Project Report

Introduction to Data Science

Semester – 2

“Telco Customer Churn dataset”

By

PIYUSH TANDALE

Reg no: 2411021240006

GitHub link: piyushT3003/IDS

Department of Computer Application

Alliance University Chandapura – Anekal Main Road,

Anekal Bengaluru – 562 106

April 2025

Project Overview

This project is a data analysis case study focused on customer churn prediction in a telecom company. Using the "Telco Customer Churn" dataset, the project explores the relationship between customer behavior and churn, aiming to identify patterns and visualize insights that could help reduce churn rates.

Introduction

Customer churn, or the loss of clients or subscribers, is a significant concern for subscription-based businesses such as telecom companies. The notebook begins by importing and cleaning the Telco Customer Churn dataset. This dataset includes customer demographic information, account details, and service usage statistics. The analysis provides a foundation for understanding which factors influence churn and helps businesses strategize better retention practices.

Project Goals

- To clean and prepare the dataset for analysis.
- To explore and visualize customer attributes and behavior.
- To identify key factors related to customer churn.
- To generate insights through visualizations (e.g., histograms, pie charts, box plots).
- To support decision-making for churn reduction strategies.

Challenges

- Missing and incorrect data: The TotalCharges column required conversion to numeric, which revealed missing or malformed entries.
- Data imbalances: As often seen in churn datasets, the number of customers who churned vs. those who didn't may be imbalanced, affecting visual clarity and any potential modeling.
- Feature interpretation: Some categorical variables required careful interpretation before visualization (e.g., contract type, payment method).
- Visualization complexity: Plotting meaningful graphs that convey insight without overcrowding the visuals.

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns


df=pd.read_csv(r"C:\Users\piyus\Downloads\WA_Fn-UseC_-Telco-Customer-Churn
(1).csv")
df
```

```
      customerID  gender  SeniorCitizen Partner Dependents  tenure  \
0     7590-VHVEG  Female              0     Yes         No       1
1     5575-GNVDE    Male              0      No         No      34
2     3668-QPYBK    Male              0      No         No       2
3     7795-CFOCW    Male              0      No         No      45
4     9237-HQITU  Female              0      No         No       2
...          ...     ...            ...     ...        ...     ...
7038  6840-RESVB    Male              0     Yes        Yes      24
7039  2234-XADUH  Female              0     Yes        Yes      72
7040  4801-JZAZL  Female              0     Yes        Yes      11
7041  8361-LTMKD    Male              1     Yes         No       4
7042  3186-AJIEK    Male              0      No         No      66

     PhoneService     MultipleLines InternetService OnlineSecurity  ...  \
0              No  No phone service             DSL             No  ...
1             Yes                No             DSL            Yes  ...
2             Yes                No             DSL            Yes  ...
3              No  No phone service             DSL            Yes  ...
4             Yes                No     Fiber optic             No  ...
...           ...               ...             ...            ...  ...
7038          Yes               Yes             DSL            Yes  ...
7039          Yes               Yes     Fiber optic             No  ...
7040           No  No phone service             DSL            Yes  ...
7041          Yes               Yes     Fiber optic             No  ...
7042          Yes                No     Fiber optic            Yes  ...

     DeviceProtection TechSupport StreamingTV StreamingMovies        Contract
\
0                  No          No          No              No  Month-to-month
1                 Yes          No          No              No        One year
2                  No          No          No              No  Month-to-month
3                 Yes         Yes          No              No        One year
4                  No          No          No              No  Month-to-month
...               ...         ...         ...             ...             ...
7038              Yes         Yes         Yes             Yes        One year
7039              Yes          No         Yes             Yes        One year
7040               No          No          No              No  Month-to-month
7041               No          No          No              No  Month-to-month
7042              Yes         Yes         Yes             Yes        Two year

     PaperlessBilling             PaymentMethod MonthlyCharges  TotalCharges
```

```
\
0                  Yes           Electronic check          29.85         29.85
1                   No               Mailed check          56.95        1889.5
2                  Yes               Mailed check          53.85        108.15
3                   No  Bank transfer (automatic)          42.30       1840.75
4                  Yes           Electronic check          70.70        151.65
...                ...                        ...            ...           ...
7038               Yes               Mailed check          84.80        1990.5
7039               Yes    Credit card (automatic)         103.20        7362.9
7040               Yes           Electronic check          29.60        346.45
7041               Yes               Mailed check          74.40         306.6
7042               Yes  Bank transfer (automatic)         105.65        6844.5

     Churn
0       No
1       No
2      Yes
3       No
4      Yes
...    ...
7038    No
7039    No
7040    No
7041   Yes
7042    No

[7043 rows x 21 columns]
```

1. Check for missing values and data types

```
# Check for missing values
print("Missing Values:\n", df.isnull().sum())

# Check data types
print("\nData Types:\n", df.dtypes)
```

```
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

Data Types:
 customerID          object
gender              object
SeniorCitizen        int64
Partner             object
Dependents          object
tenure               int64
PhoneService        object
MultipleLines       object
InternetService     object
OnlineSecurity      object
OnlineBackup        object
DeviceProtection    object
TechSupport         object
StreamingTV         object
StreamingMovies     object
Contract            object
PaperlessBilling    object
PaymentMethod       object
MonthlyCharges     float64
TotalCharges        object
Churn               object
dtype: object
```

1. A brief descriptive statistics overview

```
# Summary statistics
df.describe(include='all')
```

```
        customerID gender  SeniorCitizen Partner Dependents       tenure  \
count         7043   7043    7043.000000    7043       7043  7043.000000
unique        7043      2            NaN       2          2          NaN
top     7590-VHVEG   Male            NaN      No         No          NaN
freq             1   3555            NaN    3641       4933          NaN
mean           NaN    NaN       0.162147     NaN        NaN    32.371149
std            NaN    NaN       0.368612     NaN        NaN    24.559481
min            NaN    NaN       0.000000     NaN        NaN     0.000000
25%            NaN    NaN       0.000000     NaN        NaN     9.000000
50%            NaN    NaN       0.000000     NaN        NaN    29.000000
75%            NaN    NaN       0.000000     NaN        NaN    55.000000
max            NaN    NaN       1.000000     NaN        NaN    72.000000
```

```
       PhoneService MultipleLines InternetService OnlineSecurity  ...  \
count          7043          7043            7043           7043  ...
unique            2             3               3              3  ...
top             Yes            No     Fiber optic             No  ...
freq           6361          3390            3096           3498  ...
mean            NaN           NaN             NaN            NaN  ...
std             NaN           NaN             NaN            NaN  ...
min             NaN           NaN             NaN            NaN  ...
25%             NaN           NaN             NaN            NaN  ...
50%             NaN           NaN             NaN            NaN  ...
75%             NaN           NaN             NaN            NaN  ...
max             NaN           NaN             NaN            NaN  ...

       DeviceProtection TechSupport StreamingTV StreamingMovies  \
count              7043        7043        7043            7043
unique                3           3           3               3
top                  No          No          No              No
freq               3095        3473        2810            2785
mean                NaN         NaN         NaN             NaN
std                 NaN         NaN         NaN             NaN
min                 NaN         NaN         NaN             NaN
25%                 NaN         NaN         NaN             NaN
50%                 NaN         NaN         NaN             NaN
75%                 NaN         NaN         NaN             NaN
max                 NaN         NaN         NaN             NaN

              Contract PaperlessBilling     PaymentMethod MonthlyCharges  \
count             7043             7043              7043    7043.000000
unique               3                2                 4            NaN
top     Month-to-month              Yes  Electronic check            NaN
freq              3875             4171              2365            NaN
mean               NaN              NaN               NaN      64.761692
std                NaN              NaN               NaN      30.090047
min                NaN              NaN               NaN      18.250000
25%                NaN              NaN               NaN      35.500000
50%                NaN              NaN               NaN      70.350000
75%                NaN              NaN               NaN      89.850000
max                NaN              NaN               NaN     118.750000

        TotalCharges Churn
count           7043  7043
unique          6531     2
top                     No
freq              11  5174
mean             NaN   NaN
std              NaN   NaN
min              NaN   NaN
25%              NaN   NaN
50%              NaN   NaN
75%              NaN   NaN
```

```
max                NaN    NaN

[11 rows x 21 columns]
```

1. Convert 'TotalCharges' to numeric and handle non-numeric entries

```
# Convert 'TotalCharges' to numeric, forcing errors to NaN
df['TotalCharges'] = pd.to_numeric(df['TotalCharges'], errors='coerce')

# Check how many rows were affected
print("Missing values after conversion:", df['TotalCharges'].isnull().sum())

# Fill or drop missing values
df = df.dropna(subset=['TotalCharges'])

Missing values after conversion: 11
```

1. Sort by 'tenure'

```
df = df.sort_values(by='tenure')
```

1. Check basic info

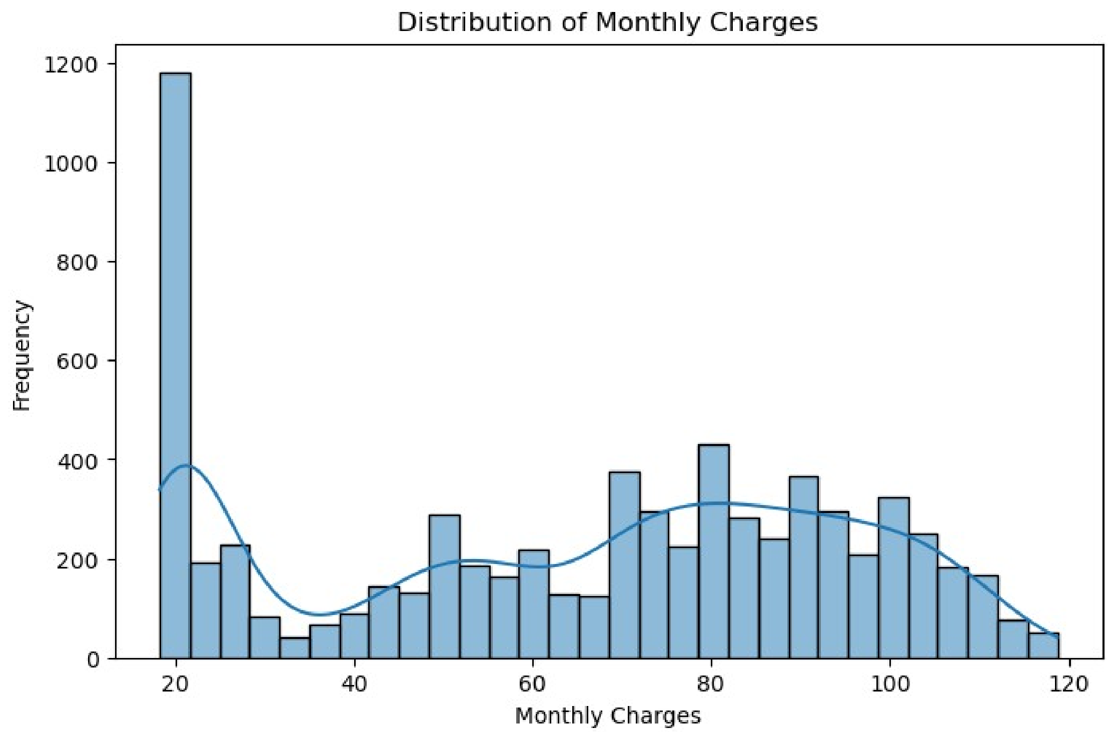
```
df.info()

<class 'pandas.core.frame.DataFrame'>
Index: 7032 entries, 0 to 3543
Data columns (total 21 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   customerID        7032 non-null   object
 1   gender            7032 non-null   object
 2   SeniorCitizen     7032 non-null   int64
 3   Partner           7032 non-null   object
 4   Dependents        7032 non-null   object
 5   tenure            7032 non-null   int64
 6   PhoneService      7032 non-null   object
 7   MultipleLines     7032 non-null   object
 8   InternetService   7032 non-null   object
 9   OnlineSecurity    7032 non-null   object
 10  OnlineBackup      7032 non-null   object
 11  DeviceProtection  7032 non-null   object
 12  TechSupport       7032 non-null   object
 13  StreamingTV       7032 non-null   object
 14  StreamingMovies   7032 non-null   object
 15  Contract          7032 non-null   object
 16  PaperlessBilling  7032 non-null   object
 17  PaymentMethod     7032 non-null   object
 18  MonthlyCharges    7032 non-null   float64
 19  TotalCharges      7032 non-null   float64
 20  Churn             7032 non-null   object
dtypes: float64(2), int64(2), object(17)
memory usage: 1.2+ MB
```

1. Visualizing the Data (requires matplotlib and seaborn)

```
import matplotlib.pyplot as plt
import seaborn as sns
```

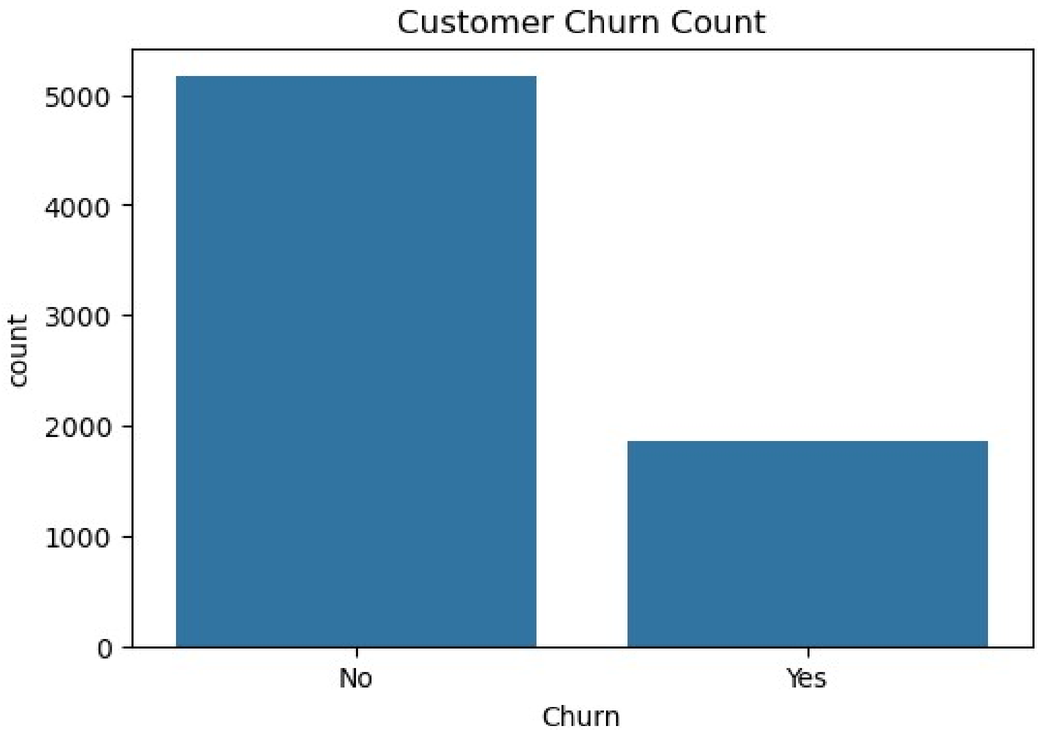
1. Histogram of 'MonthlyCharges'

```
plt.figure(figsize=(8, 5))
sns.histplot(df['MonthlyCharges'], bins=30, kde=True)
plt.title("Distribution of Monthly Charges")
plt.xlabel("Monthly Charges")
plt.ylabel("Frequency")
plt.show()
```



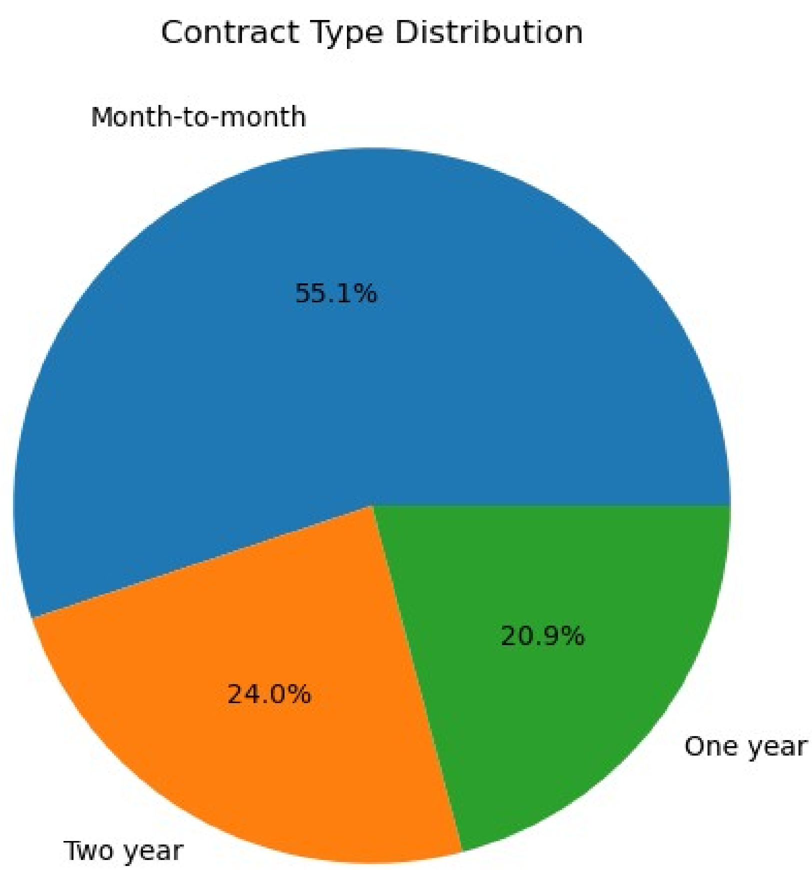
1. Bar plot: Churn count

```
plt.figure(figsize=(6, 4))
sns.countplot(x='Churn', data=df)
plt.title("Customer Churn Count")
plt.show()
```



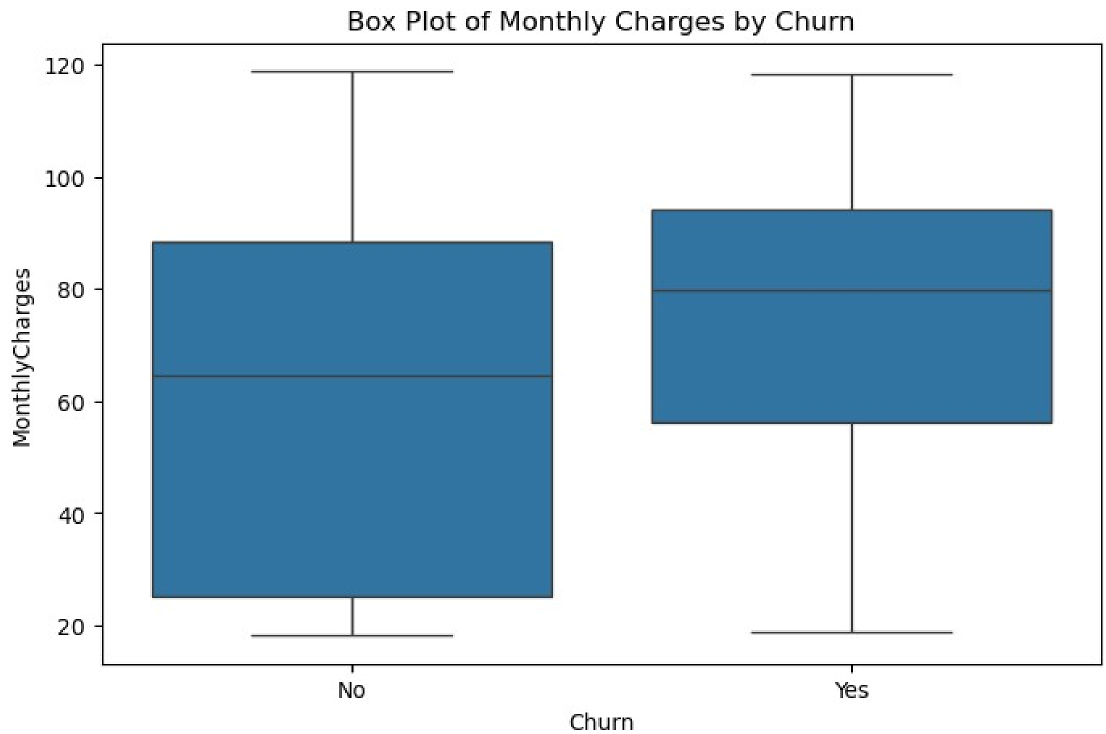
1. Pie chart of 'Contract' types

```
contract_counts = df['Contract'].value_counts()
plt.figure(figsize=(6, 6))
plt.pie(contract_counts, labels=contract_counts.index, autopct='%1.1f%%')
plt.title("Contract Type Distribution")
plt.show()
```



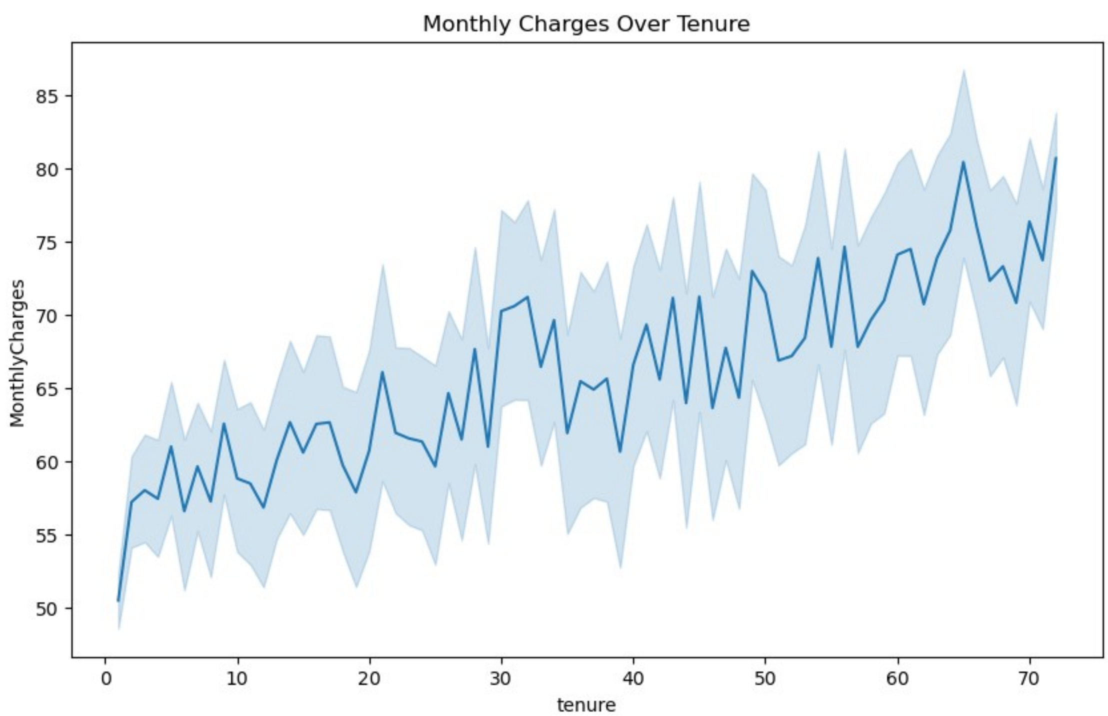
1. Creating a Box Plot (Monthly Charges by Churn)

```
plt.figure(figsize=(8, 5))
sns.boxplot(x='Churn', y='MonthlyCharges', data=df)
plt.title("Box Plot of Monthly Charges by Churn")
plt.show()
```



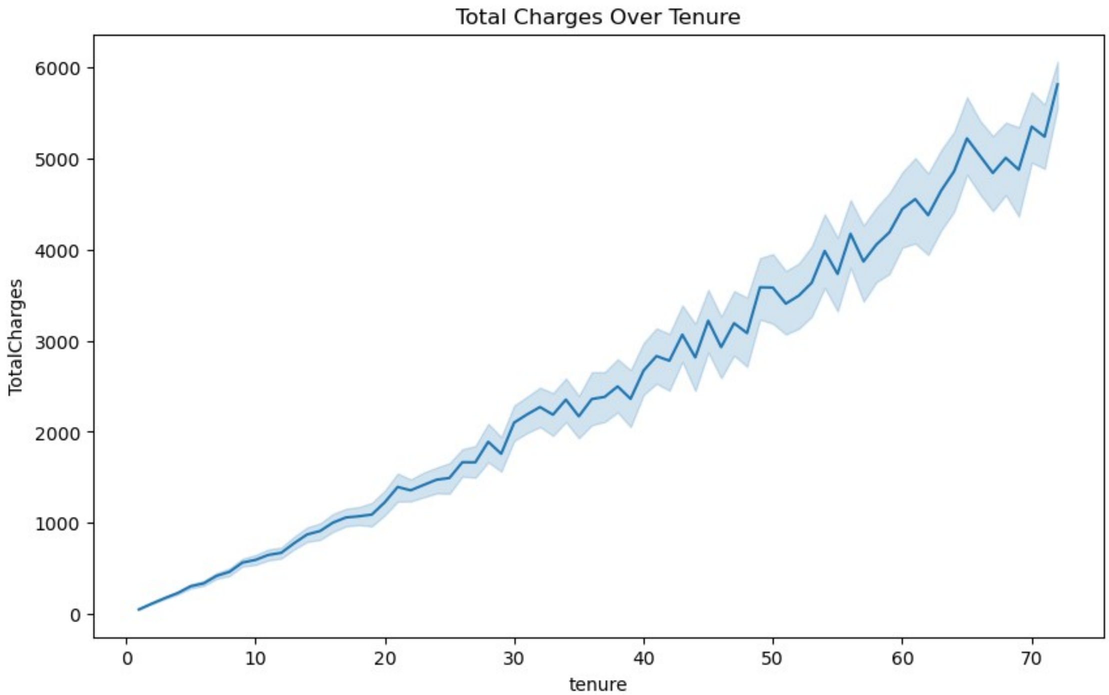
1. Plotting MonthlyCharges over Tenure

```
plt.figure(figsize=(10, 6))
sns.lineplot(x='tenure', y='MonthlyCharges', data=df)
plt.title("Monthly Charges Over Tenure")
plt.show()
```



1. Displaying Volume trends — using 'tenure' as a time-like feature

```
plt.figure(figsize=(10, 6))
sns.lineplot(x='tenure', y='TotalCharges', data=df)
plt.title("Total Charges Over Tenure")
plt.show()
```



1. Creating log feature (log of TotalCharges)

```
import numpy as np
df['LogTotalCharges'] = np.log(df['TotalCharges'] + 1)
```

1. Creating Features and Target

```
features = df.drop(['customerID', 'Churn'], axis=1)
features = pd.get_dummies(features, drop_first=True)
target = df['Churn'].apply(lambda x: 1 if x == 'Yes' else 0)
```

1. Train-test split

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(features, target,
test_size=0.2, random_state=42)
```

1. Training a model (Logistic Regression)

```
from sklearn.linear_model import LogisticRegression

model = LogisticRegression(max_iter=1000)
model.fit(X_train, y_train)
```

```
c:\Users\piyus\anaconda3\Lib\site-packages\sklearn\linear_model\_logistic.py:469: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(

LogisticRegression(max_iter=1000)
```

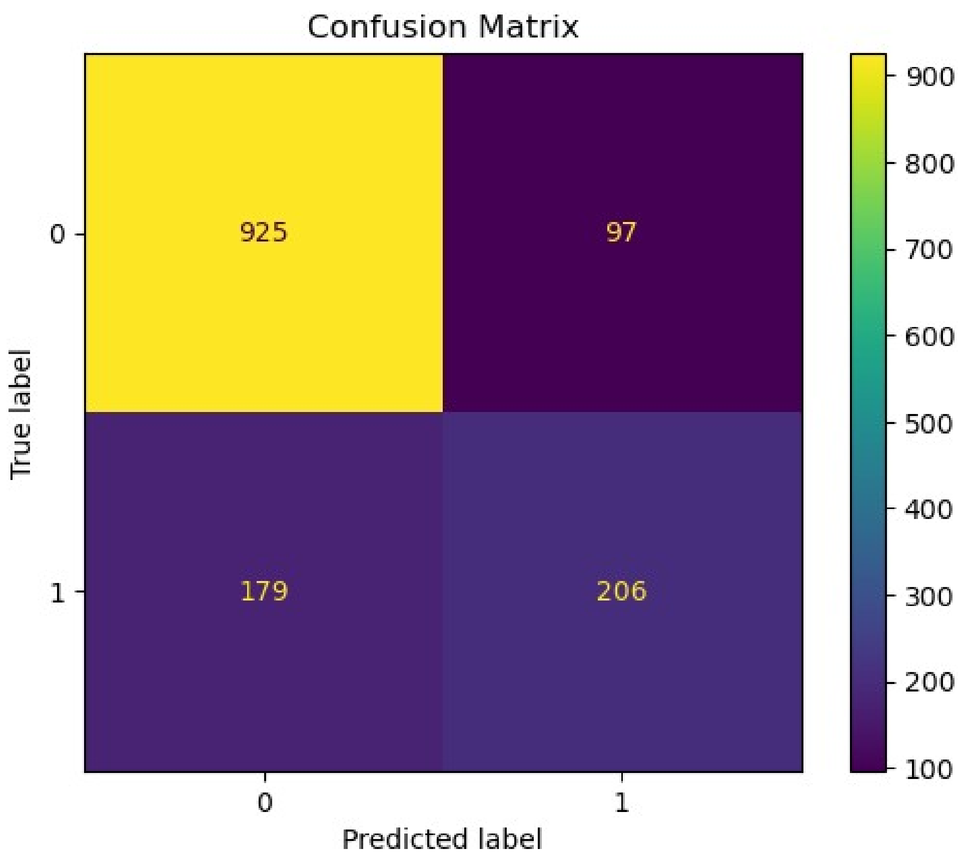
1. Evaluating the model

```
from sklearn.metrics import accuracy_score, classification_report

y_pred = model.predict(X_test)
print("Accuracy:", accuracy_score(y_test, y_pred))
print("Classification Report:\n", classification_report(y_test, y_pred))
```

```
Accuracy: 0.8038379530916845
Classification Report:
               precision    recall  f1-score   support

           0       0.84      0.91      0.87      1022
           1       0.68      0.54      0.60       385

    accuracy                           0.80      1407
   macro avg       0.76      0.72      0.73      1407
weighted avg       0.79      0.80      0.80      1407
```

1. Confusion Matrix

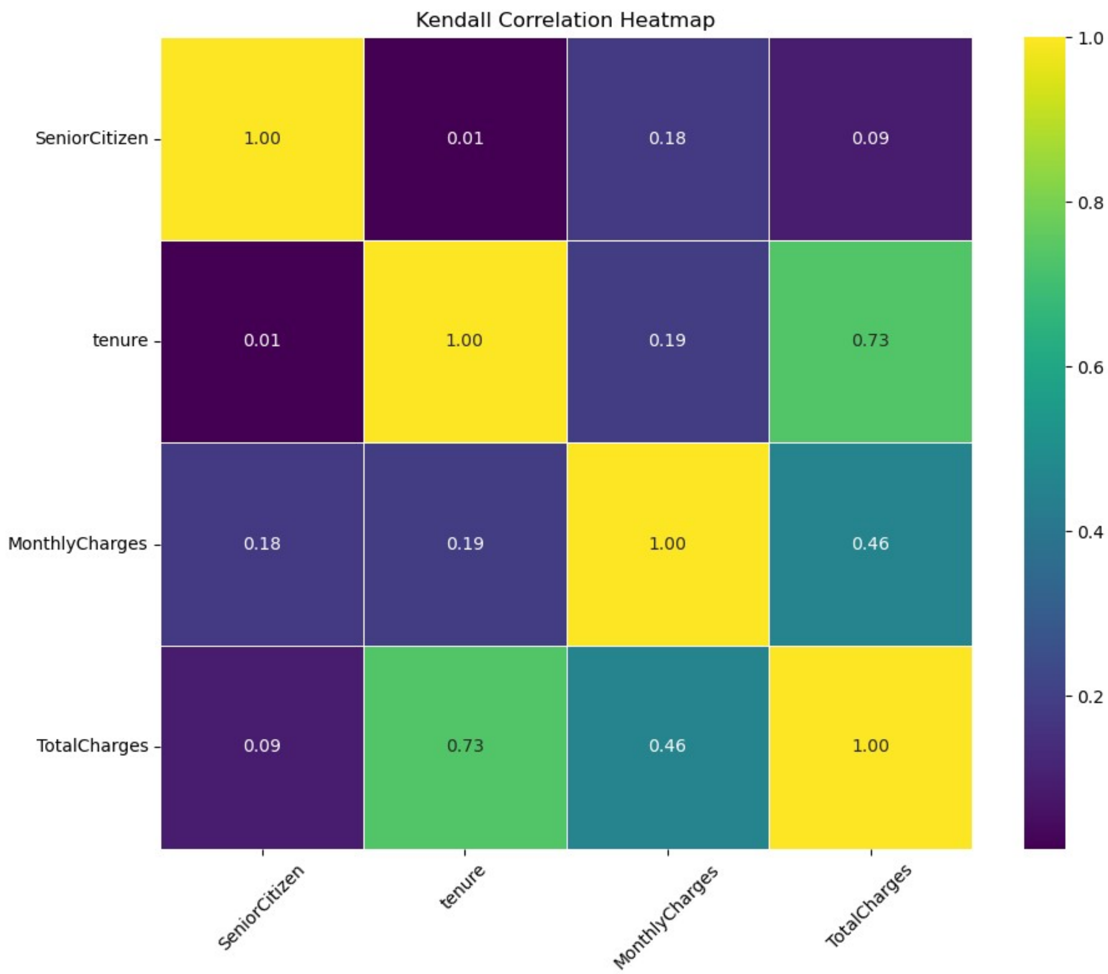
```
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay

cm = confusion_matrix(y_test, y_pred)
ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=model.classes_).plot()
plt.title("Confusion Matrix")
plt.show()
```



1. Creating a Correlation Heatmap

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Load the dataset
df = pd.read_csv(r"C:\Users\piyus\Downloads\WA_Fn-UseC_-Telco-Customer-Churn (1).csv")

# Convert 'TotalCharges' to numeric, coercing errors to NaN
df['TotalCharges'] = pd.to_numeric(df['TotalCharges'], errors='coerce')

# Drop rows with missing values in 'TotalCharges'
df.dropna(subset=['TotalCharges'], inplace=True)

# Select only numeric columns
numeric_df = df.select_dtypes(include=['number'])

# Compute Kendall correlation matrix
corr_matrix = numeric_df.corr(method='kendall')

# Plot the heatmap
```

```
plt.figure(figsize=(10, 8))
sns.heatmap(corr_matrix, annot=True, fmt=".2f", cmap="viridis", square=True,
linewidths=0.5)
plt.title("Kendall Correlation Heatmap")
plt.xticks(rotation=45)
plt.yticks(rotation=0)
plt.tight_layout()
plt.show()
```



Conclusion

This exploratory analysis highlighted key trends in customer churn behavior. For instance, churn was more common among customers with month-to-month contracts and higher monthly charges. Visual tools like histograms, pie charts, and box plots helped simplify complex relationships, making the data easier to interpret. These insights can serve as a foundation for building predictive models or crafting customer retention strategies.