CUSTOMER CHURN PREDICTION

A Machine Learning Approach to Predict Customer Churn at Sunbase

By Divya Gazinkar

TABLE OF CONTENTS

Dataset

We are provided with a dataset in Excel format (customer_churn_data.xlsx) containing historical customer information, including customer attributes, interactions, and whether they churned or not. The dataset contains the following columns:

1. CustomerID
2. Name
3. Age
4. Gender
5. Location
6. Subscription_Length_Months
7. Monthly_Bill
8. Total_Usage_GB
9. Churn

Data Preprocessing

EDA

- The dataset contains 100000 the number of rows and 9 columns.
- Numerical data in columns are CustomerID, Age, Gender, Location, Subscription_Length_Months, Monthly_Bill, Total_Usage_GB, and Churn
- Non Numerical data in columns are Name and Location
- 50.221% are non churners and 49.779 % are churners
- There are no null values in the dataset
- Handling outliers is an important step in data preprocessing,
    - Box Plots: are a simple and effective way to visualize outliers. They display the distribution of a dataset and show potential outliers as individual data points beyond the "whiskers" of the box. Outliers can be identified as points outside the upper and lower whiskers.
    - Z-score: measures how many standard deviations a data point is away from the mean.
    - IQR: The IQR method defines outliers as data points that fall below Q1 - 1.5 * IQR or above Q3 + 1.5 * IQR, where Q1 is the 25th percentile and Q3 is the 75th percentile

    - No significant outliers were detected in the dataset
- Created a copy of base data for manipulation & processing
- The maximum of Subscription_Length_Months is 24
- The maximum age is 70
- Dropped columns " CustomerID " and "Name" as they don't provide significant insight for churn prediction.
- Divided customers into bins based on 'Subscription_Length_Months_grouping' and 'Age_grouping' for visualization
- Feature Engineering: created new features 'Total_Bill' by taking the product of 'Monthly_Bill' and 'Subscription_Length_Months', that might help improve the model's performance.
- Created KDE (Kernel Density Estimation) plots. These plots are data visualization techniques used to estimate the probability density function of a continuous random variable. It provides a smooth curve that represents the distribution of the data, helping to visualize the underlying probability distribution.
    - Created KDE plots on monthly bills and total bills by churn.
    - Surprising insight as higher Churn at lower Total Bill
    - However, if we combine the insights of 3 parameters i.e. Subscription_Length_Months, Monthly bill & Total bill then the picture is bit clear:- Higher Monthly bill at lower Subscription_Length_Months results in lower Total bill. Hence, all these 3 factors viz Higher Monthly bill, Lower Subscription_Length_Months, and Lower Total bill are linked to High Churn.
- Performed correlation_matrix heatmap by dropping Subscription_Length_Months', 'Age_grouping
- Location "Houston" shows the highest non-churners wrt male, whereas location "Los Angeles" shows the highest non-churners wrt female
- Location "Miami" shows the highest churners wrt male, whereas location "New York" shows the highest churners wrt female
- Encoded categorical variables "Gender" and "Location" using label encoder.
    - Label encoding assigns a unique integer to each category in a categorical column

- Dropped columns 'Subscription_Length_Months_grouping', and 'Age_grouping' for building the model.
- Split the data into training and testing sets, using a test size of 0.2

Feature Scaling

StandardScaler is used for feature scaling or standardization, and SMOTE-ENN (Synthetic Minority Over-sampling Technique - Edited Nearest Neighbors) technique for handling class imbalance in your data. SMOTE-ENN combines the oversampling of the minority class using SMOTE with the undersampling of the majority class using ENN

Model Building

1. KNN- used for both classification and regression tasks. Knn Classifier implementing the k-nearest neighbors vote. Without the SMOTE-ENN technique, it showed an accuracy of 0.5. and with SMOTE-ENN it improved its accuracy to 0.69.
   Without SMOTE-ENN:

```
Accuracy: 0.5004

              precision    recall  f1-score   support

           0       0.50      0.50      0.50     10079
           1       0.50      0.50      0.50      9921

    accuracy                           0.50     20000
   macro avg       0.50      0.50      0.50     20000
weighted avg       0.50      0.50      0.50     20000

Confusion_matrix:

[[5075 5004]
 [4988 4933]]
```

With SMOTE-ENN:

```
Accuracy: 0.7088361230050604

Classification Report:

              precision    recall  f1-score   support

           0       0.72      0.69      0.70      1275
           1       0.70      0.73      0.72      1294

    accuracy                           0.71      2569
   macro avg       0.71      0.71      0.71      2569
weighted avg       0.71      0.71      0.71      2569

Confusion_matrix:

[[876 399]
 [349 945]]
```

2. DecisionTree- is a popular supervised machine learning algorithm.

```
Accuracy: 0.5325029194239004

Classification Report:

              precision    recall  f1-score   support

           0       0.56      0.28      0.37      1275
           1       0.52      0.79      0.63      1294

    accuracy                           0.53      2569
   macro avg       0.54      0.53      0.50      2569
weighted avg       0.54      0.53      0.50      2569

Confusion_matrix:

[[ 352  923]
 [ 278 1016]]
```

3. Random Forest- used for both classification and regression tasks. It is an extension of decision trees and combines the predictions of multiple decision trees to improve accuracy and reduce overfitting.

```
Accuracy: 0.5391202802646944

Classification Report:

              precision    recall  f1-score   support

           0       0.56      0.32      0.41      1275
           1       0.53      0.75      0.62      1294

    accuracy                           0.54      2569
   macro avg       0.55      0.54      0.52      2569
weighted avg       0.55      0.54      0.52      2569

Confusion_matrix:

[[410 865]
 [319 975]]
```

Evaluated the model's performance:

1. Accuracy Score (accuracy_score): This function calculates the accuracy of a classification model by comparing the predicted labels to the true labels. It measures the proportion of correctly classified samples.

2. Classification Report (classification_report): This function generates a detailed report that includes metrics such as precision, recall, F1-score, and support for each class in a multi-class classification problem. It's a valuable summary of classification performance.

3. Confusion Matrix (confusion_matrix): The confusion matrix is a table that shows the number of true positives, true negatives, false positives, and false negatives. It provides insight into the model's performance in terms of correct and incorrect classifications.

Model Optimization

Since KNN gave the highest f1 score compared to other models, hence chose the KNN model for Optimization

1. Cross-validation: is a technique used to assess the performance of a model while preventing overfitting. Commonly used methods include k-fold cross-validation.
Results:

```
n_neighbors=3: Mean Accuracy=0.76
n_neighbors=5: Mean Accuracy=0.71
n_neighbors=7: Mean Accuracy=0.67
n_neighbors=9: Mean Accuracy=0.65
n_neighbors=11: Mean Accuracy=0.64
```

2. Grid Search: is a systematic way to search for the best combination of hyperparameters. It combines cross-validation with different hyperparameter values to find the optimal configuration.

Results:

```
Best n_neighbors: 3
Best Accuracy: 0.76
```

After identifying the best hyperparameter values, created a KNN model with those values and evaluated it on the test set to assess its performance

Results:

```
Accuracy: 0.7613857532113663

Classification Report:

              precision    recall  f1-score   support

           0       0.77      0.75      0.76      1275
           1       0.76      0.78      0.77      1294

    accuracy                           0.76      2569
   macro avg       0.76      0.76      0.76      2569
weighted avg       0.76      0.76      0.76      2569

Confusion_matrix:

[[ 952  323]
 [ 290 1004]]
```

Model Deployment

Performed Model Serialization by saving the trained KNN model to disk.

Conclusion

In this comprehensive analysis, we explored a dataset of 100,000 customer records to predict churn effectively. Through extensive data preprocessing, feature engineering, and visualization, we uncovered valuable insights, including the impact of factors like 'Monthly_Bill,' 'Subscription_Length_Months,' and 'Location' on churn. Leveraging machine learning, we developed and optimized a K-Nearest Neighbors (KNN) model with SMOTE-ENN to address class imbalance, achieving high accuracy and F1-score. This predictive model, serialized for deployment, holds great potential for real-time customer churn predictions. As a result, businesses can now proactively target at-risk customers and implement retention strategies, ultimately fostering customer loyalty and business growth