Analyzing Customer Churn in a Telecommunications Company

July 24, 2024

[3]:

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

# 1. Import the dataset
file_path = "C:\\Users\\MUBASHIR␣
 ↪KHAN\\Desktop\\jupyter\\DMV\\WA_Fn-UseC_-Telco-Customer-Churn.csv"
df = pd.read_csv(file_path)

# 2. Explore the dataset
print("Dataset Head:\n", df.head())
print("\nDataset Info:\n", df.info())
print("\nSummary Statistics:\n", df.describe())

# 3. Handle missing values
# Checking for missing values
print("\nMissing Values:\n", df.isnull().sum())

# Filling missing values or dropping
# Example: If missing values are found in a column, fill with median or drop␣
 ↪rows/columns
numeric_columns = df.select_dtypes(include=[np.number]).columns
df[numeric_columns] = df[numeric_columns].fillna(df[numeric_columns].median())

# 4. Remove duplicate records
df.drop_duplicates(inplace=True)

# 5. Check for inconsistent data and standardize
# Example: Standardizing 'TotalCharges' as it may contain spaces and need to be␣
 ↪numeric
df['TotalCharges'] = pd.to_numeric(df['TotalCharges'], errors='coerce')

# Rechecking missing values after conversion
print("\nMissing Values after conversion:\n", df.isnull().sum())
df['TotalCharges'].fillna(df['TotalCharges'].median(), inplace=True)
```

```
# 6. Convert columns to the correct data types
# Example: Converting 'SeniorCitizen' from integer to boolean
df['SeniorCitizen'] = df['SeniorCitizen'].astype(bool)

# 7. Identify and handle outliers
# Example: Using IQR to handle outliers in 'tenure' column
Q1 = df['tenure'].quantile(0.25)
Q3 = df['tenure'].quantile(0.75)
IQR = Q3 - Q1
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR
df = df[(df['tenure'] >= lower_bound) & (df['tenure'] <= upper_bound)]

# 8. Perform feature engineering
# Example: Creating 'TotalServices' as the count of all services used by a␣
 ↪customer
services = ['PhoneService', 'MultipleLines', 'InternetService',␣
 ↪'OnlineSecurity', 'OnlineBackup',
            'DeviceProtection', 'TechSupport', 'StreamingTV', 'StreamingMovies']
df['TotalServices'] = df[services].apply(lambda x: x.eq('Yes').sum(), axis=1)

# 9. Normalize or scale the data if necessary
# Example: Scaling numerical features
scaler = StandardScaler()
numerical_features = ['tenure', 'MonthlyCharges', 'TotalCharges',␣
 ↪'TotalServices']
df[numerical_features] = scaler.fit_transform(df[numerical_features])

# 10. Split the dataset into training and testing sets
X = df.drop(columns=['Churn'])
y = df['Churn'].apply(lambda x: 1 if x == 'Yes' else 0)  # Assuming 'Churn' is␣
 ↪the target column
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)

# 11. Export the cleaned dataset for future analysis or modeling
cleaned_file_path = "C:\\Users\\MUBASHIR␣
 ↪KHAN\\Desktop\\jupyter\\DMV\\Cleaned_Telco_Customer_Churn.csv"
df.to_csv(cleaned_file_path, index=False)

print(f"Cleaned dataset saved to {cleaned_file_path}")
```

```
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]
<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

Dataset Info:
 None

Summary Statistics:
        SeniorCitizen       tenure  MonthlyCharges
count    7043.000000  7043.000000     7043.000000
mean        0.162147    32.371149       64.761692
std         0.368612    24.559481       30.090047
min         0.000000     0.000000       18.250000
25%         0.000000     9.000000       35.500000
50%         0.000000    29.000000       70.350000
75%         0.000000    55.000000       89.850000
max         1.000000    72.000000      118.750000

Missing Values:
 customerID          0
gender              0
SeniorCitizen       0
Partner             0
Dependents          0
tenure              0
PhoneService        0
MultipleLines       0
InternetService     0
OnlineSecurity      0
OnlineBackup        0
DeviceProtection    0
TechSupport         0
StreamingTV         0
StreamingMovies     0
Contract            0
PaperlessBilling    0
PaymentMethod       0
MonthlyCharges      0
TotalCharges        0
Churn               0
dtype: int64

Missing Values after conversion:
 customerID           0
gender               0
SeniorCitizen        0
```

```
Partner              0
Dependents           0
tenure               0
PhoneService         0
MultipleLines        0
InternetService      0
OnlineSecurity       0
OnlineBackup         0
DeviceProtection     0
TechSupport          0
StreamingTV          0
StreamingMovies      0
Contract             0
PaperlessBilling     0
PaymentMethod        0
MonthlyCharges       0
TotalCharges        11
Churn                0
dtype: int64
Cleaned dataset saved to C:\Users\MUBASHIR
KHAN\Desktop\jupyter\DMV\Cleaned_Telco_Customer_Churn.csv
```

```
[ ]:
```