Customer Churn Prediction

1 BUSINESS UNDERSTANDING

The goal of this project is to develop a predictive model to identify customers at risk of churning, enabling companies to adopt effective strategies for retaining existing customers. This is particularly important in the highly competitive telecommunication industry, where customer movement between service providers can significantly impact profitability. Retaining customers is less expensive than acquiring new ones or up-selling, making it a critical focus for businesses aiming to maximize profits and sustain long-term growth.

1.1 Define Business Goals:

Reduce Customer Churn by Identifying High-Risk Customers

Churn prediction identifies customers likely to leave by analyzing patterns like complaints, reduced usage, or pricing dissatisfaction. This helps businesses address root causes, reduce attrition, and retain customers, which is more cost-effective than acquiring new ones.

Enhance Customer Retention by Implementing Targeted Strategies

After identifying high-risk customers, businesses can implement targeted strategies like personalized offers (e.g., discounts or loyalty rewards) and prioritized customer support to address concerns and enhance their experience, boosting retention.

Improve Customer Satisfaction and Loyalty by Addressing Key Dissatisfaction Factors

Enhancing customer satisfaction is key to reducing churn. Issues like poor service, unresponsive support, and unmet expectations can be tackled by improving reliability, resolving issues promptly, ensuring fair pricing, and leveraging customer feedback. Addressing dissatisfaction fosters loyalty, boosts retention, and strengthens market reputation.

1.2 Assess the Current Situation:

Overview of the Telecommunication Industry and Customer Churn

The telecom industry faces increased competition, leading to higher customer churn as dissatisfied customers switch to competitors. To tackle this, businesses must evaluate their retention strategies and address gaps in service quality and customer engagement.

Importance of Early Churn Detection

Many telecom companies lack effective systems to detect early churn signals, leaving rich customer data underutilized. This limits their ability to identify at-risk customers and implement retention strategies, resulting in missed opportunities to improve satisfaction and reduce churn.

Dataset Overview: The dataset (Customer_Churn_cleaned-set 1.csv) contains 10,000 records with 11 attributes, including demographics (age, gender, tenure), financial metrics (credit score, balance, salary), product usage, activity status, and a churn label (1: churned, 0: stayed), capturing key customer behaviors.

1.3 Identify Potential Risks:

Class Imbalance: Churn datasets often have more non-churners, causing biased predictions. To improve performance, techniques like oversampling the minority class, under sampling the majority class, or using class-weighted loss functions are needed.

Data Quality and Preprocessing: Poor data quality, like missing values or irrelevant features, can reduce model accuracy. Preprocessing steps such as data cleaning, normalization, and outlier detection are crucial for improving performance. Feature selection techniques also help reduce dimensionality, enhancing model efficiency and accuracy in churn prediction.

Reactive vs. Proactive Approach Challenges: Accurately predicting churn is challenging, as false positives can lead to unnecessary retention efforts and wasted resources. On the other hand, failing to identify true churners results in lost revenue opportunities. Balancing false positives and false negatives is essential to implement effective retention strategies and minimize resource waste.

2 DATA UNDERSTANING

2.1 Objective:

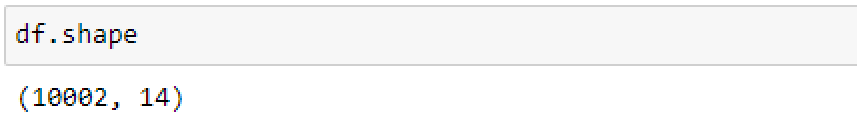
The objective of this phase is to collect and familiarize yourself with the dataset, determine its quality, gain insights, and identify any potential issues that could impact analysis or modeling efforts. This step is crucial for determining the suitability of the data for the modelling phase and ensuring it aligns with the business problem.

Data Collection: The dataset provided (Customer_Churn_cleaned-set 1.csv) contains detailed customer data, including both demographic and financial information. It consists of 10,000 records and 11 attributes, capturing key customer behaviours and characteristics.

Initial Data Analysis: The first few rows of the data have been examined using `df. head ()`. The dataset contains information about customer demographics, account details, and churn behavior, which will be analyzed to predict customer churn.

2.2 Data Description:

Shape: The dataset contains **10,000 rows** and **14 columns**, comprising both numeric and categorical features.



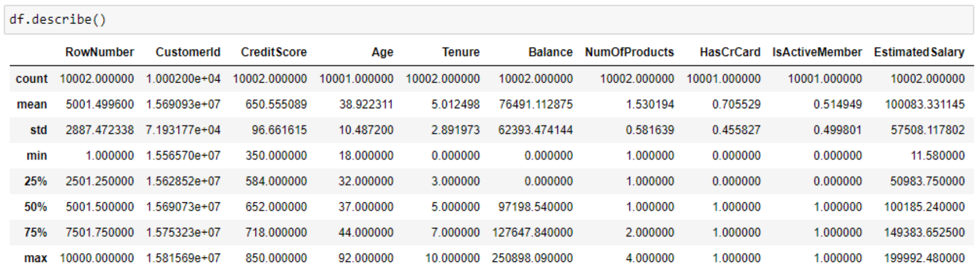
```
df.shape
```

```
(10002, 14)
```

Summary Statistics: A summary of the numerical features was generated using df.describe(), which highlights the following key statistics for the dataset:

- **CreditScore**: Ranges from a minimum of 350 to a maximum of 850, with a mean around 650, indicating varied customer creditworthiness.
- **Age**: Customers' ages range between 18 and 92, with a mean age in the mid-30s to 40s.
- **Balance**: Some customers have a balance of 0, while others have balances exceeding 200,000, reflecting significant variation in account activity.
- **EstimatedSalary**: Salaries range from near 0 to over 200,000, with a mean of approximately 100,000.

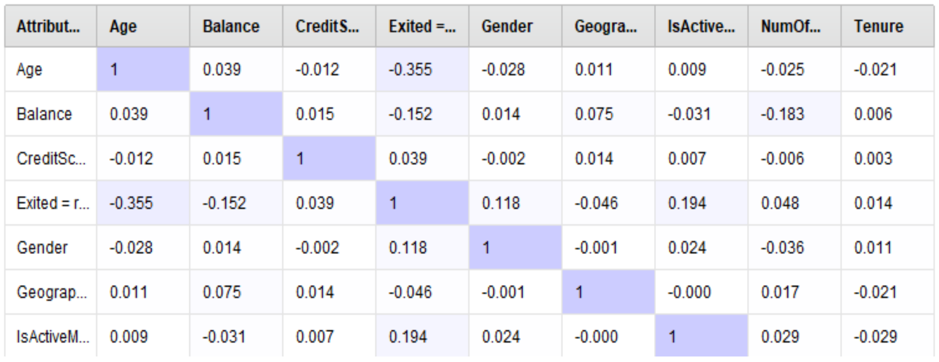
- **Tenure**: Values range from 0 to 10 years, reflecting customer relationships of varying lengths.



```
df.describe()
```

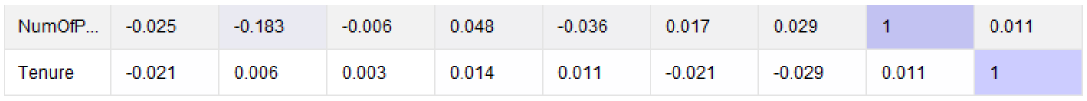
	RowNumber	CustomerId	CreditScore	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary
count	10002.000000	1.000200e+04	10002.000000	10001.000000	10002.000000	10002.000000	10002.000000	10001.000000	10001.000000	10002.000000
mean	5001.499600	1.569093e+07	650.555089	38.922311	5.012498	76491.112875	1.530194	0.705529	0.514949	100083.331145
std	2887.472338	7.193177e+04	96.661615	10.487200	2.891973	62393.474144	0.581639	0.455827	0.499801	57508.117802
min	1.000000	1.556570e+07	350.000000	18.000000	0.000000	0.000000	1.000000	0.000000	0.000000	11.580000
25%	2501.250000	1.562852e+07	584.000000	32.000000	3.000000	0.000000	1.000000	0.000000	0.000000	50983.750000
50%	5001.500000	1.569073e+07	652.000000	37.000000	5.000000	97198.540000	1.000000	1.000000	1.000000	100185.240000
75%	7501.750000	1.575323e+07	718.000000	44.000000	7.000000	127647.840000	2.000000	1.000000	1.000000	149383.652500
max	10000.000000	1.581569e+07	850.000000	92.000000	10.000000	250898.090000	4.000000	1.000000	1.000000	199992.480000

2.3 Correlation results:



Attribut...	Age	Balance	CreditS...	Exited =...	Gender	Geogra...	IsActive...	NumOf...	Tenure
Age	1	0.039	-0.012	-0.355	-0.028	0.011	0.009	-0.025	-0.021
Balance	0.039	1	0.015	-0.152	0.014	0.075	-0.031	-0.183	0.006
CreditSc...	-0.012	0.015	1	0.039	-0.002	0.014	0.007	-0.006	0.003
Exited = r...	-0.355	-0.152	0.039	1	0.118	-0.046	0.194	0.048	0.014
Gender	-0.028	0.014	-0.002	0.118	1	-0.001	0.024	-0.036	0.011
Geograp...	0.011	0.075	0.014	-0.046	-0.001	1	-0.000	0.017	-0.021
IsActiveM...	0.009	-0.031	0.007	0.194	0.024	-0.000	1	0.029	-0.029

2.4 Correlation results:



NumOfP...	-0.025	-0.183	-0.006	0.048	-0.036	0.017	0.029	1	0.011
Tenure	-0.021	0.006	0.003	0.014	0.011	-0.021	-0.029	0.011	1

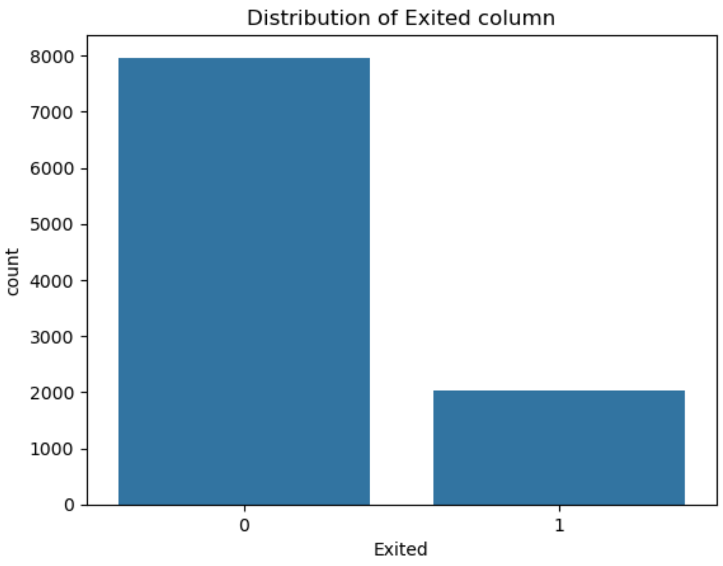
Data Quality Check:

Missing Values: There are missing values in the HasCrCard column as evident from the NaN values in the initial data preview. These missing values will need to be handled during the preprocessing phase.

Data Types: The data types of the columns were checked to distinguish between categorical (e.g., `proto`, `service`, `state`, `attack_cat`) and numerical features (e.g., `dur`, `spkts`, `dpkts`, `rate`, etc.). This classification helps in selecting the appropriate methods for data pre-processing and feature engineering.

Outliers: Numerical features such as CreditScore, Age, Balance, and EstimatedSalary were inspected for outliers using box plots. Observations with extremely high or low values need further investigation to confirm whether they are valid or should be treated.

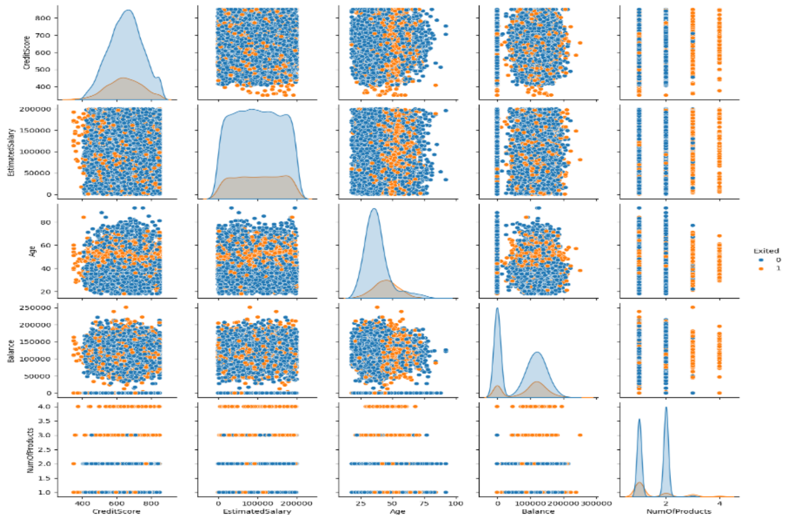
Target Variable Distribution: The target variable *Exited* indicates customer churn (0: no churn, 1: churn). The data is imbalanced, with 7,963 non-churned and 2,037 churned instances, risking bias toward the majority class. Techniques like oversampling, under sampling, or using class weights will address this during model training.



2.5 Exploratory Data Analysis (EDA):

Statistical Summary: The dataset was explored to understand the distribution of key numerical variables like `rate`, `sload`, `dload`, `sloss`, and others.

Pair plot: A pairplot visualizing features like CreditScore, Age, Balance, and NumOfProducts against the target *Exited* reveals patterns and clusters distinguishing churned (Exited = 1) from non-churned (Exited = 0) customers.



.

3 DATA PREPARATION

The Data Preparation phase is essential for building a successful machine learning model. It involves cleaning, transforming, integrating, and selecting relevant features to structure raw data for training. This phase ensures the data is ready for effective model development.

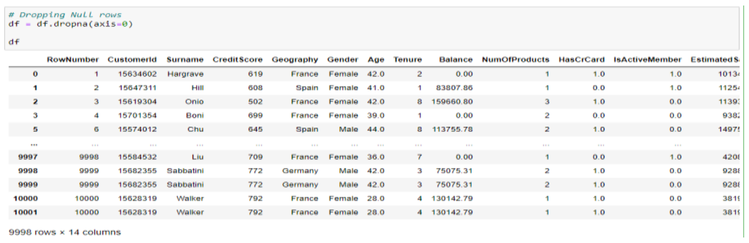
3.1 Objective:

The goal of the Data Preparation phase is to ensure that the dataset is clean, well-structured, and ready for the modelling phase. This involves handling missing values, encoding categorical variables, transforming data into suitable formats, and selecting the most relevant features for the learning task.

3.2 Key Tasks in Data Preparation

1. **Data Cleaning:** Data cleaning is one of the first and most important tasks. Raw datasets are often messy, containing errors, missing values, and inconsistencies. Cleaning ensures that the data is usable by removing or correcting these issues.

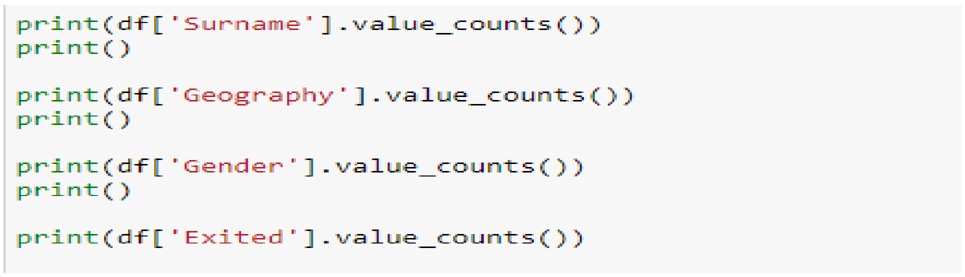
- **Handling Missing Values**: Rows with missing values were removed using *df.dropna(axis=0)* to ensure data integrity and avoid issues during model training. This step improved dataset reliability, preventing biased results, although it slightly reduced the dataset size.



```python
# Dropping Null rows
df = df.dropna(axis=0)

df
```

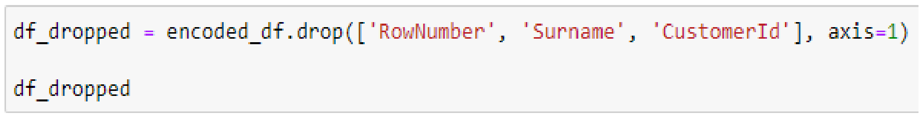
```
       RowNumber  CustomerId  Surname    CreditScore  Geography  Gender  Age   Tenure  Balance    NumOfProducts  HasCrCard  IsActiveMember  EstimatedS
    0          1    15634602  Hargrave           619     France  Female  42.0       2       0.00              1        1.0             1.0        1013
    1          2    15647311      Hill           608      Spain  Female  41.0       1   83807.86              1        0.0             1.0        1125
    2          3    15619304      Onio           502     France  Female  42.0       8  159660.80              3        1.0             0.0        1139
    3          4    15701354      Boni           699     France  Female  39.0       1       0.00              2        0.0             0.0         938
    5          6    15574012       Chu           645      Spain    Male  44.0       8  113755.78              2        1.0             0.0        1497
  ...        ...         ...       ...           ...        ...     ...   ...     ...        ...            ...        ...             ...
 9997       9998    15584532       Liu           709     France  Female  36.0       7       0.00              1        0.0             1.0         420
 9998       9999    15682355  Sabbatini          772    Germany    Male  42.0       3   75075.31              2        1.0             0.0         928
 9999       9999    15682355  Sabbatini          772    Germany    Male  42.0       3   75075.31              2        1.0             0.0         928
10000      10000    15628319    Walker           792     France  Female  28.0       4  130142.79              1        1.0             0.0         381
10001      10000    15628319    Walker           792     France  Female  28.0       4  130142.79              1        1.0             0.0         381

9998 rows × 14 columns
```

- **Analysis of Categorical Features and Class Distributions**: The dataset's categorical features were analyzed to understand customer demographics. Geography showed a majority from France, with near-balanced gender distribution. The target variable, Exited, was imbalanced, highlighting the need for sampling techniques to improve prediction accuracy. Surname showed diversity but little value for modeling.



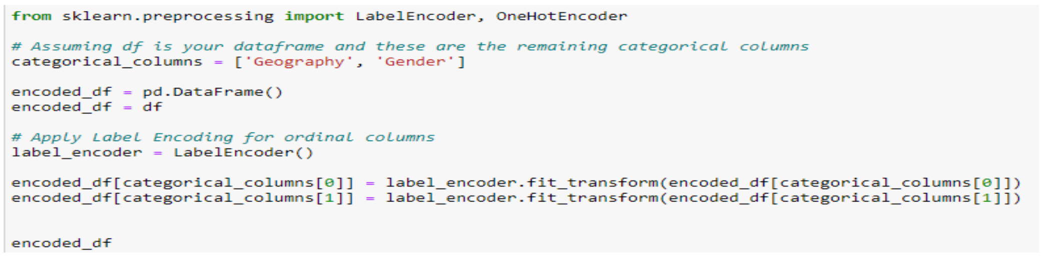
```python
print(df['Surname'].value_counts())
print()

print(df['Geography'].value_counts())
print()

print(df['Gender'].value_counts())
print()

print(df['Exited'].value_counts())
```

2. **Dropping Un-necessary columns from the dataset:** Irrelevant columns, such as RowNumber, Surname, and CustomerId, were dropped using df.drop(), retaining only features essential for prediction.



```python
df_dropped = encoded_df.drop(['RowNumber', 'Surname', 'CustomerId'], axis=1)

df_dropped
```

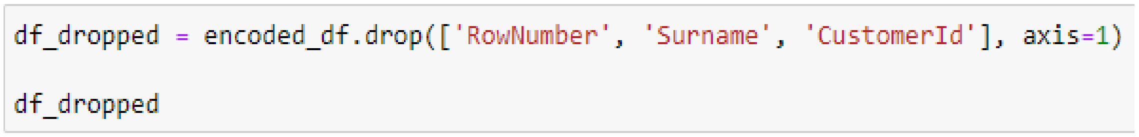
- **Encoding Categorical Features:** Categorical features Geography and Gender were converted to numerical representations using Label Encoder. This step transformed the dataset into a machine-learning-friendly format while preserving essential information.



```python
from sklearn.preprocessing import LabelEncoder, OneHotEncoder

# Assuming df is your dataframe and these are the remaining categorical columns
categorical_columns = ['Geography', 'Gender']

encoded_df = pd.DataFrame()
encoded_df = df

# Apply Label Encoding for ordinal columns
label_encoder = LabelEncoder()

encoded_df[categorical_columns[0]] = label_encoder.fit_transform(encoded_df[categorical_columns[0]])
encoded_df[categorical_columns[1]] = label_encoder.fit_transform(encoded_df[categorical_columns[1]])


encoded_df
```

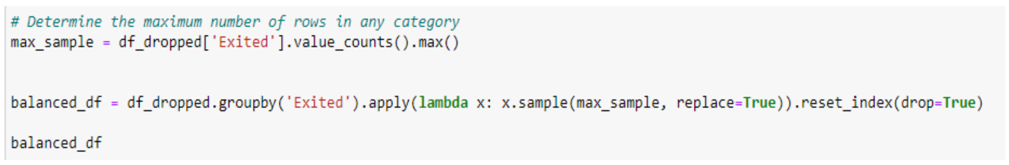
3. **Feature Selection:** Feature selection is a technique used to choose the most important and relevant features for training the model. This helps in reducing the complexity of the model, improving model accuracy, and preventing overfitting.

- **Dropping Un-necessary columns from the dataset:** Irrelevant columns, such as RowNumber, Surname, and CustomerId, were dropped using df.drop(), retaining only features essential for prediction.



```python
df_dropped = encoded_df.drop(['RowNumber', 'Surname', 'CustomerId'], axis=1)

df_dropped
```

- **Selecting Important Features**: Additional feature selection methods, such as statistical tests, correlation analysis, and domain expertise, can help identify the most relevant features for training. Removing highly correlated features is important to avoid multicollinearity, which can negatively impact model performance.
- **Data Balancing:** The target variable, Exited, was imbalanced with 7,960 non-exited and 2,038 exited cases. Oversampling was applied using the maximum sampling method to balance the dataset by replicating minority class samples.



```
# Determine the maximum number of rows in any category
max_sample = df_dropped['Exited'].value_counts().max()


balanced_df = df_dropped.groupby('Exited').apply(lambda x: x.sample(max_sample, replace=True)).reset_index(drop=True)

balanced_df
```

4. **Resulting Dataset:** The final dataset contains 11 columns, including numerical features like CreditScore, Age, Balance, and encoded categorical features such as Geography and Gender. The balanced dataset is ready for use in the modeling phase, ensuring fairness and accuracy during training.
5. **Splitting the Data into Training and Testing Sets:** The dataset was split into training (80%) and testing (20%) sets using *train_test_split* from Scikit-learn. Features (X) were separated from the target variable (Exited), ensuring the model is trained on one portion and validated on another for better generalization and evaluation. The split was done with a fixed random state for reproducibility.

4 MODELING

objective:

The primary goal in model selection was to identify a machine learning algorithm that could deliver high accuracy while efficiently handling complex data structures, class imbalances, and potential overfitting. Given the critical nature of accurate classification, the focus was on an algorithm that could adapt well to varied data distributions and ensure robust generalization.

Selection of Modelling Techniques: A range of supervised and unsupervised learning methods was employed, including Gradient Boosted Trees, Deep Learning, Random Forest, Logistic Regression, Naive Bayes, and Decision Tree classifiers.

4.1 RESULTS FROM RAPID MINNER

Model Comparison Based on Accuracy

MODEL	Accuracy	Standard Deviation	Gains	Total Time	Training Time	Scoring Time
Naive Bayes	0.715259	0.005346	1958	52089	15.57789	224.5603
Generalized Linear Model	0.687336	0.002125	1704	56999	46.73367	200.5339
Logistic Regression	0.686676	0.001923	1698	55150	17.90201	145.7286
Fast Large Margin	0.608924	0.010153	990	83166	152.4497	143.3731
Deep Learning	0.777754	0.003291	2526	106061	416.7714	228.0151
Decision Tree	0.698396	0.008939	1804	60688	13.94472	141.8028
Random Forest	0.737465	0.01052	2160	236086	22.86432	472.6759
Gradient Boosted Trees	0.814686	0.007397	2862	310644	293.4045	590.9234

Model Accuracy Comparison

In RapidMiner, various machine learning models were evaluated based on accuracy, standard deviation, training, and scoring time. Gradient Boosted Trees had the highest accuracy at 81.47%, followed by Deep Learning (77.78%). Random Forest scored 73.75%, while Naive Bayes (71.53%) and Logistic Regression (68.67%) performed well on simpler tasks. The Generalized Linear Model (68.73%) and Decision Tree (69.84%) showed moderate results, and Fast Large Margin had the lowest accuracy at 60.89%.

4.2 Model Comparison Based on Precision

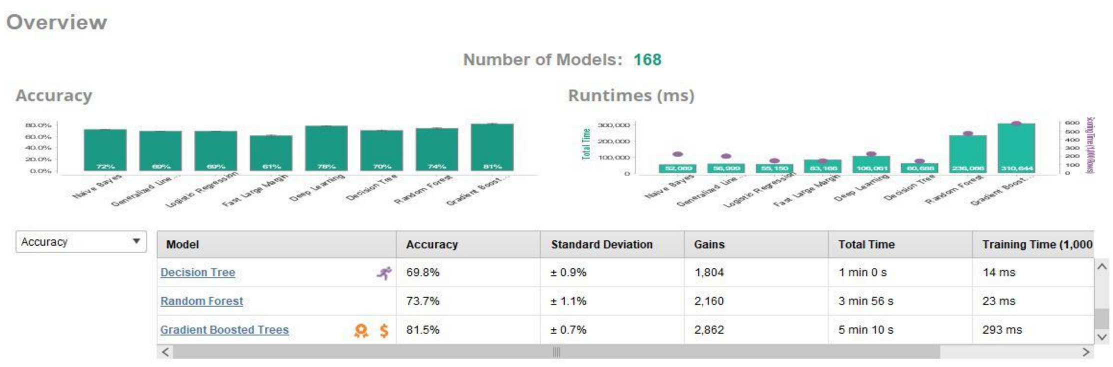
MODEL	Precision	Standard Deviation	Gains	Total Time	Training Time	Scoring Time
Naive Bayes	0.74767	0.008592	1958	52089	15.57789	224.5603
Generalized Linear Model	0.651945	0.002558	1704	56999	46.73367	200.5339
Logistic Regression	0.651573	0.002521	1698	55150	17.90201	145.7286
Fast Large Margin	0.611217	0.007327	990	83166	152.4497	143.3731
Deep Learning	0.754876	0.00738	2526	106061	416.7714	228.0151
Decision Tree	0.825588	0.015412	1804	60688	13.94472	141.8028
Random Forest	0.775983	0.01422	2160	236086	22.86432	472.6759
Gradient Boosted Trees	0.792466	0.007527	2862	310644	293.4045	590.9234

4.3 Model Comparison Table (Recall)

Model	Recall	Standard Deviation	Gain	Total Time	Training Time	Scoring Time
Naive Bayes	0.649954	0.005622	1958	52089	15.57789	224.5603
Generalized Linear Model	0.803869	0.007218	1704	56999	46.73367	200.5339
Logistic Regression	0.80255	0.006676	1698	55150	17.90201	145.7286
Fast Large Margin	0.597614	0.027175	990	83166	152.4497	143.3731
Deep Learning	0.822781	0.013399	2526	106061	416.7714	228.0151
Decision Tree	0.503074	0.016235	1804	60688	13.94472	141.8028
Random Forest	0.667987	0.016426	2160	236086	22.86432	472.6759
Gradient Boosted Trees	0.852685	0.015586	2862	310644	293.4045	590.9234

Gradient Boosted Trees achieved the highest recall at 85.27%, followed by Deep Learning at 82.28%. Generalized Linear Model (80.39%) and Logistic Regression (80.25%) provided reliable recall. Random Forest (66.80%) and Naive Bayes (64.99%) were moderate, while Fast Large Margin (59.76%) and Decision Tree (50.31%) struggled. For recall-focused tasks, Gradient Boosted Trees and Deep Learning are the most effective models.

4.4 ACCURACY COMPARISON

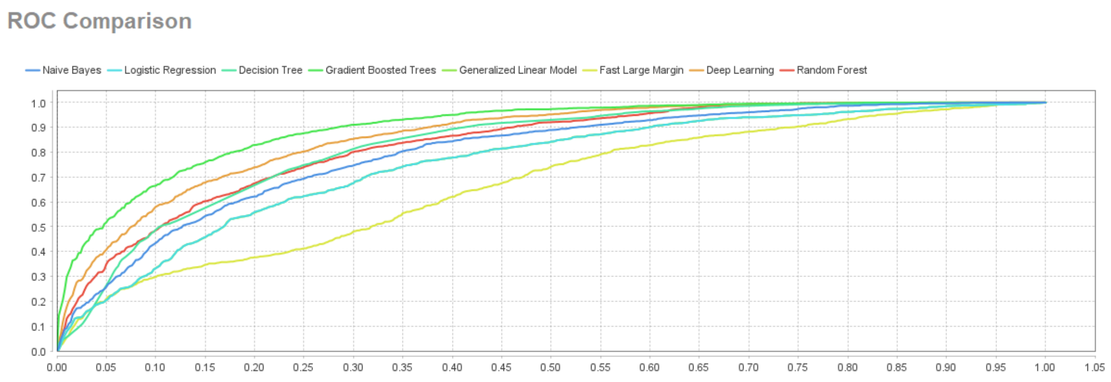


Model	Accuracy	Standard Deviation	Gains	Total Time	Training Time (1,000
Decision Tree	69.8%	± 0.9%	1,804	1 min 0 s	14 ms
Random Forest	73.7%	± 1.1%	2,160	3 min 56 s	23 ms
Gradient Boosted Trees	81.5%	± 0.7%	2,862	5 min 10 s	293 ms

Conclusion:

Gradient Boosted Trees led with the highest recall (85.27%) and accuracy (81.47%), followed by Deep Learning (recall: 82.28%, accuracy: 77.78%). Logistic Regression and Generalized Linear Model had balanced recall (80.25%–80.39%) and moderate accuracy (68.67%–68.73%). Random Forest showed good accuracy (73.75%) but lower recall (66.80%), while Naive Bayes performed moderately. Fast Large Margin and Decision Tree underperformed. Overall, Gradient Boosted Trees outperformed all models.

4.5 ROC COMPARISON OF EACH MODEL

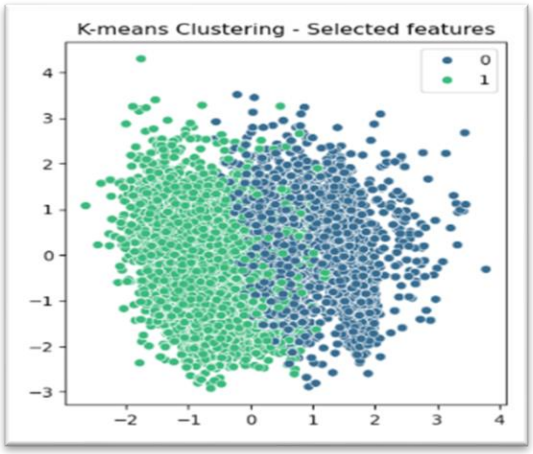


4.6 Model Building and Performance Analysis:

Supervised Learning:

Gradient Boosted Trees led with the highest accuracy (81.47%) and recall (85.27%), followed by Deep Learning (77.78% accuracy, 82.28% recall). Logistic Regression and Generalized Linear Models had balanced recall and moderate accuracy. Random Forest showed good accuracy (73.75%) but lower recall (66.80%), while Decision Tree and Fast Large Margin underperformed.

Unsupervised Learning: K-means clustering on the reduced feature set, with PCA for dimensionality reduction, revealed clusters. The silhouette score of 0.206 indicated moderate clustering effectiveness, visualized through scatter plots..



4.7 Model Selection:

The XGBoost Classifier (XGBClassifier) was chosen for its ability to handle large datasets, high accuracy, and effectiveness in mitigating overfitting through gradient boosting. Its iterative training approach, where each model corrects the previous one's errors, makes it ideal for high-stakes classification tasks.

4.8 Model Initialization:

The XGBoost Classifier was selected for its strong performance on structured and imbalanced datasets. It was initialized with use_label_encoder=False to avoid warnings and eval_metric='logloss' to optimize for logarithmic loss during training.

4.9 Model Training:

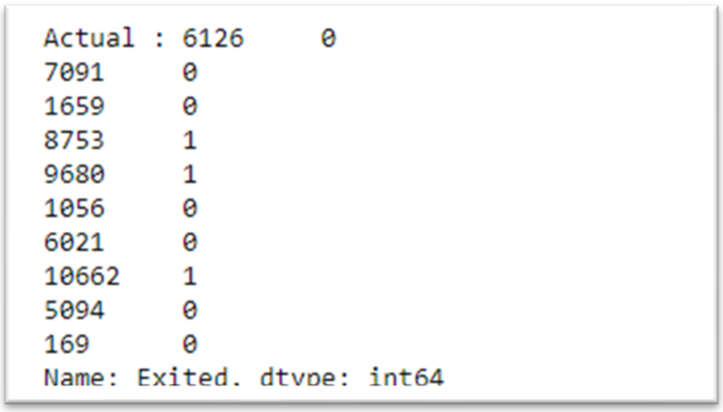
The model was trained using the scaled and selected features dataset, *X_train_scaled_sel_features* for features and *y_train_sel_features* for target labels, with the fit () method to learn patterns.

4.10 Prediction Generation:

After training, the **XGBoost Classifier** was applied to the test dataset (X_test_scaled_sel_features) to assess its performance on unseen data. The models predict () function was used to generate predicted labels, which were then compared to the actual labels (y_test_sel_features) from the test set.

Actual Labels (first 10):

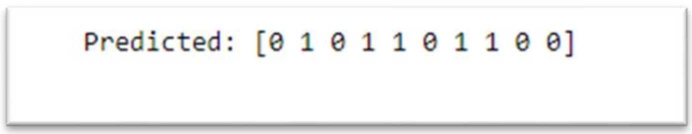
These represent the true outcomes for the test set:



```
Actual : 6126     0
7091     0
1659     0
8753     1
9680     1
1056     0
6021     0
10662    1
5094     0
169      0
Name: Exited, dtype: int64
```

Predicted Labels (first 10):

These represent the outcomes predicted by the XGBoost model:



```
Predicted: [0 1 0 1 1 0 1 1 0 0]
```

5 EVALUATION

The evaluation phase is crucial for assessing the XGBoost model's effectiveness in predicting churn, ensuring alignment with business goals, and validating its readiness for deployment. It involves reviewing performance, identifying areas for improvement, and addressing key business issues.

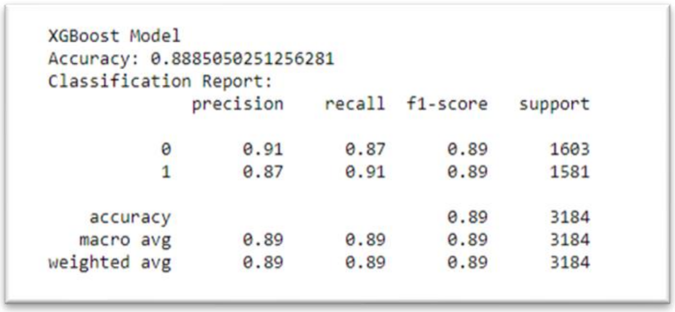
5.1 Accuracy Assessment

The XGBoost model achieved 88.85% accuracy in predicting customer churn, demonstrating its ability to effectively classify instances. However, in imbalanced datasets like churn prediction, accuracy alone can be misleading, making additional metrics essential for a thorough evaluation.

To address this, the model's performance was further assessed using the following metrics:

1. **Precision** - Indicates the proportion of correctly predicted churned customers (or non-churned) out of all instances predicted as such. High precision ensures that false positives are minimized.
2. **Recall** - Measures the model's ability to correctly identify all actual churned customers, highlighting its sensitivity to the positive class.
3. **F1-Score** - Balances precision and recall, providing a single metric that is particularly useful in cases of imbalanced data.

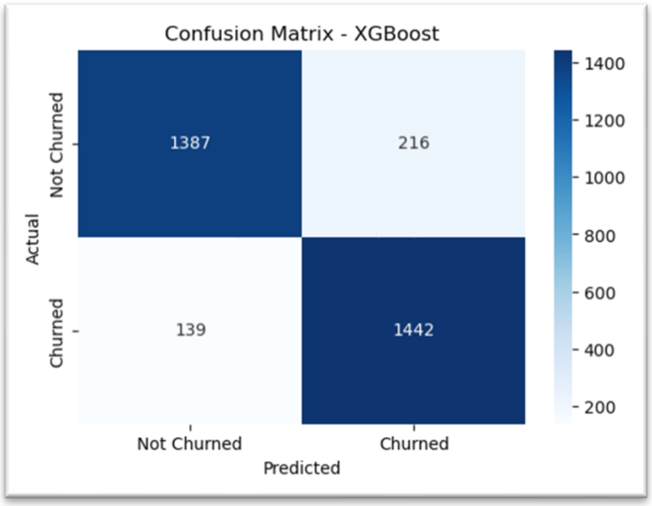
5.2 Precision, Recall, and F1-Score Evaluation



```
XGBoost Model
Accuracy: 0.8885050251256281
Classification Report:
              precision    recall  f1-score   support

           0       0.91      0.87      0.89      1603
           1       0.87      0.91      0.89      1581

    accuracy                           0.89      3184
   macro avg       0.89      0.89      0.89      3184
weighted avg       0.89      0.89      0.89      3184
```

5.3 Confusion Matrix Analysis

The confusion matrix for the XGBoost model showed 87% accuracy for non-churned customers (Class 0) and 91% for churned customers (Class 1), with minimal misclassifications and low false positives and negatives.



5.4 Conclusion

The XGBoost model demonstrated excellent performance in predicting customer churn, achieving an accuracy of **88.85%**, along with balanced precision, recall, and F1-scores across both churned and non-churned classes. The confusion matrix analysis confirmed the model's ability to effectively differentiate between customers likely to churn and those who will remain, with minimal misclassification errors. These results highlight the model's capability to handle the complexity of customer behavior and its suitability for deployment in real-world scenarios.

5.5 Business Impact

Accurate churn prediction enables businesses to implement targeted retention strategies, allocate resources efficiently, and enhance customer satisfaction through early intervention. By reducing costs andpanies can drive revenue growth, lower acquisition costs, and make data-driven decisions to refine marketing strategies and adapt to customer needs.

6 DEPLOYMENT

6.1 Objective:

The primary goal of the deployment phase is to implement the XGBoost model in a real-world environment to provide actionable insights for decision-making. The deployment ensures that the model is accessible to end-users, integrates seamlessly with existing systems, and remains robust through monitoring and maintenance.

6.2 Deployment Plan:

A detailed deployment plan outlines the steps required to transition the model from development to production. For this project, the plan includes:

- **Platform Selection**: Hosting the model on a cloud-based platform, such as AWS or Azure, to ensure scalability and reliability.
- **Integration**: Connecting the model with the customer relationship management (CRM) system to process customer data in real-time.

- **User Accessibility**: Developing an interface, such as a web application or API, that allows business users to interact with the model's predictions.

6.3 Model Integration and Automation

- The XGBoost model is wrapped in an API using a framework like Flask or FastAPI to serve predictions on-demand.
- Automation pipelines are created to fetch and preprocess incoming customer data, pass it to the model, and return the results in a user-friendly format.

6.4 Documentation:

Comprehensive documentation is created to ensure transparency and ease of use. This includes:

- **Technical Documentation**: Detailed explanations of the model's architecture, preprocessing steps, and dependencies.
- **User Guide**: Instructions for business users on interpreting model predictions and leveraging them for decision-making.
- **Maintenance Procedures**: Guidelines for updating the model, retraining it with new data, and addressing potential issues.

6.5 Monitoring and Maintenance:

To maintain the model's performance and reliability, a robust monitoring system is established:

- **Performance Metrics**: Regular tracking of accuracy, precision, recall, and F1-scores to detect potential drift in model predictions.
- **Data Drift Detection**: Monitoring incoming data for shifts in customer behavior that may impact the model's predictions.