```
# Abdurrahman Bulut
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix,
classification_report
```

Task 1

Adım 1

```
df = pd.read_csv('Telco-Customer-Churn.csv')

print(df.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]
```

```
# Sütun isimleri
print(df.columns)
```

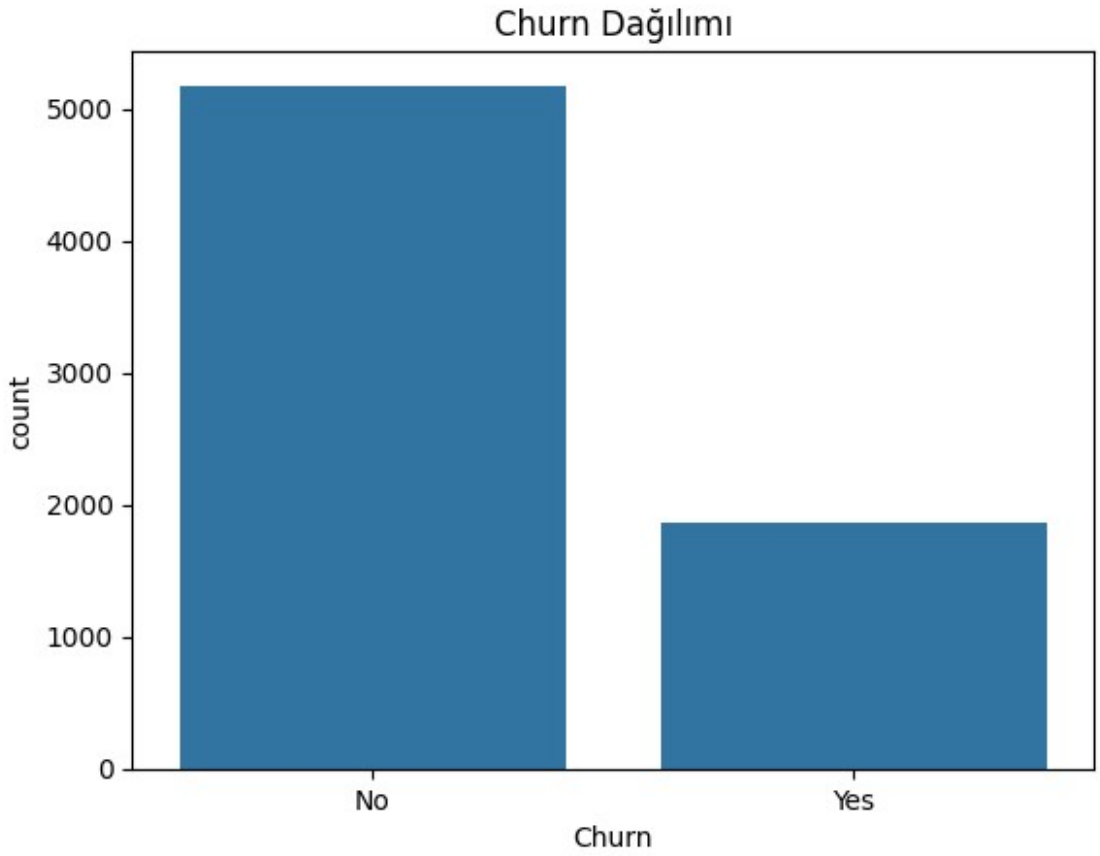
```
Index(['customerID', 'gender', 'SeniorCitizen', 'Partner',
'Dependents',
       'tenure', 'PhoneService', 'MultipleLines', 'InternetService',
       'OnlineSecurity', 'OnlineBackup', 'DeviceProtection',
'TechSupport',
       'StreamingTV', 'StreamingMovies', 'Contract',
'PaperlessBilling',
     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

```
print(df.describe())
print(df.info())
```

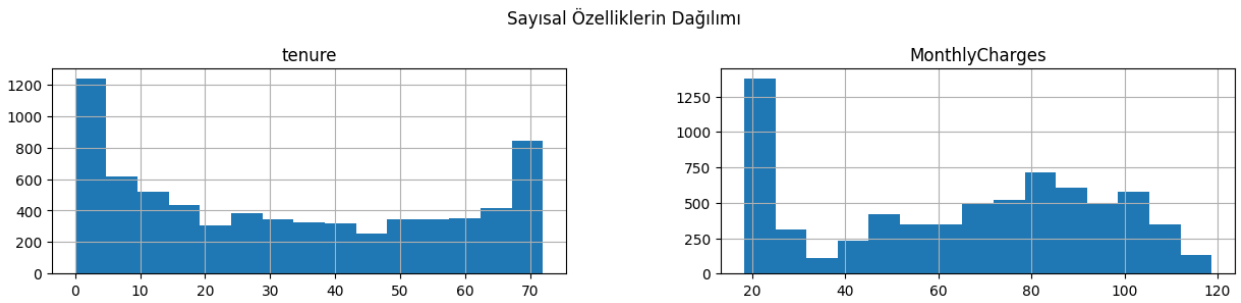
```
       SeniorCitizen       tenure  MonthlyCharges
count    7043.000000  7043.000000     7043.000000
mean        0.162147    32.371149       64.761692
std         0.368612    24.559481       30.090047
min         0.000000     0.000000       18.250000
25%         0.000000     9.000000       35.500000
50%         0.000000    29.000000       70.350000
75%         0.000000    55.000000       89.850000
max         1.000000    72.000000      118.750000
<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
None

# Eksik değerleri kontrol etme
print(df.isnull().sum())

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
```

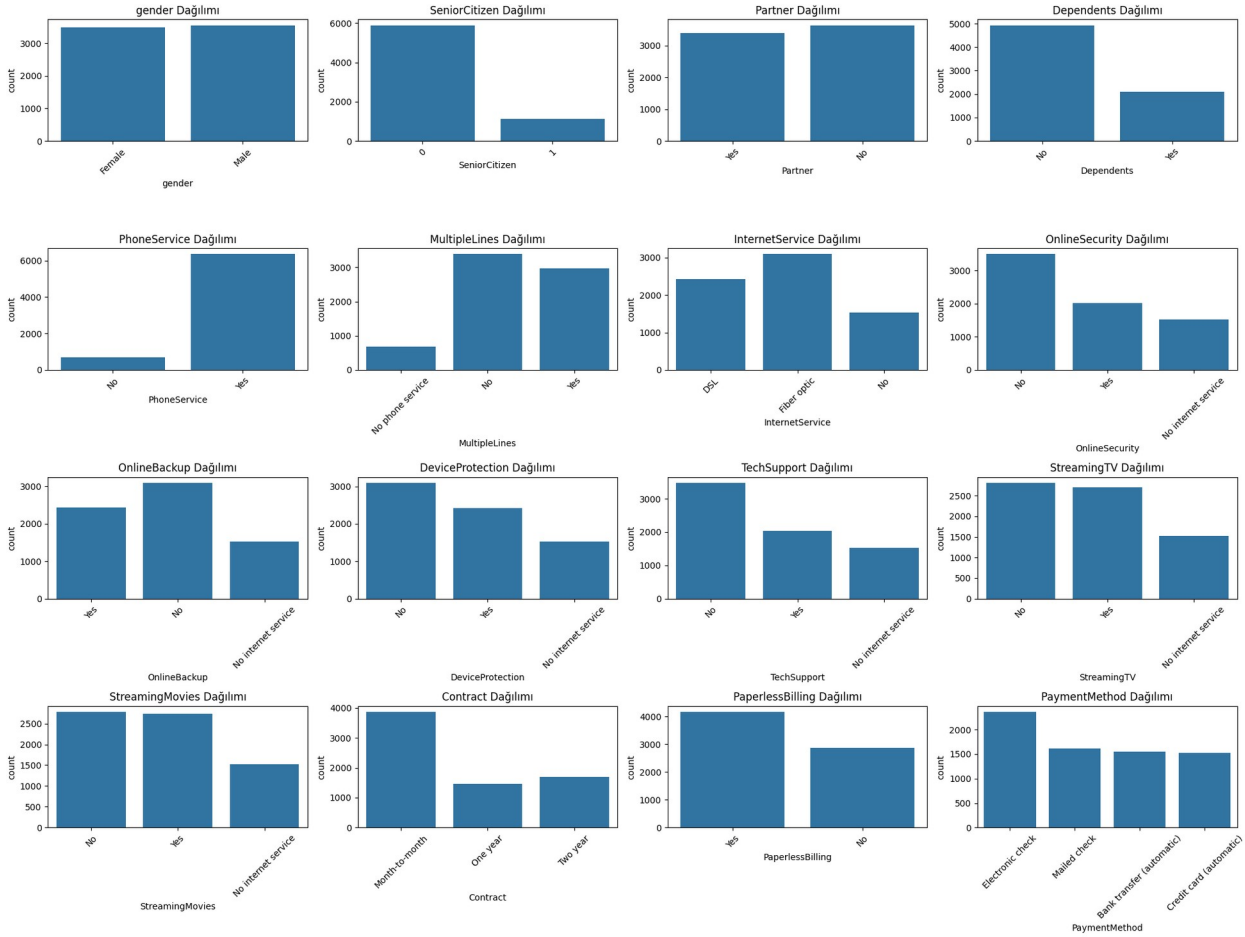
```
# Hedef değişken 'Churn' dağılımının kontrolü
sns.countplot(x='Churn', data=df)
plt.title('Churn Dağılımı')
plt.show()
```




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

# Sayısal özelliklerin dağılımının kontrolü
numerical_features = ['tenure', 'MonthlyCharges', 'TotalCharges']
df[numerical_features].hist(bins=15, figsize=(15, 6), layout=(2, 2))
plt.suptitle('Sayısal Özelliklerin Dağılımı')
plt.show()
```



```python
# Kategorik özelliklerin dağılımı
categorical_features = ['gender', 'SeniorCitizen', 'Partner',
'Dependents',
                        'PhoneService', 'MultipleLines',
'InternetService',
                        'OnlineSecurity', 'OnlineBackup',
'DeviceProtection',
                        'TechSupport', 'StreamingTV',
'StreamingMovies',
                        'Contract', 'PaperlessBilling',
'PaymentMethod']

plt.figure(figsize=(20, 15))
for i, feature in enumerate(categorical_features, 1):
    plt.subplot(4, 4, i)
    sns.countplot(x=feature, data=df)
    plt.title(f'{feature} Dağılımı')
    plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



Adım 2

```
# 'TotalCharges' sütununu sayısal verilere dönüştürme
df['TotalCharges'] = pd.to_numeric(df['TotalCharges'],
errors='coerce')

# Sayısal ve kategorik değişkenler
numerical_features = df.select_dtypes(include=['int64',
'float64']).columns.tolist()
categorical_features =
df.select_dtypes(include=['object']).columns.tolist()

print("Numerik Değişkenler:", numerical_features)
print("Kategorik Değişkenler:", categorical_features)
```

```
Numerik Değişkenler: ['SeniorCitizen', 'tenure', 'MonthlyCharges',
'TotalCharges', 'Churn']
Kategorik Değişkenler: ['customerID', 'gender', 'Partner',
'Dependents', 'PhoneService', 'MultipleLines', 'InternetService',
'OnlineSecurity', 'OnlineBackup', 'DeviceProtection', 'TechSupport',
'StreamingTV', 'StreamingMovies', 'Contract', 'PaperlessBilling',
'PaymentMethod']
```

```
# 'customerID' sütununu kaldırma
categorical_features.remove('customerID')

# Eksik değerler
print(df.isnull().sum())
```

```
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
```

```python
# Eksik değerleri ortalama ile doldurma
df['TotalCharges'].fillna(df['TotalCharges'].mean(), inplace=True)
print(df.isnull().sum())
```

```
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
```

Adım 3

```python
# Sayısal değişkenlerin analizi
print("Numerik Değişkenler:")
print(df[numerical_features].describe())
```

```
Numerik Değişkenler:
       SeniorCitizen       tenure  MonthlyCharges  TotalCharges
count    7043.000000  7043.000000     7043.000000   7043.000000
mean        0.162147    32.371149       64.761692   2283.300441
std         0.368612    24.559481       30.090047   2265.000258
min         0.000000     0.000000       18.250000     18.800000
25%         0.000000     9.000000       35.500000    402.225000
50%         0.000000    29.000000       70.350000   1400.550000
75%         0.000000    55.000000       89.850000   3786.600000
max         1.000000    72.000000      118.750000   8684.800000
```

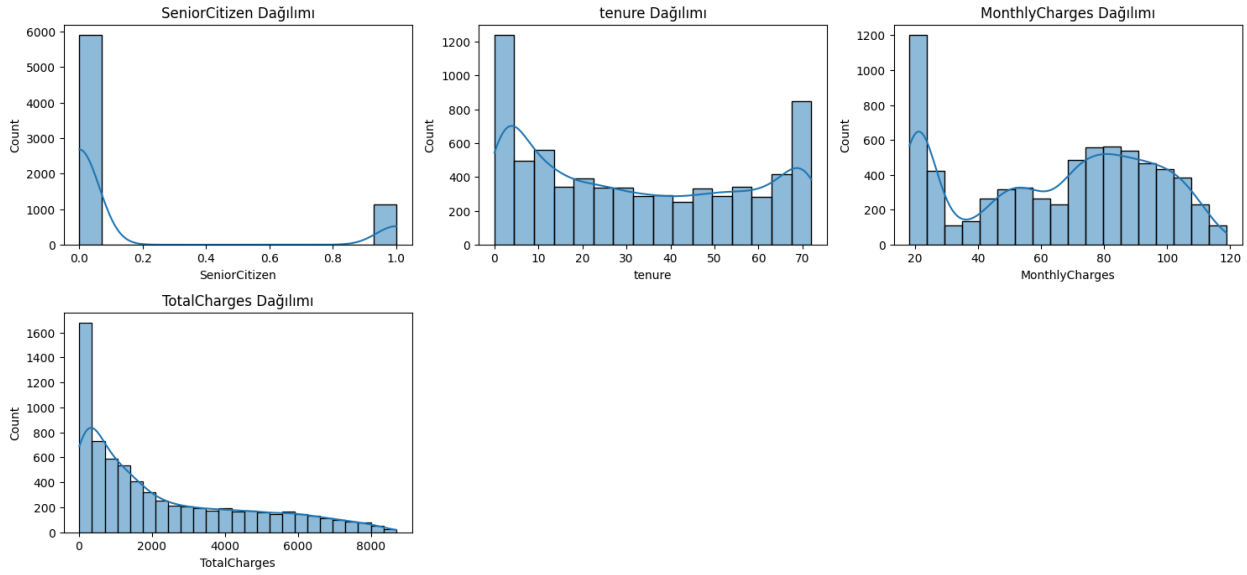
```python
# Kategorik değişkenlerin analizi
print("\nKategorik Değişkenler:")
for col in categorical_features:
    print(df[col].value_counts())
    print()
```

```
Kategorik Değişkenler:
customerID
7590-VHVEG    1
3791-LGQCY    1
6008-NAIXK    1
5956-YHHRX    1
5365-LLFYV    1
             ..
9796-MVYXX    1
2637-FKFSY    1
1552-AAGRX    1
4304-TSPVK    1
3186-AJIEK    1
Name: count, Length: 7043, dtype: int64

gender
Male      3555
Female    3488
Name: count, dtype: int64

Partner
No     3641
Yes    3402
Name: count, dtype: int64

Dependents
No     4933
Yes    2110
Name: count, dtype: int64

PhoneService
Yes    6361
No      682
Name: count, dtype: int64

MultipleLines
No                  3390
Yes                 2971
No phone service     682
Name: count, dtype: int64

InternetService
Fiber optic    3096
DSL            2421
No             1526
Name: count, dtype: int64

OnlineSecurity
No                     3498
```

```
Yes                     2019
No internet service    1526
Name: count, dtype: int64

OnlineBackup
No                      3088
Yes                     2429
No internet service    1526
Name: count, dtype: int64

DeviceProtection
No                      3095
Yes                     2422
No internet service    1526
Name: count, dtype: int64

TechSupport
No                      3473
Yes                     2044
No internet service    1526
Name: count, dtype: int64

StreamingTV
No                      2810
Yes                     2707
No internet service    1526
Name: count, dtype: int64

StreamingMovies
No                      2785
Yes                     2732
No internet service    1526
Name: count, dtype: int64

Contract
Month-to-month    3875
Two year          1695
One year          1473
Name: count, dtype: int64

PaperlessBilling
Yes    4171
No     2872
Name: count, dtype: int64

PaymentMethod
Electronic check             2365
Mailed check                 1612
Bank transfer (automatic)    1544
Credit card (automatic)      1522
```

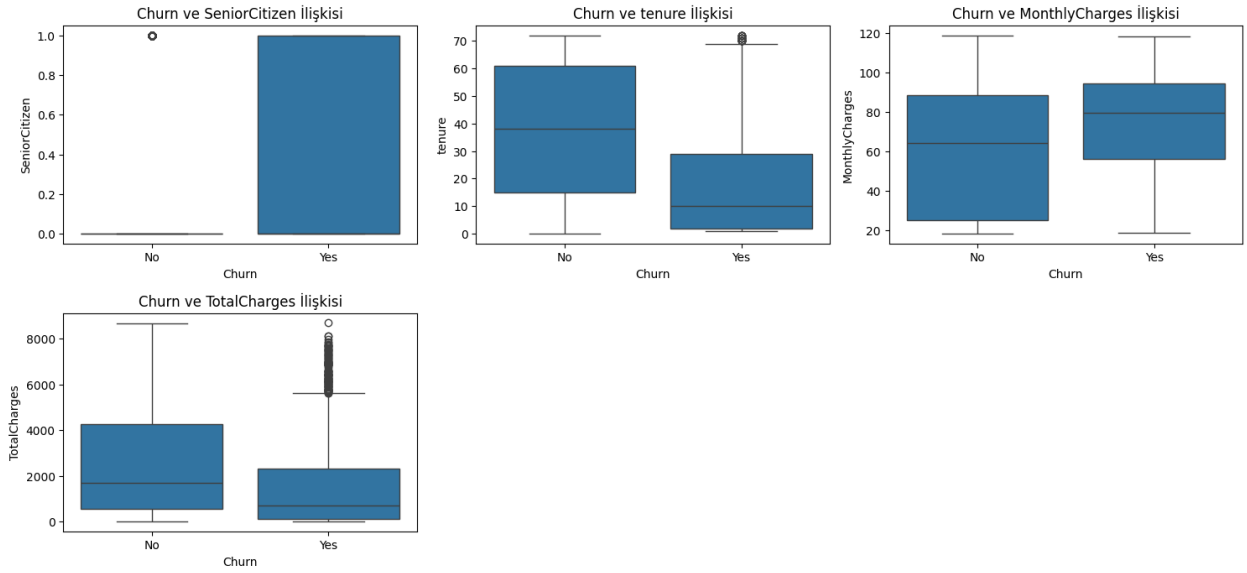
```
Name: count, dtype: int64

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

```python
# Sayısal değişkenlerin dağılımını görselleştirme
plt.figure(figsize=(15, 10))
for i, feature in enumerate(numerical_features, 1):
    plt.subplot(3, 3, i)
    sns.histplot(df[feature], kde=True)
    plt.title(f'{feature} Dağılımı')
plt.tight_layout()
plt.show()
```




```python
# Kategorik değişkenlerin dağılımını görselleştirme
plt.figure(figsize=(20, 20))
for i, feature in enumerate(categorical_features, 1):
    plt.subplot(5, 4, i)
    sns.countplot(x=feature, data=df)
    plt.title(f'{feature} Dağılımı')
    plt.xticks(rotation=45)
plt.tight_layout()
plt.show()

# Hedef değişken 'Churn' ile sayısal değişkenlerin ilişkisini
görselleştirme
plt.figure(figsize=(15, 10))
for i, feature in enumerate(numerical_features, 1):
    plt.subplot(3, 3, i)
```

```
    sns.boxplot(x='Churn', y=feature, data=df)
    plt.title(f'Churn ve {feature} İlişkisi')
plt.tight_layout()
plt.show()
```




```
# Hedef değişken 'Churn' ile kategorik değişkenlerin ilişkisini görselleştirme
plt.figure(figsize=(20, 20))
for i, feature in enumerate(categorical_features, 1):
    plt.subplot(5, 4, i)
    sns.countplot(x=feature, hue='Churn', data=df)
    plt.title(f'Churn ve {feature} İlişkisi')
    plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

Adım 4

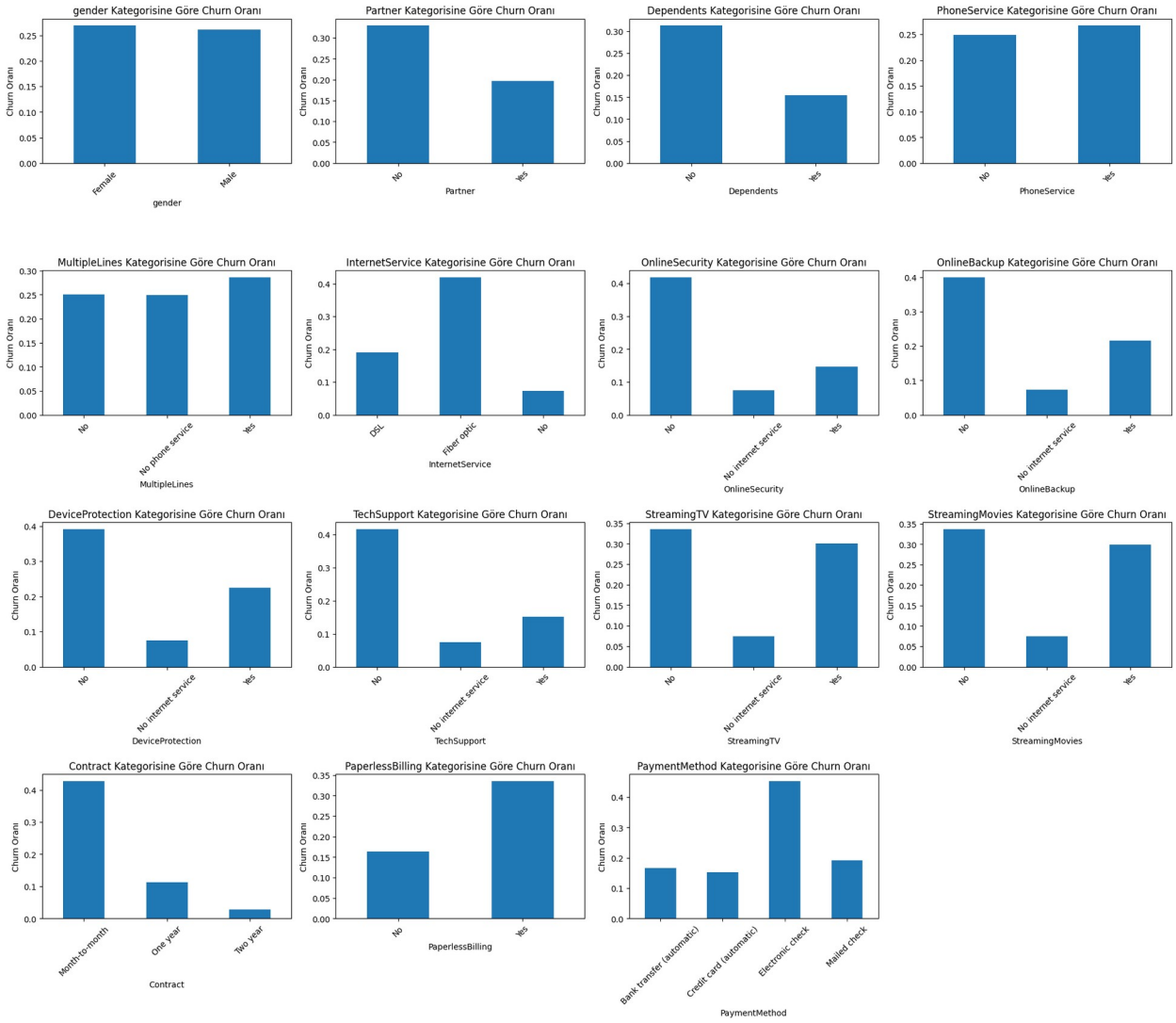
```
# Hedef değişken analizi: Kategorik değişkenlere göre hedef değişkenin ortalaması
print("Kategorik Değişkenlere Göre Hedef Değişkenin Ortalaması:")
for col in categorical_features:
    churn_mean = df.groupby(col)['Churn'].mean()
    print(churn_mean)
    print()
```

```
Kategorik Değişkenlere Göre Hedef Değişkenin Ortalaması:
gender
Female    0.269209
Male      0.261603
Name: Churn, dtype: float64
```

```
Partner
No     0.329580
Yes    0.196649
Name: Churn, dtype: float64

Dependents
No     0.312791
Yes    0.154502
Name: Churn, dtype: float64

PhoneService
No     0.249267
Yes    0.267096
Name: Churn, dtype: float64

MultipleLines
No                  0.250442
No phone service    0.249267
Yes                 0.286099
Name: Churn, dtype: float64

InternetService
DSL            0.189591
Fiber optic    0.418928
No             0.074050
Name: Churn, dtype: float64

OnlineSecurity
No                     0.417667
No internet service    0.074050
Yes                    0.146112
Name: Churn, dtype: float64

OnlineBackup
No                     0.399288
No internet service    0.074050
Yes                    0.215315
Name: Churn, dtype: float64

DeviceProtection
No                     0.391276
No internet service    0.074050
Yes                    0.225021
Name: Churn, dtype: float64

TechSupport
No                     0.416355
No internet service    0.074050
Yes                    0.151663
```

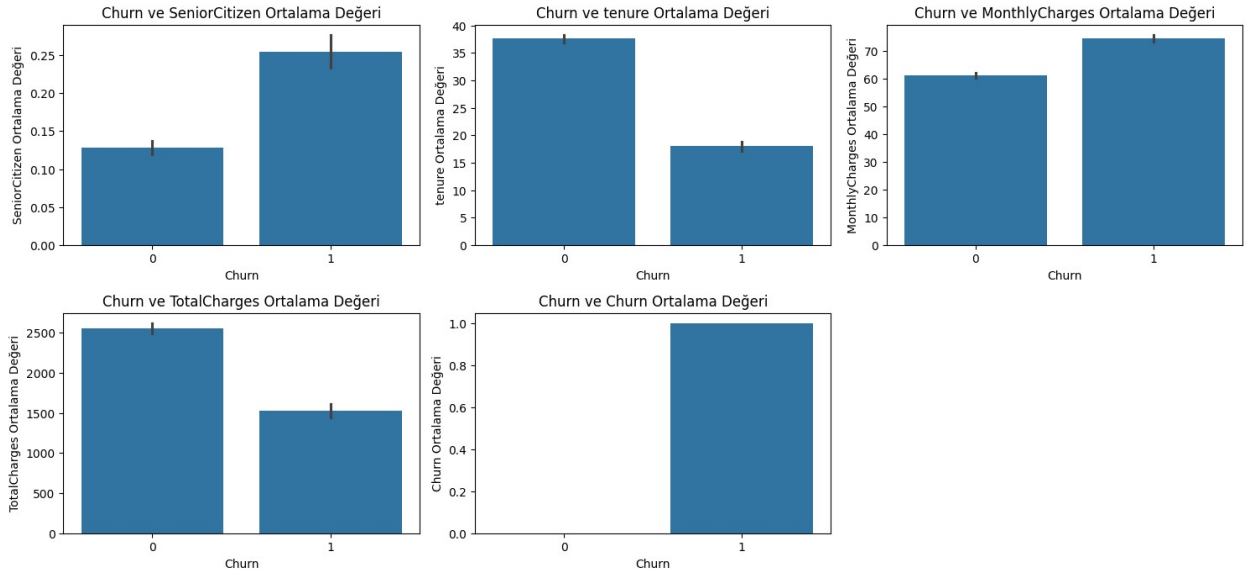
```
Name: Churn, dtype: float64

StreamingTV
No                     0.335231
No internet service    0.074050
Yes                    0.300702
Name: Churn, dtype: float64

StreamingMovies
No                     0.336804
No internet service    0.074050
Yes                    0.299414
Name: Churn, dtype: float64

Contract
Month-to-month    0.427097
One year          0.112695
Two year          0.028319
Name: Churn, dtype: float64

PaperlessBilling
No     0.163301
Yes    0.335651
Name: Churn, dtype: float64

PaymentMethod
Bank transfer (automatic)    0.167098
Credit card (automatic)      0.152431
Electronic check             0.452854
Mailed check                 0.191067
Name: Churn, dtype: float64


# Hedef değişken analizi: Hedef değişkene göre sayısal değişkenlerin ortalaması
print("Hedef Değişkene Göre Sayısal Değişkenlerin Ortalaması:")
churn_numeric_means = df.groupby('Churn')[numerical_features].mean()
print(churn_numeric_means)

Hedef Değişkene Göre Sayısal Değişkenlerin Ortalaması:
       SeniorCitizen     tenure  MonthlyCharges  TotalCharges  Churn
Churn
0           0.128721  37.569965       61.265124   2555.344141    0.0
1           0.254682  17.979133       74.441332   1531.796094    1.0

# Görselleştirme: Kategorik değişkenlere göre hedef değişkenin ortalaması
plt.figure(figsize=(20, 20))
for i, col in enumerate(categorical_features, 1):
    plt.subplot(5, 4, i)
    churn_mean = df.groupby(col)['Churn'].mean()
```

```
    churn_mean.plot(kind='bar')
    plt.title(f'{col} Kategorisine Göre Churn Oranı')
    plt.ylabel('Churn Oranı')
    plt.xlabel(col)
    plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```




```
# Görselleştirme: Hedef değişkene göre sayısal değişkenlerin ortalaması
plt.figure(figsize=(15, 10))
for i, feature in enumerate(numerical_features, 1):
    plt.subplot(3, 3, i)
    sns.barplot(x='Churn', y=feature, data=df)
    plt.title(f'Churn ve {feature} Ortalama Değeri')
    plt.ylabel(f'{feature} Ortalama Değeri')
```

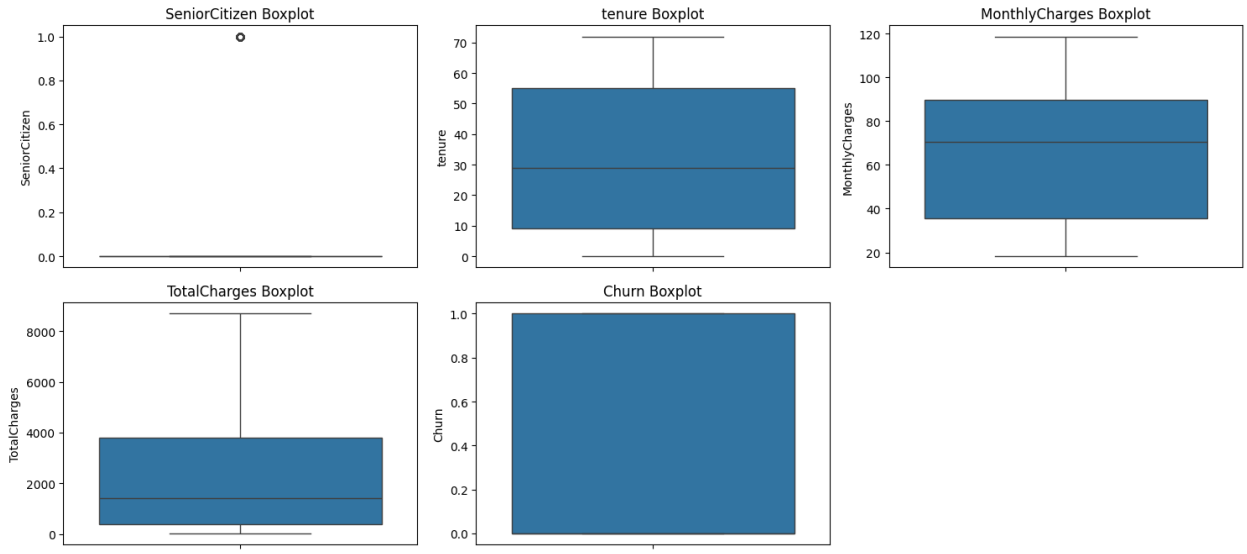
```
    plt.xlabel('Churn')
plt.tight_layout()
plt.show()
```



Adım 5

```
# boxplot ile Aykırı gözlemleri bulmak ve görselleştirme
plt.figure(figsize=(15, 10))
for i, feature in enumerate(numerical_features, 1):
    plt.subplot(3, 3, i)
    sns.boxplot(y=df[feature])
    plt.title(f'{feature} Boxplot')
    q1 = df[feature].quantile(0.25)
    q3 = df[feature].quantile(0.75)
    iqr = q3 - q1
    lower_limit = q1 - 1.5 * iqr
    upper_limit = q3 + 1.5 * iqr
    print(f"{feature}: Lower Limit = {lower_limit}, Upper Limit =
{upper_limit}")
plt.tight_layout()
plt.show()
```

```
SeniorCitizen: Lower Limit = 0.0, Upper Limit = 0.0
tenure: Lower Limit = -60.0, Upper Limit = 124.0
MonthlyCharges: Lower Limit = -46.02499999999999, Upper Limit =
171.375
TotalCharges: Lower Limit = -4688.481250000001, Upper Limit =
8884.66875
Churn: Lower Limit = -1.5, Upper Limit = 2.5
```

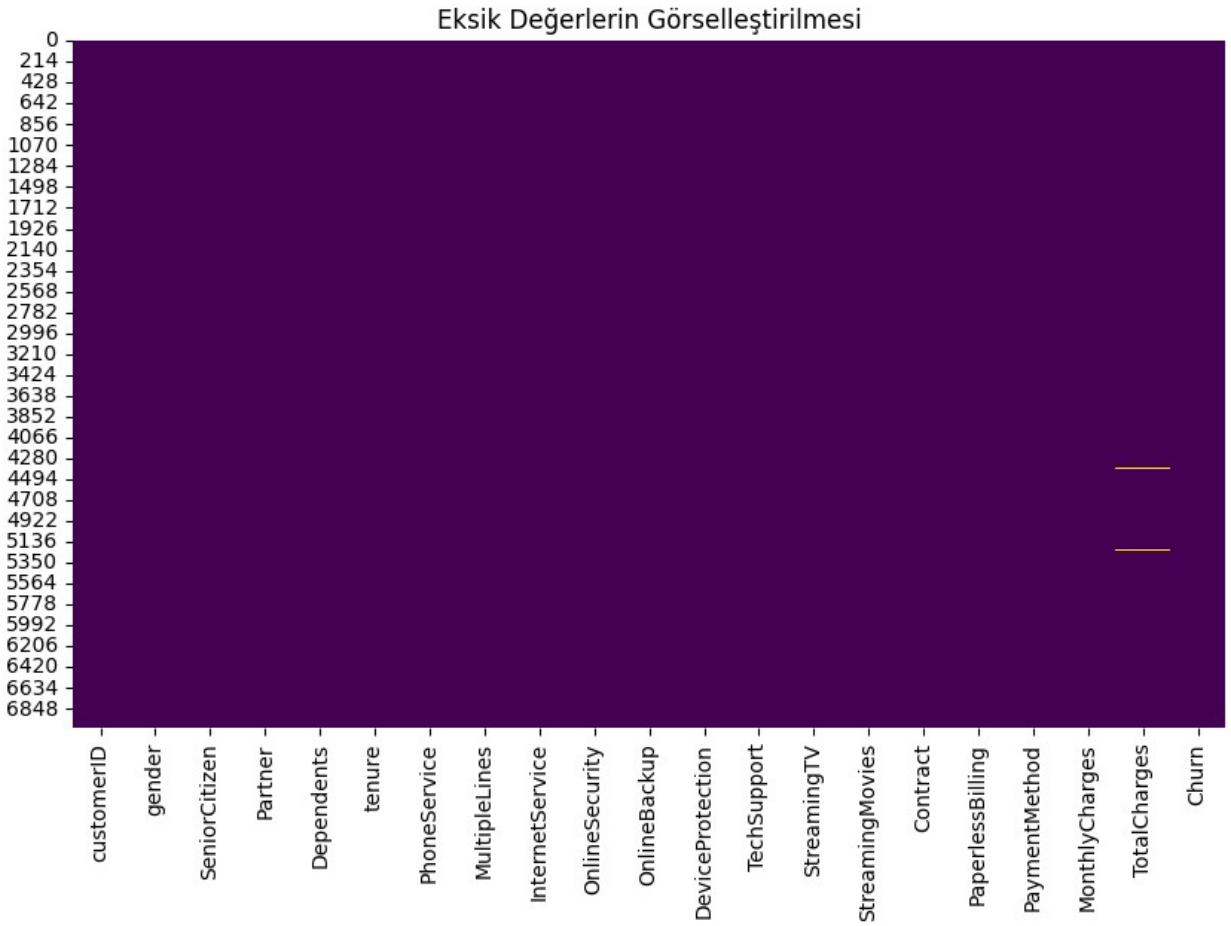


Adım 6

```
# Eksik değerler
missing_values = df.isnull().sum()

# Eksik değerlerin yüzdesi
missing_percentage = (missing_values / len(df)) * 100

# Eksik değerleri ve yüzdelerini bir DataFrame olarak görüntüleme
missing_data = pd.DataFrame({'Missing Values': missing_values,
'Percentage': missing_percentage})
print(missing_data)
```

	Missing Values	Percentage
customerID	0	0.000000
gender	0	0.000000
SeniorCitizen	0	0.000000
Partner	0	0.000000
Dependents	0	0.000000
tenure	0	0.000000
PhoneService	0	0.000000
MultipleLines	0	0.000000
InternetService	0	0.000000
OnlineSecurity	0	0.000000
OnlineBackup	0	0.000000
DeviceProtection	0	0.000000
TechSupport	0	0.000000
StreamingTV	0	0.000000
StreamingMovies	0	0.000000
Contract	0	0.000000
PaperlessBilling	0	0.000000
PaymentMethod	0	0.000000
MonthlyCharges	0	0.000000

```
TotalCharges                 11    0.156183
Churn                         0    0.000000

# Eksik değerleri görselleştirme
plt.figure(figsize=(10, 6))
sns.heatmap(df.isnull(), cbar=False, cmap='viridis')
plt.title('Eksik Değerlerin Görselleştirilmesi')
plt.show()
```



Adım 7

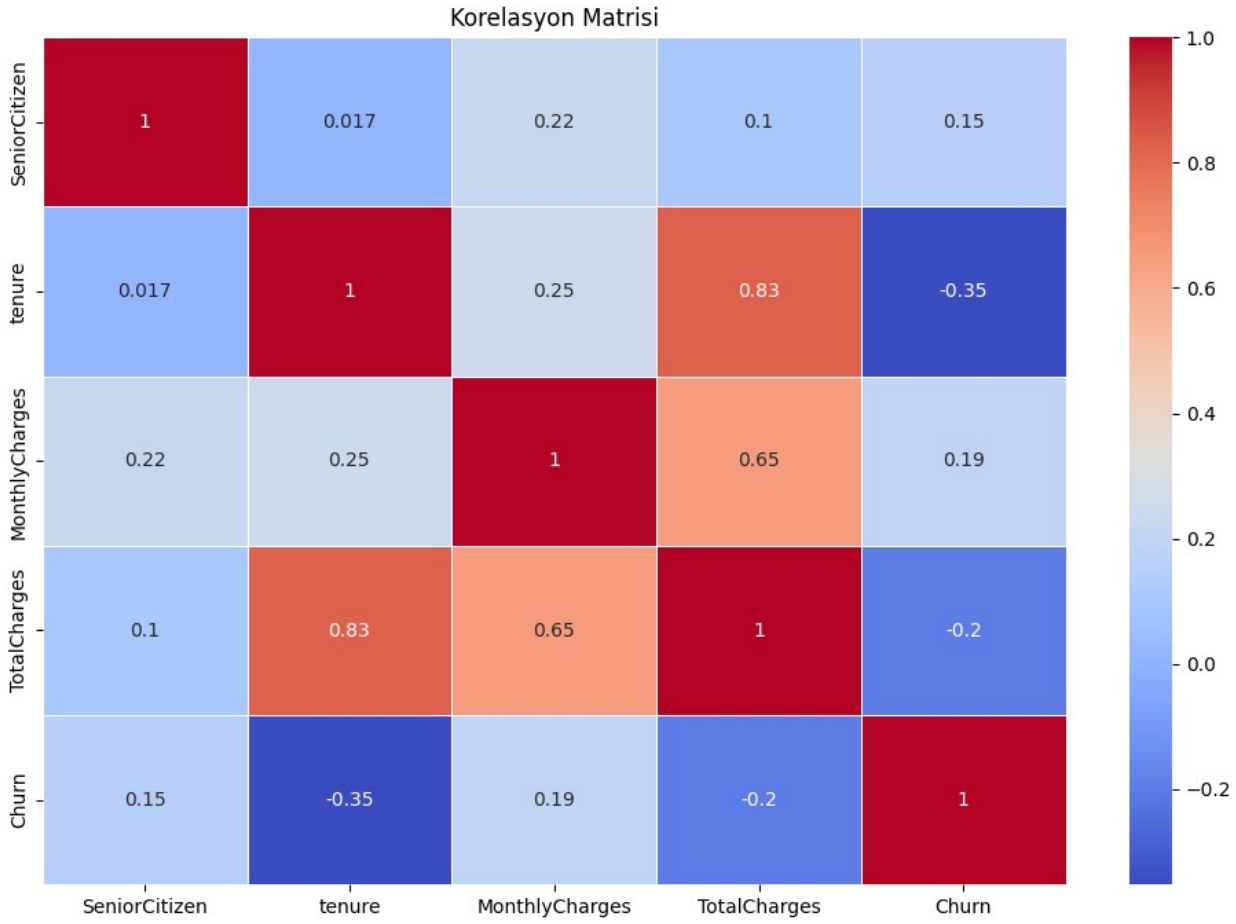
```
correlation_matrix = df[numerical_features].corr()

print(correlation_matrix)
```

```
                SeniorCitizen    tenure  MonthlyCharges  TotalCharges
\
SeniorCitizen        1.000000  0.016567        0.220173      0.102411

tenure               0.016567  1.000000        0.247900      0.825880

MonthlyCharges       0.220173  0.247900        1.000000      0.651065
```

```
TotalCharges          0.102411  0.825880         0.651065       1.000000

Churn                 0.150889 -0.352229         0.193356      -0.199484
```

```
                     Churn
SeniorCitizen     0.150889
tenure           -0.352229
MonthlyCharges    0.193356
TotalCharges     -0.199484
Churn             1.000000
```

```
# Korelasyon matrisi
plt.figure(figsize=(12, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm',
linewidths=0.5)
plt.title('Korelasyon Matrisi')
plt.show()
```



Task 2: Feature Engineering

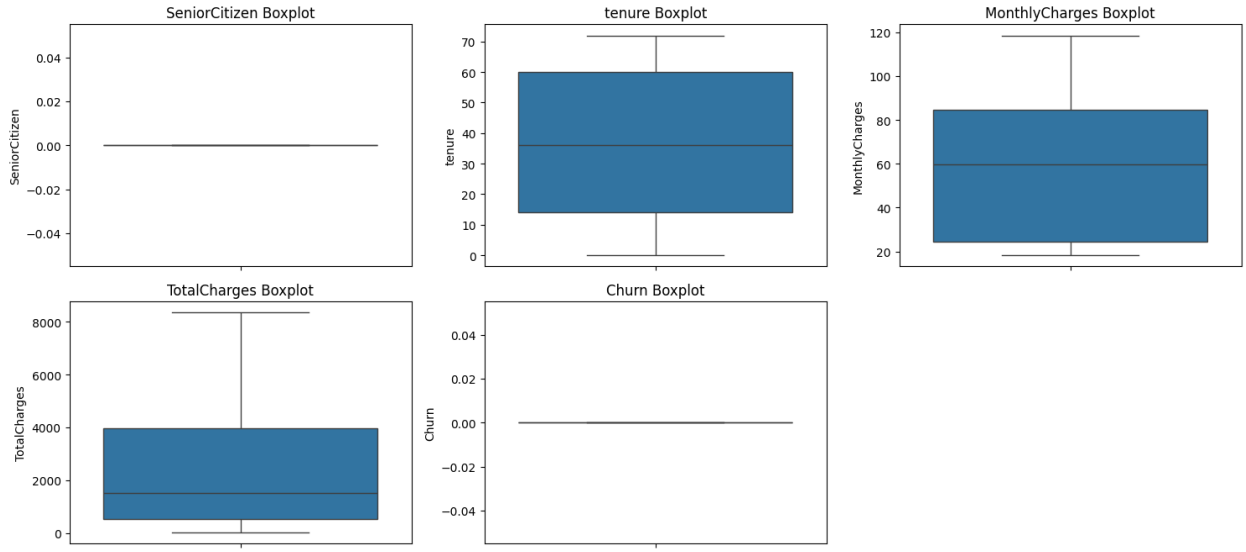
Adım 1

```
# Eksik değerleri doldurma
df['TotalCharges'].fillna(df['TotalCharges'].mean(), inplace=True)

# Aykırı değerleri kaldırma fonksiyonu
def remove_outliers(df, column):
    q1 = df[column].quantile(0.25)
    q3 = df[column].quantile(0.75)
    iqr = q3 - q1
    lower_limit = q1 - 1.5 * iqr
    upper_limit = q3 + 1.5 * iqr
    df = df[(df[column] >= lower_limit) & (df[column] <= upper_limit)]
    return df

# Aykırı değerleri kaldırma işlemi
for feature in numerical_features:
    df = remove_outliers(df, feature)

# Aykırı değerler kaldırıldıktan sonra veri
print("Aykırı değerler kaldırıldıktan sonra veri seti:")
print(df.describe())
```

```
Aykırı değerler kaldırıldıktan sonra veri seti:
       SeniorCitizen       tenure  MonthlyCharges  TotalCharges
Churn
count         4490.0  4490.000000     4490.000000   4490.000000
4490.0
mean             0.0    36.766370       58.386615   2391.577184
0.0
std              0.0    24.120262       30.810165   2252.570162
0.0
min              0.0     0.000000       18.250000     18.800000
0.0
25%              0.0    14.000000       24.400000    522.500000
0.0
50%              0.0    36.000000       59.650000   1506.700000
0.0
75%              0.0    60.000000       84.750000   3964.512500
0.0
max              0.0    72.000000      118.600000   8349.700000
0.0
```

```
# Kalan veriyi görselleştirme
plt.figure(figsize=(15, 10))
for i, feature in enumerate(numerical_features, 1):
    plt.subplot(3, 3, i)
    sns.boxplot(y=df[feature])
```

```
    plt.title(f'{feature} Boxplot')
plt.tight_layout()
plt.show()
```



Adım 2: Yeni değişkenler oluşturma

```
# Müşterinin toplam gelirini hesaplayan değişken
df['TotalRevenue'] = df['tenure'] * df['MonthlyCharges']

# Müşterinin ortalama aylık gelirini hesaplayan değişken
df['AverageMonthlyRevenue'] = df['TotalRevenue'] / df['tenure']

# Müşterinin uzun vadeli olup olmadığını belirten değişken
# 12 aydan fazla kalan müşteriler uzun vadeli olarak kabul edilecektir
df['LongTermCustomer'] = np.where(df['tenure'] > 12, 1, 0)

# Aylık ücretin ortalama aylık ücretin üzerinde olup olmadığını
belirten değişken
average_monthly_charge = df['MonthlyCharges'].mean()
df['AboveAverageMonthlyCharge'] = np.where(df['MonthlyCharges'] >
average_monthly_charge, 1, 0)

# Yeni değişkenler
print(df[['TotalRevenue', 'AverageMonthlyRevenue', 'LongTermCustomer',
'AboveAverageMonthlyCharge']].head())
```

```
   TotalRevenue  AverageMonthlyRevenue  LongTermCustomer  \
0         29.85                  29.85                 0
1       1936.30                  56.95                 1
3       1903.50                  42.30                 1
6       1960.20                  89.10                 1
7        297.50                  29.75                 0
```

```
   AboveAverageMonthlyCharge
0                          0
1                          0
3                          0
6                          1
7                          0
```

```
# Yeni değişkenlerin özet istatistikleri
print(df[['TotalRevenue', 'AverageMonthlyRevenue', 'LongTermCustomer',
'AboveAverageMonthlyCharge']].describe())
```

```
       TotalRevenue  AverageMonthlyRevenue  LongTermCustomer  \
count   4490.000000            4479.000000       4490.000000
mean    2386.270635              58.428288          0.767038
std     2255.937927              30.815918          0.422765
min        0.000000              18.250000          0.000000
25%      518.712500              24.400000          1.000000
50%     1490.400000              59.700000          1.000000
75%     3979.137500              84.800000          1.000000
max     8467.200000             118.600000          1.000000

       AboveAverageMonthlyCharge
count                4490.000000
mean                    0.512695
std                     0.499894
min                     0.000000
25%                     0.000000
50%                     1.000000
75%                     1.000000
max                     1.000000
```

Adım 3

```
df.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
3  7795-CFOCW    Male              0      No         No      45
No
6  1452-KIOVK    Male              0      No        Yes      22
Yes
7  6713-OKOMC  Female              0      No         No      10
No

      MultipleLines InternetService OnlineSecurity  ...
Contract  \
```

```
0  No phone service             DSL             No  ...  Month-to-
month
1                No             DSL            Yes  ...        One
year
3  No phone service             DSL            Yes  ...        One
year
6               Yes     Fiber optic             No  ...  Month-to-
month
7  No phone service             DSL            Yes  ...  Month-to-
month

  PaperlessBilling              PaymentMethod MonthlyCharges
TotalCharges  \
0              Yes           Electronic check          29.85
29.85
1               No               Mailed check          56.95
1889.50
3               No  Bank transfer (automatic)          42.30
1840.75
6              Yes    Credit card (automatic)          89.10
1949.40
7               No               Mailed check          29.75
301.90

  Churn TotalRevenue AverageMonthlyRevenue  LongTermCustomer  \
0     0        29.85                 29.85                 0
1     0      1936.30                 56.95                 1
3     0      1903.50                 42.30                 1
6     0      1960.20                 89.10                 1
7     0       297.50                 29.75                 0

   AboveAverageMonthlyCharge
0                          0
1                          0
3                          0
6                          1
7                          0

[5 rows x 25 columns]
```

```python
# Label Encoding (Evet/Hayır gibi ikili kategoriler için)
binary_features = ['Partner', 'Dependents', 'PhoneService',
'PaperlessBilling']
le = LabelEncoder()
for feature in binary_features:
    df[feature] = le.fit_transform(df[feature])

# One-Hot Encoding (Çoklu kategoriler için)
df = pd.get_dummies(df, columns=categorical_features, drop_first=True)
```

```
df.head()
```

```
   customerID  SeniorCitizen  tenure  MonthlyCharges  TotalCharges
Churn  \
0  7590-VHVEG              0       1           29.85         29.85
0
1  5575-GNVDE              0      34           56.95       1889.50
0
3  7795-CFOCW              0      45           42.30       1840.75
0
6  1452-KIOVK              0      22           89.10       1949.40
0
7  6713-OKOMC              0      10           29.75        301.90
0

   TotalRevenue  AverageMonthlyRevenue  LongTermCustomer  \
0         29.85                  29.85                 0
1       1936.30                  56.95                 1
3       1903.50                  42.30                 1
6       1960.20                  89.10                 1
7        297.50                  29.75                 0

   AboveAverageMonthlyCharge  ...  StreamingTV_No internet service  \
0                          0  ...                             False
1                          0  ...                             False
3                          0  ...                             False
6                          1  ...                             False
7                          0  ...                             False

   StreamingTV_Yes  StreamingMovies_No internet service
StreamingMovies_Yes  \
0            False                                False
False
1            False                                False
False
3            False                                False
False
6             True                                False
False
7            False                                False
False

   Contract_One year  Contract_Two year  PaperlessBilling_1  \
0              False              False                True
1               True              False               False
3               True              False               False
6              False              False                True
7              False              False               False

   PaymentMethod_Credit card (automatic)  PaymentMethod_Electronic
```

```
check  \
0                                    False
True
1                                    False
False
3                                    False
False
6                                     True
False
7                                    False
False

   PaymentMethod_Mailed check
0                      False
1                       True
3                      False
6                      False
7                       True

[5 rows x 36 columns]
```

Adım 4

```
# Sayısal değişkenleri standartlaştırma
scaler = StandardScaler()
df[numerical_features] = scaler.fit_transform(df[numerical_features])

df.head()

   customerID  SeniorCitizen    tenure  MonthlyCharges  TotalCharges
Churn  \
0  7590-VHVEG            0.0 -1.483000       -0.926311     -1.048576
0.0
1  5575-GNVDE            0.0 -0.114703       -0.046633     -0.222916
0.0
3  7795-CFOCW            0.0  0.341395       -0.522179     -0.244560
0.0
6  1452-KIOVK            0.0 -0.612266        0.996970     -0.196321
0.0
7  6713-OKOMC            0.0 -1.109828       -0.929557     -0.927789
0.0

   TotalRevenue  AverageMonthlyRevenue  LongTermCustomer  \
0         29.85                  29.85                 0
1       1936.30                  56.95                 1
3       1903.50                  42.30                 1
6       1960.20                  89.10                 1
7        297.50                  29.75                 0

   AboveAverageMonthlyCharge  ...  StreamingTV_No internet service  \
```

```
0                           0  ...                                False
1                           0  ...                                False
3                           0  ...                                False
6                           1  ...                                False
7                           0  ...                                False

   StreamingTV_Yes  StreamingMovies_No internet service
StreamingMovies_Yes  \
0            False                                False
False
1            False                                False
False
3            False                                False
False
6             True                                False
False
7            False                                False
False

   Contract_One year  Contract_Two year  PaperlessBilling_1  \
0              False              False                True
1               True              False               False
3               True              False               False
6              False              False                True
7              False              False               False

   PaymentMethod_Credit card (automatic)  PaymentMethod_Electronic
check  \
0                                  False
True
1                                  False
False
3                                  False
False
6                                   True
False
7                                  False
False

   PaymentMethod_Mailed check
0                      False
1                       True
3                      False
6                      False
7                       True

[5 rows x 36 columns]
```

Adım 5

```
df.head()
```

```
   customerID  SeniorCitizen    tenure  MonthlyCharges  TotalCharges
Churn  \
0  7590-VHVEG            0.0 -1.483000       -0.926311     -1.048576
0.0
1  5575-GNVDE            0.0 -0.114703       -0.046633     -0.222916
0.0
3  7795-CFOCW            0.0  0.341395       -0.522179     -0.244560
0.0
6  1452-KIOVK            0.0 -0.612266        0.996970     -0.196321
0.0
7  6713-OKOMC            0.0 -1.109828       -0.929557     -0.927789
0.0

   TotalRevenue  AverageMonthlyRevenue  LongTermCustomer  \
0         29.85                  29.85                 0
1       1936.30                  56.95                 1
3       1903.50                  42.30                 1
6       1960.20                  89.10                 1
7        297.50                  29.75                 0

   AboveAverageMonthlyCharge  ...  StreamingTV_No internet service  \
0                          0  ...                             False
1                          0  ...                             False
3                          0  ...                             False
6                          1  ...                             False
7                          0  ...                             False

   StreamingTV_Yes  StreamingMovies_No internet service
StreamingMovies_Yes  \
0            False                                False
False
1            False                                False
False
3            False                                False
False
6             True                                False
False
7            False                                False
False

   Contract_One year  Contract_Two year  PaperlessBilling_1  \
0              False              False                True
1               True              False               False
3               True              False               False
6              False              False                True
7              False              False               False

   PaymentMethod_Credit card (automatic)  PaymentMethod_Electronic
```

```
check  \
0                                    False
True
1                                    False
False
3                                    False
False
6                                     True
False
7                                    False
False

   PaymentMethod_Mailed check
0                      False
1                       True
3                      False
6                      False
7                       True

[5 rows x 36 columns]
```

```
# Eksik değerleri tekrar kontrol edin
print("Eksik değerler:")
print(df.isnull().sum())
```

```
Eksik değerler:
customerID                                0
SeniorCitizen                             0
tenure                                    0
MonthlyCharges                            0
TotalCharges                              0
Churn                                     0
TotalRevenue                              0
AverageMonthlyRevenue                    11
LongTermCustomer                          0
AboveAverageMonthlyCharge                 0
gender_Male                               0
Partner_1                                 0
Dependents_1                              0
PhoneService_1                            0
MultipleLines_No phone service            0
MultipleLines_Yes                         0
InternetService_Fiber optic               0
InternetService_No                        0
OnlineSecurity_No internet service        0
OnlineSecurity_Yes                        0
OnlineBackup_No internet service          0
OnlineBackup_Yes                          0
DeviceProtection_No internet service      0
DeviceProtection_Yes                      0
```

```
TechSupport_No internet service           0
TechSupport_Yes                           0
StreamingTV_No internet service           0
StreamingTV_Yes                           0
StreamingMovies_No internet service       0
StreamingMovies_Yes                       0
Contract_One year                         0
Contract_Two year                         0
PaperlessBilling_1                        0
PaymentMethod_Credit card (automatic)     0
PaymentMethod_Electronic check            0
PaymentMethod_Mailed check                0
dtype: int64
```

```
# Eksik değerleri doldurma
df["AverageMonthlyRevenue"].fillna(df["AverageMonthlyRevenue"].mean(),
inplace=True)

# Özellikler ve hedef değişkeni belirleme
X = df.drop(columns=['Churn', 'customerID'])
y = df['Churn']

X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)

from sklearn.tree import DecisionTreeClassifier

model = DecisionTreeClassifier(random_state=42)
model.fit(X_train, y_train)
```

```
DecisionTreeClassifier(random_state=42)
```

```
y_pred = model.predict(X_test)

accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
class_report = classification_report(y_test, y_pred)

print("Model Doğruluğu:", accuracy)
print("Confusion Matrix:")
print(conf_matrix)
print("Classification Report:")
print(class_report)
```

```
Model Doğruluğu: 1.0
Confusion Matrix:
[[898]]
Classification Report:
              precision    recall  f1-score   support

         0.0       1.00      1.00      1.00       898
```

```
    accuracy                           1.00       898
   macro avg       1.00      1.00      1.00       898
weighted avg       1.00      1.00      1.00       898
```