Department of Physics, University of Colombo

PH3022 – Machine Leaning & Neural Computation

Predicting Churn in ABC Bank Customers

Name: P. V. Nipun Lakshitha

Index No: 15367

Registration No: 2020s18114

Group No: 04

Date: 07/02/2024

Table of Contents

Introduction

The dataset provided consists of information about ABC Bank customers, including various features such as credit score, country, gender, age, tenure, account balance, number of products, credit card status, active membership, estimated salary, and the churn status. The dataset contains 10,006 rows of data, each representing a unique customer.

The primary goal of this project is to develop a machine learning model that classifies bank customers based on their churn status. Customer churn prediction is a critical task as it directly impacts customer retention and overall revenue stability for ABC Bank. The project aims to categorize customers into two groups: positive (churn) or negative (no churn).

Importance of Churn Prediction

Identifying potential churners in advance is critical for the bank since it allows them to develop focused client retention measures. Keeping current clients is more cost-effective than obtaining new ones, and this proactive approach reduces customer loss. For a business to be sustainable, it needs steady revenue streams, and reducing loss of clients goes a long way toward obtaining these streams. By anticipating and reducing loss of clients, the bank may optimize its strategies, improve customer happiness, and maintain loyal customers, all of which contribute to long-term success and competitiveness in the marketplace.

Dataset Description

- **Customer ID:** Customer identification number (Integer)
- **Credit score**: Credit scores of customers (Integer, Range: 300 – 900)
- **Country**: Country of the customer (String, Values: France/Spain/Germany)
- **Gender**: Gender of the customer (String, Values: Female/Male)
- **Age**: Age of the customer (Integer, Range: 18 – 93)
- **Tenure**: Years with the bank (Integer, Range: 0 – 10)
- **Account balance**: Currently existing account balance (Float, Range: 0 – 260,000.00)
- **Products number**: Number of products from the bank (Integer, Range: 1 – 4)
- **Credit card**: Whether the customer has a credit card (Boolean, Values: Yes/No)
- **Active member**: Whether the customer is an active member (Boolean, Values: Yes/No)
- **Estimated salary**: Estimated monthly salary of the customer (Float, Range: 10 – 200,000.0)
- **Churn**: Churn status of the customer (Boolean, Values: Yes/No)

Machine Learning Models

For this project, Support Vector Machines (SVM) and Logistic Regression have been selected. Logistic Regression is chosen for its simplicity, interpretability, and efficiency, while Support Vector Machines are selected for their ability to handle non-linear relationships and robustness to outliers. Combining these models enables a thorough examination of the dataset while finding a balance between ease of use and the capacity to identify complex patterns.

Feature Engineering

- The Customer_ID column acts as a unique identification and has no churn prediction value. Dropping it guarantees that Customer_ID has no impact on the model.
- One-hot encoding is used to encode both gender and country. By doing this, categorical variables are transformed into binary matrices that can be used with machine learning models.
- If there were any missing values, appropriate strategies such as approximation (using mean, median, or other methods) or removal of rows/columns might be applied.
- Numerical features like 'Credit_score,' 'Age,' 'Tenure,' 'Account_balance,' and 'Estimated_salary' are not explicitly mentioned to be scaled in the initial steps. However, depending on the choice of machine learning models, such as Support Vector Machines, normalizing or scaling numerical features might be considered in subsequent iterations.

Progress of the Project

Data Preprocessing:

- The dataset was loaded, and initial exploration was conducted.
- Irrelevant columns were dropped, and categorical variables were encoded.
- The data was split into training and testing sets.

Model Implementation:

- Logistic Regression was implemented using scikit-learn.
- The model was trained on the training set and evaluated on the testing set.

Evaluation:

- Initial evaluation metrics, including accuracy, confusion matrix, and classification report, were obtained.
- Further hyperparameter tuning and model optimization are planned.

```python
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import OneHotEncoder
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report

# Load the dataset
df = pd.read_csv('Bank Customer Churn Prediction Classification Dataset.csv')

# Display the first few rows of the DataFrame to verify
df.head(10)

# Data Preprocessing
# Drop irrelevant columns
df = df.drop(columns='customer_id', axis=1)

# One-hot encode categorical variables
categorical_columns = ['country', 'gender']
df = pd.get_dummies(df, columns=categorical_columns, drop_first=True)

# Fill missing values in numerical columns with the mean
numerical_columns = df.select_dtypes(include=['float64', 'int64']).columns
df[numerical_columns] = df[numerical_columns].fillna(df[numerical_columns].mean())

# Display the first few rows of the DataFrame to verify
df.head(10)

# Split the data into features (X) and target variable (y)
X = df.drop(columns=['churn (Churn Status)'])
y = df['churn (Churn Status)']

# Split the data into training and testing sets (80% training, 20% testing)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Model Implementation - Logistic Regression
logistic_model = LogisticRegression(random_state=42)

# Train the model on the training set
```

```
logistic_model.fit(X_train, y_train)

# Make predictions on the testing set
logistic_predictions = logistic_model.predict(X_test)

# Evaluation
accuracy = accuracy_score(y_test, logistic_predictions)
conf_matrix = confusion_matrix(y_test, logistic_predictions)
classification_report_str = classification_report(y_test,
logistic_predictions)

# Print the evaluation metrics
print(f"Accuracy: {accuracy}")
print(f"Confusion Matrix:\n{conf_matrix}")
print(f"Classification Report:\n{classification_report_str}")
```

Output:

```
Accuracy: 0.8021978021978022
Confusion Matrix:
[[1583   46]
 [ 350   23]]
Classification Report:
              precision    recall  f1-score   support

           0       0.82      0.97      0.89      1629
           1       0.33      0.06      0.10       373

    accuracy                           0.80      2002
   macro avg       0.58      0.52      0.50      2002
weighted avg       0.73      0.80      0.74      2002
```