

DATA SCIENCE HANDBOOK

MELİH GÜLÜM

2023

CRM	5
RFM	5
CLTV	8
CLTV PREDICTION	9
Ölçümleme Problemleri	13
Rating Products.....	13
Sorting Products.....	13
Sorting Reviews.....	14
AB TEST.....	15
Recommender Systems	19
Association Rule Learning	19
Apriori Algoritması.....	19
Content Based Filtering.....	21
Metinleri Vektörel Temsilleri	22
Collaborative Filtering	23
Item-Based Collaborative Filtering.....	24
User-Based Collaborative Filtering.....	24
Neighbourhood-Based Collaborative Filtering.....	24
Hybird	25
Model-Based Matrix Factorization	25
Matrix Factorization.....	25
Gradient Descent.....	28
Feature Engineering	29
Outliers:.....	29
Missing Values	31
Encoding	32
Label Encoding:.....	32
One Hot Encoding:.....	32
Rare Encoding:	32
Feature Scaling:	33
Feature Engineering:.....	34
Machine Learning	35
Temel Kavramlar	35
Değişken Tipleri.....	35
Öğrenme Türleri.....	35
Model Başarı Değerlendirme Yöntemleri	36

Model Doğrulama Yöntemleri	36
Bias-Variance Tradeoff	37
Linear Regression	38
Logistic Regression.....	41
KNN.....	45
CART (Classification and Regression Tree)	46
Gelişmiş Ağaç Yöntemleri	48
Random Forest.....	48
Gradient Boosting Machines (GBM)	50
XGBoost (eXtreme Gradient Boosting)	52
LightGBM	52
CatBoost (Categoric Boostin).....	52
Unsupervised Learning	52
K-Means	52
Hierarchical Cluster Analysis	53
Principal Component Analysis (PCA):	54
SQL (Structred Query Language).....	55
Temel Kavramlar	55
İlişkisel Veritabanı Kavramı (RDBMS):.....	56
Veri Tipleri ve Normalizasyon	56
Tam Sayı Veri Tipleri	56
String Veri Tipleri.....	57
Ondalık Sayı Veri Tipleri	58
Tarih ve Saat Veri Tipleri.....	58
Diğer Veri Tipleri.....	58
SQL Dili:	59
Aggregate Functions ve Groupby	60
İlişkisel Veritabanı Kavramı (RDMS)	62
Subquery:.....	63
Time Series.....	64
Temel Kavramlar:	64
Stationary (Durağanlık)	64
Trend	65
Seasonality (Mevsimsellik)	65
Cycle (Döngü)	66
Moving Average (Hareketli Ortalama)	66

Weighted Average (Ağırlıklı Ortalama).....	67
Smoothing Yöntemleri (Holt-Winters)	68
SES (Single Exponential Smoothing)	68
Double Exponential Smoothing (DES)	69
Triple Exponential Smoothing (aka Holt-Winters).....	70
İstatistik Metodlar	71
AR(p): Autoregression	71
MA(q): Moving Average.....	71
ARMA(p, q).....	72
ARIMA.....	72
SARIMA	73
Zaman Serileri Ekstra Bilgiler	74

CRM

CRM dünyasındaki kavramlara bakacak olursak;

- Customer lifecycle/journey/funnel,
- İletişim (dil, renk, görseller, kampanyalar),
- Müşteri edinme / bulma çalışmaları,
- Müşteri elde tutma (terk) çalışmaları,
- Çapraz - üst satış (cross - up sell)
- Müşteri segmentasyon çalışmaları karşımıza çıkar.

KPI:

- **Customer Acquisition Rate** (Müşteri Kazanma Oranı — Belirli bir zamanda kazanılan müşteri yüzdesi)
- **Customer Retention Rate** (Müşteri Elde Tutma Oranı)
- **Customer Churn Rate** (Müşteri Terk Oranı)
- **Conversion Rate** (Dönüşüm Oranı)
- **Growth Rate** (Büyüme Oranı)

RFM

- RFM Analizi müşteri segmentasyonu için kullanılan bir tekniktir.
- Müşterilerin satın alma alışkanlıklarını üzerinden gruplara ayrılması (bu yüzden de Monetary alınmıyor) ve bu gruplar özelinde stratejiler geliştirilebilmesini sağlar
- CRM çalışmaları için birçok başlıkta veriye dayalı aksiyon alama imkan sağlar:

RFM Metrikleri:

- Recency (Yenilik) = Müşterinin bizden en son ne zaman alışveriş yaptığı ifade etmektedir.
- Frequency (Sıklık) = Örneğin müşterinin toplam yaptığı alışveriş sayısıdır, diğer bir ifadeyle işlem sayısıdır
- Monetary (Parasal değer) = Müşterilerin bize bıraktığı parasal değeri ifade eder.

	<i>R</i>	<i>F</i>	<i>M</i>
<i>Müşteri1</i>	80	250	5200
<i>Müşteri2</i>	7	560	2300
<i>Müşteri3</i>	1	120	300
...
<i>Müşterix</i>	35	300	4500

- Buradaki değerleri hem kendi içinde hem de birbirleri arasında kıyaslanabilir yapmak gerekmektedir.
- RFM Metriklerini RFM Skorlarına çevirmek gerekmektedir.

RFM Skorları:

- Metrikleri skorlara çevirmek demek hepsini aynı cinsten ifade etmek demektir. Böylece kıyaslanabilir hale getirmiş olacağız.

	<i>R</i>	<i>F</i>	<i>M</i>	<i>RFM</i>
<i>Müşteri1</i>	1	4	5	145
<i>Müşteri2</i>	4	5	4	454
<i>Müşteri3</i>	5	1	3	513
...
<i>Müşterix</i>	2	4	4	244

- Recency Metriğinin küçük olması skoru yüksek tutar diğerlerinin aksine. Skorları bir araya getirerek RFM Skoru oluşturulur.
- RFM skorunun ortaya çıkaracağı çok fazla sayıda kombinasyon vardır. Bütün kombinasyonlar ile uğraşmak, istediğimiz gibi segmentlere ayıramamak demektir.
- Bu yüzden daha mantıklı gruplar/segmentler oluşturmak için yaygın olarak kullanılan tablo vardır.

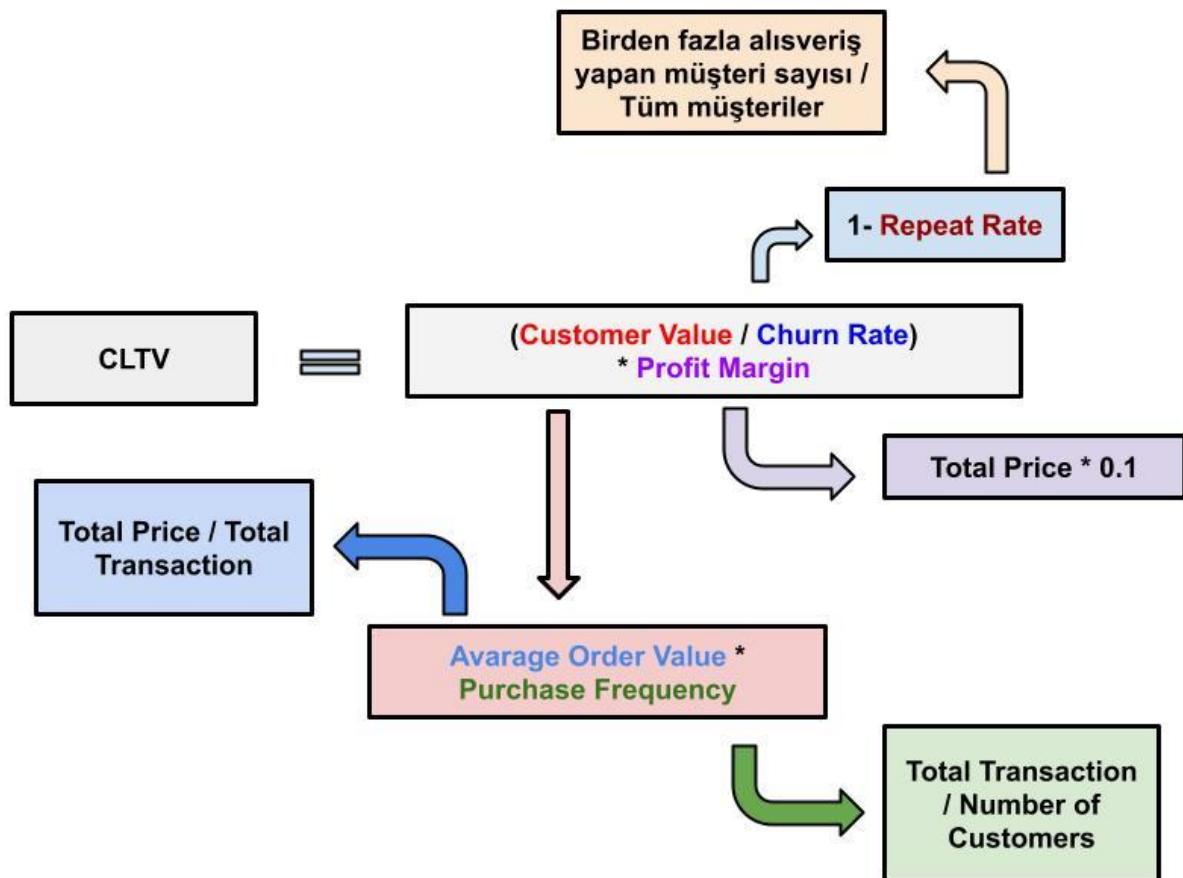
RFM Table:



- Monetary yok.
- CRM Analitигinde mүsterilerin bizle kurduğu ilişkiler frekans/transaction daha önemlidir. Az işlem yapan birinin
- Monetary değerine bakmak daha az anlamlı olacaktır.

	<i>R</i>	<i>F</i>	<i>M</i>	<i>RFM</i>	<i>Segment</i>
<i>Müsteri1</i>	1	4	5	145	At Risk
<i>Müsteri2</i>	4	5	4	454	Loyal Customer
<i>Müsteri3</i>	5	1	3	513	New Customer
...
<i>Müsterix</i>	2	4	4	244	At Risk

CLTV



- Customer LifeTime Value = Bir müşterinin bir şirkette kurduğu ilişki-iletişim süresince bu şirkete kazandıracağı parasal değerdir.

$$\text{Customer Lifetime Value} = \frac{\text{Customer Value}}{\text{Churn Rate}} * \text{Profit Margin}$$

$$\text{Customer Value} = \text{Average Order Value} * \text{Purchase Frequency}$$

$$\text{Average Order Value} = \frac{\text{Total Price}}{\text{Total Transaction}}$$

$$\text{Purchase Frequency} = \frac{\text{Total Transaction}}{\text{Total Number of Customers}}$$

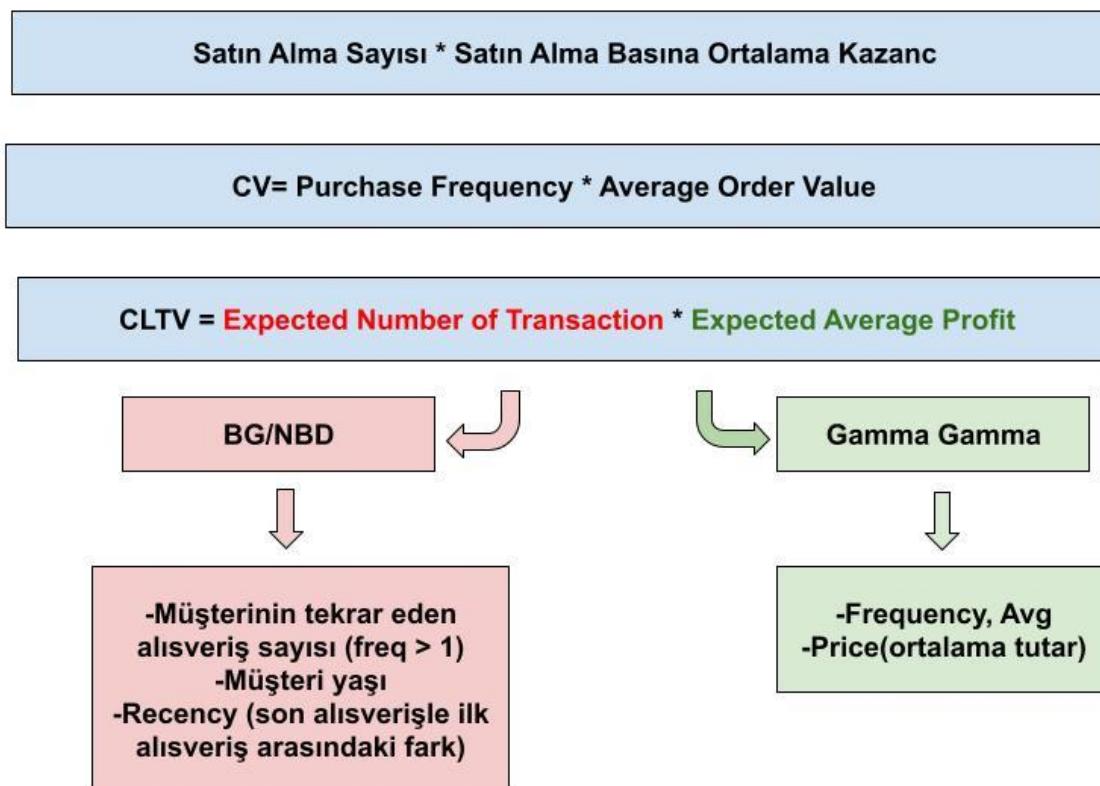
$$\text{Churn Rate} = 1 - \text{Repeat Rate}$$

$$\text{Repeat Rate} = \frac{\text{Birden Fazla Alışveriş Yapan Müşteri Sayısı}}{\text{Tüm Müşteriler}}$$

$\text{Profit Margin} = \text{Total Price} * 0.10 \rightarrow 0.10$ bir sabittir ve şirket tarafından belirlenmekte.

- Churn Rate = Müşteri terk oranıdır.
- Repeat Rate = Tekrar oranı, Retention rate, elde tutma oranıdır.
- Her bir müşteri için hesaplanacak CLTV değerlerine göre bir sıralama yapıldığında ve CLTV değerlerine göre belirli noktalardan bölme işlemi yapılarak gruplar oluşturulduğunda müşterilerimiz segmentlere ayrılmış olacaktır.

CLTV PREDICTION



$$\text{Customer Lifetime Value} = \frac{\text{Customer Value}}{\text{Churn Rate}} * \text{Profit Margin}$$

$$\text{Customer Value} = \text{Purchase Frequency} * \text{Average Order Value}$$

- Şimdi öyle bir şey yapılmalı ki bütün kitlenin satın alma davranışları ve bütün kitlenin işlem başına ortalama bırakacağı kazancı olasılıksal olarak modelleyebilelim ve bu olasılıksal modele bir kişinin özelliklerini girerek edinilen genel kitle davranışının üzerinden bir tahmin de bulunabilelim.

$$CLTV = \text{Conditional Expected Number of Transaction} * \text{Conditional Expected Average Profit}$$

- Aslında bütün kitlenin satın alma davranışlarını olasılık dağılımı ile modelleyeceğiz. Daha sonra bu olasılık dağılımı ile modellediğimiz davranış biçimlerini koşullu yani kişi özelinde biçimlendirecek şekilde kullanarak her bir kişi için beklenen satın alma sayılarını, beklenen işlem sayılarını tahmin edeceğiz. Average Profit için de aynı şey geçerli. Kişi özelinde koşullayacağımız ve beklenen kazancı bulacağız.

Bu işleri bizim için yapan iki model var:

1. BG/NBD Model
2. Gamma Gamma Submodel

$$CLTV = BGNBD \text{ Model} * \text{Gamma Gamma Submodel}$$

$$\text{Customer Lifetime Value} = \frac{\text{Customer Value}}{\text{Churn Rate}} * \text{Profit Margin}$$

$$\text{Customer Value} = \text{Purchase Frequency} * \text{Average Order Value}$$

$$CLTV = \text{Conditional Expected Number of Transaction} * \text{Conditional Expected Average Profit}$$

$$CLTV = BGNBD \text{ Model} * \text{Gamma Gamma Submodel}$$

BG/NBD (Beta Geometric / Negative Binomial Distribution):

- Expected ifadesi bir rassal değişkenin beklenen değerini ifade etmek için kullanılır. Bir rassal değişkenin beklenen değeri ise o değişkenin ortalaması demektir. Rassal değişken ise değerlerini bir deneyin sonucundan alan değişkenlere denir.
- Expected Number of Transaction with BG/NBD yerine Expected Sales Forecasting with BG/NBD de denilebilirdi. Zaten BG/NBD modeli tek başına satın alma sayılarını/satış tahmin etmek için kullanılır.
- BG/NBD modeli literatürde Buy Till You Die olarak da anılmaktadır. BG/NBD modeli, Expected Number of Transaciton için iki süreci olasılıksal olarak modeller.

1. Satın alma süreci - Transaction Process (Buy)

2. Drop olma, Churn Olma, inaktiv olma oranı var - Dropout Process (Till You Die)

İşte BG/NBD modeli bunu olasılıksal olarak modellemektedir.

Transaction Process (Buy):

- Alive olduğu sürece, belirli bir zaman periyodunda, bir müşteri tarafından gerçekleştirilecek işlem sayısı transaction rate parametresi ile possion dağılır.
- Transaction rate'ler her bir müşteriye göre değişir ve tüm kitle için gamma dağılır. (r, a)

Dropout Process (Till You Die):

- Her bir müşterinin p olasılığı ile dropout rate (probability)'i vardır. Yani Churn olma.
- Dropout Rate'ler her bir müşteriye göre değişir ve tim kütle için beta dağılır. (a, b)

BG/NBD Modelinde Denklemi açıklaması:

$$E(Y(t)|X = x, t_x, T, r, \alpha, a, b) = \frac{a + b + x - 1}{a - 1} \times \frac{\left[1 - \left(\frac{\alpha + T}{\alpha + T + t} \right)^{r+x} {}_2F_1(r + x, b + x; a + b + x - 1; \frac{t}{\alpha + T + t}) \right]}{1 + \delta_{(x>0)} \frac{a}{b + x - 1} \left(\frac{\alpha + T}{\alpha + t_x} \right)^{r+x}}$$

x = bir müşterinin tekrar eden satış sayısıdır. Yani en az 2. kez alışveriş yapma durumu

t_x = recency değeridir. RFM^den farkı today_Date'e göre değil müşteri özelinde olmasıdır. Yani kendi ilk ve son

alışveriş arasındaki farktır. (haftalık)

T = Müşterilerin ilk satın alma süresi üzerinden geçen zamandır. Müşterinin yaşı (haftalık)

r, alpha (a) = Transaction rate'i modelleyen Gamma dağılımının parametreleridir.

a, b = İnaktif olasılığı, drop rate olasılığını modelleyen Beta'nın parametreleridir.

r, a, a ve b kitemizden öğreneceğiz olumlu olasılık dağılımı parametreleridir.

x, x_t ve T her bireyin özelinde özelleştirecek olumlu değerlerdir.

O halde belirli bir t periyodunda beklenen transaction sayısını bulmuş olacağız.

*** Normalde bizim için Recency'i düşük olması iyiydi. Fakat Buy Till You Die Kavramı derki düzenli ortalama bir işlem kapasitesi olan müşterin eğer Churn olmadıysa müşterinin kendi içindeki, kullanıcı özelindeki Recency'si arttıkça satın alma olasılığı yükseliyor der. Çünkü müşteri alışveriş yaptı ve kısmı Churn oldu. Alma ihtiyacı bitti vs gibi sebeplerden dolayı müşteri kısmı drop oluyor. Bekler bekler ve tekrar satın alma ihtiyacı ortaya çıkılmaya başlar. Diğer değişkenler de incelenmeli fakat müşteri yaşı (T) ve recency değerleri oldukça yüksek veya birbirlerine yakın değerler ise bizim için potensiyelli bir müşteridir ve CLV değeri yüksek gelebilir.

Gamma Gamma Submodel:

$$CLTV = \text{Conditional Expected Number of Transaction} * \text{Conditional Expected Average Profit}$$

$$CLTV = BGNBD \text{ Model} * \text{Gamma Gamma Submodel}$$

- Average Order Value'nın olasılıksal halidir.
- Bir müşterinin işlem başına ortalama ne kadar kar getirebileceğini tahmin etmek için kullanılır.
- Bir müşterinin işlemleri parasal değeri (monetary) transaction value'larının ortalaması etrafında rastgele dağılır.
- Ortalama transaction value, zaman içinde kullanıcılar arasında değişebilir fakat tek bir kullanıcı için değişmez.
- Ortalama transaction value tüm müşteriler arasında gamma dağılır.

Gamma Gamma Submodel Denklemi:

$$E(M|p, q, \gamma, m_x, x) = \frac{(\gamma + m_x x)p}{px + q - 1} = \left(\frac{q - 1}{px + q - 1} \right) \frac{\gamma p}{q - 1} + \left(\frac{px}{px + q - 1} \right) m_x$$

x = bir müşterinin tekrar eden satış sayısıdır. Yani en az 2. kez alışveriş yapma durumu

m_x = monetary'dır. Yani gözlemlenen transaction value'larıdır. Yani total_price/total_transaction demektir.

Kişi ve dağılım özelliklerini girildiğinde monetary değerinin beklenen değerini verir.

Ölçümleme Problemleri

Rating Products

- Olası faktörleri göz önünde bulundurarak ağırlıklı ürün puanlama gerçekleştireceğiz.
- Şirketlerin ürünlerine nasıl bir puan vermesi gerektiğini hesaplamaya çalışacağız.

Time-Based Weighted Average =

Ürünü mean ile incelemek bazen doğru olmayabilir. Çünkü ürün pozitif/negatif trende uğramış olabilir. Bütün puanlar ağırlığını aynı oranda hissettirdiğinden dolayı daha çok ya da daha az beğenme trendi kaçıyor olacaktır. O nedenle Sadece puan ortalaması almak yerine Puan Zamanlarına Göre Ağırlıklı Ortalama yapılabilir. Örneğin son 30 güne şu ağırlık son 60 güne şu ağırlık ver denilebilir.

User-Based Weighted Average =

Bütün kullanıcıların verdiği puanlar aynı ağırlığa mı sahip olmalı? Kursu az izleyen ile bitirenin ağırlığı aynı mı olmalı? Burada 1 ürün almış 1 yorum yapmış olan kişiler olabilir. Bu sahtekarlıklarının/aldatıcılıkların önüne geçilmiş olunuyor.

Weighted Rating = 2 Ağırlığı bir araya getirerek tek bir ağırlık oluşturacağız. 0,

Sorting Products

- Default rating'e bazı durumlarda güvenemeyebiliriz. Çünkü sadece rating'e göre sıralarsak diğer önemli olabilecek faktörlerin değeri ezilmektedir. O yüzden purchase_count ve comment_count'a bakabiliriz. Ama tek başına purchase_count'a ve comment_count'a bakarsak bir anlam ifade etmeyecek. Bu sıralamayı yapmanın yolu bu 3 faktörü belirli bir standarta (scaler) getirip daha sonra tek bir skor haline getirmeliyiz.

Sorting by Rating, Comment and Purchase:

- Eğer 3'ünüde çarpalım dersek; yüksek purchase veya comment'e sahip olanlar ezici üstünlüğe sahip olabilir. O yüzden hepsini aynı cinse getirmeliyiz. Mesela rating cinsine, 1-5 aralığına getirelim. Daha sonra önem derecesine göre ağırlıklar verip tek bir skor haline getirebiliriz.

Bayesian Average Rating Score:

- Puan dağılımlarının üzerinden ağırlıklı ortalama hesabı yapar.

- Bu konu karşımıza şu iki şekilde de gelebilir:
 1. Sorting Products with 5 Star Rated,
 2. Sorting Products According to Distribution of 5 Star Rating
- bar_score rating'lerin dağılımına bakarak oluşturulan bir skor. O yüzden rating hesabı için de kullanılabilir.
- bar_score'da gözden kaçırılan hususlar söz konusu: Social Proofs (Yorum, Satın Alma)
- Tek odağımız rating ise bar_score kullanılabilir.
- bar_score ve weighted_sorting_score'lar ikisi de farklı sıralamalar getirdi. bar_score 'da yine orada olmasını beklememiş olduğumuz kurslar yukarı çıktı. Neden beklemiyoruz çünkü Purchase-comment'i düşük.
- bar_score aslında potansiyeli temsil etmektedir.

Hybrid Sorting :

Sorting Reviews

Up-Down Difference Score =

- (up ratings) - (down ratings)
- (600+, 400-), (5500+, 4500-)
- Burada farklardan dolayı kazanan ikinci yorum olacaktır. Ama yüzdelere baktığımızda ise ilk yorum %60, ikinci ise %55 olacaktır. Farklılıkla dolayı ikinci kazanıyor gibi gözüke de yüzdelikten dolayı birinci yorum kazanmaktadır.
- Up-Down Difference Score bir miktar yanlışlık barındırmaktadır.
- Burada bir frekans bilgisi gerekli, bir threshold belirlenmeli.
- Up-Down işleminde fark üzerine yapılan yorum bu nedenle yeteri kadar doğru olmamaktadır.

Average Rating/Up Ratio =

- (up ratings) / (all ratings)
- Burada da yukarıda örnekte birinci yorum 0.6, ikinci yorum 0.55 değerini alır.
- Başka bir yorumu bakalım:

1. 2+, 0-

2. 100+, 1-

- Burada ise ilk yorum 1, ikinci yorum ise 0.99 değerini alacaktır ve en yukarıda ilk yorum gösterilecektir. Ama ikincinin yukarıda gözükmek gerekmektedir.
- Frekans yüksekliğini, sayı yüksekliğini göz önünde bulundurmadık.

Wilson Lower Bound Score =

- alacağımız örneklemelerin yüzde 95 oranında (alt ve üst skor) çıkacağı aralığı verir
- İkili interaction'lar (like/dislike) barındıran herhangi ürün veya yorumu skorlama imkanı sağlar.
- Bernoulli (olasılık dağılımının, iki olayın gerçekleşmesi olasılığını hesaplamak için kullanılır) parametresi p için bir güven aralığı hesaplar ve bu güven aralığının alt skorunu WLB olarak kabul eder.
- Eğer skorlar 1-5 arasındaysa 1-3 negatif, 4-5 pozitif olarak işaretlenir ve bernoulli'ye uygun hale getirilebilir. Bu beraberinde bazı problemleri de getirir. Bu sebeple bayesian average rating yapmak gereklidir (Skorların 1-5 arasında olduğu durumlarda, skorun '0' olarak hesaplama ihtimalinin olması).
- Elimizdeki örnekleme ilişkin up rate oranının istatistiksel olarak %95 güven ve %5 hata payı ile hangi aralıkta olabileceğini biliyoruz ve en alt sınırdaki referans noktasını aldık.

AB TEST

Sampling (Örneklem) =

- Bir ana kitle içerisindeki bu ana kitlenin özelliklerini iyi taşıdığı/temsil ettiği varsayılan bir alt kümedir.
- Örneklem sayısı arttıkça bu örneklemelere ait toplam ortalama da popülasyona yakınsıyor olacaktır.

Descriptive Statistics =

- Eğer elimizdeki değişkenin dağılımı çarpık ise, yani içerisinde aykırı değerler var ise bu durumda bu değişkeni temsil etmek için mean değil medyan (%50) kullanılmalıdır. Aksi halde bir miktar yanıltıcı olabilecektir.
- Aykırı değer olup olmadığını nereden anlarız? Mean ile medyanı kıyaslarsak birbirine çok yakın değerler ise ilgili değişkeni temsil etmek için kullanacak olduğumuz istatistik

medyan ya da mean olması fark etmeyecektir. Eğer ikisi arasında uçurum varsa mean, medyan ve std'yi verip değerleri bunlardır demek mantıklı olacaktır.

- Mean ayrıkırı değerlerden etkilenmektedir.

Confidence Intervals =

- Peki neden güven aralığı diye bir şey ortaya çıkmış? İlgiilenilen olayın Gerçekleşecek olan sonuçların, olaylar gerçekleşmiş olsa olasılıksal olarak bir aralık ile bunun ne olacağını çıkarıyoruz. Tamamen ihtiyaç, Bakkal Mehmet.
- Anakütle parametresinin tahmini değerini (istatistik) kapsayabilecek iki sayıdan oluşan bir aralık bulunmasıdır.
- Formülden mantıken eğer Standart Sapma yüksek olanın, std düşük olana göre güven aralığının daha geniş olmasını bekleriz.

Correlation =

- Değişkenler arasındaki ilişki, bu ilişkinin yönü ve şiddeti ile ilgili bilgiler sağlayan istatistiksel bir yöntemdir.

Pozitif Korelasyon = Bir değişkenin değerleri artarken diğerinin de artacağı anlamına gelir.

Negatif Korelasyon = Bir değişkenin değerleri artarken diğerinin ise azalacağı anlamına gelir.

Hypothesis testing =

- **Neden H_0 'a göre yorum yapılır?** H_0 red olunca hatanın matematiksel karşılığı (%5 hata yapma olasılığımız var.) var ama H_1 'nin bir matematiksel karşılığı yok. O yüzden H_0 'a göre yorum yapılır.
- P-value = P-değeri, olayların şans eseri meydana gelme olasılığının (yani sıfır hipotezin doğru olması) ne kadar olası olduğunu açıklayan bir sayıdır
- Hipotez, bir inanışı, savı test etmek için kullanılan istatistiksel yöntemlerdir.
- Hipotez testi kapsamında odaklanacak olduğumuz konu "Grup Karşılaştırmaları" olacaktır. Grup Karşılaştırmalarındaki temel amaç olası farklılıkların şans eseri ortaya çıkıp olmadığını göstermeye çalışmaktadır.
- Şans demek senin kontrolünde olmayan belirli olaylar, inşaat vs.
- Farklılıkların bilimsel/istatistikî olarak (şansa bırakmadan) ortaya çıkıp olmadığını ispat etmek lazımdır.
- **Pvalue 0.051 gelince ne yapacağımız peki red mi edeceğiz?**

- Hata payı artırılabilir
- Örneklem sayısı artırılabilir. Örneklem dağılımı olmalı.
- Doğru bir örneklem mi seçildi? O kontrol edilir.
- Veriyi manipüle etmeden gözlemlemeyi sürdürürüm.
- **20k 70k** arasında bariz bir fark var. Ama belirli bir sonuca varmak istiyorsak sonucu belli olmasına rağmen yapabilir.

AB Test (İki Grup Ortalamasını Karşılaştırma - Bağımsız İki Örneklem T Testi) =

- Burada A ve B ismi kontrol grubu ve deney grubunu temsil etmektedir.
- AB testi dendiğinde yaygın olarak:
 1. İki Grubun ortalaması kıyaslanıyor.
 2. İki gruba ilişkin oranlar kıyaslanıyor.
- Ele alacak olduğumuz konu Bağımsız İki Örneklem T Testidir. Yani 2 grup karşılaştırma testidir. AB testi iki grup ortalaması arasında karşılaştırma yapmak istendiğinde kullanacağız.
- Yapacak olduğumuz işlemler:
 1. Hipotezleri Kur
 2. Varsayımlar Kontrolü
 - i. Normallik Varsayımları
 1. İlgili grupların dağılımının normal olması varsayımlı
 - ii. Varyans Homojenliği
 1. İki grubun varyanslarının dağılımının birbirine benzer olması
 3. Hipotezin Uygulanması
 - i. Varsayımlar sağlanıyorsa bağımsız iki örneklem t testi (parametrik test)
 - ii. Varsayımlar sağlanmıyorsa mannwhitneyu testi (non-parametrik test)
 4. p-value değerine göre sonuçları yorumla

Not:

- Normallik sağlanmıyorsa direk 2 numara. Varyans homojenliği sağlanmıyorsa 1 numaraya argüman girilir. Argüman olarak T testini kullan ama Varyans homojenliği sağlanmadığı bilgisi girilir.
- Normallik incelemesi öncesi aykırı değer incelemesi ve düzeltmesi yapmak faydalı olabilir.

shapiro testi = Bir değişkenin dağılımının normal olup olmadığını test eder.

levene testi = Varyans dağılımın homojen olup olmadığını incelemek için

t testi = varsayımlar sağlanıyorsa kullanılır. Eğer varyans homojenliği yoksa equal_var=False yapılır ve arka tarafta Welch testi yapar.

mannwhitney = Non-parametrik medyan/sıralama kıyaslama testidir.

AB Testi (İki Örneklem Oran Testi) =

- İki oran arasında karşılaştırma yapmak için kullanılır.
- Örneklem sayısı, n 30'dan büyük olmalıdır.

proportion_ztest = birinci bölüme başarı sayıları, ikinci bölüme toplam gözlem sayıları verilir, her biri için ayrı ayrı

İkiden Fazla Grup Ortalaması Karşılaştırması (ANOVA - Analysis of Variance) =

- Önceki bölümde gördüğümüz gibi ikişerli gruplar halinde yapsaydık hata değerlendirme işleri sağlıklı olmayacağından. Bu sebeple 2+ grup varsa ANOVA yöntemi önerilir.
- Varsayımlar sağlanıyorsa one way anova
 - parametrik anova testi = f_oneway metodu
- Varsayımlar sağlanmıyorsa kruskal
 - non-parametrik anova testi = kruskal
- Peki farklılık var ama hangi gruptan kaynaklı bir farklılık ortaya çıktı? Bunu için statsmodel'deki çoklu karşılaştırmayı (MultiComparison) kullanacağız.
- Bu işlem sonucunda iki karşılaştırmalar yapıldığında kayda değer bir fark olmayabilir. Ama ANOVA'da fark bulmuştu. Peki ne yapabilirim?
 1. alpha değeri, p ile oynanabilir.
 2. Bütün gruba bakınca fark bulduk ama bu farkı ikili karşılaştırmalarda göremedik dolayısıyla fark yokmuş muamelesi yapmak tercih edilebilir. Peki bunda bir yanlışlık yok mu? Yok. Bütün gruba ANOVA ile bakmak ile ikili karşılaştırma ile bakmak birbirinden farklıdır.

Recommender Systems

- Kullanıcıya bazı teknikler kullanarak ürün ya da hizmet önermek/tavsiye etmek demektir.
- Temel amacımız bol olan içerik, ürün, hizmet, film arasından kullanıcının ilgilenebileceği daha alt bir içerik setini kullanıcıya ulaştırmak (filtreleme yapmak)

Simple Recommender Systems:

- İş bilgisi ya da basit tekniklerle yapılan genel öneriler
- Kategorinin en yüksek puanlıları, trend olanlar, efsaneler ... (Kural Tabanlı vs)
- O kadar yöntem varken neden bu seçilir?
 - Veri yoksa
 - Basittir ve etkilidir. Herkes trendleri görmek ister.

Association Rule Learning:

- Birlikteki analizi ile öğrenilen kurallara göre ürün önerileri

Content Based Filtering:

- Ürünlerin benzerliğine göre öneriler yapan uzaklık temelli yöntemler

Collaborative Filtering

- Topluluğun kullanıcı ya da ürün bazında ortak kanaatlerini yansıtın yöntemlerdir.
- User-Based
- Item-Based
- Model-Based (Matrix Factorization)
- User-Based ve Item-Based bazen **Memory-Based** olarak, Model-Based ise **Latent Methods (Factors)** olarak görebilirsiniz.

Association Rule Learning

- Veri içerisindeki örüntüleri (pattern, ilişki, yapı) bulmak için kullanılan kural tabanlı bir makine öğrenmesi tekniğidir.

Apriori Algoritması

- Sepet analizi yöntemidir.
- Ürünlerin birliktekliliklerini ortaya çıkarmak için kullanılır.

- Apriori üzerinden hesaplayacak olduğumuz ve yorumlarının değerli olduğu 3 temel metrik vardır. Bunlar ile veri seti içindeki ilişki, örüntüleri gözleme imkanı buluruz.

1. Support

X ve Y'nin birlikte görünme olasılığı / Tüm işler

$$\text{Support}(X, Y) = \text{Freq}(X, Y)/N$$

(X ve Y'nin birlikte görülme olasılığı)

2. Confidence

$$\text{Confidence}(X, Y) = \text{Freq}(X, Y) / \text{Freq}(X)$$

(X satın alındığında Y'nin satılması olasılığı)

3. Lift

$$\text{Lift} = \text{Support}(X, Y) / (\text{Support}(X) * \text{Support}(Y))$$

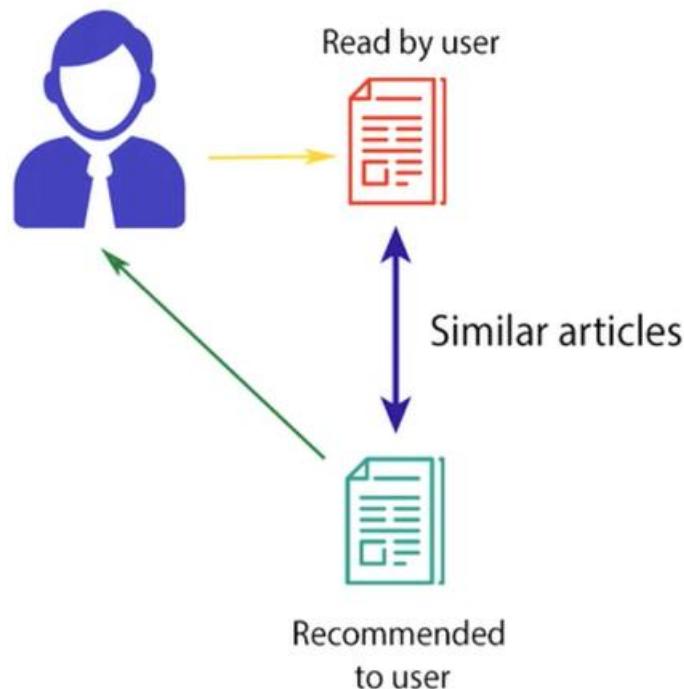
(X satın alındığında Y'nin satın alınma olasılığı lift kat kadar artar)

association_rules fonksiyonundan

- association_rules fonksiyonundan çıkan **Leverage** değeri, Lift'e benzer bir değerdir. Fakat Leverage değeri Support'u yüksek olan değerlere öncelik verme eğilimindedir. Bundan dolayı ufak bir **yanlılığı** vardır.
- **Lift** değeri daha **az sıkıkta** olmasına rağmen bazı ilişkileri yakalayabilmektedir. Dolayısıyla bizim için daha değerli bir metriktir. **Yansızdır**.
- **Conviction** ise Y ürünü olmadan X ürününün beklenen değeridir, frekansıdır veya X olmadan Y ürününün beklenen frekansıdır.
- Pratikte genelde şu kombinasyonlar ile işlem yapılır. Support'u şu değerden büyük olanlar, Confidence şu değerden büyük olan ve Lift değeri şu değerden büyük olanlara gibi birkaç tane olası kombinasyon üzerinden değerlendirmeler yapılabilir.

Content Based Filtering

- Ürün içeriklerinin (meta bilgileri üzerinden) benzerlikleri üzerinden tavsiyeler geliştirir.



- Metinlerden nasıl uzaklık bulunacak? Bunun için metinleri matematiksel ifadelere çevirmek gerekiyor. 5,4, ... Uzaklıği bulmak için izlenecek adımlar:
 - 1. Metinleri Matematiksel Olarak Temsil Et (Metin Vektörleştirme)

	Word 1	Word 2	Word 3	.	.	.	Word n
Movie 1	1	2	0				4
Movie 2	5	1	3				3
Movie 3	3	5	3				2
.							
.							
.							
Movie m	3	5	4				1

- 2. Benzerlikleri Hesapla

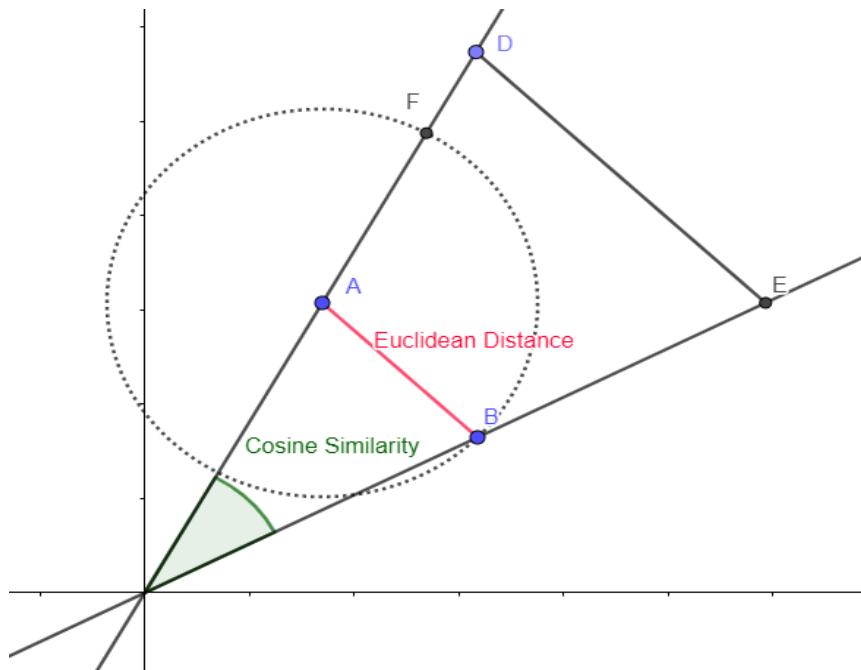
- Peki burada benzerlikleri nasıl hesaplayacağız. **Öklid, Cosine Similiarity** gibi yöntemler kullanıp hesaplayacağız.
 - Öklid uzaklığı: iki vektörün birbirleri arasındaki uzaklığın karelerinin toplamlarının kareköküdür.

$$d(\mathbf{p}, \mathbf{q}) = \sqrt{\sum_{i=1}^n (q_i - p_i)^2}$$

- Cosine Similarity :

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} = \frac{\sum_{i=1}^n A_i B_i}{\sqrt{\sum_{i=1}^n A_i^2} \sqrt{\sum_{i=1}^n B_i^2}}$$

Cosine Similarity A ve B adındaki vektörün arasındaki açıyı hesaplamaya odaklanır. Öklid Uzaklığı ise iki nokta arasındaki uzaklığını hesaplamaktadır. Yani Cosine Similarity iki vektörün benzerliğini ölçmektedir. Öklid Uzaklığı ise uzaklığını ölçmektedir. İkisi de bizim için aynı anlama geliyor.



Metinleri Vektörel Temsilleri

Vektörleştirme işlemini yapmak için yaygın olarak kullanılan yöntemler vardır.

1. Count Vector (Word Count)

2. TF-IDF

Count Vectorizer

- Elimizdeki metinleri nasıl matematik hesabı yapacağımız problemini çözmek için kullanılır.
- Bu yorumları vs. ölçülebilir bir formata getirmeyi amaçlıyoruz.
- Bunun için şu adımları takip edeceğiz:
 1. Eşsiz Tüm Terimleri Sütunlara, Bütün Dokümanları Satırlara Yerleştir.
 2. Terimlerin Dokümanlarda Geçme Frekanslarını Hücrelere Yerleştir.

TF-IDF

- Kelimelerin hem kendi metinlerinde hem de bütün Corpus'ta geçme frekansları üzerinden bir normalizasyon işlemi yapar. Böylece Count Vector'den çıkabilecek olan bazı yanlılıklarını giderir.
 1. Ama bazen o yanlışlık işe yarayabilir. Mesela Brad Pitt'in için önerে çıkar
 2. Bir kelime bir cümlede 30 defa geçmiş olabilir. İşte böyle yüksek sayılar ilgili dokümantasyon (cümle vs.) lehine bazı yanlılıklar ortaya çıkarmaktadır. Bunu ortadan kaldırmak için TF-IDF kullanılır, normalize eder.
- Bunun için şu adımları takip edeceğiz:
 1. Count Vectorizer'ı Hesapla
 2. TF – Term Frequency'i Hesapla
 - T teriminin ilgili dokümantasyon frekansı / dokümandaki toplam terim sayısı
 3. IDF'lerin Hesaplanması (Inverse Document Frequency)
 - $1 + \log_e ((\text{toplam doküman sayısı} + 1) / (\text{ince} t \text{ terim olan doküman sayısı} + 1))$
 4. TF * IDF Hesapla
 5. L2 Normalizasyonu Yap
 - Satırların kareleri toplamının karekökünü bul, ilgili satırlardaki tüm hücreleri bulduğumn değere böl.

Collaborative Filtering

- Topluluğun ortak kanaatlerine göre filtreleme yapmaktadır.
- İşbirlikçi Filtreleme denilince aklımıza gelen 3 önemli başlık vardır.
 1. Item-Based Collaborative Filtering

- 2. User-Based Collaborative Filtering
- 3. Model-Based Collaborative Filtering
- Item-Based ve User-Based karşımıza **Memory-Based**, Model-Based ile **Latent-Based** olarak gelebilir.

Item-Based Collaborative Filtering

- Item benzerliği üzerinden öneriler yapılır. Yani bu Item'a **benzer beğenilme yapısı** gösterenler önerilir.
- Filmlerin beğenme düzeyleri aşağıda verilmiştir. Aşk-ı Memnu ile benzer beğenilme yapısı gösteren film "Movie n". Bu 2 filmin beğenilme alışkanlıkları arasında bir **korelasyon** var gibi gözüküyor.

	Movie 1	Aşk-ı Memnu	Movie 3	.	.	.	Movie n
User 1	1	4	1				4
User 2	5	2	5				3
User 3	3	5	2				5
.							
.							
.							
User m	3	1	4				1

- İçeriğe göre sıralama da çok başarılı idi. Item-Based Collaborative Filtering'de (benzer beğenme korelasyonuna göre) benzer sonuçlar verdi ama farklı film önerileri de verdi. Yani aslında **öneriyi zenginleştirdik** ve arkasına **kocaman bir topluluğun fikir birliğini aldık**.

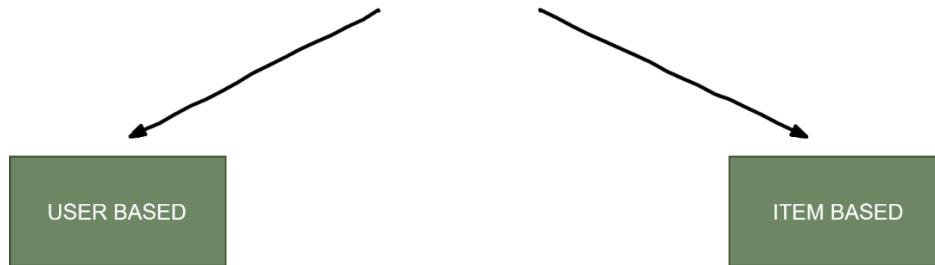
User-Based Collaborative Filtering

- Kullanıcı benzerlikleri üzerinden öneriler yapılır.
- Weighted Rating = corr * rating
 - Korelasyon ve rating'i aynı etkide alıyoruz.
 - Korelasyonları rating'e göre düzeltmiş olduk.

Neighbourhood-Based Collaborative Filtering

- Komşuluk Tabanlı İşbirlikçi Filtreleme (Neighborhood-based Collaborative Filtering) işlemlerinin temel işleyişi aşağıdaki anlayışlardan hangisine dayanır?
 - Benzer kullanıcıların benzer davranışları sergileme eğilimi vardır

NEIGHBOURHOOD BASED COLLABORATIVE FILTERING



Hybird

- İki veya daha fazla tavsiye stratejisinin birbirlerini tamamlayıcı şekilde birlikte kullanılması ile oluşan tavsiye sistemidir.

Model-Based Matrix Factorization

Matrix Factorization

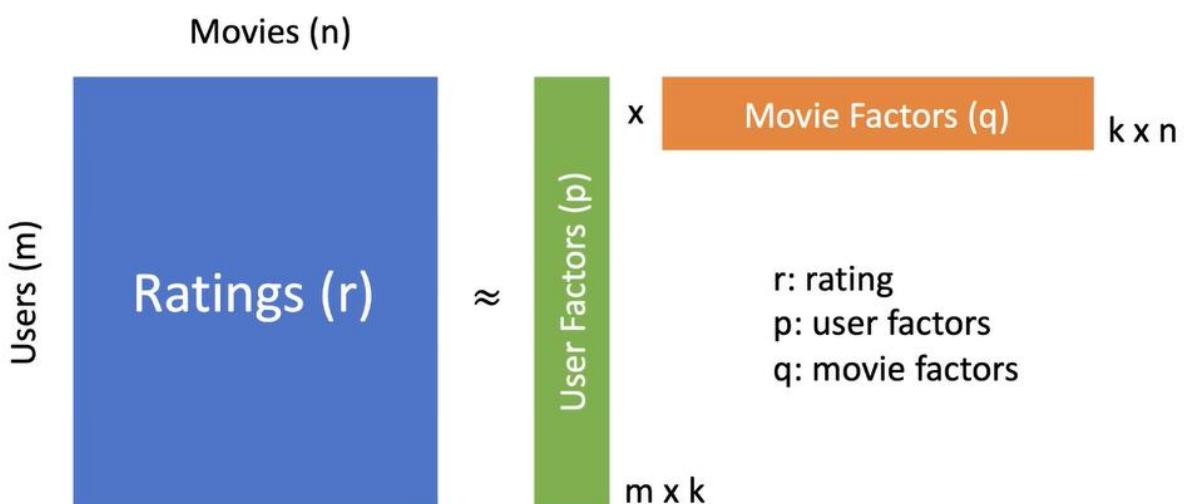
- Tavsiye sistemlerinde Model-Based yöntemlere giriş yapmış bulunuyoruz. Burada probleme daha **bütüncül bir şekilde** ve optimize edilecek bazı problemler varmış gibi yaklaşacağız.
- Problemiz tablodaki boşlukları (NA) doldurmak. Kullanıcı o filmi izlememiş.
 - Daha önceki tavsiyelerde bütüncül olarak değil de çeşitli uzaklık yöntemleri** kullanarak öneri geliştirmeye çalıştık.

	Movie 1	Movie 2	Movie 3	.	.	.	Movie n
User 1	1	2	BLANK				4
User 2	5	BLANK	3				3
User 3	3	5	5				BLANK
.							
.							
.							
User m	BLANK		2		1		1

- Buradaki amacımız boşluklara ne geleceğini tahmin etmek ve doldurmak olacaktır. Öyle bir şey yapalım ki hem bu **kullanıcın puan verme alışkanlıkları** hem de ilgili **filmlerin bazı özelliklerini** kullanarak kullanıcıların filme (blank olanlara) kaç puan vereceğini tahmin edeceğiz.

	Movie 1	Movie 2	Movie 3	.	.	.	Movie n
User 1	1	2	3	.	.	.	4
User 2	5	2	3	.	.	.	3
User 3	3	5	5	.	.	.	1
.				.	.	.	
.				.	.	.	
.				.	.	.	
User m	5	2	1	.	.	.	1

- Boşlukları doldurmak için user'lar ve movie'ler için var olduğu **varsayılan latent feature'ların ağırlıkları** var olan veri üzerinden bulunur ve ağırlıklar ile var olmayan gözlemler için tahmin yapılır.
 - Latent Feature, filmin türü, yönetmen, oyuncu vs olması nedeniyle beğenilmiştir. Yani gizli featuraler olabilir. Görünmeyen ama olabilecek sebepler değerlendiriliyor.
 - Filmde Brad Pitt olduğu için beğenmiştir. Film macera olduğu için beğenmiştir.
- Elimizde bir **User-Item matrisi** var. Bu matrisi 2 tane daha az boyutlu matrise ayırtırılır.
- 2 matristen User-Item matrisine gidişin **latent factor'ler ile gerçekleştiği varsayımda** bulunulur.
- Dolu olan gözlemler üzerinden latent factors (gizli faktör) ağırlıkları bulunur.
- Bulunan ağırlıklar ile boş olan gözlemler doldurulur.



- Rating matrisinin iki faktör matrisin çarpımı (dot product) ile oluşturduğu varsayıılır.

- Kullanıcı ve filmlerin latent features için skorlara sahip olduğu düşünülür.
- Bu ağırlıklar önce veri üzerinden bulunur ve daha sonra boş olan veriler bu ağırlıklar ile doldurulur.

Ratings (r)							p (user factors)					q (movie factors)				
	M1	M2	M3	M4			F1	F2	x	F1	M1	M2	M3	M4		
U1	1	2		4			U1	p_{11}	p_{12}	F1	q_{11}	q_{12}	q_{13}	q_{14}		
U2	5		3	3			U2	p_{21}	p_{22}	F2	q_{21}	q_{22}	q_{23}	q_{24}	$k \times n$	
U3	3	5	5				U3	p_{31}	p_{32}							
U4		2	1	1			U4	p_{41}	p_{42}							

\approx

$m \times n$

$m \times k$

$$\begin{aligned}
 r_{11} &= p_{11} * q_{11} + p_{12} * q_{21} & r_{21} &= p_{21} * q_{11} + p_{22} * q_{21} & r_{33} &= p_{31} * q_{13} + p_{32} * q_{23} \\
 1 &= p_{11} * q_{11} + p_{12} * q_{21} & 5 &= p_{21} * q_{11} + p_{22} * q_{21} & 5 &= p_{31} * q_{13} + p_{32} * q_{23}
 \end{aligned}$$

Peki p ve q değerleri nasıl bulunacak?

- Var olan değerler üzerinden iteratif şekilde tüm p ve q'lar bulunur ve sonra kullanılır.
- **Başlangıçta rastgele p ve q değerleri** ile rating matrisindeki değerler tahmin edilmeye çalışılır.
- Her iterasyonda hatalı tahminler düzeltilerek rating matrisindeki değerlere yaklaşılmasına çalışılır.
- Böylece belirli bir iterasyon sonucunda p ve q matrisleri doldurulmuş olur.

Ratings (r)							p (user factors)					q (movie factors)				
	M1	M2	M3	M4			F1	F2	x	F1	M1	M2	M3	M4		
U1	1	2		4			U1	0.8	0.5	F1	0.5	2.3	0.1	4.2		
U2	5		3	3			U2	1.2	3.2	F2	0.05	1.6	4.2	1.2	$k \times n$	
U3	3	5	5				U3	1.3	5.2							
U4		2	1	1			U4	0.2	0.1							

\approx

$m \times n$

$m \times k$

Ratings (r)									
	M1	M2	M3	M4					
U1	1	2	2,18	4					
U2	5	?	3	3					
U3	3	5	5	?					
U4	?	2	1	1					

$\hat{r}(1,3) = p_{11} * q_{13} + p_{12} * q_{23}$

$2,18 = 0.8 * 0.1 + 0.5 * 4.2$

$$\min_{q^*, p^*} \sum_{(u,i) \in \kappa} (r_{ui} - q_i^T p_u)^2 + \lambda (\|q_i\|^2 + \|p_u\|^2)$$

Rating associated with user u , item i
 Regularization parameter
 User vector u
 Item vector i
 Magnitude of item vector q
 Magnitude of user vector p

Find the minimum matrices q, p
 Loop over pairs of users and items

Gradient Descent

- Fonksiyon minimizasyonu için kullanılan bir optimizasyon tekniğidir.
- Gradyanın negatif olarak tanımlanan “en dik iniş” yönünde iteratif olarak parametre değerlerini güncelleyerek ilgili fonksiyonun minimum değerini verecek parametreleri bulur.
- **Bir fonksiyonun o noktadaki türevi o fonksiyonun maksimum artış yönünü verir.**

Repeat until convergence {

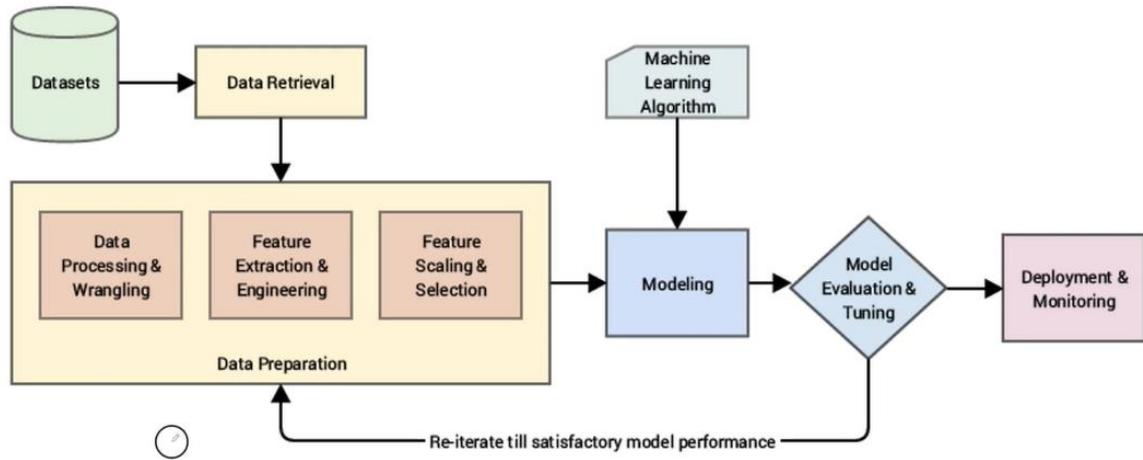
$$\theta_j \leftarrow \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

}

$$\begin{aligned}
 p_{ki}^{t+1} &= p_{ki}^t + 2\gamma(r_{ij} - p_i^t q_j^t) q_{kj}^t \\
 q_{kj}^{t+1} &= q_{kj}^t + 2\gamma(r_{ij} - p_i^t q_j^t) p_{ki}^t
 \end{aligned}$$

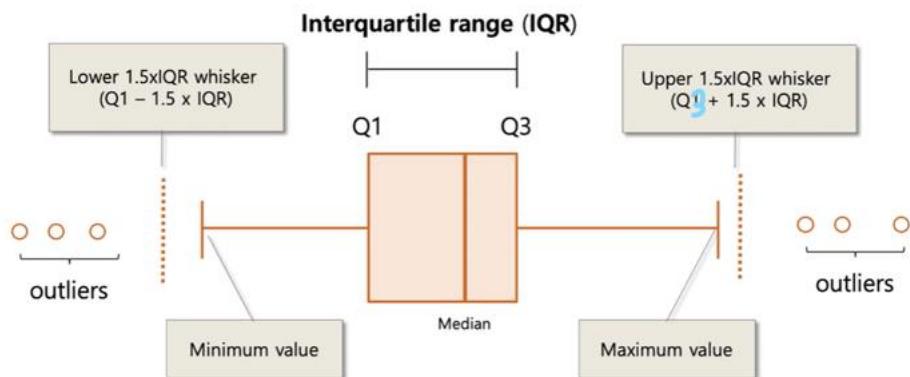
Feature Engineering

- Özellik Mühendisliği: Özellikler üzerinde gerçekleştirilen çalışmalardır. Ham veriden değişken üretmek.
- Veri Ön İşleme: Çalışmalar öncesi verinin uygun hale getirilmesi sürecidir.



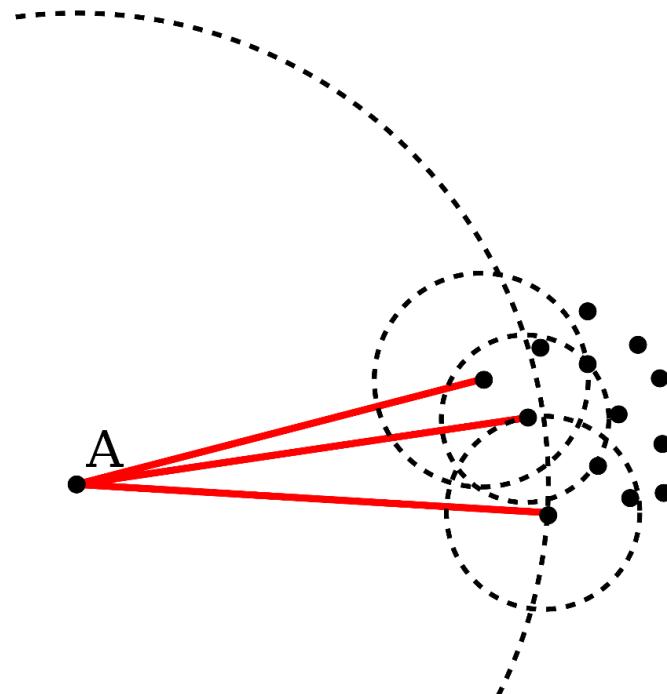
Outliers:

- Verideki genel eğilimin oldukça dışına çıkan değerlere aykırı değer denir.
- Özellikle **doğrusal problemlerde** aykırı değerlerin etkileri daha şiddetlidir. Ağaç yöntemlerinde daha düşüktür.
- Aykırı değerler nasıl belirlenir?
 - Sektör Bilgisi
 - Standart Sapma Yaklaşımı
 - Z-Skoru Yaklaşımı
 - Boxplot (IQR) Yöntemi – Tek Değişkenli Olarak
 - LOF (Local Outlier Factor) – Çok Değişkenli Olarak

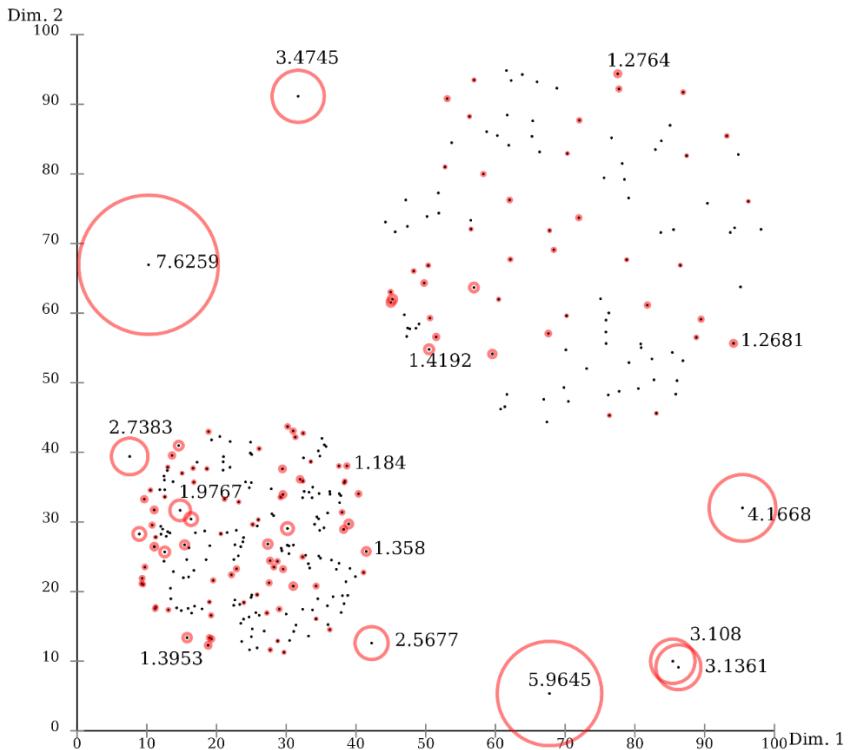


LOF =

- Tek başına aykırı olamayacak değerler birlikte ele alınınca aykırılık yaratıyor olabilir.
(17 yaşında 3 kere evlenen biri)
- LOF yöntemi ne yapar?
 - Gözlemleri bulundukları konumda yoğunluk tabanlı skorlayarak buna göre bir aykırı olabilecek değerleri tanıma imkanı sağlar.
 - Bir noktanın lokal yoğunluğu demek ilgili noktanın etrafındaki komşuluklar demektir. Eğer bir nokta komşularının yoğunluğundan anlamlı bir şekilde düşük ise bu nokta daha seyrek bir bölümdedir. Aykırı değer yorumu yapılabilir.



- LOF yöntemi bir skor verir ve bu skor 1'e ne kadar yakınsa o kadar iyi skor verir der. 1'den uzaklaştıkça ilgili gözlem artık **outlier** olur. LOF bir threshold belirleyerek üstünü outlier olarak ele alabilme imkanı verir.



Missing Values

- Gözlemlerin eksik olmasını ifade etmektedir.

Araç Fiyatı	KM	Vites Türü	Hasar Durumu	Marka	Model
10000	300000	Manuel	Evet	A	A1
54000	40000	Manuel	Hayır	A	A2
46999	NA	Manuel	Evet	B	B1
89000	1000000	Otomatik	Evet	C	C1
70000	78000	Otomatik	Evet	B	B2
50000	30000	Manuel	Hayır	C	C2
NA	600000	Manuel	Hayır	C	C3
68900	50000	Otomatik	Hayır	B	B2
12000	200000	Manuel	Hayır	A	A1

- Eksik veri problemi nasıl çözülür?
 - Silme
 - Değer Atama Yöntemleri
 - Tahmine Dayalı Yöntemler
- Eksik veri ile çalışırken **göz önünde bulundurulması gereken bir konu** vardır:
 - Eksik verinin **rassallığı** (Eksikliğin rastgele ortaya çıkıp çıkmadığı bilinmesi gereği)

- Yoksa belirli bir bağımlılıktan mı ortaya olmuş. Tablo birleştirmesi vs.

Encoding

- **Target Encoding** modelin overfit olmasına (aşırı öğrenme durumu) neden olduğu söylenebilir (Target Leaking).
- Değişkenlerin temsil şekillerinin değiştirilmesi

Label Encoding:

- Eğer değişkenin 2 sınıfı varsa karşımıza **Binary Encoding** olarak da gelebilir.

Original Data		Label Encoded Data	
Team	Points	Team	Points
A	25	0	25
A	12	0	12
B	15	1	15
B	14	1	14
B	19	1	19
B	23	1	23
C	25	2	25
C	29	2	29

One Hot Encoding:

- Label Encoding ile oluşturabilecek ölçüm problemini One-Hot-Encoding ile aşabiliriz.
- Burada bir dummy değişken durumu söz konusu. Drop_first=True diyerek ilk sınıfı drop ederek bu tuzaktan kurtuluruz.

Original Data		One-Hot Encoded Data			
Team	Points	Team_A	Team_B	Team_C	Points
A	25	1	0	0	25
A	12	1	0	0	12
B	15	0	1	0	15
B	14	0	1	0	14
B	19	0	1	0	19
B	23	0	1	0	23
C	25	0	0	1	25
C	29	0	0	1	29

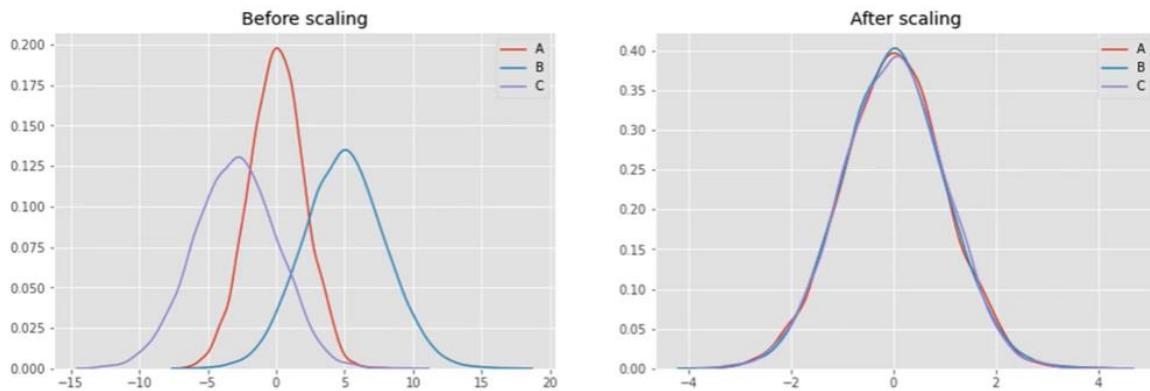
Rare Encoding:

- Veri setindeki bir kategorik değişkenin sınıflarındaki az gözlemlenen değerleri kendi belirlediğimiz belirli bir eşik değerine göre bir araya getirmek.

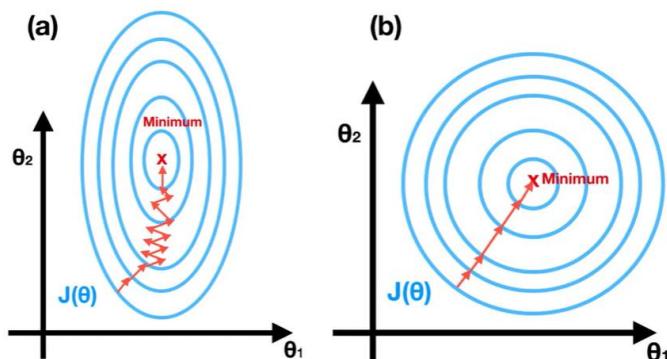
CITY	CITY_COUNT	CITY	CITY_COUNT	CITY	CITY_COUNT
A	56	A	56	A	56
B	84	B	84	B	84
C	54	C	54	C	54
D	2	D	2	F	60
E	12	E	12	G	3
F	60	F	60	H	5
G	3	G	3	K	25
H	5	H	5	M	36
K	25	K	25	Z	45
L	1	L	1	RARE (D,E,G,H,L)	
M	36	M	36	23	
Z	45	Z	45		

Feature Scaling:

1. Tüm değişkenleri eşit şartlar altında değerlendirebilmek adına ölçeklendirme yapılır.
 - o Değişkenler arasındaki ölçüm farklılıklarını gidermektedir. **Yanlılık** giderilmiş olur.



2. Özellikle Gradient Descent kullanan algoritmaların **eğitim sürelerini kısaltmasıdır**.



3. Uzaklık temelli yöntemlerde büyük değerlere sahip değişkenler dominantlık sergilemektedir. Yani ezicilik sergilemektedir.
4. Ağaç algoritmaların çoğu etkilenmez.

StandardScaler: Klasik standartlaştırma. Ortalamayı çıkar, standart sapmaya böl. $z = (x - \mu) / s$

RobustScaler: Medyanı çıkar iqr'a böl.

Standart Scaler'a göre aykırı değerlere karşı dayanıklıdır.

MinMaxScaler: Verilen 2 değer arasında değişken dönüşümü

Feature Engineering:

- Ham veriden değişken üretmek

Timestamp	Year	Month	Day	Hour	Day Name
2021-02-05 07:45	2021	2	5	7	Friday
2021-02-04 21:05	2021	2	4	21	Thursday
2019-05-17 09:51	2019	5	17	9	Friday
2019-05-16 21:27	2019	5	16	21	Thursday
2019-05-16 13:40	2019	5	16	13	Thursday

1. Binary Features
2. Text Features
3. Regex Features
4. Date Features
5. Feature Interaction (değişkenlerin birbiri ile etkileşime girmesi, çarpım vs.)

Encoding Vs Scaling:

- Ölçeklendirme dağılım yapısını (varyans) koruyarak temsil şeklini değiştirmek.
- Yaş değişkenini encode etmek ile scale etmek arasındaki fark nedir?
 - Sayısal değişken çeşitliliği yüksek (varyansı yüksek) değişken demektir. Eğer bunu 0-1 arasına dönüştürürsek verinin dağılımına ilişkin bilgi kaybı olur (varyans, std vs kaybolur.).
 - Kategorik de ise değişkeni bozarsın (sadece count alınabilir). **Yani zenginliği kırpılmış oluyoruz.**

Machine Learning

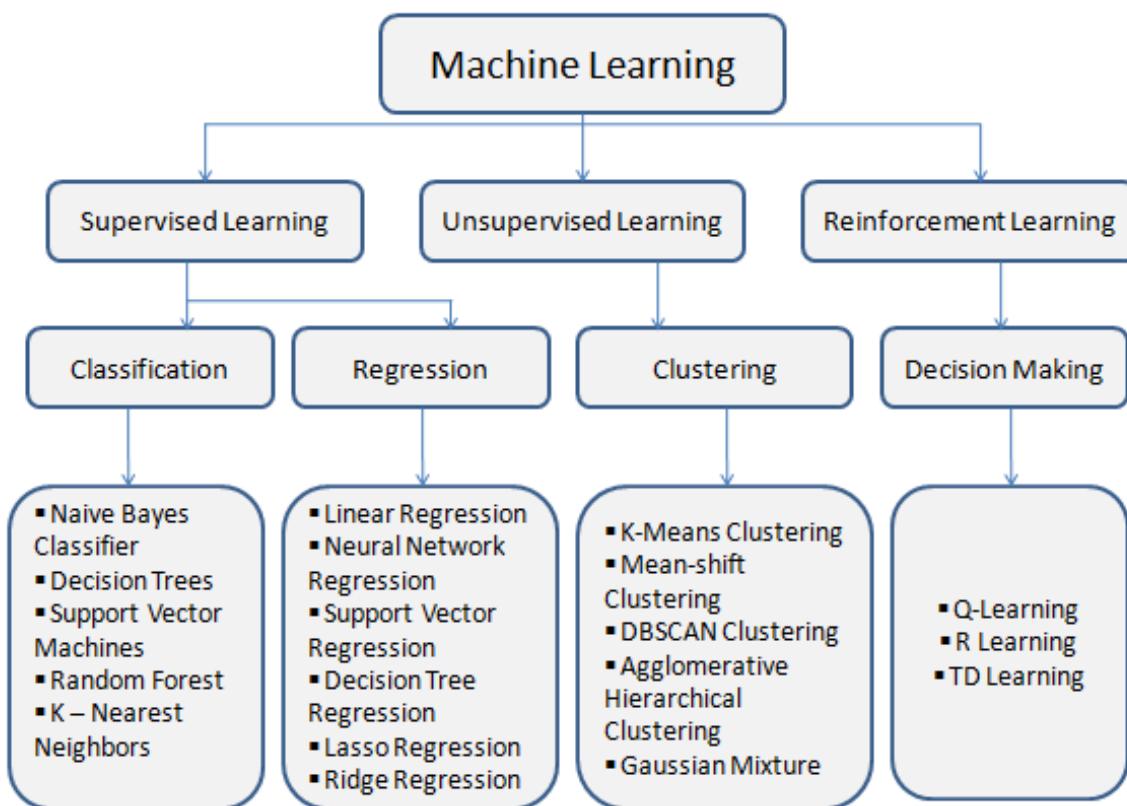
- Makine Öğrenmesi, bilgisayarın insanlara benzer şekilde öğrenmesini sağlamak maksadıyla çeşitli algoritma ve tekniklerin geliştirilmesi için çalışılan bilimsel çalışma alanıdır.
- Bağımlı değişkeni, hedef değişken, sayısal olan problemlere **regresyon problemi** denir.

Temel Kavramlar

Değişken Tipleri

- Sayısal Değişkenler
 - Kesikli ya da sürekli olabilir.
- Kategorik Değişkenler (Nominal, Ordinal)
 - Nominalde sınıflar arası fark yoktur, Kadın-Erkek.
 - Ordinal de ise sınıflar arası fark vardır, Eğitim durumu (İlkokul, lise...)
- Bağımlı Değişkenler (target, dependent, output, response)
- Bağımsız Değişken (Feature, independent, input, column, predictor, explanatory)

Öğrenme Türleri



- Eğer veri setinde label'larımız yer alıysa bu durumda Gözetimli Öğrenme yapılır.
- Eğer ilgili veri setinde label'lar yoksa Denetimsiz Öğrenme yapılır.

Model Başarı Değerlendirme Yöntemleri

- Regresyon Modellerinde Başarı Değerlendirme Yöntemleri

$$MSE = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2}$$

$$MAE = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i|$$

- Sınıflandırma Modellerinde Başarı Değerlendirme Yöntemleri

$$\text{Accuracy} = \frac{\text{Doğru Sınıflandırma Sayısı}}{\text{Toplam Sınıflandırılan Gözlem Sayısı}}$$

- MSE ne kadar düşükse o kadar iyi, Accuracy ne kadar yüksekse o kadar iyi

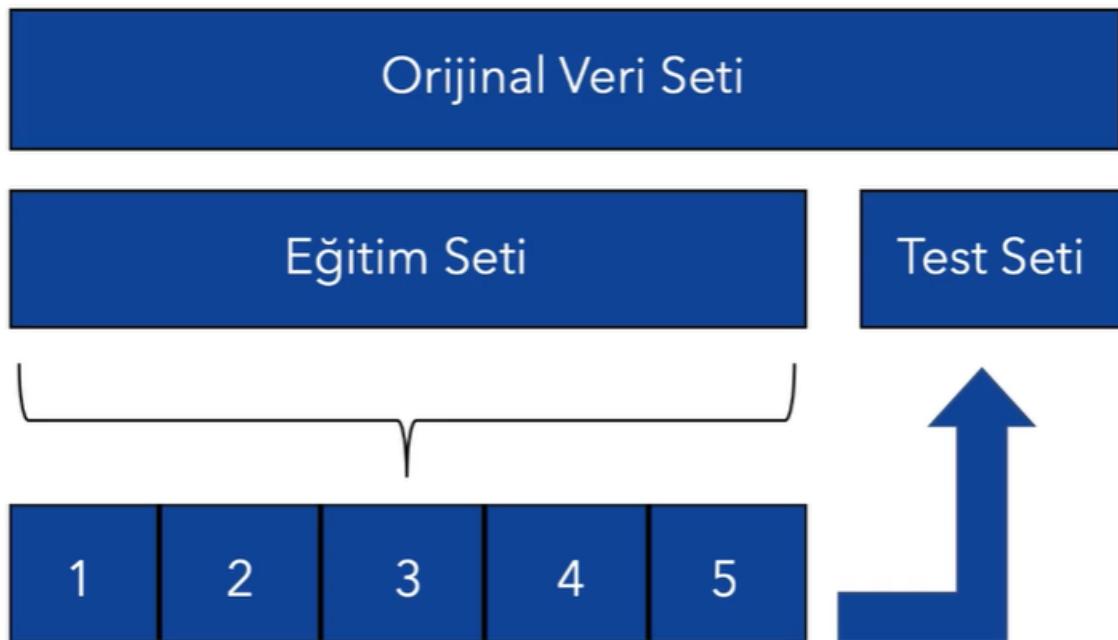
Model Doğrulama Yöntemleri

Orijinal Veri Seti

Eğitim Seti

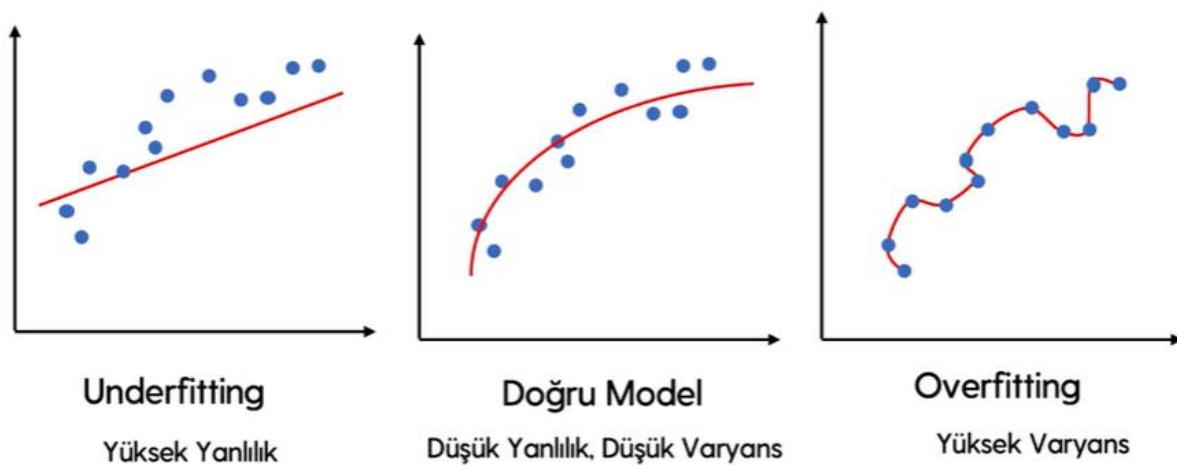
Test Seti

K Fold Cross Validation

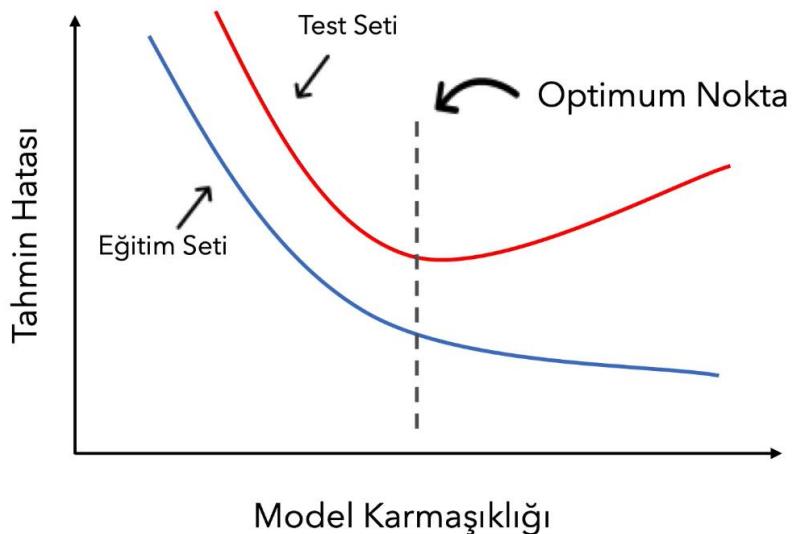


- 2 şekilde kullanılabilir.
 - Orijinal veri setinin 5'e bölündüğünü düşünelim. 4 parça ile eğit diğer ile test et. Bu kombinasyonu hep tekrar eder.
 - Diğer yöntem ise eğitim setinin 5'e bölünüp test edilmesidir. En son hiç görmediği bir test seti ile test edilir.

Bias-Variance Tradeoff



- Aşağıdaki grafikte iki hatanın birbirinden ayrılmaya başladığı, çatallanmanın başladığı nokta itibarı ile aşırı öğrenme başlamıştır denir.



Linear Regression

- Bağımlı ve bağımsız değişkenler arasındaki ilişkiyi doğrusal olarak modellemektedir.

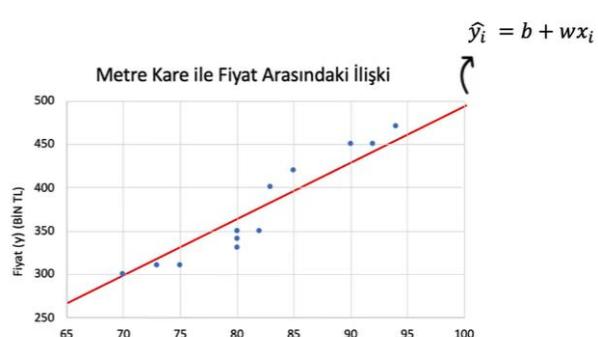
$$\hat{y}_i = b + w x_i$$

$$\hat{y}_i = b + w_1 x_1 + w_2 x_2 + w_3 x_3 + \dots + w_p x_p$$

Ağırlıkların Bulunması

- Gerçek değerler ile tahmin edilen değerler arasındaki farkların karelerinin toplamını/ortalamasını minimum yapabilecek b ve w değerleri bulunarak ağırlıkları bulabiliriz.
- Burada Cost ifadesine MSE diyebiliriz.

metre_kare (x_i)	fiyat (y_i)
70	300
73	310
75	310
80	330
80	340
80	350
82	350
83	400
85	420
90	450
92	450
94	470



$$Cost(b, w) = \frac{1}{2m} \sum_{i=1}^m ((b + w x_i) - y_i)^2$$

Regresyon Modellerinde Başarı Değerlendirme

$$MSE = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

Gerçek Değerler Tahmin Edilen Değerler

Gözlem Sayısı

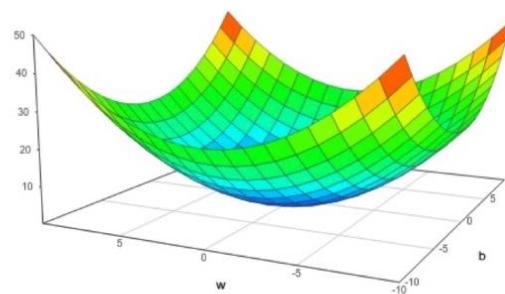
- MSE'de kareden dolayı bir şişme meydana geliyor. O yüzden RMSE kök alarak bunu daha doğru değerlendirmeye çalışır.

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2}$$

$$MAE = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i|$$

Parametrelerin Tahmin Edilmesi (Ağırlıkların Bulunması)

$$Cost(b, w) = \frac{1}{2m} \sum_{i=1}^m ((b + wx_i) - y_i)^2$$



- Ağırlıkları bulmanın 2 tane çözüm yolu vardır:
 - Analitik Çözüm: Normal Denklemler Yöntemi (En Küçük Kareler Yöntemi)
 - Neden Normal Denklemler Yöntemi yerine Gradient Descent kullanalım?
MLP'de tersi (-1) alınma işlemi neden yapılır?
 - Gözlem sayısı ve değişken sayısı çok fazla arttığında matrisin tersini alma işlemi zorlaşmakta ve Gradient Descent gibi optimizasyona dayalı yöntemlerin ihtiyacı kendisini hissettirir.

Simple Linear Regression

$$SSE = \sum_{i=1}^n (y_i - \hat{y}_i)^2 = \sum_{i=1}^n (y_i - (b_0 + b_1 x_i))^2$$

$$b_1 = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2}$$

$$b_0 = \bar{y} - \hat{b}_1 \bar{x}$$

Multiple Linear Regression

$$SSE = \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

$$\hat{\beta} = (X^T \cdot X)^{-1} X^T \cdot Y$$

- Optimizasyon Çözümü: Gradient Descent

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^m (h_\theta(x^{(i)}) - y^{(i)})$$

$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^m (h_\theta(x^{(i)}) - y^{(i)}) \cdot x^{(i)}$$

Doğrusal Regresyon için Gradient Descent:

- Gradient Descent bir optimizasyon yöntemidir.
- Gradyanın negatifi olarak tanımlanan “en dik iniş” yönünde iteratif olarak parametre değerlerini güncelleyerek ilgili fonksiyonun minimum değerini verebilecek parametreleri bulur.
- Amacı herhangi türevlenebilir bir fonksiyonu minimum yapabilecek parametre değerlerini bulmaktır.

- İlgili fonksiyonun kısmi türevini alır ve gradyanın negatifine doğru giderek parametrenin eski değerinde değişiklik yaparak her iterasyonda hatanın azalmasını sağlar.

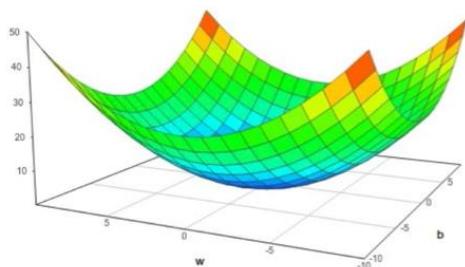
Repeat until convergence {

$$\theta_j \leftarrow \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

}

Makine Öğrenmesi için Gradient Descent:

- Cost fonksiyonunu minimize edebilecek parametreleri bulmak için kullanılır.



$$Cost(b, w) = \frac{1}{2m} \sum_{i=1}^m ((b + wx_i) - y_i)^2$$

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^m (h_\theta(x^{(i)}) - y^{(i)})^2$$

$$\underset{\theta_0, \theta_1}{\text{minimize}} J(\theta_0, \theta_1)$$

- Değişken Sayısı arttıkça R^2 şismeye meyillidir.

Logistic Regression

- Amaç **sınıflandırma** problemi için bağımlı ve bağımsız değişkenler arasındaki ilişkiyi **doğrusal** olarak modellemektir. (y_{hat} **sigmoid** fonksiyonudur. Her sınıf için olasılık verir)

$$\hat{y}_i = \frac{1}{1 + e^{-(z)}}$$

$$z = b + w_1x_1 + w_2x_2 + w_3x_3 + \dots + w_px_p$$

- Bir optimizasyon problemi olarak göze alındığında nasıl çözülür?
 - Gerçek değerler ile tahmin edilen değerler arasındaki farklara ilişkin **log loss** değerini minimum yapabilecek ağırlıkları bularak.

$$\hat{y}_i = \frac{1}{1 + e^{-(z)}}$$

$$z = b + w_1x_1 + w_2x_2 + w_3x_3 + \dots + w_px_p$$

$$\text{Log Loss} = \frac{1}{m} \left(\sum_{i=1}^m -y_i * \log(p(\hat{y}_i)) - (1 - y_i) \log(1 - p(\hat{y}_i)) \right)$$

Lojistik Regresyon için Gradient Descent

- Bu cross entropy formülüdür ($J(\Theta)$).
 - Entropi ne kadar yüksekse çeşitlilik o kadar fazladır.
 - O yüzden gerçek değer ile tahmin edilen değer açısından, entropinin az olmasını çeşitliliğin az olmasını isteriz.
 - Gerçek değer ile gerçek değerlerin gerçekleşmesi olasılıkları ifadeleri birbirine ne kadar yakınsa loss değeri o kadar küçük olur.

$$J(\theta) = \frac{1}{m} \sum_{i=1}^m Cost(h_\theta(x^{(i)}), y^{(i)})$$

$$J(\theta) = \frac{1}{m} \left[\sum_{i=1}^m -y^{(i)} \log(h_\theta(x^{(i)})) + (1 - y^{(i)}) \log(1 - h_\theta(x^{(i)})) \right]$$

```

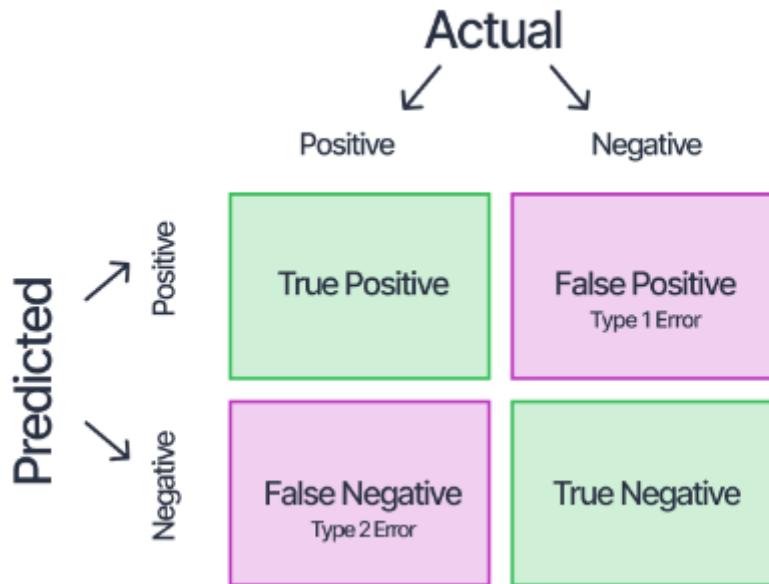
Repeat {
     $\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$ 
}

```

- Bu bir adımdı. Gradient Descent olması için bunun tekrar edilmesi gerekiyor. Türevlenebilen bir fonksiyonun kısmi türevini alarak belirli bir learning rate ile işleme alınıp eski değerden çıkarılarak parametre güncellenir ve hatanın düşmesi beklenir.

Sınıflandırma Problemlerinde Başarı Değerlendirme

- Eğer veri setinde sınıflar dengeli ise Accuracy'ı kullanabiliriz. Aksi halde dengesizse kullanamayız. Recall ve Precision'a bakmak gereklidir.



- Accuracy:** Doğru sınıflandırma oranıdır.

$$\frac{(TP + TN)}{(TP + TN + FP + FN)}$$

- Precision:** Pozitif sınıf tahminlerinin başarı oranıdır.

$$\frac{TP}{(TP + FP)}$$

- Recall:** Pozitif sınıfın doğru tahmin edilme oranıdır.

$$\frac{TP}{(TP + FN)}$$

- F1 Score:**

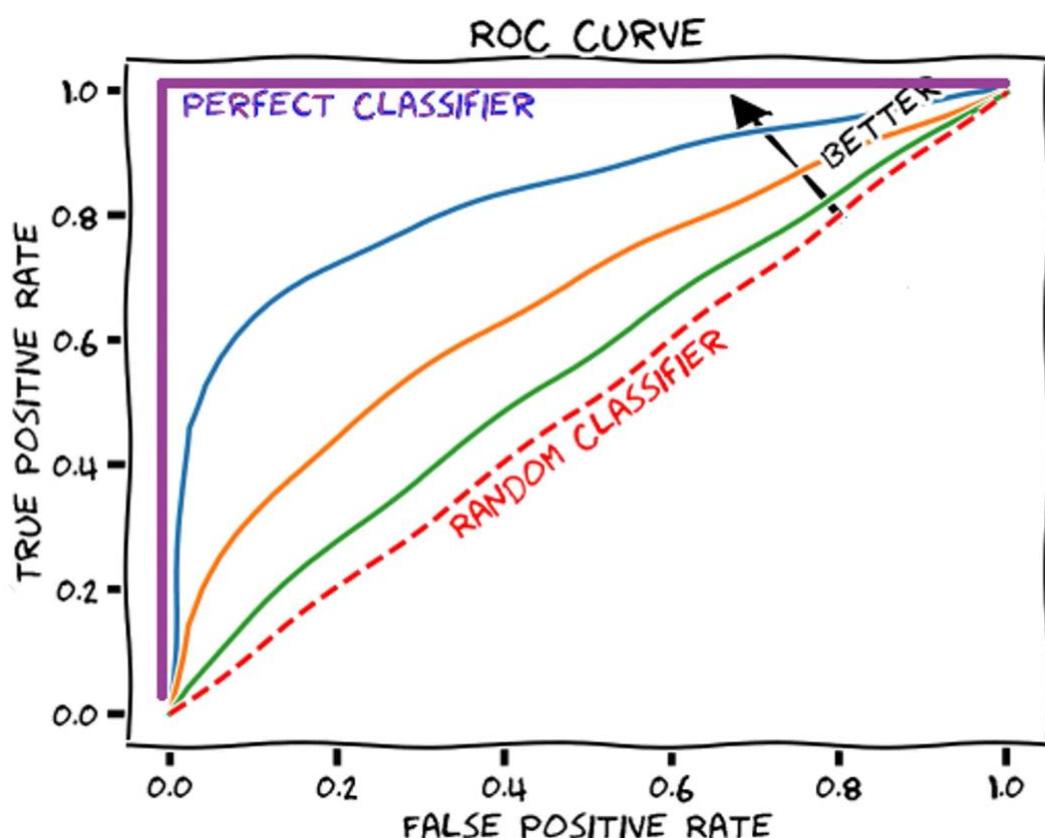
$$\frac{2 * (Precision * Recall)}{(Precision + Recall)}$$

Classification Threshold:

- Belirlenen eşik değere göre metriklerimizin sonuçları değişmektedir. O halde öyle bir şey yapalım ki olası bütün eşik değer değişimleri de göz önünde bulundurarak hesaplama yapılsın ve buna göre bir matematiksel ifade bulalım. Bunun için **ROC Curve** yöntemi kullanılır.

ROC Curve

- Eğer hiçbir model kurulmasaydı ve rastgele 1-0 atamaları yapılsaydı elde edeceğimiz başarı kırmızı kısa çizgi olacaktır. Bu temel karşılaştırma noktasıdır.
- Kırmızıdan yukarı çıkıldıkça, kapladıkları alan arttıkça başarı alanı arttığı düşünülür.
- Bu eğrilerin altında kalan alanın **integrali alınırsa Area Under Curve (AUC)** metriği elde edilir.



Area Under Curve (AUC):

- ROC eğrisinin sayısal bir şekilde ifade edilmesidir.
- AUC, tüm olası sınıflandırma eşikleri için toplu bir performans ölçüsüdür.

- O halde **ilk dikkat** edilmesi gereken şey **sınıf dağılımlarının** eşit olup olmadığıının kontrolüdür. Eğer veri seti dengesizde Recall, Precision, F1 Score'a bakıyoruz. AUC de bakılır.

LOG Loss

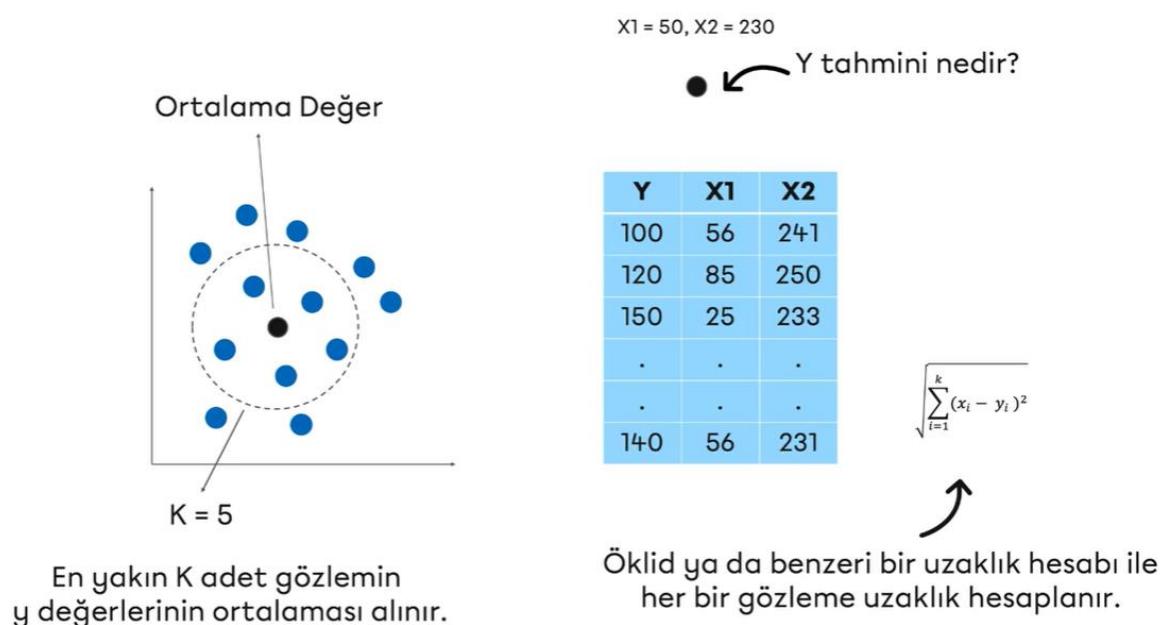
$$\text{Log Loss} = \frac{1}{m} \left(\sum_{i=1}^m -y_i * \log(p(\hat{y}_i)) - (1 - y_i) \log(1 - p(\hat{y}_i)) \right)$$

- Log Loss bir başarı metriğidir. Hem modelin başarısı için hem de hiper parametreleri vs bulmak adına optimize etmek için odaklanacağımız fonksiyonudur.
- Cross Entropy metriğidir.

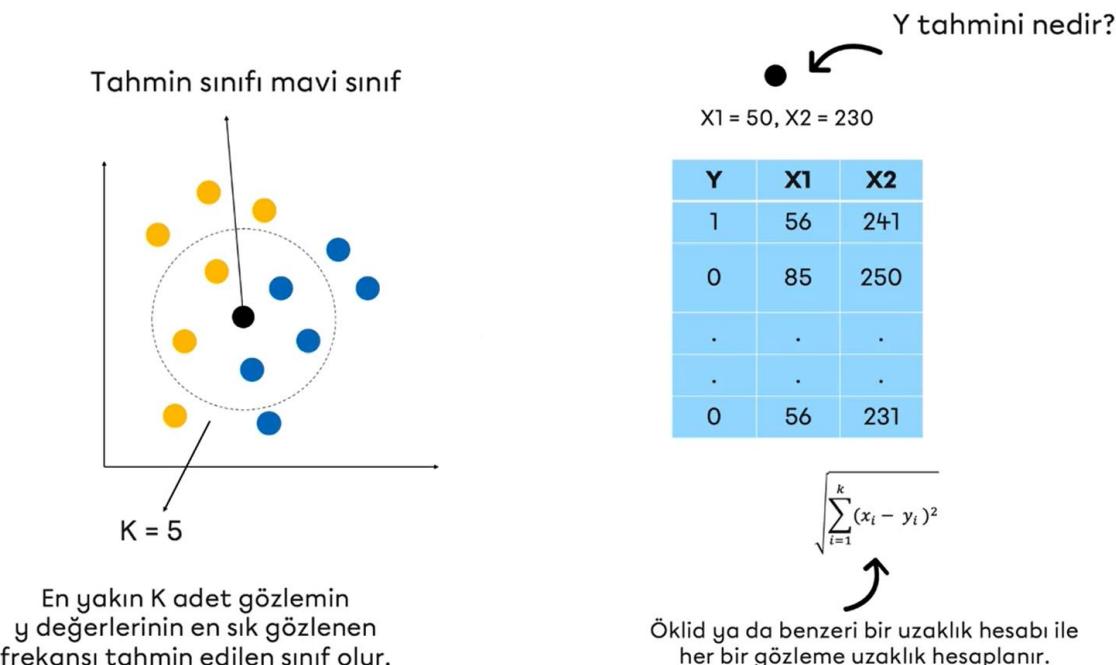
KNN

- Gözlemlerin birbirine olan benzerlikleri üzerinden tahmin yapılır.
- Öklid ya da benzeri bir uzaklık hesabı ile her bir gözleme uzaklık hesabı yapılır.

KNN Regresyon



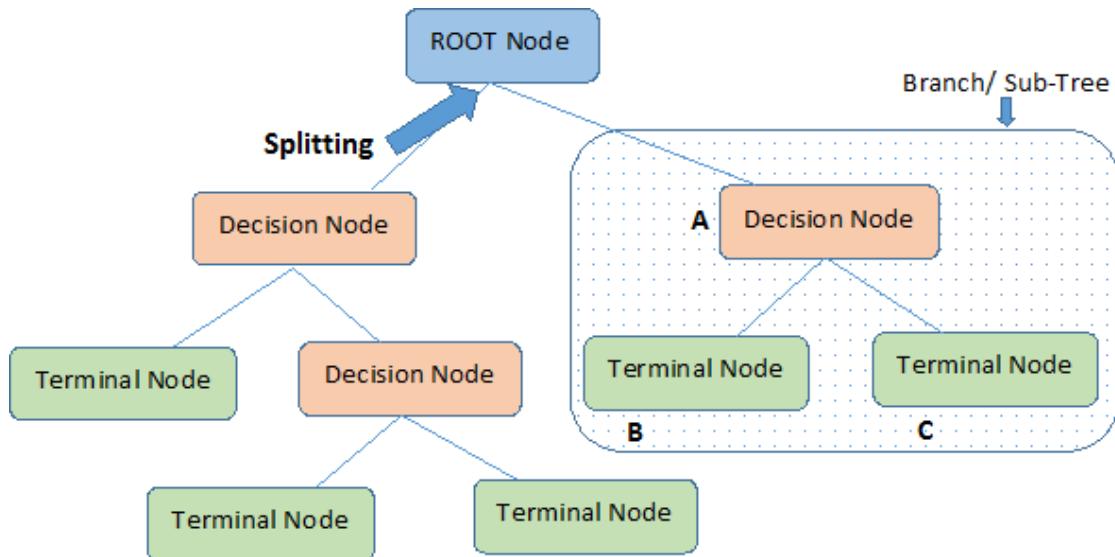
KNN Sınıflandırma



CART (Classification and Regression Tree)



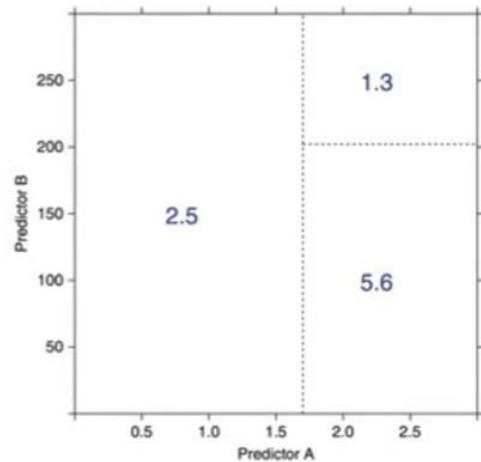
- 1984 yılında Leo Breiman tarafından ortaya konmuştur.
- **Random Forest'in temelini oluşturur.**
- CART'ın amacı veri setindeki karmaşık yapıları basit karar yapılarına dönüştürmektir.
- Heterojen veri setleri belirlenmiş bir hedef değişikene göre homojen alt gruplara ayrılır.
- Bir karar ağacında en tepedeki değişken en önemli (en düşük hatayı veren) değişkendir.
 - **Dallara ayırmada en fazla kullanılan değişken olabilir.**



Note:- A is parent node of B and C.

- Terminal Node, dallanmalar sonrası kalan gözlem birimlerinin **ortalaması**, logaritmasıdır.
- Karar ağacı kullanıldığı bu algoritmalar bize **bir karar kuralı çıkarır**.

```
if Predictor A >= 1.7 then
|   if Predictor B >= 202.1 then Outcome = 1.3
|   else Outcome = 5.6
else Outcome = 2.5
```



CART Regression için Cost/Loss Fonksiyonu

$$\text{RSS (SSE): } \sum_{j=1}^J \sum_{i \in R_j} (y_i - \hat{y}_{R_j})^2$$

R_j : bölge/yaprak/kutu

- Bölme noktaları değişikçe hata da değişir. Hataya göre bölme noktasına karar verebiliriz.
- CART'larda karar vermemiz gereken iki temel husus (Overfit için):
 - 1. Ne kadar dallanma olacak? Ne kadar derine inilecek? `max_depth`
 - 2. Son dalda 3 değer kaldı. Bölmeye devam edilecek mi? `min_sample_split`
- Ağaç yöntemleri çok başarılıdır. Overfit olmaya aşırı meyillidir. Dallara ayıra ayıra en dibe kadar inerler ve çok iyi öğrenirler. Bunun **dezavantajı** genellenebilirlik kabiliyetini kaybetmektir.

CART Classification için Cost/Loss Fonksiyonu

- Sınıflandırma için kullanılan fonksiyonlara genelde **Saflik Ölçümleri (Benzerlik/Benzemezlik)** denir.
- **Gini** ve **Entropy**'i kullanabiliriz. Ne kadar düşük o kadar iyidir.
 - Entropy çeşitliliktir. O yüzden ne kadar az o kadar iyi.

$$\mathcal{L}_G(\mathcal{N}_m) = \sum_{k=1}^K \hat{p}_{mk}(1 - \hat{p}_{mk}),$$

$$\mathcal{L}_E(\mathcal{N}_m) = - \sum_{k=1}^K \hat{p}_{mk} \log \hat{p}_{mk}.$$

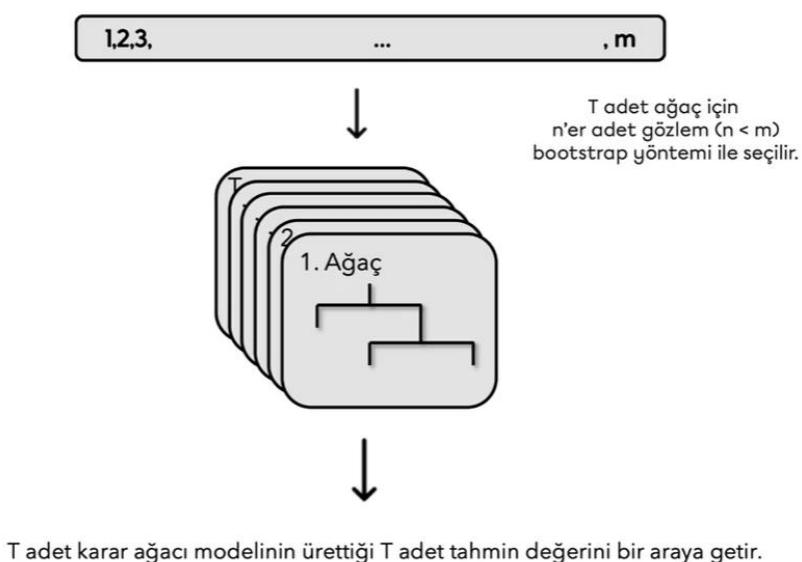
The weighted loss: $\mathcal{L}(S_m) = f_L \cdot \mathcal{L}(\mathcal{C}_m^L) + f_R \cdot \mathcal{L}(\mathcal{C}_m^R).$

Gelişmiş Ağaç Yöntemleri

Random Forest



- 2001 yılında Leo Breiman tarafından ortaya konmuştur.
- Temeli birden çok karar ağacının ürettiği tahminlerin bir araya getirilerek değerlendirilmesine dayanır.
- **Bagging** (Breiman 1996) ve **Random Subspace** (Ho 1998) yöntemlerinin birleşimi ile oluşturulmuştur.
- Ağaçlar için gözlemler bootstrap rastgele örnek seçimi yöntemi ile **değişkenler random subspace yöntemi ile seçilir**.
- Karar ağacının her bir düğümünde en iyi dallara ayırcı (bilgi kazancı) değişken tüm değişkenler arasından rastgele seçilen daha az sayıdaki değişken arasından seçilir.
- Random Forest neden Random ifadesi var? Rastgeleliği nereden alıyor?
 - 1. Gözlem birimlerinden rastgele seçerek n tane ağaç oluşturur.
 - 2. Oluşturduğu ağaçlarda bölmeye işlemlerine başlamadan önce değişkenlerden rastgele değişkenler (daha az 100'den 20 mesela) seçer.
 - Yani değişken seçimi ve gözlem seçimi ile rassallığı korumuş olur.
- Ağaçları oluşturmada veri setinin $2/3$ 'ü kullanılır. Dışarıda kalan veri, ağaçların performans değerlendirmesi ve değişken önemini belirlenmesi için kullanılır.
- Her düğüm noktasında rastgele değişken seçimi yapılır.
 - Regresyonda $p/3$
 - Sınıflandırmada karekök p
- Bagging Yöntemi:



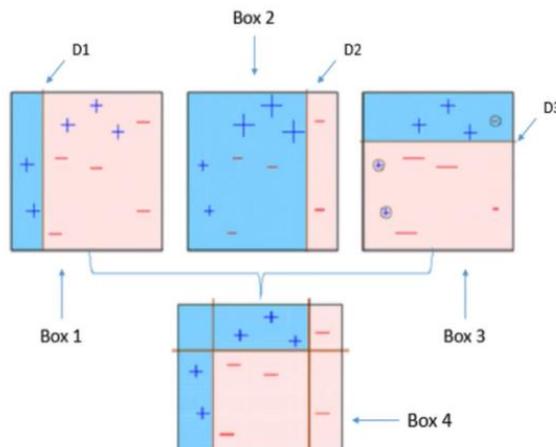
- Bagging ile Boosting yönteminin temel farkı nedir?
 - Bagging yönteminde ağaçlarının birbirine bağımlılıkları yoktur.
 - Boosting yönteminde ise ağaçlar artıkların üzerine kurulur. Bağımlılık vardır.

Gradient Boosting Machines (GBM)

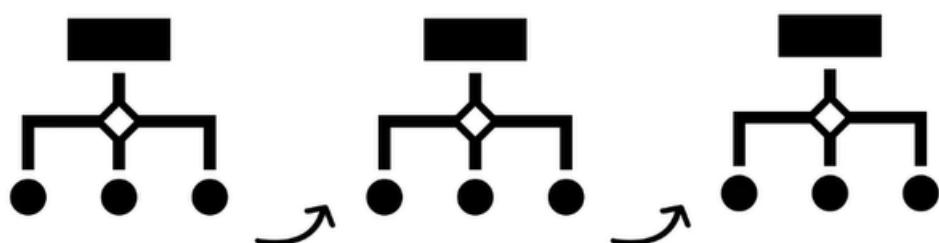
- Artık optimizasyonlarına dayanan ağaç yöntemlerine boosting ve gradient descent uygulanmasıdır.
- Öncesinde AdaBoost'e gitmek faydalı olacaktır.

AdaBoost (Adaptive Boosting)

- GBM'in temelli buna dayalıdır. GBM bazı kısıtlamalarını kaldırıp bir de Gradient Descent Yöntemini ekleyerek daha genel bir formunu ortaya çıkarmıştır.
- AdaBoost ise zayıf sınıflandırıcıların bir araya gelerek güçlü bir sınıflandırıcı oluşturması fikrine dayanır.



- **GBM** ise Hatalar/artıklar üzerine tek bir tahminsel model formunda olan modeller serisi kurulur.

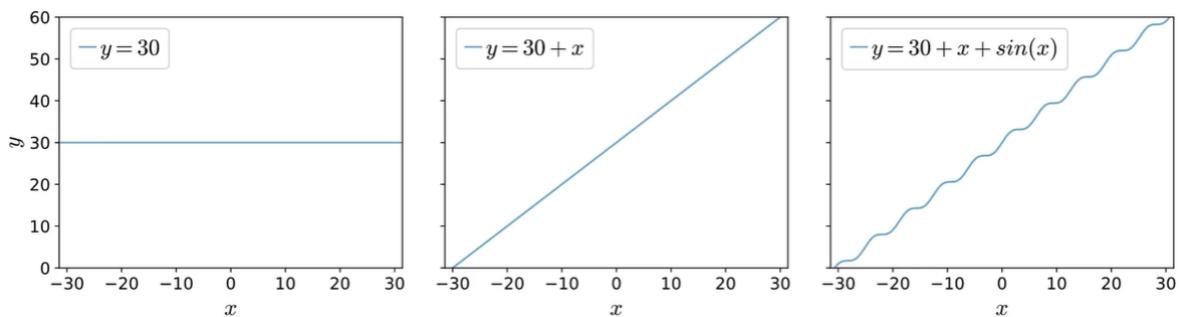


- Boosting + Gradient
- Gradient Boosting tek bir tahminsel model formunda olan modeller serisi oluşturur.

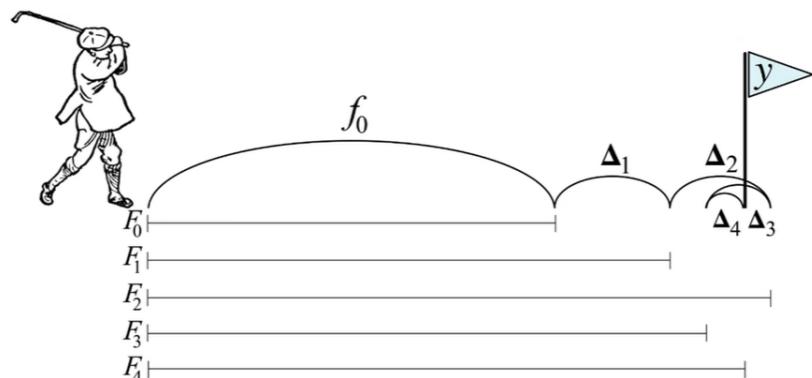
- Seri içerisindeki bir model serideki bir önceki modelin tahmin artıklarının/hatalarının (residuals) üzerine kurularak oluşturulur.
- GBM diferansiyellenebilen herhangi bir kayıp fonksiyonunu optimize eden Gradient Descent algoritmasını kullanmaktadır.
- Tek bir tahminsel model formunda olan modeller serisi **additive** şekilde kurulur.

Additive Modeling

- Kutuyu çeşitli bölgelere ayıryorduk. Grafikte ise ayırma işlemini git gide daha da hassaslaştırıyoruz.
- Sabit 30 skoru yerine artıkları modelledik.



- Aşağıdaki görsellerde ekleme yapılarak modelin daha başarılı sonuç verdiğiini anlayabiliriz. Bu işlemin formüle edilmiş hali için 2. Görsele bakılmalıdır.



$$\begin{aligned}\hat{y} &= f_0(\mathbf{x}) + \Delta_1(\mathbf{x}) + \Delta_2(\mathbf{x}) + \dots + \Delta_M(\mathbf{x}) \\ &= f_0(\mathbf{x}) + \sum_{m=1}^M \Delta_m(\mathbf{x}) \\ &= F_M(\mathbf{x})\end{aligned}$$

$$\begin{aligned}F_0(\mathbf{x}) &= f_0(\mathbf{x}) \\ F_m(\mathbf{x}) &= F_{m-1}(\mathbf{x}) + \Delta_m(\mathbf{x})\end{aligned}$$

XGBoost (eXtreme Gradient Boosting)

- XGBoost, GBM'in hız ve tahmin performansını artırmak üzere optimize edilmiş ölçülebilir ve farklı platformlara entegre edilebilir versiyonudur.
- Bir müddet çok başarılı idi. Ama daha sonra LightGBM geldi ve tahtından etti.

LightGBM

- LightGBM, XGBoost'un eğitim süresi performansını artırmaya yönelik geliştirilen bir diğer GBM türüdür.
- Level-wise büyümeye stratejisi yerine Leaf-wise büyütme stratejisi ile daha hızlıdır.
- XGBoost geniş kapsamlı bir ilk arama yaparken LightGBM derinlemesine ilk arama yapmaktadır.
- LightGBM'de en önemli hiper parametresi tahmin sayısıdır (`n_estimators`). Tahmin sayısı iterasyon sayısıdır. Yani boosting saysıdır.
 - Bütün hiper parametre optimizasyonu yapıldıktan sonra tek başına en uygun `n_estimators` aranabilir. `n_estimators`'u artışı gidilebilir.

CatBoost (Categoric Boosting)

- Kategorik değişkenler ile otomatik olarak mücadele edebilen, hızlı, başarılı bir GBM türevidir.

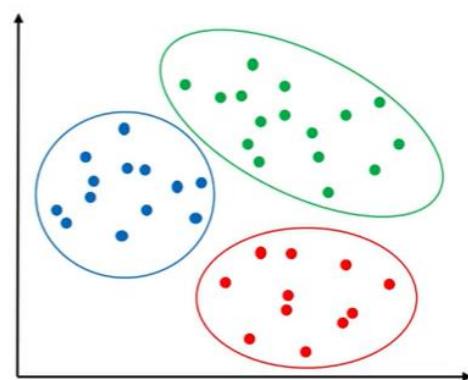
Unsupervised Learning

K-Means

- Amaç gözlemleri birbirlerine olan benzerliklerine göre kümelere ayırmaktır.
 - Küme içi homojen, kümelerarası heterojen yapmak.



K-Means'ten Önce

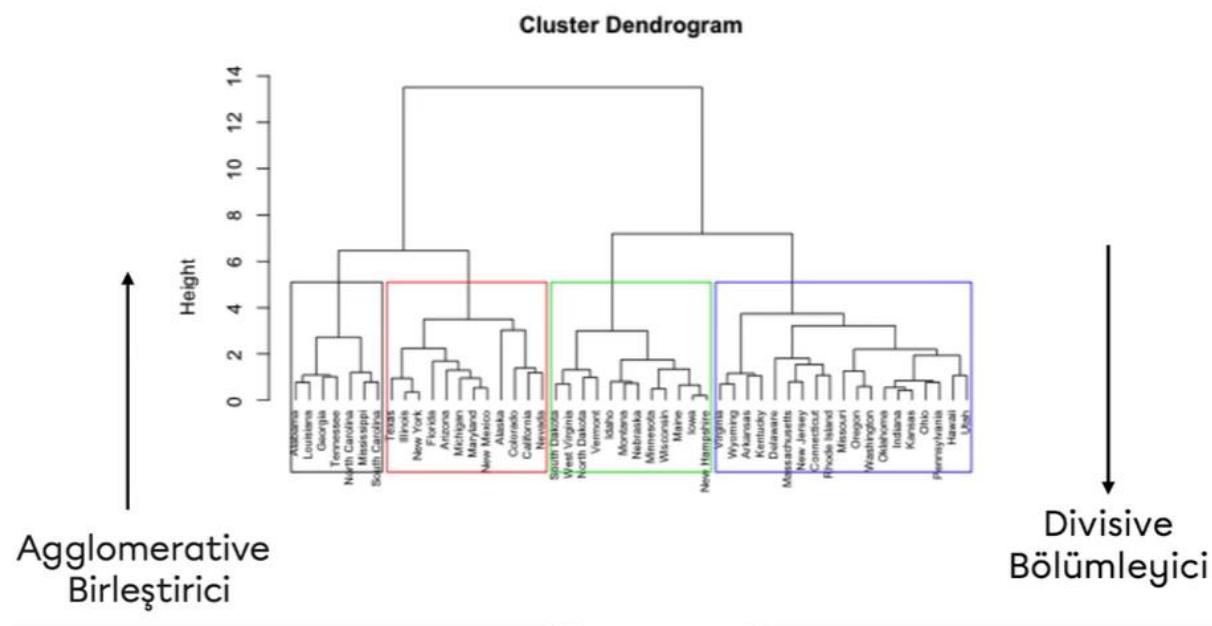


K-Means'ten Sonra

- K-Means nasıl çalışır?
 - Adım 1: Küme sayısı belirlenir.
 - Adım 2: Rastgele k adet merkez seçilir.
 - Adım 3: Her bir gözlem için k merkezlere uzaklıklar hesaplanır.
 - Adım 4: Her gözlem en yakın olduğu merkeze yani kümeye atanır.
 - Adım 5: Atama işlemlerinden sonra oluşan kümeler için tekrar merkez hesaplamaları yapılır.
 - Adım 6: Bu işlem belirlenen bir iterasyon adedince tekrar edilir ve küme içi hata kareler toplamlarının toplamı (**total with-in cluster variation**) minimum olduğu durumdaki gözlemlerin kümeleme yapısı nihai kümelenme olarak seçilir.
 - **k_cluster parametresi** gözlem birimine yaklaşıkça SSE düşer. Ama her bir gözlemin bir küme olması istediğimiz bir şey değil. Optimum küme sayısını belirlemeye çalışıyoruz.

Hierarchical Cluster Analysis

- Amaç gözlemleri birbirlerine olan benzerliklerine göre kümelere ayırmaktır.
 - Ama burada kümelere ayırma işlemi hiyerarşik olarak yapılıyor.

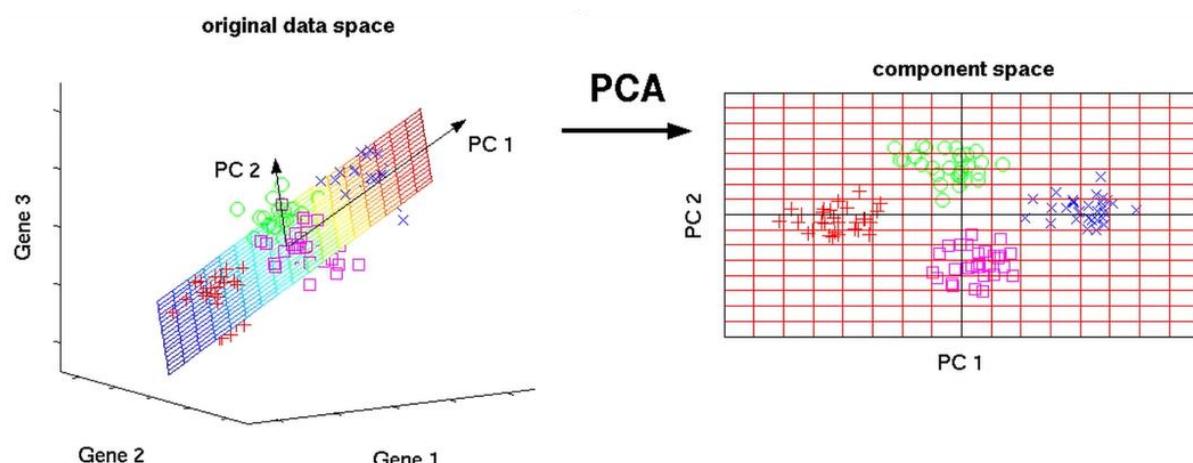


- **Agglomerative:**
 - Çalışmanın başında bütün gözlem birimleri tek bir küme gibi düşünülüp yukarıya doğru iki gözlem birimi bir araya getirilip bir küme oluşturulur ya da üç gözlem bir araya gelip bir küme sonra bu ikili ve üçlü gözlem bir araya gelip bir küme olusturur şeklinde yukarıya doğru birleştirerek işlemler yapılır.

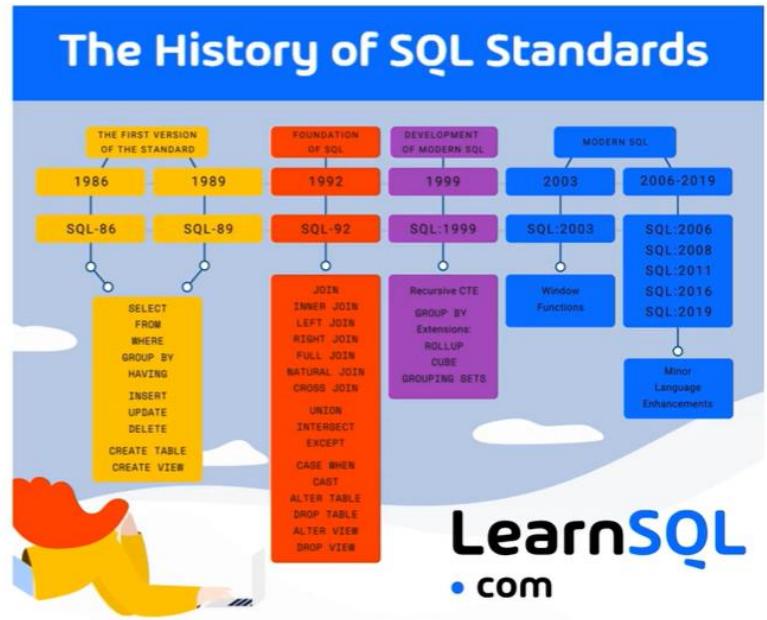
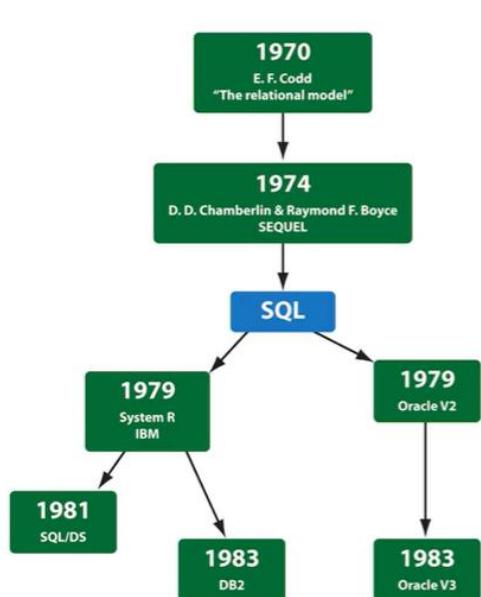
- **Divisive:**
 - Tüm gözlemler bir arada olduğunda tek bir kümedir. Ondan sonra iki küme elde edilir. Böyle böyle aşağıya doğru bölümlenerek işlemler yapılır.
- **K-means ile Hierarchical Cluster'in farkı nedir?**
 - K-means'de küme oluşturma sürecine dışarıdan müdahale edemiyorduk. Dolayısıyla bir gözleme imkanımız yoktu.
 - Ama Hiyerarşik Kümelemede (Alaska) tek başına kalmış diğerleri ile kümeleyelim diyebiliriz. Yani görsel teknik üzerinden çeşitli kümelenme seviyelerinde yeni kümleri tanımlayabiliriz.

Principal Component Analysis (PCA):

- Çok değişkenli verinin ana özelliklerini daha az sayıda değişken/bileşen ile temsil etmektir.
 - Yani; küçük miktarda bir bilgi kaybını göze alıp değişken boyutunu azaltmaktadır.
 - PCA uygulanan sette **korelasyon kalmaz**.



SQL (Structred Query Language)



Temel Kavramlar

- Veritabanı: Temel anlamda verileri listeler halinde tablo ve satırlarda tutan her yapı aslında kendi içinde veritabanıdır.
 - Veritabanı bileşenleri:
 - Tablolar
 - Sütunlar
 - Satırlar
 - İndeksler
 - Excel, Access gibi yapılar birer veritabanı örneğidir.
- Veritabanı Yönetimi Sistemi: MsSQL, Oracle, MySQL, PostgreSQL birer veritabanı yönetim sistemleridir.
 - Bu sistemler gelişmiş yapılar olduğu için sadece SQL dili ile yetinmezler. SQL dili komutları az olan basit bir dildir.
 - Bu yapılar hem SQL dilini içeren hem de nitelikli sorgulama yapmak için ve daha çok gelişmiş işlemleri yapmak için kendilerine özel diller geliştirmiştir.
 - Oracle – PL/SQL
 - MSSQL – T-SQL
- Veritabanı Sunucusu:

İlişkisel Veritabanı Kavramı (RDBMS):

- Tekrar eden verileri tekilleştirmek amacıyla yapılan veritabanı sistemleridir.
 - Tekrar eden verileri sürekli girmekten kaynaklanan emek ve iş gücü kaybı
 - Tekrar eden verilerin gereksiz yere işgal etmesi ve kaynak israf etmesi
 - Veri girerken insan hatası sebebiyle veri bütünlüğünün bozulması
 - Veride geçmişe dönük olarak güncelleme yapmanın zor olması

Veri Tipleri ve Normalizasyon

Tam Sayı Veri Tipleri

- Normalizasyonun önemi:

KAYIT SAYISI	200.000.000						
KOLON ADI	KOLON TÜRÜ	KAYIT SAYISI	HAFIZADA KAPLANAN ALAN (BYTE)	TOPLAM ALAN	TOPLAM ALAN MB	TOPLAM ALAN GB	
ID	BIGINT	200.000.000		8 1.600.000.000	1.525,88	1,49	
CITYID	BIGINT	200.000.000		8 1.600.000.000	1.525,88	1,49	
TOWNID	BIGINT	200.000.000		8 1.600.000.000	1.525,88	1,49	
DISTRICTID	BIGINT	200.000.000		8 1.600.000.000	1.525,88	1,49	
							5,96
KOLON ADI	KOLON TÜRÜ	KAYIT SAYISI	HAFIZADA KAPLANAN ALAN (BYTE)	TOPLAM ALAN	TOPLAM ALAN MB	TOPLAM ALAN GB	
ID	INT	200.000.000		4 800.000.000	762,94	0,75	
CITYID	TINYINT	200.000.000		1 200.000.000	190,73	0,19	
TOWNID	SMALLINT	200.000.000		2 400.000.000	381,47	0,37	
DISTRICTID	INT	200.000.000		4 800.000.000	762,94	0,75	
							2,05

Tamsayı Veri Tipleri

bigint	Minimum: - 2^{63} (-9,223,372,036,854,775,808)	8 Byte
	Maksimum: $2^{63}-1$ (9,223,372,036,854,775,807)	
int	Minimum: - 2^{31} (-2,147,483,648)	4 Byte
	Maksimum: $2^{31}-1$ (2,147,483,647)	
smallint	Minimum: - 2^{15} (-32,768)	2 Byte
	Maksimum: $2^{15}-1$ (32,767)	
tinyint	Minimum: 0	1 Byte
	Maksimum: 255	
bit	0 ya da 1 değerini alır.	Eğer tabloda 8 ya da daha az bit kolonu varsa 1 byte, 8'den fazla ise 2 byte yer kaplar.

String Veri Tipleri

- Metin Veri Tiplerinin Normalizasyonu

KAYIT SAYISI	200.000.000						
KOLON ADI	KOLON TÜRÜ	KAYIT SAYISI	HAFIZADA KAPLANAN ALAN (BYTE)	TOPLAM ALAN	TOPLAM ALAN MB	TOPLAM ALAN GB	
ID	BIGINT	200.000.000		8	1.600.000.000	1.525,88	1,49
CITYID	BIGINT	200.000.000		8	1.600.000.000	1.525,88	1,49
TOWNID	BIGINT	200.000.000		8	1.600.000.000	1.525,88	1,49
DISTRICTID	BIGINT	200.000.000		8	1.600.000.000	1.525,88	1,49
NAMESURNAME	CHAR(200)	200.000.000		200	40.000.000.000	38.146,97	37,25
							43,21

KOLON ADI	KOLON TÜRÜ	KAYIT SAYISI	HAFIZADA KAPLANAN ALAN (BYTE)	TOPLAM ALAN	TOPLAM ALAN MB	TOPLAM ALAN GB
ID	INT	200.000.000		4	800.000.000	762,94
CITYID	TINYINT	200.000.000		1	200.000.000	190,73
TOWNID	SMALLINT	200.000.000		2	400.000.000	381,47
DISTRICTID	INT	200.000.000		4	800.000.000	762,94
NAMESURNAME	VARCHAR(200)	200.000.000		20	4.000.000.000	3.814,70
						3,73
						5,77

KAYIT SAYISI	200.000.000						
KOLON ADI	KOLON TÜRÜ	KAYIT SAYISI	HAFIZADA KAPLANAN ALAN (BYTE)	TOPLAM ALAN	TOPLAM ALAN MB	TOPLAM ALAN GB	
ID	BIGINT	200.000.000		8	1.600.000.000	1.525,88	1,49
CITYID	BIGINT	200.000.000		8	1.600.000.000	1.525,88	1,49
TOWNID	BIGINT	200.000.000		8	1.600.000.000	1.525,88	1,49
DISTRICTID	BIGINT	200.000.000		8	1.600.000.000	1.525,88	1,49
NAMESURNAME	NCHAR(200)	200.000.000		400	80.000.000.000	76.293,95	74,51
							80,47
KOLON ADI	KOLON TÜRÜ	KAYIT SAYISI	HAFIZADA KAPLANAN ALAN (BYTE)	TOPLAM ALAN	TOPLAM ALAN MB	TOPLAM ALAN GB	
ID	INT	200.000.000		4	800.000.000	762,94	0,75
CITYID	TINYINT	200.000.000		1	200.000.000	190,73	0,19
TOWNID	SMALLINT	200.000.000		2	400.000.000	381,47	0,37
DISTRICTID	INT	200.000.000		4	800.000.000	762,94	0,75
NAMESURNAME	NVARCHAR(200)	200.000.000		40	8.000.000.000	7.629,39	7,45
							9,50

char	0 ile 8000 arasında	Tanımlandığı değer kadar Byte.
		Char(10) -> 10 Byte
varchar	0 ile 8000 arasında	Girilen değer uzunluğu+ 2 Byte
varchar(MAX)	0 ile 2 147 483 647 arasında	Maksimum değeri: 2^31-1 (2,147,483,647) Byte
text	0 ile 2,147,483,647 arasında	Maksimum değeri: 2^31-1 (2,147,483,647) Byte
ntext	0 ile 1,073,741,823 arasında	Maksimum değeri: 2^30-1 (1,073,741,823) Byte

Ondalık Sayı Veri Tipleri

decimal/ numeric	Minimum: -10^38 +1	Hassasiyetine göre diskte kapladığı alan değişir. 1'den 9'a kadar Hassasiyet için: 5 byte
	Maksimum: 10^38 -1.	10'dan 19'a kadar Hassasiyet için: 9 byte 20'den 28'a kadar Hassasiyet için: 13 byte 29'dan 38'e kadar Hassasiyet için: 17 byte
money	Minimum: -922,337,203,685,477.5808 Maksimum: 922,337,203,685,477.5807	8 Byte
smallmoney	Minimum: -214,748.3648 Maksimum: 214,748.3647	4 Byte
float	-1.79308 ile -2.23308, 0 2.23308 ile 1.79308	7 basamağa kadar 4 Byte 15 basamağa kadar 8 Byte
Real	-3.438 ile -1.1838, 0 1.1838 ile 3.438	4 Byte

Tarih ve Saat Veri Tipleri

date	Minimum: 0001-01-01 Maksimum: 9999-12-31	4 Byte
smalldate	Minimum: 1900-01-01 Maksimum: 2079-06-06	3 Byte
datetime	Minimum: 1753-01-01 00:00:00.000 Maksimum: 9999-12-31 23:59:59.997	8 Byte
datetime2	Minimum: 0001-01-01 00:00:00.0000000	1-2 Hassasiyet İçin = 6 Byte
	Maksimum: 9999-12-31 23:59:59.9999999	3-4 Hassasiyet İçin = 7 Byte 5-7 Hassasiyet İçin = 8 Byte
datetimeoffset	Minimum: 0001-01-01 00:00:00.0000000	1-2 Hassasiyet İçin = 8 Byte
	Maksimum: 9999-12-31 23:59:59.9999999	3-4 Hassasiyet İçin = 9 Byte
	Time zone offset Aralığı: -14:00 / +14:00	5-7 Hassasiyet İçin = 10 Byte
time	Minimum: 00:00:00.0000000 Maksimum: 23:59:59.9999999	5 Byte(Default olarak kullanılırsa)

Diğer Veri Tipleri

image		Maksimum değeri: 2^31-1 (2,147,483,647) Byte
binary	0 ile 8000 arasında	Tanımlandığı değer kadar Byte. Binary(10) -> 10 Byte
varbinary	0 ile 8000 arasında	Tanımlandığı değer + 2 Byte
varbinary(MAX)	0 ile 2 147 483 647 arasında	Tanımlandığı değer + 2 Byte Maksimum değeri: 2^31-1 (2,147,483,647) Byte
Xml		Xml veriler için kullanılır.
Table		Sonradan kullanım amacıyla bir sonuç kümelerini saklamak için kullanılır.

	GUID(global olarak tekilliği garanti eder) veriyi tutar.
	select NEWID() script'ini çalıştırıldığınızda aşağıdaki gibi bir GUID veri oluşturur.
uniqueidentifier	A4C5DB26-7F18-4B4F-A898-E7DE26A8446A
	Bazen veritabanlarında tekilliği sağlamak için kullanılır. Ama bu amaçla kullanıldığında genelde performansı düşürür.
hierarchyid	Hiyerarşik yapılarda, hiyerarşideki pozisyonları temsil etmek için kullanılır.
geography	Dünyadaki koordinat sistemini tutar.
	Euclidean (flat) sistemi ile koordinat sistemini tutar.
geometry	Sadece 2 düzlem üzerinden hesaplanır. Dünyanın eğimlerini hesaba katmaz.

SQL Dili:

SELECT	Veritabanındaki tablolardan kayıtları çeker.
INSERT	Tabloya yeni kayıt ekler.
UPDATE	Tablodaki verilerin bir ya da birden çok alanını değiştirir.
DELETE	Tablodan kayıt siler.
TRUNCATE	Tablonun içini boşaltır.
CREATE	Bir veritabanı nesnesini oluşturur. (table,view,database...)
ALTER	Bir veritabanı nesnesinin özelliğini değiştirir.
DROP	Bir veritabanı nesnesini siler.

- Yorum satırı olmasını istersek “--” kullanılmalıdır.
- Truncate Delete Farkı?
 - Truncate tabloyu ilk oluşturulduğu hale siler yani ID **sıfırlanır**, Delete ise siler ve Auto Id varsa silinen yerden devam eder.
 - Delete sorularında **where** kullanabiliriz. Ama Truncate'de kullanamayız.
- Where komutunda eşitlik “==” ile **eşit değildir ise “<>”** olarak ifade edilir.

- Where'de Like operatörü ise “ile başlar ile biter içerir” şeklindeki verileri getirir.
 - SELECT * FROM CUSTOMERS WHERE NAMESURNAME LIKE "Ali%"
 - Adı-Soyadı Ali ile başlayanlar geldi.
 - SELECT * FROM CUSTOMERS WHERE NAMESURNAME LIKE "KOÇ%"
 - Adı-Soyadı KOÇ ile bitenler geldi.
 - SELECT * FROM CUSTOMERS WHERE NAMESURNAME LIKE "%Ali%"
 - Ali kelimesinin cümleinin herhangi bir yerinde geçenleri getir.
- Where için eğer birden fazla şartın sağlanmasını istiyorsak IN kullanılır.
 - SELECT * FROM CUSTOMERS WHERE CITY IN ("ANKARA", "İZMİR")
- DISTINCT komutu tekrar eden satırları tekilleştirmek amacıyla kullanılır.
 - Mesela şehirler
 - SELECT DISTINCT CITY, TOWN FROM CUSTOMERS WHERE CITY == "İstanbul"
- ORDER BY komutu istenilen sıraya göre sıralı gelmesini sağlayan komuttur.
 - SELECT * FROM CUSTOMERS ORDER BY NAMESURNAME
 - Order by deafult ascending'dır.
 - SELECT * FROM CUSTOMERS ORDER BY NAMESURNAME DESC
- TOP komutu bir SQL sorgusundan belirli sayıda satırın gelmesini sağlayan komuttur.
 - SELECT TOP 10 * FROM CUSTOMERS

Aggregate Functions ve Groupby

- Bir ham veri üzerinden veriyi tek satırda özetleyecek işlemlere Aggregation denir.
 - Aggregate Functions
 - Min
 - Max
 - Sum
 - Count
 - ...

```
SELECT SUM(PRICE),COUNT(ID),MIN(PRICE),MAX(PRICE),
AVG(PRICE)
FROM TABLOADI
WHERE CITY='İSTANBUL'
```

- Ham veriyi özetledik ama kırılımlar üzerinde görmek istersek **Group By** kullanıyoruz.

```

SELECT
    CITY,
    COUNT(*) SATIRSAYISI,
    SUM(TOTALPRICE) TOPLAMCIRO, AVG(TOTALPRICE) ORTALAMA_BIRIM_SATIS_TUTARI,
    MIN(TOTALPRICE) ENDUSUKFIYAT,
    MAX(TOTALPRICE) ENYUKSEKFIYAT
FROM SALES
WHERE CITY IN ('İSTANBUL', 'ANTALYA')

GROUP BY CITY
  
```

	CITY	SATIRSAYISI	TOPLAMCIRO	ORTALAMA_BIRIM_SATIS_TUTARI	ENDUSUKFIYAT	ENYUKSEKFIYAT
1	İstanbul	33780	2339719.54891231	69.2634561548936	0.166498704	9990
2	Antalya	2260	174176.97377908	77.0694574243717	0.228	7992

- Alias** kullanımına örnek: (AS ile de kullanılabilir.)
 - Alias'larda Türkçe karakter kullanmak istiyorsak çift tırnak veya köşeli parantez içinde yazmalıyız.

```

SELECT COUNT(*) SATIRSAYISI,
    SUM(TOTALPRICE) TOPLAMCIRO, AVG(TOTALPRICE) ORTALAMA_BIRIM_SATIS_TUTARI,
    MIN(TOTALPRICE) ENDUSUKFIYAT,
    MAX(TOTALPRICE) ENYUKSEKFIYAT
FROM SALES
WHERE CITY='ANTALYA'
  
```

	SATIRSAYISI	TOPLAMCIRO	ORTALAMA_BIRIM_SATIS_TUTARI	ENDUSUKFIYAT	ENYUKSEKFIYAT
1	2260	174176.97377908	77.0694574243717	0.228	7992

- ORDER BY**'da Alias ile ismi değişen sütunu kullanabiliriz. Ama **GROUP BY**'da ve **WHERE**'de kullanamayız.
- Bir Aggeragate Function WHERE içinde kullanılamıyor. Eğer Aggeragate Function'ının filtre olarak kullanmak istersek **HAVING** ile kullanmalıyız.

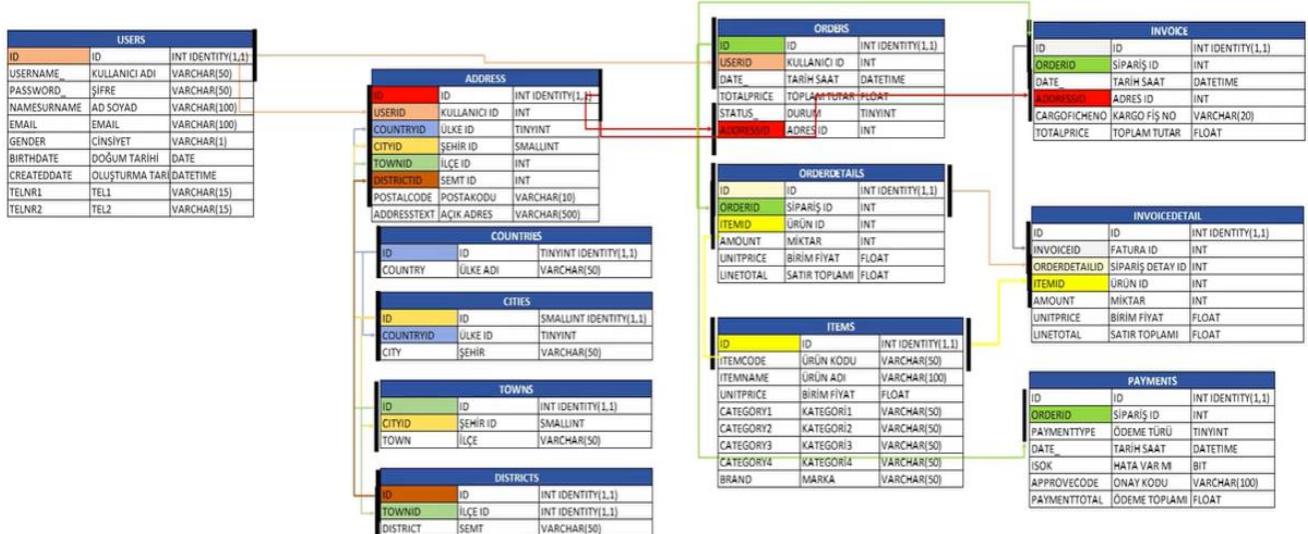
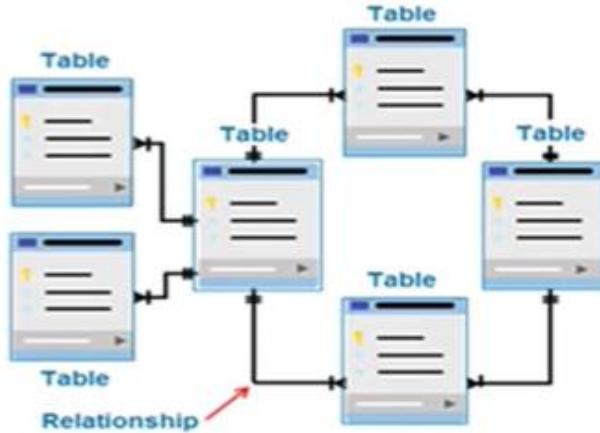
```

SELECT
    CITY, SUM(TOTALPRICE) TOTALSALE
FROM SALES

GROUP BY CITY
HAVING SUM(TOTALPRICE)>100000
ORDER BY 2 DESC
  
```

İlişkisel Veritabanı Kavramı (RDMS)

- Tekrar eden verileri tekilleştirmek amacıyla yapılandırılan veritabanı sistemleridir.



- İki farklı tablodan veri çekmek istersek Tablolara Alias tanımlayabiliriz.

```

SELECT
A.KOLON1,A.KOLON2,B.KOLON3,B.KOLON4
FROM TABLO1 A,TABLO2 B
WHERE A.PK_KOLON=B.FK_KOLON
    
```

- JOIN ile de birleştirilebilir.

```

SELECT
A.KOLON1,A.KOLON2,B.KOLON3,B.KOLON4
FROM TABLO1 A
JOIN TABLO2 B ON A.PK_KOLON=B.FK_KOLON
    
```

Subquery:

- İlişkisel veritabanlarında farklı bir birleştirme yöntemidir.
- Bir sorgu içinde başka bir sorguyu kolan gibi kullandığımız yapıya **Subquery** denir.

The screenshot shows a SQL query window in SSMS. The query is:

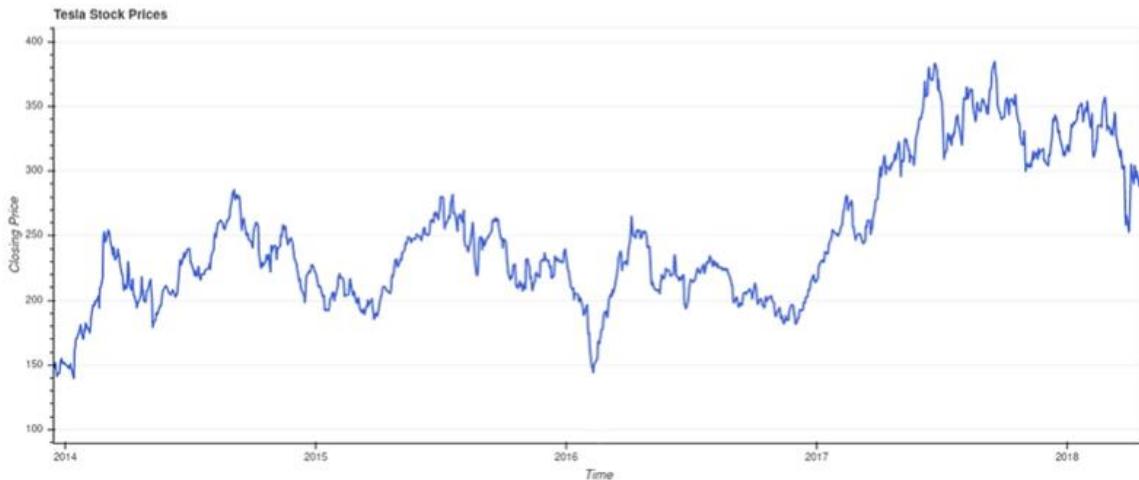
```
SELECT I.ID, I.ITEMCODE, I.ITEMNAME, I.BRAND, I.CATEGORY1,
       (SELECT SUM(AMOUNT) FROM ORDERDETAILS WHERE ITEMID=3)
FROM ITEMS I
```

The results pane shows a table with 17 rows of data from the ITEMS table. The last column, which is part of the subquery result, has the header '(No column name)'.

ID	ITEMCODE	ITEMNAME	BRAND	CATEGORY1	(No column name)
1	5	PIL KODAK XTRA HEAVY 9 V	KODAK	EV	40
2	6	PIL KODAK AA'2 MAX ALKALIN KALEM	KODAK	EV	40
3	9	PILLI SESLİ UCAK	OYUNCAK	OYUNCAK	40
4	11	VAKUMLU TASİYICI TIR	OYUNCAK	OYUNCAK	40
5	15	OYUNCAK KUTULU METAL CEK BIRAK	OYUNCAK	OYUNCAK	40
6	16	OYUNCAK BEZ BEBEK	OYUNCAK	OYUNCAK	40
7	17	OFICA SERIT DAKSIL	KIRTASIYELER	EV	40
8	18	OFICA MAKET BİGAGI UCU	KIRTASIYELER	EV	40
9	19	OFICA FH4550 HESAP MAKİNASI	KIRTASIYELER	EV	40
10	21	PRT UDAG WA0030 SAKLAMA KABI	UDAG	EV	40
11	22	PRT UDAG WA0046 TUVALET KAĞITLIGI	UDAG	TEMİZLİK	40
12	23	MARUL KIVİRCİK	MAROL	YESILLİK	40
13	29	CARPEX TYPES KLIMA KOKU SUMMER BREEZE	CARPEX	TEMİZLİK	40
14	30	PIL KODAK XTRA HEAVY KALEM 2'LÜ	KODAK	EV	40
15	35	OYUNCAK KUTUDA RENKLİ SACLI BEBEK	OYUNCAK	OYUNCAK	40
16	36	FIGURLU ÖĞRENCİ MAKASI	KIRTASIYELER	EV	40
17	37	CİCİT DOSYA	KIRTASIYELER	EV	40

Time Series

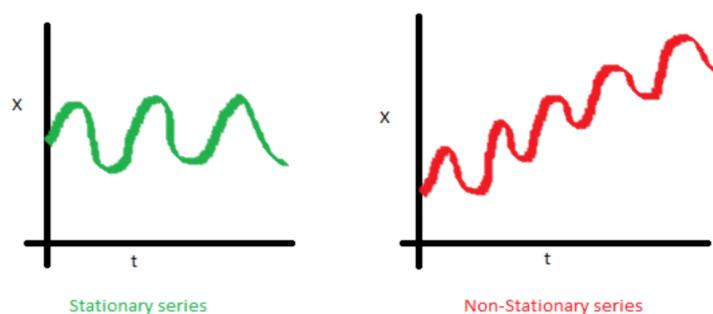
- Bir zaman serisi zaman göre sıralanmış gözlem değerlerinden oluşan veridir.



Temel Kavramlar:

Stationary (Durağanlık)

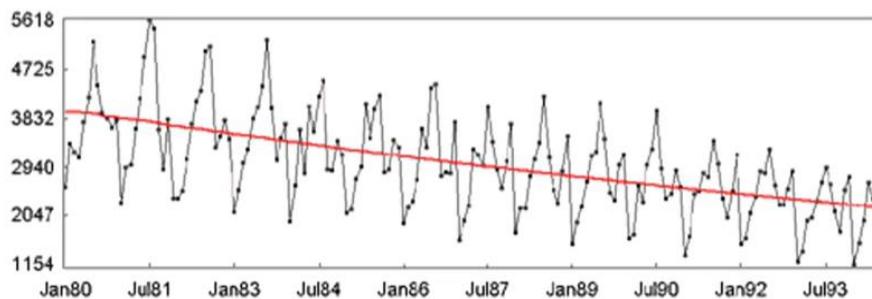
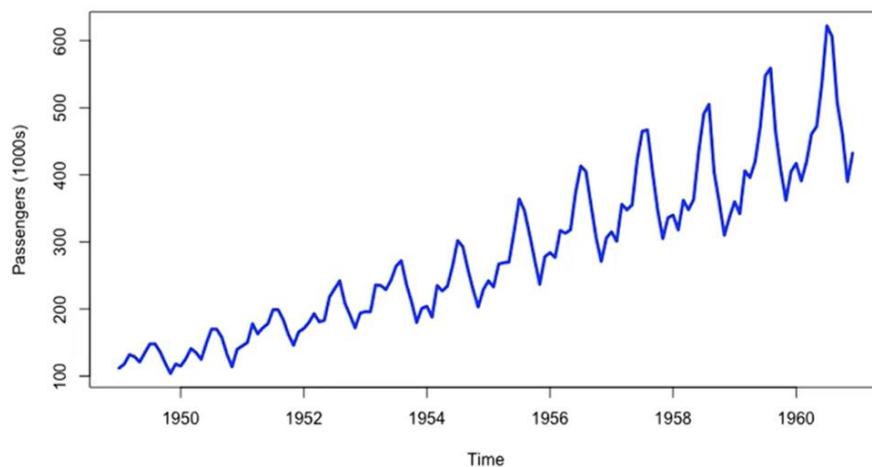
- Durağanlık, serinin istatistiksel özelliklerinin zaman içerisinde değişmemesidir.
 - Bir zaman serisinin **varyansı**, **ortalaması** ve **kovaryansı** zaman boyunca sabit kalıyorsa, serinin durağan olduğu kabul edilir.
- **Durağanlık neden burada önemli?**
 - Teorik olarak, zaman serisinin yapısı belirli bir örüntüde, belirli bir tahmin edilebilirlikte olursa daha tahmin edilebilir bir yapı vardır denir.
 - Durağan ise daha kolay tahminlerde bulunulabilir bekłentisi vardır.
 - Daya iyi tahminler ortaya çıkabilir.



- Durağanlıktan kurtulmak için kullanılan en yaygın yöntem “**Fark Alma**” yöntemidir.
 - Bugünden bir önceki günün değerleri çıkarılır ve durağan olması beklenir

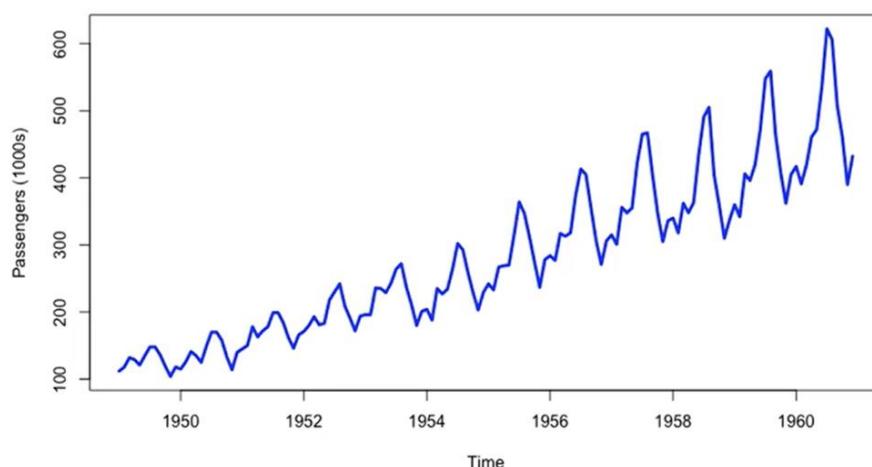
Trend

- Bir zaman serisinin uzun vadede artış ya da azalış gösterdiği yapıya trend denir.



Seasonality (Mevsimsellik)

- Zaman serisinin belirli bir davranışının belirli periyotlarla tekrar etmesi durumuna mevsimsellik denir.
 - Artış trendine sahip olan aşağıdaki görselde mevsimsellik vardır.



Cycle (Döngü)

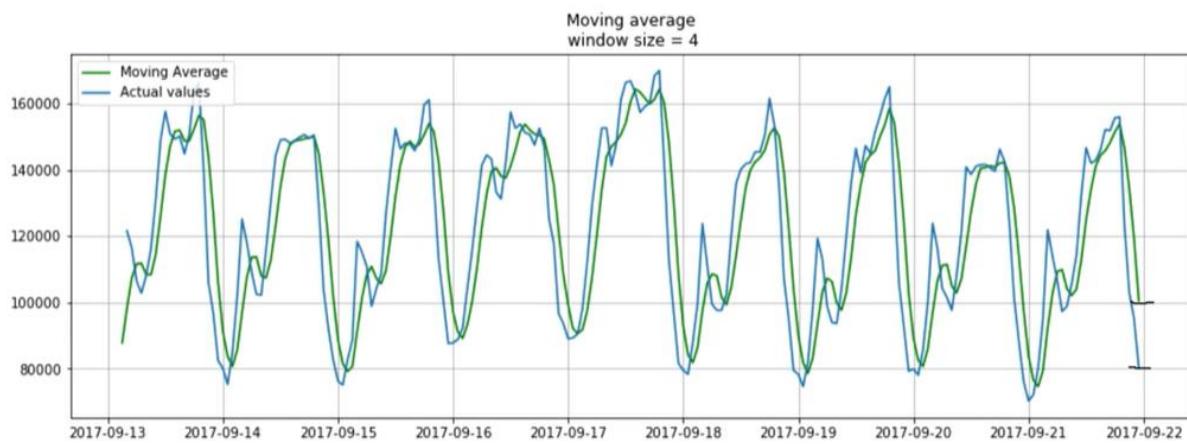
- Mevsimsellik belirli zaman periyotları ile örtüşecek şekilde gerçekleşir. Gün, ay, yıl gibi yapılar ile örtüşmektedir.
- Döngüsellik ise daha uzun vadeli, daha belirsiz bir yapıdadır. Gün, ay, yıl gibi yapılarla örtüşmez. Daha çok yapısal nedenler ile, konjokturel değişimler ile veya iş/politika dünyasındaki açıklamalar vs ile şekillenir.



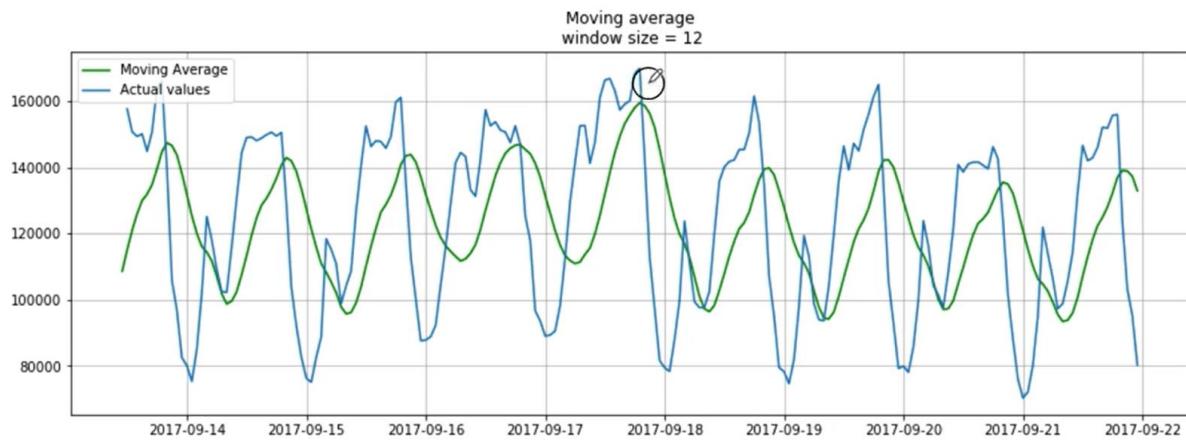
Moving Average (Hareketli Ortalama)

- Bir zaman serisinin gelecek değeri kendisinin k adet önceki değerinin ortalamasıdır.
- Burada sondaki değerlere ($t-1, t-2$ gibi) daha fazla ağırlık vereceğim olsaydım daha yakın tahmin edebilirdim.

$$\hat{y}_t = \frac{1}{k} \sum_{n=1}^k y_{t-n}$$



- Adım boyunu arttırdığımızda ortalama ile değerimiz arasındaki fark açıldı.
- Zaman boyu genişlediğinde, grafiğin normal gidişini yakalaması ve temsil etmesi geriden gelmiş.



Weighted Average (Ağırlıklı Ortalama)

- Ağırlıklı ortalama, hareketli ortalamaya benzer. Daha sonlarda olan gözlemlere daha fazla ağırlık vermek fikrini taşır.

$$\hat{y}_t = \sum_{n=1}^k \omega_n y_{t+1-n}$$

	Ads		Ads
	Time		Time
2017-09-21 19:00:00	155890	2017-09-21 19:00:00	155890
2017-09-21 20:00:00	123395	2017-09-21 20:00:00	123395
2017-09-21 21:00:00	103080	2017-09-21 21:00:00	103080
2017-09-21 22:00:00	95155	2017-09-21 22:00:00	95155
2017-09-21 23:00:00	80285	2017-09-21 23:00:00	80285

Moving Average (Son 3 gözlemin ortalaması):

$$(123395 + 103080 + 95155) / 3 = 107210$$

Weighted Average ([0.1, 0.3, 0.6]):

$$(123395 * 0.1) + (103080 * 0.3) + (95155 * 0.6) = 100356$$

- Ağırlıklı Ortalama ve Hareketli Ortalama verideki trendi yakalayamaması tahmin sonuçlarını kötü etkiler.
- Seri kendisinden önceki değerlerden çok etkileniyor. Seri içerisinde bir trend, mevsimsellik olması durumunda tahmin sonuçlarını direkt etkiliyor.

Smoothing Yöntemleri (Holt-Winters)

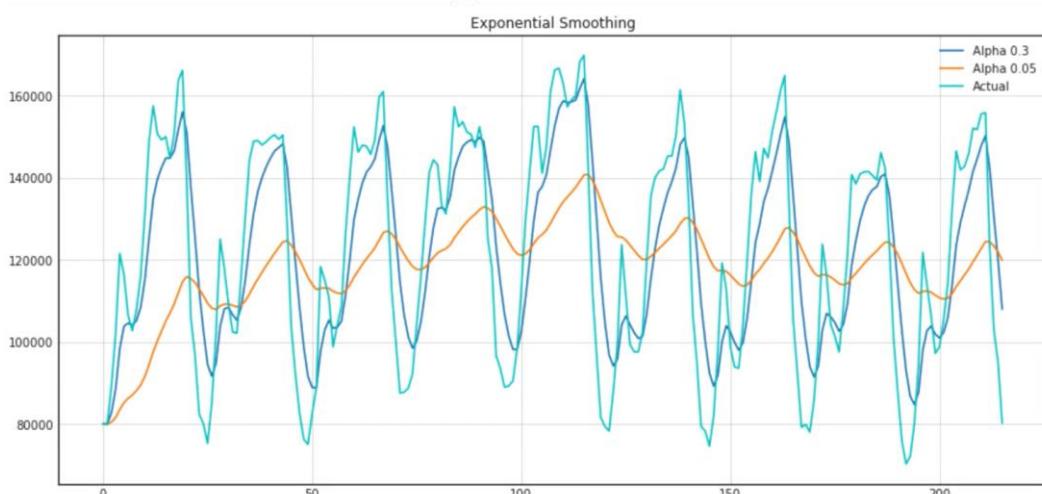
- Single Exponential Smoothing: Sadece Durağan serilerde, trend ve mevsimsellik olmamalı
- Double Exponential Smoothing: Level + Trend, ama Mevsimsellik olmamalı
- Triple Exponential Smoothing: Level + Trend + Mevsimsellik

Algorithm	Level	Trend	Seasonality	Tuning Parameters
Single HWES	Yes	No	No	α
Double HWES	Yes	Yes	No	α, β
Triple HWES	Yes	Yes	Yes	α, β, γ

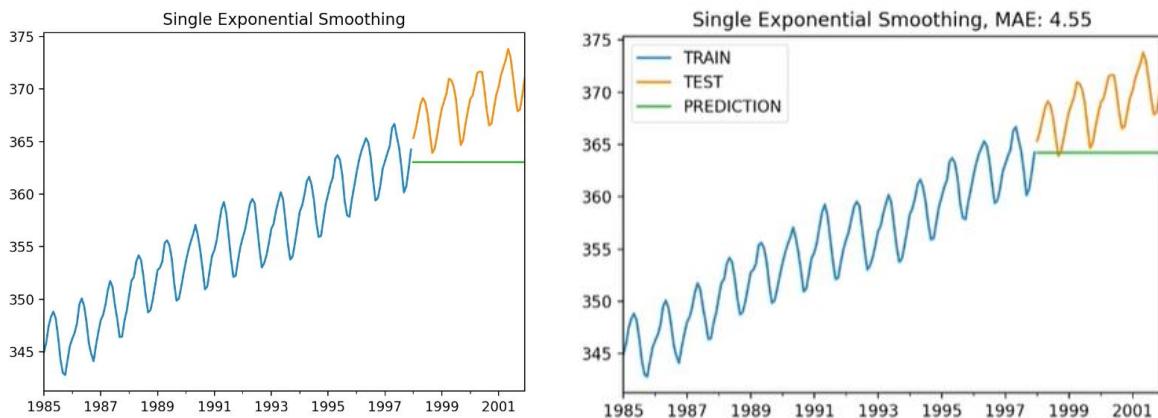
SES (Single Exponential Smoothing)

$$\hat{y}_t = \alpha \cdot y_{t-1} + (1 - \alpha) \cdot \hat{y}_{t-1}$$

- Exponential denenmesinin sebebi ağırlıklı ortalamanın genelleştirilmiş halini barındırmasıdır.
- Üssel düzeltme yaparak tahminde bulunur.
- Gelecek yakın geçmişle daha fazla ilişkilidir varsayımlıyla geçmişin etkileri ağırlıklandırılır.
- Peki formül bize ne diyor?
 - Geçmiş gerçek değerler ve geçmiş tahmin edilen değerlerin üssel olarak ağırlıklandırılmasıyla tahmin yapılır.
- Trend ve mevsimsellik içermeyen (yani durağan) tek değişkenli zaman serileri için uygundur.



- Görselden çıkaracağımız sonuç, alfa değerleri ne kadar büyükse gerçek değerlere o kadar yakın tahmin edebiliriz.



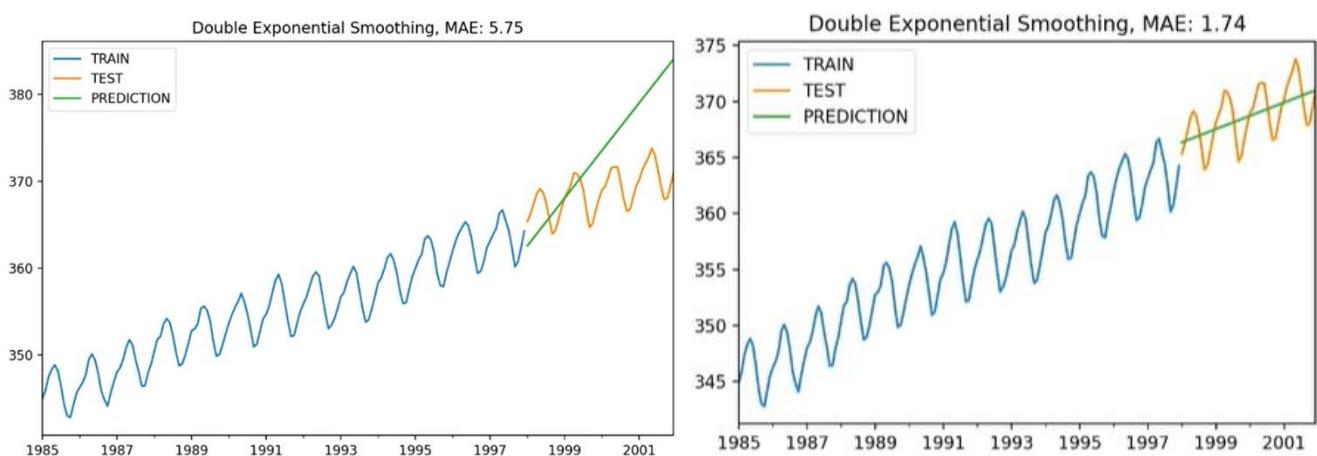
Double Exponential Smoothing (DES)

$$\ell_t = \alpha y_t + (1 - \alpha)(\ell_{t-1} + b_{t-1})$$

$$b_t = \beta(\ell_t - \ell_{t-1}) + (1 - \beta)b_{t-1}$$

$$\hat{y}_{t+1} = \ell_t + b_t$$

- Trend etkisini göz önünde bulundurarak üssel düzeltme yapar.
- DES = Level (SES) + Trend
- Temel yaklaşım aynıdır. SES'e ek olarak trend dikkate alınır.
- Trend içeren ve mevsimsellik içermeyen tek değişkenli zaman serileri için uygundur.**



Triple Exponential Smoothing (aka Holt-Winters)

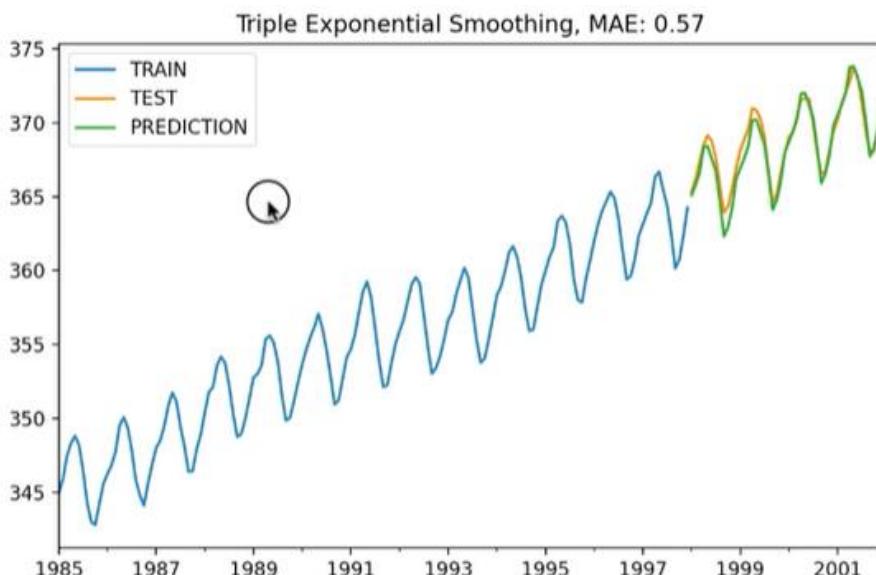
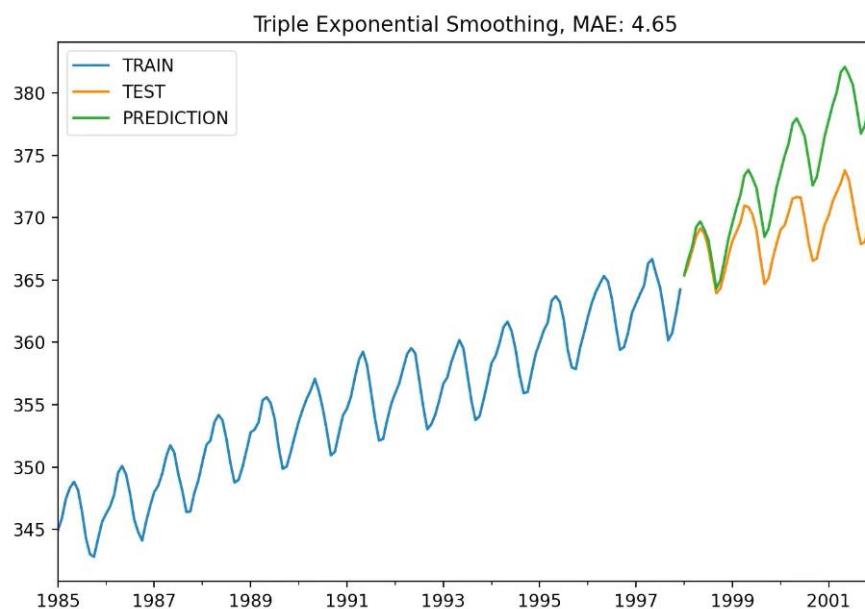
$$\ell_t = \alpha(y_t - s_{t-p}) + (1 - \alpha)(\ell_{t-1} + b_{t-1})$$

$$b_t = \beta(\ell_t - \ell_{t-1}) + (1 - \beta)b_{t-1}$$

$$s_t = \gamma(y_t - \ell_t) + (1 - \gamma)s_{t-p}$$

$$\hat{y}_{t+m} = \ell_t + mb_t + s_{t-p+1+(m-1)modp}$$

- TES = Level (SES) + Trend + Mevsimsellik
- Triple Exponential Smoothing en gelişmiş smoothing yöntemidir.
- Bu yöntem dinamik olarak level, mevsimsellik ve trend etkilerini değerlendирerek tahmin yapmaktadır.
- Trend ve/veya mevsimsellik içeren tek değişkenli serilerde kullanılabilir.



İstatistik Metodları

AR(p): Autoregression

- Önceki zaman adımlarındaki gözlemlerin **doğrusal** bir kombinasyonu ile tahmin işlemi yapılır.
 - Autoregression, bir zaman serisi kendisinden önceki gecikmeler ile tahmin edilir denir.
- Trend ve mevsimsellik içermeyen tek değişkenli zaman serileri için uygundur.
- p zaman gecikmesi sayıdır ve p=1 ise bir önceki zaman adımı ile model kurulmuş demek olur.
- SES ile benzer aslında. Temel farkları ise:
 - SES'te tahmin ve geçmiş gerçek değerler göz önünde bulunduruluyordu. AR'da ise sadece geçmiş gerçek değerler var.
 - SES'te sadece katsayı var ve iki terimin etkisini belirlemeyi sağlıyordu. AR'da ise birçok terim eklenebilir ve her biri doğrusal formda regresyon modeli olarak ele alınmış olmaktadır.

AR(1) model :

$$y_t = a_1 y_{t-1} + \epsilon_t$$

AR(2) model :

$$y_t = a_1 y_{t-1} + a_2 y_{t-2} + \epsilon_t$$

AR(p) model :

$$y_t = a_1 y_{t-1} + a_2 y_{t-2} + \dots + a_p y_{t-p} + \epsilon_t$$

MA(q): Moving Average

- Önceki zaman adımlarında elde edilen hataların doğrusal bir kombinasyonu ile elde edilir.
 - AR modelinde geçmiş gerçek değerler vardı. MA modelinde ise geçmiş hatalar vardır. MA modeli geçmiş hatalara dayalı bir model gerçekleştirirken AR modeli geçmiş gerçek değerlere dayalı bir model gerçekleştirir.
- Trend ve mevsimsellik içermeyen tek değişkenli zaman serileri için uygundur.
- q=zaman gecikmesi sayısıdır.

$$y_t = m_1 \epsilon_{t-1} + \epsilon_t$$

$$y_t = m_1 \epsilon_{t-1} + m_2 \epsilon_{t-2} + \dots + m_q \epsilon_{t-q} + \epsilon_t$$

ARMA(p, q)

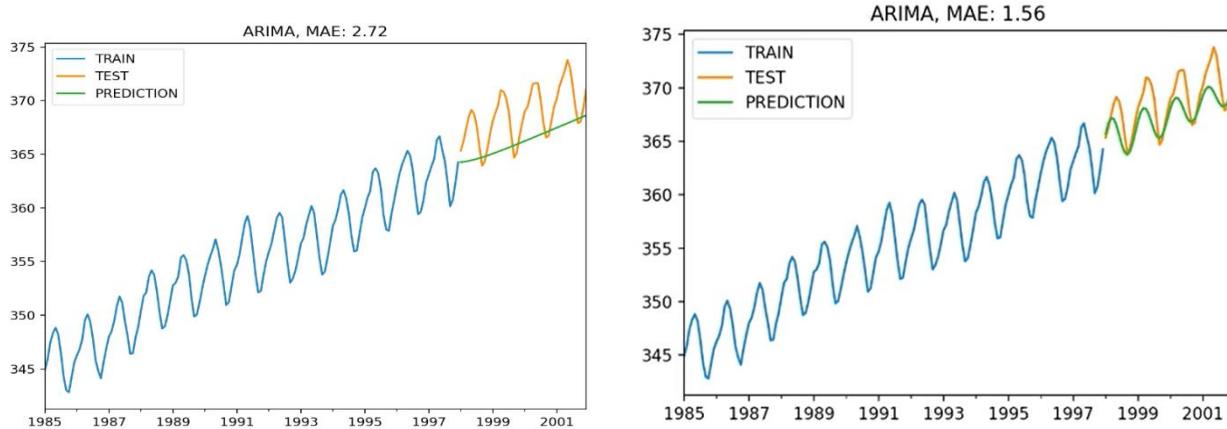
- AutoRegressive Moving Average olarak isimlendirilir.
- Geçmiş değerler ve geçmiş hataların doğrusal kombinasyonu ile tahmin yapılır.
- Trend ve mevsimsellik içermeyen tek değişkenli zaman serileri için uygundur.
- p ve q zaman gecikmesi sayılarıdır. p AR modeli için, q ise MA modeli içindir.
- ARMA Modeli ile SES modeli çok benzerdir. Farkları:
 - SES'te smoothing faktörü (alfa) adı verdigimiz bir katsayı vardı. İki tane terimin etkilerini ağırlıklandırdı. Bir terimimiz geçmiş gerçek değerlerdi ve diğer ise hatalar/artıklardı. SES'in alfa'ya ciddi bir bağımlılığı vardır. Bu nedenle tek bir form vardır gibi düşünülebilir.
 - ARMA'da ise geçmiş gerçek değerlerin ağırlığı a1 ile tahmin üzerinden oluşturulacak artıkların katsayılarından (m1) tamamen bağımsız. ARMA bir doğrusal regresyon formundadır.
 - Holt-Winters yöntemlerde terimler bir parametreye göre şekillenirken ARMA modellerinde terimlerin kendi katsayıları var ve bu katsayıları bu katsayıların bulunma işi ile ilgilenilir. Veri doğrusal bir formda modellerin ve bu model neticesinde verinin özütü öğrenilir.

$$y_t = a_1 y_{t-1} + m_1 \epsilon_{t-1} + \epsilon_t$$

ARIMA

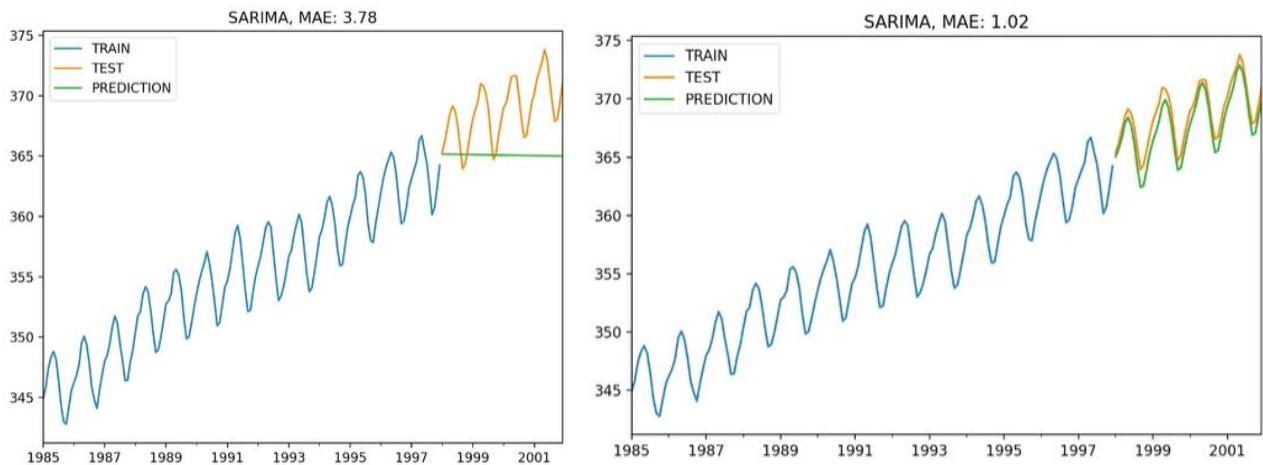
- ARIMA (p, d, q)
 - Autoregressive Integrated Moving Average
- Önceki zaman adımlarındaki **farkı alınmış** gözlemlerin ve hataların doğrusal bir kombinasyonu ile tahmin yapılır.
 - Durağan serinin tahmini daha kolay varsayımdan dolayı ARIMA durağan olmayan serilerde fark alır ve böylece seriyi durağanlaştırır.
- ARIMA'nın ARMA'dan temel farkı:
 - ARMA modeli trend ve mevsimsellik varsa başarısız sonuç verirken ARIMA fark alarak bu durumun üstesinden gelir.
- Tek değişkenli, trendi olan fakat mevsimselliği olmayan veriler için uygundur.
- p gerçek değer gecikme sayısını (otoregresif derece) ifade eder.
 - p=2 ise y_{t-1} ve y_{t-2} modeldedir.
- d fark alma işlem sayısıdır (fark derecesi, 1).

- q hata gecikme sayısıdır (hareketli ortalama derecesi).
 - $q=2$ ise et-1 ve et-2 modeldedir.
- ARMA SES'e benzerken, ARIMA trendi modelleyebildiği için DES'e benzer.



SARIMA

- SARIMA (p, d, q) (P, D, Q) m
 - Seasonal Autoregressive Integrated Moving-Average
- ARIMA + mevsimsellik
- Trend ve/veya mevsimsellik içeren tek değişkenli serilerde kullanılabilir.
- p, d ve q ARIMA'dan gelen parametrelerdir, **trend elemanlarıdır**.
 - ARIMA trendi modelleyebiliyordu.
- p gerçek değer gecikme sayısını (otoregresif derece) ifade eder.
 - $p=2$ ise $yt-1$ ve $yt-2$ modeldedir.
- d fark alma işlem sayısıdır (fark derecesi, 1).
- q hata gecikme sayısıdır (hareketli ortalama derecesi).
 - $q=2$ ise et-1 ve et-2 modeldedir.
- P, D, Q mevsimsel gecikme sayıları, Mevsimsel elemanlardır.
- m tek bir mevsimsellik dönem için zaman adım sayısıdır. Mevsimselliğin görülmeye yapısını ifade eder.
 - Haftada bir mi, ayda bir mi, 12 ayda bir mi gibi



Zaman Serileri Ekstra Bilgiler

❖ Zaman Serilerinde eksik değerler nasıl doldurulur?

- Zaman serilerinde eksik değerler alışık olduğumuz yöntemlerdeki gibi bütün verilerin ortalaması ile doldurulmaz. **Eksik değerler, kendisinden bir önceki ve bir sonraki değerin ortalaması ile ya da kendisinden önceki ve sonraki değerler ile doldurulur.**
- Bu şekilde doldurulur. Çünkü seride trend veya mevsimsellik varsa bütün verilerin ortalaması ile doldurulması hataya neden olur.

❖ Dickey-Fuller (istatistikî) testi bize serinin durağan olup olmadığı hakkında bilgi verir.

❖ Çarpımsal ve toplamsal model nedir?

- Mevsimsellik ve artık bileşenler trend'den bağımsız gibi gözüyürsa bu durumda seri toplamsal bir seridir. Öte yandan Mevsimsellik ve Artık Bileşenler trende bağımlı gibi yani trend'e göre şekilleniyor gibi gözüyürsa seri, çarpımsal bir seridir.
- $y(t) = \text{Level} + \text{Trend} + \text{Seasonality} + \text{Noise}$
- $y(t) = \text{Level} * \text{Trend} * \text{Seasonality} * \text{Noise}$
 - Çarpımsal seride fonksiyonel yapı daha bağımlı bir şekilde değişir.

Durağan	SES, AR, MA, ARMA
Trend	DES, ARIMA, SARIMA
Trend + Mevsimsellik	TES, SARIMA