology

This section describes the process developed for estimating churn score, shown in Fig. 2.1. We first give a brief description of the datasets used and it changed for our experiment. Then, we explain the pre-processing implemented in the experiment, followed by a detailed description of how the classification algorithms applied to predict outcomes.

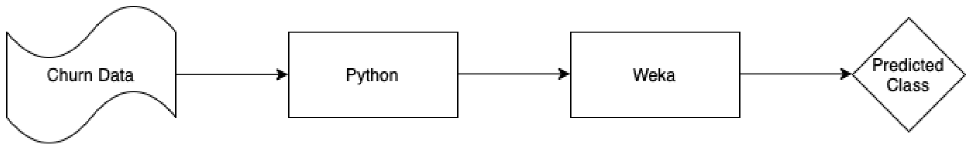


Figure 2.1: Summarized process flow of the project

2.1 Data Preparation

Python's vast library supports easy implementation of data manipulation techniques. For this project, we mainly used Pythons built-in libraries Pandas and NumPy. Using these, we performed the required data manipulation tasks.

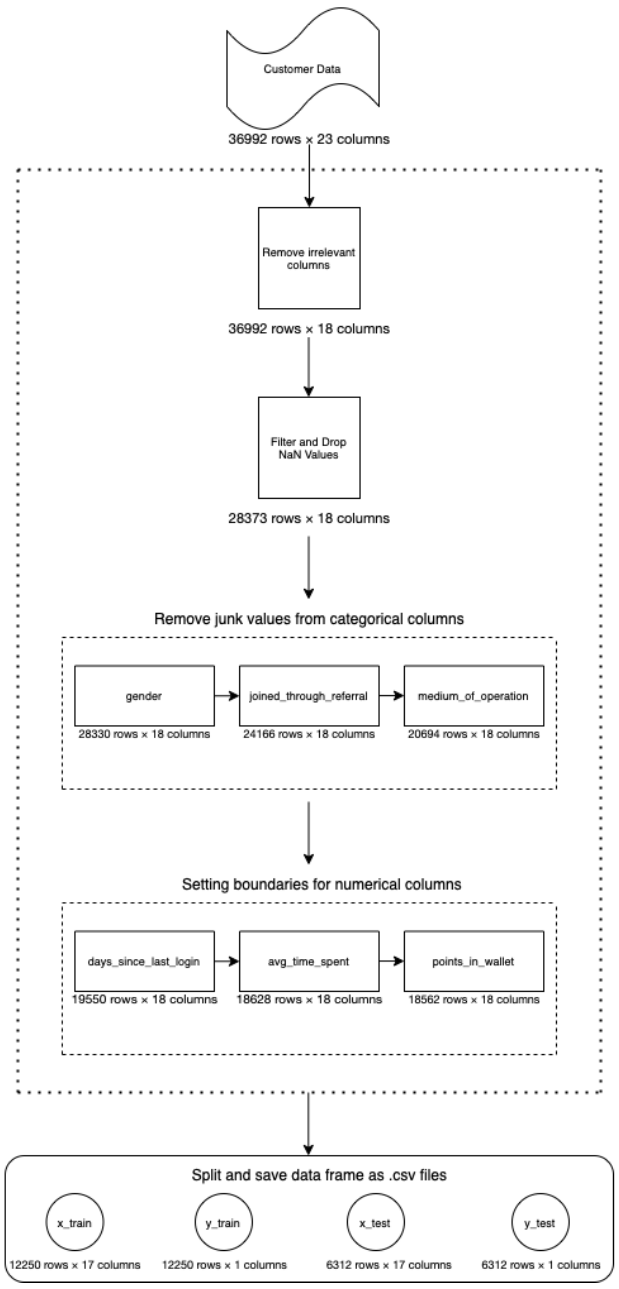
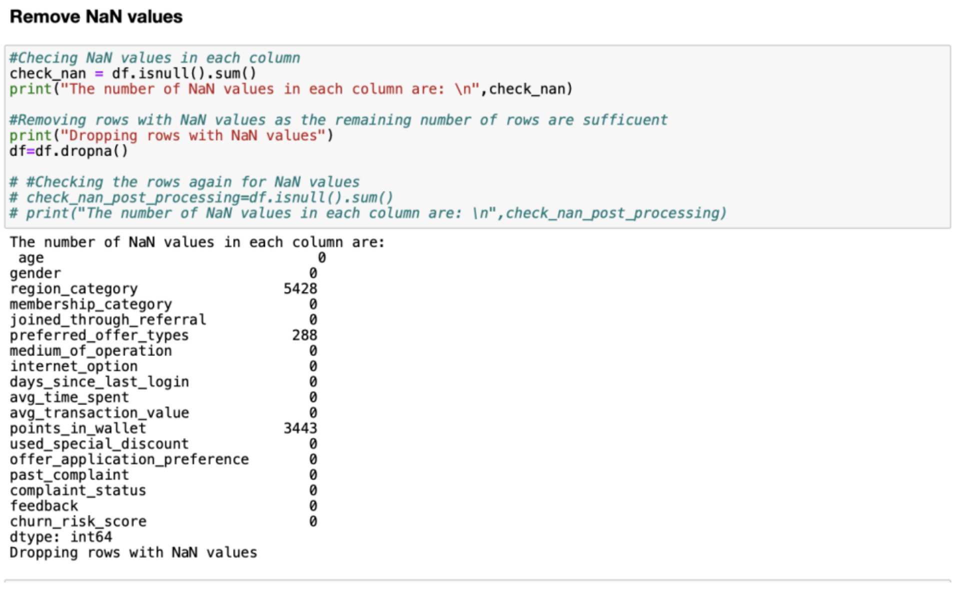


Fig 2.2: Process flow showing the steps taken in the python part of the project

As shown in Fig.2.2. We first import the csv file using Pandas built in function *pd.read_csv()* which gives us our raw rata with a shape of 36992 rows × 23 columns. Then we filter out the relevant column by manually observing the dataset. We conclude that we do not need columns representing user ids, so we remove ***security_no*** and ***referral_id,*** moving on, we remove ***joining_date*** as it is not needed in our analysis, and finally we remove ***last_visit_time*** and ***avg_frequency_login_days*** since we can conduct our analysis using ***days_since_last_login****.* After this, we search our data for NaN variables, using *df.isna().sum()*, which gives us the total NaN values in each column as shown in Fig 2.3 Since our dataset has a large number of tuples, and ***region_category*** and ***points_in_wallet*** are important aspects of our analysis, dropping the rows with NaN values. In order to do this, we use the *df.dropna(axis=0)* function to remove rows with NaN values, resulting in a DataFrame with the shape 28373 rows × 23 columns.



Remove NaN values

```
#Checing NaN values in each column
check_nan = df.isnull().sum()
print("The number of NaN values in each column are: \n",check_nan)

#Removing rows with NaN values as the remaining number of rows are sufficent
print("Dropping rows with NaN values")
df=df.dropna()

# #Checking the rows again for NaN values
# check_nan_post_processing=df.isnull().sum()
# print("The number of NaN values in each column are: \n",check_nan_post_processing)
```

```
The number of NaN values in each column are:
 age                              0
gender                           0
region_category               5428
membership_category              0
joined_through_referral          0
preferred_offer_types          288
medium_of_operation              0
internet_option                  0
days_since_last_login            0
avg_time_spent                   0
avg_transaction_value            0
points_in_wallet              3443
used_special_discount            0
offer_application_preference     0
past_complaint                   0
complaint_status                 0
feedback                         0
churn_risk_score                 0
dtype: int64
Dropping rows with NaN values
```

Fig 2.3: The total NaN values in each column

Moving on, we checked the columns for junk values and found unknown/error values to be present in some of the categorical values. To make our machine learning input as strong as possible, data needs to be of the highest standard. We could have kept these values but due to the dataset length being quite big, we dropped these rows. After cleaning the ***gender***, ***joined_through_referral***, and ***medium_of_operation***, as shown in Fig 2.4, we got a DataFrame of the dimensions 19550 rows and 18 columns.

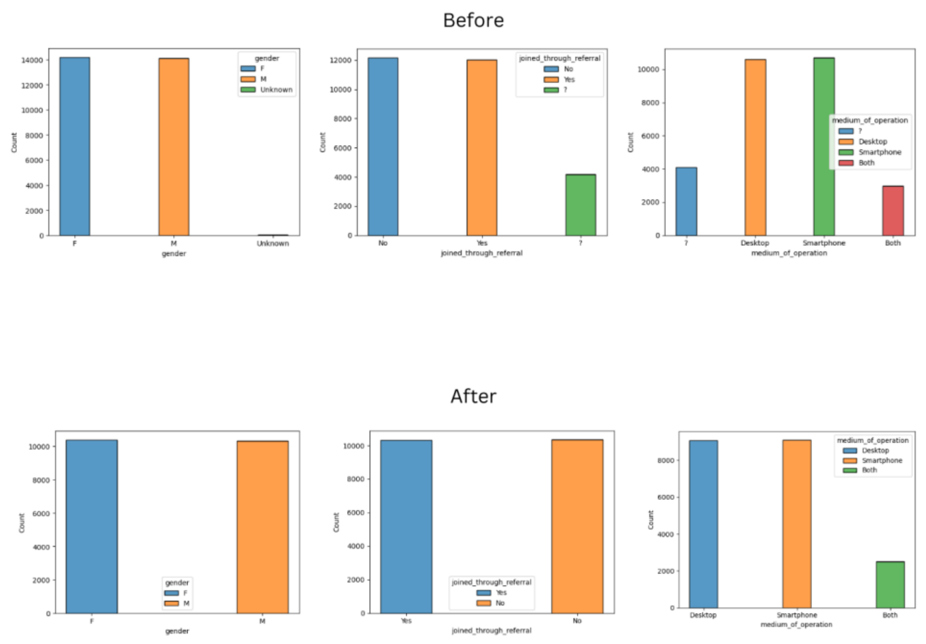


Fig 2.4: The total variables and their types in categorical columns before and after the removal of error values

Similarly, we checked the numerical variables for any discrepancies and found negative values to be present in the dataset, since the existence of negative values in our numerical columns in against logic, we further dropped the rows in ***avg_time_spent***, ***days_since_last_login***, and ***points_in_wallet*** that had negative values. The results in the form of boxplots are shown below in Fig 2.5

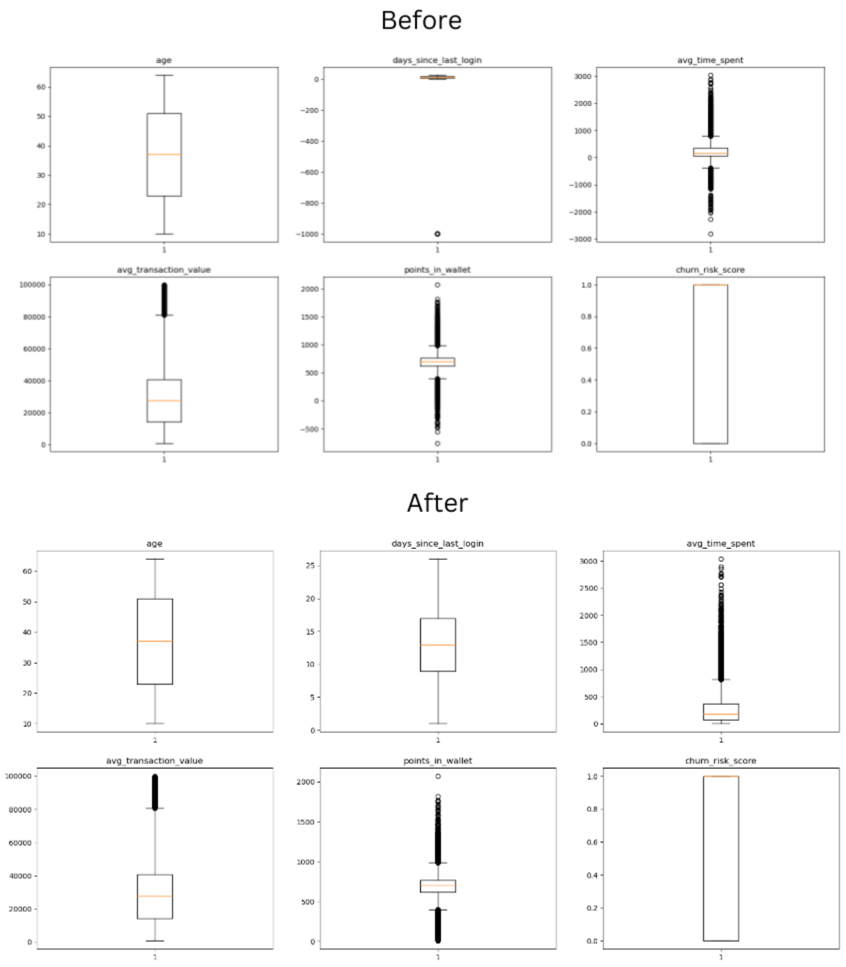


Fig 2.5: Boxplots showing the five number summary before and after the removal of negative values from the column

After the pre-processing is done, we check to see if the dataset is balanced. To make sure our classification models are not biased towards a specific label, we need to have a balanced class input. Meaning, the number of output classes in the DataFrame should be close to each other. For this purpose, we used the function *df['churn risk score'].value counts()*, and got the results as shown in Fig 2.6, through which we concluded that the dataset is fairly balanced, and we can move forward towards the next stage of the project, so we split our DataFrame into training and testing files using Scikit-Learn library, save them as .csv files.

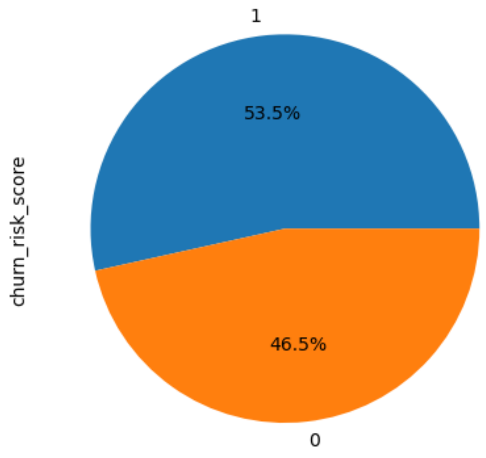
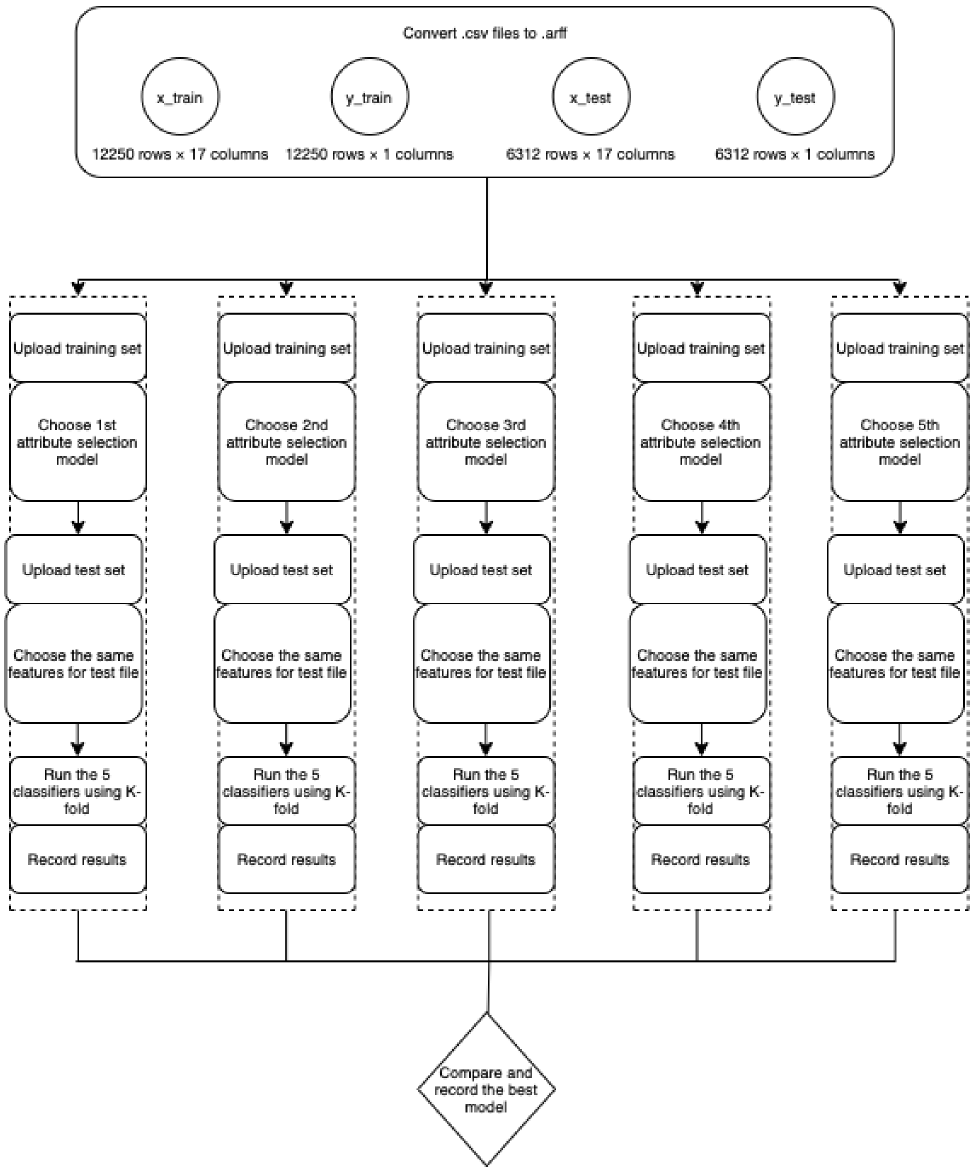


Fig 2.6: Pie chart visualizing the percentage of each target variable in the dataset

2.2 Classification Models

We ran the models with 5 random splits into the train and test sets and each of the splits resulted in a distinct set of attributes, this was followed by tailoring the test-set to follow the same set of attributes in each case, and finally the training and testing sets were run on each of the 5 classification models. As shown in Fig 2.7, this process for repeated for each feature attribute and the results were manually noted and stored into a table.



Convert .csv files to .arff
x_train
y_train
x_test
y_test
12250 rows × 17 columns
12250 rows × 1 columns
6312 rows × 17 columns
6312 rows × 1 columns
Upload training set
Choose 1st attribute selection model
Choose 2nd attribute selection model
Choose 3rd attribute selection model
Choose 4th attribute selection model
Choose 5th attribute selection model
Upload test set
Choose the same features for test file
Run the 5 classifiers using K-fold
Record results
Compare and record the best model

Fig 2.7: Process flow for Weka part of the project

As each feature selection attribute follows a distinct pattern, the attributes mentioned below are the ones that were selected when the splitting algorithm was run.

Classifier	Features Selected
Attribute Evaluator = Classifier eval attribute Attribute Selection Method = Ranker	feedback, internet_option, preferred_offer_types, joined_through_referral, membership_category, region_category, gender, medium_of_operation, days_since_last_login, complaint_status, avg_time_spent, past_complaint, offer_application_preference

Attribute Evaluator = Correlation Attribute Eval Attribute Selection Method: Ranker	membership_category, points_in_wallet, avg_transaction_value, feedback, joined_through_referral, preferred_offer_types, offer_application_preference, days_since_last_login, medium_of_operation, region_category, past_complaint, complaint_status, avg_time_spent, used_special_discount, age
Attribute Evaluator = Classifier_subset_eval Attribute Selection Method = Greedy stepwise	age, gender, region_category, membership_category, joined_through_referral, preferred_offer_types, medium_of_operation, internet_option, days_since_last_login, avg_time_spent, avg_transaction_value, points_in_wallet, used_special_discount, offer_application_preference, past_complaint, feedback
Attribute Evaluator = Classifier_Eval_attribute using Random_forest(RF) Attribute Selection Method = Ranker	membership_category, points_in_wallet, feedback, avg_transaction_value, internet_option, preferred_offer_types, offer_application_preference, joined_through_referral, medium_of_operation, complaint_status, used_special_discount, region_category, past_complaint, gender, days_since_last_login
Attribute Evaluator = Wrapper_Subset_Eval using Random_Tree(RT) Attribute Selection Method = Best_first	membership_category, points_in_wallet, offer_application_preference, past_complaint, complaint_status, feedback

Table 2.1: This table represents the columns selected by each feature attribute

2.3 Project Flow

This section shows the complete project flow, as shown in Fig 2.8, and outlines each step taken to get the results. As mentioned before, the project was divided into two parts, and an intermediate step was taken to convert .csv files to .arff using an online software.

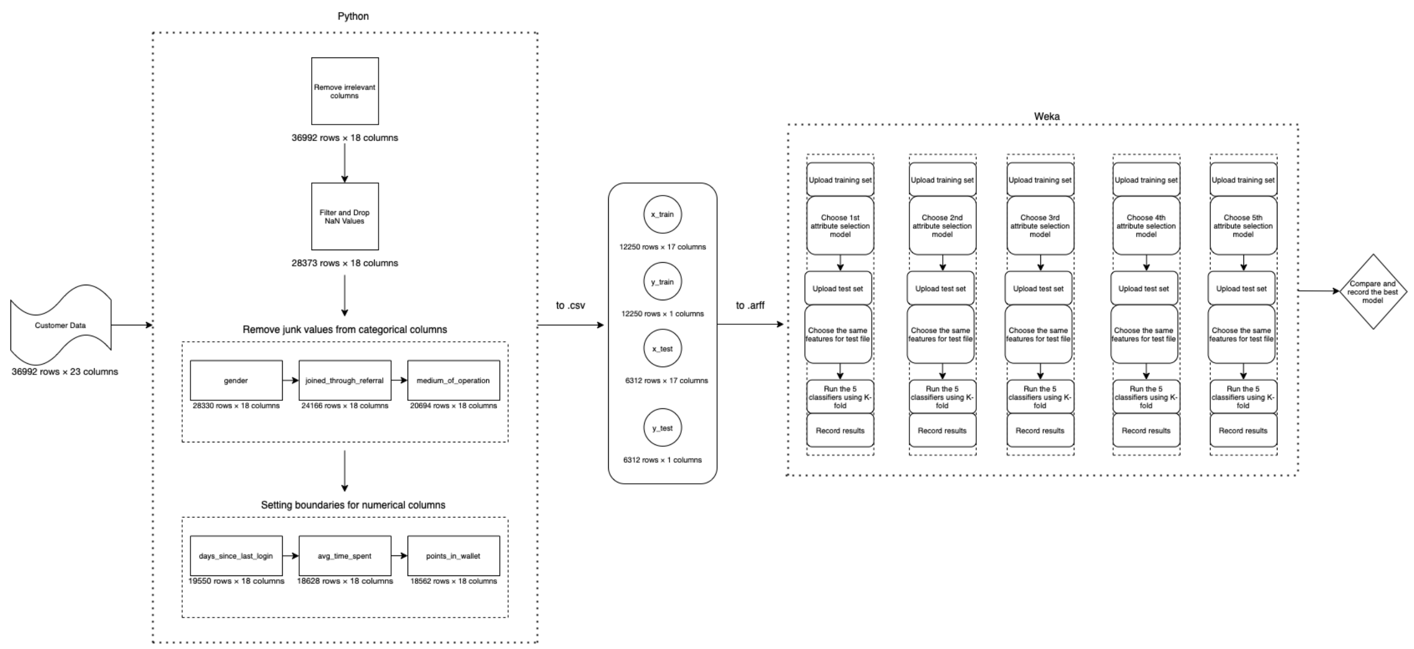


Fig 2.8: Complete process flow of the project highlighting all the steps taken in each part of the project

3. Data Mining Results and Evaluation

In this section, we will discuss how each of our feature selection models performed on the chosen classification models. After running these attributes on the selected classification models, using K-fold validation, we saw the confusion matrix and recorded the performance measure of each classification on each feature selection model. The results are summarized below.

3.1 Logistic Regression

Our Logistic Regression model performed well on each attribute, giving a median overall accuracy and TPR of 90%. One thing to note is that, for each attribute the logistic regression model performed better on the *class=0*

Feature Selection	**Class**	Accuracy (Overall)	TPR	FPR	Precision	Recall	F1 Score	MCC	ROC
Classifier eval attribute	churn_risk_score = 0	0.851236	0.893	0.185	0.808	0.893	0.848	0.706	0.941
	churn_risk_score = 1		0.815	0.107	0.897	0.815	0.854	0.706	0.941
	Weighted		0.851	0.143	0.856	0.851	0.851	0.706	0.941
Correlation Attribute Eval	churn_risk_score = 0	0.904468	0.914	0.104	0.884	0.914	0.899	0.809	0.960
	churn_risk_score = 1		0.896	0.086	0.923	0.896	0.909	0.809	0.960
	Weighted		0.904	0.094	0.905	0.904	0.905	0.809	0.960
Classifier_sub set_eval	churn_risk_score = 0	0.905894	0.914	0.101	0.887	0.914	0.900	0.812	0.960
	churn_risk_score = 1		0.899	0.086	0.923	0.899	0.911	0.812	0.960
	Weighted		0.906	0.093	0.906	0.906	0.906	0.812	0.960
Classifier_Eval _attribute(RF)	churn_risk_score = 0	0.905735	0.916	0.103	0.885	0.916	0.900	0.811	0.960
	churn_risk_score = 1		0.897	0.084	0.925	0.897	0.911	0.811	0.960
	Weighted		0.906	0.093	0.906	0.906	0.906	0.811	0.960
Wrapper_Sub set_Eval(RT)	churn_risk_score = 0	0.909537	0.924	0.103	0.886	0.924	0.905	0.819	0.962

	churn_risk_score = 1		0.897	0.076	0.931	0.897	0.914	0.819	0.962
	Weighted		0.910	0.089	0.910	0.910	0.910	0.819	0.962

Table 3.1: Performance measure of each feature attribute with Logistic Regression

		churn_risk_score = 0	churn_risk_score = 1
Classifier eval attribute	churn_risk_score = 0	2619	315
	churn_risk_score = 1	624	2754
		churn_risk_score = 0	churn_risk_score = 1
Correlation Attribute Eval	churn_risk_score = 0	2682	252
	churn_risk_score = 1	351	3027
		churn_risk_score = 0	churn_risk_score = 1
Classifier_subset_eval	churn_risk_score = 0	2681	253
	churn_risk_score = 1	341	3037
		churn_risk_score = 0	churn_risk_score = 1
Classifier_Eval_attribute(RF)	churn_risk_score = 0	2688	246
	churn_risk_score = 1	349	3029
		churn_risk_score = 0	churn_risk_score = 1
Wrapper_Subset_Eval(RT)	churn_risk_score = 0	2711	223
	churn_risk_score = 1	348	3030

Table 3.2: Confusion matrix of each feature attribute with Logistic Regression

3.2 Naïve Bayes

While our Naïve Bayes model gave a highest ROC score of 0.96 and a median 90+ TPR for *class=1*, it did not perform well for most feature selection attributes on *class=0*, as a result, bringing the overall accuracy of each feature selection down.

Feature Selection	**Class**	Accuracy	TPR	FPR	Precision	Recall	F1 Score	MCC	ROC
Classifier eval attribute	churn_risk_score = 0	0.848384	0.818	0.125	0.850	0.818	0.834	0.695	0.942
	churn_risk_score = 1		0.875	0.182	0.847	0.875	0.861	0.695	0.942
	Weighted		0.848	0.156	0.848	0.848	0.848	0.695	0.942

Correlation Attribute Eval	churn_risk_score = 0	0.866603	0.792	0.068	0.910	0.792	0.847	0.735	0.953
	churn_risk_score = 1		0.932	0.208	0.837	0.932	0.882	0.735	0.953
	Weighted		0.867	0.143	0.871	0.867	0.866	0.735	0.953
Classifier_subset_eval	churn_risk_score = 0	0.866286	0.792	0.069	0.909	0.792	0.846	0.734	0.953
	churn_risk_score = 1		0.931	0.208	0.838	0.931	0.882	0.734	0.953
	Weighted		0.866	0.143	0.871	0.866	0.865	0.734	0.953
Classifier_Eval_attribute(RF)	churn_risk_score = 0	0.867237	0.793	0.068	0.910	0.793	0.847	0.736	0.953
	churn_risk_score = 1		0.932	0.207	0.838	0.932	0.883	0.736	0.953
	Weighted		0.867	0.143	0.872	0.867	0.866	0.736	0.953
Wrapper_Subset_Eval(RT)	churn_risk_score = 0	0.908904	0.919	0.100	0.889	0.919	0.904	0.818	0.961
	churn_risk_score = 1		0.900	0.081	0.927	0.900	0.914	0.818	0.961
	Weighted		0.909	0.090	0.910	0.909	0.909	0.818	0.961

Table 3.13 Performance measure of each feature attribute with Naive Bayes

		churn_risk_score = 0	churn_risk_score = 1
Classifier eval attribute	churn_risk_score = 0	2399	535
	churn_risk_score = 1	422	2956
		churn_risk_score = 0	churn_risk_score = 1
Correlation Attribute Eval	churn_risk_score = 0	2323	611
	churn_risk_score = 1	231	3147
		churn_risk_score = 0	churn_risk_score = 1
Classifier_subset_eval	churn_risk_score = 0	2324	610
	churn_risk_score = 1	234	3144
		churn_risk_score = 0	churn_risk_score = 1
Classifier_Eval_attribute(RF)	churn_risk_score = 0	2326	608
	churn_risk_score = 1	230	3148
		churn_risk_score = 0	churn_risk_score = 1
Wrapper_Subset_Eval(RT)	churn_risk_score = 0	2696	238
	churn_risk_score = 1	337	3041

Table 3.4: Confusion matrix of each feature attribute with Naïve Bayes

3.3 J48

The J48 model gave a median TPR of 92% for *class=0* and 95% for class=1. This gives a median accuracy of roughly 94% and a highest ROC score of 0.96.

Feature Selection	**Class**	Accuracy	TPR	FPR	Precision	Recall	F1 Score	MCC	ROC
Classifier eval attribute	churn_risk_score = 0	0.843314	0.816	0.133	0.842	0.816	0.829	0.685	0.895
	churn_risk_score = 1		0.867	0.184	0.845	0.867	0.856	0.685	0.895
	Weighted		0.843	0.160	0.843	0.843	0.843	0.685	0.895
Correlation Attribute Eval	churn_risk_score = 0	0.941857	0.926	0.044	0.948	0.926	0.937	0.883	0.954
	churn_risk_score = 1		0.956	0.074	0.937	0.956	0.946	0.883	0.954
	Weighted		0.942	0.060	0.942	0.942	0.942	0.883	0.954
Classifier_subset_eval	churn_risk_score = 0	0.93853	0.916	0.042	0.950	0.916	0.933	0.877	0.945
	churn_risk_score = 1		0.958	0.084	0.929	0.958	0.943	0.877	0.945
	Weighted		0.939	0.064	0.939	0.939	0.938	0.877	0.945
Classifier_Eval_attribute(RF)	churn_risk_score = 0	0.941223	0.923	0.043	0.949	0.923	0.936	0.882	0.960
	churn_risk_score = 1		0.957	0.077	0.935	0.957	0.946	0.882	0.960
	Weighted		0.941	0.061	0.941	0.941	0.941	0.882	0.960
Wrapper_Subset_Eval(RT)	churn_risk_score = 0	0.940114	0.920	0.042	0.950	0.920	0.935	0.880	0.963
	churn_risk_score = 1		0.958	0.080	0.932	0.958	0.945	0.880	0.963
	Weighted		0.940	0.063	0.940	0.940	0.940	0.880	0.963

Table 3.5: Performance measure of each feature attribute with J48

		churn_risk_score = 0	churn_risk_score = 1
Classifier eval attribute	churn_risk_score = 0	2395	539
	churn_risk_score = 1	450	2928
		churn_risk_score = 0	churn_risk_score = 1
Correlation Attribute Eval	churn_risk_score = 0	2717	217
	churn_risk_score = 1	150	3228
		churn_risk_score = 0	churn_risk_score = 1
Classifier_subset_eval	churn_risk_score = 0	2688	246

	churn_risk_score = 1	142	3236
		churn_risk_score = 0	churn_risk_score = 1
Classifier_Eval_attribute(RF)	churn_risk_score = 0	2709	225
	churn_risk_score = 1	146	3232
		churn_risk_score = 0	churn_risk_score = 1
Wrapper_Subset_Eval(RT)	churn_risk_score = 0	2698	236
	churn_risk_score = 1	142	3236

Table 3.6: Confusion matrix of each feature attribute with J48

3.4 Random Forest

For random forest we get an extremely high ROC score of 0.974 with a maximum accuracy of 0.935 for the feature attribute classifier_subset_eval, giving out a weighted TPR of 0.937

Feature Selection	Class	Accuracy	TPR	FPR	Precision	Recall	F1 Score	MCC	ROC
Classifier eval attribute	churn_risk_score = 0	0.842997	0.822	0.139	0.837	0.822	0.830	0.684	0.941
	churn_risk_score = 1		0.861	0.178	0.848	0.861	0.854	0.684	0.941
	Weighted		0.843	0.160	0.843	0.843	0.843	0.684	0.941
Correlation Attribute Eval	churn_risk_score = 0	0.939005	0.926	0.049	0.942	0.926	0.934	0.877	0.974
	churn_risk_score = 1		0.951	0.074	0.936	0.951	0.943	0.877	0.974
	Weighted		0.939	0.063	0.939	0.939	0.939	0.877	0.974
Classifier_subset_eval	churn_risk_score = 0	0.936946	0.922	0.050	0.941	0.922	0.931	0.873	0.974
	churn_risk_score = 1		0.950	0.078	0.934	0.950	0.942	0.873	0.974
	Weighted		0.937	0.065	0.937	0.937	0.937	0.873	0.974
Classifier_Eval_attribute(RF)	churn_risk_score = 0	0.935995	0.923	0.052	0.939	0.923	0.931	0.871	0.974
	churn_risk_score = 1		0.948	0.077	0.934	0.948	0.941	0.871	0.974
	Weighted		0.936	0.066	0.936	0.936	0.936	0.871	0.974
Wrapper_Subset_Eval(RT)	churn_risk_score = 0	0.923796	0.913	0.067	0.922	0.913	0.918	0.847	0.971
	churn_risk_score = 1		0.933	0.087	0.925	0.933	0.929	0.847	0.971
	Weighted		0.924	0.078	0.924	0.924	0.924	0.847	0.971

Table 3.7: Performance measure of each feature attribute with Random Forest

		churn_risk_score = 0	churn_risk_score = 1
Classifier eval attribute	churn_risk_score = 0	2412	522
	churn_risk_score = 1	469	2909
		churn_risk_score = 0	churn_risk_score = 1
Correlation Attribute Eval	churn_risk_score = 0	2716	218
	churn_risk_score = 1	167	3211
		churn_risk_score = 0	churn_risk_score = 1
Classifier_subset_eval	churn_risk_score = 0	2706	228
	churn_risk_score = 1	170	3208
		churn_risk_score = 0	churn_risk_score = 1
Classifier_Eval_attribute(RF)	churn_risk_score = 0	2707	227
	churn_risk_score = 1	177	3201
		churn_risk_score = 0	churn_risk_score = 1
Wrapper_Subset_Eval(RT)	churn_risk_score = 0	2679	255
	churn_risk_score = 1	226	3152

Table 3.8: Confusion matrix of each feature attribute with Random Forest

3.5 Random Tree

The highest ROC score and accuracy we got was 0.923 and 0.922. While this is categorized as a good result, compared to our previous models, we can immediately identify that this is not the best performing model.

Feature Selection	**Class**	Accuracy	TPR	FPR	Precision	Recall	F1 Score	MCC	ROC
Classifier eval attribute	churn_risk_score = 0	0.810995	0.792	0.172	0.800	0.792	0.796	0.620	0.821
	churn_risk_score = 1		0.828	0.208	0.821	0.828	0.824	0.620	0.821
	Weighted		0.811	0.192	0.811	0.811	0.811	0.620	0.821
Correlation Attribute Eval	churn_risk_score = 0	0.918093	0.914	0.079	0.910	0.914	0.912	0.835	0.920
	churn_risk_score = 1		0.921	0.086	0.925	0.921	0.923	0.835	0.920

	Weighted		0.918	0.082	0.918	0.918	0.918	0.835	0.920
Classifier_subset_eval	churn_risk_score = 0	0.91128	0.905	0.083	0.904	0.905	0.905	0.822	0.913
	churn_risk_score = 1		0.917	0.095	0.917	0.917	0.917	0.822	0.913
	Weighted		0.911	0.090	0.911	0.911	0.911	0.822	0.913
Classifier_Eval_attribute(RF)	churn_risk_score = 0	0.908587	0.899	0.083	0.904	0.899	0.901	0.816	0.911
	churn_risk_score = 1		0.917	0.101	0.913	0.917	0.915	0.816	0.911
	Weighted		0.909	0.093	0.909	0.909	0.909	0.816	0.911
Wrapper_Subset_Eval(RT)	churn_risk_score = 0	0.922529	0.915	0.071	0.918	0.915	0.917	0.844	0.923
	churn_risk_score = 1		0.929	0.085	0.926	0.929	0.928	0.844	0.923
	Weighted		0.923	0.078	0.923	0.923	0.923	0.844	0.923

Table 3.9: Performance measure of each feature attribute with Random Tree

		churn_risk_score = 0	churn_risk_score = 1
Classifier eval attribute	churn_risk_score = 0	2323	611
	churn_risk_score = 1	582	2796
		churn_risk_score = 0	churn_risk_score = 1
Correlation Attribute Eval	churn_risk_score = 0	2683	251
	churn_risk_score = 1	266	3112
		churn_risk_score = 0	churn_risk_score = 1
Classifier_subset_eval	churn_risk_score = 0	2655	279
	churn_risk_score = 1	281	3097
		churn_risk_score = 0	churn_risk_score = 1
Classifier_Eval_attribute(RF)	churn_risk_score = 0	2638	296
	churn_risk_score = 1	281	3097
		churn_risk_score = 0	churn_risk_score = 1
Wrapper_Subset_Eval(RT)	churn_risk_score = 0	2684	250
	churn_risk_score = 1	239	3139

Table 3.10: Confusion matrix of each feature attribute with Random Tree

3.6 Best Model from the 25 Classifications

Classifier	Feature Selection	Class	Accuracy	TPR	FPR	Precision	Recall	F1 Score	MCC	ROC
Logistic Regression	Wrapper_Subset _Eval(RT)	churn_risk _score = 0	0.909537	0.924	0.103	0.886	0.924	0.905	0.819	0.962
		churn_risk _score = 1		0.897	0.076	0.931	0.897	0.914	0.819	0.962
		Weighted		0.910	0.089	0.910	0.910	0.910	0.819	0.962
Gaussian Naïve Bayesian	Wrapper_Subset _Eval(RT)	churn_risk _score = 0	0.908904	0.919	0.100	0.889	0.919	0.904	0.818	0.961
		churn_risk _score = 1		0.900	0.081	0.927	0.900	0.914	0.818	0.961
		Weighted		0.909	0.090	0.910	0.909	0.909	0.818	0.961
J48	Correlation Attribute Eval	churn_risk _score = 0	0.941857	0.926	0.044	0.948	0.926	0.937	0.883	0.954
		churn_risk _score = 1		0.956	0.074	0.937	0.956	0.946	0.883	0.954
		Weighted		0.942	0.060	0.942	0.942	0.942	0.883	0.954
Random Forest	Correlation Attribute Eval	churn_risk _score = 0	0.939005	0.926	0.049	0.942	0.926	0.934	0.877	0.974
		churn_risk _score = 1		0.951	0.074	0.936	0.951	0.943	0.877	0.974
		Weighted		0.939	0.063	0.939	0.939	0.939	0.877	0.974
Random Tree	Wrapper_Subset _Eval(RT)	churn_risk _score = 0	0.922529	0.915	0.071	0.918	0.915	0.917	0.844	0.923
		churn_risk _score = 1		0.929	0.085	0.926	0.929	0.928	0.844	0.923
		Weighted		0.923	0.078	0.923	0.923	0.923	0.844	0.923

Table 3.11: Table showing the best performing features in each model

We first look at the best model for each machine learning classifier from the feature selected data sets. The following models were chosen:

The above models were chosen because they have a positive MCC score and the ROC area of each one of these models is greater than 0.5. Having an MCC score of greater than 0 means that the predictions are better than random.

Now, we will select the best model from these 5 models. We have selected the J48 model with correlation attribute eval. The reason for this being that this model has the best ROC area under curve and the best MCC, it also has the best accuracy among the models. At the same time, it has the highest TPR rate and lowest miscalculation rate.

3.7 10-Fold Validation Best Model

Classification	**Class**	Accuracy	TPR	FPR	Precision	Recall	F1 Score	MCC	ROC
Logistic Regression	churn_risk_score = 0	0.901196	0.918	0.114	0.876	0.918	0.896	0.803	0.960
	churn_risk_score = 1		0.886	0.082	0.926	0.886	0.906	0.803	0.960
	Weighted		0.901	0.097	0.902	0.901	0.901	0.803	0.960
Gaussian Naïve Bayesian	churn_risk_score = 0	0.867363	0.794	0.069	0.909	0.794	0.848	0.736	0.953
	churn_risk_score = 1		0.931	0.206	0.839	0.931	0.882	0.736	0.953
	Weighted		0.867	0.142	0.872	0.867	0.866	0.736	0.953
J48	churn_risk_score = 0	0.94311	0.922	0.038	0.955	0.922	0.938	0.886	0.954
	churn_risk_score = 1		0.962	0.078	0.934	0.962	0.948	0.886	0.954
	Weighted		0.943	0.06	0.943	0.943	0.943	0.886	0.954
Random Forest	churn_risk_score = 0	0.941009	0.931	0.051	0.941	0.931	0.936	0.881	0.976
	churn_risk_score = 1		0.949	0.069	0.941	0.949	0.945	0.881	0.976
	Weighted		0.941	0.06	0.941	0.941	0.941	0.881	0.976
Random Tree	churn_risk_score = 0	0.903243	0.897	0.092	0.895	0.897	0.896	0.806	0.905
	churn_risk_score = 1		0.908	0.103	0.91	0.908	0.909	0.806	0.905
	Weighted		0.903	0.098	0.903	0.903	0.903	0.806	0.905

Fig 3.12: This table represents the best model after performing 10-fold validation

The above results have been obtained by applying the 5 classification algorithms on preprocessed data before it was split into training and test dataset.

We choose J48 to be the best model amongst the five, because it has the best average MCC, TPR and accuracy along with one of the lowest FPR as well.

Best Overall Model

After comparing the result of all 30 models, we can concluded that J48 classification algorithm performed on the 10-fold validation has the best performance in terms of high MCC, ROC and TPF and low FPR values.

Conclusion

We saw that different data set balancing techniques can give different results and can affect the metrics as well. In this project, we followed the overall process of data mining from data preparation to machine learning. We learnt about different feature selection methods and different classification algorithms. We saw that there is no one algorithm which always works the best. We need to experiment with different algorithms and then calculate the metrics to decide which one is the best for the data set. In this project, we only used 5 algorithms to test the data set. But, to select the best algorithm, we need to experiment with more classifiers and other data set balancing techniques to get an evaluation of the best model to implement for the data set.

To conclude, we made use of a customer churn dataset from Kaggle and applied machine learning classification algorithms and feature selection methods to predict which model and attribute will correlate with each other to give the best possible prediction for the risk of a customer churning. After applying logistic regression, naïve bayes, J48, random forest, and random tree, we got a reasonable accuracy score ranging from 81% to 95%. Of all the models, J48 with correlation attribute eval feature gave us the best results in terms of accuracy, MCC score, ROC score, and weighted TPR. The next step of the project will consist of deploying this project as a machine learning framework in an actual business scenario and evaluating how well the business does in keeping customers.

Contribution

Faadil, Shaikh – Data preparation, correlation feature selection, Wrapper based feature selection, Subset based feature selection, J48, Random Forest and Logistic Classifier.

Taimoor- Made Python Notebook, Data Preparation, Data Visualization, Eval attribute feature selection, Random Tree, and Naïve Bayes Classifier, Report formulation.

References

https://builtin.com/data-science/random-forest-algorithm

https://builtin.com/data-science/random-forest-algorithm

https://en.wikipedia.org/wiki/Random_forest

https://www.geeksforgeeks.org/understanding-logistic-regression/

https://blog.hubspot.com/service/what-is-customer-churn

https://towardsdatascience.com/naive-bayes-classifier-81d512f50a7c

https://www.qualtrics.com/experience-management/customer/customer-churn/

https://www.geeksforgeeks.org/naive-bayes-classifiers/

https://towardsdatascience.com/naive-bayes-classifier-81d512f50a7c

https://en.wikipedia.org/wiki/Python_(programming_language)

https://www.coursera.org/articles/what-is-python-used-for-a-beginners-guide-to-using-python