In [80]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, r2_score
from sklearn.metrics import mean_squared_error

# Seed fonksiyonu rastgele veri oluşturmak için kullanıldı.
np.random.seed(42)

# 100 satırlık vei oluşturmak istenildi.
n_samples = 100

# 20-60 yaş aralığı
ages = np.random.randint(20, 60, n_samples)

# 0: Güneşli, 1: Yağmurlu, 2: Karlı
weather_conditions = np.random.choice([0, 1, 2], n_samples)

# 0: Hafta İçi, 1: Hafta Sonu
day_types = np.random.choice([0, 1], n_samples)

# 2000-15000 arasında önceki gün adım sayısı
previous_steps = np.random.randint(2000, 15000, n_samples)

# Adım sayısı için formül oluşturuldu.

steps = previous_steps * (1 + np.random.normal(0, 0.1, n_samples))  # Gürül
tü ekleyelim
steps -= (ages - 30) * 10  # Yaş arttıkça biraz azalsın
steps *= (1 - weather_conditions * 0.1)  # Hava kötüleştikçe adım sayısı az
alsın
steps = np.maximum(steps, 1000).astype(int)  # Minimum 1000 adım olsun

# DataFrame oluşturuldu.
step_data = pd.DataFrame({
    'Age': ages,
    'Weather': weather_conditions,
    'Day Type': day_types,
    'Previous Steps': previous_steps,
    'Steps': steps
})

print(step_data.head())

# CSV olarak kaydedelim ki model eğitiminde kullanabilelim
step_data.to_csv("step_data.csv", index=False)
```

```
   Age  Day Type  Previous Steps  Steps  Weather
0   58         0            8197   5754        2
1   48         0           13383  10726        2
2   34         1           11888  10725        1
3   27         0           14183  12591        0
4   40         0           14915  12278        2
```

In [81]:
```
df = pd.read_csv("step_data.csv")
print(df.head())
```

```
   Age  Day Type  Previous Steps  Steps  Weather
0   58         0            8197   5754        2
1   48         0           13383  10726        2
2   34         1           11888  10725        1
3   27         0           14183  12591        0
4   40         0           14915  12278        2
```

In [82]:
```
# -------------------------
# LINEAR REGRESSION MODELİ
# -------------------------

# Veriyi tekrar yükleyelim
df = pd.read_csv("step_data.csv")
```

In [83]:
```
# Bağımsız değişkenler (X) ve hedef değişken (y)
X = df[['Age', 'Weather', 'Day Type', 'Previous Steps']]
y = df['Steps']
```

In [84]:
```
# Eğitim ve test verisi olarak ayıralım
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, ra
ndom_state=42)
```

In [85]:
```
# Modeli oluştur ve eğit
model = LinearRegression()
model.fit(X_train, y_train)
```

Out[85]:
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None,
         normalize=False)
```

In [86]:
```
# Tahmin yap
y_train_pred = model.predict(X_train)
y_test_pred = model.predict(X_test)
```

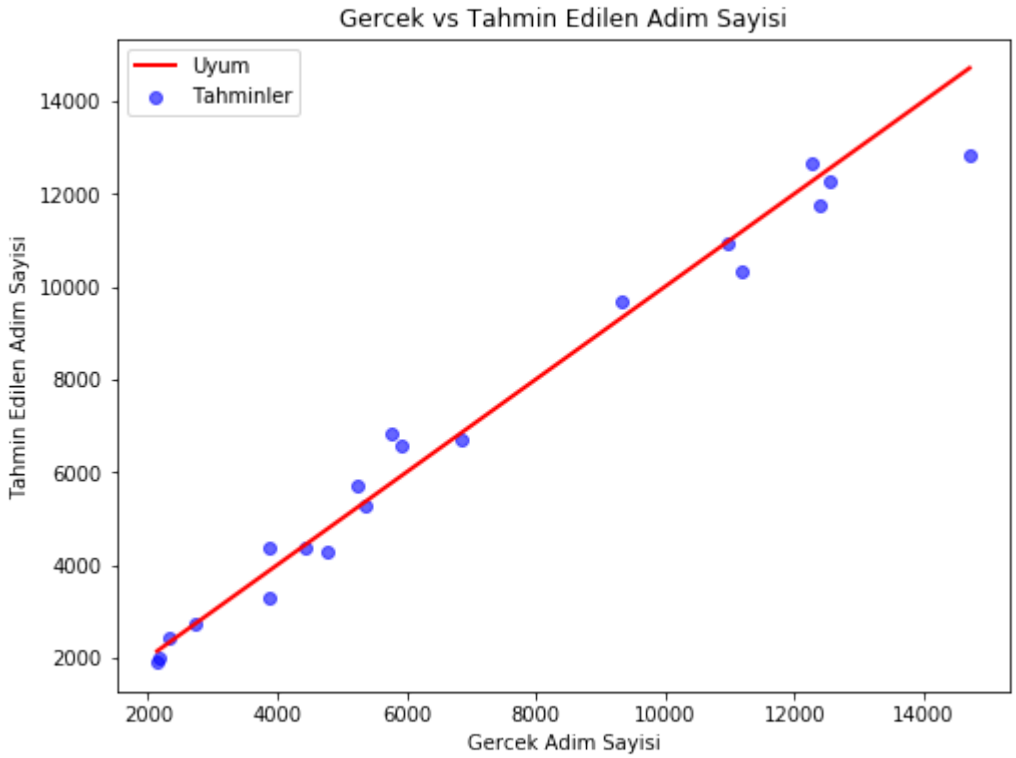
In [90]:
```
# Model başarısını ölç
train_mae = mean_absolute_error(y_train, y_train_pred)
test_mae = mean_absolute_error(y_test, y_test_pred)
train_r2 = r2_score(y_train, y_train_pred)
test_r2 = r2_score(y_test, y_test_pred)

print("Eğitim Seti Mean Absolute Error: {}".format(train_mae))
print("Test Seti Mean Absolute Error: {}".format(test_mae))
print("Eğitim Seti R² Score: {}".format(train_r2))
print("Test Seti R² Score: {}".format(test_r2))
```

```
Eğitim Seti Mean Absolute Error: 649.894611812
Test Seti Mean Absolute Error: 448.715568277
Eğitim Seti R² Score: 0.935216906486
Test Seti R² Score: 0.975416052117
```

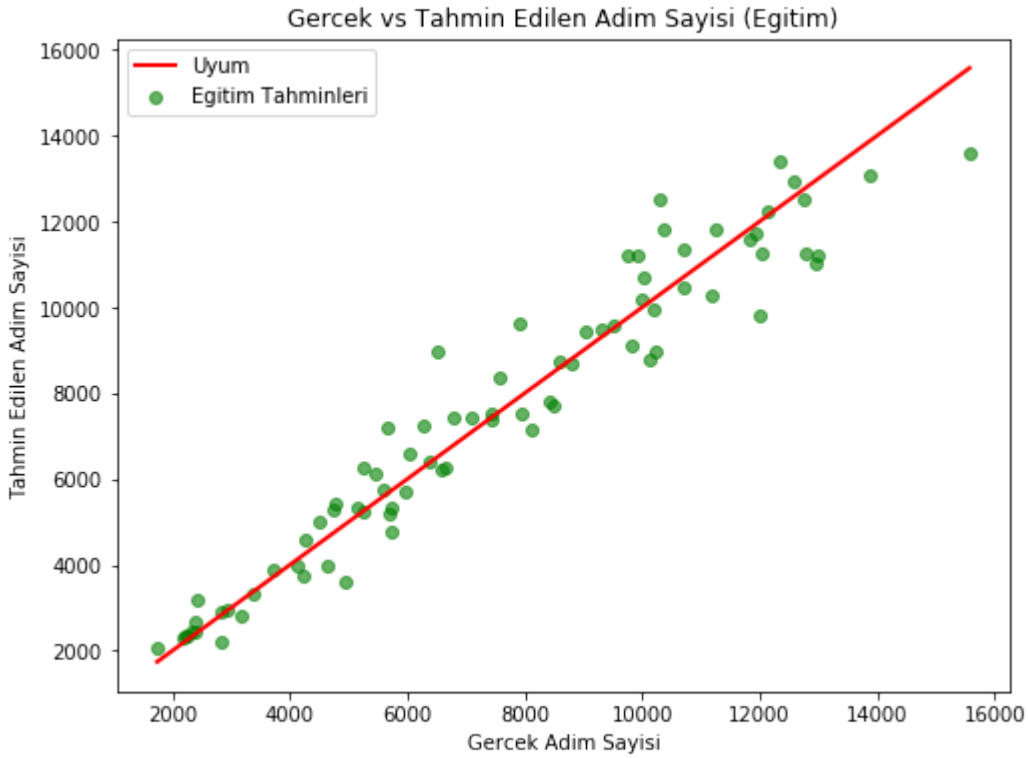
In [91]:
```
# Modelimiz ortalama olarak gerçek adım sayısından yaklaşık 449 adım sapma yapıyor.
# Öğrenme seti için baktığımızda bu sonuç 650 adımdır. Bu durumda kesinlikle overfitting yoktur.
# Bu, günlük adım sayısına kıyasla küçük bir hata sayılabilir.
# Test setindeki 0.9754 değeri, modelin değişkenler arasındaki ilişkileri çok iyi öğrendiğini ve
# yüksek doğrulukla tahmin yaptığını gösteriyor. Eğitim seti de 0.9352 olduğundann underfitting de yoktur.
#Gerçek ve tahmin edilen adım sayılarını karşılaştıralım
results = pd.DataFrame({'Gerçek Adım Sayısı': y_test.values, 'Tahmin Edilen Adım Sayısı': y_pred})
print(results)
```

	Gerçek Adım Sayısı	Tahmin Edilen Adım Sayısı
0	6844	6701.844612
1	2164	1999.532659
2	3873	3287.873278
3	2745	2742.978455
4	3875	4378.604001
5	10973	10933.585595
6	12554	12253.729699
7	2142	1911.088056
8	14711	12851.244563
9	5754	6812.712388
10	5250	5714.137763
11	9316	9673.825037
12	5365	5274.347074
13	11173	10330.727443
14	2332	2428.569397
15	12278	12674.577072
16	4790	4292.499159
17	4420	4366.237927
18	12396	11763.499458
19	5906	6562.073685

In [92]:
```
plt.figure(figsize=(8,6))
plt.scatter(y_test, y_pred, color='blue', alpha=0.6, label="Test Tahminler
i")
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'r', l
w=2, label="Uyum")
plt.xlabel("Gercek Adim Sayisi")
plt.ylabel("Tahmin Edilen Adim Sayisi")
plt.title("Gercek vs Tahmin Edilen Adim Sayisi")
plt.legend()
plt.show()
```



In [95]:
```
plt.figure(figsize=(8,6))
plt.scatter(y_train, y_train_pred, color='green', alpha=0.6, label="Egitim
Tahminleri")
plt.plot([y_train.min(), y_train.max()], [y_train.min(), y_train.max()],
'r', lw=2, label="Uyum")
plt.xlabel("Gercek Adim Sayisi")
plt.ylabel("Tahmin Edilen Adim Sayisi")
plt.title("Gercek vs Tahmin Edilen Adim Sayisi (Egitim)")
plt.legend()
plt.show()
```



In [ ]: