

# Veri Analizi

# Ömer Cengiz

**Github:** omercengiz

**Linkedin:** omercengiz96

**Twitter:** omerr.cengizz

# Bugün Ne Konuşacağız?

1. Günümüzün en önemli ve popüler kaynağı veri ile tanışacak, veri tiplerini görerek verileri anlamaya derin bir giriş yapacağız.
2. Veriyi ifade etmenin en iyi yöntemi olan grafik türlerini görerek veriyi grafiklerle nasıl ifade edeceğimizi öğreneceğiz.
3. Verileri anlamak ve kullanmak için önemli konulardan biri olan temel istatistik konusuna girecek ve verileri nasıl yorumlayabileceğimizi göreceğiz.
4. Veri Dağılımını anlamlandırabilmek ve anlayabilmek için hipotez testlerini öğreneceğiz.
5. Verilerde aykırı ve eksik değerleri tespit etmeyi öğrenecek ve bir ön işleme yöntemi olan veri temizleme operasyonunu göreceğiz.
6. Veri ön işleme ve analizi için Pandas kütüphanesinin sunmuş olduğu teknikleri ve fonksiyonları görerek, verimizi hem analize hem de makine öğrenmesi modelimize hazır hale getirmeyi öğreneceğiz.

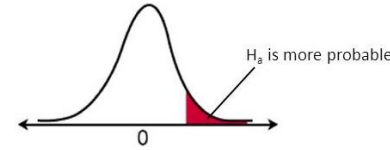
# Temel Hipotez Testleri

# Temel Hipotez Testleri

Hipotez testi süreci, bir örnek üzerinde bazı istatistiksel testler yaparak genel popülasyon veya veriler hakkında çıkarımlar yapmak ve bazı sonuçlar elde etmektir.

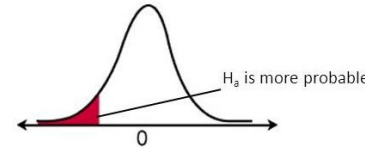
Bu çıkarımların yapılabilmesi için, hipotez testinde 2 temel varsayım kullanılır;

- **Null Hypothesis:** Parametre veya dağılım üzerinden gerçekleşen hipotez testinde her zaman kabul edilen gerçektir.
- **Alternative Hypothesis:** Null Hypothesis'e karşıt olarak oluşturulan hipotezdir. Null Hypothesis'in savunduğu önermenin tam tersini savunur.



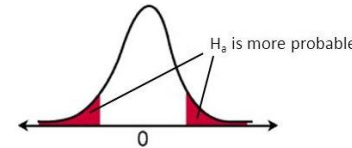
Right-tail test

$$H_a: \mu > \text{value}$$



Left-tail test

$$H_a: \mu < \text{value}$$



Two-tail test

$$H_a: \mu \neq \text{value}$$

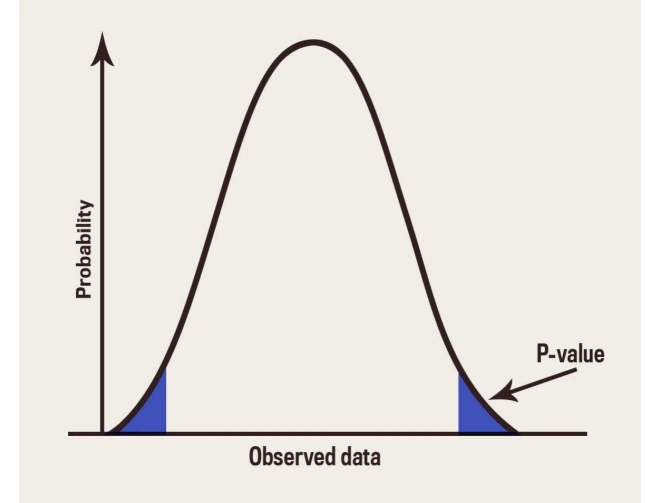
## p değeri

Bir araştırma örnekleminde değişkenler arasındaki ilişkinin, gerçekte bir ilişki yokken sadece şans eseri ortaya çıkmış olma olasılığıdır.

Bu bağlamda, Null hipotezi için kanıt veya anlamlılık düzeyidir (significance value) ve genel olarak 0.05 olarak seçilir ancak manuel olarak da verilebilir.

Küçük bir p değeri hipotezi reddetmek için güçlü kanıtların olduğu anlamına gelirken, büyük bir p değeri hipotezi reddetmek için zayıf kanıtların olduğu anlamına gelir.

İstatistiksel anlamlılık testi yaptığınızda (t-testi, ki-kare testi vb. gibi) p değerleri genellikle kullanılır. Bu testler hesaplanmış bir test istatistiği ve ilişkili p değerini döndürür.



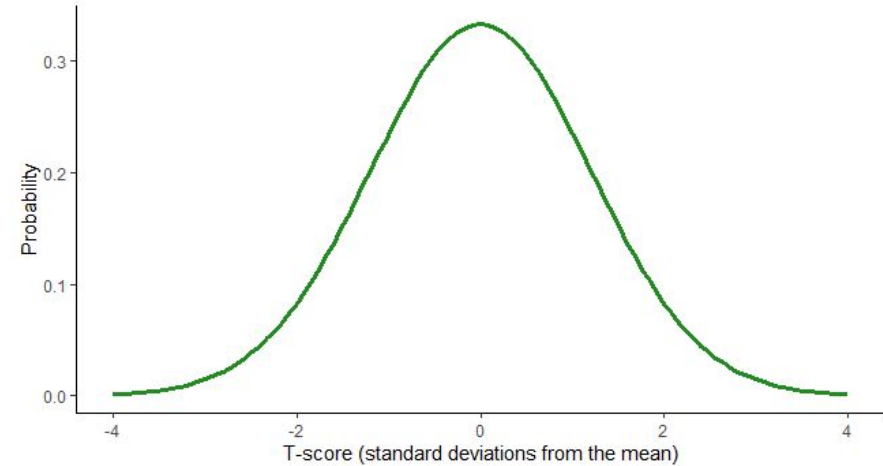
# T Testi

Popülasyon ortalaması ile varsayımsal bir değer arasında anlamlı bir farklılık olup olmadığı test etmek istendiğinde kullanılır.

Genellikle bir sürecin ilgili popülasyon üzerinde gerçekten bir etkisi olup olmadığını veya iki grubun birbirinden farklı olup olmadığını belirlemek için hipotez testlerinde kullanılır.

İki kümenin her birinden bir örnek alır ve iki ortalamanın eşit olduğuna dair null hipotez varsayarak problem ifadesini oluşturur.

Test sonuçları, p-değeri ile kıyaslanır ve bu kıyas sonucuna göre null testi kabul edilir veya reddedilir.



$$t = \frac{\bar{x}_1 - \bar{x}_2}{\sqrt{s^2 \left( \frac{1}{n_1} + \frac{1}{n_2} \right)}}$$

# Z Testi

Z testi, varyanslar bilindiğinde ve seçilen örneklem büyüklüğü büyük olduğunda popülasyon ve örneklemin ortalamasının farklı olup olmadığını belirlemek için kullanılan istatistiksel bir testtir.

Test istatistiğinin normal bir dağılıma sahip olduğu varsayılır ve doğru bir z testinin gerçekleştirilebilmesi için standart sapma gibi dağılım parametrelerinin bilinmesi gerekir.

*Örneklem büyüklüğü 30'dan küçükse ve popülasyon varyansı bilinmiyorsa, t-testi kullanılmalıdır.*

Score

650

730

510

670

480

800

690

530

590

620

710

670

640

780

650

490

800

600

510

700

$$\begin{aligned} \text{z score} &= \frac{\bar{x} - \mu}{\sigma / \sqrt{n}} \\ &= \frac{641 - 600}{100 / \sqrt{20}} \\ &= 1.8336 \end{aligned}$$

$$\text{p value} = .033357$$

$$\text{Critical Value} = 1.645$$

$$\text{Z score} > \text{Critical Value}$$

$$\text{P value} < 0.05$$



$$H_0 : \mu \leq 600$$

$$H_1 : \mu > 600$$

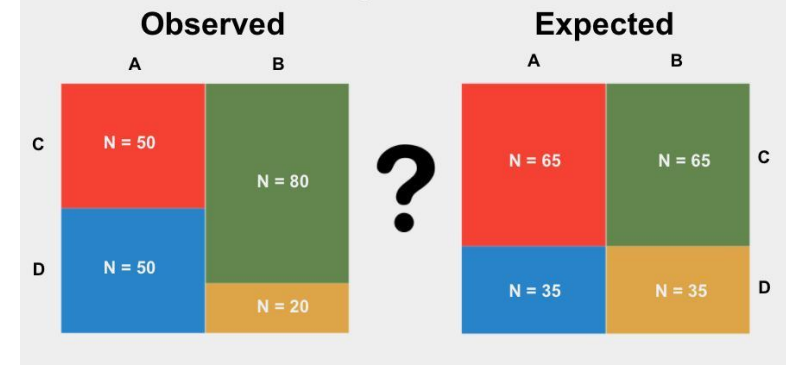




# Ki-kare (Chi-Square) Testi

Ki-kare testi, iki veya daha fazla değişkenin birbirinden bağımsız olup olmadığının araştırılmasında kullanılır. Bu test bağımsızlığı ölçse de değişkenler arasındaki ilişki hakkında yeterli bilgi sağlamaz.

- Ki-kare testi frekanslar üzerinden çalışıp çapraz tablolar oluşturacağından en önemli koşul verilerin kategorik olmasıdır. Buna ek olarak gruplar birbirinden bağımsız olmalı, bir gözlem birden fazla grup/kategori/değişken altında yer almamalıdır
- Ki-kare testi örneklem büyüklüğüne duyarlıdır. Örnek boyutu arttıkça, mutlak farklar beklenen değerin daha küçük bir oranı haline gelir

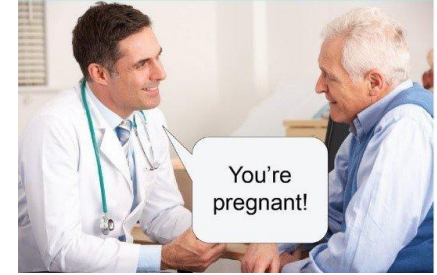


# Hipotez Testinde Hatalar

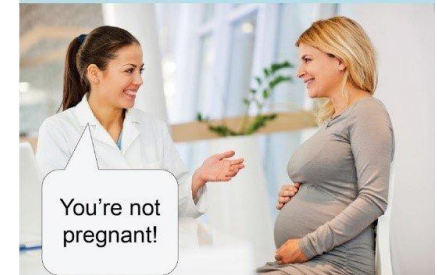
Yüksek verilerde, veri boyutunun fazla olması ve kaynakların tüm veriyi işlemek için yeterli olmaması nedeniyle, hipotez testi tüm popülasyon yerine bir veri örneği üzerinde yapılır. Tüm popülasyon yerine sadece örnek veriler üzerinde yapılan çıkarımlar nedeniyle hipotez testi, iki tür hataya yol açabilir;

1. **Tip I Hata:** Gerçekte bulgu iddiayı desteklemiyor olsa da, destekliyormuş gibi ileri sürme durumunda oluşur. Örneğin hamile olması imkansız olan bir erkeğin "hamile" olduğunu iddia etmek.
2. **Tip II Hata:** Gerçekte bulgu iddiayı destekliyor olsa da, desteklemiyormuş gibi ileri sürme durumunda oluşur. Örneğin, gerçekten de hamile bir kadının hamile olmadığını iddia etmek.

Type I Error



Type II Error



# Hipotez Testinde Hatalar

Type I and Type II Error		
Null hypothesis is ...	True	False
Rejected	Type I error False positive Probability = $\alpha$	Correct decision True positive Probability = $1 - \beta$
Not rejected	Correct decision True negative Probability = $1 - \alpha$	Type II error False negative Probability = $\beta$

# Veri Temizleme

Aykırı, Eksik, Duplike Veriler ve Bu Verilerin Tespiti

Null vs NaN

Boyut İndirgeme

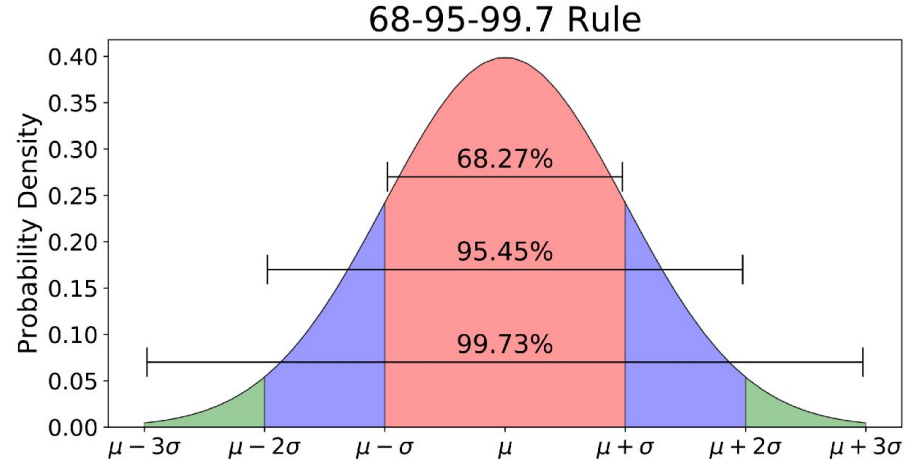
# Aykırı Değerlerin Tespiti

# 68-95-99.7 Kuralı ve 3 Sigma

68-95-99.7 Kuralına göre normal bir dağılımda ortalamadan;

- $+1(\sigma)$  ve  $-1(\sigma)$  uzaklıktaki değerler toplam popülasyonun %68.27'sini
- $+2(\sigma)$  ve  $-2(\sigma)$  uzaklıktaki değerler toplam popülasyonun %95.45'ini
- $+3(\sigma)$  ve  $-3(\sigma)$  uzaklıktaki değerler toplam popülasyonun %99.7'sini oluşturur

3 Sigma, verilerin bir ortalamadan üç standart sapma içinde olduğu istatistiksel bir hesaplama.



# Aykırı Değerlerin Tespiti

- En basit ifadeyle, aykırı değerler diğer veri noktalarından oldukça uzakta bulunan veya veri çerçevesinde görülen modelle tutarlı olmayan veri noktalarıdır
- Aykırı değerler yazım hatalarından veya ölçüm hatalarından kaynaklanabilir ya da sadece doğal bir aykırı değer de olabilir
- Aykırı değerleri saptamak için kullanılan en yaygın yöntem görselleştirmedir ve yaygın olarak kabul edilen "**Aykırı değerler ortalamadan negatif veya pozitif yönde üç veya daha fazla standart sapma uzaklıktadır (diğer adıyla 3 sigma)**" ilkesi, bize bir veri noktasının aykırı değer olduğuna dair ipuçları verir

***Not:** “Data Binning” de minör gözlem hatalarının etkilerini azaltmak için kullanılan bir “ön işleme” tekniği olarak sayılmaktadır. Bu method ve aykırı değerlerin tespiti aynı amaçla kullanılabilir.*

# Aykırı Değer Tespitinin Kahramanı: Z-Skoru

Z-skoru, bir veri noktasının ortalamadan kaç standart sapma uzaklıkta olduğunu gösteren istatistiksel bir ölçümdür.

3 Sigma kuralına göre eğer bir veri noktası ortalamadan en az 3 standart sapma uzaklıktaysa bu gözlem aykırı bir gözlemdir.

Her bir veri noktasının farklı bir Z-score'u vardır, ancak veri seti yalnızca tek bir ortalamaya ve tek bir standart sapma değerine sahiptir.

## Z Score Formula



$$Z = \frac{X - \mu}{\sigma}$$



$X$  = data

$\mu$  = mean

$\sigma$  = standard deviation



# Eksik Değerlerin Ele Alınması

# Eksik Değerleri Anlamak: NULL vs NaN

**NULL** değerler **değer yok** veya **hiçbir şey** anlamına gelir, boş bir string veya **sıfır değildir**. Python'da None olarak ifade edilir.

Bellekte null bir değer için alan ayrılmaz.

**NaN**, **Not a Number** anlamına gelir, genellikle mantıklı olmayan matematiksel bir işlemin sonucudur.

*Örneğin  $0.0/0.0$  işleminin sonucu NaN bir değerdir.*

Bellekte NaN bir değer için alan ayrılır.

Non-zero value



null



0



undefined



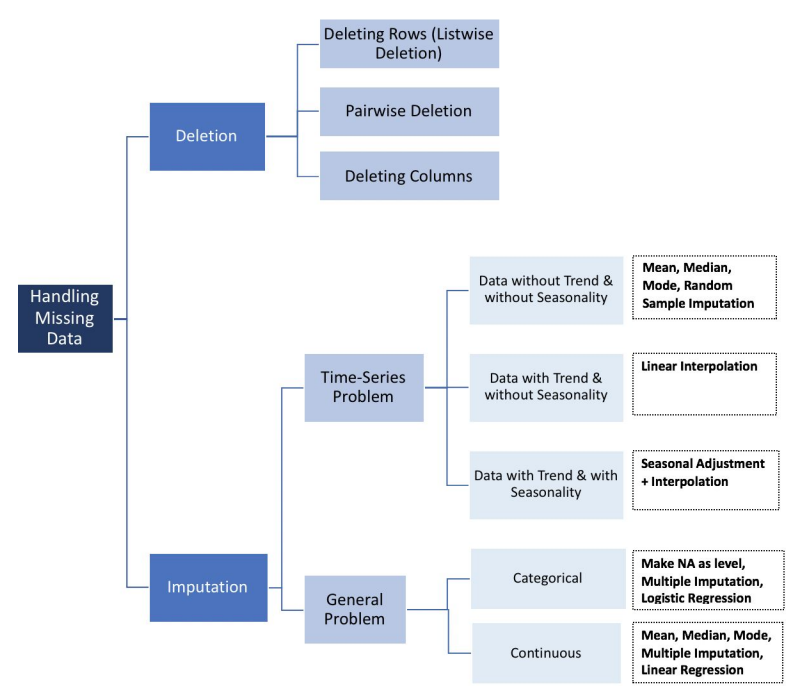
# Eksik Değerler Probleminin Üstesinden Gelmek

Eksik veri, gözlemdaki herhangi bir değişken için bir veri değerinin None veya NaN olmasıdır.

Eksik veriler ile işlem yaparken uygulanabilecek iki yaklaşım vardır;

1. Eksik Verileri Silmek (drop)
2. Eksik Verileri Doldurmak (fill / imputation)

Veride NULL veya NaN olarak kabul edilebilecek eksik değerleri ön işlemek için Pandas'ın bize sağladığı özel metotlar kullanılır.



# Eksik Değerler için Kullanılan Özel Pandas Fonksiyonları

# df.isna()

Bir DataFrame içerisindeki eksik verilerimizi bulmamızı sağlar ve DataFrame ile aynı büyüklükte bir boolean objesi döndürür. Eksik değerler (None veya numpy.NaN gibi) True kabul edilir.

Bunun dışındaki boş string "" objeleri gibi diğer her şey False kabul edilir.

```
1 my_serie = pd.Series([1, "Python", np.NaN])  
2 my_serie.isna()
```

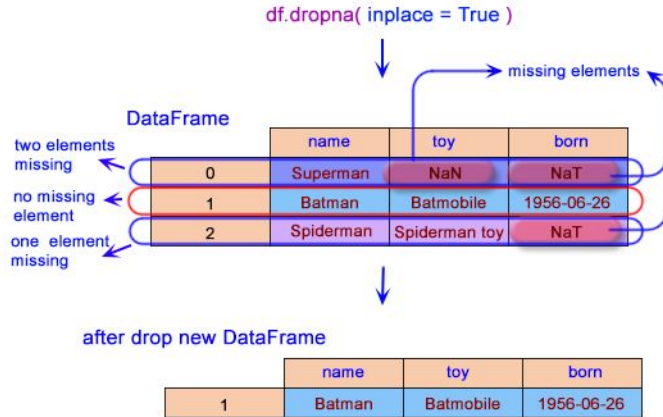
```
0    False  
1    False  
2     True  
dtype: bool
```

**Not 1:** df, DataFrame anlamına gelmektedir

**Not 2:** `numpy.inf`, `pandas.options.mode.use_inf_as_na = True` olmadığı sürece NA değer olarak kabul edilmemektedir

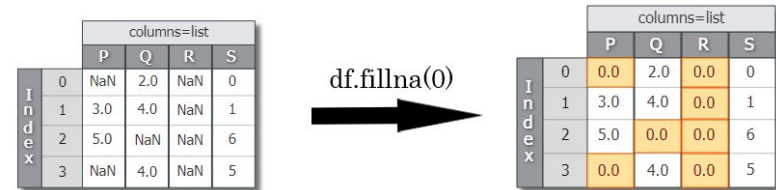
# df.dropna() vs df.fillna()

**dropna()** metodu herhangi bir değeri NaN olan bir satırı veya sütunu tamamıyla siler. Satır veya sütun silme işlemine axis değeri ile karar verir.



Bir satırı tamamen silerken modeliniz için önemli olan bilgileri de silebilirsiniz.

**fillna()** value metodu, verilen strateji (ortalama, medyan vs) veya değerle herhangi bir satırdaki eksik hücreyi doldurur.



Eğer boş bir hücreyi yanlış veri ile doldurursanız modelinizin yanlışlığı (bias) artacaktır.

# Boyut İndirgeme

# Boyutsallığın Laneti (Curse of Dimensionality)

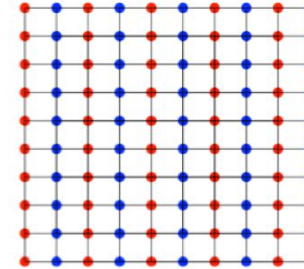
Curse of Dimensionality, Öklid uzayına ekstra boyutlar eklenmesiyle bağlantılı olarak hacimdeki ve hesaplama süresindeki üstel artışın neden olduğu problemdir.

Temel olarak özellik sayısındaki artışla birlikte hatanın artması anlamına gelir.

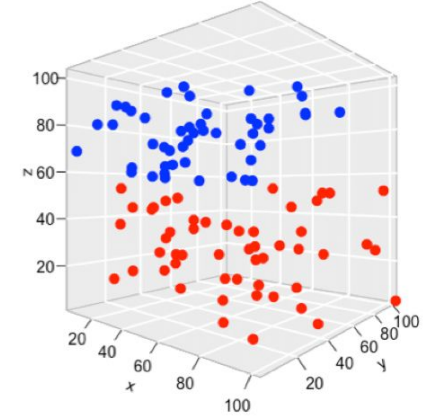
Daha yüksek sayıda boyut teorik olarak daha fazla bilginin depolanmasına izin verir, ancak gerçek dünya verilerinde daha yüksek gürültü ve fazlalık olasılığı nedeniyle pratikte nadiren yardımcı olur.



(A) 1-D



(B) 2-D



(C) 3-D



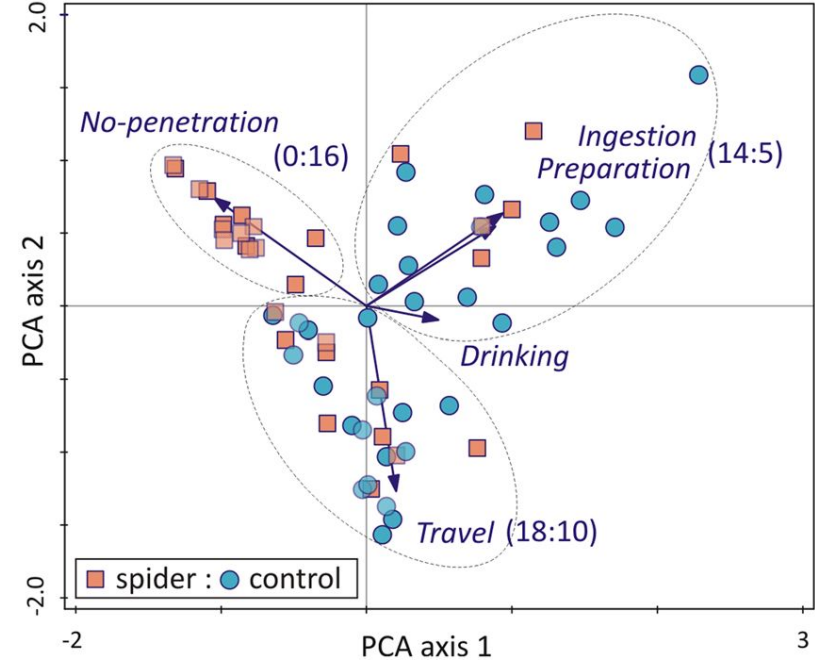


# Principal Component Analysis (PCA)

**Temel Bileşen Analizi (PCA)**, karmaşık bir veri kümesinin daha düşük bir boyuta nasıl indirgeneceğine dair bir yol haritası sağlar.

PCA algoritması verilerdeki temel özellikleri yakalayıp bu özellikleri daha az sayıda değişken ile göstermeye çalışır.

Büyük ve karmaşık veri kümelerinden ilgili bilgileri çıkartmak için boyut değiştirme, döndürme, boyut eğimi değiştirme gibi yöntemler kullanılır.



# Adım Adım PCA

**Adım 1:** Veri Kümesi standartlaştırılır

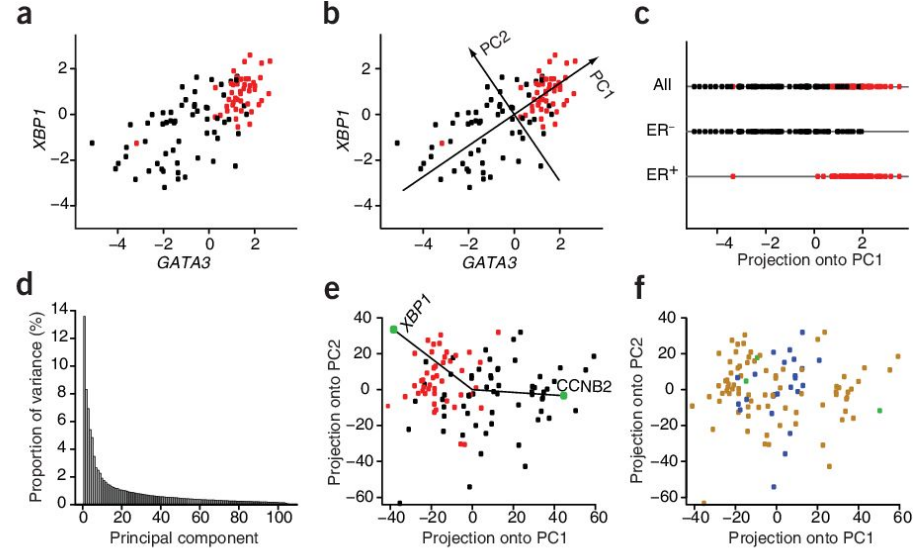
**Adım 2:** Veri kümesindeki özellikler için Covariance matrisi hesaplanır

**Adım 3:** Covariance matrisi için özdeğerleri ve özvektörler hesaplanır

**Adım 4:** Özdeğerleri ve bunlara karşılık gelen özvektörler hesaplanır

**Adım 5:** k özdeğeri seçilir ve bir özvektör matrisi oluşturulur

**Adım 6:** Orjinal matris dönüştürülür

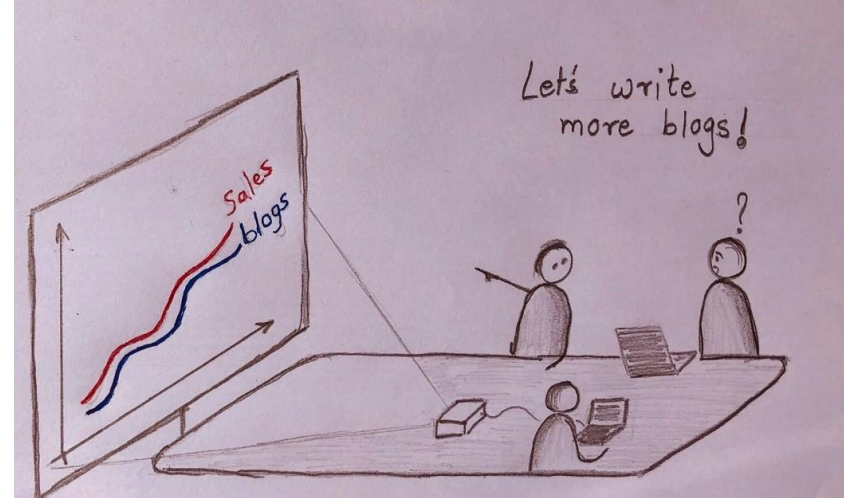


# Çoklu Doğrusal Bağlantı (Collinearity)

Çoklu doğrusallık, bağımsız değişkenler arasındaki korelasyonun yüksek olması durumudur. Bu durumda her bir değişkenin model tahminindeki etkisi bireysel olarak ölçülemez.

Bu problem tahmin edilen katsayıların kesinliğini azaltır, bu da regresyon modelinin istatistiksel gücünü zayıflatır.

Böyle bir durumda istatistiksel olarak anlamlı bağımsız değişkenleri belirlemek için p değerlerine güvenilmeyebilir.



# Veri Dönüşümü

Veri Dönüşüm Teknikleri

Özellik Mühendisliği (Feature Engineering)

# Veri Dönüşüm Teknikleri

# Round

Veriyi ondalık basamağa yuvarlar. Yuvarlama işlemi için herhangi bir basamak sağlanmazsa, sayıyı en yakın tam sayıya yuvarlar

```
round(float_num, num_of_decimals)
```

Input number	ROUND UP	ROUND DOWN	ROUND CEILING	ROUND FLOOR	ROUND HALF UP	ROUND HALF DOWN	ROUND HALF EVEN
5.5	6	5	6	5	6	5	6
2.5	3	2	3	2	3	2	2
1.6	2	1	2	1	2	2	2
1.1	2	1	2	1	1	1	1

# Round Python'da Nasıl Çalışır?

- Integer değerler round() fonksiyonu ile işleme sokulduğunda aynı değerde çıktı verir
- Float değerler, en yakın ondalık basamağa yuvarlanarak tam sayı halini alırlar
- Float değeri 0.5 olan sayılar, üst ondalığına yuvarlanır

```
# for integers
print(round(10))
# Output: 10

# for floating point
print(round(10.7))
# Output: 11

# even choice
print(round(5.5))
# Output: 6
```

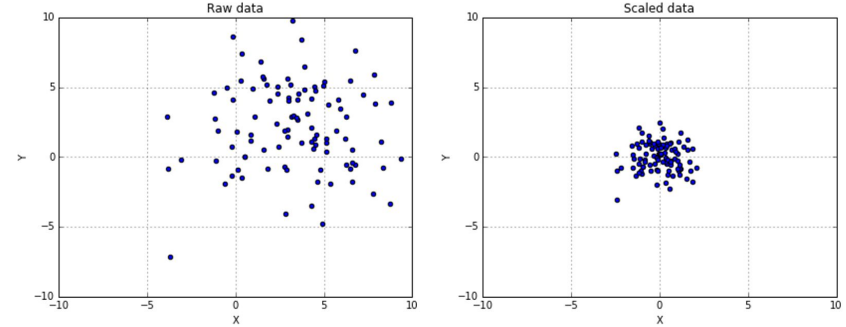


# Ölçekleme (Scaling)

Farklı ölçeklere sahip değişkenler, bir makine öğrenmesi modeline veya analize eşit olarak katkıda bulunmaz ve bir sapma (bias) yaratabilir.

Bu nedenle ölçekleme kullanarak verilerimizi belli bir standart içerisinde ifade ederiz. Ölçekleme sonrası farklı ölçekteki değişkenler eşit bir temelde karşılaştırılabilir hale gelir.

- Ölçekleme işlemi genellikle veri ön işleme adımı sırasında gerçekleşir.
- KNN ve SVM gibi veri noktaları arasındaki mesafenin önemli olduğu algoritmalarda ölçekleme çok önemlidir.



# Ölçekleme

## Min-Max Ölçekleme (Normalization):

Tüm veri değerlerini  $[0,1]$ ,  $[-1, 1]$  gibi dar bir aralığa sıkışacak şekilde orantılı olarak dönüştürür ve bu orantı sayesinde veri kümesinin dağılım şekli korunur.

- Aykırı değerlerden gerçekten etkilenir
- Veri dağılımının Gauss olmadığı bilindiği zaman kullanılması yararlıdır

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

## Standardization (z skoru):

Tüm veri değerlerinin, ortalama 0 ve standart sapma 1 olacak şekilde yeniden ölçeklendirilmesidir.

- Aykırı değerlerden çok daha az etkilenir
- Normal dağılımlı verilerde kullanılması yararlıdır

The diagram shows the Z-score formula  $Z = \frac{x - \mu}{\sigma}$  with red arrows and labels pointing to its components: 'Score' points to  $x$ , 'Mean' points to  $\mu$ , and 'SD' (Standard Deviation) points to  $\sigma$ .

$$Z = \frac{x - \mu}{\sigma}$$

# Label Encoding

Analiz ve görselleştirmeler numerik veriler üzerinden yapılır, bu nedenle kategorik veriler işlenmeden ve görselleştirilmeden önce numerik verilere çevrilmesi gerekmektedir.

Label Encoding, her bir kategorik veri için numerik bir değer ataması gerçekleştirir.

Label Encoding'te sorun, aynı sütunda farklı sayılar olduğundan model, verileri bir tür hiyerarşik sırada olacak şekilde yanlış anlayacaktır.

*Örneğin; yandaki veri için  $1 < 2 < 3$  sıralaması geçerli bir hiyerarşik sıralama değildir.*

Bu sorunun üstesinden gelmek için One Hot Encoder kullanılır.

Food Name	Categorical #
Apple	1
Chicken	2
Broccoli	3

# One Hot Encoding

Label Encoding'in aksine, One Hot Encoding her bir kategorik değer için yeni bir binary değer içerecek olan sütun oluşturur. Bu sayede kategorik bir veriyi Makine Öğrenmesi modelimizde daha anlamlı ve iyi temsil eder.

Kategorik verinin değerine karşılık gelen yeni oluşturulmuş sütun 1 değerini alırken, diğer kategorik veriler için oluşturulmuş sütunlar 0 değerini alır.

id	color
1	red
2	blue
3	green
4	blue



id	color_red	color_blue	color_green
1	1	0	0
2	0	1	0
3	0	0	1
4	0	1	0

# Stack

## df.stack()

Bir DataFrame veya Seri'yi, bir veya daha fazla iç seviyeye sahip bir DataFrame veya Seri'ye döndürür. Bir tane iç seviye varsa Seri, daha fazla iç seviye varsa DataFrame döndürülür.

Veri örnekleri kolon değerlerine göre kategorilendirilir ve bu değerlere indeksleme yöntemiyle ulaşılabilir.

*Örneğin yan taraftaki örnekte deer'in weight değerine*

`df_single_level_cols.stack()["deer"]["weight"]`  
*ile ulaşılabilir.*

**Not:** *df, DataFrame'i temsil etmektedir*

df\_single\_level\_cols.stack()



DataFrame	weight	height
deer	0	2
monkey	3	4



values in Series

deer	weight	0
	height	2
monkey	weight	3
	height	4

# Melt

## pd.melt()

Melt, Bir DataFrame'i geniş formattan uzun formata döndürür. Bu işlemi satırları sütuna çevirerek yapar.

Melt, istenen kolon isimlerini variable adı verilen bir kolona, bu kolonların her bir örnekteki değerini de value isimli kolona atayarak yapar.

***Not:** pd , Pandas'ı temsil etmektedir*

```
df.melt(id_vars=['first', 'last'])
```

	first	last	height	weight
0	John	Doe	5.5	130
1	Mary	Bo	6.0	150



	first	last	variable	value
0	John	Doe	height	5.5
1	Mary	Bo	height	6.0
2	John	Doe	weight	130
3	Mary	Bo	weight	150

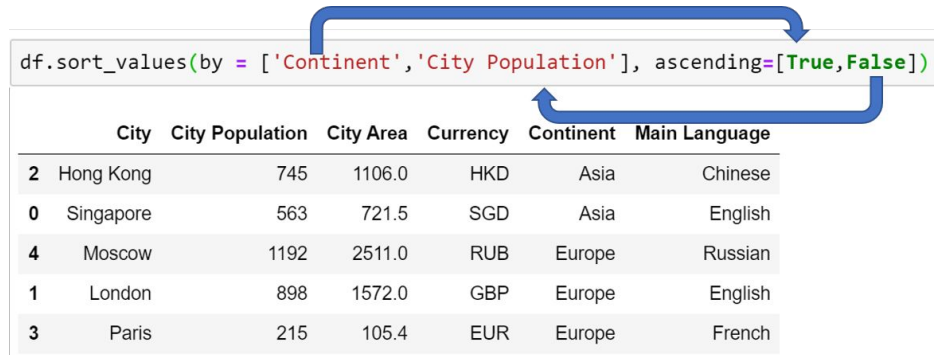
# Sort

## df.sort\_values()

Bu fonksiyon, DataFrame'i istenen sütundaki değerlere göre artan/azalan şekilde tekrardan sıralar.

- **by parametresi** ile verilen sütun ismi hangi sütun/sütunlara göre sıralama yapılacağını gösterir.
- **ascending parametresi** sıralama işlemini artan şekilde yapmak için kullanılabilir.

*Not: df, DataFrame'i temsil etmektedir*



```
df.sort_values(by = ['Continent', 'City Population'], ascending=[True, False])
```

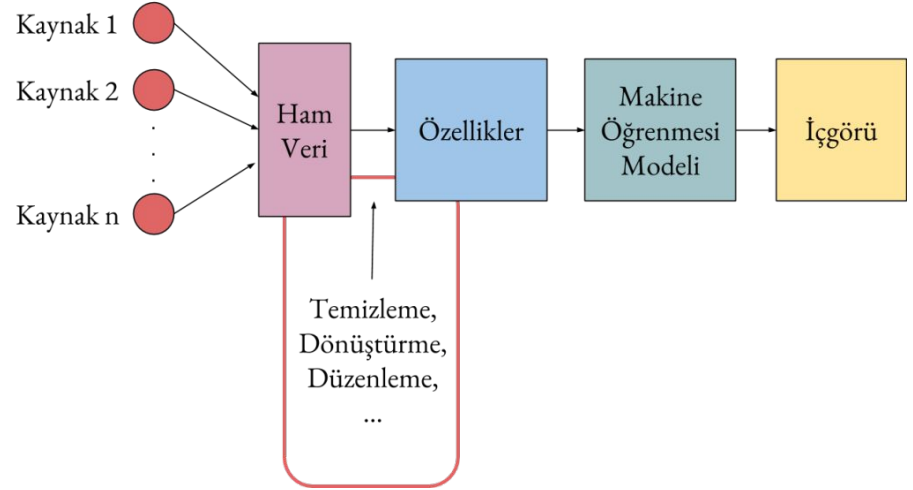
	City	City Population	City Area	Currency	Continent	Main Language
2	Hong Kong	745	1106.0	HKD	Asia	Chinese
0	Singapore	563	721.5	SGD	Asia	English
4	Moscow	1192	2511.0	RUB	Europe	Russian
1	London	898	1572.0	GBP	Europe	English
3	Paris	215	105.4	EUR	Europe	French

# Feature Engineering

Özellik mühendisliği, ham verilerden özellikleri çıkarmak için alan bilgisini kullanma sürecidir. Özellikler, tahmine dayalı modeller tarafından kullanılır ve sonuçları etkiler.

Özellikleriniz probleme uygun seçilmediğinde Makine Öğrenmesi modeliniz amaca uygun hizmet edemeyecek ve başarılı bir öğrenim gerçekleştiremeyecektir.

*Feature Engineering yapılmadan kullanılan veri ile beslenmiş model, Matematik sınavına hazırlanmak için Fizik dersi konularını çalışan öğrenciye benzetilebilir.*





Veri Azaltma

# Data Aggregation (Veri Toplama)

Veri toplama, istatistiksel analiz ve iş hedeflerine etkin bir şekilde ulaşmak için verilerin toplandığı ve özet bir biçimde sunulduğu süreçtir.

Veri toplama, büyük miktarda ham veriye dayalı kararlar almaya yardımcı olduğu için veri ambarı için hayati önem taşır.

Veri toplama, gelecekteki eğilimleri tahmin etme yeteneği sağlar ve tahmine dayalı modellemeye yardımcı olur.

Etkili veri toplama teknikleri, performans sorunlarını en aza indirmeye yardımcı olur.



# Matematiksel Fonksiyonlar ve Aggregation Çeşitleri

**Veri toplama** ile ham veriler toplanır ve istatistiksel analiz için bir özet biçiminde ifade edilir. Örneğin, ortalama, minimum, maksimum, toplam ve count gibi istatistikler sağlamak için ham veriler belirli bir süre boyunca toplanabilir.

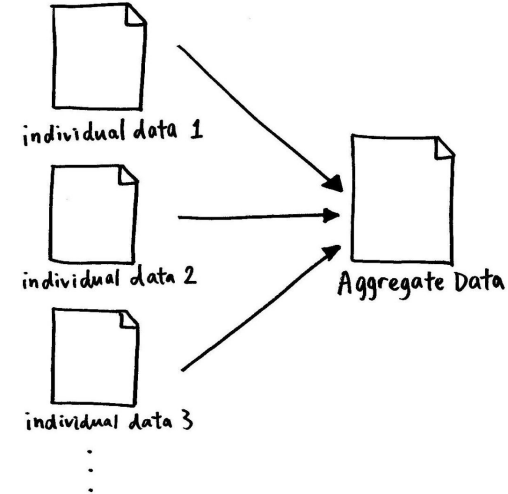
**Sum:** Bir toplamı elde etmek için belirtilen tüm verileri bir araya getirir.

**Average:** Belirli verilerin ortalama değerini hesaplar.

**Max:** Her kategori için en yüksek değeri görüntüler.

**Min:** Her kategori için en düşük değeri görüntüler.

**Count:** Her kategori için toplam veri girişi sayısını sayar.



# Colab Vakti!