Ahmet Burak Şişci

binom dağılımı

bütün dağılımlar belli parametreler ile ifade delir.

bu durum binom için 2 durumdur.

binom dağılımında ki parametreler n ve p dir.

n (Deneme Sayısı) - Parametre: p (Başarı Olasılığı)

Binom dağılımı, sonucu iki ihtimalli olan tekrarlı olaylardaki "şansı" ve "beklentiyi" matematiksel bir modele oturtmamızı sağlar. Bu sayede, sadece tahminde bulunmak yerine, olasılıkları hesaplayarak daha bilinçli ve veriye dayalı kararlar alabiliriz.

expected value = n * p

binom dağılımını uygulayabilmemiz için önecki olayın, yeni olayının sonucunun etkilememesi gerekir

(replace=False)

ikili sonuçlar varsa elimizde (başarılı-başarısız) binom dağılımım kullanılır.

bir dağılıma ait veriyi çağırmak için scipy kullanılır. Bunu unutma !!!

bunun içinde dağılımlarla alakalı modüller var.

bir dağılımdan rastgele sayılar üretmek için rvs fonksiyonu kullanılır.

rvs, "Random Variates" veya "Random Variate Samples" ifadelerinin kısaltmasıdır. Anlamı: Belirli bir olasılık dağılımından (örneğin, normal, binom, uniform) rastgele sayılar üretmek için kullanılan bir fonksiyondur. Kısacası, teorik bir dağılım modeline uyan sahte (simüle edilmiş) veri oluşturur.

random_state Nedir? Kısaca random_state, rvs gibi rastgele sayı üreten fonksiyonların her seferinde aynı rastgele sayı dizisini üretmesini sağlayan bir parametredir. Mantığı: Bilgisayarlardaki "rastgelelik" aslında tamamen rastgele değildir. Belirli bir başlangıç noktasına (seed - tohum) dayanan karmaşık bir algoritma ile üretilir. random_state parametresi, bu başlangıç noktasını manuel olarak sabitlemenizi sağlar. Eğer random_state için belirli bir sayı (örneğin, random_state=42) verirseniz, kodunuzu kim, ne zaman, hangi bilgisayarda çalıştırırsa çalıştırsın, rvs fonksiyonu her zaman birebir aynı rastgele sayıları üretecektir. Eğer random_state'i belirtmezseniz, başlangıç noktası genellikle sistem saatine veya başka bir rastgele kaynağa göre belirlenir, bu yüzden her çalıştırmada farklı rastgele sayılar elde edersiniz. random_state'in rvs Fonksiyonundaki Varsayılan (Default) Değeri scipy.stats içindeki rvs fonksiyonlarında random_state parametresinin varsayılan değeri None'dur. random_state=None ne anlama gelir? Bu, fonksiyonun her çağrıldığında tekrarlanamaz (non-reproducible), yani farklı bir rastgele sayı dizisi üreteceği anlamına gelir. Arka planda, NumPy'ın global rastgele sayı üretecini kullanır. Bu üreteç, genellikle program her başlatıldığında farklı bir "tohum" ile çalışır, bu da sonuçların her seferinde değişmesine neden olur.

In [1]:
```
# k(başarı sayısı) = binom.rvs(n(deneme sayısı), p(başarı olasılığı), size=tekra
```

In [2]:
```
from scipy.stats import binom

binom.rvs(1, 0.5, size=1)
```

Out[2]:
```
array([0])
```

In [3]:
```
# from scipy.stats import binom

binom.rvs(1, 0.5, size=8)
```

Out[3]:
```
array([0, 1, 1, 1, 0, 0, 1, 1])
```

In [4]:
```
# 8 para aynı anda havaya atılıyor
binom.rvs(8, 0.5, size=1)
```

Out[4]:
```
array([4])
```

In [5]:
```
binom.rvs(3, 0.5, size=10)
# 3 tane para elimde. 10 kere attım. kaça kere yazı geldi. bu onun hesabıdır.
```

Out[5]:
```
array([0, 2, 0, 1, 2, 3, 2, 1, 2, 2])
```

In [6]:
```
# örneğin elimizde bir tarafı ağır bir para olsun.
# Bu parada %25 yazı %75 tura gelme olasılığı olsun.
binom.rvs(3, 0.25, size=10)
```

Out[6]:
```
array([0, 1, 0, 0, 0, 0, 0, 0, 0, 0])
```

altta kalan alanı hesaplarken CDF fonksiynnu kullanılır.

probility mest fontuon

Probability Mass Function (PMF) Nedir? (Kısa Tanım)

Olasılık Kütle Fonksiyonu (PMF), bir kesikli rastgele değişkenin, tam olarak belirli bir değere eşit olma olasılığını veren fonksiyondur. Kısacası, şu soruyu cevaplar: "Değişkenin tam olarak x değerini alma olasılığı nedir?" Fonksiyonun çıktısı, o değere ait olasılık "kütlesidir".

Mantığı ve Temel Özellikleri Sadece Kesikli Değişkenler İçin: PMF, sadece sonuçları sayılabilir ve ayrı olan (1, 2, 3 gibi; 1.5 gibi değil) değişkenler için kullanılır. Örnek: Bir zar atışı, bir madeni para atışı, bir metindeki kelime sayısı. Her Sonuca Bir Olasılık Atar: PMF, bir değişkenin alabileceği her bir olası değere, o değerin gerçekleşme olasılığını atar. İki Temel Kuralı Vardır: Her bir sonucun olasılığı 0 ile 1 arasında olmalıdır (0 ≤ P(X=x) ≤ 1). Tüm olası sonuçların olasılıklarının toplamı tam olarak 1'e eşit olmalıdır. Bu, tüm olasılık "kütlesinin" toplandığında bir bütün oluşturması gerektiği anlamına gelir.

pfm alttaki değeri hesaplamaz. kesikli değerin olma ihtiimalini hesaplar.

In [7]:
```
# binom.pmf(k(başarı sayısı), n(deneme sayısı), p(başarı olasılığı))
```

In [8]:
```
# 10 jetondan 7 sinin yazı gelme olasılığı
binom.pmf(7, 10, 0.5) # eğer binom not defined falan derse kütüphaneyi dahil etm
```

Out[8]:
```
np.float64(0.11718749999999999)
```

In [9]:
```
# 7 ve daha az yazı gelme olasılığı

binom.cdf(7, 10, 0.5)
```

Out[9]:
```
np.float64(0.9453125)
```

rvs, size olası k değerlerini üretir. Rastgele ve deneyseldir.

pmf, belirli bir k değerinin ortaya çıkma olasılığını size söyler. Teorik ve kesindir.

normal distrubutuon:

- çan eğrisi şeklindedir.
- oratsı ortalamayı gösterir
- altında kalan alan = 1
- uçlar hiç bir zaman 0 olmaz. 0 a yakınsar

normal dağılımın parametreleri, ortalaması ve standart sapmasıdır.

ortalaması 0, stansrt sapamsı 1 olan normal dağılıma; standart normal dağılım denir.

Ortalaması 0 ve standart sapması 1 olan özel normal dağılıma, Standart Normal Dağılım (Standard Normal Distribution) denir.

Neden Bu Kadar Önemlidir? Farklı ortalamalara ve standart sapmalara sahip sonsuz sayıda normal dağılım vardır (örneğin, insan boyları, sınav notları, üretim hataları vb.). Her biri için ayrı ayrı olasılık hesaplamaları yapmak çok zor ve pratik değildir. Standart normal dağılım, bu sorunu çözen evrensel bir referans noktası görevi görür.

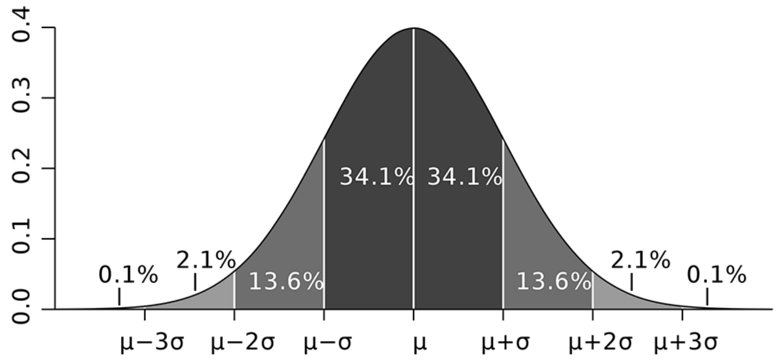
- Ortalama (Mean - μ)
- Standart Sapma (Standard Deviation - σ)

Bu iki parametreyi bildiğinizde, bir normal dağılımın şeklini, konumunu ve yayılımını tamamen tanımlamış olursunuz. Bu yüzden normal dağılım genellikle $N(\mu, \sigma^2)$ olarak gösterilir (burada σ^2 varyanstır, yani standart sapmanın karesi).

"Normal dağılıma uygun normal aralık" genellikle Ampirik Kural (Empirical Rule) veya 68-95-99.7 Kuralı olarak bilinen kavrama karşılık gelir.

görsel2Link

görselLink



Bu kural, normal dağılıma uyan bir veri setinde, verilerin ortalamanın etrafında standart sapma cinsinden nasıl bir aralıkta dağıldığını basit yüzdelerle açıklar.

1. ±1 Standart Sapma Aralığı (En "Normal" Aralık)

Verilerin yaklaşık %68'i, ortalamanın artı veya eksi bir standart sapma aralığında yer alır.

2. ±2 Standart Sapma Aralığı (Çok Geniş Kapsamlı "Normal" Aralık)

Verilerin yaklaşık %95'i, ortalamanın artı veya eksi iki standart sapma aralığında yer alır.

3. ±3 Standart Sapma Aralığı (Neredeyse Tüm Verileri Kapsayan Aralık)

Verilerin yaklaşık %99.7'si, ortalamanın artı veya eksi üç standart sapma aralığında yer alır.

Özetle: "Normal dağılıma uygun normal aralık", verilerin ezici çoğunluğunun bulunduğu, ortalama etrafındaki ±1, ±2 ve ±3 standart sapmalık dilimlerdir. Bu kural, bir verinin ne kadar "tipik" veya ne kadar "aykırı" olduğunu hızlıca anlamak için çok güçlü bir sezgisel araçtır.

normal dağğılım sğrekli bir dağılımdır.

normal dağılımda bir olasılık hesaplamak istersen CDF fonksiyonuunu kullanırsın.

log: ortalama / scale: standart sapma

- loc = Ortalama (μ)
- scale = Standart Sapma (σ)

In [10]:
```
# loc = Ortalama (μ)
# scale = Standart Sapma (σ)
# norm.cdf(k, μ(loc), σ(StdSapma))
```

In [11]:
```
# örneğin ortalaması 161cm ve standart sapması 7 olan bir normal dağılım
# düşünelim. Bu kadınların boylarına ait dağılım olsun.
# kadınların % kaçının 154 cm'den kısa olduğunu bulmak isteyelim.

from scipy.stats import norm
norm.cdf(154, 161, 7)
```

Out[11]:
```
np.float64(0.15865525393145707)
```

In [12]:
```
# 154 cm'den uzun olanların yüzdesi

1 - norm.cdf(154, 161, 7) # üsttekinin tam tersi olan alanı buldum
```

Out[12]:
```
np.float64(0.8413447460685429)
```

In [13]:
```
# 154 ile 157 cm arasındaki kadınların yüzdesi

norm.cdf(157, 161, 7) - norm.cdf(154, 161, 7)
```

Out[13]:
```
np.float64(0.1251993291672192)
```

percent point funstion

Percent Point Function (PPF) Nedir? (Kısa Tanım) PPF, kendisine verilen bir olasılık (veya yüzde) değerine karşılık gelen veri noktasını döndüren fonksiyondur. Kısacası, şu soruyu cevaplar: "Verilerin yüzde p'sinin altında kaldığı nokta (değer) nedir?"

Percent Point Function (PPF), istatistikte CDF (Kümülatif Dağılım Fonksiyonu)'nin tam tersi işi yapan çok önemli bir araçtır

Mantığı: CDF'nin Tersi

Bu kavramı anlamanın en kolay yolu, onu CDF ile karşılaştırmaktır:

- CDF (Kümülatif Dağılım Fonksiyonu):
  - Girdi: Bir veri noktası (x)
  - Çıktı: O noktaya kadar olan toplam olasılık (p)
  - Soru: "Bir öğrencinin notu 75 ise, bu öğrenci sınıfın yüzde kaçından daha başarılıdır?"
  - CDF(75) → 0.85 (Yani, sınıfın %85'inden daha başarılıdır.)
- PPF (Yüzde Noktası Fonksiyonu):
  - Girdi: Bir olasılık (p)
  - Çıktı: O olasılığa karşılık gelen veri noktası (x)
  - Soru: "Sınıfın en başarılı %15'lik dilimine girmek için en az kaç almak gerekir?" (Bu, %85'lik dilimin alt sınırını bulmakla aynıdır.)
  - PPF(0.85) → 75 (Yani, en az 75 almak gerekir.)

In [14]:
```
# p      = Olasılık veya Kuantil (0 ile 1 arasında bir değer)
# loc    = Ortalama (μ)
# scale  = Standart Sapma (σ)
# norm.ppf(p, loc, scale)
```

cdf (Cumulative Distribution Function): Değer → Olasılık ppf (Percent Point Function): Olasılık → Değer

In [15]:
```
from scipy.stats import norm
norm.ppf(0.9, 161.97, 7)
# kadınların %90'ı 169.97 cm'den kısadır
```

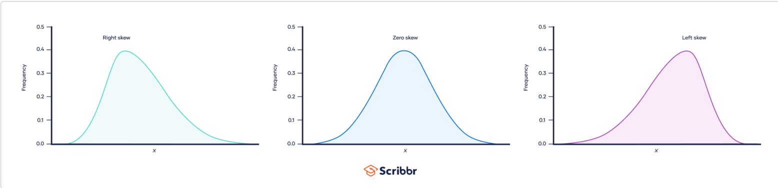
Out[15]:
```
np.float64(170.9408609588122)
```

In [16]:
```
norm.rvs(161, 7, size=10)
```

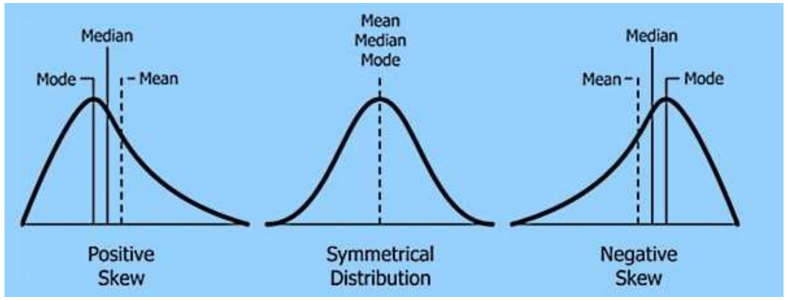
Out[16]:
```
array([169.02608583, 145.03124446, 152.85056412, 176.66487995,
       166.02656074, 157.40539435, 171.59720057, 160.62587346,
       156.84136078, 167.00758525])
```

skewness

Skewness (Çarpıklık), bir olasılık dağılımının veya veri setinin simetriden ne kadar saptığını ölçen istatistiksel bir terimdir. Kısacası, bir dağılımın grafiğinin (örneğin bir histogramın) bir tarafa doğru ne kadar "eğik" veya "kuyruklu" olduğunu anlatır.



verinin grafiğine tokat atmış gibi düşün. sağ tarafından vurup sola devriyorsan pozitif. tam tersi ies negatif çarpık olsur.



1. Pozitif Çarpıklık (Positive Skew / Sağa Çarpık)
    - Görünüm: Grafiğin sağ tarafında uzun bir kuyruk vardır. Verilerin büyük bir kısmı sol tarafa yığılmıştır.
    - Anlamı: Veri setinde, ortalamayı yukarı çeken birkaç tane aşırı yüksek değer bulunur.
    - Ortalama, Medyan, Mod İlişkisi: Ortalama > Medyan > Mod
    - Neden? Aşırı yüksek değerler ortalamayı çok etkiler ve sağa doğru çeker, ancak medyan (orta değer) bu durumdan daha az etkilenir.
    - Örnek: Gelir Dağılımı. Bir ülkedeki insanların çoğu orta veya düşük gelirliyken (sol taraftaki yığılma), çok zengin olan küçük bir azınlık (sağ taraftaki uzun kuyruk) ortalama geliri yukarı çeker.
2. Negatif Çarpıklık (Negative Skew / Sola Çarpık)
    - Görünüm: Grafiğin sol tarafında uzun bir kuyruk vardır. Verilerin büyük bir kısmı sağ tarafa yığılmıştır.
    - Anlamı: Veri setinde, ortalamayı aşağı çeken birkaç tane aşırı düşük değer bulunur.
    - Ortalama, Medyan, Mod İlişkisi: Ortalama < Medyan < Mod
    - Neden? Aşırı düşük değerler ortalamayı sola doğru çeker. Mod ise verilerin yığıldığı sağ tarafta kalır.
    - Örnek: Kolay bir sınavın notları. Öğrencilerin çoğu 80-100 arası yüksek notlar almıştır (sağ taraftaki yığılma), ancak sınava hiç çalışmayan birkaç öğrencinin aldığı çok düşük notlar (sol taraftaki uzun kuyruk) sınıf ortalamasını aşağı çeker.
3. Sıfır Çarpıklık (Zero Skew / Simetrik Dağılım)
    - Görünüm: Grafiğin iki tarafı da birbirinin ayna görüntüsüdür. Kuyruklar her iki tarafta da eşittir.
    - Anlamı: Veriler, ortalamanın etrafına simetrik bir şekilde dağılmıştır. Aykırı değerler dengelidir veya yoktur.
    - Ortalama, Medyan, Mod İlişkisi: Ortalama ≈ Medyan ≈ Mod
    - Örnek: Normal Dağılım (Çan Eğrisi), mükemmel bir simetrik dağılım örneğidir. İnsanların boyları veya IQ skorları gibi birçok doğal olay bu dağılıma uyar.

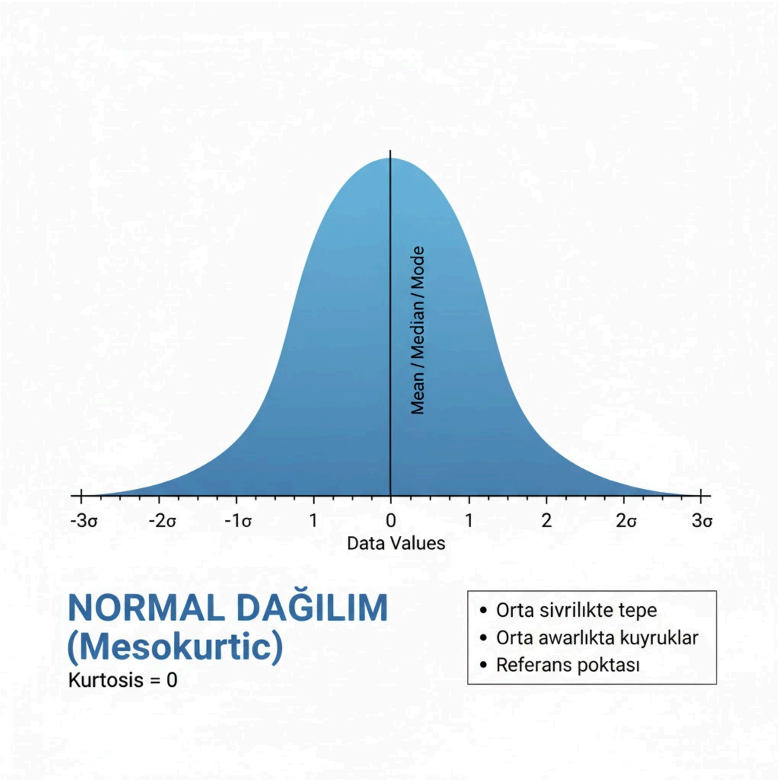
Kurtosis (Basıklık) Nedir?

şu iki soruyu aynı anda cevaplar:

- Veriler merkezde ne kadar yoğun bir şekilde toplanmış? (Tepe noktası)
- Dağılımda ne kadar fazla aşırı (aykırı) değer var? (Kuyruklar)

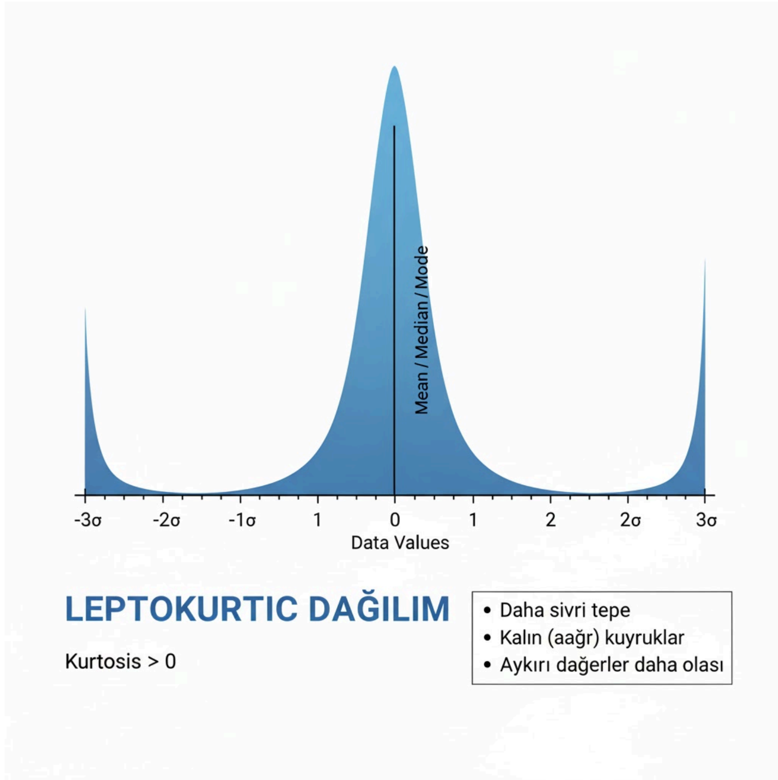
1. Mesokurtic (Normal Basıklık)

- Görünüm: Bu, referans noktamız olan Normal Dağılım'ın (Çan Eğrisi) kendisidir.
- Anlamı: Dağılımın tepe noktası ve kuyrukları "beklendiği" gibidir. Ne çok sivri ne de çok basıktır. Aykırı değerlerin sıklığı normal kabul edilen seviyededir.
- Kurtosis Değeri: Teknik olarak, basıklık katsayısı 3'tür. Ancak, kafa karışıklığını önlemek için istatistik yazılımları genellikle "Fazla Basıklık" (Excess Kurtosis) hesaplar, bu da Kurtosis - 3 demektir. Bu yüzden, bir normal dağılımın fazla basıklığı 0'dır.



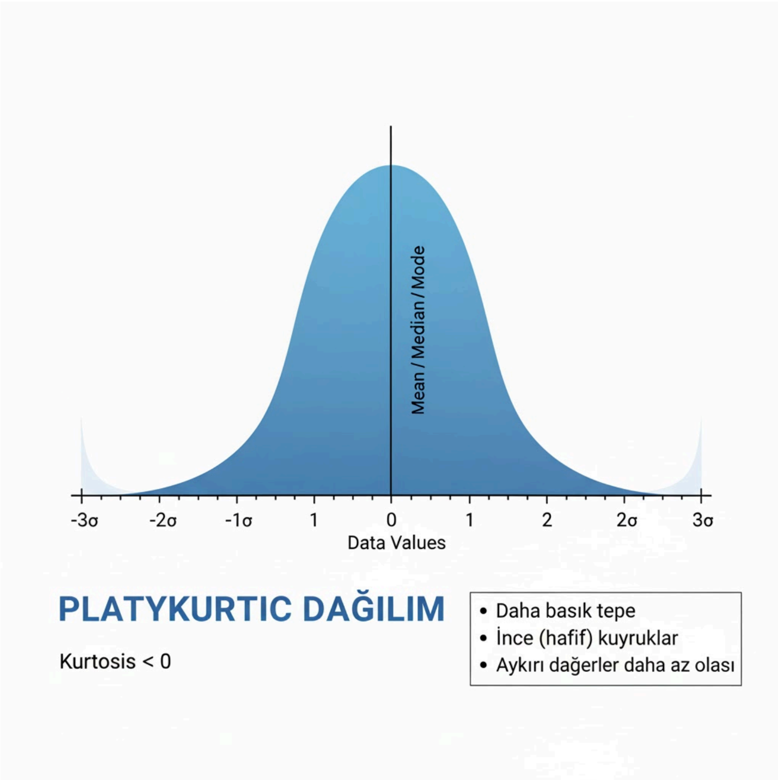
2. Leptokurtic (Pozitif Basıklık / Sivri Tepe)

- Görünüm: Normal dağılıma göre daha sivri bir tepeye ve daha kalın/ağır kuyruklara sahiptir.
- Anlamı: Bu dağılımın iki önemli özelliği vardır:
  - Verilerin büyük bir kısmı ortalamanın çok yakınına yığılmıştır (bu yüzden tepe sivridir).
  - Aynı zamanda, normal dağılıma göre daha fazla aykırı değere (outlier) sahiptir (bu yüzden kuyruklar kalındır). Yani, hem merkezde aşırı bir yığılma hem de uçlarda beklenenden fazla değer vardır.
- Fazla Basıklık Katsayısı: Pozitiftir (> 0).
- Örnek: Hisse senedi getirileri genellikle leptokurtiktir. Çoğu zaman getiriler ortalamaya çok yakındır (küçük günlük değişimler yaşanır, bu da sivri tepeyi oluşturur), ancak ara sıra aşırı büyük kazançlar veya kayıplar (kalın kuyrukları oluşturan "siyah kuğu" olayları) yaşanır.



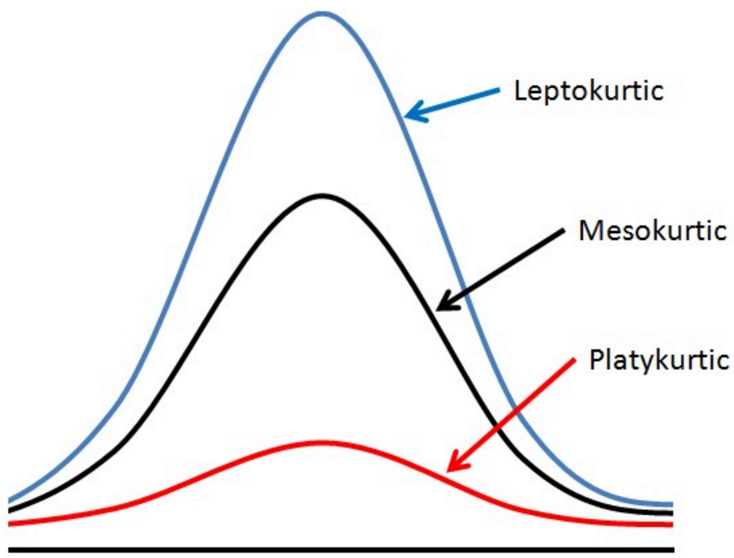
3. Platykurtic (Negatif Basıklık / Basık Tepe)

- Görünüm: Normal dağılıma göre daha basık bir tepeye ve daha ince/hafif kuyruklara sahiptir.
- Anlamı:
  - Veriler, merkezde toplanmak yerine, dağılımın "omuzlarına" doğru daha eşit bir şekilde yayılmıştır (bu yüzden tepe basıktır).
  - Normal dağılıma göre daha az aykırı değere sahiptir (bu yüzden kuyruklar incedir). Aşırı uç değerler çok nadirdir.
- Fazla Basıklık Katsayısı: Negatiftir (< 0).
- Örnek: Bir aralıktaki tüm sonuçların eşit olasılıklı olduğu Düzgün Dağılım (Uniform Distribution), platykurtik dağılımın en uç örneğidir. Grafiği sivri bir tepe yerine düz bir çizgidir ve hiç kuyruğu yoktur.



Neden Önemlidir?

- Risk Analizi: Finansta, yüksek basıklık (leptokurtik), "kuyruk riskinin" (tail risk) yani beklenmedik, aşırı büyük kayıpların yaşanma olasılığının yüksek olduğu anlamına gelir. Bu, risk yönetimi için kritik bir bilgidir.
- Veri Yapısını Anlamak: Basıklık, verilerinizde ne kadar aykırı değer olduğunu ve bu değerlerin ne kadar "aşırı" olduğunu anlamanıza yardımcı olur.
- Model Varsayımları: Tıpkı çarpıklık (skewness) gibi, basıklık da bir verinin normal dağılımdan ne kadar saptığını gösterir. Bu, normal dağılım varsayımı yapan istatistiksel testlerin geçerliliğini etkileyebilir.



bu isimler nereden geliyor:

- Leptokurtic (λεπτοκυρτικός): "İnce", "dar", "narin", "zayıf"
- Mesokurtic (μεσοκυρτικός): "Orta", "ara", "merkezde olan". (Mezopotamya - "iki nehir ortasındaki ülke" kelimesi de aynı kökten gelir)
- Platykurtic (πλατυκυρτικός): "Geniş", "düz", "yassı". (İngilizce'deki "plate" (tabak) ve "platform" kelimeleri de bu kökten türemiştir).

central limit teorem

Merkezi Limit Teoremi Nedir? (Kısa ve Sezgisel Tanım)

Merkezi Limit Teoremi, herhangi bir popülasyondan (dağılımı ne olursa olsun, ister düz, ister çarpık, ister garip şekilli) yeterince büyük ve aynı sayıda örneklemler alıp, bu örneklemlerin ortalamalarını hesapladığınızda, bu ortalamaların oluşturduğu yeni dağılımın her zaman Normal Dağılıma (Çan Eğrisine) benzeyeceğini söyleyen inanılmaz bir teoremdir. Kısacası: Örneklem ortalamalarının dağılımı, normal dağılıma yakınsar.

bir örneklem dağılımının ne akdar çok öeneklem alkarak yaparsanız o dağılımın grafiği o kadar normal dağılşıma yaklaşırç. buna merkezi limit teoremi denir

In [17]:
```
import pandas as pd
import numpy as np

die = pd.Series([1, 2, 3, 4, 5, 6])
samp_5 = die.sample(5, replace=True)
```

```
# .sample(): Bir Seri veya DataFrame'den
# rastgele elemanlar seçmek için kullanılan fonksiyondur.
samp_5
```

Out[17]:

	0
2	3
1	2
4	5
1	2
1	2

dtype: int64

In [18]:
```
samp_5.mean()
```

Out[18]: np.float64(2.8)

In [19]:
```
samp_5_2 = die.sample(5, replace=True)
samp_5_2
```

Out[19]:

	0
5	6
3	4
4	5
1	2
5	6

dtype: int64

In [20]:
```
samp_5_2.mean()
```

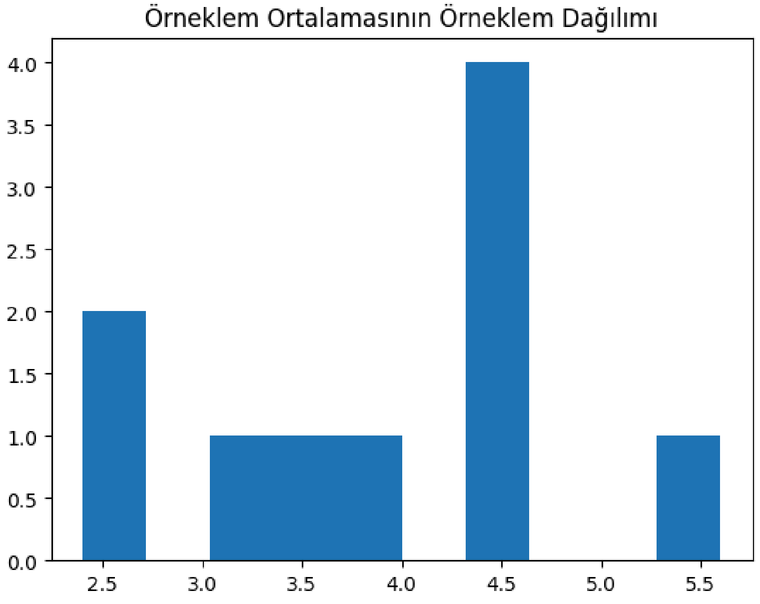
Out[20]: np.float64(4.6)

In [21]:
```
samp_5_3 = die.sample(5, replace=True)
samp_5_3.mean()
```

Out[21]: np.float64(3.6)

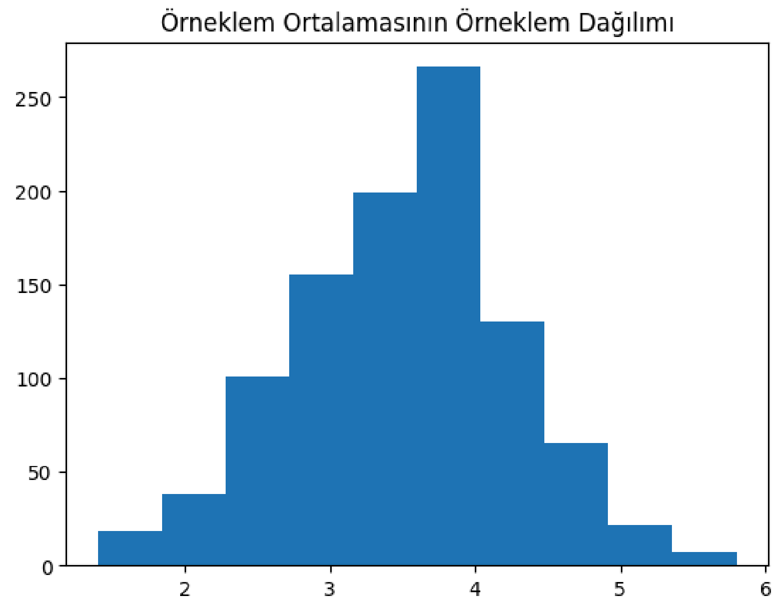
In [22]:
```
# Yukarıdaki işlem 10 defa yapılıp ortalama alınırsa
import matplotlib.pyplot as plt

sample_means = []
for i in range(10):
    samp_5 = die.sample(5, replace=True)
    sample_means.append(samp_5.mean())
plt.title("Örneklem Ortalamasının Örneklem Dağılımı")
plt.hist(sample_means)
# Böyle bir özet istatistiğinin dağılımına örneklem dağılımı denir;
```

Out[22]:
```
(array([2., 0., 1., 1., 1., 0., 4., 0., 0., 1.]),
 array([2.4 , 2.72, 3.04, 3.36, 3.68, 4.  , 4.32, 4.64, 4.96, 5.28, 5.6 ]),
 <BarContainer object of 10 artists>)
```



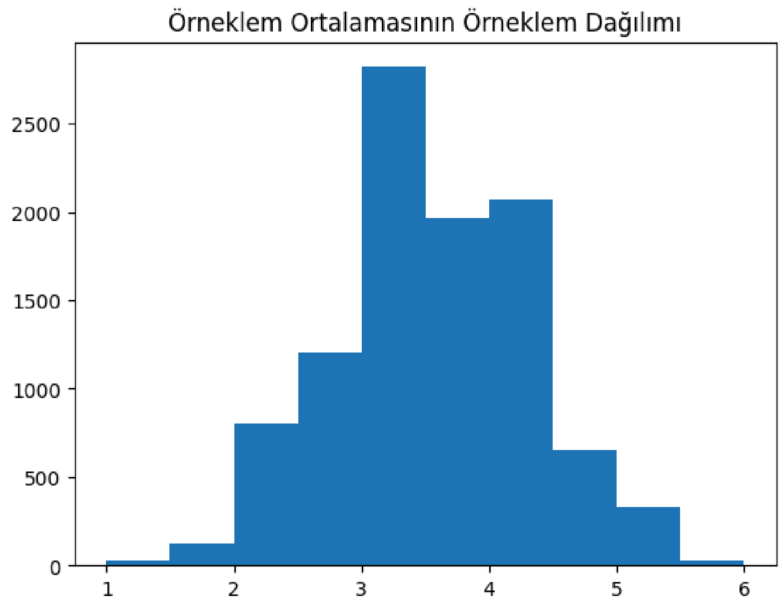
In [23]:
```
sample_means = []
for i in range(1000):
    samp_5 = die.sample(5, replace=True)
    sample_means.append(samp_5.mean())
plt.title("Örneklem Ortalamasının Örneklem Dağılımı")
plt.hist(sample_means)
```

Out[23]:
```
(array([ 18.,  38., 101., 155., 199., 266., 130.,  65.,  21.,   7.]),
 array([1.4 , 1.84, 2.28, 2.72, 3.16, 3.6 , 4.04, 4.48, 4.92, 5.36, 5.8 ]),
 <BarContainer object of 10 artists>)
```



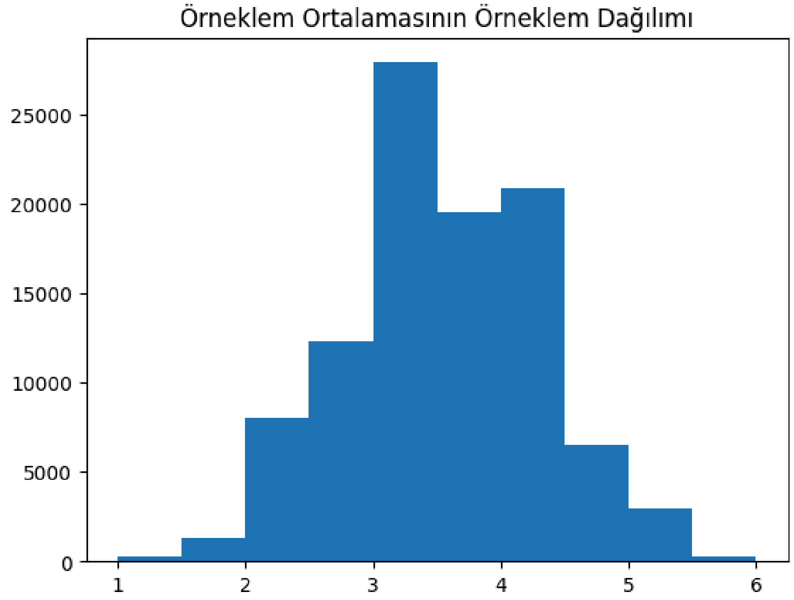
In [24]:
```python
sample_means = []
for i in range(10000):
    samp_5 = die.sample(5, replace=True)
    sample_means.append(samp_5.mean())
plt.title("Örneklem Ortalamasının Örneklem Dağılımı")
plt.hist(sample_means)
```

Out[24]:
```
(array([  24.,  125.,  803., 1200., 2819., 1962., 2066.,  648.,  327.,
          26.]),
 array([1. , 1.5, 2. , 2.5, 3. , 3.5, 4. , 4.5, 5. , 5.5, 6. ]),
 <BarContainer object of 10 artists>)
```



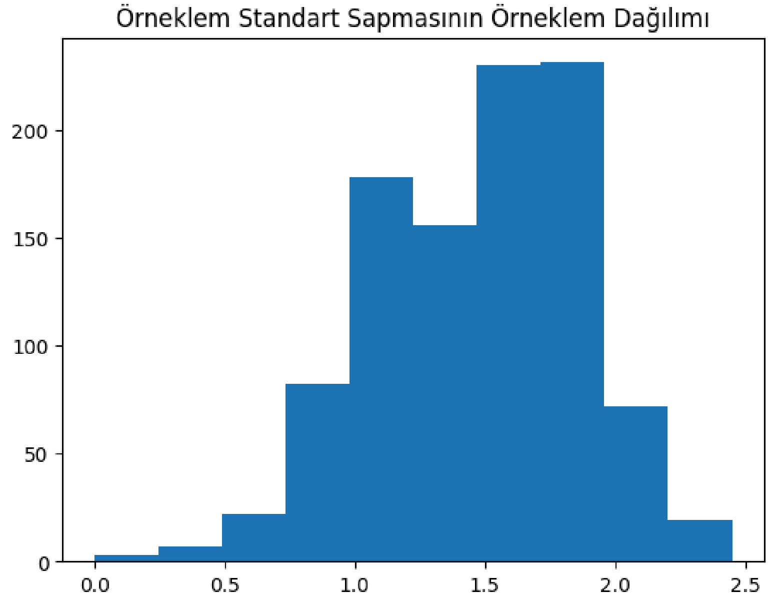
In [25]:
```python
sample_means = []
for i in range(100000):
    samp_5 = die.sample(5, replace=True)
    sample_means.append(samp_5.mean())
plt.title("Örneklem Ortalamasının Örneklem Dağılımı")
plt.hist(sample_means)
```

Out[25]:
```
(array([  254.,  1296.,  8045., 12336., 27907., 19508., 20897.,  6548.,
         2920.,   289.]),
 array([1. , 1.5, 2. , 2.5, 3. , 3.5, 4. , 4.5, 5. , 5.5, 6. ]),
 <BarContainer object of 10 artists>)
```




```
In [26]: sample_sts = []
         for i in range(1000):
             samp_5 = die.sample(5, replace=True)
             sample_sts.append(np.std(samp_5))
         plt.title("Örneklem Standart Sapmasının Örneklem Dağılımı")
         plt.hist(sample_sts)
```

```
Out[26]: (array([  3.,   7.,  22.,  82., 178., 156., 230., 231.,  72.,  19.]),
          array([0.        , 0.24494897, 0.48989795, 0.73484692, 0.9797959 ,
                 1.22474487, 1.46969385, 1.71464282, 1.95959179, 2.20454077,
                 2.44948974]),
          <BarContainer object of 10 artists>)
```



Kodun Kısa Açıklaması Bu kod, die (zar) adını verdiğimiz bir popülasyondan, 1000 kez boyunca 5'erli rastgele örneklemler alır. Her bir 5'li örneklemin standart sapmasını hesaplar ve bu 1000 adet standart sapma değerini bir listede biriktirir. Son olarak, bu 1000 standart sapma değerinin dağılımını bir histogram grafiği ile görselleştirir. Sonuçta ortaya çıkan grafik, "5'lik bir zar atışı setinin standart sapmasının tipik olarak nasıl bir dağılım gösterdiğini" anlatır.

```
In [27]: import pandas as pd
         sales_team = pd.Series(['Amir', 'Brian', 'Claire', 'Damian'])
         sales_team.sample(10, replace=True)
```

Out[27]:

	0
1	Brian
0	Amir
2	Claire
1	Brian
0	Amir
2	Claire
2	Claire
1	Brian
2	Claire
1	Brian

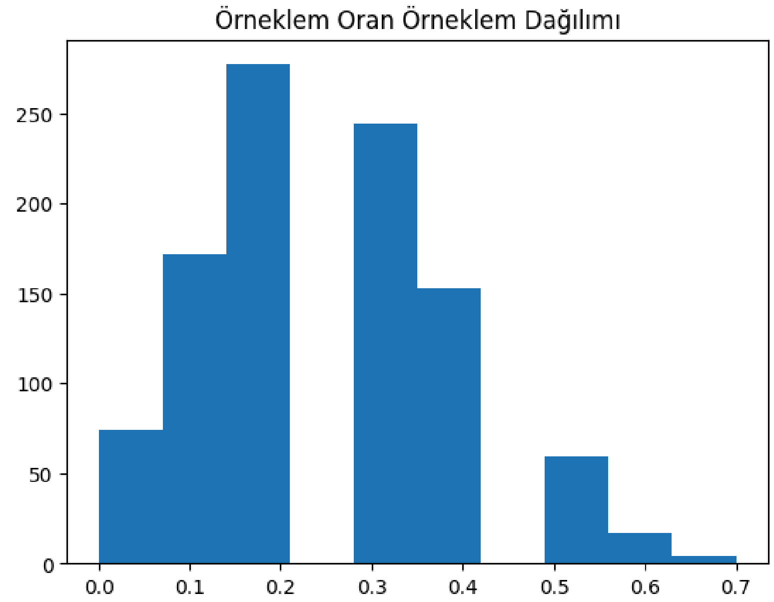
dtype: object

In [28]:

```
sample_prp = []
for i in range(1000):
    samp_5 = sales_team.sample(10, replace=True)
    try:
        sample_prp.append(samp_5.value_counts()['Claire'] / 10)
    except:
        sample_prp.append(0)
plt.title("Örneklem Oran Örneklem Dağılımı")
plt.hist(sample_prp)
```

Out[28]:

```
(array([ 74., 172., 277.,   0., 244., 153.,   0.,  59.,  17.,   4.]),
 array([0.  , 0.07, 0.14, 0.21, 0.28, 0.35, 0.42, 0.49, 0.56, 0.63, 0.7 ]),
 <BarContainer object of 10 artists>)
```



try-except, programlamada "Bir işi yapmayı dene, eğer bir hata çıkarsa programı çökertmek yerine şunu yap" demenin bir yoludur.

poisson distrubition

Poisson Dağılımı Nedir? (Kısa ve Sezgisel Tanım)

Poisson Dağılımı, belirli bir sabit zaman aralığında veya belirli bir sabit alanda, olayların ortalama gerçekleşme oranı biliniyorken, bu olayın tam olarak belirli bir sayıda (k) gerçekleşme olasılığını hesaplar. Kısacası, şu tür soruları cevaplar:

- Bir çağrı merkezine bir saat içinde ortalama 10 arama geliyorsa, bir sonraki saat içinde tam olarak 7 arama gelme olasılığı nedir?
- Bir web sitesi bir dakikada ortalama 20 ziyaretçi alıyorsa, bir sonraki dakika içinde hiç ziyaretçi gelmeme olasılığı nedir?
- Bir kitap sayfasında ortalama 0.5 yazım hatası bulunuyorsa, rastgele seçilen bir sayfada tam olarak 2 yazım hatası bulma olasılığı nedir?

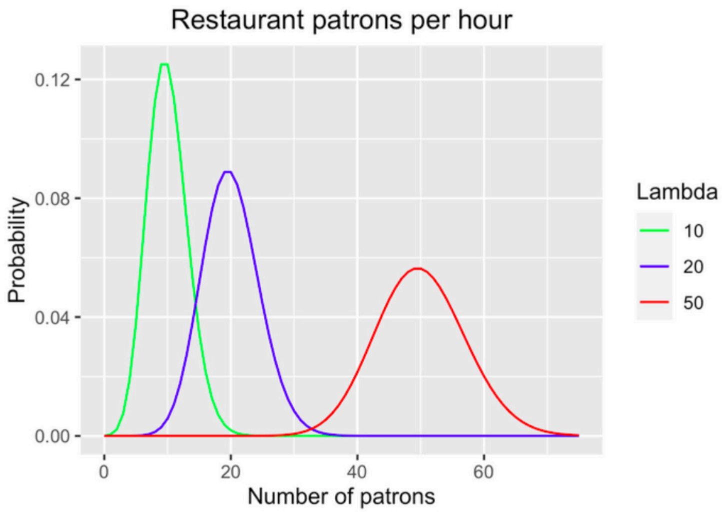
Tek Parametresi: λ (Lambda)

Binom dağılımının n ve p gibi iki parametresi varken, Poisson dağılımının sadece tek bir parametresi vardır:

- λ (Lambda): Belirtilen zaman aralığı veya alan içindeki ortalama olay sayısıdır. Bu, dağılımın hem ortalaması hem de varyansıdır.

In [29]:
```
# k       = Olay Sayısı (Olasılığı hesaplanacak tam sayı)
# mu      = Ortalama Olay Sayısı (λ - Lambda)

# poisson.pmf(k, mu)
```

ödev: haftaya kadar hocanın gruba attığı grafiği(resim) yorumla



In [30]:
```
# Haftada ortalama 8 sahiplenmenin gerçekleştiği bir sığınakta
# bir haftada 5 sahiplenme gerçekleşme olasılığı nedir?

from scipy.stats import poisson

poisson.pmf(5, 8)
```

Out[30]:
```
np.float64(0.09160366159257921)
```

In [31]:
```
# 5 veya daha az sahiplenme olasılığı

poisson.cdf(5, 8)

#5'den fazla sahiplenme olasılığı

1 - poisson.cdf(5, 8)
```

Out[31]:
```
np.float64(0.8087639379203747)
```

rvs: rastgele sayı üreteir. (unutma)

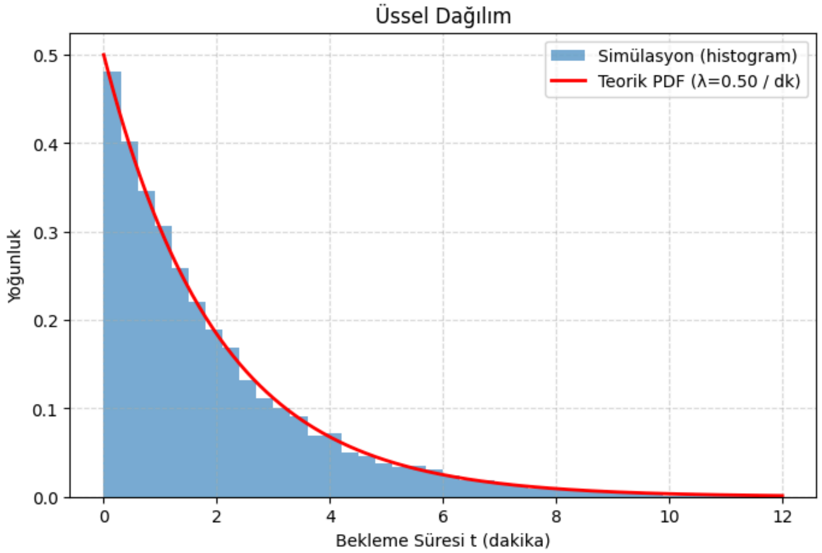
örnek sınav sorusu: grafiği çiz

Müşteri hizmetleri talepleri

Örneğin, her 2 dakikada bir müşteri hizmetleri bileti oluşturulsun. Bu bir dakikalık bir zaman aralığı cinsinden yeniden ifade edilebilir, böylece her dakika bir biletin yarısı oluşturulur. Lambda değeri olarak 0.5 kullanılır. Yarım oranlı üstel dağılım şu şekilde görünür.

hocanın verdiği cevap

In [32]:
```
from scipy.stats import expon
import matplotlib.pyplot as plt
import numpy as np

ortalama = 2
lmb = 1 / 2

x = np.linspace(0, 12, 400)

orneklem = expon.rvs(scale=ortalama, size=10000, random_state=42)

pdf = expon.pdf(x, scale=ortalama)
cdf = expon.cdf(x, scale=ortalama)

plt.figure(figsize=(8, 5))
plt.hist(orneklem, bins=40, range=(0, 12), density=True, alpha=0.6, label="Simül
plt.plot(x, pdf, 'r-', lw=2, label=f"Teorik PDF (λ={lmb:.2f} / dk)")

plt.title("Üssel Dağılım")
plt.xlabel("Bekleme Süresi t (dakika)")
plt.ylabel("Yoğunluk")
plt.legend()
plt.grid(True, linestyle="--", alpha=0.5)
```



Üssel Dağılım
Simülasyon (histogram)
Teorik PDF (λ=0.50 / dk)
Yoğunluk
0.0
0.1
0.2
0.3
0.4
0.5
0
2
4
6
8
10
12
Bekleme Süresi t (dakika)