

# Tek Bir Bireyin Günlük Adım Sayısı Verileri Üzerinden Stokastik Bir Süreç Analizi

## İçindekiler

- Önsöz
- Giriş
- Veri Kümesi
- Durum Uzayı ( $S$ )
- Durumlar
- Zaman Parametresi ( $T$ )
- Süreç Açıklaması
- İlk 30 Gün İçin Günlere Göre Durum Grafiği
  - Grafik Yorumu
  - Sonuç
- Bir Adım Geçiş Matrisi ( $P$ )
  - En Yüksek Olasılık Değerleri ve Yorumu
  - En Düşük Olasılık Değerleri ve Yorumu
  - Genel Yorum
- Üç Adım Geçiş Matrisi ( $P^3$ )
- On Adım Geçiş Matrisi ( $P^{10}$ )
- Yüz Adım Geçiş Matrisi ( $P^{100}$ )
- Kaynakça

# Önsöz

Bu çalışma, **Olasılıksal Süreçler** dersi kapsamında verilen **ara sınav ödevi** için hazırlanmıştır. Projede, **Markov zincirleri** kullanılarak bir bireyin günlük fiziksel aktivite seviyeleri (düşük, orta, yüksek) arasındaki geçişleri modelleyen bir durum analizi gerçekleştirilmiştir.

Bu amaçla iki ayrı Python kod dosyası oluşturulmuştur:

- **veri\_duzenleyici.py** dosyası, ham veriyi ödev gerekliliklerine uygun şekilde düzenlemek için kullanılmıştır.
- **markov\_sureci.py** dosyası ise, geçiş olasılıklarını hesaplamak, denge dağılımını bulmak ve grafiksel analizleri gerçekleştirmek amacıyla yazılmıştır.

Kodların yazımında ders uygulamaları kaynak olarak edinilmiş, geri kalan tüm analiz ve geliştirme süreçleri tarafimdan gerçekleştirilmiştir.

Raporun devamında analiz sonuçları grafiklerle sunulmuş, yorumları ile birlikte açıklanmıştır.

## Giriş:

Bu çalışmada, tek bir bireye ait mobil bir uygulama aracılığıyla toplanan günlük adım sayısı verileri temel alınarak, bu bireyin aktivite seviyesindeki değişimleri modelleyen bir stokastik süreç tanımlanmıştır. Belirli eşik değerler aracılığıyla tanımlanan farklı aktivite durumları arasındaki geçişler incelenerek, bireyin aktivite alışkanlıklarını ve olası değişim örüntülerini hakkında olasılıksal çıkarımlar yapılması amaçlanmaktadır. Bu bağlamda, Mart 2019 ile Mayıs 2022 arasındaki günlük adım sayıları analiz edilerek, düşük, orta ve yüksek aktivite durumları arasındaki geçişler stokastik bir çerçevede ele alınacaktır.

### 1. Veri Kümesi:

- **Kaynak**  | Step Count From Phone App
- **Zaman Aralığı:** 2019-05-11 ile 2022-03-14 arası
- **Gözlem Sayısı:** Günlük bazda 1011 tane gözlem bulunmaktadır (tek bir bireye ait).

### 2. Durum Uzayı (S):

Durum uzayı, tanımlanan stokastik sürecin alabileceği tüm olası durumların kümesidir:

$$S = \{\text{Düşük Aktivite, Orta Aktivite, Yüksek Aktivite}\}$$

Etiketlenmiş durum uzayı (Düşük Aktivite=0, Orta Aktivite=1, Yüksek Aktivite =2)

$$S = \{0,1,2\}$$

### 3. Durumlar:

Günlük toplam adım sayısına göre üç farklı aktivite durumu tanımlanmıştır:

- **Düşük Aktivite:** Günlük adım sayısı  $\leq 4000$
- **Orta Aktivite:** Günlük adım sayısı  $4001 - 9000$
- **Yüksek Aktivite:** Günlük adım sayısı  $> 9000$

### 4. Zaman Parametresi (T):

Zaman parametresi, stokastik sürecin gözlemlendiği ayrık zaman noktalarını temsil eder:

$$T = \{1, 2, 3, \dots, n\}$$

Burada her bir ölçüm bir zaman adımını ifade etmektedir. Örneğin:

- 1. ölçüm: 2019-05-11
- ...
- 10. ölçüm: 2019-05-19
- ...
- 1011. ölçüm: 2022-03-14

### 5. Süreç Açıklaması:

Tanımlanan bu stokastik süreç, **tek bir bireyin** günlük fiziksel aktivite seviyesinin zaman içindeki değişimini modellemektedir. Belirlenen durumlar (Düşük, Orta, Yüksek Aktivite), bu bireyin bir gün içerisindeki toplam adım sayısına dayanmaktadır. Bu süreç, ayrık zamanlı bir Markov süreci olarak ele alınır. Bu çerçevede, farklı aktivite durumları arasındaki geçiş olasılıkları analiz edilerek, bu bireyin aktivite örüntüleri ve bir durumdan diğerine geçiş olasılıkları incelenebilir. Örneğin, düşük aktiviteli bir günün ardından orta veya yüksek aktiviteli bir gün geçirme olasılığı hesaplanabilir. Bu analiz, bireyin aktivite alışkanlıklarını anlamak ve olası gelecekteki aktivite seviyelerini tahmin etmek için değerli bilgiler sunabilir.

Bu işlemleri yapmamı sağlayan **veri\_duzenleyici.py** :

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import os

# Veri kaynağı: https://www.kaggle.com/datasets/l3llff/step-count-from-phone-app
# Açıklama: Telefon uygulamasından alınan günlük adım sayısı verileri.

def load_data(user_id=4):
    """CSV dosyasını oku ve kullanıcı verilerini filtrele."""
    current_directory = os.getcwd()
```

```
file_path = f'{current_directory}/step-count-from-phone-app.csv'
if not os.path.exists(file_path):
    raise FileNotFoundError("Veri dosyası bulunamadı!")
df = pd.read_csv(file_path)
df = df[df["user_id"] == user_id]
return df

def durum_belirle(value):
    """Adım sayısına göre aktivite durumunu belirle."""
    if value <= 4000:
        return "Düşük Aktivite"
    elif value <= 9000:
        return "Orta Aktivite"
    else:
        return "Yüksek Aktivite"

def turn_to_datetime(df):
    """Tarih kolonlarını datetime formatına çevir."""
    df['start'] = pd.to_datetime(df['start'])
    df['end'] = pd.to_datetime(df['end'])
    return df

def gunluk_veriler(df):
    """Günlük bazda verileri gruplar ve durumları belirler."""
    df['gun'] = df['start'].dt.date
    gunluk_df = df.groupby('gun').agg({
        'value': 'sum',
        'start': 'first',
        'end': 'last'
    }).reset_index()
    gunluk_df["durum"] = gunluk_df["value"].apply(durum_belirle)
    gunluk_df = gunluk_df[['gun', 'value', 'durum']]
    return gunluk_df

def filtre_ardisik_gunler(df):
    """Ardışık olmayan günleri veriden çıkar."""
    df = df.sort_values(by='gun')
    df['gun'] = pd.to_datetime(df['gun'])
    df['gun_farki'] = (df['gun'] - df['gun'].shift()).dt.days
    df = df[df['gun_farki'] == 1].drop(columns=['gun_farki'])
    return df

def main():
    """Ana fonksiyon."""
    df = load_data()
    df = turn_to_datetime(df)
    gunluk_df = gunluk_veriler(df)
    gunluk_df = filtre_ardisik_gunler(gunluk_df)
    gunluk_df.to_csv('gunluk_veriler.csv', index=False, encoding='utf-8')
```

```

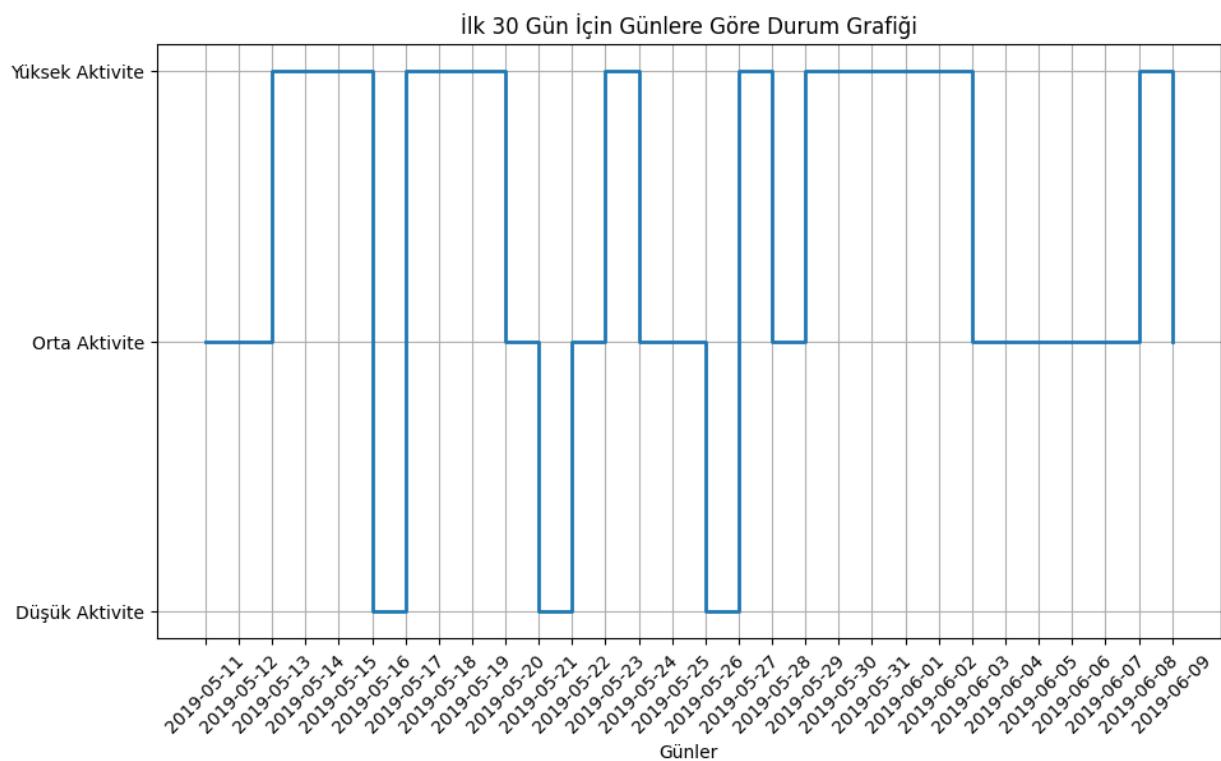
        print("Günlük veriler 'gunluk_veriler.csv' dosyasına kaydedildi.")
        print(f"Bir günde atılan maksimum adım sayısı: {gunluk_df['value'].max()},",
Bir günde atılan minimum adım sayısı: {gunluk_df['value'].min()}")
        print(gunluk_df["durum"].value_counts())

if __name__ == "__main__":
    main()

```

Bundan sonraki işlemler **markov\_sureci.py** adlı Python kodunda gerçekleşmiştir.

### **İlk 30 Gün İçin Günlere Göre Durum Grafiği :**



### **Grafik Yorumu:**

Bu grafik, **tek bir bireyin** yaklaşık olarak 30 günlük (11 Mayıs 2019'dan 9 Haziran 2019'a kadar) aktivite seviyesinin zaman içindeki değişimini görselleştirmektedir.

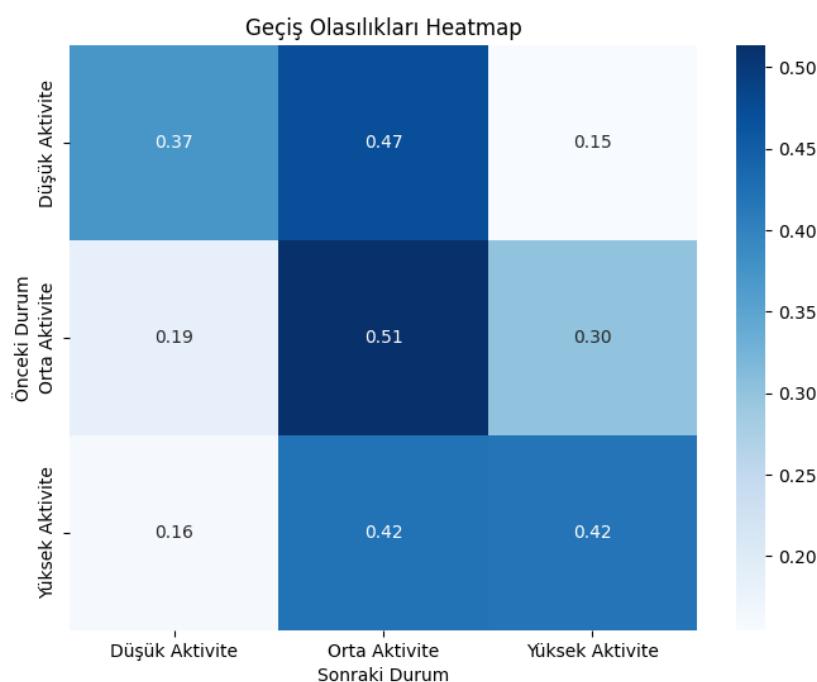
- **Y Ekseni (Durum Uzayı):** Y ekseni, tanımladığımız üç farklı aktivite durumunu temsil etmektedir:
  - **Düşük Aktivite:** Günlük adım sayısı  $\leq 4000$
  - **Orta Aktivite:** Günlük adım sayısı  $4001 - 9000$
  - **Yüksek Aktivite:** Günlük adım sayısı  $> 9000$
- **X Ekseni (Zaman Parametresi):** X ekseni, gözlemlenen günleri göstermektedir. Her bir dikey çizgi bir günü temsil eder ve tarihler alt kısmında belirtilmiştir. Gözlemler arasındaki zaman aralığı görüldüğü gibi **sabittir (günlük)**.

### **Sonuç:**

Bu grafik, bireyin günlük aktivite seviyesinin dinamik bir süreç olduğunu görsel olarak teyit etmektedir. Farklı aktivite durumları arasında sıkça geçişler yaşanmaktadır ve belirli örüntüler gözlemlenmektedir. Ancak, bu kısa zaman aralığı, uzun vadeli davranış veya denge dağılımı hakkında kesin sonuçlar çıkarmak için yeterli değildir. Daha uzun süreli verilerle yapılacak analizler, bireyin aktivite alışkanlıklarını ve tanımladığımız Markov zincirinin özelliklerini hakkında daha kapsamlı bir anlayış sağlayacaktır.

### **Bir-adım geçiş matrisi P :**

Geçiş Olasılıkları Matrisi:			
	Düşük Aktivite	Orta Aktivite	Yüksek Aktivite
Düşük Aktivite	0.372727	0.472727	0.154545
Orta Aktivite	0.185031	0.513514	0.301455
Yüksek Aktivite	0.159091	0.422078	0.418831



### **En Yüksek Olasılık Değerleri ve Yorumu:**

- 0.51:** Bu değer, önceki gün aktivite seviyesi **Orta Aktivite** olan bireyin, sonraki gün de **Orta Aktivite** seviyesinde kalma olasılığını göstermektedir. Bu, orta aktivite durumunun diğer durumlara göre daha kararlı veya "yapışkan" bir özellik taşıdığını düşündürmektedir. Çünkü düşükte kalma olasılığı(0.37) ve yüksekte kalma olasılığı(0.42) den büyüktür. Birey orta seviyede bir aktivite ritmine girdiğinde, bu ritmi sürdürme olasılığı oldukça yüksektir.

- **0.47:** Bu değer, önceki gün aktivite seviyesi **Düşük Aktivite** olan bireyin, sonraki gün aktivite seviyesini **Orta Aktivite**ye çıkışma olasılığını göstermektedir. Bu, düşük aktiviteden orta aktiviteye doğru belirgin bir geçiş eğilimi olduğunu işaret etmektedir. Düşük aktiviteli bir günün ardından bireyin aktivitesini artırma olasılığı oldukça yüksektir.

#### **En Düşük Olasılık Değerleri ve Yorumu:**

- **0.15:** Bu değer, önceki gün aktivite seviyesi **Düşük Aktivite** olan bireyin, sonraki gün doğrudan **Yüksek Aktivite** seviyesine geçme olasılığını göstermektedir. Düşük bir olasılık olması, bireyin aktivite seviyesinde ani ve büyük bir artışın nispeten nadir gerçekleştiğini düşündürmektedir. Aktivite seviyesindeki yükselişin genellikle daha kademeli olduğu söylenebilir.
- **0.16:** Bu değer, önceki gün aktivite seviyesi **Yüksek Aktivite** olan bireyin, sonraki gün doğrudan **Düşük Aktivite** seviyesine düşme olasılığını göstermektedir. Yüksek aktiviteden doğrudan düşük aktiviteye geçişin düşük bir olasılıkta olması, aktivite seviyesindeki ani ve büyük düşüşlerin de seyrek olduğunu işaret etmektedir.

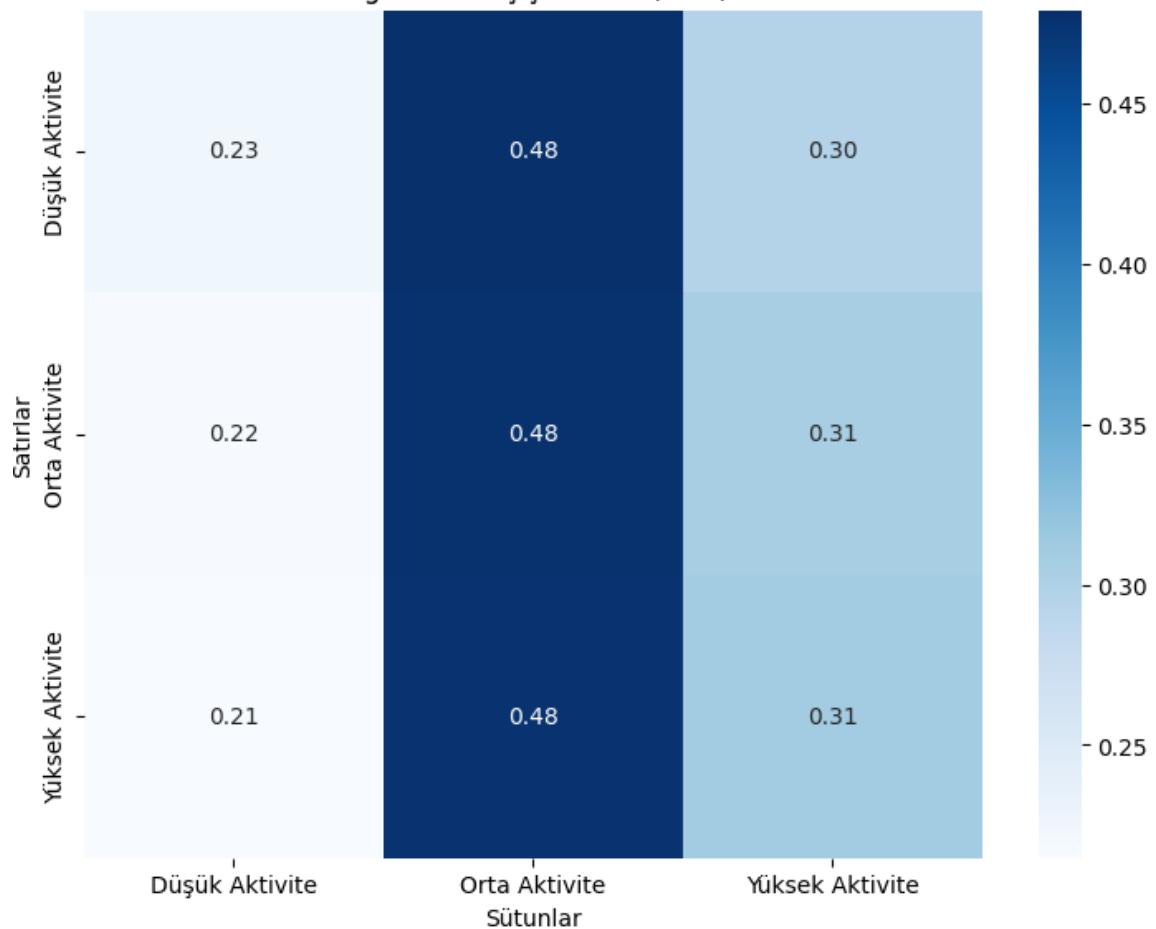
#### **Genel Yorum:**

Bu geçiş olasılıkları matrisi, bireyin günlük aktivite seviyesindeki değişimlerin genellikle mevcut seviyeye yakın seviyelerde gerçekleşme eğiliminde olduğunu göstermektedir. Birey, bulunduğu aktivite seviyesini sürdürme veya bir sonraki en yakın seviyeye geçme konusunda daha yüksek bir olasılığa sahiptir. Büyük sıçramalar (düşükten yükseğe veya yüksektен düşüğe doğrudan geçişler) ise daha az olasıdır. Bu durum, bireyin aktivite alışkanlıklarında bir miktar denge ve süreklilik arayışında olduğunu düşündürmektedir. Özellikle orta aktivite durumunun kararlılığı ve düşük aktiviteden orta aktiviteye geçişin yüksek olasılığı dikkat çekicidir.

#### **Üç-adım geçiş matrisi $P^3$ :**

3 günlük Geçiş Matrisi ( $P^3$ ):			
	Düşük Aktivite	Orta Aktivite	Yüksek Aktivite
Düşük Aktivite	0.225269	0.479061	0.295669
Orta Aktivite	0.216779	0.476556	0.306665
Yüksek Aktivite	0.214838	0.475270	0.309892

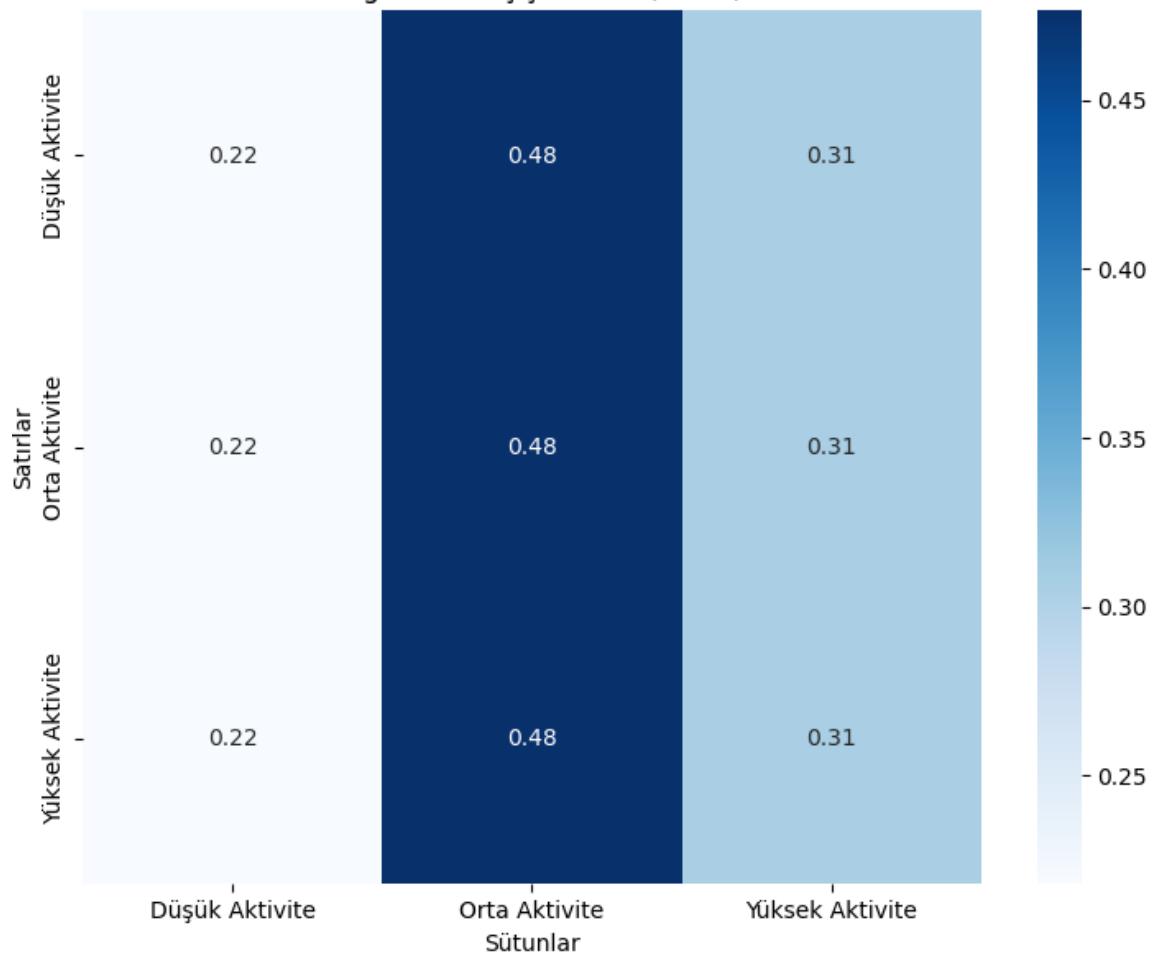
3 günlük Geçiş Matrisi ( $P^3$ )



**On-adım geçiş matrisi  $P^{10}$  :**

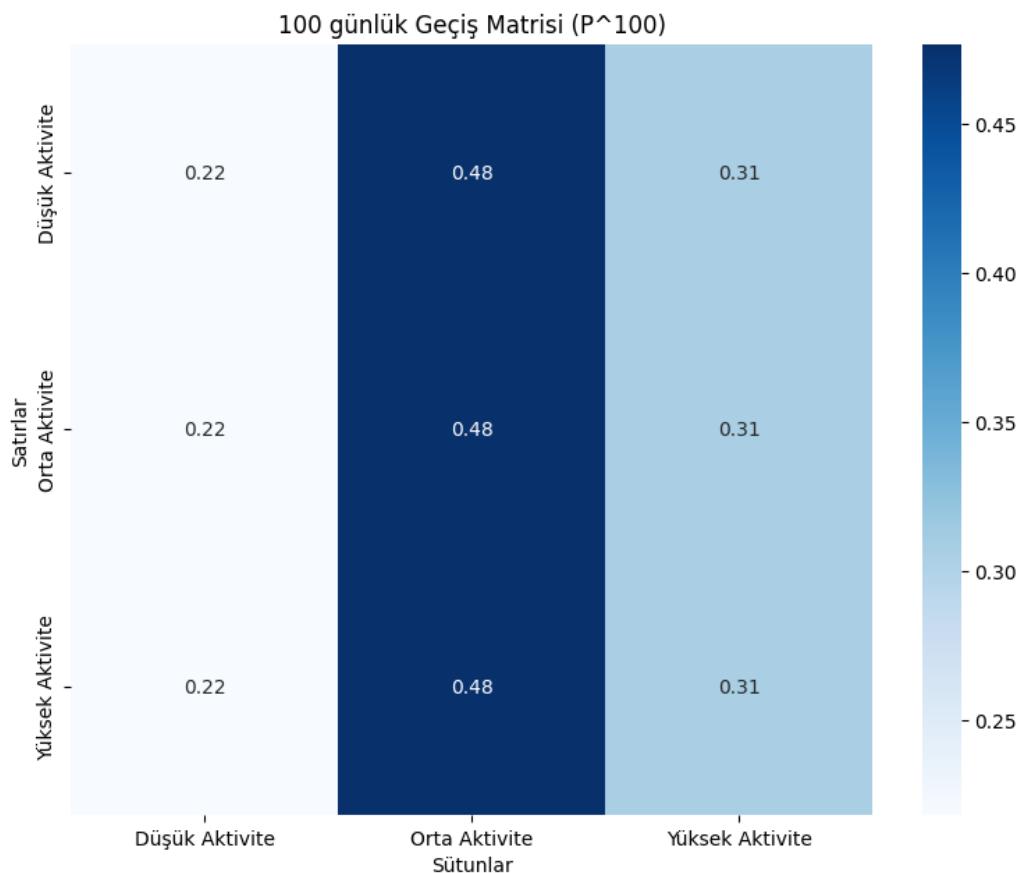
10 günlük Geçiş Matrisi ( $P^{10}$ ):			
	Düşük Aktivite	Orta Aktivite	Yüksek Aktivite
Düşük Aktivite	0.218038	0.47671	0.305252
Orta Aktivite	0.218038	0.47671	0.305253
Yüksek Aktivite	0.218038	0.47671	0.305253

10 günlük Geçiş Matrisi ( $P^{10}$ )



Yüz-adım geçiş matrisi  $P^{100}$  :

100 günlük Geçiş Matrisi ( $P^{100}$ ):			
	Düşük Aktivite	Orta Aktivite	Yüksek Aktivite
Düşük Aktivite	0.218038	0.47671	0.305253
Orta Aktivite	0.218038	0.47671	0.305253
Yüksek Aktivite	0.218038	0.47671	0.305253



**İndirgenebilir mi?**

**1-Kapalı küme yok:**

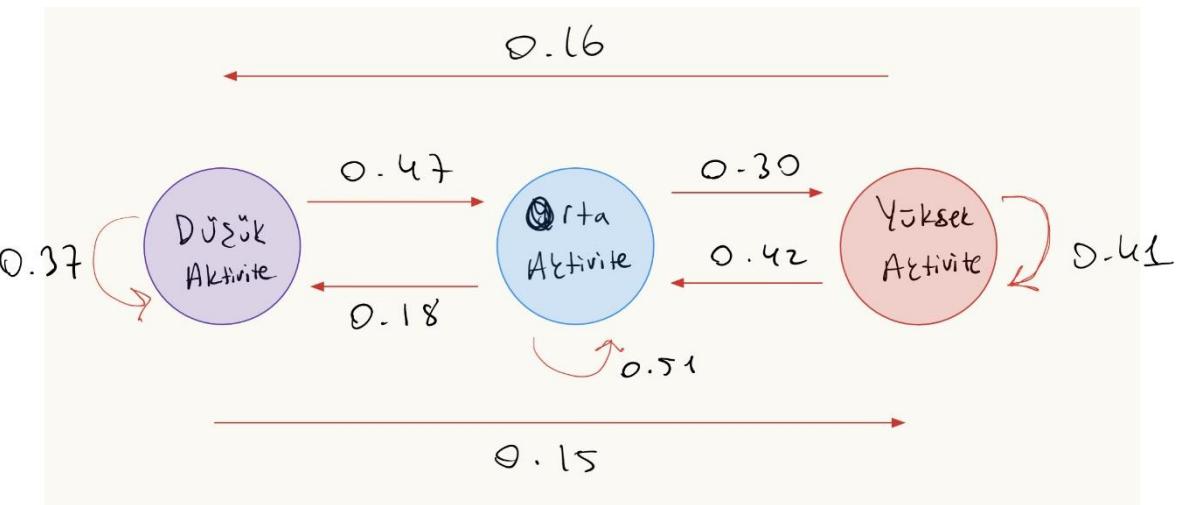
**Geçiş Olasılıkları Matrisi:**

	Düşük Aktivite	Orta Aktivite	Yüksek Aktivite
Düşük Aktivite	0.372727	0.472727	0.154545
Orta Aktivite	0.185031	0.513514	0.301455
Yüksek Aktivite	0.159091	0.422078	0.418831

$$P = \begin{bmatrix} 0.37 & 0.47 & 0.15 \\ 0.18 & 0.51 & 0.30 \\ 0.16 & 0.42 & 0.41 \end{bmatrix}$$

Bu geçiş matrisine göre **kapalı küme yoktur**. Tüm aktivite durumlarından diğerlerine geçiş olasılığı sıfırdan farklıdır, yani sistemde bir kez girildiğinde çıkışın mümkün olmadığı bir durum veya durum grubu bulunmamaktadır.

## 2-Periyodik Durum yok:



Matrisinin k adım sayısı ve  $k \geq 1$  i.durumdan yine i.duruma k'nın katlarında tekrar geçiş yapıyorsa zincire periyodik zincir denir ama burada tüm durumlar 1 adımda kendine tekrar geçtiği için 1'in katlarında yani her adımda kendine dönmedikleri için **aperiyodiktir**.

## 3-Yutucu Durum yok:

Geçiş olasılıkları matrisinin köşegen elemanları (bir durumda kalma olasılıkları) 1 değildir:

- $P(\text{Düşük Aktivite} \rightarrow \text{Düşük Aktivite}) = 0.372727 \neq 1$
- $P(\text{Orta Aktivite} \rightarrow \text{Orta Aktivite}) = 0.513514 \neq 1$
- $P(\text{Yüksek Aktivite} \rightarrow \text{Yüksek Aktivite}) = 0.418831 \neq 1$

Ayrıca, her bir satırda sıfırdan farklı başka olasılıklar da bulunmaktadır. Bu, her bir durumdan diğer durumlara geçişin mümkün olduğu anlamına gelir.

Matristeki hiçbir durum, bir kez girildiğinde çıkışın imkansız olduğu bir özelliğe sahip değildir. Birey, herhangi bir aktivite seviyesinden diğer aktivite seviyelerine geçiş yapabilir. Bu nedenle, bu Markov sürecinde **yutucu bir durum bulunmamaktadır**.

Bu aktivite seviyesi Markov zinciri, **kapalı veya periyodik durum içermediğinden indirgenemez ve aperiyodiktir**.

$$P = \begin{bmatrix} 0.37 & 0.47 & 0.15 \\ 0.18 & 0.51 & 0.30 \\ 0.16 & 0.42 & 0.41 \end{bmatrix}$$

- Düşük Aktivite:**  $P(\text{Düşük} \rightarrow \text{Düşük})=0.372727>0$ . Yani, düşük aktiviteden tekrar düşük aktiviteye dönme olasılığı pozitiftir.
- Orta Aktivite:**  $P(\text{Orta} \rightarrow \text{Orta})=0.513514>0$ . Yani, orta aktiviteden tekrar orta aktiviteye dönme olasılığı pozitiftir.
- Yüksek Aktivite:**  $P(\text{Yüksek} \rightarrow \text{Yüksek})=0.418831>0$ . Yani, yüksek aktiviteden tekrar yüksek aktiviteye dönme olasılığı pozitiftir.

### İndirgenemezlik ve Pozitif Geri Dönüş:

İndirgenemez bir Markov zincirinde, eğer bir durum pozitif geri dönen ise, tüm durumlar pozitif geri dönendir. Bu da Markov zincirinin denge dağılımına sahip olduğunu gösterir.

#### 100 günlük Geçiş Matrisi ( $P^{100}$ ):

	Düşük Aktivite	Orta Aktivite	Yüksek Aktivite
Düşük Aktivite	0.218038	0.47671	0.305253
Orta Aktivite	0.218038	0.47671	0.305253
Yüksek Aktivite	0.218038	0.47671	0.305253

### Denge Dağılımı Yorumu

100 günlük geçiş matrisinde tüm satırların aynı olması, Markov zincirinin uzun vadede kararlı bir duruma ulaştığını göstermektedir. Bu durum, sistemin başlangıç durumundan bağımsız olarak belirli bir dağılıma yaklaştığını ve bu dağılımda kaldığını ifade eder. Bu nedenle elde edilen vektör, sistemin **denge dağılımını** (stationary distribution) temsil eder.

Bu durumda, sistem uzun vadede:

- %21.8 olasılıkla **Düşük Aktivite**,
- %47.7 olasılıkla **Orta Aktivite**,
- %30.5 olasılıkla **Yüksek Aktivite** durumunda olacaktır.

Bu dağılım, geçiş matrisinin sabit noktasıdır ve şu denklemi sağlar:

Denge Dağılımı:	Stationary Probability
Düşük Aktivite	0.218038
Orta Aktivite	0.476710
Yüksek Aktivite	0.305253

İşlemleri yapmamı sağlayan **markov\_sureci.py**:

```

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

def load_data():
    """
    Veriyi yükler ve gerekli ön işlemleri yapar.
    """
    current_directory = os.getcwd()
    df = pd.read_csv(f'{current_directory}/gunluk_veriler.csv')
    States = ["Düşük Aktivite", "Orta Aktivite", "Yüksek Aktivite"]

    # Geçiş çiftlerini oluştur
    transitions = list(zip(df['durum'][:-1], df['durum'][1:]))

    # Geçişleri say
    transition_counts = pd.DataFrame(0, index=States, columns=States)
    for (prev, curr) in transitions:
        transition_counts.loc[prev, curr] += 1

    return df, States, transition_counts

def plot_heatmap(df, title="Heatmap", cmap="Blues", annot=True, fmt=".2f"):
    """
    Verilen bir DataFrame için heatmap çizer.

    Args:
        df (pd.DataFrame): Heatmap çizilecek DataFrame.
        title (str): Heatmap başlığı.
        cmap (str): Renk haritası (örn. "Blues", "coolwarm").
        annot (bool): Hücrelere değerleri yazdırmak için True.
        fmt (str): Hücre değerlerinin formatı (örn. ".2f").
    """
    plt.figure(figsize=(10, 8))
    sns.heatmap(df, annot=annot, fmt=fmt, cmap=cmap, xticklabels=df.columns,
                yticklabels=df.index)
    plt.title(title)
    plt.xlabel("Sütunlar")
    plt.ylabel("Satırlar")
    plt.tight_layout()
    plt.show()

def calculate_stationary_distribution(transition_matrix):
    """
    Geçiş matrisinin denge dağılımını hesaplar.
    """

```

```

    """
    eigvals, eigvecs = np.linalg.eig(transition_matrix.T)
    idx = np.argmin(np.abs(eigvals - 1.0))
    stationary = np.real(eigvecs[:, idx])
    stationary /= stationary.sum()
    return pd.DataFrame(stationary, index=transition_matrix.index,
columns=["Stationary Probability"])

def verify_stationary_distribution(transition_matrix,
stationary_distribution):
    """
    Denge dağılımını doğrular:  $\pi * P = \pi$  denklemini kontrol eder.
    """
    stationary_vector = stationary_distribution.values.flatten()
    return np.allclose(stationary_vector @ transition_matrix.values,
stationary_vector)

def soru2(df, States):
    """
    İlk 30 gün için durum grafiğini çizer.
    """
    df['durum_kod'] = df['durum'].astype('category').cat.codes
    df_first_30 = df.head(30)

    plt.figure(figsize=(12, 6))
    sns.lineplot(x=df_first_30['gun'], y=df_first_30['durum_kod'],
drawstyle="steps-post", linewidth=2)
    plt.yticks(ticks=range(len(States)), labels=States)
    plt.xticks(rotation=45)
    plt.xlabel("Günler")
    plt.ylabel("Durum")
    plt.title("İlk 30 Gün İçin Günlere Göre Durum Grafiği")
    plt.grid()
    plt.tight_layout()
    plt.show()

def soru3(transition_counts):
    """
    Geçiş olasılıkları matrisini ve denge dağılımını hesaplar.
    """
    transition_matrix = transition_counts.div(transition_counts.sum(axis=1),
axis=0).fillna(0)

    print("Geçiş Sayıları Matrisi:\n", transition_counts)
    print("\nGeçiş Olasılıkları Matrisi:\n", transition_matrix)

```

```

plot_heatmap(transition_matrix, title="Geçiş Olasılıkları Heatmap",
cmap="Blues", annot=True, fmt=".2f")

stationary_df = calculate_stationary_distribution(transition_matrix)
print("\nDenge Dağılımı:\n", stationary_df)

valid_stationary = verify_stationary_distribution(transition_matrix,
stationary_df)
print("\nDenge dağılımı geçerli mi?:", valid_stationary)

def soru4(transition_counts):
    """
    3, 10 ve 100 adımlı geçiş matrislerini hesaplar ve görselleştirir.
    """
    transition_matrix = transition_counts.div(transition_counts.sum(axis=1),
axis=0).fillna(0)

    for steps in [3, 10, 100]:
        P = np.linalg.matrix_power(transition_matrix.values, steps)
        P_df = pd.DataFrame(P, index=transition_matrix.index,
columns=transition_matrix.columns)
        print(f"\n{steps} günlük Geçiş Matrisi ( $P^{{steps}}$ ):\n", P_df)
        plot_heatmap(P_df, title=f"{steps} günlük Geçiş Matrisi ( $P^{{steps}}$ )",
cmap="Blues", annot=True, fmt=".2f")

def soru6(verify_stationary_distribution, calculate_stationary_distribution,
transition_matrix, stationary_distribution):
    """
    Denge dağılımını doğrular ve hesaplar.
    """
    print("Denge dağılımı var mı?", verify_stationary_distribution(transition_matrix, stationary_distribution))
    print("Denge dağılımı:", calculate_stationary_distribution(transition_matrix))

def main():
    """
    Ana fonksiyon: Veriyi yükler ve tüm soruları çalıştırır.
    """
    # Veriyi yükle
    df, States, transition_counts = load_data()

    # Soru 2: İlk 30 gün için durum grafiği
    soru2(df, States)

    # Soru 3: Geçiş olasılıkları matrisi ve denge dağılımı

```

```
soru3(transition_counts)

# Soru 4: 3, 10 ve 100 adımlı geçiş matrisleri
soru4(transition_counts)

# Soru 6: Denge dağılımını doğrula
transition_matrix = transition_counts.div(transition_counts.sum(axis=1),
axis=0).fillna(0)
stationary_distribution =
calculate_stationary_distribution(transition_matrix)
soru6(verify_stationary_distribution, calculate_stationary_distribution,
transition_matrix, stationary_distribution)

if __name__ == "__main__":
    main()
```

## Kaynakça

- Kadılar, G. Ö. **2425B\_307\_329\_İST374** Kodlu Ders Notları, Prof. Dr. Gamze Özel Kadılar, Hacettepe Üniversitesi, 2024.
- Kaggle. *Step Count from Phone App Dataset*. <https://www.kaggle.com/datasets/l3llff/step-count-from-phone-app/data> (Erişim Tarihi: 18 Nisan 2025).