```python
# Abdurrahman Bulut

import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, classification_report
```

Adım 1

```python
df = pd.read_csv("diabetes.csv")

# 1. Başlık Bilgisi
print(df.columns)
```

```
Index(['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness',
'Insulin',
       'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome'],
      dtype='object')
```

```python
print(df.head())
```

```
   Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin
BMI  \
0            6      148             72             35        0  33.6

1            1       85             66             29        0  26.6

2            8      183             64              0        0  23.3

3            1       89             66             23       94  28.1

4            0      137             40             35      168  43.1


   DiabetesPedigreeFunction  Age  Outcome
0                     0.627   50        1
1                     0.351   31        0
2                     0.672   32        1
3                     0.167   21        0
4                     2.288   33        1
```

```python
print(df.dtypes)
```

```
Pregnancies                   int64
Glucose                       int64
```

```
BloodPressure                 int64
SkinThickness                 int64
Insulin                       int64
BMI                         float64
DiabetesPedigreeFunction    float64
Age                           int64
Outcome                       int64
dtype: object
```

Eksik Değerler

```
print(df.isnull().sum())
```

```
Pregnancies                 0
Glucose                     0
BloodPressure               0
SkinThickness               0
Insulin                     0
BMI                         0
DiabetesPedigreeFunction    0
Age                         0
Outcome                     0
dtype: int64
```

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

```
       Pregnancies     Glucose  BloodPressure  SkinThickness
Insulin  \
count   768.000000  768.000000     768.000000     768.000000
768.000000
mean      3.845052  120.894531      69.105469      20.536458
79.799479
std       3.369578   31.972618      19.355807      15.952218
115.244002
min       0.000000    0.000000       0.000000       0.000000
0.000000
25%       1.000000   99.000000      62.000000       0.000000
0.000000
50%       3.000000  117.000000      72.000000      23.000000
30.500000
75%       6.000000  140.250000      80.000000      32.000000
127.250000
max      17.000000  199.000000     122.000000      99.000000
846.000000

              BMI  DiabetesPedigreeFunction         Age     Outcome
count  768.000000                768.000000  768.000000  768.000000
mean    31.992578                  0.471876   33.240885    0.348958
std      7.884160                  0.331329   11.760232    0.476951
min      0.000000                  0.078000   21.000000    0.000000
```

```
25%     27.300000                 0.243750   24.000000    0.000000
50%     32.000000                 0.372500   29.000000    0.000000
75%     36.600000                 0.626250   41.000000    1.000000
max     67.100000                 2.420000   81.000000    1.000000
```

```
print("\nKategorik Değişkenlerin Dağılımı:")
for column in df.select_dtypes(include=['object']).columns:
    print(f"\n{column}:")
    print(df[column].value_counts())
```

```
Kategorik Değişkenlerin Dağılımı:
```

Adım 2

```
# Numerik değişkenler
numeric_cols = df.select_dtypes(include=['int64',
'float64']).columns.tolist()

# Kategorik değişkenler
categorical_cols =
df.select_dtypes(include=['object']).columns.tolist()

print("Numerik Değişkenler:")
print(numeric_cols)

print("\nKategorik Değişkenler:")
print(categorical_cols)
```

```
Numerik Değişkenler:
['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness',
'Insulin', 'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome']

Kategorik Değişkenler:
[]
```

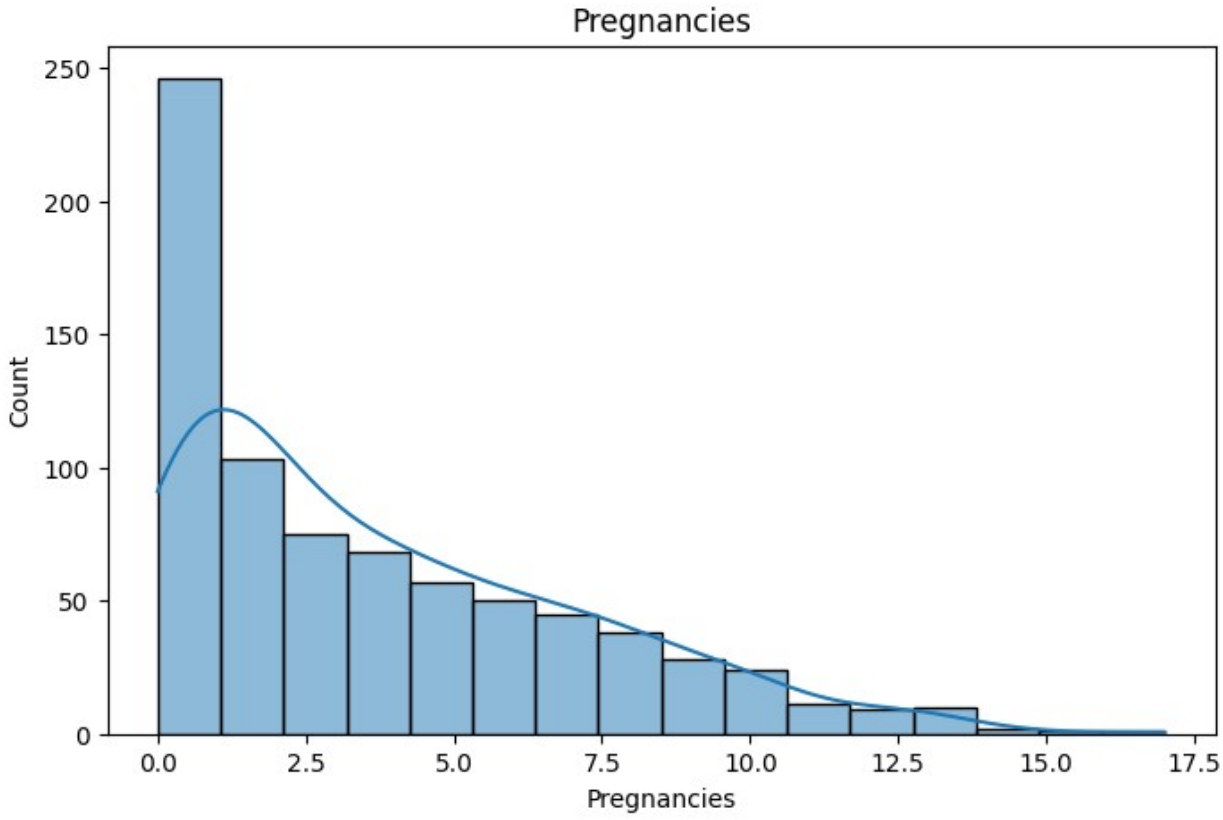
Adım 3

```
# Numerik değişkenlerin analizi
print("Numerik Değişkenlerin Analizi:")
for col in numeric_cols:
    print("\n" + col + ":")
    print("Ortalama:", df[col].mean())
    print("Medyan:", df[col].median())
    print("Standart Sapma:", df[col].std())
    print("Minimum Değer:", df[col].min())
    print("Maksimum Değer:", df[col].max())
```

```
    print("25. Yüzdelik:", df[col].quantile(0.25))
    print("50. Yüzdelik (Medyan):", df[col].quantile(0.50))
    print("75. Yüzdelik:", df[col].quantile(0.75))
    print("Histogram:")
    plt.figure(figsize=(8, 5))
    sns.histplot(df[col], kde=True)
    plt.title(col)
    plt.show()
```

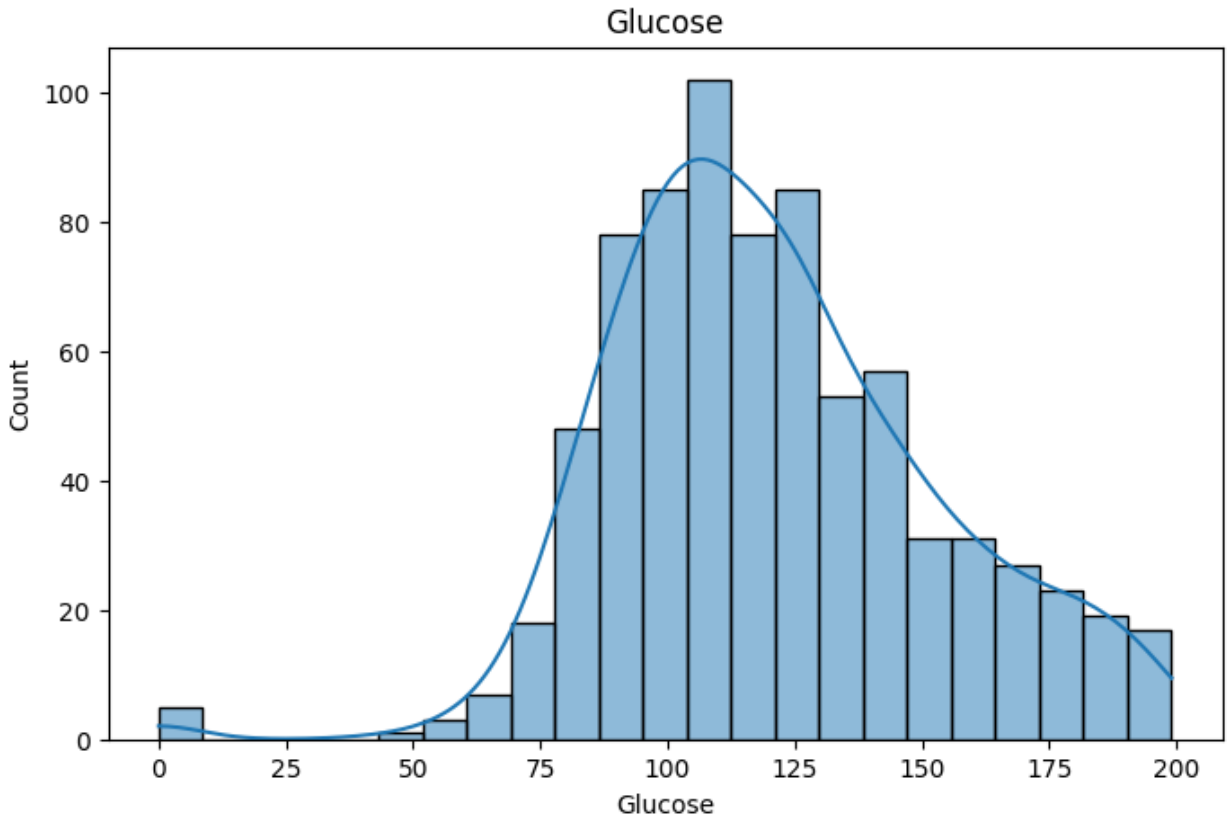
```
Numerik Değişkenlerin Analizi:

Pregnancies:
Ortalama: 3.8450520833333335
Medyan: 3.0
Standart Sapma: 3.3695780626988694
Minimum Değer: 0
Maksimum Değer: 17
25. Yüzdelik: 1.0
50. Yüzdelik (Medyan): 3.0
75. Yüzdelik: 6.0
Histogram:
```

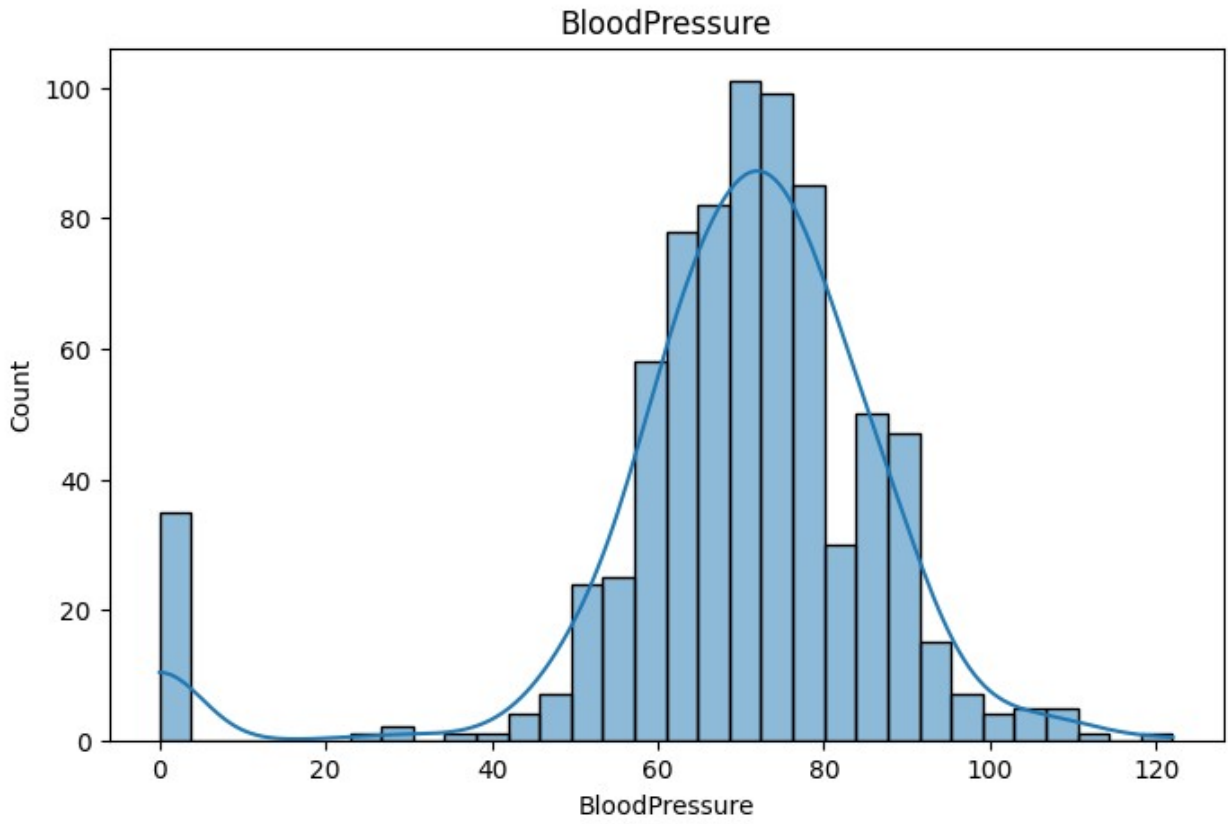



```
Glucose:
```

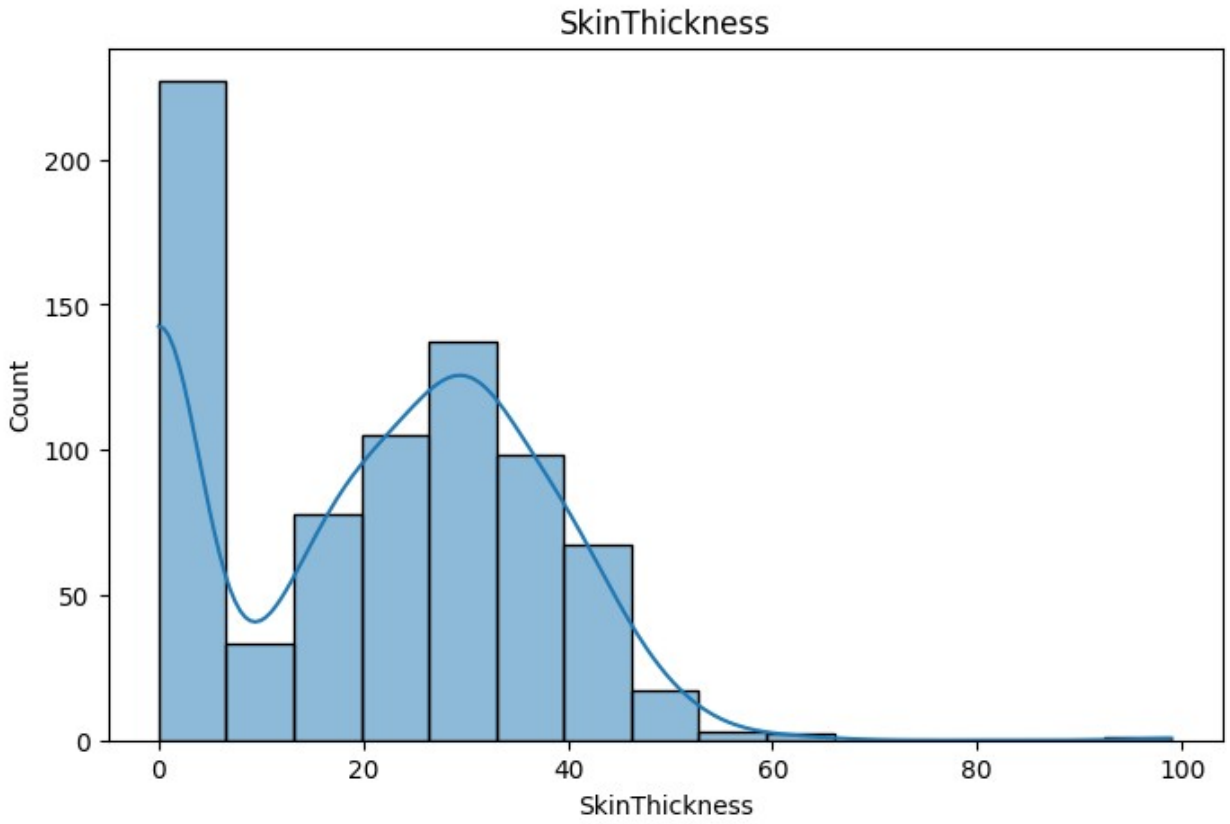
```
Ortalama: 120.89453125
Medyan: 117.0
Standart Sapma: 31.97261819513622
Minimum Değer: 0
Maksimum Değer: 199
25. Yüzdelik: 99.0
50. Yüzdelik (Medyan): 117.0
75. Yüzdelik: 140.25
Histogram:
```



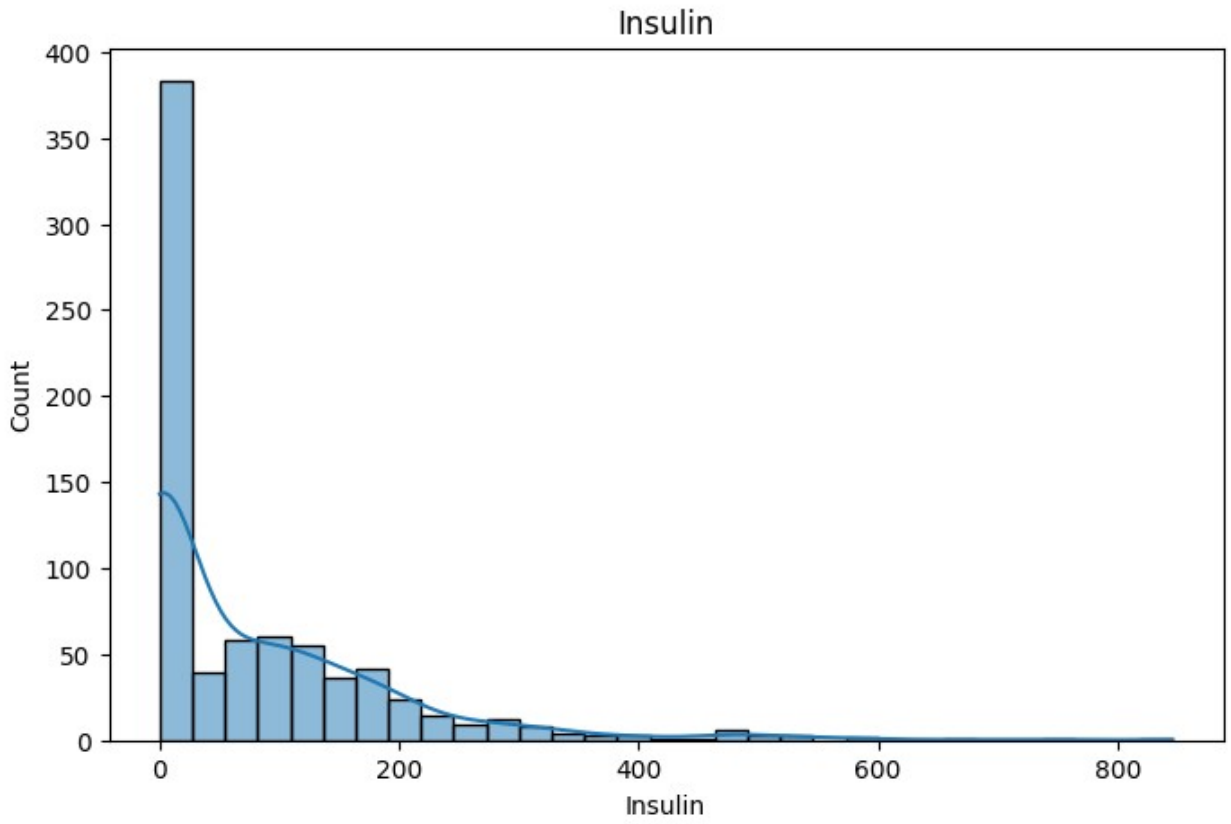

```
BloodPressure:
Ortalama: 69.10546875
Medyan: 72.0
Standart Sapma: 19.355807170644777
Minimum Değer: 0
Maksimum Değer: 122
25. Yüzdelik: 62.0
50. Yüzdelik (Medyan): 72.0
75. Yüzdelik: 80.0
Histogram:
```



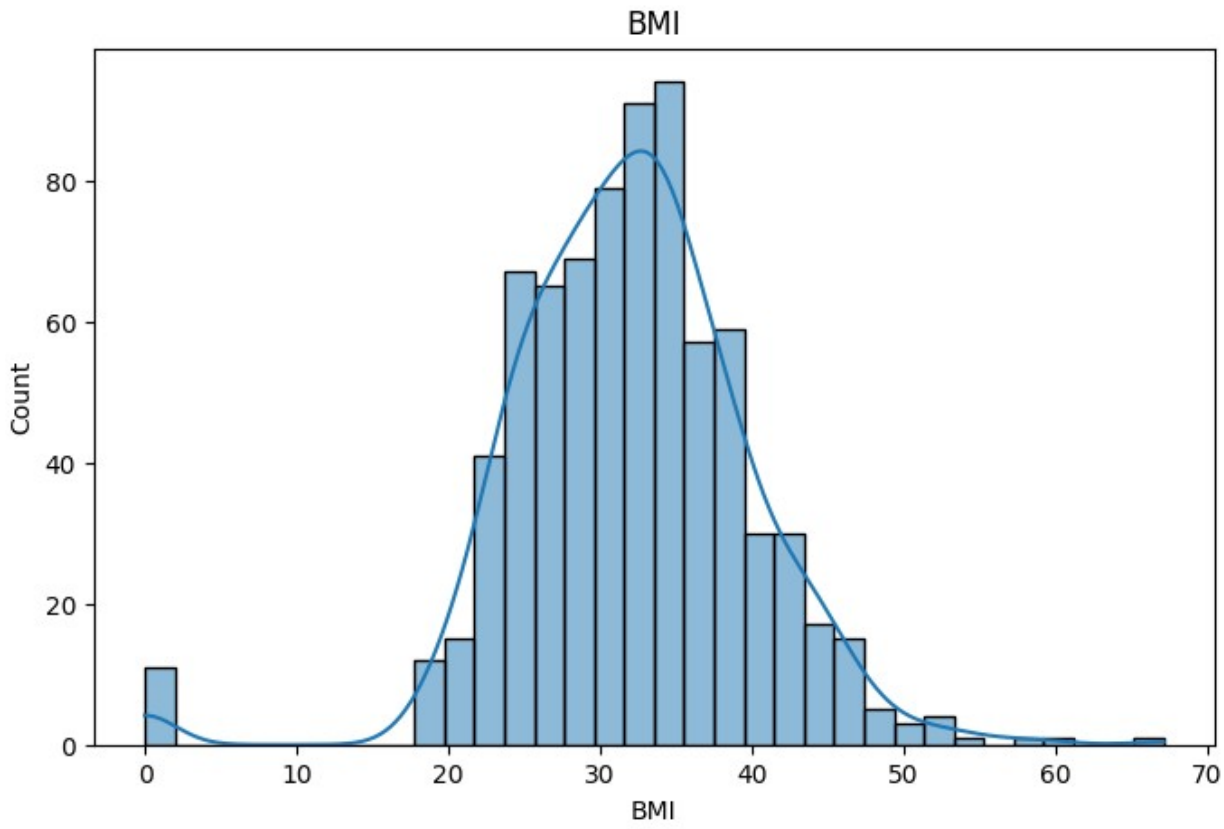

```
SkinThickness:
Ortalama: 20.536458333333332
Medyan: 23.0
Standart Sapma: 15.952217567727637
Minimum Değer: 0
Maksimum Değer: 99
25. Yüzdelik: 0.0
50. Yüzdelik (Medyan): 23.0
75. Yüzdelik: 32.0
Histogram:
```



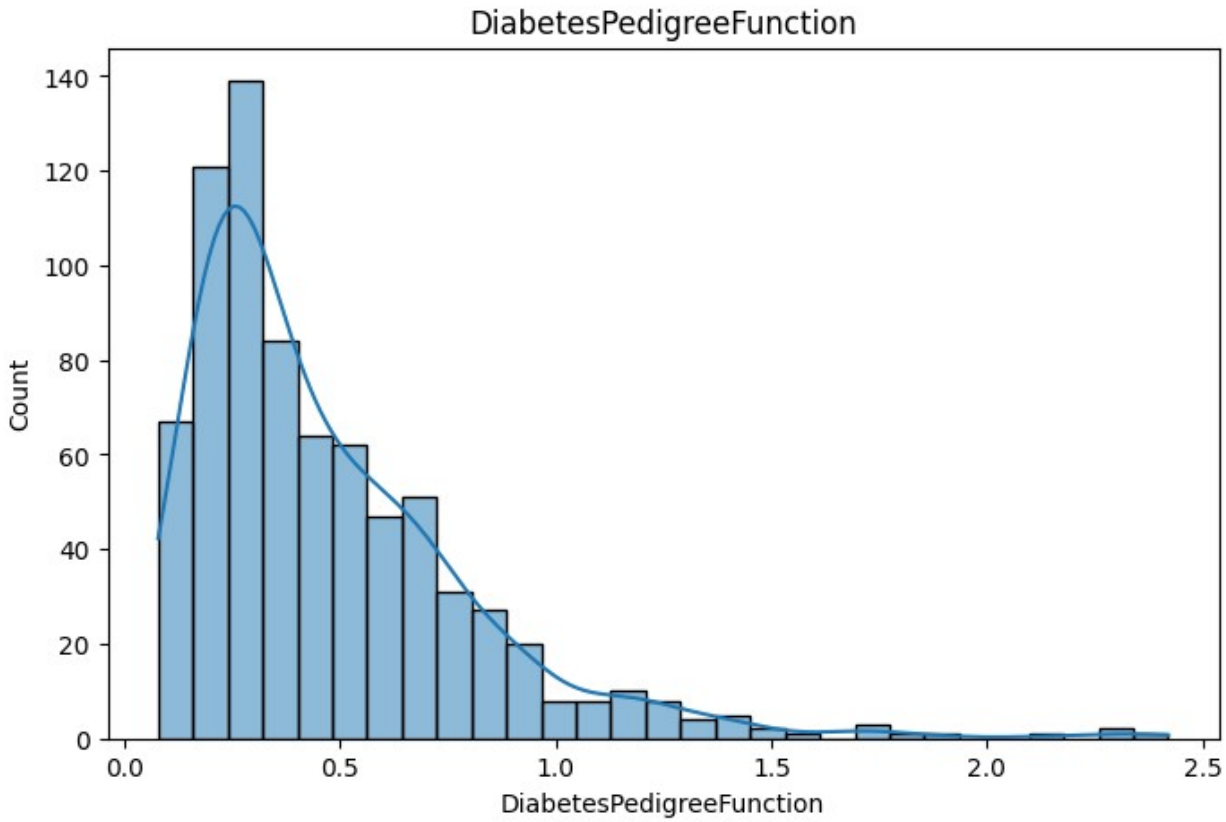

```
Insulin:
Ortalama: 79.79947916666667
Medyan: 30.5
Standart Sapma: 115.24400235133817
Minimum Değer: 0
Maksimum Değer: 846
25. Yüzdelik: 0.0
50. Yüzdelik (Medyan): 30.5
75. Yüzdelik: 127.25
Histogram:
```



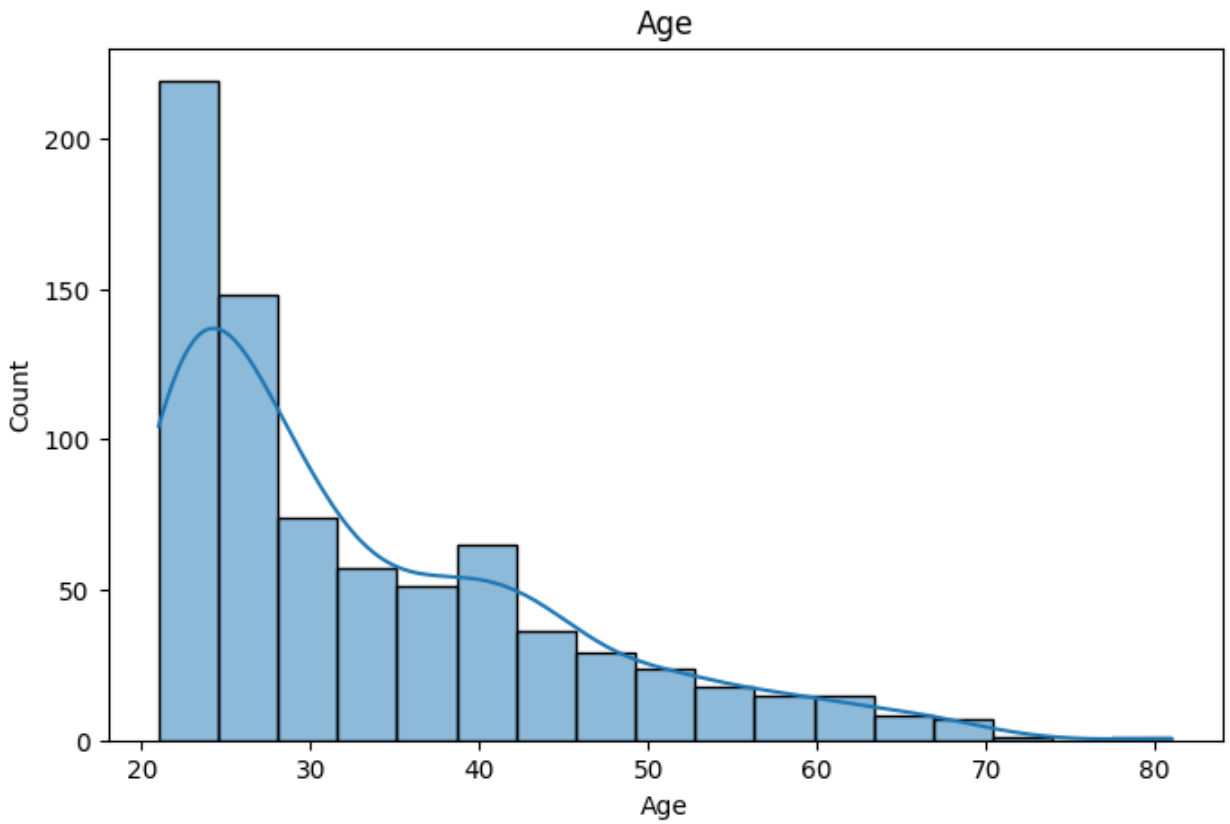

```
BMI:
Ortalama: 31.992578124999998
Medyan: 32.0
Standart Sapma: 7.884160320375446
Minimum Değer: 0.0
Maksimum Değer: 67.1
25. Yüzdelik: 27.3
50. Yüzdelik (Medyan): 32.0
75. Yüzdelik: 36.6
Histogram:
```



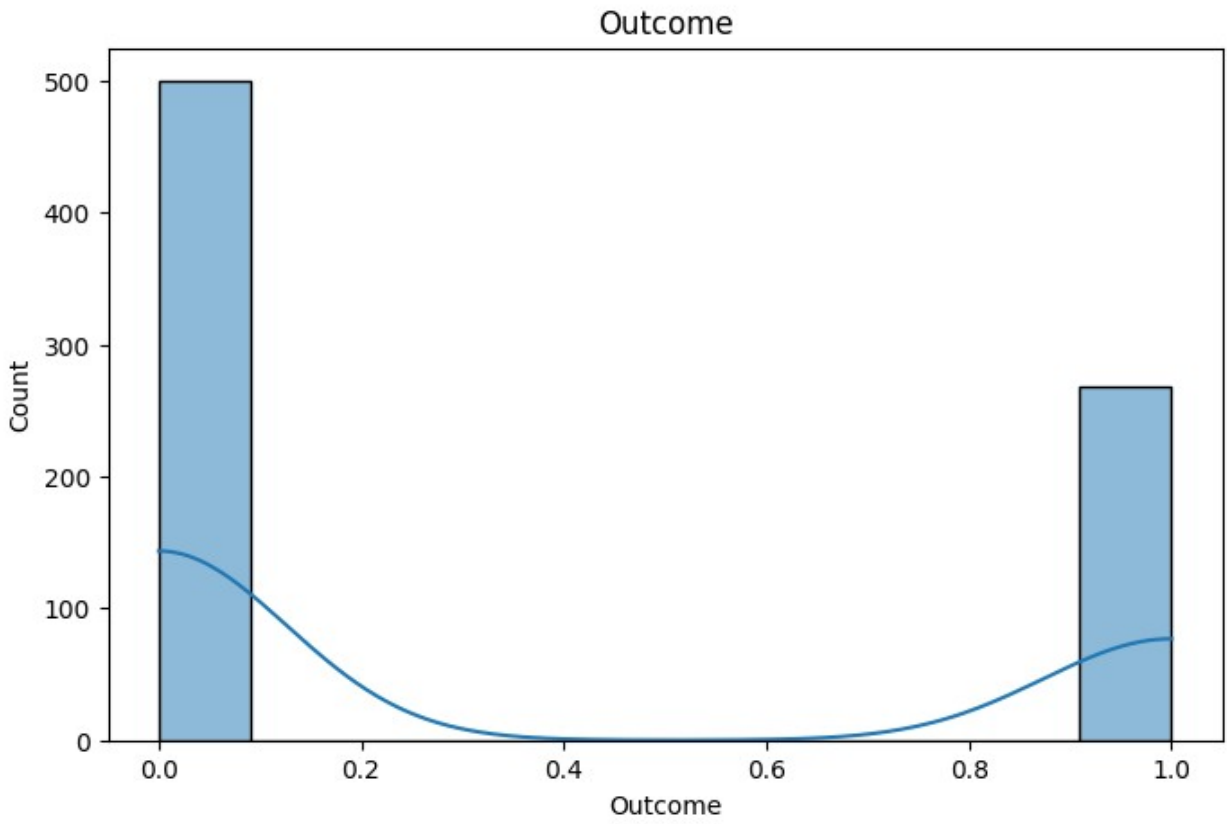

```
DiabetesPedigreeFunction:
Ortalama: 0.47187630208333325
Medyan: 0.3725
Standart Sapma: 0.3313285950127749
Minimum Değer: 0.078
Maksimum Değer: 2.42
25. Yüzdelik: 0.24375
50. Yüzdelik (Medyan): 0.3725
75. Yüzdelik: 0.62625
Histogram:
```




```
Age:
Ortalama: 33.240885416666664
Medyan: 29.0
Standart Sapma: 11.760231540678685
Minimum Değer: 21
Maksimum Değer: 81
25. Yüzdelik: 24.0
50. Yüzdelik (Medyan): 29.0
75. Yüzdelik: 41.0
Histogram:
```




```
Outcome:
Ortalama: 0.3489583333333333
Medyan: 0.0
Standart Sapma: 0.47695137724279896
Minimum Değer: 0
Maksimum Değer: 1
25. Yüzdelik: 0.0
50. Yüzdelik (Medyan): 0.0
75. Yüzdelik: 1.0
Histogram:
```




```
# Kategorik değişkenlerin analizi
print("\nKategorik Değişkenlerin Analizi:")
for col in categorical_cols:
    print("\n" + col + ":")
    print("Sınıf Sayısı:", df[col].nunique())
    print("Sınıflar ve Frekansları:")
    print(df[col].value_counts())
```

```
Kategorik Değişkenlerin Analizi:
```

Adım 4

```
for col in categorical_cols:
    print("\n" + col + " Kategorisine Göre Outcome Ortalaması:")
    print(df.groupby(col)['Outcome'].mean())

print("\nNumerik Değişkenlere Göre Outcome Ortalaması:")
print(df.groupby('Outcome')[numeric_cols].mean())
```

```
Numerik Değişkenlere Göre Outcome Ortalaması:
         Pregnancies     Glucose  BloodPressure  SkinThickness
```

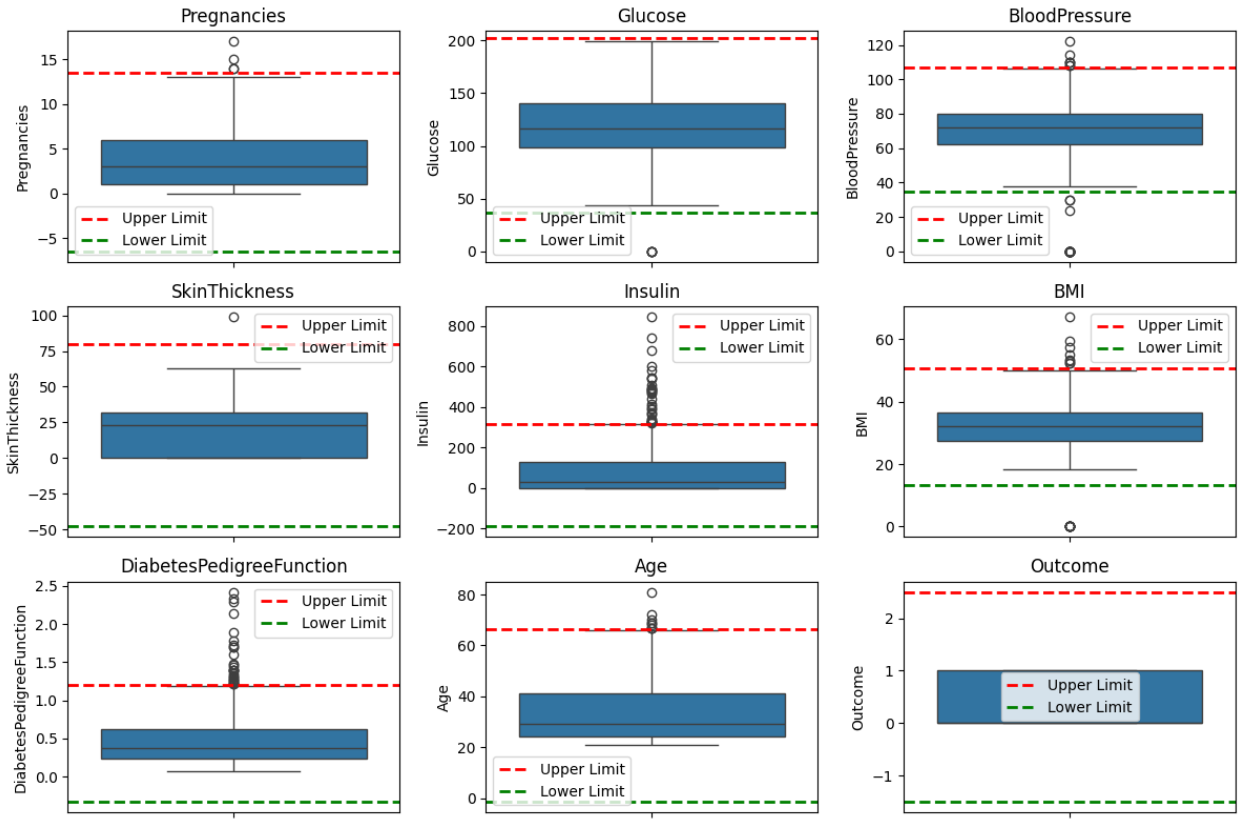
```
Insulin  \
Outcome

0           3.298000  109.980000      68.184000      19.664000
68.792000
1           4.865672  141.257463      70.824627      22.164179
100.335821

                BMI  DiabetesPedigreeFunction        Age  Outcome
Outcome
0         30.304200                  0.429734  31.190000      0.0
1         35.142537                  0.550500  37.067164      1.0
```

Adım 5

```
# Aykırı gözlemleri görselleştirme
plt.figure(figsize=(12, 8))
for i, col in enumerate(numeric_cols, 1):
    plt.subplot(3, 3, i)
    sns.boxplot(y=df[col])
    q1 = df[col].quantile(0.25)
    q3 = df[col].quantile(0.75)
    iqr = q3 - q1
    upper_limit = q3 + 1.5 * iqr
    lower_limit = q1 - 1.5 * iqr
    plt.axhline(y=upper_limit, color='r', linestyle='--', linewidth=2,
label='Upper Limit')
    plt.axhline(y=lower_limit, color='g', linestyle='--', linewidth=2,
label='Lower Limit')
    plt.title(col)
    plt.legend()
plt.tight_layout()
plt.show()
```



Adım 6

```
missing_values = df.isnull().sum()
print("Eksik Gözlem Analizi:")
print(missing_values)
```

```
Eksik Gözlem Analizi:
Pregnancies                 0
Glucose                     5
BloodPressure              35
SkinThickness             227
Insulin                   374
BMI                        11
DiabetesPedigreeFunction    0
Age                         0
Outcome                     0
dtype: int64
```

```
# Sıfır değerlerini NaN olarak atama
zero_cols = ['Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
'BMI']

for col in zero_cols:
    df[col].replace(0, np.nan, inplace=True)
```

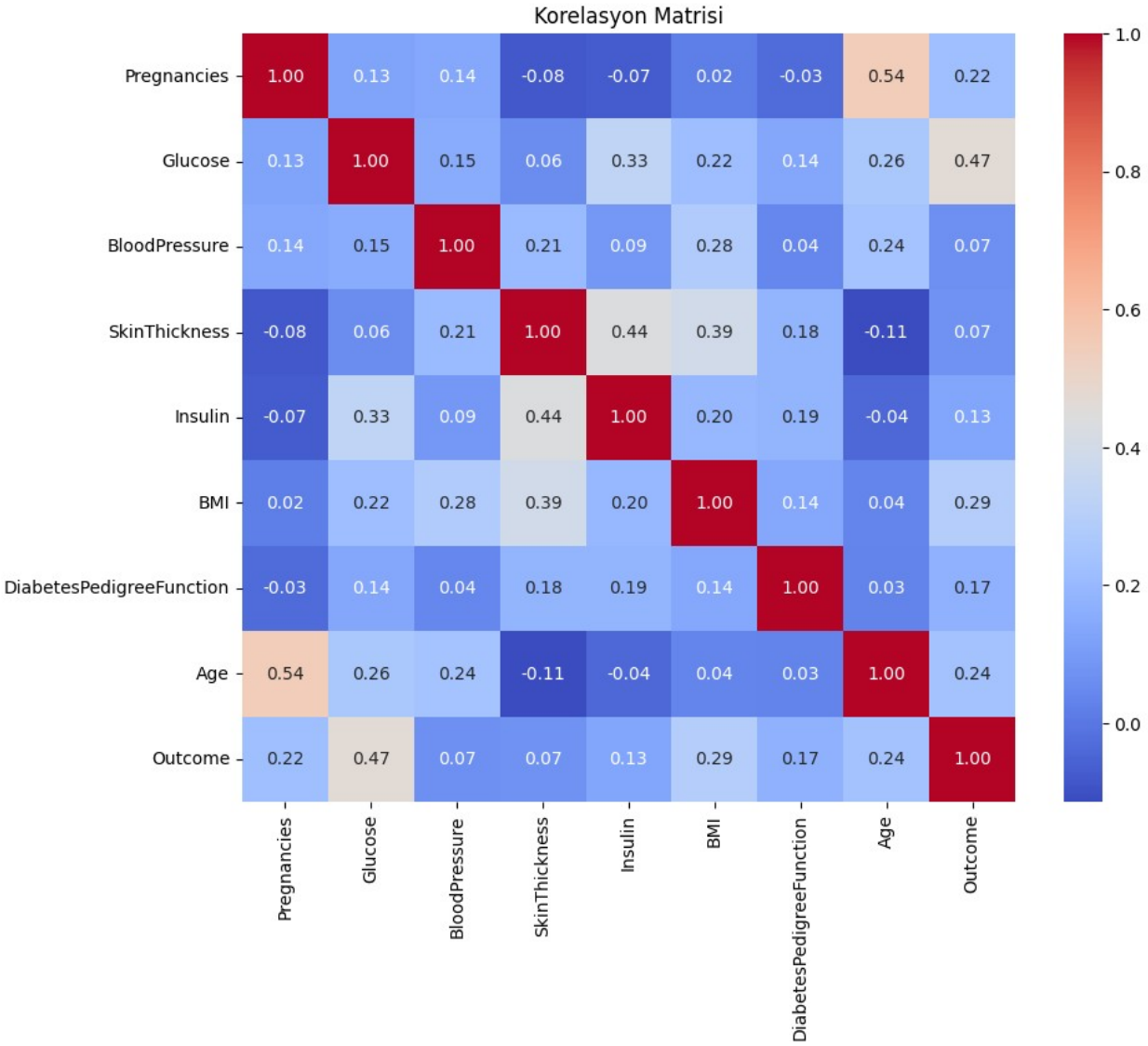
```
# Eksik değerlerin tekrar kontrol edilmesi
missing_values = df.isnull().sum()
print("Eksik Gözlem Analizi (Sıfır Değerleri NaN Olarak Atandıktan
Sonra):")
print(missing_values)
```

```
Eksik Gözlem Analizi (Sıfır Değerleri NaN Olarak Atandıktan Sonra):
Pregnancies                   0
Glucose                       5
BloodPressure                35
SkinThickness               227
Insulin                     374
BMI                          11
DiabetesPedigreeFunction      0
Age                           0
Outcome                       0
dtype: int64
```

Adım 7

```
# Korelasyon matrisini hesaplama
correlation_matrix = df.corr()

# Korelasyon matrisini görselleştirme
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm',
fmt=".2f")
plt.title("Korelasyon Matrisi")
plt.show()
```



Görev 2

Adım 1

```
# Sıfır değerlerini NaN olarak atama
zero_cols = ['Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
'BMI']

for col in zero_cols:
    df[col].replace(0, np.nan, inplace=True)

# Eksik değerlerin tekrar kontrol edilmesi
missing_values = df.isnull().sum()
```

```python
print("Eksik Gözlem Analizi (Sıfır Değerleri NaN Olarak Atandıktan Sonra):")
print(missing_values)
```

```
Eksik Gözlem Analizi (Sıfır Değerleri NaN Olarak Atandıktan Sonra):
Pregnancies                   0
Glucose                       5
BloodPressure                35
SkinThickness               227
Insulin                     374
BMI                          11
DiabetesPedigreeFunction      0
Age                           0
Outcome                       0
dtype: int64
```

```python
# Sayısal değişkenler için ortalama veya medyan ile doldurma
for col in zero_cols:
    df[col].fillna(df[col].median(), inplace=True)

df
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148.0	72.0	35.0	125.0	33.6	0.627	50	1
1	1	85.0	66.0	29.0	125.0	26.6	0.351	31	0
2	8	183.0	64.0	29.0	125.0	23.3	0.672	32	1
3	1	89.0	66.0	23.0	94.0	28.1			
4	0	137.0	40.0	35.0	168.0	43.1			
..	...	...	...	...	...	...			
763	10	101.0	76.0	48.0	180.0	32.9			
764	2	122.0	70.0	27.0	125.0	36.8			
765	5	121.0	72.0	23.0	112.0	26.2			
766	1	126.0	60.0	29.0	125.0	30.1			
767	1	93.0	70.0	31.0	125.0	30.4			

```
3                       0.167   21        0
4                       2.288   33        1
..                        ...  ...      ...
763                     0.171   63        0
764                     0.340   27        0
765                     0.245   30        0
766                     0.349   47        1
767                     0.315   23        0

[768 rows x 9 columns]
```

```python
def remove_outliers(df, column):
    q1 = df[column].quantile(0.25)
    q3 = df[column].quantile(0.75)
    iqr = q3 - q1
    upper_limit = q3 + 1.5 * iqr
    lower_limit = q1 - 1.5 * iqr
    df_cleaned = df[(df[column] >= lower_limit) & (df[column] <=
upper_limit)]
    return df_cleaned

# Aykırı gözlemleri temizleme
outlier_removed_df = df.copy()
for col in numeric_cols:
    outlier_removed_df = remove_outliers(outlier_removed_df, col)

outlier_removed_df
```

```
     Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin   BMI
\
0              6    148.0           72.0           35.0    125.0  33.6

1              1     85.0           66.0           29.0    125.0  26.6

2              8    183.0           64.0           29.0    125.0  23.3

5              5    116.0           74.0           29.0    125.0  25.6

7             10    115.0           72.0           29.0    125.0  35.3

..           ...      ...            ...            ...      ...   ...

761            9    170.0           74.0           31.0    125.0  44.0

762            9     89.0           62.0           29.0    125.0  22.5

764            2    122.0           70.0           27.0    125.0  36.8

766            1    126.0           60.0           29.0    125.0  30.1

767            1     93.0           70.0           31.0    125.0  30.4
```

```
     DiabetesPedigreeFunction  Age  Outcome      BMI_Group
0                       0.627   50        1          Obese
1                       0.351   31        0     Overweight
2                       0.672   32        1  Normal Weight
5                       0.201   30        0     Overweight
7                       0.134   29        0          Obese
..                        ...  ...      ...            ...
761                     0.403   43        1          Obese
762                     0.142   33        0  Normal Weight
764                     0.340   27        0          Obese
766                     0.349   47        1          Obese
767                     0.315   23        0          Obese

[332 rows x 10 columns]
```

Adım 2

```
# BMI Grupları
def bmi_group(bmi):
    if bmi < 18.5:
        return 'Underweight'
    elif 18.5 <= bmi < 25:
        return 'Normal Weight'
    elif 25 <= bmi < 30:
        return 'Overweight'
    else:
        return 'Obese'

outlier_removed_df['BMI_Group'] =
outlier_removed_df['BMI'].apply(bmi_group)

outlier_removed_df
```

```
     Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin   BMI
\
0              6    148.0           72.0           35.0    125.0  33.6

1              1     85.0           66.0           29.0    125.0  26.6

2              8    183.0           64.0           29.0    125.0  23.3

5              5    116.0           74.0           29.0    125.0  25.6

7             10    115.0           72.0           29.0    125.0  35.3
```

```
..           ...      ...            ...            ...      ...   ...

761            9    170.0           74.0           31.0    125.0  44.0

762            9     89.0           62.0           29.0    125.0  22.5

764            2    122.0           70.0           27.0    125.0  36.8

766            1    126.0           60.0           29.0    125.0  30.1

767            1     93.0           70.0           31.0    125.0  30.4


     DiabetesPedigreeFunction  Age  Outcome      BMI_Group
0                       0.627   50        1          Obese
1                       0.351   31        0     Overweight
2                       0.672   32        1  Normal Weight
5                       0.201   30        0     Overweight
7                       0.134   29        0          Obese
..                        ...  ...      ...            ...
761                     0.403   43        1          Obese
762                     0.142   33        0  Normal Weight
764                     0.340   27        0          Obese
766                     0.349   47        1          Obese
767                     0.315   23        0          Obese

[332 rows x 10 columns]
```

Adım 3

```
# One-Hot Encoding
df_encoded = pd.get_dummies(outlier_removed_df, columns=['BMI_Group'],
drop_first=True)

df_encoded
```

```
     Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin   BMI
\
0              6    148.0           72.0           35.0    125.0  33.6

1              1     85.0           66.0           29.0    125.0  26.6

2              8    183.0           64.0           29.0    125.0  23.3

5              5    116.0           74.0           29.0    125.0  25.6

7             10    115.0           72.0           29.0    125.0  35.3

..           ...      ...            ...            ...      ...   ...
```

```
761            9    170.0           74.0           31.0    125.0  44.0

762            9     89.0           62.0           29.0    125.0  22.5

764            2    122.0           70.0           27.0    125.0  36.8

766            1    126.0           60.0           29.0    125.0  30.1

767            1     93.0           70.0           31.0    125.0  30.4


     DiabetesPedigreeFunction  Age  Outcome  BMI_Group_Obese  \
0                       0.627   50        1             True
1                       0.351   31        0            False
2                       0.672   32        1            False
5                       0.201   30        0            False
7                       0.134   29        0             True
..                        ...  ...      ...              ...
761                     0.403   43        1             True
762                     0.142   33        0            False
764                     0.340   27        0             True
766                     0.349   47        1             True
767                     0.315   23        0             True

     BMI_Group_Overweight  BMI_Group_Underweight
0                   False                  False
1                    True                  False
2                   False                  False
5                    True                  False
7                   False                  False
..                    ...                    ...
761                 False                  False
762                 False                  False
764                 False                  False
766                 False                  False
767                 False                  False

[332 rows x 12 columns]
```

Adım 4

```
# Standart Ölçeklendirme
scaler = StandardScaler()
numeric_cols = ['Pregnancies', 'Glucose', 'BloodPressure',
'SkinThickness', 'Insulin', 'BMI', 'DiabetesPedigreeFunction', 'Age']


# Standart ölçeklendirme işlemi
```

```
df_encoded[numeric_cols] =
scaler.fit_transform(df_encoded[numeric_cols])

df_encoded
```

```
     Pregnancies   Glucose  BloodPressure  SkinThickness  Insulin
BMI  \
0       0.490124  1.003027      -0.153355       1.439678      0.0
0.366217
1      -1.005476 -1.158229      -0.740823       0.015726      0.0 -
0.790311
2       1.088364  2.203725      -0.936646       0.015726      0.0 -
1.335532
5       0.191004 -0.094754       0.042468       0.015726      0.0 -
0.955530
7       1.686604 -0.129060      -0.153355       0.015726      0.0
0.647089
..           ...       ...            ...            ...      ...
...
761     1.387484  1.757752       0.042468       0.490377      0.0
2.084489
762     1.387484 -1.021007      -1.132469       0.015726      0.0 -
1.467706
764    -0.706356  0.111080      -0.349178      -0.458924      0.0
0.894916
766    -1.005476  0.248303      -1.328292       0.015726      0.0 -
0.212047
767    -1.005476 -0.883784      -0.349178       0.490377      0.0 -
0.162481

     DiabetesPedigreeFunction       Age  Outcome  BMI_Group_Obese  \
0                    1.204203  1.186805        1             True
1                   -0.077565 -0.355186        0            False
2                    1.413186 -0.274028        1            False
5                   -0.774177 -0.436343        0            False
7                   -1.085331 -0.517501        0             True
..                        ...       ...      ...              ...
761                  0.163928  0.618703        1             True
762                 -1.048178 -0.192871        0            False
764                 -0.128650 -0.679816        0             True
766                 -0.086853  0.943333        1             True
767                 -0.244752 -1.004445        0             True

     BMI_Group_Overweight  BMI_Group_Underweight
0                   False                  False
1                    True                  False
2                   False                  False
5                    True                  False
7                   False                  False
..                    ...                    ...
```

```
761                  False                   False
762                  False                   False
764                  False                   False
766                  False                   False
767                  False                   False

[332 rows x 12 columns]
```

Adım 5

```
# Bağımsız değişkenlerin ve bağımlı değişkenin belirlenmesi
X = df_encoded.drop('Outcome', axis=1)
y = df_encoded['Outcome']

X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.3, random_state=42)

model = DecisionTreeClassifier()

model.fit(X_train, y_train)

DecisionTreeClassifier()

y_pred = model.predict(X_test)

accuracy = accuracy_score(y_test, y_pred)
print("Model Performansı:")
print(f"Accuracy: {accuracy}")

Model Performansı:
Accuracy: 0.66

print("Classification Report:")
print(classification_report(y_test, y_pred))

Classification Report:
              precision    recall  f1-score   support

           0       0.76      0.74      0.75        69
           1       0.45      0.48      0.47        31

    accuracy                           0.66       100
   macro avg       0.61      0.61      0.61       100
weighted avg       0.67      0.66      0.66       100
```