VERİ BİLİMİ İÇİN İSTATİSTİK

1.ÖRNEKLEM TEORİSİ

In [328…
```
import numpy as np
```

In [297…
```
population = np.random.randint(0,80,10000)
```

1.1 Örneklem Seçimi

In [298…
```
np.random.seed(115)
```

np.random.seed() her seferinde rastgele değer çekilmesini engellemek adına kullanılır.

In [299…
```
orneklem = np.random.choice(a = population, size = 100)
```

var1 = np.random.choice(a = dataset, size= X) şeklinde kullanılır. Bu şu demek; *verilen veri setinden rastgele olarak X boyutundan veri çek ve bunu bir değişkene ata*

In [300…
```
orneklem[0:100]
```

Out[300]:
```
array([64, 73, 57, 72, 35, 49, 63, 54, 67, 12, 66,  3, 18, 54, 34, 39, 11,
       19, 57,  0, 39, 64,  7, 17, 35, 72, 44,  0, 42, 79, 73, 59,  5, 59,
        4, 67, 71, 13, 29, 40, 49, 61, 26,  0, 47, 40, 35, 75, 38, 27, 63,
       74, 25, 34, 58, 41, 38, 29, 65, 52, 15, 52,  5, 56, 36, 44, 10, 16,
       44, 17, 29, 44, 34, 78, 42, 26, 68, 74, 47, 53, 68, 40,  7, 65, 34,
       74, 77, 27, 25, 68,  2,  4, 22,  6, 48, 18, 33, 49, 19,  3])
```

In [301…
```
orneklem.mean()
```

Out[301]:
```
40.22
```

In [302…
```
population.mean()
```

Out[302]:
```
39.3583
```

1.2 Örneklem Dağılımı

In [303…
```
np.random.seed(10)
orneklem1 = np.random.choice(a = population, size = 100)
orneklem2 = np.random.choice(a = population, size = 100)
orneklem3 = np.random.choice(a = population, size = 100)
orneklem4 = np.random.choice(a = population, size = 100)
orneklem5 = np.random.choice(a = population, size = 100)
orneklem6 = np.random.choice(a = population, size = 100)
orneklem7 = np.random.choice(a = population, size = 100)
orneklem8 = np.random.choice(a = population, size = 100)
orneklem9 = np.random.choice(a = population, size = 100)
orneklem10 = np.random.choice(a = population, size = 100)
(orneklem1.mean()+orneklem2.mean()+orneklem3.mean()+orneklem4.mean()+orneklem5.mean()+orneklem6.mean()+orneklem
```

Out[303]:
```
39.94199999999999
```

Önce Population veri setimizden 10 farklı örnek aldık. Daha sonra bu örneklemlerin kendi ortalamarını alıp, bunları birbirleri ile toplayıp, toplam değerimizin ortalamasını aldık. Bu **Merkezi Limit Teorimi** uygulamasına bir örnektir. Daha fazla örneklem çektiğimizde, bulduğumuz ortalamanın population ortalamasına daha yakın olması beklenir.

2. BETİMSEL İSTATİSTİKLER

In [304…
```
import seaborn as sns
tips = sns.load_dataset("tips")
df = tips.copy()
```

Seaborn kütüphanesi içersinde yer alan "tips" verisetimizi projeye dahil ediyoruz. Tips veriseti veri bilimi eğitimi için kullanılan bir verisetidir.

In [305…
```
df.describe().T
```

Out[305]:

	count	mean	std	min	25%	50%	75%	max
total_bill	244.0	19.785943	8.902412	3.07	13.3475	17.795	24.1275	50.81
tip	244.0	2.998279	1.383638	1.00	2.0000	2.900	3.5625	10.00
size	244.0	2.569672	0.951100	1.00	2.0000	2.000	3.0000	6.00

describe() fonksiyonu bize verisetine dair istatiksel bilgileri verir. **describe().T** yaptığımızda ise bize verilen istatiksel bilgileri transpozunu alarak sunar, bu da verisetinin istatiksel bilgilerini okumamızda kolaylık sağlar.

In [306…

```
!pip install researchpy
import researchpy as rp
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: researchpy in c:\users\tahat\appdata\roaming\python\python39\site-packages (0.3.
5)
Requirement already satisfied: patsy in c:\programdata\anaconda3\lib\site-packages (from researchpy) (0.5.2)
Requirement already satisfied: pandas in c:\programdata\anaconda3\lib\site-packages (from researchpy) (1.4.4)
Requirement already satisfied: numpy in c:\programdata\anaconda3\lib\site-packages (from researchpy) (1.21.5)
Requirement already satisfied: scipy in c:\programdata\anaconda3\lib\site-packages (from researchpy) (1.9.1)
Requirement already satisfied: statsmodels in c:\programdata\anaconda3\lib\site-packages (from researchpy) (0.1
3.2)
Requirement already satisfied: pytz>=2020.1 in c:\programdata\anaconda3\lib\site-packages (from pandas->researc
hpy) (2022.1)
Requirement already satisfied: python-dateutil>=2.8.1 in c:\programdata\anaconda3\lib\site-packages (from panda
s->researchpy) (2.8.2)
Requirement already satisfied: six in c:\programdata\anaconda3\lib\site-packages (from patsy->researchpy) (1.16
.0)
Requirement already satisfied: packaging>=21.3 in c:\programdata\anaconda3\lib\site-packages (from statsmodels-
>researchpy) (21.3)
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in c:\programdata\anaconda3\lib\site-packages (from pac
kaging>=21.3->statsmodels->researchpy) (3.0.9)
```

In [307…

```
rp.summary_cont(df[["total_bill","tip","size"]])
```

Out[307]:

	Variable	N	Mean	SD	SE	95% Conf.	Interval
0	total_bill	244.0	19.7859	8.9024	0.5699	18.6633	20.9086
1	tip	244.0	2.9983	1.3836	0.0886	2.8238	3.1728
2	size	244.0	2.5697	0.9511	0.0609	2.4497	2.6896

burada verilen istatiksel bilgiler describe() fonksiyonu ile benzerlik gösterir, fakat bu veriseti göz önüne alındığında bu bilgiler daha anlamlı sonuçlar gösteriyor. bu sayısal değişkenler için gözlemlenen istatiksel değerlerdir.

In [308…

```
rp.summary_cat(df[["sex","smoker","day"]])
```

Out[308]:

	Variable	Outcome	Count	Percent
0	sex	Male	157	64.34
1		Female	87	35.66
2	smoker	No	151	61.89
3		Yes	93	38.11
4	day	Sat	87	35.66
5		Sun	76	31.15
6		Thur	62	25.41
7		Fri	19	7.79

kategorik değişkenlerin sınıf frekanslarını görüyoruz

In [309…

```
df[["tip","total_bill"]].cov()
```

Out[309]:

	tip	total_bill
tip	1.914455	8.323502
total_bill	8.323502	79.252939

cov() fonksiyonu ile iki değişken arasındaki kovaryans bilgisine erişiyoruz. Kovaryans; *iki değişken arasındaki ilişkinin değişkenlik değeridir*

In [310…

```
df[["tip","total_bill"]].corr()
```

Out[310]:

	tip	total_bill
tip	1.000000	0.675734
total_bill	0.675734	1.000000

corr() fonksiyonu ise bize iki değişken arasındaki korelasyon bilgisini verir. Korelasyon; *iki değişken arasındaki ilişkiyi, ilişkinin anlamlı olup olmadığını, ilişkinin şiddetini ve yönünü ifade eder.*

2.1 UYGULAMA: FİYAT STRATEJİSİ

Problem :

- Şirket CEO'su fiyat belirleme konusunda bilimsel bir dayanak ve esneklik istiyor.

Detaylar :

- Satıcı,alıcı ve ürün var.
- Alıcılara "Ürüne ne kadar ödersiniz?" diye soruluyor.(Alıcılar diğer fiyat teklifini göremiyor.)
- Optimum fiyat bilimsel ve esnek olarak bulunmak isteniyor.

In [311…
```
fiyatlar = np.random.randint(10,110,1000)
```

elimizde bir ürün var ve 1000 alıcıya bu ürüne ne kadar fiyat biçtiklerini sorduğumuzda bize 10 ve 110 arasında fiyat teklifleri geliyor.

In [312…
```
fiyatlar.mean()
```

Out[312]:
```
58.492
```

In [313…
```
import statsmodels.stats.api as sms
```

In [314…
```
sms.DescrStatsW(fiyatlar).tconfint_mean()
```

Out[314]:
```
(56.67953887736034, 60.30446112263965)
```

Burada gözlemlenen çıktı **güven aralığı**dır. bu bize bilimsel bir dayanak sunar. oranı kabul gören değer %95'tir. yani çıktı üzerinden konuşacak olursak; 100 müşteriden 95 tanesi 56,68 TL ile 60,30TL verebilir.

3. OLASILIK DAĞILIMLARI

3.1 Bernoulli Dağılımı

In [315…
```
from scipy.stats import bernoulli
```

In [316…
```
p = 0.3
```

In [317…
```
yt = bernoulli(p)
yt.pmf(k = 0)
```

Out[317]:
```
0.7
```

Bernoulli dağılımı iki sonuçlu olayları inceleyen kesikli bir olasılık dağılımıdır.

p = tura durumumuz olsun ve deneyler sonucunda 0.3 oranında bir tura gelme istatiğine sahibiz diyelim.

pmf = olasılık kütle fonksiyonu

k = 1 -> tura gelme durumu, k = 0 -> yazı gelme durumu olur. (tura olmayan.)

Matematiksel olarak ifade edecek olursak = TÜM OLAYLAR EKSİ-BEKLENEN OLAY. (yazı için ; 1-p, yazı için ise 1-(1-p))

3.2 Büyük Sayılar Yasası

Bir rassal değişkenin uzun vadeli kararlığını tanımlayan olasılık teoremidir.

In [318…
```
import numpy as np
rng = np.random.RandomState(123)
for i in np.arange(1,21):
    deney_sayisi = 2**i
    yazi_tura = rng.randint(0,2, size = deney_sayisi)
    yazi_olasilik = np.mean(yazi_tura)
    print("Atış Sayısı = ", deney_sayisi,"---","Yazı Olasılığı: %.2f" % (yazi_olasilik))
```

```
Atış Sayısı =  2 --- Yazı Olasılığı: 0.50
Atış Sayısı =  4 --- Yazı Olasılığı: 0.00
Atış Sayısı =  8 --- Yazı Olasılığı: 0.62
Atış Sayısı =  16 --- Yazı Olasılığı: 0.44
Atış Sayısı =  32 --- Yazı Olasılığı: 0.47
Atış Sayısı =  64 --- Yazı Olasılığı: 0.56
Atış Sayısı =  128 --- Yazı Olasılığı: 0.51
Atış Sayısı =  256 --- Yazı Olasılığı: 0.53
Atış Sayısı =  512 --- Yazı Olasılığı: 0.53
Atış Sayısı =  1024 --- Yazı Olasılığı: 0.50
Atış Sayısı =  2048 --- Yazı Olasılığı: 0.49
Atış Sayısı =  4096 --- Yazı Olasılığı: 0.49
Atış Sayısı =  8192 --- Yazı Olasılığı: 0.50
Atış Sayısı =  16384 --- Yazı Olasılığı: 0.50
Atış Sayısı =  32768 --- Yazı Olasılığı: 0.50
Atış Sayısı =  65536 --- Yazı Olasılığı: 0.50
Atış Sayısı =  131072 --- Yazı Olasılığı: 0.50
Atış Sayısı =  262144 --- Yazı Olasılığı: 0.50
Atış Sayısı =  524288 --- Yazı Olasılığı: 0.50
Atış Sayısı =  1048576 --- Yazı Olasılığı: 0.50
```

Deney sayısı arttıkça ortaya konulması beklenen olasılıksal ifadeler kendini açığa çıkarmak zorundadır. Yani bizim yazı turada beklediğimiz sonuç %50 %50dir. Parayı 4 kere attığımızda görüyoruz ki hiç yazı gelmemiş. Bu durumda biz bundan sonra hep tura gelecek diyemeyiz. Deney sayısı arttıkça değerin %50 olduğunu görüyoruz. Bu Büyük Sayısal Yasası'nın mantıklı ve dayanıklı bir açıklamasıdır. Ayrıca bu işlem tekrar tekrar yapıldığı için **Binom Dağılımı**na örnektir.

3.2.1 UYGULAMA: REKLAM HARCAMASI

Problem :

- Çeşitli mecralara reklam veriliyor, reklamların tıklanma ve geri dönüşüm oranları optimize edilmeye çalışılıyor. Buna yönelik olarak belirli bir mecrada çeşitli senaryolara göre reklama tıklama olasılıkları hesaplanmak isteniyor.

Detaylar :

- Bir mecrada reklam verilecek.
- Dağılım ve reklama tıklanma olasılığı biliniyor.(0.01)
- Soru : Reklamı 100 kişi gördüğünde 1,5,10 tıklanması olasığı nedir?

In [319...
```python
from scipy.stats import binom
```

In [320...
```python
p = 0.01
n = 100
rv = binom(n,p)
print(rv.pmf(1))
print(rv.pmf(5))
print(rv.pmf(10))
```

```
0.36972963764972666
0.002897787123761478
7.006035693977194e-08
```

3.3 Poisson Dağılımı

Belirli bir zaman aralığında belirli bir alanda nadiren rastlanan olayların olasılıklarını hesaplamak için kullanılır. (Nadir olay = n>50 ve n*p<5 olmak zorunda)

Örnek :

- 10 bin kelimeden oluşan kitapta hatalı kelime sayısı
- 4 bin öğrencili bir okulda notun yanlış girilmesi
- Kredi kartı işlemlerinde sahtekarlık

3.3.1 UYGULAMA: HATALI İLAN GİRİŞİ SAYISI

Problem:

- Hatalı ilan girişi olasılıkları hesaplanmak isteniyor.

Detaylar:

- Bir yı süresince ölçümler yapılıyor.
- Dağılım biliniyor(Poisson) ve lambda 0.1
- Hiç hata olmaması, 3 hata ve 5 hata durumu

In [321...
```python
from scipy.stats import poisson
```

```
In [322…  rv = poisson (mu = 0.1)
print(rv.pmf(k = 0))
print(rv.pmf(k = 3))
print(rv.pmf(k = 5))
```

```
0.9048374180359595
0.00015080623633932676
7.54031181696634e-08
```

3.4 Normal Dağılım

Normal dağıldığı bilinen sürekli rassal değişkenler için olasılık hesaplanmasında kullanılır.

3.4.1 UYGULAMA: SATIŞ OLASILIKLARININ HESAPLANMASI

Problem:

- Bir yatırım öncesinde gelecek ay ile iligli satışkarın belirli değerlerde gerçekleşmesi olasılıkları belirlenmek isteniyor.

Detay:

- Dağılımın normal olduğu biliniyor
- Aylık ortalama satış sayısı 80K, standart sapması 5k

- 90k'dan fazla satış yapma olasılığı nedir?

In [323…
```
from scipy.stats import norm
```

In [324…
```
#90'dan fazla olma olasılığı
1-norm.cdf(90,80,5)
```

Out[324]:
```
0.022750131948179209
```

In [325…
```
#70den fazla olma olasılığı
1-norm.cdf(70,80,5)
```

Out[325]:
```
0.9772498680518208
```

In [326…
```
#73'den az olması olasılığı
norm.cdf(73,80,5)
```

Out[326]:
```
0.08075665923377107
```

In [327…
```
#85-90 arası olasılığı
norm.cdf(90,80,5)-norm.cdf(85,80,5)
```

Out[327]:
```
0.13590512198327787
```

4. HİPOTEZ TESTLERİ

$H_0 : \mu = 50$ Hipotez
$H_1 : \mu \neq 50$ Alternatif, Hipotez

$H_0 : \mu <= 50, h1 : \mu > 50$
$H_1 : \mu >= 50, h1 : \mu < 50$

	$h_0 reddedilmedi$	$h_0 reddedildi$
$h_0 doğru$	Doğru Karar (1-\alpha) Güven Düzeyi	1.Tip Hata(\alpha)
h_{0} yanlış	2.Tip Hata(\beta)	Doğru Karar(1-\beta) Testin Gücü

h_{0} doğruykan ve biz bunu reddeymediysek bu **güven düzeyi** dir.

eğer h_{0} doğru iken biz bunu reddedersek bu \alpha hatasıdır.

eğer yanlış olan şeyi kabul ettiysek(h_{0} yanlışken, h_{0} reddedilmediyse) bu \beta hatasıdır

eğer h_{0} yanlışken biz bunun yanlış olduğunu düşünürsek bu **testin gücü**dür

!!! h_{0}'ı kabul etmek istatiksel açıdan doğru değildir. Çünkü h0 doğru iken ve onu reddettiğimizde yapacağımız hatayı biliyorken, ho'ı kabul ettiğimizde yapacağımız hatayı bilemeyiz.

4.1 p-Value

$p<0.05$ olmalıdır. Böylece ilgili h_{0} reddedilmiş olur. Eğer ortaya çıkan p-value değeri kabul edilebilir olan hata miktarı olan α'dan küçük ise, bu durumda h_{0} reddedilmiş olur. Fakat her zaman $p<0.05$, h_{0} 'ı reddetmiş sayılmayabilir. Dağılıma uygunluk testleri yapıldığında h_{0} hipotezi reddedilmek istenmez. Burada $h0$ örnek dağılımı ile teorik dağılım arasında fark yok der ve biz burada h_{0} ı reddetmek istemeyiz.

4.2 Hipotez Testi Adımları

- *1* - Hipotezlerin kurulması ve yönlerinin belirlenmesi
- *2* - Anlamlılık düzeyinin ve tablo değerinin belirlenmesi
- *3* - Test istatistiğinin belirlenmesi ve test istatistiğinin hesaplanması
- *4* - Hesaplanan test istatistiği ile alfa'ya karşılık gelen tablo değerinin karşılaştırılması
  ```
  Test istatistiği (Zh) > tablo değeri(Zt)
  veya -Zh < -Zt ise h0 reddedilmeli
  ```
- *5* - Yorum

4.3 Tek Örneklem T Testi

Popülasyon ortalaması ile varsayımsal bir değer arasında istatiksel olarak anlamlı bir farklılık olup olmadığını test etmek için kullanılan bir parametrik testtir.

- **Eğer anakütle standart sapması biliniyorsa z istatistiği kullanılır.**
- **Eğer anakütle standart sapması bilinmiyorsa ve n > 30 ise z istatistiği kullanılır.**
- **Eğer anakütle standart sapması bilinmiyor ve n < 30 ise t istatistiği kullanılır.**

!!!! n sayısı arttıkça t istatistiği z istatistiğine yaklaşır

4.3.1 UYGULAMA: ÜRÜN SATIN ALMA ADIM OPTİMİZASYONU

Problem:

- Sepete ürün ekleme işlemi sonrasında ödeme ekranında 5 adım vardır ve bu adımların birisi sorgulanmaktadır.

Detay:

- Her adımın 20'şer saniye olması hedefi var. 4. adım sorgulanıyor.
- Bu durumu test etmek için 100 örnek alınıyor.
- Örneğin standart sapması 5 saniyedir. Örnek ortalaması ise 19 saniyedir.

Adım 1: Hipotezlerin kurulması ve yönlerinin belirlenmesi

- $H_{0}:\mu = 20$
- $H_{1}:\mu \not= 20$

Adım2 : Anlamlılık düzeyinin ve tablo değerinin belirlenmesi

a = 0,05 --> a/2 = 0,025

Burada α değerini ikiye bölmemizin sebebi, alternatif hipotezimizin iki yönlü olmasından dolayıdır.

Ztablo olasılık değeri : 0,5 - 0,025 = 0,475

Ztablo kritik değer: -/+ 1,96

Adım 3: Test istatistiğinin belirlenmesi ve değerinin hesaplanması

Zhesap = $\frac {\hat{x} -\mu }{\sigma{(s)}/ \sqrt{n}}$ olduğuna göre;

$\hat{x} = 19$
$\mu = 20$
$\sigma{(s)} = 5$
$n = 100$ değerlerini yerine koyduğumuzda cevabımız = $-2,00$ olur

Adım 4: Ztablo ve Zhesap değerlerinin karşılaştırılması

$-Z_{hesap} = -2,00 < -Z_{tablo} = -1,196$ olduğundan $h0$ reddedilmeli

Eğer bu durum dışında olsaydı h_{0} hipotezini reddetmek için yeterli kanıt bulunamamıştır diyecektik. Yani istatiksel olarak anlamlı bir sonuç elde edilmeyecekti

Adım 5: Yorum

Adım 4'te geçirilen sürenin 20 sn olduğunu iddaa eden h_{0} reddedilmiştir. Buna göre kullanıcılar istatiksel olarak %95 güvenilirlik ile 4.adımda 20 saniyeden fazla süre geçirmektedir.

Not: Tek örneklem T testinde normallik varsayımı incelenmelidir. Normallik varsayımı sağlanmıyorsa non-parametric T testi yapılmalıdır.

4.3.2 UYGULAMA: WEB SİTESİNDE GEÇİRİLEN SÜRENİN TESTİ

Problem:

- Bir web sitesinde geçirilen ortalama sürenin 170s olduğu iddaa ediliyor.

Detaylar:

- Yazılımlardan elde edilen bilgiler ışığında bir web sitesinde geçirilen sürenin bilgisi vardır.
- Bu veriler incelendiğinde bir yönetici bu değerlerin böyle olmadığına yönelik düşünceler taşıyor ve bu durumu test etmek istiyor. Bu yüzden bu durumu test etmek istiyor.

In [9]:
```
import numpy as np
sureler = np.random.randint(17,251,36)
```

In [10]:
```
import scipy.stats as stats
```

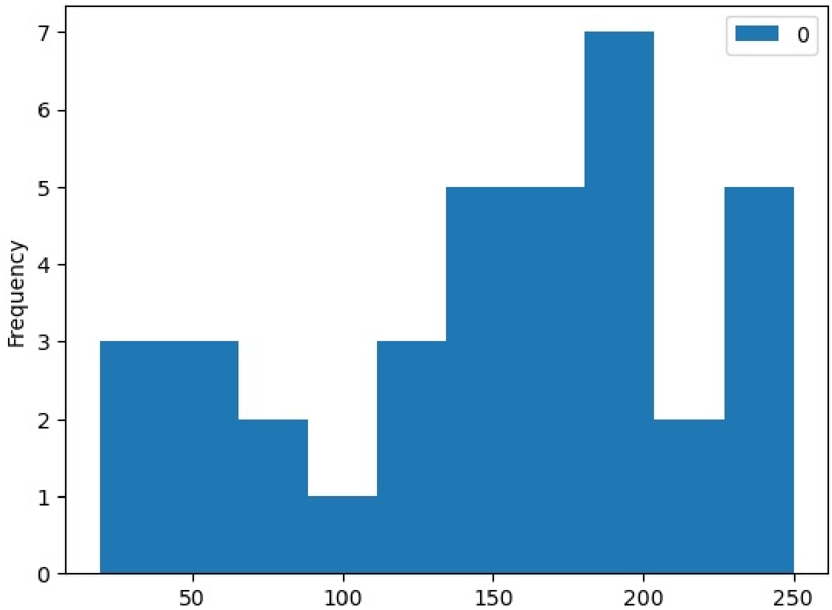
In [11]:
```
stats.describe(sureler)
```

Out[11]:
```
DescribeResult(nobs=36, minmax=(19, 250), mean=151.75, variance=4245.85, skewness=-0.41760527158009214, kurtosi
s=-0.8327230266919203)
```

Görüyoruz ki bizim örneğimizde ortalama:151 iken bize verilen değer: 170. 151<170 olduğundan hipotez testi uygulamaya gerek var mı? sorusu akıla gelebilir. Peki ya 151 şans eseri ortaya çıktıysa?

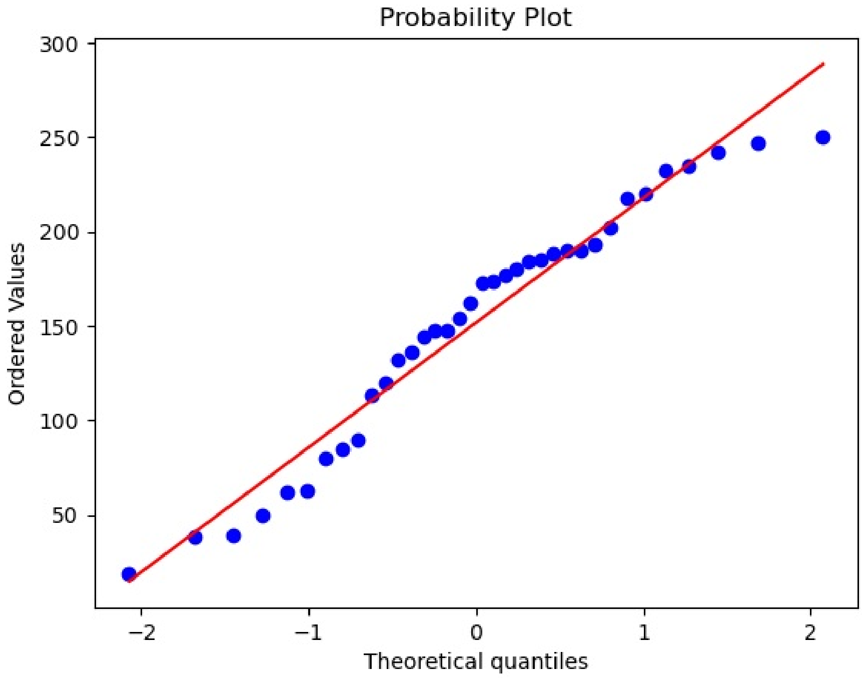
4.4 Tek örneklem T Testi Varsayım Kontrolü

In [12]:
```
#histogram grafiği
import pandas as pd
pd.DataFrame(sureler).plot.hist();
```



Grafikten dağılım normal olduğu gözlemleniyor. Bir de **pylab** kütüphanesinden değerlendirelim

In [15]:
```
##qqplot
import pylab
stats.probplot(sureler, dist="norm",plot = pylab)
pylab.show()
```



Sol taraftaki değer örnek dağılım, alt tarafta yer alan değer ise teorik dağılımı gösterir. Böylece biz ikisi arasındaki değerlendirmeyi sağlar. Normal dağılımda biz; verilerin kırmızı doğru etrafında konumlanmasını bekleriz.

In [16]:
```
#Shapiro-Wilks Testi
```

h0 : örnek dağılım ile teorik normal dağılım arasında istatistiksel olarak anlamlı bir farklılık yoktur.-- h1 : Fark vardır

In [17]:
```
from scipy.stats import shapiro
shapiro(sureler)
```

Out[17]:
```
ShapiroResult(statistic=0.9487962126731873, pvalue=0.09583521634340286)
```

p-value değeri 0,05'ten küçük olmadığını gözlemliyoruz. Bu durumuda h0 '**ı reddedemeyiz. Bu durumda örnek dağılım ile teorik normal dağılım arasında istatiksel olarak anlamlı bir farklılık yoktur.**

In [20]:
```
print("T Hesap İstatistiği: " +str(shapiro(sureler)[0]))
print("Hesaplanan p-value: " +str(shapiro(sureler)[1]))
```

```
T Hesap İstatistiği: 0.9487962126731873
Hesaplanan p-value: 0.09583521634340286
```

4.4.1 Hipotezin Uygulanması

In [22]:
```
stats.ttest_1samp(sureler, popmean=170)
```

Out[22]:
```
Ttest_1sampResult(statistic=-1.6804739926780656, pvalue=0.10177034369535037)
```

p-value 0.05'ten küçük olmadığı için h0 reddedilemez

4.4.2 Non-Parametric Tek Örneklem Testi

In [23]:
```
from statsmodels.stats.descriptivestats import sign_test
sign_test(sureler,170)
```

Out[23]:
```
(0.0, 1.0)
```

Merkezi eğilime göre 170s test edilir. Yukarıdaki parametrik testten zaten dağılımın normal olduğunu görmüştük. Burada amacımız, nonparametric testi göstermekti. Eğer yukarıda Normallik varsayımı sağlanmasaydı bu durumda non-parametric testi kullanmış olacaktık.

4.5 Tek Örneklem Oran Testi

Oransal bir ifade test edilmek istenildiğinde kullanılır.

H_{0}: p = p_{0}
H_{1}: p \not = p_{0}

H_{0}: p < p_{0}
H_{1}: p > = p_{0}

H_{0}: p > p_{0}
H_{1}: p < = p_{0}

Z = \hat{p} - p_{0} \sqrt{p_{0}*(1-p_{0})/n}

\hat{p} = örnek üzerinden elde ettiğimiz değer

p0 = sınamak istediğimiz değer

n>30 olmak zorundadır.

4.5.1 UYGULAMA: DÖNÜŞÜM ORANI

Problem:

- Bir yazılım ile bir mecrada reklam verilmiş ve bu reklama ilişkin yazılım tarafından 0,125 dönüşüm oranı(reklamdan gelen kullanıcı) elde edildiği farkedilmiş. Fakat bu durumu kontrol edilmek isteniyor. Çünkü bu yüksek bir oran ve gelirler incelendiğinde örtüşmüyor.

Detaylar:

- 500 kişi dış mecrada tıklamış ve 40 kişi geri dönüş yapmış.
- Örnek üzerinden elde edilen dönüşüm oranı: 40/500 = 0,08

HO : p = 0,125

H1 : p \not = 0,125

In [25]:
```
from statsmodels.stats.proportion import proportions_ztest
```

In [31]:
```
basari_sayisi = 40
gozlem_sayisi = 500
p = 0.125
ztest = proportions_ztest(basari_sayisi, gozlem_sayisi,p)
print("Test İstatistiği: "  +str(ztest[0]))
print("p-value: "  +str(ztest[1]))
```

```
Test İstatistiği: -3.7090151628513017
p-value: 0.0002080669689845979
```

p-value değeri 0,05'ten küçük olduğu için H0 reddedilir. 0,125 değeri yanlıştır demek istiyoruz.

5. BAĞIMSIZ İKİ ÖRNEKLEM T TESTİ

İki grup ortalaması arasında karşılaştırma yapılmak istenildiğinde kullanılır.

Eğer örnek sayıları aynı, varyanslar homojen ise:

t =\frac{{\hat{x_{1}}-\hat{x_{2}}}}{\sqrt[S_{p}]\frac{2}{n}}

S_{p} = \sqrt\frac{s^{2}x_{1}+s^{2}x_{2}}{2}

Eğer örnek sayıları farkı, varyanslar homojen ise:

t =\frac{{\hat{x_{1}}-\hat{x_{2}}}}{\sqrt[S_{p}]\frac{n_{1}+n_{2}}{n_{1}*n_{2}}}

S_{p} = \sqrt\frac{(n_{1}-1)s^{2}x_{1}+(n_{2}-1)s^{2}x_{2}}{n_{1}+n_{2}-2}

Eğer örnek sayıları farklı, varyanslar homojen değilse:

t =\frac{{\hat{x_{1}}-\hat{x_{2}}}}{S_{\hat{\Delta}}}

S_{\hat{\Delta}} = \sqrt\frac{s_{1}^{2}n_{2}+s_{2}^{2}n_{1}}{n_{1}n_{2}}

5.1 Varsayımlar

- Normallik Varsayımı
- Varyans Homojenliği Varsayımı

Varyans Homojenliği: Grupların varyanslarının benzerliği demektir.

5.2 UYGULAMA: ML MODELİNİN BAŞARI TESTİ (A/B TESTİ)

Problem:

Problem:

- Bir ML projesine yatırım yapılmış. Ürettiği tahminler neticesinde oluşan gelir ile eski sistemin ürettiği gelirler karşılaştırılıp anlamlı farklılık olup olmadığı test edilmek isteniyor.

Detaylar:

- Model geliştirilmiş ve web sitesine entegre edilmiş.
- Site kullanıcıları belirli bir kurala göre ikiye bölünmüş olsun.
- A grubu eski B grubu yeni sistem.

- Gelir anlamında anlamlı bir iş yapılıp yapılmadığı test edilmek isteniyor.

5.2.1 Hipotez

H_{0}: \mu_{1} = \mu_{2}
H_{1}: \mu_{1} \not= \mu_{2}

In [100…
```
#VERI TIPI I
```

In [101…
```
A = pd.DataFrame([30,27,21,27,29,30,20,20,27,32,35,22,24,23,25,27,23,27,23,25,21,18,24,26,33,26,27,28,19,25])
B = pd.DataFrame([37,39,31,31,34,38,30,36,29,28,38,28,37,37,30,32,31,31,27,32,33,33,33,31,32,33,26,32,33,29])
```

In [102…
```
A_B = pd.concat([A,B], axis=1)
A_B.columns = ["A","B"]
A_B.head(10)
```

Out[102]:

	A	B
0	30	37
1	27	39
2	21	31
3	27	31
4	29	34
5	30	38
6	20	30
7	20	36
8	27	29
9	32	28

Not: Normal iş yaşantısında proje kapsamında ya veritabanından veriler gelir ya da verilerin yer aldığı csv,excel dosyası gelecektir. Bu dosyaların içerisinde test edilmek istenen iki grubun değerleri yer alırBu veri fonksiyona olduğu gibi gönderildiğinde bir hata ile karşılaşılır. Çünkü ilgili fonksiyonları yazan kişiler istatistik literatüründe olduğu şekliyle yazılmış olmasını sağladıklarından dolayı bize gerçek hayatta verinin geldiği şekli bu testi yapmaya izin vermeyecek.

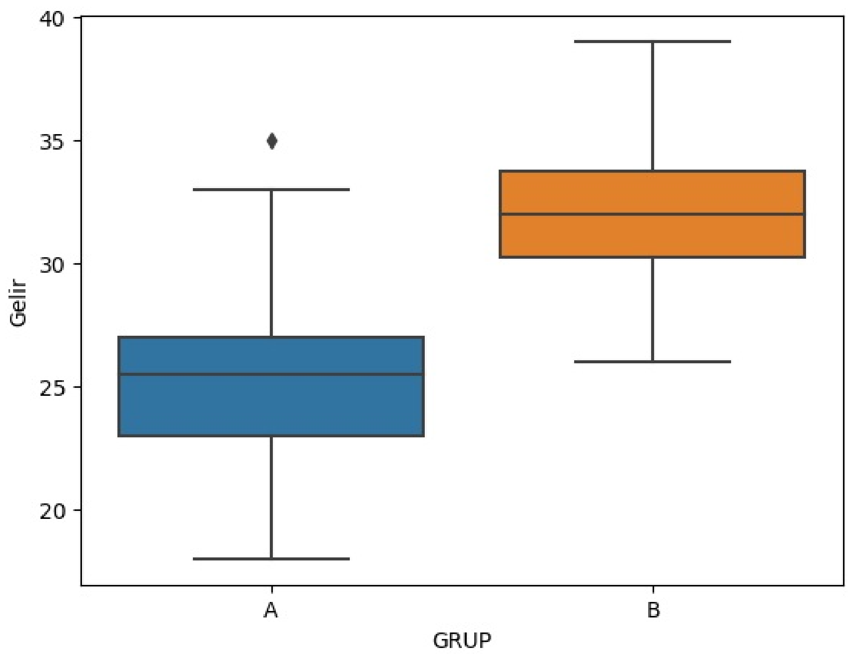
In [103…
```
# VERI TIPI II
```

In [104…
```
#verinin düzenlenmesinin en zor şeklidir
A = pd.DataFrame([30,27,21,27,29,30,20,20,27,32,35,22,24,23,25,27,23,27,23,25,21,18,24,26,33,26,27,28,19,25])
B = pd.DataFrame([37,39,31,31,34,38,30,36,29,28,38,28,37,37,30,32,31,31,27,32,33,33,33,31,32,33,26,32,33,29])

#A ve A'nın grubu:
group_A = np.arange(len(A))
group_A = pd.DataFrame(group_A)
group_A[:] = "A"
A= pd.concat([A, group_A], axis = 1)

#B ve B'nın grubu:
group_B = np.arange(len(B))
group_B = pd.DataFrame(group_B)
group_B[:] = "B"
B= pd.concat([B, group_B], axis = 1)

#Tüm veriler
AB = pd.concat([A,B])
AB.columns = ["Gelir","GRUP"]
print(AB.head())
print(AB.tail())
```

```
   Gelir GRUP
0     30    A
1     27    A
2     21    A
3     27    A
4     29    A
    Gelir GRUP
25     33    B
26     26    B
27     32    B
28     33    B
29     29    B
```

In [105…

```
import seaborn as sns
sns.boxplot(x = "GRUP",y ="Gelir",data =AB);
```



B grubu verileri grafiksel olarak üstte yer alıyor. Fakat bunun şans eseri olup olmadığını bilmiyoruz.

5.2.2 Varsayım Kontrolü

In [92]:

```
#normallik varsayımı
from scipy.stats import shapiro
print("A_B veriseti A değişkeni için Hesaplanan p-value: " +str(shapiro(A_B.A)[1]))
print("A_B veriseti B değişkeni için Hesaplanan p-value: " +str(shapiro(A_B.B)[1]))
```

```
A_B veriseti A değişkeni için Hesaplanan p-value: 0.7962851524353027
A_B veriseti B değişkeni için Hesaplanan p-value: 0.24584470689296722
```

AB veriseti içerisinden A değişkeni çekilip normallik testi yapıldı. p-value değeri $H{0} için reddedilemez.
 A_B veriseti içerisinden B değişkeni çekilip normallik testi yapıldı. p-value değeri H_{0}$ için reddedilemez.
İki değer için de normallik varsayımı başarı ile sağlanmıştır.

In [99]:

```
#varyans homojenliği varsayımı--#levene testi uygulanır
print("A_B veriseti A ve B değişkenleri varyans homojenliği için hesaplanan Levene Test İstatistiği: "
      +str(stats.levene(A_B.A,A_B.B)[0]))
print("A_B veriseti A ve B değişkenleri varyans homojenliği için hesaplanan p-value: " +str(stats.levene(A_B.A,…
```

```
A_B veriseti A ve B değişkenleri varyans homojenliği için hesaplanan Levene Test İstatistiği: 1.110180275715800
4
A_B veriseti A ve B değişkenleri varyans homojenliği için hesaplanan p-value: 0.2964124900636569
```

p-value değeri H_{0}ın reddedilemeyeceğini söylüyor. Bu da varyans homojenliği varsayımının sağlandığı anlamına gelmektedir.

5.2.3 Hipotez T Testi

In [110…

```
print("A_B veriseti için T Testi P-value değeri: " +str(stats.ttest_ind(A_B["A"],A_B["B"], equal_var=True)[1]))
```

```
A_B veriseti için T Testi P-value değeri: 2.6233215605475075e-09
```

Sonuçlar incelendiğinde p-value değeri < 0,05 gözlemlenir. Farklılık olmadığını iddaa eden H_{0} hipotezi reddedilir. Yani anakitle ortalamaları birbirine eşit değildir. Yani eski sistem ile yeni sistem arasında istatiksel olarak anlamlı bir fark vardır ve bu fark yeni sistemin **lehinedir**. Ortalamalarını incelediğimizde B grubunun ortalaması daha yüksekti.

5.3 Nonparametrik Bağımsızlık İki Örneklem Testi

Varsayalım ki önceki bölümde gerçekleştirilmiş olduğumuz iki tane varsayım testinin sonuçları negatif oldu. Normallik ve varyans homojenliği varsayımları sağlanmıyor. Bu durumda kullanılacak testtir. Bu testin adı **Mann Whitney U** testidir.

In [111…
```
stats.mannwhitneyu(A_B["A"],A_B["B"])
```
Out[111]:
```
MannwhitneyuResult(statistic=89.5, pvalue=9.557950378612535e-08)
```

In [116…
```
test, pvalue = stats.mannwhitneyu(A_B["A"],A_B["B"])
print("Test İstatistiği = %.4f, P-value = %.4f" %(test,pvalue))
```
```
Test İstatistiği = 89.5000, P-value = 0.0000
```

p- value <0,05 olduğundan Nonparametric olarak da *fark vardır yani* H_{0} hipotezi reddedilmiştir.

6. BAĞIMLI İKİ ÖRNEKLEM T TESTİ

Bağımlı iki grup ortalaması arasında karşılaştırma yapılmak istenildiğinde kullanılır. Aynı kitleye iki farklı uygulama yapıldığında ve bunun sonuçları incelendiğinde bağımlılık durumu söz konusudur.

H_{0}: \mu_{ö} = \mu_{s}
H_{1}: \mu_{ö} \not= \mu_{s}

H_{0}: \mu_{ö} < \mu_{s}
H_{1}: \mu_{ö} >= \mu_{s}

H_{0}: \mu_{ö} > \mu_{s}
H_{1}: \mu_{ö} <= \mu_{s}

t =\frac{{\hat{x_{D}}-{m_{ö}}}}{\frac{S_{D}}{\sqrt{n}}}
Varsayımlar:

- Normallik Varsayımı
- Varyans Homojenliği Varsayımı

6.1 UYGULAMA: ŞİRKET İÇİ EĞİTİM PERFORMANS ETKİSİ ÖLÇÜMÜ

6.1.1 Hipotez

Problem:

- Belirli uğraşlar sonucunda alınan eğitimin katma değer sağlayıp sağlamadığı ölçülmek isteniyor

Detaylar:

- Bir departman bir konuda eğitim talep ediyor.
- Gerekli/Gereksiz değerlendirmeleri neticesinde eğitim alınıyor.
- Eğitimden önce ve sonra olacak şekilde gerekli ölçümler yapılıyor.
- Eğitim sonrasında eğitimin sağladığı katma değer test edilmek isteniyor.

H_{0}: \mu_{ö} = \mu_{s}
H_{1}: \mu_{ö} \not= \mu_{s}

In [231…
```
once = pd.DataFrame([123,119,119,116,123,123,121,120,121,120,117,118,121,121,123,119,119])
sonra = pd.DataFrame([118,127,122,132,129,123,129,132,128,130,128,138,140,130,133,127,155])
```

In [232…
```
#BIRINCI VERİ SETİ
ayrik = pd.concat([once,sonra], axis = 1)
ayrik.columns = ["ONCE","SONRA"]
print("'AYRIK' Veri Seti :\n \n", ayrik.head(), "\n\n")
```
```
'AYRIK' Veri Seti :

    ONCE  SONRA
0   123    118
1   119    127
2   119    122
3   116    132
4   123    129
```

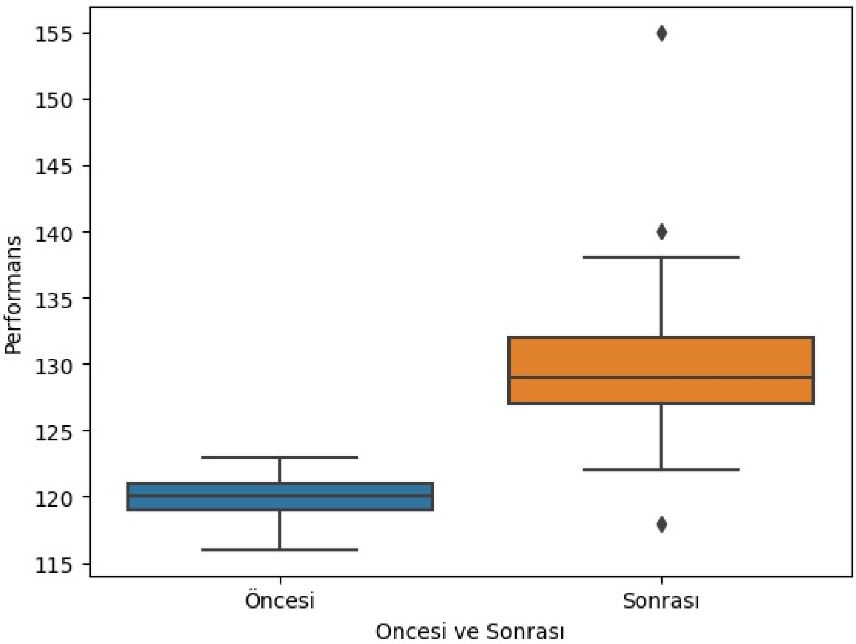
In [233…
```
#IKINCI VERİ SETİ
#Öncesi Flat Tag
g_once = np.arange(len(once))
g_once = pd.DataFrame(g_once)
```

```
g_once[:]="Öncesi"
#Flag ve Öncesi değerlerini bir araya getirme
A = pd.concat([once,g_once], axis = 1)

#Sonrasi Flag Tag
g_sonra = np.arange(len(sonra))
g_sonra = pd.DataFrame(g_sonra)
g_sonra[:]="Sonrası"
#Flag ve Öncesi değerlerini bir araya getirme
B = pd.concat([sonra,g_sonra], axis = 1)

#Tüm veriyi bir araya getirme
Birlikte = pd.concat([A,B])

#İsimlendirme
Birlikte.columns = ["Performans","Oncesi ve Sonrası"]
print("'Birlikte' veriseti: \n\n", Birlikte,"\n" )
```

```
'Birlikte' veriseti: 

     Performans Oncesi ve Sonrası
0           123            Öncesi
1           119            Öncesi
2           119            Öncesi
3           116            Öncesi
4           123            Öncesi
5           123            Öncesi
6           121            Öncesi
7           120            Öncesi
8           121            Öncesi
9           120            Öncesi
10          117            Öncesi
11          118            Öncesi
12          121            Öncesi
13          121            Öncesi
14          123            Öncesi
15          119            Öncesi
16          119            Öncesi
0           118           Sonrası
1           127           Sonrası
2           122           Sonrası
3           132           Sonrası
4           129           Sonrası
5           123           Sonrası
6           129           Sonrası
7           132           Sonrası
8           128           Sonrası
9           130           Sonrası
10          128           Sonrası
11          138           Sonrası
12          140           Sonrası
13          130           Sonrası
14          133           Sonrası
15          127           Sonrası
16          155           Sonrası
```

In [234…

```
import seaborn as sns
sns.boxplot(x ="Oncesi ve Sonrası",y = "Performans", data = Birlikte);
```



6.1.2 Varsayım kontrol

In [235…

```
#normallik varsayımı
from scipy.stats import shapiro
print("P-Value Öncesi: " + str(shapiro(ayrik.ONCE)[1]))
print("P-Value Öncesi: " + str(shapiro(ayrik.SONRA)[1]))
```

```
P-Value Öncesi: 0.22514021396636963
P-Value Öncesi: 0.019879667088389397
```

$p > 0,05$ Bu durumda H_{0} hipotezi reddedilemez. Dağılım normaldir

In [236…

```
#varyans homojenliği
print("P-Value Öncesi: " + str(stats.levene(ayrik.ONCE,ayrik.SONRA)[1]))
```

```
P-Value Öncesi: 0.03782794304836367
```

$p <0,05$ olduğundan H_{0} hipotezini reddetmek durumundayız. Varyanslığın homojenliği durumu sağlanmamaktadır.

Bunun için veriseti üzerinde aykırılıklar varsa belki düzenleme işlemleri yapılarak bu varsayımlar tekrar incelenebilir. Bağımlı T testinde varyans homojenliği söz konusu değilse **bu bir miktar göz ardı edilebilir.**

6.1.3 T TESTİ

In [244…

```
print("Test İstatistiği: " + str(stats.ttest_rel(ayrik.ONCE,ayrik.SONRA)[0]))
print("P-Value: " + str(stats.ttest_rel(ayrik.ONCE,ayrik.SONRA)[1]))
```

```
Test İstatistiği: -4.777985483030638
P-Value: 0.00020534438694064702
```

$H_{0}: \mu_{ö} = \mu_{s}$
$H_{0}: \mu_{ö} \not= \mu_{s}$

$p<0,05$ olduğundan H_{0} hipotezi reddedilir. Yani eğitim işe yaramıştır denilir.

6.1.3 Non-Parametric İki Bağımlı Örneklem T TESTİ

In [245…

```
print("P-Value: " + str(stats.wilcoxon(ayrik.ONCE,ayrik.SONRA)[1]))
```

```
P-Value: 0.0006363373100722408
```

Varyans homojenliği varsayımları sağlanmadığı için nonparametrik t testine başvurulur. Wilcoxon testinin sonucuna göre de H_{0} hipotezi reddedilir. Öncesi ve sonrası arasında anlamlı bir farklılık vardır.

7. İKİ ÖRNEKLEM ORAN TESTİ

İki oran arasında karşılaştırma yapmak için kullanılır.

$H_{0}: p_{1} = p_{2}$
$H_{1}: p_{1} \not= p_{s}$

$H_{0}: p_{1} < p_{2}$
$H_{1}: p_{1} >= p_{2}$

$H_{0}: p_{1} > p_{2}$
$H_{1}: p_{1} <= p_{2}$

$$Z_{h} =\frac{(p_{1}-p_{2})}{\sqrt{p(1-p)((n_{2}+n_{1})/n_{1}*n_{2})}}$$

$n_{1} > 30, n_{2} > 30$ olmalıdır.

7.1 UYGULAMA: KULLANICI ARAYÜZÜ DENEYİ

Problem:

- Kırmızı Buton mu Yeşil Buton mu?

Detaylar:

- Yeşil Butonu Gören 1000 kişiden 300 kişi tıklamış
- Kırmızı butonu gören 1100 kişiden 250 kişi tıklamış

7.1.1 İki Örneklem Oran Testi

In [248…

```
from statsmodels.stats.proportion import proportions_ztest
```

In [246…
```
from statsmodels.stats.proportion import proportions_ztest
basari = np.array([300,250])
gozlem = np.array([1000,1100])
```

In [247…
```
print("P-Value: " + str(proportions_ztest(count = basari, nobs = gozlem)[1]))
```

```
P-Value: 0.0001532232957772221
```

H_{0} hipotezi $p\text{-value} < 0{,}05$ olduğundan reddedilmiştir. Yani iki butonun dönüşüm oranı birbirine eşit değildir. Bu farklılık yeşil butonun lehine olacak şekildedir.

8. VARYANS ANALİZİ

İkiden fazla grup olduğunda kullanılacak hipotez testi yaklaşımıdır. Varyans analizi oldukça geniş bir konudur, bu başlık sadece ikiden fazla grup olduğunda bu gruplar arasındaki farklılığı değerlendirmek adına ele alınacaktır.

$H_{0}: \mu_{1} = \mu_{2} = \mu_{3}$
$H_{1}: \mu_{1} \not= \mu_{2} \not= \mu_{3}$ (En az biri farklıdır)

$F_{s} = \frac{MS_{between}}{MS_{within}}$

$MS_{between}$ = gruplar arası ortalama hata
MS_{within} = grup içi ortalama hata

8.1 VARSAYIMLAR

- Gözlemlerin birbirinden bağımsız olması
- Normal Dağılım
- Varyans Homojenliği

8.2 UYGULAMA: ANASAYFA İÇERİK STRATEJİSİ BELİRLEME

Problem:

- Anasayfada geçirilen süre arttırılmak isteniyor.

Detaylar:

- Bir web sitesi başarı kriterleri; ziyaret süresi, hemen çıkış oranı vb
- Uzun zaman geçiren kullanıcıların reklamlara daha fazla tıkladığı ve markaya olan bağlılıkları bildiriliyor.
- Buna yönelik olarak benzer haberler farklı resimler ya da farklı formatlarda hazırlanarak oluşturulan test gruplarına gösteriliyor.

$h_{0}: \mu_{1} = \mu_{2} = \mu_{3}$ geçirilen süreler arasında bir farklılık yoktur.
h_{1}: fark vardır

In [250…
```
A = np.random.randint(25,35,20)
A = pd.DataFrame(A)

B = np.random.randint(28,38,20)
B = pd.DataFrame(B)

C = np.random.randint(30,40,20)
C = pd.DataFrame(C)
```

In [254…
```
dfs = [A,B,C]
ABC = pd.concat(dfs,axis = 1)
ABC.columns = ["GRUP A","GRUP B","GRUP C"]
```

8.2.1 Varsayımlar

In [266…
```
print("A grubu P-Value değeri:  " + str(shapiro(ABC["GRUP A"])[1]))
```

```
A grubu P-Value değeri:  0.05198747664690018
```

In [267…
```
print("B grubu P-Value değeri:  " + str(shapiro(ABC["GRUP B"])[1]))
```

```
B grubu P-Value değeri:  0.17187827825546265
```

In [268…
```
print("C grubu P-Value değeri:  " + str(shapiro(ABC["GRUP C"])[1]))
```

```
C grubu P-Value değeri:  0.1510951966047287
```

$p<0{,}05$ sağlanmadığından h_{0} hipotezi reddedilemiyor. Üç grup için de normallik varsayımı sağlanmaktadır.

In [269…
```
print("ABC VERİSETİ İÇİN  P-Value değeri:  " + str(stats.levene(ABC["GRUP A"],ABC["GRUP B"],ABC["GRUP C"])[1]))
```

```
ABC VERİSETİ İÇİN  P-Value değeri:  0.4685217335960813
```

$p<0,05$ sağlanmadığından h_{0} hipotezi reddedilemiyor. Üç grup için de varyans homojen varsayımı sağlanmaktadır.

8.2.2 F TESTİ

In [271…
```
from scipy.stats import f_oneway
print('{:.5f}'.format((f_oneway(ABC["GRUP A"],ABC["GRUP B"],ABC["GRUP C"])[1])))
```

```
0.00008
```

$p<0,05$ sağlandığından h_{0} hipotezi reddedilir. Üç grup için de f testi sağlanmamaktadır. Yani üç farklı anasayfada geçirilen süreler farklıdır.

In [272…
```
ABC.describe().T
```

Out[272]:

	count	mean	std	min	25%	50%	75%	max
GRUP A	20.0	30.20	2.894641	25.0	28.00	30.0	33.00	34.0
GRUP B	20.0	33.25	2.510504	29.0	31.00	33.5	35.25	37.0
GRUP C	20.0	34.30	3.079645	30.0	31.75	34.0	37.00	39.0

8.2.3 NonParametrik Hipotez Testi(Kruskal)

In [274…
```
from scipy.stats import kruskal
print('{:.5f}'.format((kruskal(ABC["GRUP A"],ABC["GRUP B"],ABC["GRUP C"])[1])))
```

```
0.00058
```

$p<0,05$ sağlandığından h_{0} hipotezi reddedilir. Üç grup için de f testi sağlanmamaktadır. Yani üç farklı anasayfada geçirilen süreler farklıdır.

9. KORELASYON ANALİZİ

Değişkenler arasındaki ilişki, bu ilişkinin yönü ve şiddeti ile ilgili bilgiler sağlayan istatiksel yöntemdir.

- Mükemmel Pozitif Korelasyon : 1
- Yüksek Pozitif Korelasyon : 0.9
- Düşük Pozitif Korelasyon : 0.5
- Korelasyon yok : 0
- Düşük Negatif Korelasyon : -0.5
- Yüksek Negatif Korelasyon : -0.9
- Mükemmel Negatif Korelasyon : -1

Reklam bütçesi arttıkça satışların artması; **pozitif korelasyon**
Araç km arttıkça fiyatının düşmesi; **negatif korelasyon**

korelasyon anlamlılığın testi:
$H_{0}: p = 0$ değişkenler arasında korelasyon yoktur.
$H_{1}: p \not= 0$ değişkenler arasında korelasyon ilişki vardır.

$$r_{xy} = \frac{\sum x_{i}y_{i}-n\hat{x}\hat{y}}{\sqrt{(\sum x_{i}^{2}-n\hat{x}^{2})}\sqrt{(\sum y_{i}^{2}-n\hat{y}^{2})}}$$

$$t = r \sqrt{\frac{n-2}{1-r^{2}}}$$

9.1 UYGULAMA: BAHŞİŞ İLE ÖDENEN HESAP ARASINDAKİ İLİŞKİNİN İNCELENMESİ

In [275…
```
import seaborn as sns
tips = sns.load_dataset('tips')
df = tips.copy()
df.head()
```

Out[275]:

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4

In [276…
```
#total bill = bahşiş ve vergiler dahil
```

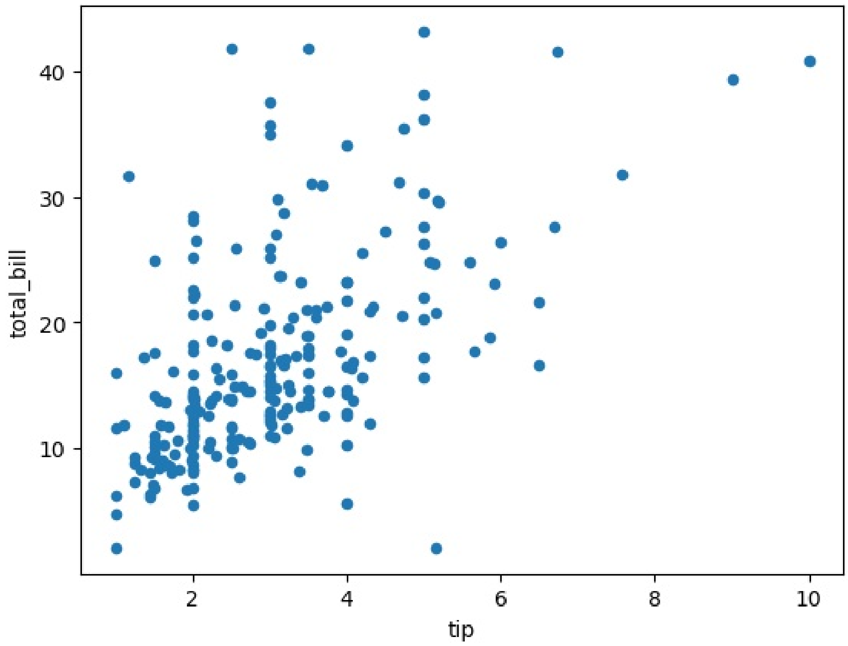
```
In [277… df["total_bill"] = df["total_bill"] - df["tip"] # toplam hesaptan bahşişi çıkarıyoruz.
```

```
In [278… df.head()
```

Out[278]:

	total_bill	tip	sex	smoker	day	time	size
0	15.98	1.01	Female	No	Sun	Dinner	2
1	8.68	1.66	Male	No	Sun	Dinner	3
2	17.51	3.50	Male	No	Sun	Dinner	3
3	20.37	3.31	Male	No	Sun	Dinner	2
4	20.98	3.61	Female	No	Sun	Dinner	4

```
In [279… df.plot.scatter("tip", "total_bill");
```



9.1.1 Varsayımlar

```
In [281… print("Bahşiş için P-Value değeri:  " + str(shapiro(df["tip"])[1]))
         print("Toplam Hesap için P-Value değeri:  " + str(shapiro(df["total_bill"])[1]))
```

```
Bahşiş için P-Value değeri:  8.200817629144819e-12
Toplam Hesap için P-Value değeri:  1.1060685700670092e-10
```

p<0,05 olduğu için h_{0} reddedilir. Yani fark vardır. Örnek dağılım ile teorik dağılım birbirine benzemiyor.

9.1.2 Hipotez Testi

```
In [282… #korelasyon katsayısı
         df["tip"].corr(df["total_bill"])
```

Out[282]: 0.5766634471096381

.corr() fonksiyonu öntanımlı olarak Pearson Korelasyon katsayısını verir. Ancak değişkenler için normallik varsa Pearson Korelasyon Katsayısı kullanılabilir.

```
In [284… df["tip"].corr(df["total_bill"],method = "spearman")
```

Out[284]: 0.593691939408997

Değişkenlerin arasında pozitif yönlü bir ilişki var. İlişkinin yönü orta şiddetli. Peki bu ilişki anlamlı mı?

9.1.2.1 Anlamlılık Testi

```
In [288… from scipy.stats.stats import pearsonr
         print("P-value: " +str(pearsonr(df["tip"],df["total_bill"])[1]))
```

```
P-value: 5.01829008494899e-23
C:\Users\tahat\AppData\Local\Temp\ipykernel_6324\1911111396.py:1: DeprecationWarning: Please use `pearsonr` fro
m the `scipy.stats` namespace, the `scipy.stats.stats` namespace is deprecated.
  from scipy.stats.stats import pearsonr
```

p<0,05 olduğundan değişkenler arasında anlamlı bir ilişki vardır diyebiliriz.

9.1.2.2 NonParametrik Hipotez Testi

In [293…
```
print("SpearmanR testine göre veriseti Korelasyon katsayısı değeri:" +str(stats.spearmanr(df["tip"],df["total_b
print("SpearmanR testine göre veriseti p-value değeri:" +str(stats.spearmanr(df["tip"],df["total_bill"])[1]))
```

```
SpearmanR testine göre veriseti Korelasyon katsayısı değeri:0.593691939408997
SpearmanR testine göre veriseti p-value değeri:1.2452285137560276e-24
```

p<0,05 olduğundan *değişkenler arası ilişki yoktur* diyen h_{0} hipotezi reddedilir. yani ilişki vardır.

In [294…
```
print("KendallTau testine göre veriseti Korelasyon katsayısı değeri:" +str(stats.kendalltau(df["tip"],df["total
print("KendallTau testine göre veriseti p-value değeri:" +str(stats.kendalltau(df["tip"],df["total_bill"])[1]))
```

```
KendallTau testine göre veriseti Korelasyon katsayısı değeri:0.4400790074919885
KendallTau testine göre veriseti p-value değeri:7.131027725873621e-24
```

p<0,05 olduğundan *değişkenler arası ilişki yoktur* diyen h_{0} hipotezi reddedilir. yani ilişki vardır.

Taha Talha Özcan

Business Owner| Jr.Data Scientist

04.03.2023

In [ ]:

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js