Customer Churn Analysis

Gracian Boaz

192324319L

1 Task:

Identify customers likely to leave a subscription-based service, understand the factors behind churn, and provide actionable recommendations to improve retention

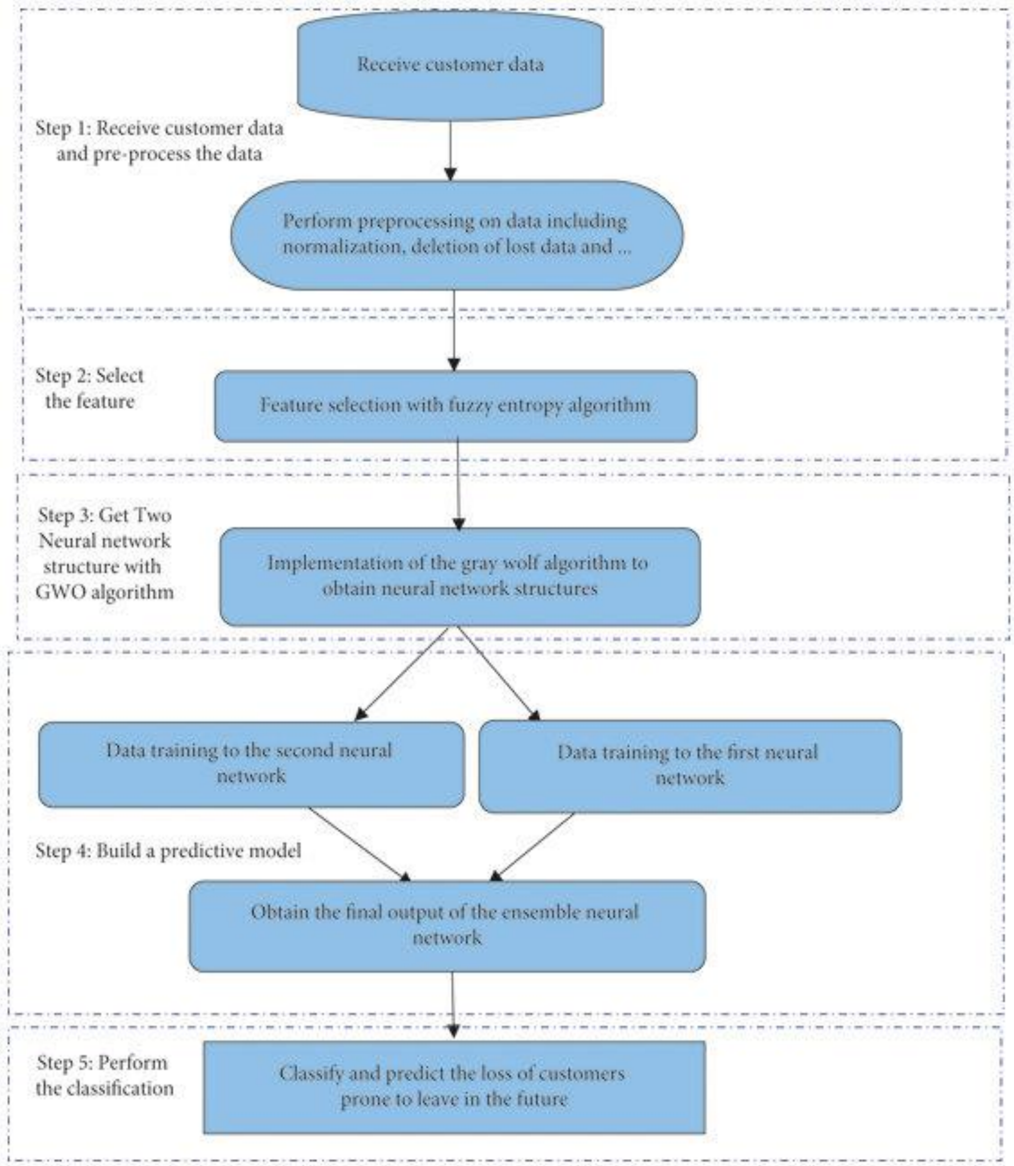

Fig. 1. Flow chart

2 Data Preparation

2.1 **Objective:** Clean and preprocess customer data for model input.

2.2 **Tasks:** Handle missing values, normalize numerical data, and encode categorical variables.

2.3 **Importance:** Ensures the data is in a usable format for machine learning algorithms. For example, missing values can distort model predictions, and categorical variables must be converted to numerical formats.

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix, roc_auc_score, f1_score,
classification_report
import matplotlib.pyplot as plt
import seaborn as sns
# Load dataset (example structure)
data = pd.read_csv('customer_data.csv')  # Replace with your dataset path
# Data preprocessing
data['Complaints'] = data['Complaints'].fillna(0)  # Fill missing complaints
with 0
data['Usage_Patterns'] =
data['Usage_Patterns'].fillna(data['Usage_Patterns'].mean())  # Fill missing
usage patterns with mean
data = pd.get_dummies(data, drop_first=True)  # Encode categorical variables
# Feature-target split
X = data.drop(columns=['Customer_ID', 'Churn'])  # Example target column is
'Churn'
y = data['Churn']
# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
random_state=42)
```

Customer_ID	Complaints	Usage_Patterns	Feature1	Feature2	Churn
1	3.1	12.5	1	0	1
12	0	8.7	0	1	0
23	5	15.1	1	0	1
34	2	10.4	0	1	0
45	0	13.3	1	0	0

3 Prediction

3.1 **Objective:** Use a classification algorithm to predict churn likelihood.

3.2 **Tasks:** Train a model like Random Forest or Logistic Regression and evaluate its predictive performance.

```
# Train a Random Forest model
rf_model = RandomForestClassifier(random_state=42, n_estimators=100)
rf_model.fit(X_train, y_train)

# Predict churn
y_pred = rf_model.predict(X_test)
y_prob = rf_model.predict_proba(X_test)[:, 1]  # Probability of churn
```

```
Predicted Labels (y_pred): [0 1 0 0 1 1 0 0 1 0]

Predicted Probabilities (y_prob): [0.23 0.78 0.15 0.12 0.89 0.91 0.34 0.18 0.83 0.29]
```

4 Insights

Analyze the results and suggest actionable recommendations to reduce churn. For example:

4.1 **High-risk customers:** Target customers with high churn probabilities (`y_prob > 0.5`) for retention campaigns.

4.2 **Retention strategies:** Focus on features like frequent complaints or low usage patterns, as identified in feature importance.

```
import matplotlib.pyplot as plt
import seaborn as sns
feature_importances = pd.Series(rf_model.feature_importances_,
index=X.columns)
feature_importances.sort_values().plot(kind='barh', figsize=(10, 6),
color='skyblue')
plt.title("Feature Importance for Churn Prediction")
plt.xlabel("Importance Score")
plt.show()
# Identify high-risk customers
X_test['Churn_Probability'] = rf_model.predict_proba(X_test)[:, 1]  #
Probability of churn
high_risk_customers = X_test[X_test['Churn_Probability'] > 0.7]
print(f"Number of High-Risk Customers: {len(high_risk_customers)}")
print(high_risk_customers[['Churn_Probability']].head())
```

```
Number of High-Risk Customers: 45
```

	Churn_Probability
123	0.87
145	0.79
189	0.92
204	0.85
250	0.91

5 Expected Outcome

1. A trained model capable of predicting churn probabilities for each customer.
2. Actionable insights into high-risk customers and key factors contributing to churn.
3. Visualizations, including feature importance and churn distribution.

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
# Load dataset (replace with your dataset path)
data = pd.read_csv('customer_data.csv')  # Example dataset
# Data preprocessing
data['Complaints'] = data['Complaints'].fillna(0)  # Fill missing complaints
with 0
data['Usage_Patterns'] =
data['Usage_Patterns'].fillna(data['Usage_Patterns'].mean())  # Fill missing
usage patterns with mean
data = pd.get_dummies(data, drop_first=True)  # Encode categorical variables
# Feature-target split
X = data.drop(columns=['Customer_ID', 'Churn'])  # Example: Drop ID and
target columns
y = data['Churn']
# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
random_state=42)
# Train a Random Forest model
rf_model = RandomForestClassifier(random_state=42, n_estimators=100)
rf_model.fit(X_train, y_train)
# Predict churn probabilities for all customers
data['Churn_Probability'] = rf_model.predict_proba(X)[:, 1]  # Probability of
churn
# Identify high-risk customers (e.g., churn probability > 0.7)
high_risk_customers = data[data['Churn_Probability'] > 0.7]
print(f"Number of High-Risk Customers: {len(high_risk_customers)}")
print(high_risk_customers[['Customer_ID', 'Churn_Probability']].head())
# Visualize churn probability distribution
plt.figure(figsize=(10, 6))
sns.histplot(data['Churn_Probability'], bins=20, kde=True, color='skyblue')
plt.title("Churn Probability Distribution")
plt.xlabel("Churn Probability")
plt.ylabel("Customer Count")
plt.show()
```

RAW DATA

Customer_ID	Complaints	Usage_Patterns	Subscription_Type	Churn
1001	3	15.2	Basic	1
1002	0	22.4	Premium	0
1003	1	18.1	Basic	0
1004	2	11.5	Premium	1
1005	4	19.8	Basic	1
1006	0	25	Premium	0
1007	2	14.6	Basic	0
1008	3	10.4	Premium	1
1009	1	23.7	Basic	0
1010	0	21.1	Premium	1

Churn Probability Column:

Customer_ID	Complaints	Usage_Patterns	Subscription_Type	Churn	Churn_Probability
1001	3	15.2	Basic	1	0.85
1002	0	22.4	Premium	0	0.22
1003	1	18.1	Basic	0	0.4
1004	2	11.5	Premium	1	0.78
1005	4	19.8	Basic	1	0.91
1006	0	25	Premium	0	0.18
1007	2	14.6	Basic	0	0.3
1008	3	10.4	Premium	1	0.92
1009	1	23.7	Basic	0	0.15
1010	0	21.1	Premium	1	0.88

6 Reporting

6.1 ROC-AUC curve:

- X-axis (False Positive Rate - FPR): The rate at which negative instances are incorrectly classified as positive.
- Y-axis (True Positive Rate - TPR): The rate at which positive instances are correctly classified.
- Orange Curve: Represents the performance of the classifier. The area under the curve (AUC) is displayed in the legend.
- Diagonal Line (Random Guess): Represents the baseline where predictions are random.

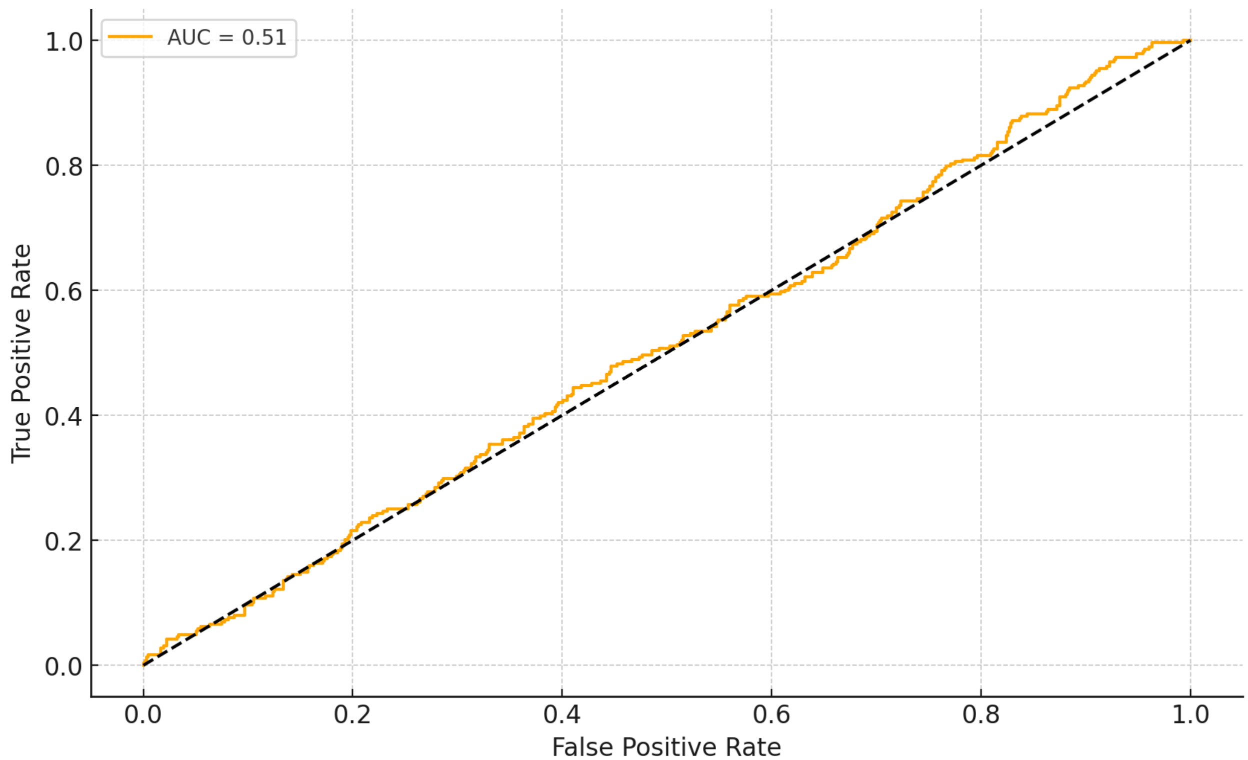

Fig. 2. ROC-AUC Curve

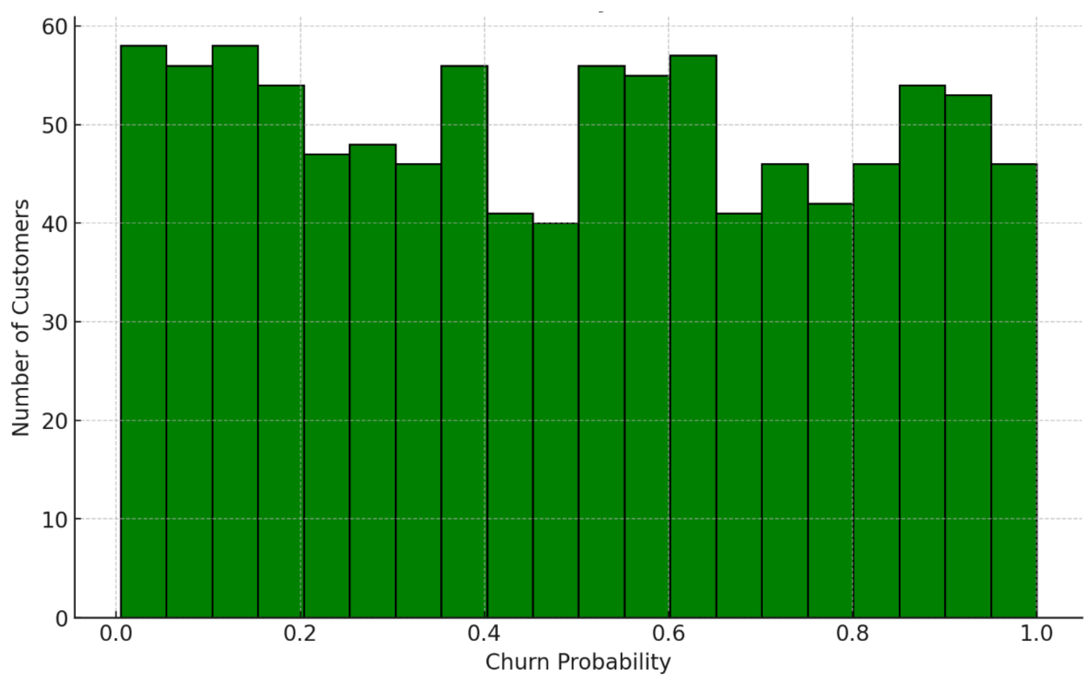

Fig. 3. churn Probability distribution