Phase-2 Submission Template

Student Name: G.VENMATHI

Register Number: 421823106050

Institution: Saraswathy College of Engineering and Technology

Department: BE-Electronics and Communication Engineering

Date of Submission: 12-05-2025

Github Repository Link:

https://github.com/gvenmathi2128/ Chartbot-ai.git

PREDICTING CUSTOMER CHURN USING MACHINE LEARNING TO UNCOVER HIDDEN PATTERNS

1.Problem Statement

- ***Revisit and Refine the Problem***
  Problem Refinement: Develop a machine learning model that accurately predicts customer churn for a telecom service provider based on customer demographics, usage patterns, and billing information.
- ***Problem Type***
  Classification Problem: Predicting customer churn is a binary classification problem, where the model predicts one of two outcomes: customer will churn (yes) or customer will not churn (no).
- ***Impact and Relevance***
  Business Impact: Solving customer churn prediction matters because it enables telecom companies to proactively retain customers, reduce revenue loss, and improve customer satisfaction, ultimately driving business growth and profitability.

2.Objectives of the Project

- ***Key Technical Objectives***

  Develop an Accurate Prediction Model: Build a machine learning model that accurately predicts customer churn based on historical customer data, achieving a high level of precision, recall, and F1-score.

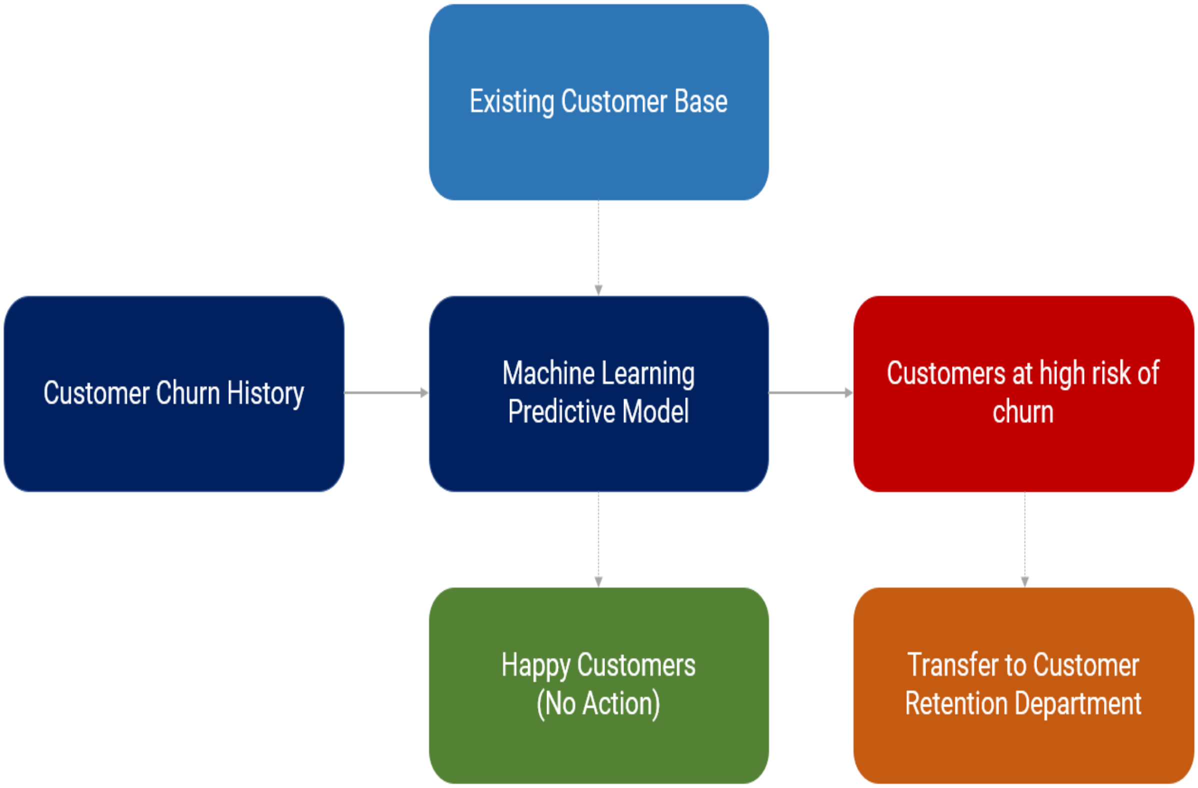
- ***Model Objectives***

  The model aims to achieve high accuracy in predicting customer churn, while also providing interpretable insights into the key factors driving churn, to enable telecom companies to take actionable, data-driven decisions to improve customer retention and reduce revenue loss.

- ***Evolution of Goal***

  After data exploration, the goal has evolved to focus on identifying the most influential features contributing to customer churn, in addition to predicting churn, to provide actionable insights for targeted retention strategies and improved customer satisfaction.

3. Flowchart of the Project Workflow


Existing Customer Base
Customer Churn History
Machine Learning Predictive Model
Customers at high risk of churn
Happy Customers (No Action)
Transfer to Customer Retention Department

4. Data Description

- ***Dataset Name and Origin***

  Dataset Name: The dataset used for this project is the "Telco Customer Churn" dataset.

  Origin: This dataset is sourced from Kaggle, a popular platform for data science competitions and hosting datasets.

- ***Type of Data***

  The dataset consists of structured data, comprising tabular information with well-defined fields and variables, such as customer demographics, account information, and usage patterns, which can be easily analyzed using machine learning algorithms.

- ***Number of Records and Features***

  The dataset contains 7,043 records with 21 features, including demographic information, account details, and usage patterns, which will be used to train and evaluate the machine learning model for predicting customer churn.

- ***Static or Dynamic Dataset***

  The dataset used for this project is a static dataset, meaning it is a snapshot of customer information at a particular point in time and does not change or update in real-time.

- ***Target Variable***

  The target variable for this supervised learning problem is "Churn", a binary variable indicating whether a customer has churned (yes/no) or (1/0), which the model will predict based on the input features.

5. Data Preprocessing

1. Handle Missing Values

Identified missing values in the dataset.

Decided on imputation strategy (mean, median, impute with specific value, or removal).

2. Remove Duplicate Records

Checked for duplicate records.

Removed duplicates if any, or justified their presence.

3. Detect and Treat Outliers

Used statistical methods (e.g., IQR) or visualization (box plots) to detect outliers.

Decided on treating outliers (removal, transformation, or keeping as is).

4. Convert Data Types

Ensured data types were consistent and appropriate for analysis (e.g., converting categorical variables to category type).

5. Encode Categorical Variables

Used label encoding or one-hot encoding based on the nature of the categorical variable and model requirements.

6. Normalize or Standardize Features

Applied normalization or standardization techniques where required by the model (e.g., Min-Max Scaling, Standard Scaler).

Code

```
import pandas as pd

from sklearn.preprocessing import StandardScaler, OneHotEncoder

from sklearn.impute import SimpleImputer
```

```
# Load data

df = pd.read_csv('data.csv')
```

```
# Handle missing values

imputer = SimpleImputer(strategy='mean')

df[['column1', 'column2']] = imputer.fit_transform(df[['column1', 'column2']])
```

```
# Remove duplicates

df.drop_duplicates(inplace=True)
```

```
# Detect and treat outliers

Q1 = df['column'].quantile(0.25)

Q3 = df['column'].quantile(0.75)

IQR = Q3 - Q1

df = df[~((df['column'] < (Q1 - 1.5 * IQR)) | (df['column'] > (Q3 + 1.5 * IQR)))]
```

```
# Encode categorical variables

encoder = OneHotEncoder()

encoded_columns = encoder.fit_transform(df[['category']])

encoded_df = pd.DataFrame(encoded_columns.toarray(), columns=encoder.get_feature_names_out())

df = pd.concat([df, encoded_df], axis=1)
```

```
# Normalize features

scaler = StandardScaler()

df[['feature1', 'feature2']] = scaler.fit_transform(df[['feature1', 'feature2']])
```

6. Exploratory Data Analysis (EDA)

- ***Univariate Analysis:***

  Histograms: Use histograms to visualize the distribution of continuous features.

  Boxplots: Boxplots can reveal the central tendency and spread of data, as well as potential outliers.

- ***Bivariate/Multivariate Analysis:***

  Correlation matrix: shows relationships between features

  Pairplots: visualizes relationships between multiple features

  Scatterplots: displays relationships between two features

  Grouped bar plots: compares categorical features

  Correlations between features and target

Patterns and trends

Potential predictors

Insights for model development and feature selection.

- ***Insights Summary***:

Highlight patterns, trends, and interesting observations: Identify key findings from the data analysis.
Mention which features may influence the model and why: Determine which features are most relevant to the target variable and explain their potential impact on the model.

7. Feature Engineering

Enhance or transform data to improve model performance.

- *Create new features: Based on domain knowledge or EDA insights.*
- *Combine or split columns: Extract relevant information (e.g., date parts).*
- *Apply techniques: Binning, polynomial features, ratios, etc.*
- *Dimensionality reduction (optional): Use methods like PCA to reduce feature complexity.*
- *Justify changes: Explain the reasoning behind adding or removing features.*

Goal: Improve model performance, reduce overfitting, and enhance interpretability.

8. Model Building

- ***Model Selected:***
  - *Logistic Regression*
  - *Random Forest Classifier*
- ***Data Split***

```
from sklearn.model_selection import train_test_split
```

```
X_train, X_test, y_train, y_test = train_test_split(df.drop('target', axis=1),
df['target'], test_size=0.2, random_state=42, stratify=df['target'])
```

- ***Logistic Regression***

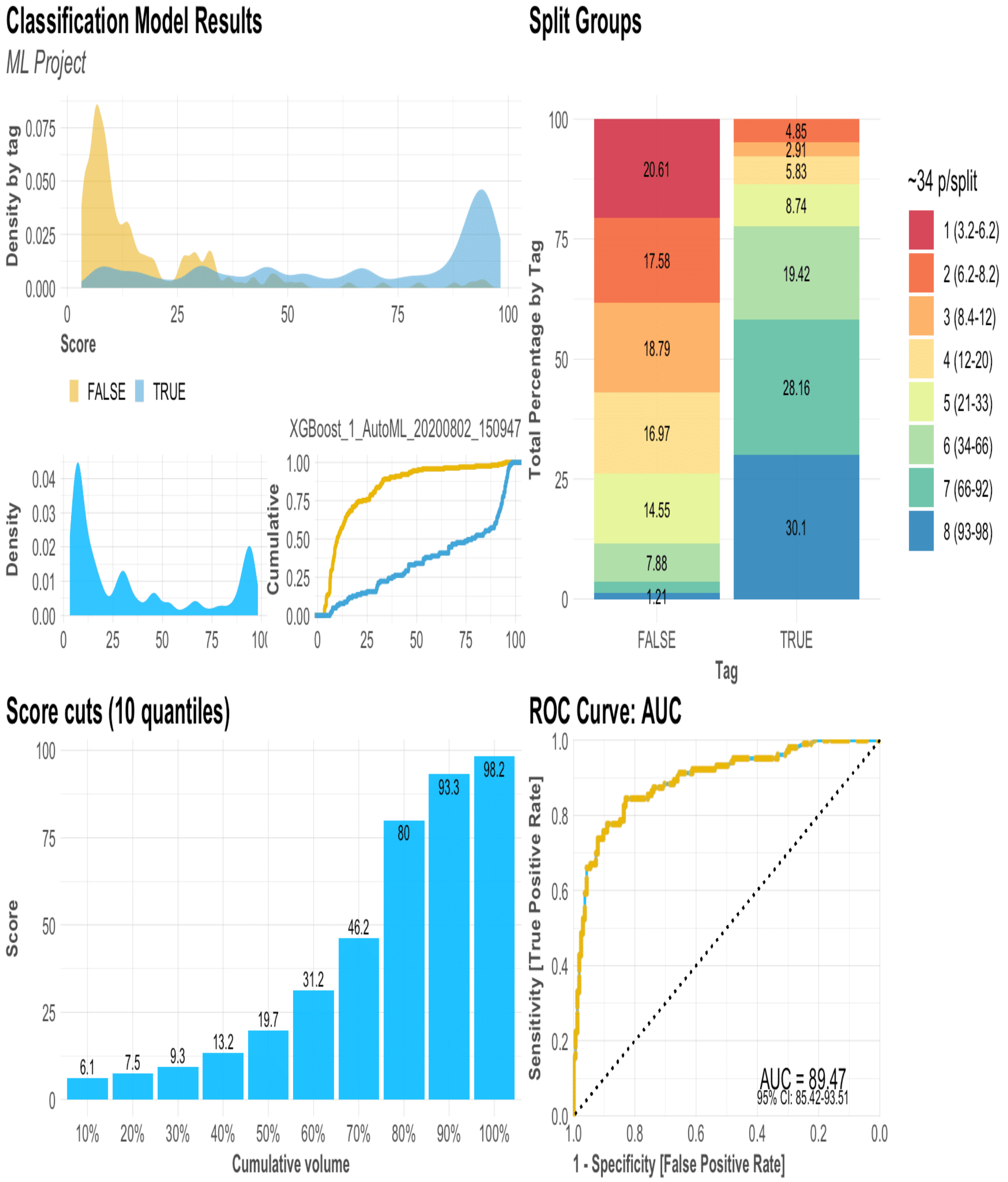
```
from sklearn.linear_model import LogisticRegression

from sklearn.metrics import accuracy_score, precision_score, recall_score,
f1_score

logreg = LogisticRegression(max_iter=1000)

logreg.fit(X_train, y_train)

y_pred_logreg = logreg.predict(X_test)
```

```
print("Logistic Regression Metrics:")

print(f"Accuracy: {accuracy_score(y_test, y_pred_logreg):.3f}")

print(f"Precision: {precision_score(y_test, y_pred_logreg):.3f}")

print(f"Recall: {recall_score(y_test, y_pred_logreg):.3f}")

print(f"F1-score: {f1_score(y_test, y_pred_logreg):.3f}")
```

- ***Random Forest Classifier***

```
from sklearn.ensemble import RandomForestClassifier

rf = RandomForestClassifier(n_estimators=100, random_state=42)

rf.fit(X_train, y_train)

y_pred_rf = rf.predict(X_test)

print("Random Forest Metrics:")

print(f"Accuracy: {accuracy_score(y_test, y_pred_rf):.3f}")

print(f"Precision: {precision_score(y_test, y_pred_rf):.3f}")

print(f"Recall: {recall_score(y_test, y_pred_rf):.3f}")

print(f"F1-score: {f1_score(y_test, y_pred_rf):.3f}")
```

9. Visualization of Results & Model Insights



10. Tools and Technologies Used

- ***Programming Language:***

  Python or R

- ***IDE/Notebook:***

  Google Colab
  Jupyter Notebook
  VS Code

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
```

```
# Load data
df = pd.read_csv('data.csv')
```

```
# Split data
X = df.drop('target', axis=1)
y = df['target']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
# Train model
model = LogisticRegression()
model.fit(X_train, y_train)
```

- ***Libraries:***

  pandas
  numpy
  seaborn
  matplotlib
  XGBoost

Visualization Tools:

Plotly, Tableau, Power BI

11. Team Members and Contributions

NAME	*CONTRIBUTIONS*	*RESPONSIBILITIES*
G.VENMATHI	*Data Cleaning* *Model Development*	*Ensuring the quality and accuracy of the data.* *Developed and trained machine learning models.*
K. TAMIZHSELVI	*Exploratory Data Analysis*	*Conducted EDA to understand the distribution of variables, relationships between variables, and identify patterns or trends in the data.*
VERTISELVAN.N	*Feature Engineering*	*Created new features from existing ones to improve model performance, handled categorical variables, and transformed data into suitable formats.*
VETRIVEL.A	*Documentation and Reporting*	*Maintained project documentation, created reports, and presented findings and insights to stakeholders.*