

Comp 462 Project Report Customer Churn Prediction

Prepared by:

ALI KEMAL KILIC

ORCUN YASAR

ALI BEDIRHAN AKSOY

Date: 18/05/2025

Problem Understanding & Motivation

Problem Definition

Customer churn is the termination of a customer relationship with a company, that is, the termination of the company's products or services. This situation appears when customers cancel their subscriptions or move to another provider. Customer churn is a significant problem for companies because acquiring a new customer is usually more costly than retaining an existing customer.

The main goal of this project is to develop and compare machine learning models that predict which customers are most likely to leave a company using customer data from a fictional telecommunications company provided by IBM. The reason we chose this problem is that we thought it would be an easier kick-start for us since other projects at some point required image processing, which is beyond our skills for now.

Importance of the Problem and Real World Context

The telecom sector is quite competitive right now. The viability of these businesses depends on maintaining client loyalty and reducing customer attrition because switching service providers is a fairly easy decision for any customer. On the contrary, high churn rates can also result in lost revenue and damage to the telecom's reputation. Because of this, considering they have enough resources to create solutions, being able to predict customer churn means everything for these big companies. For example, they can identify customers likely to leave early and offer them promotions, discounts, or services to keep them from churning. Alternatively, services can be improved just by analysing the causes of churn.

Motivation and Objectives of the Project

The main motivation of this project is to demonstrate how machine learning techniques can be applied to customer churn and to compare the performance of different models using specific metrics. Our objectives are, firstly, to analyse and understand the provided IBM Telco customer data. Secondly, implementing data preprocessing and feature engineering steps. After that, choose the most important features and modify the dataset accordingly. Following that, training and evaluating at least two different machine learning models learned in the course, in our case, are Random Forest and XGBoost, to predict customer churn. Later, comparing the performance of the models using metrics of accuracy, precision, recall, F1 score and Matthews's correlation coefficient (MCC). Lastly, interpreting the results and drawing conclusions about which model or approach is more successful.

Dataset Definition and Exploration

In this section, the IBM Telco Customer Churn dataset used in the project will be introduced in detail, and the basic exploratory data analysis (EDA) findings will be presented.

Dataset Source and Overview

The "Telco Customer Churn" dataset published by IBM is used in this project. The dataset contains information about a fictional telecommunications company that provides home phone and internet services to 7043 customers in California. It includes information such as which services the customers subscribe to, their demographic information, subscription duration, payment methods, and most importantly, whether they have left the company (churn status). The dataset initially contains 33 different features (variables) for each customer.

Our main target variable is the Churn Label (Yes/No) or its numerical equivalent, Churn Value (1/0) columns, which indicate whether the customer has left the company. The `churn_value` is used as the target variable in our code.

Features

Our dataset contains information in several categories that can impact customer churn:

Customer Identifier: CustomerID

Demographics: Gender, SeniorCitizen, Partner, Dependents

Account Information: Tenure Months, Contract, PaperlessBilling, PaymentMethod, MonthlyCharges, TotalCharges.

Service Subscriptions: PhoneService, MultipleLines, InternetService, OnlineSecurity, OnlineBackup, DeviceProtection, TechSupport, StreamingTV, StreamingMovies.

Target Variables: Churn Label, Churn Value.

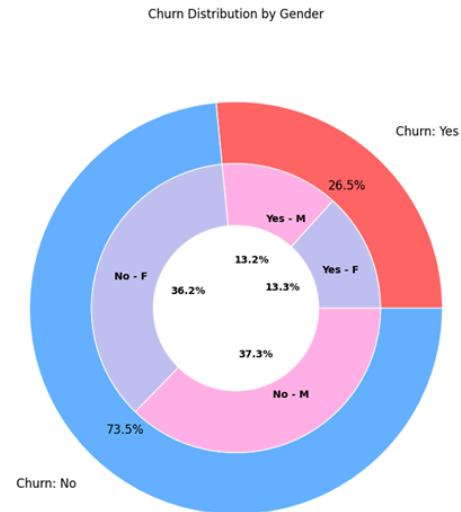
Other Information: (not directly used or omitted in the code) Count, Country, State, City, Zip Code, Lat Long, Latitude, Longitude, Churn Score, CLTV, Churn Reason

Exploratory Data Analysis (EDA)

Beforehand, we made some visualisations to reveal likely potential relationships between features and churn that we expected.

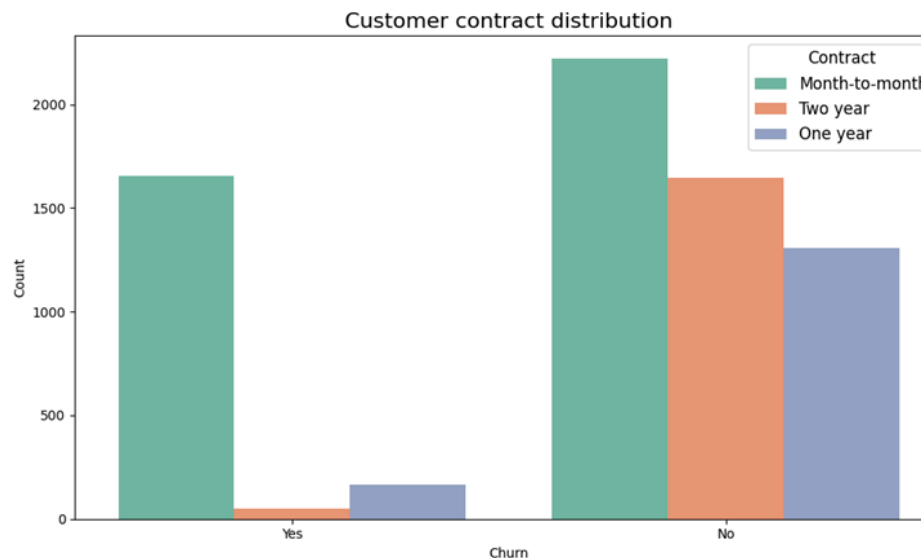
Churn Distribution by Gender:

The nested pie chart in the code shows the overall churn and the churn trend by gender. These percentages show that the gender distributions within the leaving and staying customer groups are very close. Therefore, we earned a first impression from this graph that gender is not the sole factor in customer churn.



Churn Distribution by Contract Type:

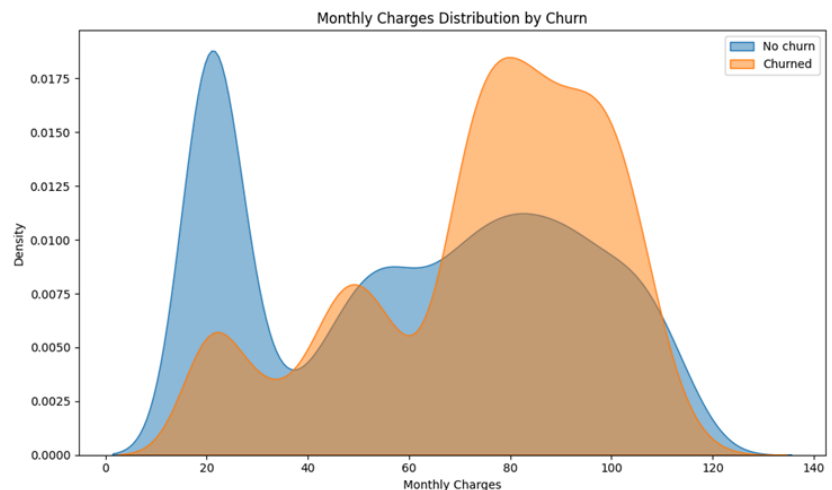
This chart compares churn rates for customers with different contract types (monthly, one-year, two-year). This observation suggests that shorter contracts reduce customer loyalty, and customers with this type of contract are much more likely to leave the company.



Churn Distribution by Monthly Fee:

This density graph compares the monthly fee distributions of customers who left the company and those who stayed.

The graph suggests that customers who pay high monthly fees tend to leave. It should also be noted that there is a large group that stays by paying low fees. This made us assume that the relationship between customer churn and monthly fees is not monotonic and probably interacts with other services (e.g. type of internet service).



Dataset Statistics and Class Imbalance:

We used the `pd.info()` and `pd.describe()` outputs to observe our data. Accordingly, there are 7043 records and 33 columns in our dataset. Most columns are of object (categorical) data type; columns such as Monthly Charges, Latitude, Longitude are of float64 data type, and columns such as Tenure Months, Count, Zip Code, Churn Value, Churn Score, and CLTV are of int64 data type.

When looking at the missing data, except for the Churn Reason column (1869 non-null, i.e. 7043 - 1869 = 5174 missing values), there is no missing data in the other columns at first glance. However, the fact that the Total Charges column is of object type shows us that it may contain non-numeric values (such as blanks), which is addressed in the preprocessing section.

The `pd.describe()` output provides basic statistics for numeric columns. The mean of the Churn Value column is 0.265370. This indicates that approximately 26.5% of customers leave the company (Churn Value = 1), while 73.5% stay with the company (Churn Value = 0). This reveals a class imbalance in our dataset; that is, one class (non-leaving customers) is more than the other class (leaving customers).

Data Preprocessing and Feature Engineering

This section will explain the data cleaning, transformation and feature engineering steps applied to make the raw dataset suitable for machine learning models.

Data Cleaning

Removal of Unnecessary Columns:

Some columns that we thought were of no use to the model performance or could cause data leakage were removed from the dataset by hand.

Reasoning:

CustomerID: A unique identifier for each customer, which does not provide a meaningful pattern for the model.

Zip Code, Lat Long: Geographic information was not used directly for this analysis.

Churn Reason: Contains information about why the customer left. Since this information is obtained after the customer has left, using it in a model that tries to predict the event of departure in advance would lead to data leakage.

Churn Label: We omitted it because it is the text-based equivalent of our target variable `churn_value`, and `churn_value` (0/1) is already present.

Multiple Model Training:

We used the in-built methods of `SelectFromModel` and `RFE` from `sklearn.feature_selection` to further cherry-pick the important features to improve the performance of our two models (the method will be explained in the next sections). Resulting in eliminating columns such as Country, State, City, and Count.

Handling of Missing/Invalid Data Charges:

When examining the Total Charges column, it is seen that some values are not numeric (possibly containing spaces or empty strings), and this situation is converted to NaN (Not a Number) values using the `pd.to_numeric` function and the `errors='coerce'` parameter.

These NaN values are padded with the customer's monthly charge (Monthly Charges) multiplied by the number of months they have been with the company (Tenure Months) (`calc_charges = df['Monthly Charges'] * df['Tenure Months']`).

Data Transformation

Normalisation of Column Names:

The spaces and capital letters in the column names in the data set have been brought to a standard format for consistency and ease of use. The column names were first stripped of leading and trailing spaces (`str.strip()`), then all letters were converted to lowercase (`str.lower()`), and finally the space characters were replaced with underscores (`_`) (`str.replace(' ', '_')`).

Converting Categorical Features:

The columns with object data type in the dataset were converted to a numeric format with `.astype('category')` so that machine learning algorithms can process them. Later, the One-Hot Encoding method was applied using the `pd.get_dummies()` function.

The `drop_first=True` parameter also helped prevent the multicollinearity problem by dropping the first of the dummy variables created for each categorical feature. Nevertheless, the target variable `churn_value` is not affected by this step since it is already in a numeric (0 or 1) format.

Feature Scaling

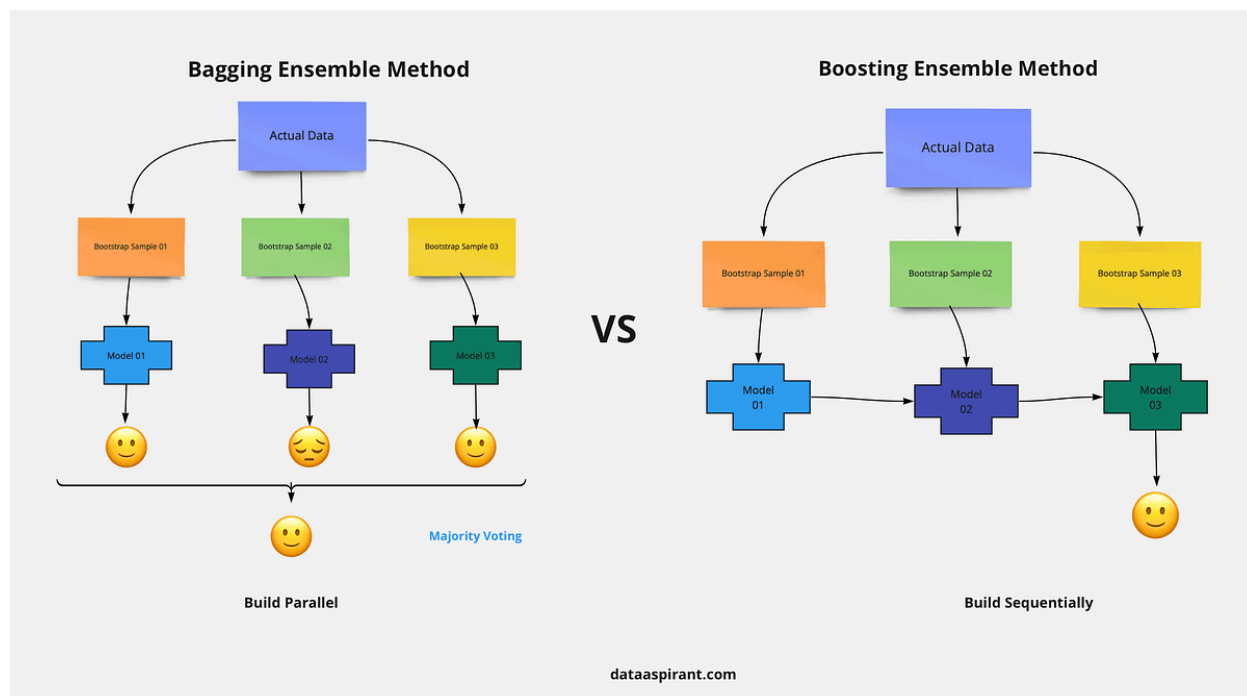
StandardScaler transforms each feature into a mean of 0 and a standard deviation of 1 (Z-score normalisation). This process is learned with `fit_transform` on the training data and applied to the training and test data with `transform`. Although we use tree-based models, which are generally less sensitive to feature scaling, we kept using feature scaling, considering it affected the performance of our models positively.

Model Implementation

In this section, after we have gone through the data preprocessing and feature selection steps, the training processes of the selected classification models for customer churn prediction and the methods used in this process are detailed

The Used Models and Reasonings.

When we started this assignment, our primary goal was to compare two powerful models: Random Forest (bagging) and XGBoost (boosting).



As shown above, Random Forest separates the data into multiple decision trees and reduces diversity by averaging. However, XGBoost builds the trees one after another and makes corrections to optimise the trees in each iteration. While the performance gap between them may vary in some cases, they are generally similar. As a result, we made both models use the same data and compared their performance metrics to see which approach is better.

Model Training

In this part, we will discuss in detail our model training process for our classification models after the data preprocessing and feature selection are completed. As it is stated before, Random Forest and XGBoost algorithms are our main tools. To observe our models' capability limits and how they differ, we implemented both models in two different ways with embedded Feature Importance (FI)-based sub-feature sets and Recursive Feature Elimination (RFE)-determined sub-feature sets, resulting in four model configurations in total.

These are:

- **Random Forest + FI** (Feature Importance)
- **Random Forest + RFE** (Recursive Feature Elimination)
- **XGBoost + FI**
- **XGBoost + RFE**

Hyperparameter Selection

Similar model configurations have identical shared hyperparameter settings to ensure a fair comparison between them.

We observed and tested different parameters and concluded on these parameters since they showed the highest performance. For Random Forest, we chose **n_estimators = 200**, **max_depth = 5**, and **random_state = 42**. These choices increase the generalisation and stability of the model, also preventing overfitting, and since we fixed our seed value, it is a plus for reproducibility. For XGBoost, we set **n_estimators = 50**, **max_depth = 3**, **eval_metric = 'logloss'**, and **random_state = 42**. Limiting the number of trees and their depths allowed us to decrease the complexity and training time, moreover, the logloss metric is used since, as we discussed, our dataset contains imbalanced examples of churn_value, our target value.

Feature Selection Integration

Feature Importance (FI): We trained a **RandomForestClassifier** on the entire feature set, calculated the **feature_importances_** value, and set a threshold at the mean plus one standard deviation. Using **SelectFromModel**, we selected features above this threshold and retrained both Random Forest and XGBoost on the reduced set.

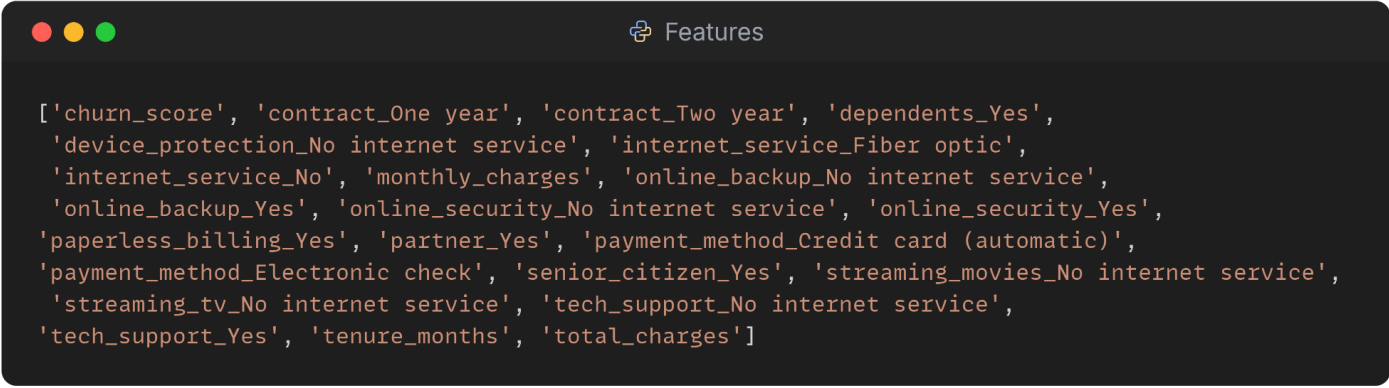
Recursive Feature Elimination (RFE): We used a Logistic Regression estimator to iteratively eliminate the least important features, eventually selecting the top 20. We then retrained both models on this subset.

Training Procedure

The model training workflow consisted of four main steps. First, we split the dataset into training (80%) and test (20%) subsets using stratified sampling to preserve the class distribution. Second, a **StandardScaler** was fitted to the training data and applied to the training and test sets to normalise the feature scales. Third, each of the four model configurations was trained on the selected feature subset via the **fit()** method. Finally, we evaluated the performance by generating predictions with **predict()** on the test set and calculating accuracy, precision, recall, F1 score, and Matthews correlation coefficient (MCC).

Evaluation and Performance Comparison

After successfully preprocessing and feature selection, we obtained a whopping total of 1163 features thanks to the **.dummies** method. Later on, using FI and RFE, we successfully reduced significant features to 23. Thus, helping to eliminate redundancy and noise while retaining predictive variables. The final set of features is:



```
['churn_score', 'contract_One year', 'contract_Two year', 'dependents_Yes',  
 'device_protection_No internet service', 'internet_service_Fiber optic',  
 'internet_service_No', 'monthly_charges', 'online_backup_No internet service',  
 'online_backup_Yes', 'online_security_No internet service', 'online_security_Yes',  
 'paperless_billing_Yes', 'partner_Yes', 'payment_method_Credit card (automatic)',  
 'payment_method_Electronic check', 'senior_citizen_Yes', 'streaming_movies_No internet service',  
 'streaming_tv_No internet service', 'tech_support_No internet service',  
 'tech_support_Yes', 'tenure_months', 'total_charges']
```

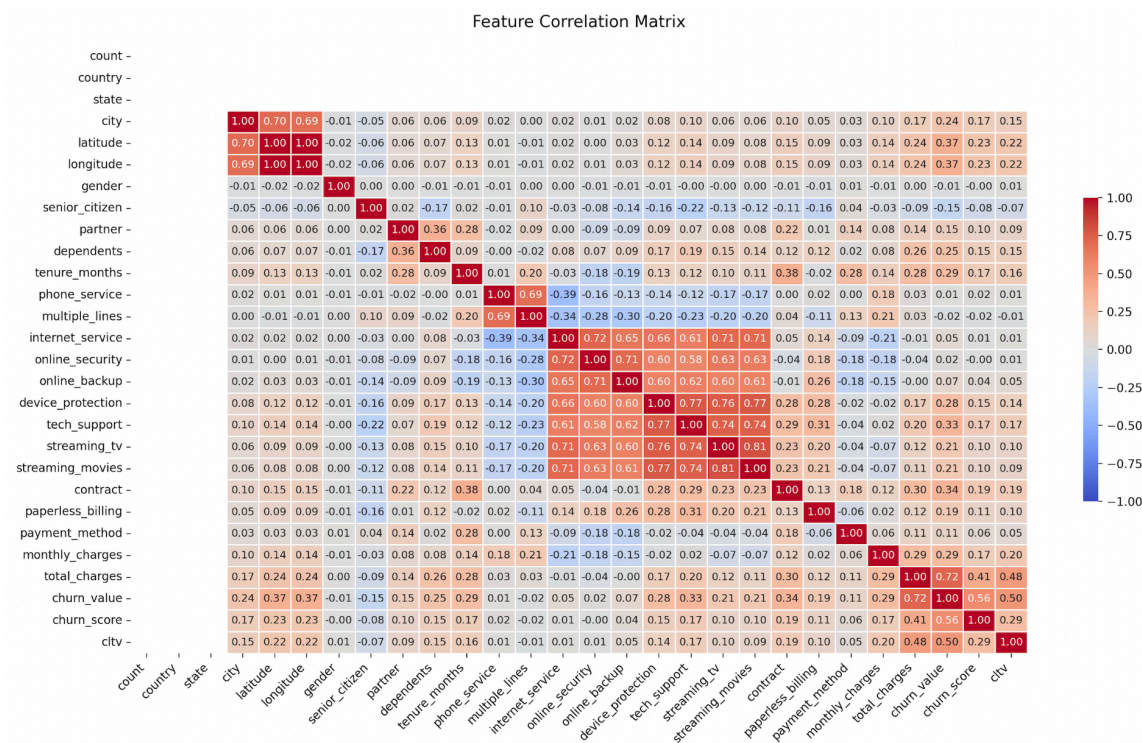
Feature Selection Results and Correlation Analysis

This feature space includes several numerical and categorical variables, which have been variously encoded for model compatibility.

A **correlation matrix** was examined to see the interrelationships among variables (see **Figure 1**) of interest in the matrix are:

- There exist **strong positive correlations** involving churn-related services such as tech_support, online_security, streaming_tv, streaming_movies, online_backup, which means customers subscribing to multiple services tend to stay longer.
- **Negative correlation** exists between tenure_months and churn_score at **-0.28**, which means customers with longer tenure are less likely to churn.
- **A strong correlation of 0.72** exists between churn_value and the total_charges, which means customers with higher accumulated charges may be considered to be at risk for churn.
- contract_Two year and tech_support_Yes stand as moderately positively correlated with customer retention.

These correlations do confirm the said expectations: sustained contract, multiple services and higher tenure act as factors against churn, whereas low tenure and fewer services increase chances of customers churning.



(Figure 1)

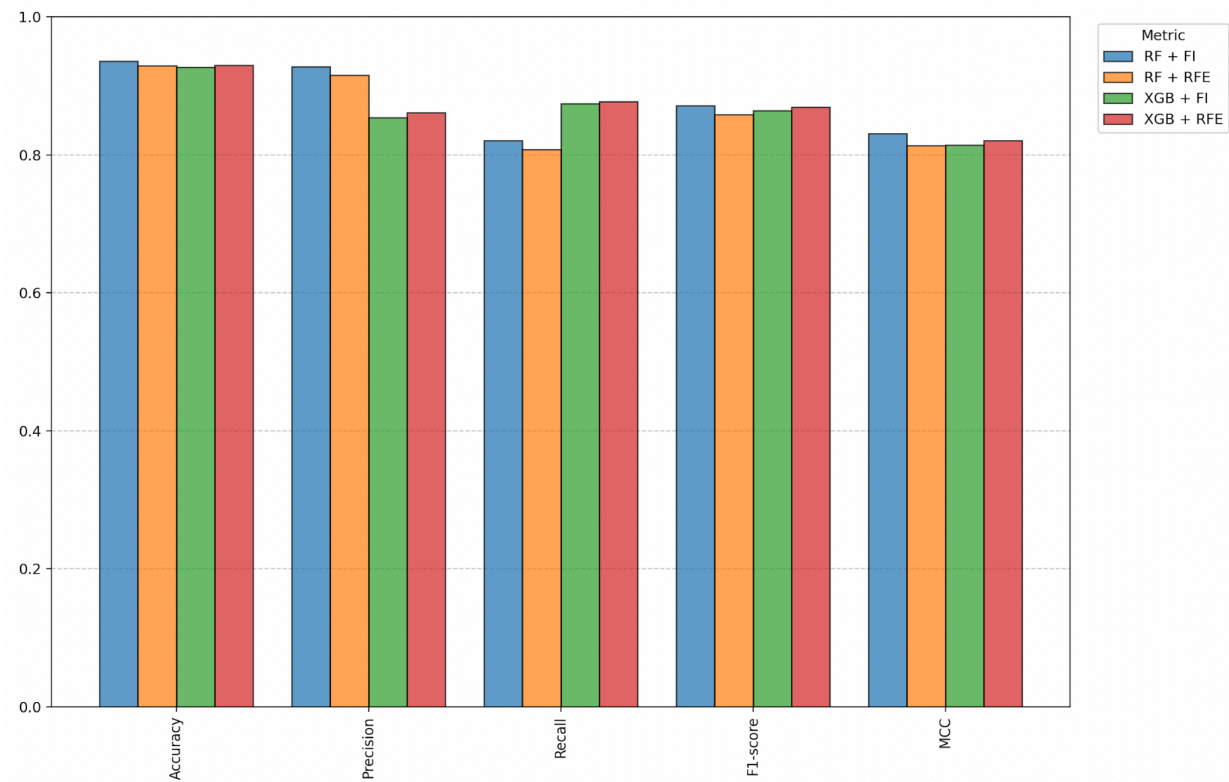
Model Performance Metrics

We experimented with four alternative model configurations by combining two different algorithms (**Random Forest** and **XGBoost**) with two methods of feature selection (**Feature Importance** and **RFE**). The models themselves were then evaluated to assess their performance using **accuracy**, **precision**, **recall**, **F1-score**, and **Matthews Correlation Coefficient (MCC)**.

The summarisation of the performance results is found in **Table 1**:

Model	Accuracy	Precision	Recall	F1-Score	MCC
Random Forest + FI	0.935	0.927	0.821	0.871	0.831
Random Forest + RFE	0.927	0.854	0.874	0.864	0.814
XGBoost + FI	0.929	0.915	0.807	0.858	0.814
XGBoost + RFE	0.930	0.861	0.877	0.869	0.821

(Figure 2)



Confusion Matrices

Additionally, the confusion matrices for each model are shown below:

RandomForest + FI:

```
[[1011 24]
 [ 67 307]]
```

RandomForest + RFE:

```
[[1007 28]
 [ 72 302]]
```

XGBoost + FI:

```
[[979 56]
 [ 47 327]]
```

XGBoost + RFE:

```
[[982 53]
 [ 46 328]]
```

It is clear from the confusion matrices that **XGBoost models outperformed RandomForest** in identifying consumers who are truly at risk of churn, as seen by the fact that they produced fewer false negatives.

Discussion of Results

The four models all performed extremely well, with **accuracy more than 92%** and **MCC greater than 0.81**, demonstrating high matching rates between predicted and actual churn statuses.

RandomForest + Feature Importance achieved the highest **accuracy (93.5%)** and **precision (92.7%)**, showing its strength at minimising false positives.

XGBoost + RFE achieved the highest **recall (87.7%)** and **F1-score (86.9%)**, demonstrating the best capacity to correctly identify customers that would churn.

The balance between **precision** and **recall** is critical in predicting customer churn. For real-life purposes, recall tends to matter more, since predicting a churn-risky customer may result in lost revenue. So, although RandomForest was more precise, **XGBoost + RFE is likely to be the better choice for proactive retention**, despite slightly lower precision.

The **feature correlation matrix (Figure 1)** supported these findings by confirming that such variables as `contract_One year`, `tech_support_Yes`, and `churn_score` play important roles in churn behaviour. Customers with longer contracts and active tech support subscriptions were less likely to churn, while higher churn scores and no tech support were associated with higher churn risk.

Importantly, despite the use of different feature selection methods (FI and RFE), the performance of the models remained comparable, which implies that the predictive power resides in a set of highly important features.

Moreover, decreasing features from 1163 to 23 significantly reduced the model computing load, yet retained as much information as possible.

Conclusion

In this, we developed and trained machine learning models to predict customer churn from the Telco Customer Churn dataset of IBM. We employed **Random Forest** and **XGBoost**, both with **Feature Importance** and **RFE** as feature selection.

The models were accurate in prediction, and key observations were:

- **RandomForest + Feature Importance** was best in **accuracy (93.5%)** and **precision (92.7%)**, and it performed well in avoiding false alarms.
- **XGBoost + RFE** performed the highest **recall (87.7%)** and **F1-score (86.9%)** and was therefore the best performing model for correctly identifying churners.
- All the models performed great **MCC (>0.81)**, demonstrating well-balanced performance even under small class imbalance.
- Feature selection identified **contract type**, **tenure**, **tech support**, and **churn_score** as key predictors, providing actionable details for customer retention.

From a business perspective, **XGBoost + RFE** is the **most suitable model** since it has improved recall, which will enable more customers at risk to be identified for intervention.

Our work can mean something by exploring hyperparameter tuning, cost-sensitive learning, and real-time deployment of churn prediction to provide more business value.

Team Contribution Statement

- The problem definition and brainstorming were done by the **whole team** during a common meeting.

Then, we divided all segments of the project into **3 objectives**: Data Preprocessing, Model Implementation and Optimisation, and Visualisation and Discussion of Results.

- Data Preprocessing was done by **Ali Kemal Kilic**.
- Model Implementation and optimization were done by **Orcun Yasar**.
- Visualisation and Discussion of Results were done by **Ali Bedirhan Aksoy**.