Problem Statement: The telecom industry faces the challenge of customer churn, where customers cancel or do not renew their subscriptions, leading to potential revenue loss. Identifying potential churn customers is vital for the company's growth and customer retention.

Aim: The aim of this analysis is to develop predictive models to classify potential churn customers based on numerical and categorical features. The analysis involves binary classification, where customers are categorized as likely to churn (Yes) or not (No).

Importance of Churn Analysis: Churn analysis provides valuable insights into customer behavior and preferences. It helps companies build effective strategies to retain customers, enhance service quality, and conduct targeted marketing campaigns.

Strategy: The analysis focuses on building accurate predictive models that address class imbalance to make informed decisions about potential churn customers.

Benefits: The analysis empowers telecom companies to optimize customer retention efforts, reduce churn rates, and cultivate trust with customers.

Practical Use: The outcomes of churn analysis aid companies in developing customer-centric approaches, improving customer satisfaction, and ultimately driving business growth.

```
import os
import warnings
warnings.filterwarnings('ignore')
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

import os: This line imports the 'os' module, which provides a way of using operating system-dependent functionality.

import warnings: This line imports the 'warnings' module, allowing us to handle or suppress warning messages during the code execution.

warnings.filterwarnings('ignore'): This line sets a filter to ignore warning messages. It's a good practice to suppress warnings that might not be critical to our analysis.

import numpy as np: This line imports the 'numpy' library and assigns it the alias 'np'. 'numpy' is a powerful library for numerical computing and handling arrays and matrices.

import pandas as pd: This line imports the 'pandas' library and assigns it the alias 'pd'. 'pandas' is widely used for data manipulation and analysis, providing powerful data structures like DataFrames.

import matplotlib.pyplot as plt: This line imports the 'matplotlib.pyplot' module and assigns it the alias 'plt'. 'matplotlib' is a popular plotting library in Python, and 'pyplot' provides a MATLAB-like interface for creating visualizations.

import seaborn as sns: This line imports the 'seaborn' library and assigns it the alias 'sns'. 'seaborn' is another data visualization library built on top of 'matplotlib', offering additional high-level plotting functions and attractive styles.

```
dataset= pd.read_csv('/content/Telcom Data.csv')
```

Data Loading The above code reads the telecom dataset from a CSV (Comma Separated Values) file named "Telcom Data.csv" located in the '/content' directory. It uses the 'pd.read_csv()' function provided by the 'pandas' library to load the data and store it in a DataFrame called 'dataset'.

```
dataset.head()
```

```
   customerID  gender  SeniorCitizen Partner Dependents  tenure
PhoneService  \
0  7590-VHVEG  Female              0     Yes         No       1
No
1  5575-GNVDE    Male              0      No         No      34
Yes
2  3668-QPYBK    Male              0      No         No       2
Yes
3  7795-CFOCW    Male              0      No         No      45
No
4  9237-HQITU  Female              0      No         No       2
Yes

      MultipleLines InternetService OnlineSecurity  ...
DeviceProtection  \
0  No phone service             DSL             No  ...
No
1                No             DSL            Yes  ...
Yes
2                No             DSL            Yes  ...
No
3  No phone service             DSL            Yes  ...
Yes
4                No     Fiber optic             No  ...
No
```

```
  TechSupport StreamingTV StreamingMovies        Contract
PaperlessBilling  \
0          No          No              No  Month-to-month
Yes
1          No          No              No        One year
No
2          No          No              No  Month-to-month
Yes
3         Yes          No              No        One year
No
4          No          No              No  Month-to-month
Yes

               PaymentMethod MonthlyCharges  TotalCharges Churn
0           Electronic check          29.85         29.85    No
1               Mailed check          56.95        1889.5    No
2               Mailed check          53.85        108.15   Yes
3  Bank transfer (automatic)          42.30       1840.75    No
4           Electronic check          70.70        151.65   Yes

[5 rows x 21 columns]
```

The dataset contains information about telecom customers and various attributes related to their interactions and services with the company. Here is a more detailed description of each attribute:

customerID: A unique identifier for each customer.

gender: Indicates whether the customer is male or female.

SeniorCitizen: A binary indicator (1 or 0) that specifies whether the customer is a senior citizen (1: Yes, 0: No).

Partner: A binary indicator (Yes or No) representing whether the customer has a partner.

Dependents: A binary indicator (Yes or No) showing whether the customer has dependents (e.g., family members).

tenure: The number of months the customer has stayed with the company.

PhoneService: A binary indicator (Yes or No) indicating whether the customer has a phone service.

MultipleLines: Indicates whether the customer has multiple phone lines (Yes, No, or No phone service).

InternetService: Specifies the customer's internet service provider (DSL, Fiber optic, or No internet service).

OnlineSecurity: A binary indicator (Yes or No) showing whether the customer has online security service or not (or No internet service).

OnlineBackup: A binary indicator (Yes or No) showing whether the customer has online backup service or not (or No internet service).

DeviceProtection: A binary indicator (Yes or No) showing whether the customer has device protection service or not (or No internet service).

TechSupport: A binary indicator (Yes or No) indicating whether the customer has tech support service or not (or No internet service).

StreamingTV: A binary indicator (Yes or No) showing whether the customer has streaming TV service or not (or No internet service).

StreamingMovies: A binary indicator (Yes or No) showing whether the customer has streaming movie service or not (or No internet service).

Contract: The contract term of the customer (Month-to-month, One year, Two years).

PaperlessBilling: A binary indicator (Yes or No) showing whether the customer has opted for paperless billing.

PaymentMethod: Specifies the customer's payment method (Electronic check, Mailed check, Bank transfer (automatic), Credit card (automatic)).

MonthlyCharges: The amount charged to the customer on a monthly basis.

TotalCharges: The total amount charged to the customer over their entire tenure.

Churn: The target variable that indicates whether the customer has churned (left the telecom service) or not (Yes or No).

The dataset is relevant for conducting customer churn analysis, as it contains valuable information about customer behavior and characteristics that can be used to predict whether a customer is likely to churn or not. By utilizing machine learning algorithms and techniques, telecom companies can gain insights into customer retention strategies and improve overall customer satisfaction.

```
dataset.shape
```

```
(7043, 21)
```

```
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7043 entries, 0 to 7042
Data columns (total 21 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   customerID        7043 non-null   object
 1   gender            7043 non-null   object
 2   SeniorCitizen     7043 non-null   int64
 3   Partner           7043 non-null   object
 4   Dependents        7043 non-null   object
 5   tenure            7043 non-null   int64
```

```
 6   PhoneService      7043 non-null   object
 7   MultipleLines     7043 non-null   object
 8   InternetService   7043 non-null   object
 9   OnlineSecurity    7043 non-null   object
 10  OnlineBackup      7043 non-null   object
 11  DeviceProtection  7043 non-null   object
 12  TechSupport       7043 non-null   object
 13  StreamingTV       7043 non-null   object
 14  StreamingMovies   7043 non-null   object
 15  Contract          7043 non-null   object
 16  PaperlessBilling  7043 non-null   object
 17  PaymentMethod     7043 non-null   object
 18  MonthlyCharges    7043 non-null   float64
 19  TotalCharges      7043 non-null   object
 20  Churn             7043 non-null   object
dtypes: float64(1), int64(2), object(18)
memory usage: 1.1+ MB
```

The code dataset.info() provides information about the DataFrame 'dataset', which contains telecom customer data. It gives an overview of the columns in the DataFrame, the number of non-null values in each column, and the data types of each column.

The 'object' data type represents strings or text, while 'int64' and 'float64' represent integer and floating-point numeric values, respectively.

It is important to check the data types as they can influence data processing and analysis. For instance, 'TotalCharges' is expected to be a numeric column, but it is currently identified as 'object'. This suggests that there might be some non-numeric values in the 'TotalCharges' column. To handle this issue, the column may need to be converted to a numeric data type.

```
dataset['Churn'].value_counts()
```

```
No     5174
Yes    1869
Name: Churn, dtype: int64
```

The code dataset['Churn'].value_counts() is used to count the number of occurrences of each unique value in the 'Churn' column of the DataFrame 'dataset'. The 'Churn' column represents whether a customer churned (canceled or did not renew their subscription) or not.

'No': This value appears 5174 times in the 'Churn' column, indicating that there are 5174 customers who did not churn.

'Yes': This value appears 1869 times in the 'Churn' column, indicating that there are 1869 customers who churned. The value_counts() function is a convenient way to quickly understand the distribution of values in a categorical column like 'Churn'. This information is useful for gaining insights into the churn rate and understanding the proportion of customers who have churned compared to those who have not.

```
dataset['Churn']= dataset['Churn'].replace({'Yes':1, 'No':0})
```

By using the replace() function with the argument {'Yes': 1, 'No': 0}, we are mapping 'Yes' to 1 and 'No' to 0, converting the column to a binary representation. This transformation is essential for binary classification tasks, as many machine learning algorithms require numeric input features and target labels.

```
dataset.columns
```

```
Index(['customerID', 'gender', 'SeniorCitizen', 'Partner',
'Dependents',
       'tenure', 'PhoneService', 'MultipleLines', 'InternetService',
       'OnlineSecurity', 'OnlineBackup', 'DeviceProtection',
'TechSupport',
       'StreamingTV', 'StreamingMovies', 'Contract',
'PaperlessBilling',
       'PaymentMethod', 'MonthlyCharges', 'TotalCharges', 'Churn'],
      dtype='object')
```

```
for i in dataset.columns:
  print("****************************************", i,
        "****************************************")
  print()
  print(set(dataset[i].tolist()))
  print()
```

The code iterates over each column in the DataFrame 'dataset' and prints information about each column, including the column name, unique values present in the column, and the data type of the column.

The code provided is helpful because it allows you to quickly understand the unique values present in each column of the dataset. It provides insights into the nature of the data in each column, especially for categorical columns, which is crucial for data exploration and analysis. By printing the unique values for each column, you can gain a better understanding of the different categories or levels in the dataset.

Using AutoML Pycaret

PyCaret provides a high-level API that allows users to perform complex machine learning tasks with just a few lines of code. It automates many of the repetitive tasks involved in building and evaluating machine learning models, making it easier for data scientists and machine learning practitioners to experiment with different models and techniques.

```
!pip install pycaret
```

```
Collecting pycaret
  Downloading pycaret-3.0.4-py3-none-any.whl (484 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 484.4/484.4 kB 8.3 MB/s eta
0:00:00
ent already satisfied: ipython>=5.5.0 in
```

```
/usr/local/lib/python3.10/dist-packages (from pycaret) (7.34.0)
Requirement already satisfied: ipywidgets>=7.6.5 in
/usr/local/lib/python3.10/dist-packages (from pycaret) (7.7.1)
Requirement already satisfied: tqdm>=4.62.0 in
/usr/local/lib/python3.10/dist-packages (from pycaret) (4.65.0)
Requirement already satisfied: numpy<1.24,>=1.21 in
/usr/local/lib/python3.10/dist-packages (from pycaret) (1.22.4)
Requirement already satisfied: pandas<2.0.0,>=1.3.0 in
/usr/local/lib/python3.10/dist-packages (from pycaret) (1.5.3)
Requirement already satisfied: jinja2>=1.2 in
/usr/local/lib/python3.10/dist-packages (from pycaret) (3.1.2)
Requirement already satisfied: scipy<2.0.0 in
/usr/local/lib/python3.10/dist-packages (from pycaret) (1.10.1)
Requirement already satisfied: joblib>=1.2.0 in
/usr/local/lib/python3.10/dist-packages (from pycaret) (1.3.1)
Requirement already satisfied: scikit-learn<1.3.0,>=1.0 in
/usr/local/lib/python3.10/dist-packages (from pycaret) (1.2.2)
Collecting pyod>=1.0.8 (from pycaret)
  Downloading pyod-1.1.0.tar.gz (153 kB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 153.4/153.4 kB 20.7 MB/s eta
0:00:00
etadata (setup.py) ... ent already satisfied: imbalanced-learn>=0.8.1
in /usr/local/lib/python3.10/dist-packages (from pycaret) (0.10.1)
Collecting category-encoders>=2.4.0 (from pycaret)
  Downloading category_encoders-2.6.1-py2.py3-none-any.whl (81 kB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 81.9/81.9 kB 10.8 MB/s eta
0:00:00
ent already satisfied: lightgbm>=3.0.0 in
/usr/local/lib/python3.10/dist-packages (from pycaret) (3.3.5)
Requirement already satisfied: numba>=0.55.0 in
/usr/local/lib/python3.10/dist-packages (from pycaret) (0.56.4)
Requirement already satisfied: requests>=2.27.1 in
/usr/local/lib/python3.10/dist-packages (from pycaret) (2.27.1)
Requirement already satisfied: psutil>=5.9.0 in
/usr/local/lib/python3.10/dist-packages (from pycaret) (5.9.5)
Requirement already satisfied: markupsafe>=2.0.1 in
/usr/local/lib/python3.10/dist-packages (from pycaret) (2.1.3)
Collecting importlib-metadata>=4.12.0 (from pycaret)
  Downloading importlib_metadata-6.8.0-py3-none-any.whl (22 kB)
Requirement already satisfied: nbformat>=4.2.0 in
/usr/local/lib/python3.10/dist-packages (from pycaret) (5.9.1)
Requirement already satisfied: cloudpickle in
/usr/local/lib/python3.10/dist-packages (from pycaret) (2.2.1)
Collecting deprecation>=2.1.0 (from pycaret)
  Downloading deprecation-2.1.0-py2.py3-none-any.whl (11 kB)
Collecting xxhash (from pycaret)
  Downloading xxhash-3.3.0-cp310-cp310-
manylinux_2_17_x86_64.manylinux2014_x86_64.whl (194 kB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 194.1/194.1 kB 23.2 MB/s eta
```

```
0:00:00
ent already satisfied: matplotlib>=3.3.0 in
/usr/local/lib/python3.10/dist-packages (from pycaret) (3.7.1)
Collecting scikit-plot>=0.3.7 (from pycaret)
  Downloading scikit_plot-0.3.7-py3-none-any.whl (33 kB)
Requirement already satisfied: yellowbrick>=1.4 in
/usr/local/lib/python3.10/dist-packages (from pycaret) (1.5)
Requirement already satisfied: plotly>=5.0.0 in
/usr/local/lib/python3.10/dist-packages (from pycaret) (5.13.1)
Collecting kaleido>=0.2.1 (from pycaret)
  Downloading kaleido-0.2.1-py2.py3-none-manylinux1_x86_64.whl (79.9
MB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 79.9/79.9 MB 11.1 MB/s eta
0:00:00
draw==0.15 (from pycaret)
  Downloading schemdraw-0.15-py3-none-any.whl (106 kB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 106.8/106.8 kB 14.8 MB/s eta
0:00:00
pler>=0.8.3.1 (from pycaret)
  Downloading plotly_resampler-0.9.1-py3-none-any.whl (73 kB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 73.4/73.4 kB 10.1 MB/s eta
0:00:00
ent already satisfied: statsmodels>=0.12.1 in
/usr/local/lib/python3.10/dist-packages (from pycaret) (0.13.5)
Collecting sktime!=0.17.1,!=0.17.2,!=0.18.0,>=0.16.1 (from pycaret)
  Downloading sktime-0.21.0-py3-none-any.whl (17.1 MB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 17.1/17.1 MB 52.1 MB/s eta
0:00:00
 pycaret)
  Downloading tbats-1.1.3-py3-none-any.whl (44 kB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 44.0/44.0 kB 6.0 MB/s eta
0:00:00
darima!=1.8.1,<3.0.0,>=1.8.0 (from pycaret)
  Downloading pmdarima-2.0.3-cp310-cp310-
manylinux_2_17_x86_64.manylinux2014_x86_64.manylinux_2_28_x86_64.whl
(1.8 MB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.8/1.8 MB 56.8 MB/s eta
0:00:00
 pycaret)
  Downloading wurlitzer-3.0.3-py3-none-any.whl (7.3 kB)
Requirement already satisfied: patsy>=0.5.1 in
/usr/local/lib/python3.10/dist-packages (from category-
encoders>=2.4.0->pycaret) (0.5.3)
Requirement already satisfied: packaging in
/usr/local/lib/python3.10/dist-packages (from deprecation>=2.1.0-
>pycaret) (23.1)
Requirement already satisfied: threadpoolctl>=2.0.0 in
/usr/local/lib/python3.10/dist-packages (from imbalanced-learn>=0.8.1-
>pycaret) (3.2.0)
```

```
Requirement already satisfied: zipp>=0.5 in
/usr/local/lib/python3.10/dist-packages (from importlib-
metadata>=4.12.0->pycaret) (3.16.2)
Requirement already satisfied: setuptools>=18.5 in
/usr/local/lib/python3.10/dist-packages (from ipython>=5.5.0->pycaret)
(67.7.2)
Collecting jedi>=0.16 (from ipython>=5.5.0->pycaret)
  Downloading jedi-0.19.0-py2.py3-none-any.whl (1.6 MB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.6/1.6 MB 91.0 MB/s eta
0:00:00
ent already satisfied: decorator in /usr/local/lib/python3.10/dist-
packages (from ipython>=5.5.0->pycaret) (4.4.2)
Requirement already satisfied: pickleshare in
/usr/local/lib/python3.10/dist-packages (from ipython>=5.5.0->pycaret)
(0.7.5)
Requirement already satisfied: traitlets>=4.2 in
/usr/local/lib/python3.10/dist-packages (from ipython>=5.5.0->pycaret)
(5.7.1)
Requirement already satisfied: prompt-toolkit!=3.0.0,!
=3.0.1,<3.1.0,>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from
ipython>=5.5.0->pycaret) (3.0.39)
Requirement already satisfied: pygments in
/usr/local/lib/python3.10/dist-packages (from ipython>=5.5.0->pycaret)
(2.14.0)
Requirement already satisfied: backcall in
/usr/local/lib/python3.10/dist-packages (from ipython>=5.5.0->pycaret)
(0.2.0)
Requirement already satisfied: matplotlib-inline in
/usr/local/lib/python3.10/dist-packages (from ipython>=5.5.0->pycaret)
(0.1.6)
Requirement already satisfied: pexpect>4.3 in
/usr/local/lib/python3.10/dist-packages (from ipython>=5.5.0->pycaret)
(4.8.0)
Requirement already satisfied: ipykernel>=4.5.1 in
/usr/local/lib/python3.10/dist-packages (from ipywidgets>=7.6.5-
>pycaret) (5.5.6)
Requirement already satisfied: ipython-genutils~=0.2.0 in
/usr/local/lib/python3.10/dist-packages (from ipywidgets>=7.6.5-
>pycaret) (0.2.0)
Requirement already satisfied: widgetsnbextension~=3.6.0 in
/usr/local/lib/python3.10/dist-packages (from ipywidgets>=7.6.5-
>pycaret) (3.6.4)
Requirement already satisfied: jupyterlab-widgets>=1.0.0 in
/usr/local/lib/python3.10/dist-packages (from ipywidgets>=7.6.5-
>pycaret) (3.0.8)
Requirement already satisfied: wheel in
/usr/local/lib/python3.10/dist-packages (from lightgbm>=3.0.0-
>pycaret) (0.41.0)
Requirement already satisfied: contourpy>=1.0.1 in
```

```
/usr/local/lib/python3.10/dist-packages (from matplotlib>=3.3.0-
>pycaret) (1.1.0)
Requirement already satisfied: cycler>=0.10 in
/usr/local/lib/python3.10/dist-packages (from matplotlib>=3.3.0-
>pycaret) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in
/usr/local/lib/python3.10/dist-packages (from matplotlib>=3.3.0-
>pycaret) (4.41.1)
Requirement already satisfied: kiwisolver>=1.0.1 in
/usr/local/lib/python3.10/dist-packages (from matplotlib>=3.3.0-
>pycaret) (1.4.4)
Requirement already satisfied: pillow>=6.2.0 in
/usr/local/lib/python3.10/dist-packages (from matplotlib>=3.3.0-
>pycaret) (9.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in
/usr/local/lib/python3.10/dist-packages (from matplotlib>=3.3.0-
>pycaret) (3.1.0)
Requirement already satisfied: python-dateutil>=2.7 in
/usr/local/lib/python3.10/dist-packages (from matplotlib>=3.3.0-
>pycaret) (2.8.2)
Requirement already satisfied: fastjsonschema in
/usr/local/lib/python3.10/dist-packages (from nbformat>=4.2.0-
>pycaret) (2.18.0)
Requirement already satisfied: jsonschema>=2.6 in
/usr/local/lib/python3.10/dist-packages (from nbformat>=4.2.0-
>pycaret) (4.3.3)
Requirement already satisfied: jupyter-core in
/usr/local/lib/python3.10/dist-packages (from nbformat>=4.2.0-
>pycaret) (5.3.1)
Requirement already satisfied: llvmlite<0.40,>=0.39.0dev0 in
/usr/local/lib/python3.10/dist-packages (from numba>=0.55.0->pycaret)
(0.39.1)
Requirement already satisfied: pytz>=2020.1 in
/usr/local/lib/python3.10/dist-packages (from pandas<2.0.0,>=1.3.0-
>pycaret) (2022.7.1)
Requirement already satisfied: tenacity>=6.2.0 in
/usr/local/lib/python3.10/dist-packages (from plotly>=5.0.0->pycaret)
(8.2.2)
Collecting dash<3.0.0,>=2.11.0 (from plotly-resampler>=0.8.3.1-
>pycaret)
  Downloading dash-2.11.1-py3-none-any.whl (10.4 MB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 10.4/10.4 MB 105.3 MB/s eta
0:00:00
 plotly-resampler>=0.8.3.1->pycaret)
  Downloading orjson-3.9.2-cp310-cp310-
manylinux_2_17_x86_64.manylinux2014_x86_64.whl (138 kB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 138.7/138.7 kB 16.3 MB/s eta
0:00:00
 plotly-resampler>=0.8.3.1->pycaret)
```

```
  Downloading trace_updater-0.0.9.1-py3-none-any.whl (185 kB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 185.2/185.2 kB 20.8 MB/s eta
0:00:00
ple==0.1.2 (from plotly-resampler>=0.8.3.1->pycaret)
  Downloading tsdownsample-0.1.2-cp310-cp310-
manylinux_2_17_x86_64.manylinux2014_x86_64.whl (2.3 MB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2.3/2.3 MB 75.6 MB/s eta
0:00:00
ent already satisfied: Cython!=0.29.18,!=0.29.31,>=0.29 in
/usr/local/lib/python3.10/dist-packages (from pmdarima!
=1.8.1,<3.0.0,>=1.8.0->pycaret) (0.29.36)
Requirement already satisfied: urllib3 in
/usr/local/lib/python3.10/dist-packages (from pmdarima!
=1.8.1,<3.0.0,>=1.8.0->pycaret) (1.26.16)
Requirement already satisfied: six in /usr/local/lib/python3.10/dist-
packages (from pyod>=1.0.8->pycaret) (1.16.0)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.10/dist-packages (from requests>=2.27.1-
>pycaret) (2023.7.22)
Requirement already satisfied: charset-normalizer~=2.0.0 in
/usr/local/lib/python3.10/dist-packages (from requests>=2.27.1-
>pycaret) (2.0.12)
Requirement already satisfied: idna<4,>=2.5 in
/usr/local/lib/python3.10/dist-packages (from requests>=2.27.1-
>pycaret) (3.4)
Collecting deprecated>=1.2.13 (from sktime!=0.17.1,!=0.17.2,!
=0.18.0,>=0.16.1->pycaret)
  Downloading Deprecated-1.2.14-py2.py3-none-any.whl (9.6 kB)
Collecting scikit-base<0.6.0 (from sktime!=0.17.1,!=0.17.2,!
=0.18.0,>=0.16.1->pycaret)
  Downloading scikit_base-0.5.0-py3-none-any.whl (118 kB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 118.2/118.2 kB 13.7 MB/s eta
0:00:00
ent already satisfied: Flask<2.3.0,>=1.0.4 in
/usr/local/lib/python3.10/dist-packages (from dash<3.0.0,>=2.11.0-
>plotly-resampler>=0.8.3.1->pycaret) (2.2.5)
Collecting Werkzeug<2.3.0 (from dash<3.0.0,>=2.11.0->plotly-
resampler>=0.8.3.1->pycaret)
  Downloading Werkzeug-2.2.3-py3-none-any.whl (233 kB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 233.6/233.6 kB 26.1 MB/s eta
0:00:00
l-components==2.0.0 (from dash<3.0.0,>=2.11.0->plotly-
resampler>=0.8.3.1->pycaret)
  Downloading dash_html_components-2.0.0-py3-none-any.whl (4.1 kB)
Collecting dash-core-components==2.0.0 (from dash<3.0.0,>=2.11.0-
>plotly-resampler>=0.8.3.1->pycaret)
  Downloading dash_core_components-2.0.0-py3-none-any.whl (3.8 kB)
Collecting dash-table==5.0.0 (from dash<3.0.0,>=2.11.0->plotly-
resampler>=0.8.3.1->pycaret)
```

```
  Downloading dash_table-5.0.0-py3-none-any.whl (3.9 kB)
Requirement already satisfied: typing-extensions>=4.1.1 in
/usr/local/lib/python3.10/dist-packages (from dash<3.0.0,>=2.11.0-
>plotly-resampler>=0.8.3.1->pycaret) (4.7.1)
Collecting retrying (from dash<3.0.0,>=2.11.0->plotly-
resampler>=0.8.3.1->pycaret)
  Downloading retrying-1.3.4-py3-none-any.whl (11 kB)
Collecting ansi2html (from dash<3.0.0,>=2.11.0->plotly-
resampler>=0.8.3.1->pycaret)
  Downloading ansi2html-1.8.0-py3-none-any.whl (16 kB)
Requirement already satisfied: nest-asyncio in
/usr/local/lib/python3.10/dist-packages (from dash<3.0.0,>=2.11.0-
>plotly-resampler>=0.8.3.1->pycaret) (1.5.6)
Requirement already satisfied: wrapt<2,>=1.10 in
/usr/local/lib/python3.10/dist-packages (from deprecated>=1.2.13-
>sktime!=0.17.1,!=0.17.2,!=0.18.0,>=0.16.1->pycaret) (1.14.1)
Requirement already satisfied: jupyter-client in
/usr/local/lib/python3.10/dist-packages (from ipykernel>=4.5.1-
>ipywidgets>=7.6.5->pycaret) (6.1.12)
Requirement already satisfied: tornado>=4.2 in
/usr/local/lib/python3.10/dist-packages (from ipykernel>=4.5.1-
>ipywidgets>=7.6.5->pycaret) (6.3.1)
Requirement already satisfied: parso<0.9.0,>=0.8.3 in
/usr/local/lib/python3.10/dist-packages (from jedi>=0.16-
>ipython>=5.5.0->pycaret) (0.8.3)
Requirement already satisfied: attrs>=17.4.0 in
/usr/local/lib/python3.10/dist-packages (from jsonschema>=2.6-
>nbformat>=4.2.0->pycaret) (23.1.0)
Requirement already satisfied: pyrsistent!=0.17.0,!=0.17.1,!
=0.17.2,>=0.14.0 in /usr/local/lib/python3.10/dist-packages (from
jsonschema>=2.6->nbformat>=4.2.0->pycaret) (0.19.3)
Requirement already satisfied: ptyprocess>=0.5 in
/usr/local/lib/python3.10/dist-packages (from pexpect>4.3-
>ipython>=5.5.0->pycaret) (0.7.0)
Requirement already satisfied: wcwidth in
/usr/local/lib/python3.10/dist-packages (from prompt-toolkit!=3.0.0,!
=3.0.1,<3.1.0,>=2.0.0->ipython>=5.5.0->pycaret) (0.2.6)
Requirement already satisfied: notebook>=4.4.1 in
/usr/local/lib/python3.10/dist-packages (from
widgetsnbextension~=3.6.0->ipywidgets>=7.6.5->pycaret) (6.4.8)
Requirement already satisfied: platformdirs>=2.5 in
/usr/local/lib/python3.10/dist-packages (from jupyter-core-
>nbformat>=4.2.0->pycaret) (3.9.1)
Requirement already satisfied: itsdangerous>=2.0 in
/usr/local/lib/python3.10/dist-packages (from Flask<2.3.0,>=1.0.4-
>dash<3.0.0,>=2.11.0->plotly-resampler>=0.8.3.1->pycaret) (2.1.2)
Requirement already satisfied: click>=8.0 in
/usr/local/lib/python3.10/dist-packages (from Flask<2.3.0,>=1.0.4-
>dash<3.0.0,>=2.11.0->plotly-resampler>=0.8.3.1->pycaret) (8.1.6)
```

```
Requirement already satisfied: pyzmq>=17 in
/usr/local/lib/python3.10/dist-packages (from notebook>=4.4.1-
>widgetsnbextension~=3.6.0->ipywidgets>=7.6.5->pycaret) (23.2.1)
Requirement already satisfied: argon2-cffi in
/usr/local/lib/python3.10/dist-packages (from notebook>=4.4.1-
>widgetsnbextension~=3.6.0->ipywidgets>=7.6.5->pycaret) (21.3.0)
Requirement already satisfied: nbconvert in
/usr/local/lib/python3.10/dist-packages (from notebook>=4.4.1-
>widgetsnbextension~=3.6.0->ipywidgets>=7.6.5->pycaret) (6.5.4)
Requirement already satisfied: Send2Trash>=1.8.0 in
/usr/local/lib/python3.10/dist-packages (from notebook>=4.4.1-
>widgetsnbextension~=3.6.0->ipywidgets>=7.6.5->pycaret) (1.8.2)
Requirement already satisfied: terminado>=0.8.3 in
/usr/local/lib/python3.10/dist-packages (from notebook>=4.4.1-
>widgetsnbextension~=3.6.0->ipywidgets>=7.6.5->pycaret) (0.17.1)
Requirement already satisfied: prometheus-client in
/usr/local/lib/python3.10/dist-packages (from notebook>=4.4.1-
>widgetsnbextension~=3.6.0->ipywidgets>=7.6.5->pycaret) (0.17.1)
Requirement already satisfied: argon2-cffi-bindings in
/usr/local/lib/python3.10/dist-packages (from argon2-cffi-
>notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.6.5-
>pycaret) (21.2.0)
Requirement already satisfied: lxml in /usr/local/lib/python3.10/dist-
packages (from nbconvert->notebook>=4.4.1->widgetsnbextension~=3.6.0-
>ipywidgets>=7.6.5->pycaret) (4.9.3)
Requirement already satisfied: beautifulsoup4 in
/usr/local/lib/python3.10/dist-packages (from nbconvert-
>notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.6.5-
>pycaret) (4.11.2)
Requirement already satisfied: bleach in
/usr/local/lib/python3.10/dist-packages (from nbconvert-
>notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.6.5-
>pycaret) (6.0.0)
Requirement already satisfied: defusedxml in
/usr/local/lib/python3.10/dist-packages (from nbconvert-
>notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.6.5-
>pycaret) (0.7.1)
Requirement already satisfied: entrypoints>=0.2.2 in
/usr/local/lib/python3.10/dist-packages (from nbconvert-
>notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.6.5-
>pycaret) (0.4)
Requirement already satisfied: jupyterlab-pygments in
/usr/local/lib/python3.10/dist-packages (from nbconvert-
>notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.6.5-
>pycaret) (0.2.2)
Requirement already satisfied: mistune<2,>=0.8.1 in
/usr/local/lib/python3.10/dist-packages (from nbconvert-
>notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.6.5-
>pycaret) (0.8.4)
```

```
Requirement already satisfied: nbclient>=0.5.0 in
/usr/local/lib/python3.10/dist-packages (from nbconvert-
>notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.6.5-
>pycaret) (0.8.0)
Requirement already satisfied: pandocfilters>=1.4.1 in
/usr/local/lib/python3.10/dist-packages (from nbconvert-
>notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.6.5-
>pycaret) (1.5.0)
Requirement already satisfied: tinycss2 in
/usr/local/lib/python3.10/dist-packages (from nbconvert-
>notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.6.5-
>pycaret) (1.2.1)
Requirement already satisfied: cffi>=1.0.1 in
/usr/local/lib/python3.10/dist-packages (from argon2-cffi-bindings-
>argon2-cffi->notebook>=4.4.1->widgetsnbextension~=3.6.0-
>ipywidgets>=7.6.5->pycaret) (1.15.1)
Requirement already satisfied: soupsieve>1.2 in
/usr/local/lib/python3.10/dist-packages (from beautifulsoup4-
>nbconvert->notebook>=4.4.1->widgetsnbextension~=3.6.0-
>ipywidgets>=7.6.5->pycaret) (2.4.1)
Requirement already satisfied: webencodings in
/usr/local/lib/python3.10/dist-packages (from bleach->nbconvert-
>notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.6.5-
>pycaret) (0.5.1)
Requirement already satisfied: pycparser in
/usr/local/lib/python3.10/dist-packages (from cffi>=1.0.1->argon2-
cffi-bindings->argon2-cffi->notebook>=4.4.1-
>widgetsnbextension~=3.6.0->ipywidgets>=7.6.5->pycaret) (2.21)
Building wheels for collected packages: pyod
  Building wheel for pyod (setup.py) ... e=pyod-1.1.0-py3-none-any.whl
size=185330
sha256=f3bf183f051e4e73152a21c30338d5d277000bbc8207886662b7d03b8f4d8b7
a
  Stored in directory:
/root/.cache/pip/wheels/36/8e/e2/e932956b10b843eb6be9eefa70b5c1bee7b56
1be14c423b136
Successfully built pyod
Installing collected packages: trace-updater, kaleido, dash-table,
dash-html-components, dash-core-components, xxhash, wurlitzer,
Werkzeug, tsdownsample, scikit-base, schemdraw, retrying, orjson,
jedi, importlib-metadata, deprecation, deprecated, ansi2html, sktime,
scikit-plot, pyod, dash, pmdarima, plotly-resampler, category-
encoders, tbats, pycaret
  Attempting uninstall: Werkzeug
    Found existing installation: Werkzeug 2.3.6
    Uninstalling Werkzeug-2.3.6:
      Successfully uninstalled Werkzeug-2.3.6
  Attempting uninstall: importlib-metadata
    Found existing installation: importlib-metadata 4.6.4
```

```
    Uninstalling importlib-metadata-4.6.4:
      Successfully uninstalled importlib-metadata-4.6.4
Successfully installed Werkzeug-2.2.3 ansi2html-1.8.0 category-
encoders-2.6.1 dash-2.11.1 dash-core-components-2.0.0 dash-html-
components-2.0.0 dash-table-5.0.0 deprecated-1.2.14 deprecation-2.1.0
importlib-metadata-6.8.0 jedi-0.19.0 kaleido-0.2.1 orjson-3.9.2
plotly-resampler-0.9.1 pmdarima-2.0.3 pycaret-3.0.4 pyod-1.1.0
retrying-1.3.4 schemdraw-0.15 scikit-base-0.5.0 scikit-plot-0.3.7
sktime-0.21.0 tbats-1.1.3 trace-updater-0.0.9.1 tsdownsample-0.1.2
wurlitzer-3.0.3 xxhash-3.3.0
```

```
from pycaret.classification import *

exp_clf = setup(data=dataset, target= 'Churn', pca= True ,
pca_components=0.95 , session_id=123)
```

```
<pandas.io.formats.style.Styler at 0x7877b27193c0>
```

In the given code, we use the setup() function from PyCaret to set up the machine learning environment for the churn analysis problem using the telecom dataset (dataset). Here's what each parameter does:

data: The input dataset that contains the features and the target variable (Churn in this case).

target: The name of the target variable to be predicted, which is "Churn" in this dataset.

pca: A boolean parameter indicating whether Principal Component Analysis (PCA) should be applied to the data. In this case, it is set to True, which means PCA will be used to reduce the dimensionality of the features while retaining 95% of the variance.

pca_components: The desired percentage of variance to be preserved after PCA is applied. In this case, it is set to 0.95, meaning that the top principal components will be retained to represent 95% of the total variance in the data.

session_id: An optional parameter that sets a random seed for reproducibility. It is set to 123 in this case. By using setup(), PyCaret automatically performs various preprocessing steps, such as handling missing values, encoding categorical variables, splitting the data into training and testing sets, and applying PCA to reduce the feature dimensions. Additionally, it performs automatic feature selection and engineering based on the data and target variable.

After calling setup(), the machine learning environment is ready, and we can proceed with training and evaluating models using PyCaret's simple and efficient API.

```
#comparing models

compare_models()
```

```
<IPython.core.display.HTML object>
```

```
<pandas.io.formats.style.Styler at 0x78782f864dc0>
```

{"model_id":"235a24c2dd2243e08c4a24c5aff7f356","version_major":2,"version_minor":0}

```
<IPython.core.display.HTML object>

GradientBoostingClassifier(ccp_alpha=0.0, criterion='friedman_mse',
init=None,
                           learning_rate=0.1, loss='log_loss',
max_depth=3,
                           max_features=None, max_leaf_nodes=None,
                           min_impurity_decrease=0.0,
min_samples_leaf=1,
                           min_samples_split=2,
min_weight_fraction_leaf=0.0,
                           n_estimators=100, n_iter_no_change=None,
                           random_state=123, subsample=1.0,
tol=0.0001,
                           validation_fraction=0.1, verbose=0,
                           warm_start=False)
```

We use the compare_models() function from PyCaret to automatically train and evaluate multiple classification models on the dataset. This function helps us quickly identify the top-performing models based on default hyperparameters and performance metrics.

Here's what the compare_models() function does:

Training Multiple Models: The compare_models() function trains several classification models on the training data. The models used for comparison include popular algorithms such as Decision Tree, Random Forest, Logistic Regression, Gradient Boosting, K-Nearest Neighbors (KNN), Support Vector Machine (SVM), etc.

Cross-Validation: By default, PyCaret performs a stratified k-fold cross-validation (with k=10) during the model training process. Cross-validation helps to assess the model's performance on different subsets of the training data, reducing the risk of overfitting.

Evaluation Metrics: For each model, PyCaret calculates and displays various evaluation metrics, such as accuracy, area under the receiver operating characteristic curve (AUC-ROC), recall, precision, F1-score, kappa, and Matthews correlation coefficient (MCC). These metrics provide insights into the model's predictive performance.

Model Ranking: The compare_models() function ranks the models based on their performance on the given dataset. It displays the models in descending order of their average accuracy score by default.

Return Value: The function returns a table that shows the performance metrics of each trained model. The table helps us easily compare the performance of different models and select the most suitable one for further tuning.

Using compare_models() is a powerful and time-saving approach to quickly identify promising models for the given classification problem. After analyzing the results, we can choose the best-performing model and proceed with further tuning and evaluation to optimize its performance.

```
train_data = dataset.sample(frac=0.8,
random_state=101).reset_index(drop=True)
test_data = dataset.drop(train_data.index).reset_index(drop=True)

#since gbc is giving best results we are proceeding with gbc i.e
gradient boosting classifier
gbc = create_model('gbc')
```

```
<IPython.core.display.HTML object>

<pandas.io.formats.style.Styler at 0x787828245a20>

{"model_id":"d66d6eb22ab348a1b9abb074df230a60","version_major":2,"vers
ion_minor":0}

<IPython.core.display.HTML object>
```

we use the create_model() function from PyCaret to create a Gradient Boosting Classifier (GBC) model. This function automatically trains the GBC model on the training data, performs cross-validation, and returns the trained model object.

```
tuned_gbc = tune_model(gbc)
```

```
<IPython.core.display.HTML object>

<pandas.io.formats.style.Styler at 0x787828246fb0>

{"model_id":"e24b0a32c0ce4443bb7ee9a90d2c204f","version_major":2,"vers
ion_minor":0}

Fitting 10 folds for each of 10 candidates, totalling 100 fits

<IPython.core.display.HTML object>

Original model was better than the tuned model, hence it will be
returned. NOTE: The display metrics are for the tuned model (not the
original one).
```

In the code tuned_gbc = tune_model(gbc), we are using the tune_model() function from PyCaret to perform hyperparameter tuning on the Gradient Boosting Classifier (gbc) model that we created earlier.

Hyperparameter tuning is the process of finding the best set of hyperparameters for a machine learning model that results in optimal performance. Hyperparameters are model settings that are not learned during training and need to be specified before training the model. By tuning the hyperparameters, we can improve the model's performance and make it more suitable for our specific dataset.

When we call tune_model(gbc), PyCaret will automatically search for the best hyperparameters for the Gradient Boosting Classifier using a technique called "Random Grid Search." It will create a grid of possible hyperparameter values and randomly sample combinations from this grid to

train and evaluate the model. The combination of hyperparameters that gives the best performance will be returned as the tuned_gbc model.

By tuning the hyperparameters, we can potentially improve the accuracy and generalization of the Gradient Boosting Classifier model on our specific dataset. After hyperparameter tuning, the tuned_gbc model will be ready for evaluation and prediction.

We can now use the evaluate_model() function to assess the performance of the tuned model on the training data and use the predict_model() function to make predictions on new, unseen data. This helps us understand how well the model is generalizing to unseen data and how it performs in real-world scenarios.

When we used the tune_model(gbc) function, PyCaret performed hyperparameter tuning on the gbc model and evaluated its performance using cross-validation and evaluation metrics. However, it found that the original gbc model, without hyperparameter tuning, had better performance compared to the tuned model.

Overall, it's not uncommon to encounter cases where hyperparameter tuning does not lead to significant improvements in model performance. In such cases, sticking with the original model is a reasonable choice to avoid overfitting and maintain simplicity.

```
#sticking to original model for evaluation
evaluate_model(gbc)
```


```
{"model_id":"6e84fa63b4374641aac66139e4766128","version_major":2,"vers
ion_minor":0}
```


we are using the evaluate_model() function from PyCaret to assess the performance of the Gradient Boosting Classifier (gbc) model on the training data. This function provides a comprehensive report containing various evaluation metrics that help us understand how well the model is performing.

When we call evaluate_model(gbc), PyCaret performs the following tasks:

Computes Accuracy: The accuracy metric measures the proportion of correctly classified instances out of the total instances. It tells us how often the model makes correct predictions.

Computes AUC (Area Under the Curve): AUC is a performance metric for binary classification models that measures the area under the Receiver Operating Characteristic (ROC) curve. It provides an overall measure of the model's ability to discriminate between the positive and negative classes.

Computes Recall: Recall, also known as sensitivity or true positive rate, measures the proportion of actual positive instances correctly predicted by the model. It indicates how well the model identifies positive cases.

Computes Precision: Precision is the proportion of true positive instances out of the total instances predicted as positive by the model. It measures the accuracy of positive predictions made by the model.

Computes F1 Score: The F1 score is the harmonic mean of precision and recall. It provides a balance between precision and recall, especially in imbalanced datasets.

Computes Kappa: The Kappa statistic measures the agreement between the actual and predicted classifications, taking into account the possibility of agreement occurring by chance.

Computes MCC (Matthews Correlation Coefficient): MCC is a measure of the quality of binary classifications. It takes into account true and false positives and negatives and is considered a balanced metric for imbalanced datasets.

By evaluating the gbc model using evaluate_model(gbc), we can get a comprehensive understanding of how well the model is performing and how it is handling the classification task on the training data. This analysis helps us identify potential areas for improvement or model adjustments to optimize its performance on unseen data.

Running the code in Google Colab can be advantageous as it provides faster execution times compared to local Jupyter notebooks due to the availability of better hardware resources and integration with Google Cloud services. Additionally, Colab allows us to leverage GPUs and TPUs, which can significantly speed up computations for machine learning tasks.

In the given code, we are using the evaluate_model() function from PyCaret to assess the performance of the Gradient Boosting Classifier (gbc) model on the training data. This function provides a comprehensive report containing various evaluation metrics that help us understand how well the model is performing.

```
predict_model(gbc)
```

```
<pandas.io.formats.style.Styler at 0x7877b278d6f0>
```

```
      customerID  gender  SeniorCitizen Partner Dependents  tenure  \
2937  5312-IRCFR  Female              0     Yes        Yes      64
3276  4210-QFJMF  Female              0      No         No       4
4374  2718-YSKCS    Male              0     Yes        Yes      71
4375  9896-UYMIE    Male              0      No         No      66
237   9903-LYSAB    Male              0     Yes         No      18
...          ...     ...            ...     ...        ...     ...
4496  9489-JMTTN  Female              0     Yes        Yes      72
921   8942-DBMHZ    Male              0      No         No      12
5904  9402-CXWPL  Female              0      No         No      70
3088  5751-USDBL    Male              0     Yes        Yes      46
253   3282-ZISZV    Male              0      No        Yes      32

     PhoneService MultipleLines InternetService
OnlineSecurity  ...  \
2937          Yes           Yes     Fiber optic
No  ...
3276          Yes            No     Fiber optic
No  ...
4374          Yes            No              No  No internet
service  ...
4375          Yes           Yes     Fiber optic
```

```
Yes  ...
237           Yes           Yes     Fiber optic
No  ...
...           ...           ...             ...                   ...
...
4496          Yes           Yes             DSL
Yes  ...
921           Yes            No              No  No internet
service  ...
5904          Yes           Yes     Fiber optic
No  ...
3088          Yes            No             DSL
Yes  ...
253           Yes           Yes             DSL
No  ...

              StreamingTV      StreamingMovies        Contract  \
2937                   No                  Yes        One year
3276                   No                  Yes  Month-to-month
4374  No internet service  No internet service        Two year
4375                  Yes                  Yes        One year
237                    No                   No  Month-to-month
...                   ...                  ...             ...
4496                  Yes                  Yes        Two year
921   No internet service  No internet service  Month-to-month
5904                  Yes                  Yes        One year
3088                  Yes                  Yes        Two year
253                   Yes                  Yes        One year

     PaperlessBilling              PaymentMethod MonthlyCharges
TotalCharges  \
2937              Yes           Electronic check      92.849998
5980.75
3276              Yes           Electronic check      79.150002
317.25
4374              Yes  Bank transfer (automatic)      19.600000
1387.45
4375              Yes  Bank transfer (automatic)     114.300003
7383.7
237               Yes           Electronic check      73.150002
1305.95
...               ...                        ...             ...
...
4496               No    Credit card (automatic)      89.750000
6595.9
921                No               Mailed check      20.450001
255.35
5904               No           Electronic check      98.900002
6838.6
```

```
3088               No               Mailed check      81.000000
3846.35
253                No    Credit card (automatic)      83.699997
2633.3

     Churn  prediction_label prediction_score
2937     0                 0           0.9048
3276     1                 1           0.6998
4374     0                 0           0.9776
4375     0                 0           0.9390
237      0                 0           0.6802
...    ...               ...              ...
4496     0                 0           0.9048
921      0                 0           0.9074
5904     0                 0           0.8913
3088     0                 0           0.7811
253      0                 0           0.7097

[2113 rows x 23 columns]
```

By using predict_model(gbc), we can gain insights into how well the trained model performs on new, unseen data. This allows us to understand its generalization capabilities and how it would behave in real-world scenarios. The predictions can be further analyzed and used for making business decisions, such as identifying potential churn customers and implementing targeted retention strategies to reduce churn rate.

```
unseen_prediction = predict_model(gbc , data = test_data)
unseen_prediction

<pandas.io.formats.style.Styler at 0x7877b278f670>

      customerID  gender  SeniorCitizen Partner Dependents  tenure  \
0     2320-JRSDE  Female              0     Yes        Yes       1
1     2087-QAREY  Female              0     Yes         No      22
2     0601-WZHJF    Male              0     Yes         No      14
3     4423-JWZJN    Male              0     Yes        Yes      64
4     5143-WMWOG    Male              0      No         No       1
...          ...     ...            ...     ...        ...     ...
1404  6840-RESVB    Male              0     Yes        Yes      24
1405  2234-XADUH  Female              0     Yes        Yes      72
1406  4801-JZAZL  Female              0     Yes        Yes      11
1407  8361-LTMKD    Male              1     Yes         No       4
1408  3186-AJIEK    Male              0      No         No      66

     PhoneService     MultipleLines InternetService
OnlineSecurity  ...  \
0             Yes                No              No  No internet
service  ...
1             Yes                No             DSL
No  ...
```

```
2              No  No phone service             DSL
No  ...
3             Yes               Yes     Fiber optic
No  ...
4             Yes                No              No  No internet
service  ...
...           ...               ...             ...                   .
..  ...
1404          Yes               Yes             DSL
Yes  ...
1405          Yes               Yes     Fiber optic
No  ...
1406           No  No phone service             DSL
Yes  ...
1407          Yes               Yes     Fiber optic
No  ...
1408          Yes                No     Fiber optic
Yes  ...

              StreamingTV      StreamingMovies        Contract  \
0     No internet service  No internet service  Month-to-month
1                      No                  No  Month-to-month
2                     Yes                 Yes  Month-to-month
3                      No                 Yes        One year
4     No internet service  No internet service  Month-to-month
...                   ...                 ...             ...
1404                  Yes                 Yes        One year
1405                  Yes                 Yes        One year
1406                   No                  No  Month-to-month
1407                   No                  No  Month-to-month
1408                  Yes                 Yes        Two year

     PaperlessBilling              PaymentMethod MonthlyCharges
TotalCharges  \
0                 Yes           Electronic check      19.900000
19.9
1                 Yes               Mailed check      54.700001
1178.75
2                  No           Electronic check      46.349998
667.7
3                  No    Credit card (automatic)      90.250000
5629.15
4                  No           Electronic check      19.950001
19.95
...               ...                        ...            ...
...
1404              Yes               Mailed check      84.800003
1990.5
1405              Yes    Credit card (automatic)     103.199997
```

```
7362.9
1406              Yes           Electronic check      29.600000
346.45
1407              Yes               Mailed check      74.400002
306.6
1408              Yes  Bank transfer (automatic)     105.650002
6844.5

     Churn  prediction_label prediction_score
0        1                 0           0.5923
1        0                 0           0.8096
2        1                 0           0.7387
3        0                 0           0.9407
4        1                 0           0.6575
...    ...               ...              ...
1404     0                 0           0.6241
1405     0                 0           0.8876
1406     0                 0           0.8415
1407     1                 1           0.5972
1408     0                 0           0.8009

[1409 rows x 23 columns]

unseen_prediction1 = predict_model(gbc , data = train_data)
unseen_prediction1

<pandas.io.formats.style.Styler at 0x787828167d30>

      customerID  gender  SeniorCitizen Partner Dependents  tenure  \
0     8659-IOOPU  Female              0     Yes         Yes      71
1     0887-HJGAR    Male              0      No          No       1
2     1029-QFBEN    Male              0      No          No       1
3     7579-OOPEC  Female              1     Yes          No       2
4     8473-VUVJN    Male              1      No          No       1
...          ...     ...            ...     ...         ...     ...
5629  1428-GTBJJ    Male              0      No          No      11
5630  7310-EGVHZ    Male              0      No          No       1
5631  3045-XETSH  Female              0      No          No      10
5632  2237-ZFSMY  Female              0      No          No      39
5633  0178-CIIKR  Female              0      No          No       3

     PhoneService MultipleLines InternetService
OnlineSecurity  ...  \
0             Yes           Yes     Fiber optic
Yes  ...
1             Yes            No             DSL
No  ...
2             Yes            No              No  No internet
service  ...
3             Yes            No             DSL
```

```
No  ...
4             Yes           Yes     Fiber optic
No  ...
...           ...           ...             ...                   ...
...
5629          Yes            No     Fiber optic
No  ...
5630          Yes            No              No  No internet
service  ...
5631          Yes           Yes     Fiber optic
No  ...
5632          Yes            No     Fiber optic
Yes  ...
5633          Yes            No              No  No internet
service  ...

              StreamingTV      StreamingMovies        Contract  \
0                      No                  Yes        Two year
1                      No                   No  Month-to-month
2     No internet service  No internet service  Month-to-month
3                      No                   No  Month-to-month
4                      No                   No  Month-to-month
...                   ...                  ...             ...
5629                   No                   No  Month-to-month
5630  No internet service  No internet service  Month-to-month
5631                   No                  Yes  Month-to-month
5632                  Yes                   No        One year
5633  No internet service  No internet service  Month-to-month

     PaperlessBilling              PaymentMethod MonthlyCharges
TotalCharges  \
0                  No           Electronic check     100.449997
7159.7
1                 Yes               Mailed check      45.700001
45.7
2                  No               Mailed check      19.549999
19.55
3                  No    Credit card (automatic)      50.150002
115.1
4                 Yes           Electronic check      73.650002
73.65
...               ...                        ...            ...
...
5629              Yes           Electronic check      74.550003
824.75
5630               No  Bank transfer (automatic)      20.200001
20.2
5631              Yes           Electronic check      94.849998
953.45
```

```
5632               Yes           Electronic check      95.550003
3692.85
5633                No               Mailed check      19.950001
58

     Churn  prediction_label prediction_score
0        0                 0           0.8913
1        1                 1           0.5702
2        1                 0           0.6972
3        1                 1           0.5598
4        1                 1           0.6372
...    ...               ...              ...
5629     1                 0           0.5680
5630     0                 1           0.5234
5631     1                 1           0.7292
5632     1                 0           0.6720
5633     0                 0           0.7939

[5634 rows x 23 columns]
```

By using predict_model(gbc, data=test_data), we can now analyze and interpret the model's performance on new, unseen data. This allows us to understand how well the model generalizes to new customers and how accurately it predicts churn behavior on real-world data.

It is essential to evaluate the predictions on unseen data and compare them with the actual outcomes to assess the model's effectiveness and reliability. This validation process helps in identifying potential areas for improvement and fine-tuning the model to enhance its performance for future predictions.

```
save_model(gbc , 'gbc_model')

Transformation Pipeline and Model Successfully Saved

(Pipeline(memory=FastMemory(location=/tmp/joblib),
          steps=[('numerical_imputer',
                  TransformerWrapper(exclude=None,
                                     include=['SeniorCitizen',
'tenure',
                                              'MonthlyCharges'],

transformer=SimpleImputer(add_indicator=False,

copy=True,

fill_value=None,

keep_empty_features=False,

missing_values=nan,
```

```
strategy='mean',

verbose='deprecated'))),
                 ('categorical_imputer',
                  Transform...
                                                criterion='friedman_mse',
init=None,
                                                learning_rate=0.1,
loss='log_loss',
                                                max_depth=3,
max_features=None,
                                                max_leaf_nodes=None,

min_impurity_decrease=0.0,
                                                min_samples_leaf=1,
                                                min_samples_split=2,

min_weight_fraction_leaf=0.0,
                                                n_estimators=100,
                                                n_iter_no_change=None,
                                                random_state=123,
subsample=1.0,
                                                tol=0.0001,
validation_fraction=0.1,
                                                verbose=0,
warm_start=False))],
          verbose=False),
 'gbc_model.pkl')
```