VERİ ANALİZİ OKULU

2025-2026

Makine Öğrenmesi ve Yapay Zekaya Giriş

Dr. Öğr. Üyesi M. Fuat KINA

Marmara Üniversitesi
Nüfus ve Sosyal Araştırmalar Enstitüsü

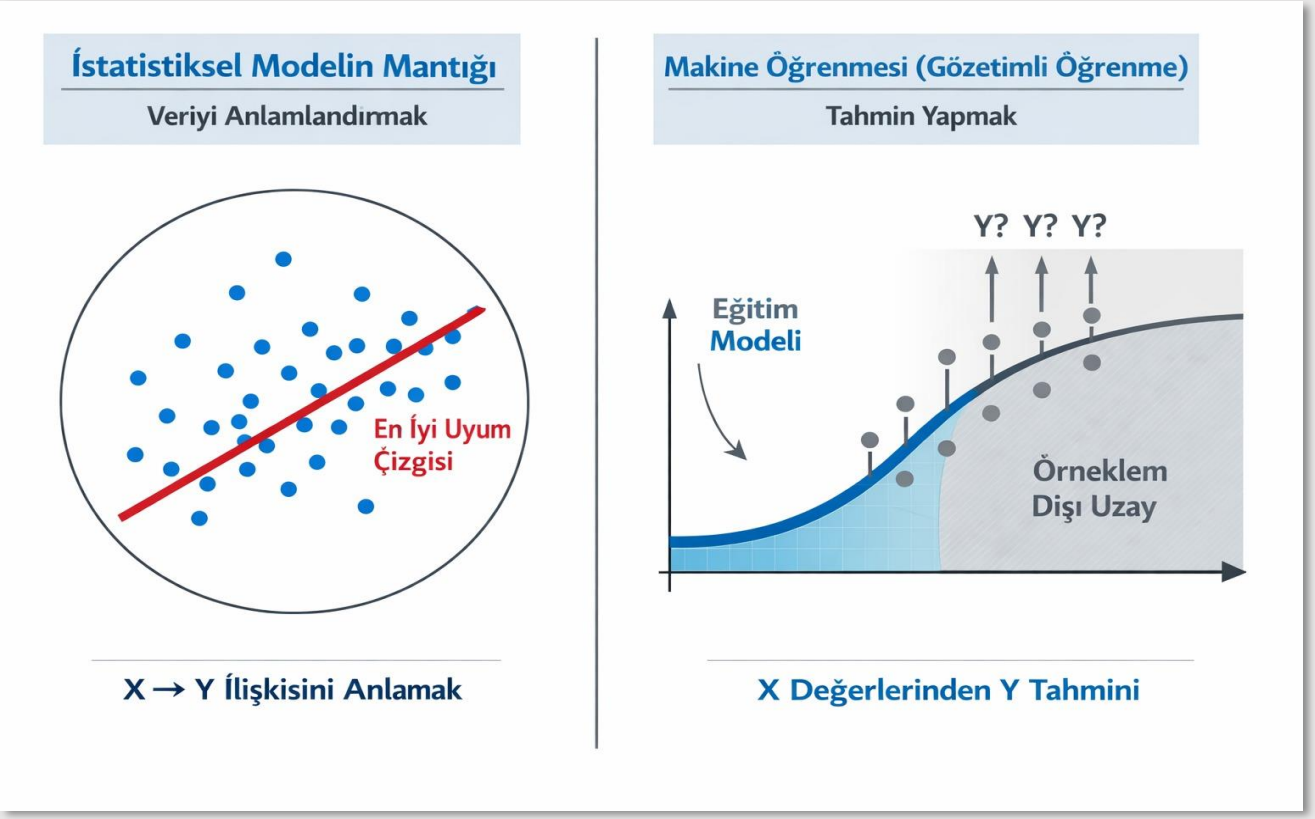
Dersin Çerçevesi

- Makine öğrenmesi nedir?
- Makine öğrenmesi türleri (gözetimli, gözetimsiz)
- Gözetimli öğrenme: *sınıflandırma* (k-NN), *regresyon* (düzenlileştirme-regularizasyon)
- Hiperparametre ayarı
- Ön-işleme ve pipeline
- Gözetimsiz öğrenme: **k-means** kümeleme

Makine Öğrenmesi Nedir?

- Bilgisayarlara, açıkça programlamadan, veriden öğrenerek iş yapmayı öğretme teknolojisi.
  - Örnek: Bir e-postanın içerik ve gönderen bilgisine bakarak spam / değil diye karar verme.
  - Örnek: Wikipedia girdilerini, içerdiği kelimelere göre kümelere ayırma.
- **İstatistik**ten **makine öğrenmesi**ne geçiş:
  - Veriden anlamlı neden-sonuç ilişkileri geliştirmek ➖
  - İçinde yaşadığımız dünyayı tahmin etmek (*nowcasting / forecasting*) ➕
  - **Temel fark:** Makine öğrenmesi modeli (bir istatistik modelinin aksine) geliştirildiği yerdeki değil, bunun dışında yer alan uzaydaki ilişkileri öğrenebildiği ölçüde başarılı kabul edilir.
- Etiketler varsa: gözetimli öğrenme (supervised).
- Etiketler yoksa: gözetimsiz öğrenme (unsupervised).

Makine Öğrenmesi Nedir?



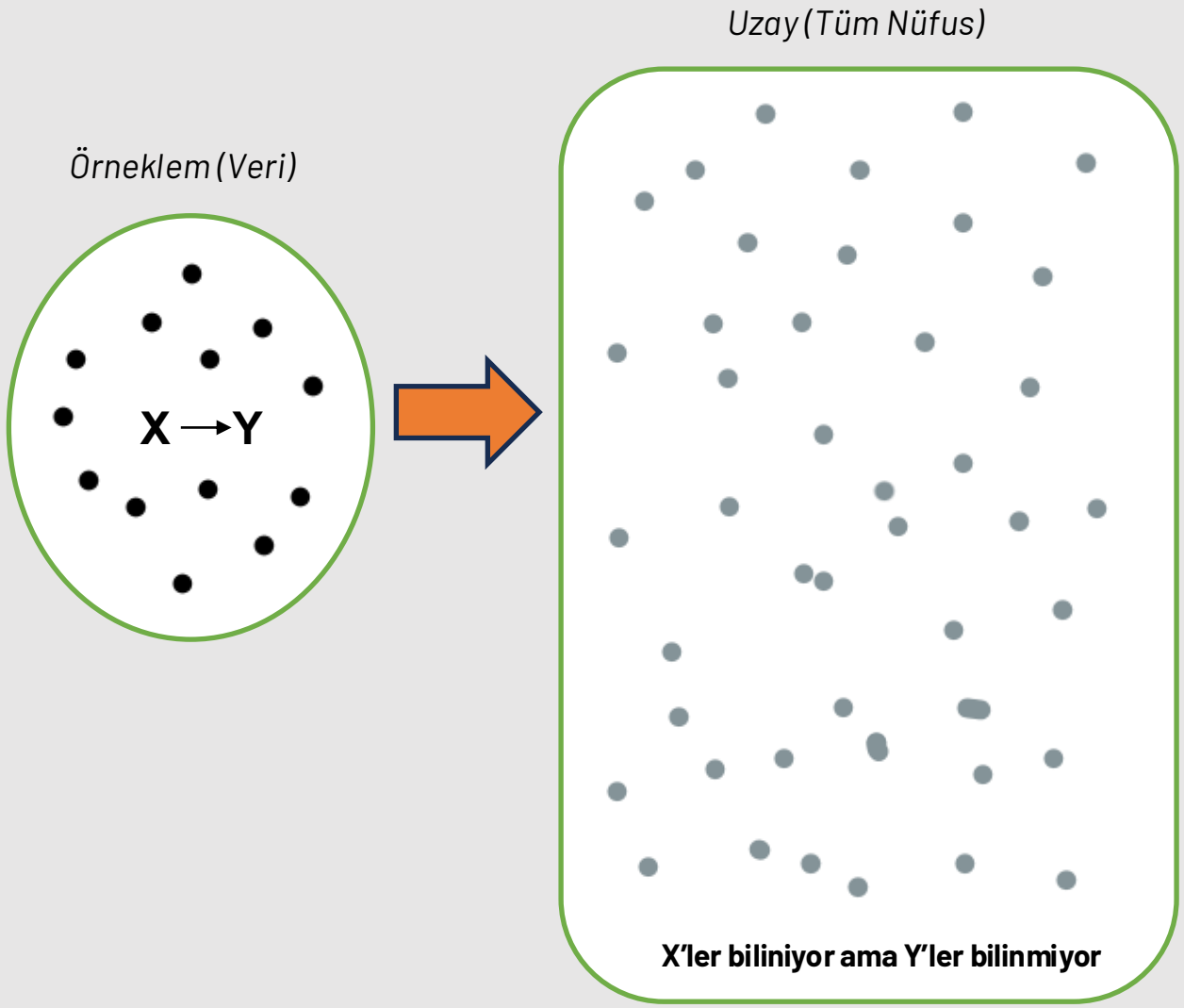
İstatistiksel çıkarım: 1000 kişilik bir anket datası içerisinde eğitimin gelire etkisi

- Amaç: Neden-sonuç ilişkisi kurup hipotezleri test etmek
- Örneklem dışı uzay (out-of-sample) önemsiz

Makine öğrenmesi: Haber metnine bakıp protesto-eylem var mı yok mu karar veren bir model.

- Amaç: Henüz tedavülde olmayan bir veriyi yaratmak
- Örneklem dışı uzay (out-of-sample) modelin temel ilgisi

Makine Öğrenmesi Nedir?



Örneklem (Veri): Hem X'leri hem Y'leri bildiğim yer. Makine Öğrenmesi için eğitim alanı, tahminlemenin başladığı yer. İstatistik içinse kurulacak olan iddianın hem başladığı hem bittiği yer.

Uzay (Tüm Nüfus): Tahminleme yaptığım alan. Tüm insanlık, tüm Türkiye, tüm haber metinleri, tüm tweetler

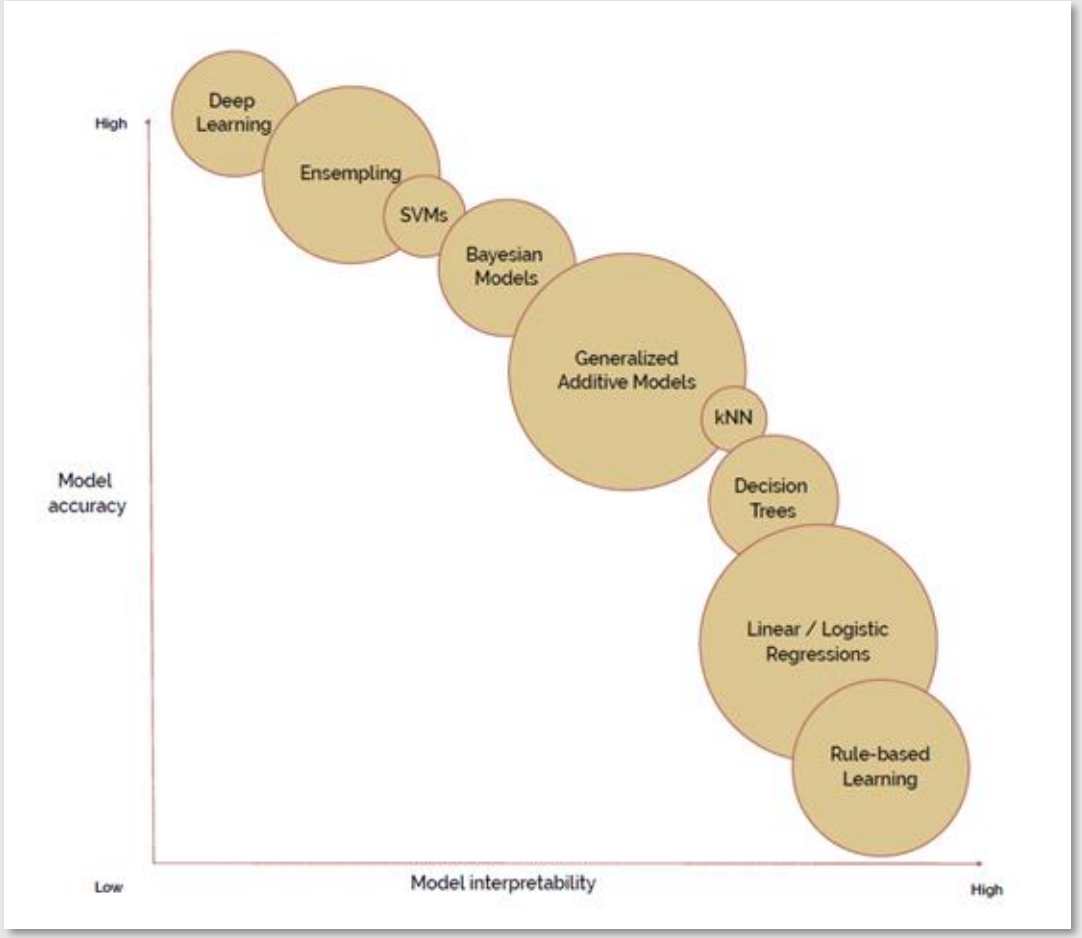
- İstatistiksel çıkarımda örneklemden uzaya geçiş temsiliyet problemiyle ilgilidir.
- Makine öğrenmesi ise bu ilişkiyi temsiliyete dayalı bilimsel varsayımla değil test sonuçlarıyla edinir. Eğitim datası dışında (out-of-sample) X'ler Y'leri doğru bir şekilde tahmin edebiliyorsa model başarılıdır.

Makine Öğrenmesi Nedir?

- Modeller büyük ölçüde 1980'lerde geliştirildi, ancak 2000'lerde uygulamaya konulabildi.

- Modellerin karmaşıklığı zaman içerisinde arttıkça yorumlanabilirlik azalırken, tahminleme kapasitesi yükseldi.
  - Makine öğrenmesi (2000-2010)
  - Derin öğrenme ve sinir ağları (2010-2020)
  - Üretken yapay zeka (2020'ler)

(Şekil için: Toosi et al. 2022)



Gözetimli ve Gözetimsiz Öğrenme

Gözetimsiz, etiketsiz

- Etiketlenmemiş datanın içerisindeki saklı desen ve yapıları ortaya çıkartır
- Kümeleme, boyut eksiltme, yapılandırılmamış konu modellemesi

Gözetimli, etiketli

- Amaç, doğru çıktısını bildiğimiz veriden öğrenerek, çıktısını bilmediğimiz yeni veriler için tahmin yapabilmektir
  Zaman alıcı veya maliyetli bir manuel işi otomatikleştirmek; gelecek hakkında tahminlerde bulunmak vb.
  Etiketli veriye ihtiyaç duyar (tarihsel veriler, kitle kaynaklı etiketleme vb.)

1. Gözetimli Öğrenme

Gözetimli Öğrenme

- Tahmin değişkenlerini (özellikleri) ve bir hedef değişkeni kullanarak, hedef değişkeni tahmin edebilen bir model kurmak
  - Sürekli hedef değişken
    - **Regresyon**
  - İkili (binary) hedef değişken
    - **Sınıflandırma (classification)**
- **Feature** ya da bağımsız değişken = *predictor* (tahmin değişkeni)
- **Target** ya da bağımlı değişken = *response* / *outcome* (hedef / çıktı)
- Python: scikit-learn kütüphanesi (**sklearn**)

Bunlardan hangisi bir **sınıflandırma problemi**dir? Yandaki makine öğrenmesine ilişkin 4 örnek uygulama verilmiştir. Bunlardan hangisi **gözetimli bir sınıflandırma problemi**dir?

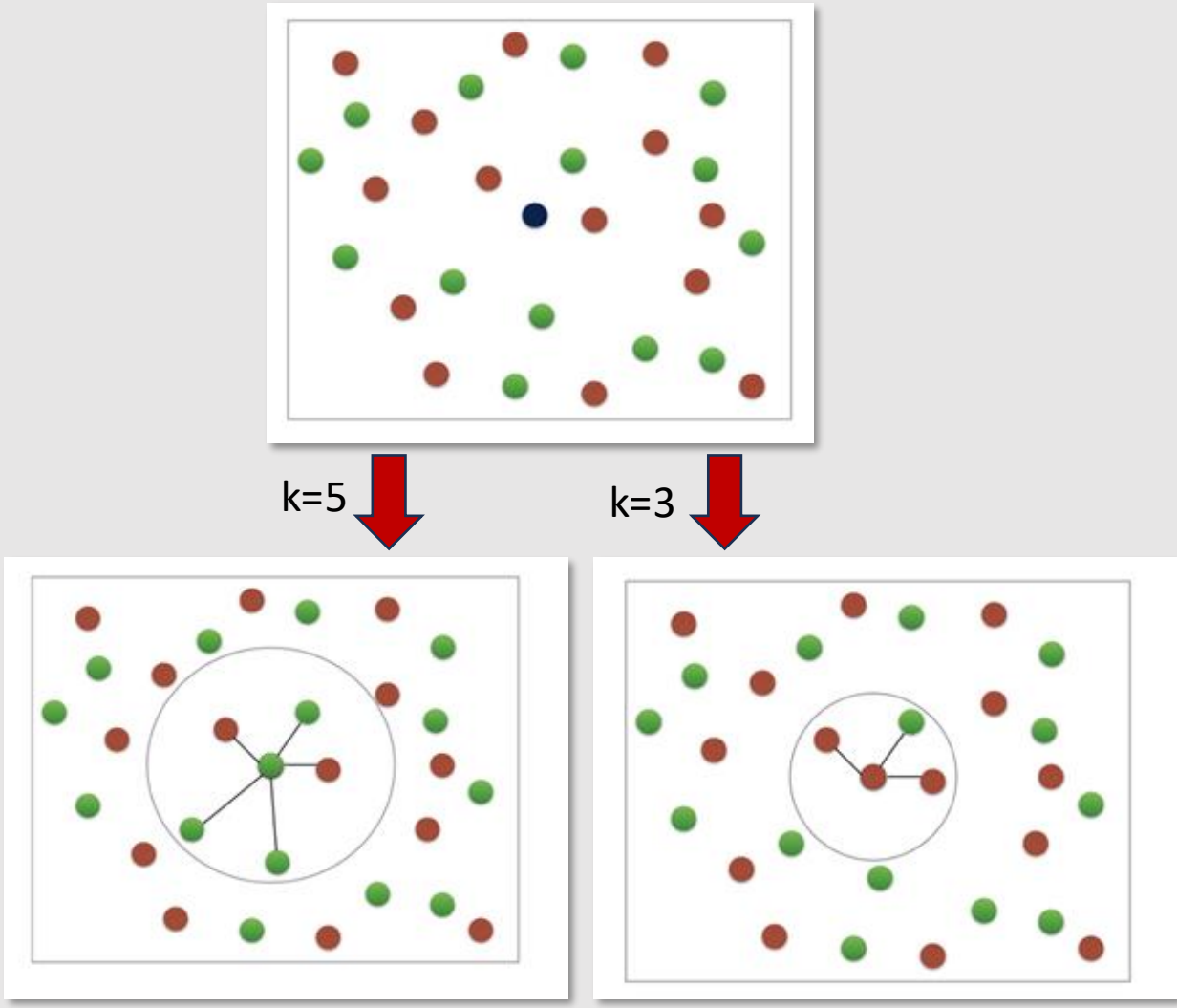
- A. Etiketlenmiş finansal verileri kullanarak, bir hissenin değerinin gelecek hafta **yukarı mı yoksa aşağı mı** gideceğini tahmin etmek.
- B. Etiketlenmiş konut fiyatı verilerini kullanarak, çeşitli özelliklere göre yeni bir evin **fiyatını** tahmin etmek.
- C. Etiketsiz verileri kullanarak, çevrimiçi bir eğitim şirketinin öğrencilerini **öğrenme stillerine göre** farklı kategorilere/kümelere ayırmak.
- D. Etiketlenmiş finansal verileri kullanarak, bir hissenin **gelecek hafta tam olarak kaç değerinde** olacağını tahmin etmek.

1. A. Sınıflandırma

Sınıflandırma

k-En Yakın Komşu (k-Nearest Neighbors)

- En yakın **k** tane etiketli veri noktasına bakar ve **çoğunluk oyunu** alır.
- Sınıflar arasında **karar sınırları (decision boundaries)** çizer.
- *5 komşu vs 3 komşu*

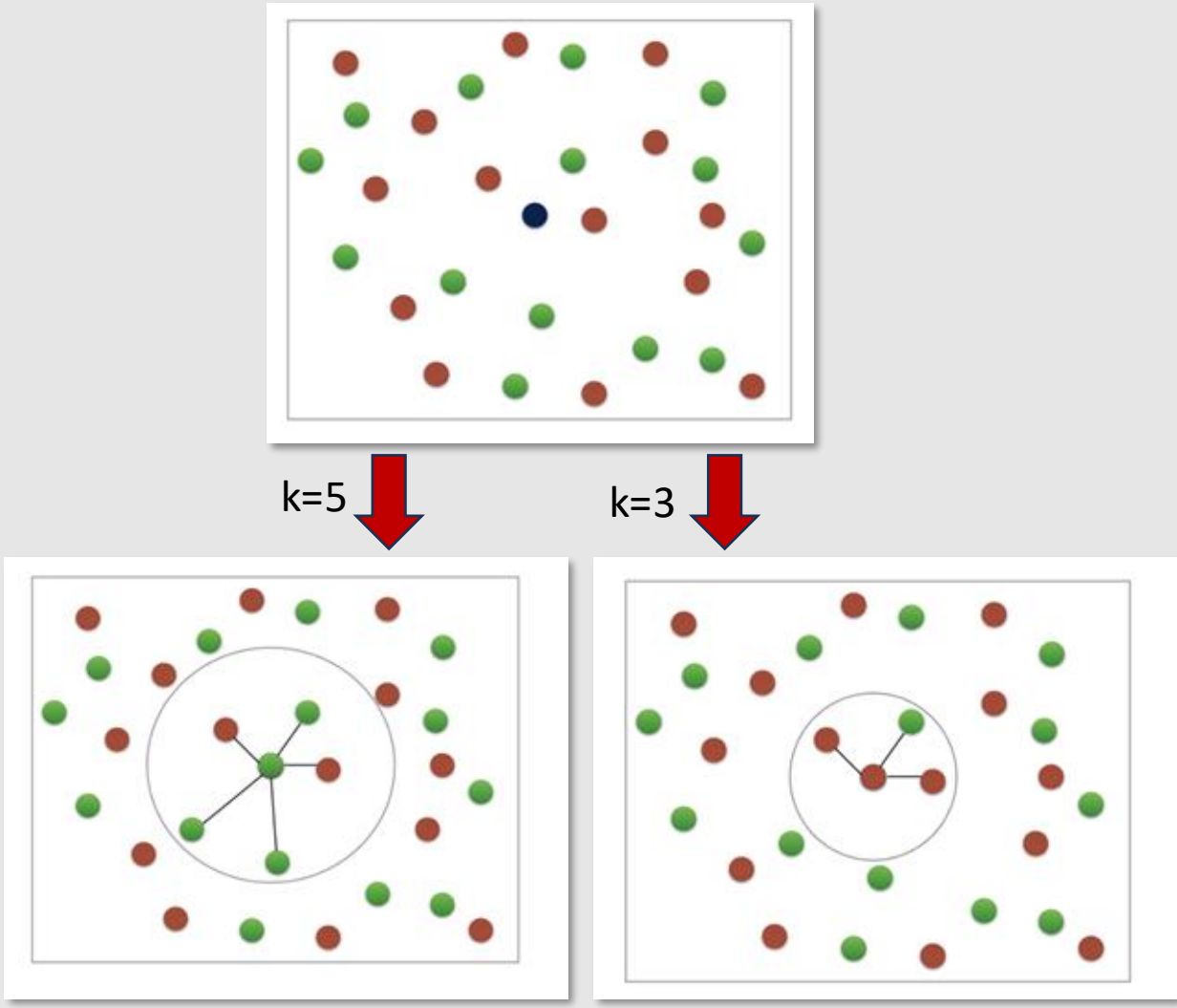


- Makine öğrenmesinde iki adım vardır: **öğrenme** ve **tahmin etme**
- Veriler üzerinde bir modeli eğitmek = .fit()
- Yeni veri için etiketleri tahmin etmek = .predict()

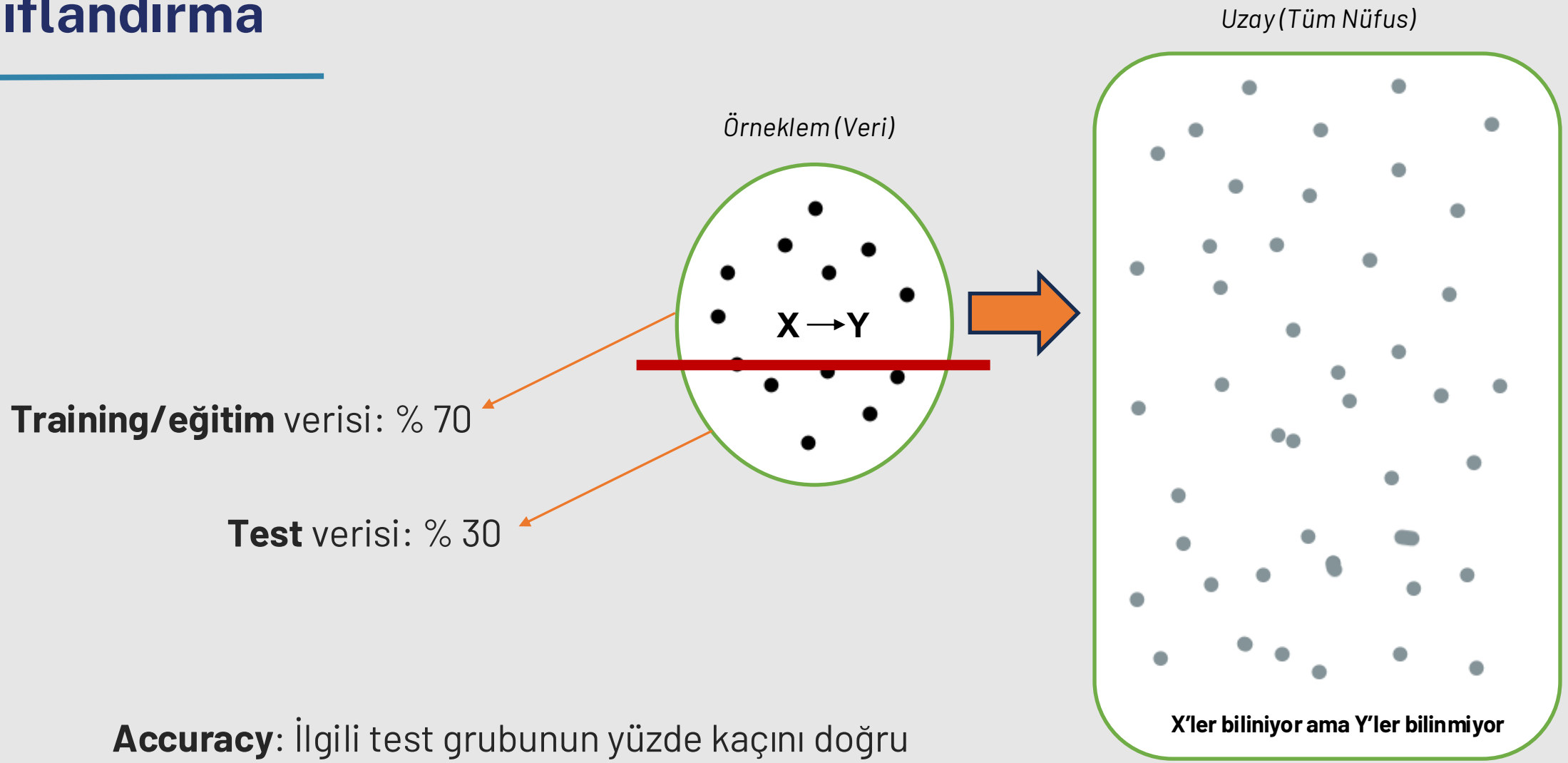
Sınıflandırma

Modelin performansını nasıl ölçeriz?

- **Doğruluk (accuracy)** = Doğru tahminlerin tüm tahminlere oranı
- Peki doğruluğu hangi veri üzerinde kontrol etmeliyim: **modeli bina ettiğim (eğitim) veri mi, yoksa yeni (tahmin) veri mi?**
- Çözüm: **Train/test split (eğitim/test ayrımı)**
  - Sınıflandırıcıyı **eğitim seti** üzerinde fit et
  - **Test seti** üzerinde tahmin yap ve doğruluğu orada ölç



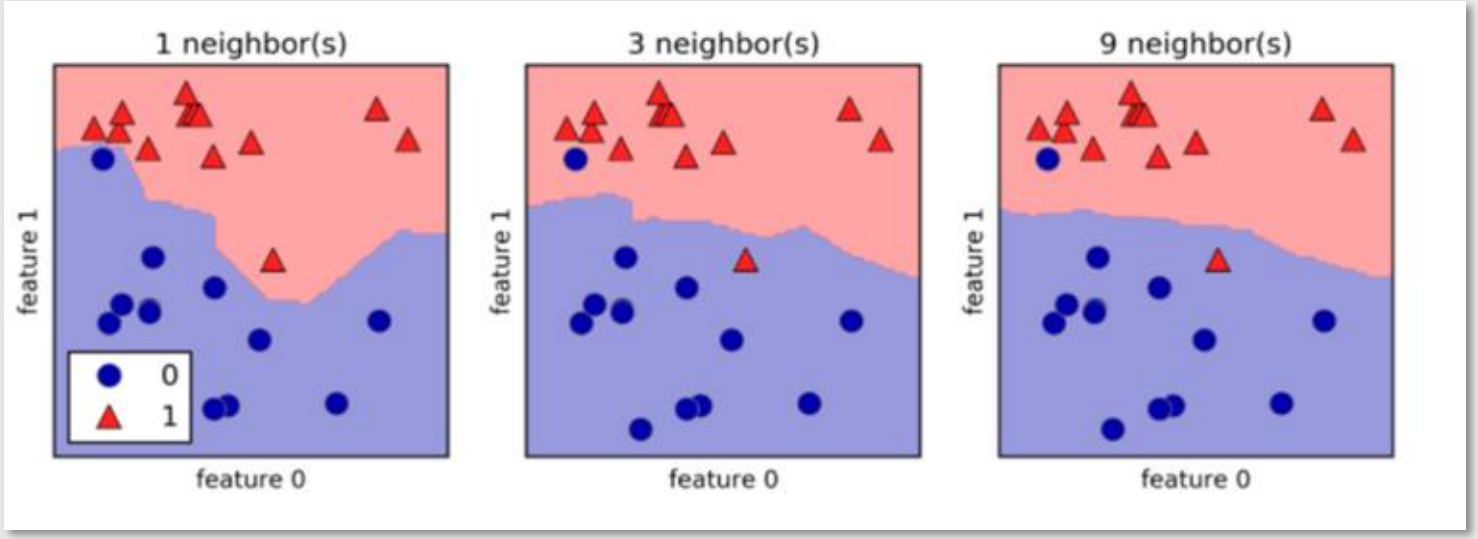
Sınıflandırma



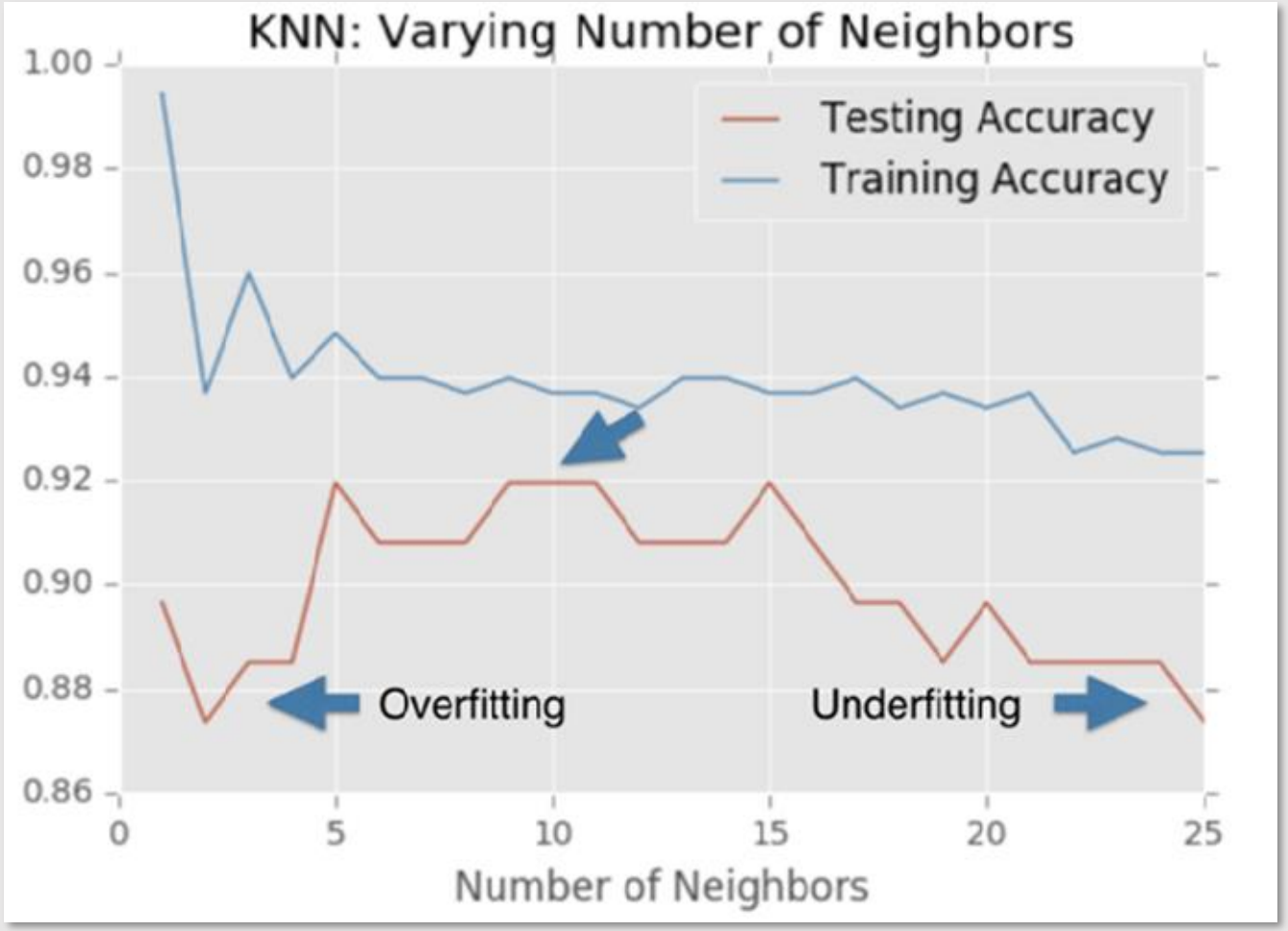
Accuracy: İlgili test grubunun yüzde kaçını doğru tahmin edebilmişim?

Sınıflandırma

Model Kompleksitesi



- Daha büyük **k** ⇒ daha pürüzsüz karar sınırları ⇒ **daha az karmaşık model**
- Daha küçük **k** ⇒ **daha karmaşık model** ⇒ aşırı uyuma (*overfitting*) yol açabilir


KNN: Varying Number of Neighbors
Testing Accuracy
Training Accuracy
Overfitting
Underfitting
1.00
0.98
0.96
0.94
0.92
0.90
0.88
0.86
0
5
10
15
20
25
Number of Neighbors

1. B. Regresyon

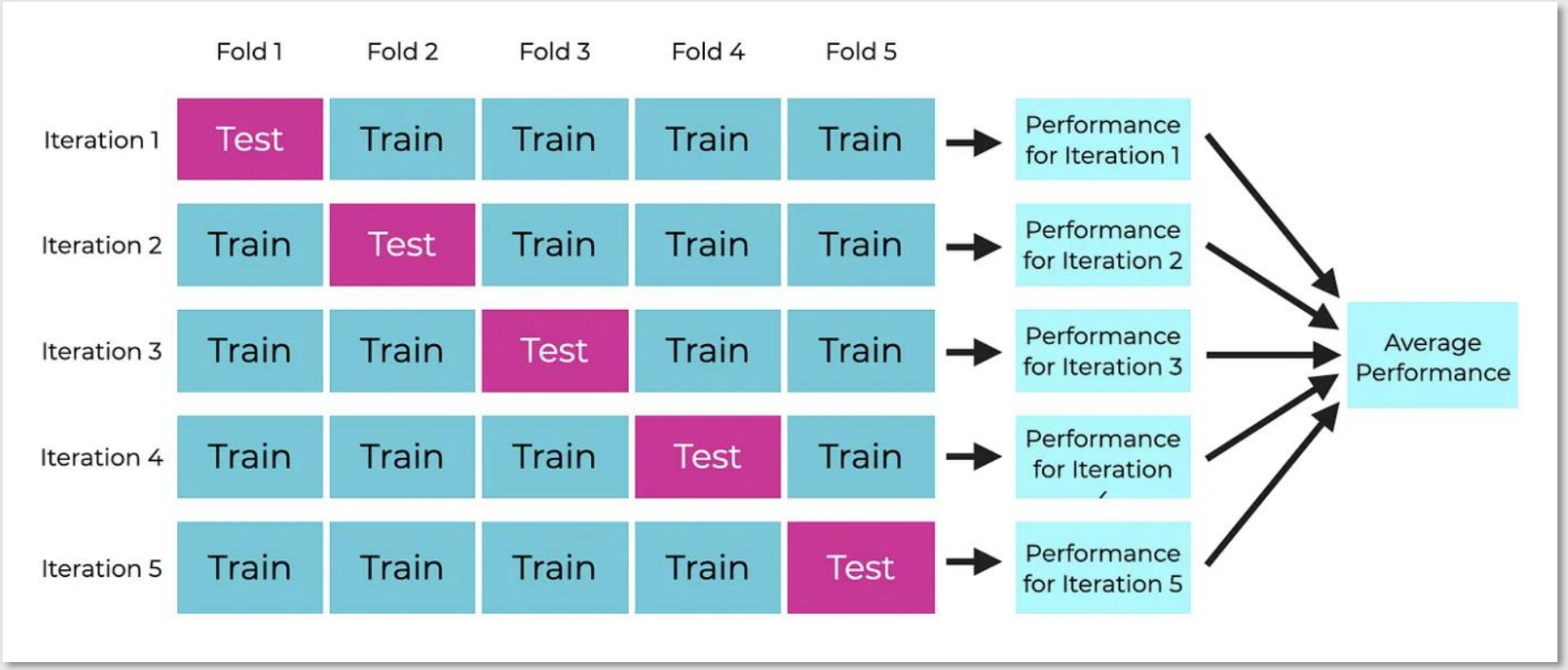
Doğrusal Regresyon

Makine öğrenmesi modellerinde gerçekten "yeni" olan ne?

- Çapraz doğrulama (cross-validation)
- Düzenlileştiriciler (regularizers)
  - Lasso
  - Ridge
- Modeli ince ayarlamak (fine-tuning)
  - ROC eğrisi
  - AUC (eğri altındaki alan) hesaplama
- Ön-işleme adımları ve bunların **pipeline**'lar içinde otomatikleştirilmesi

Doğrusal Regresyon

Çapraz doğrulama

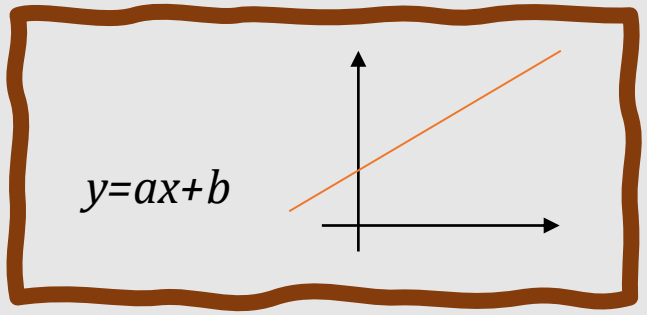


Doğrusal Regresyon

Aşağıdakilerden hangisi bir doğrusal regresyon problemidir?

1. Bir e-ticaret şirketinin, etiketlenmiş müşteri verilerini kullanarak bir müşterinin belirli bir ürünü satın alıp almayacağını tahmin etmesi.
2. Bir sağlık şirketinin, kanser tümörlerine dair verileri (örneğin geometrik ölçümlerini) kullanarak yeni bir tümörün iyi huylu mu kötü huylu mu olduğunu tahmin etmesi.
3. Bir restoranın, yorum verilerini kullanarak bir yoruma pozitif ya da negatif duygu ataması.
4. Bir bisiklet paylaşım şirketinin, zaman ve hava durumu verilerini kullanarak herhangi bir saatte kiralanan bisiklet sayısını tahmin etmesi.

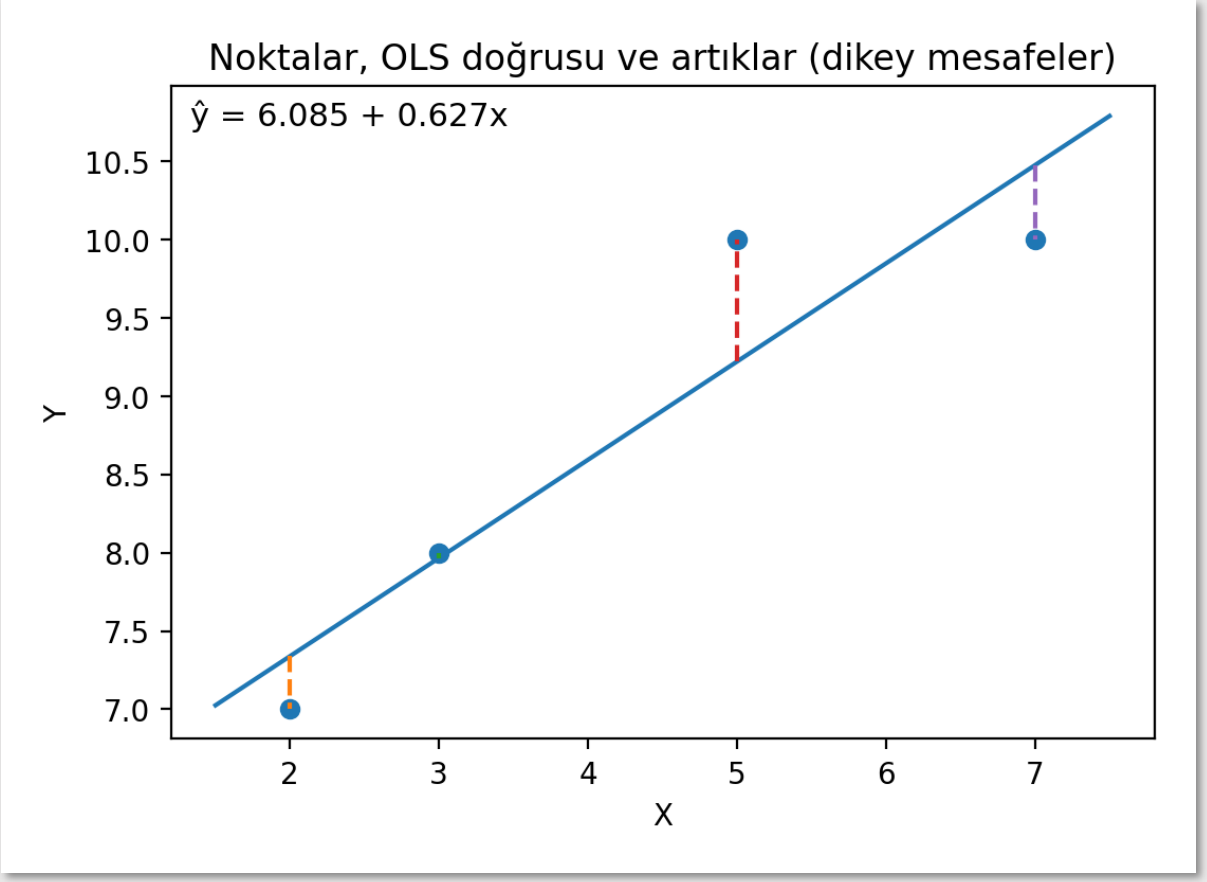
Doğrusal Regresyon



- Amacımız, **hata / kayıp fonksiyonunu en aza indiren** bir doğru bulmak.
- NEDEN doğrusal? Çünkü şunu varsayıyoruz: X'teki bir birim artış Y'yi sabit miktarda arttırıyor.
- OLS (En Küçük Kareler Yöntemi), **artıkların (hataların) kareleri toplamını** mümkün olduğunca küçültmeye çalışır.
- Klasik bir OLS modelini sklearn kullanarak çalıştırabiliriz.

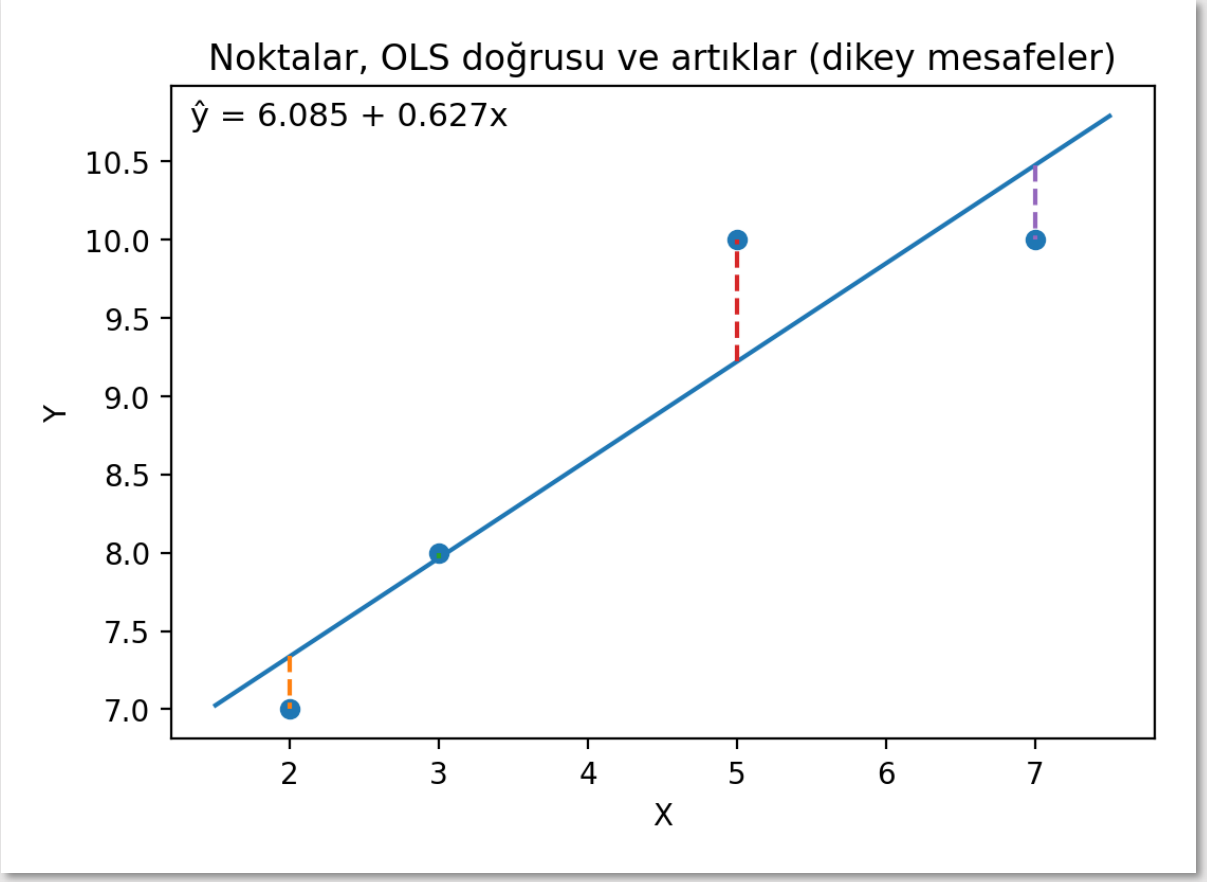
Doğrusal Regresyon

- Yaşları sırasıyla 2, 3, 5 ve 7 olan 4 kardeşin baş parmak boyları sırasıyla 7, 8, 10 ve 10 cm'dir.



Doğrusal Regresyon

- Yaşları sırasıyla 2, 3, 5 ve 7 olan 4 kardeşin baş parmak boyları sırasıyla 7, 8, 10 ve 10 cm'dir.



1 yaş artışı, yaklaşık 0.627 cm ekstra parmak boyu getirir.

Doğrunun eğimi: 0.627

Doğrunun Y eksenini kestiği değer: 6.085

Doğrusal Regresyon

x_i	y_i
2	7
3	8
5	10
7	10

$$y_i = \hat{\beta}_0 + \hat{\beta}_1 \cdot x_i + \varepsilon_i$$

hata fonksiyonu

$$\sum_i^n \underbrace{\left(y_i - \hat{\beta}_0 - \hat{\beta}_1 \cdot x_i\right)^2}_{\varepsilon_i} = \text{Min}$$

$$\hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \cdot \bar{x}$$

$$\hat{\beta}_1 = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sum (x_i - \bar{x})^2}$$

$\bar{y} = 8.75$

$\bar{x} = 4.25$

$\beta_0 = 6.085$

$\beta_1 = 0.627$

x_i	y_i	$\hat{y}_i$	ε_i
2	7	7.3	−0.3
3	8	7.96	0.004
5	10	9.2	0.8
7	10	10.47	0.47

Doğrusal Regresyon

x_i	y_i
2	7
3	8
5	10
7	10

$$y_i = \hat{\beta}_0 + \hat{\beta}_1 \cdot X_i + \varepsilon_i$$

hata fonksiyonu

$$\sum_i^n \underbrace{\left(y_i - \hat{\beta}_0 - \hat{\beta}_1 \cdot X_i\right)}_{\varepsilon_i}^2 = \text{Min}$$

$$\hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \cdot \bar{x}$$

$$\hat{\beta}_1 = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sum (x_i - \bar{x})^2}$$

$\bar{y} = 8.75$

$\bar{x} = 4.25$

$\beta_0 = 6.085$

$\beta_1 = 0.627$

x_i	y_i	$\hat{y}_i$	ε_i
2	7	7.3	−0.3
3	8	7.96	0.004
5	10	9.2	0.8
7	10	10.47	0.47

- Residual sum of square = RSS

$$\min_{\beta} \text{RSS}(\beta) = \sum_{i=1}^{n} (y_i - x_i^\top \beta)^2$$

Düzenlileştiriciler (regularizers)

- OLS mükemmel bir “overfitting" örneğidir.
- Düzenlileştirmek, temelde katsayıları cezalandırmak anlamına gelir.

Ridge:

- Kayıp fonksiyonu = OLS kayıp fonksiyonu + Ridge ceza terimi
- **Alpha (α)** modelin karmaşıklığını kontrol eder
- α = 0 olduğunda model klasik OLS’tir (overfitting riski yüksek)
- Çok yüksek α ise modeli fazla sadeleştirip **underfitting**’e yol açabilir

Lasso:

- Kayıp fonksiyonu = OLS kayıp fonksiyonu + Lasso ceza terimi
- **Alpha (α)** modelin karmaşıklığını kontrol eder
- α = 0 olduğunda model klasik OLS’tir (overfitting riski yüksek)
- Çok yüksek α ise modeli aşırı sadeleştirerek **underfitting**’e yol açabilir
- Lasso, **en önemli özellikleri** ortaya çıkarmaya ve **göreli olarak daha önemsiz** özellikleri bastırmaya / sıfırlamaya yardımcı olur.

Düzenlileştiriciler (regularizers)

- OLS mükemmel bir “overfitting" örneğidir.
- Düzenlileştirmek, temelde katsayıları cezalandırmak anlamına gelir.

Ridge:

- Kayıp fonksiyonu = OLS kayıp fonksiyonu + Ridge ceza terimi
- **Alpha (α)** modelin karmaşıklığını kontrol eder
- α = 0 olduğunda model klasik OLS’tir (overfitting riski yüksek)
- Çok yüksek α ise modeli fazla sadeleştirip **underfitting**’e yol açabilir

Lasso:

- Kayıp fonksiyonu = OLS kayıp fonksiyonu + Lasso ceza terimi
- **Alpha (α)** modelin karmaşıklığını kontrol eder
- α = 0 olduğunda model klasik OLS’tir (overfitting riski yüksek)
- Çok yüksek α ise modeli aşırı sadeleştirerek **underfitting**’e yol açabilir
- Lasso, **en önemli özellikleri** ortaya çıkarmaya ve **göreli olarak daha önemsiz** özellikleri bastırmaya / sıfırlamaya yardımcı olur.

$$\min_{\beta} \text{RSS}(\beta) = \sum_{i=1}^{n} (y_i - x_i^\top \beta)^2$$

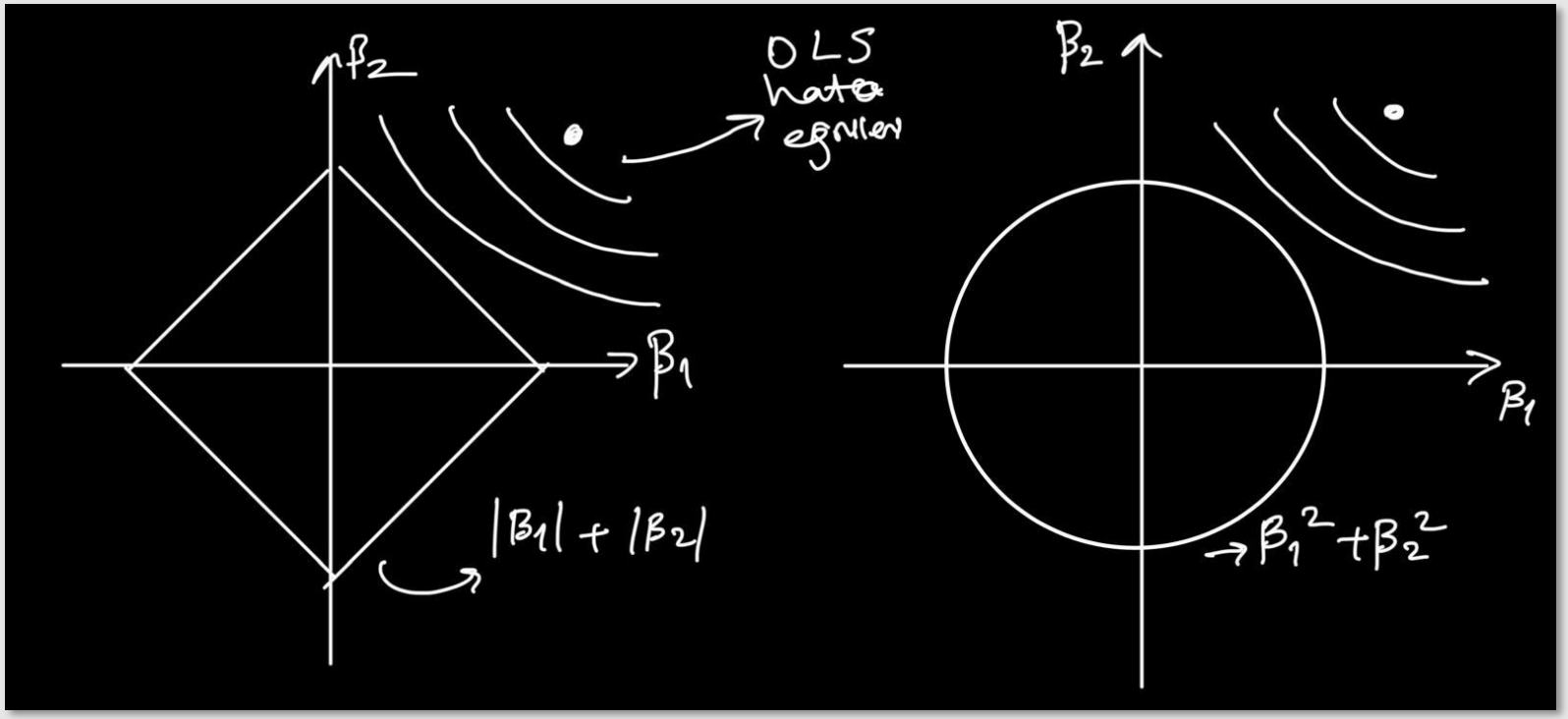
- **Lasso (L1 ceza):**

$$\min_{\beta} \text{RSS}(\beta) + \lambda \sum_{j=1}^{p} |\beta_j|$$

- **Ridge (L2 ceza):**

$$\min_{\beta} \text{RSS}(\beta) + \lambda \sum_{j=1}^{p} \beta_j^2$$

Düzenlileştiriciler (regularizers)



Lasso **Ridge**

$$\min_{\beta} \text{RSS}(\beta) = \sum_{i=1}^{n} (y_i - x_i^\top \beta)^2$$

- Lasso (L1 ceza):

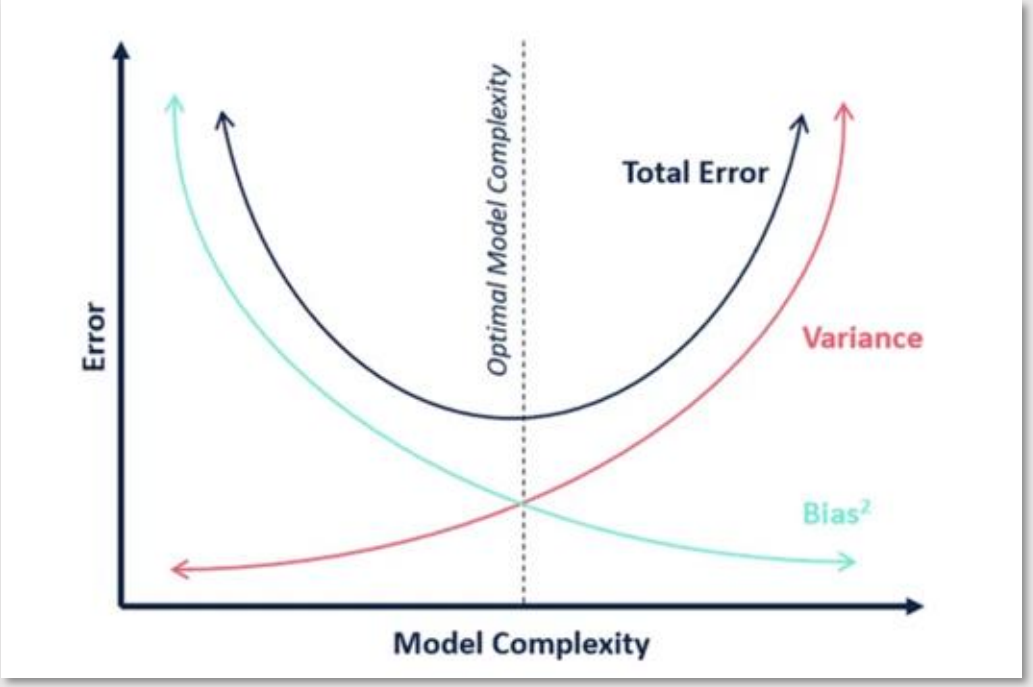
$$\min_{\beta} \text{RSS}(\beta) + \lambda \sum_{j=1}^{p} |\beta_j|$$

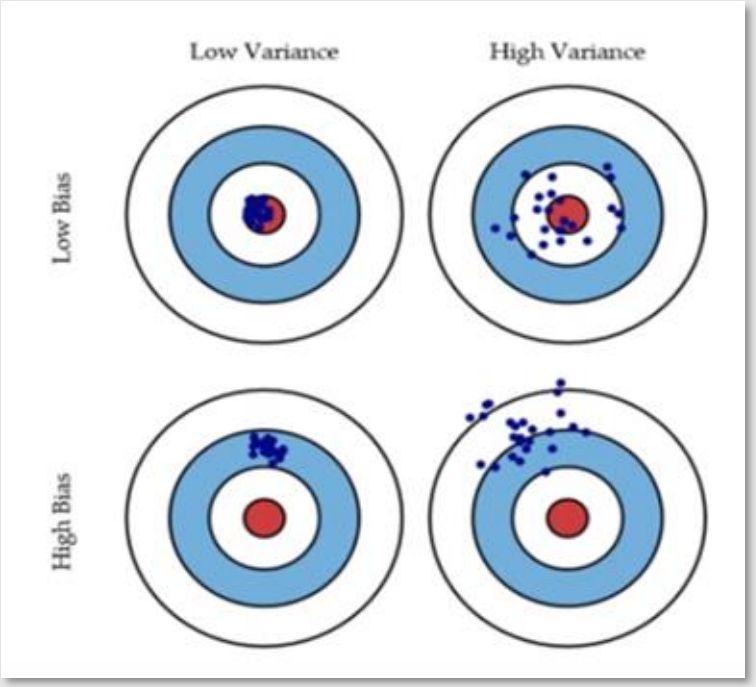
- Ridge (L2 ceza):

$$\min_{\beta} \text{RSS}(\beta) + \lambda \sum_{j=1}^{p} \beta_j^2$$

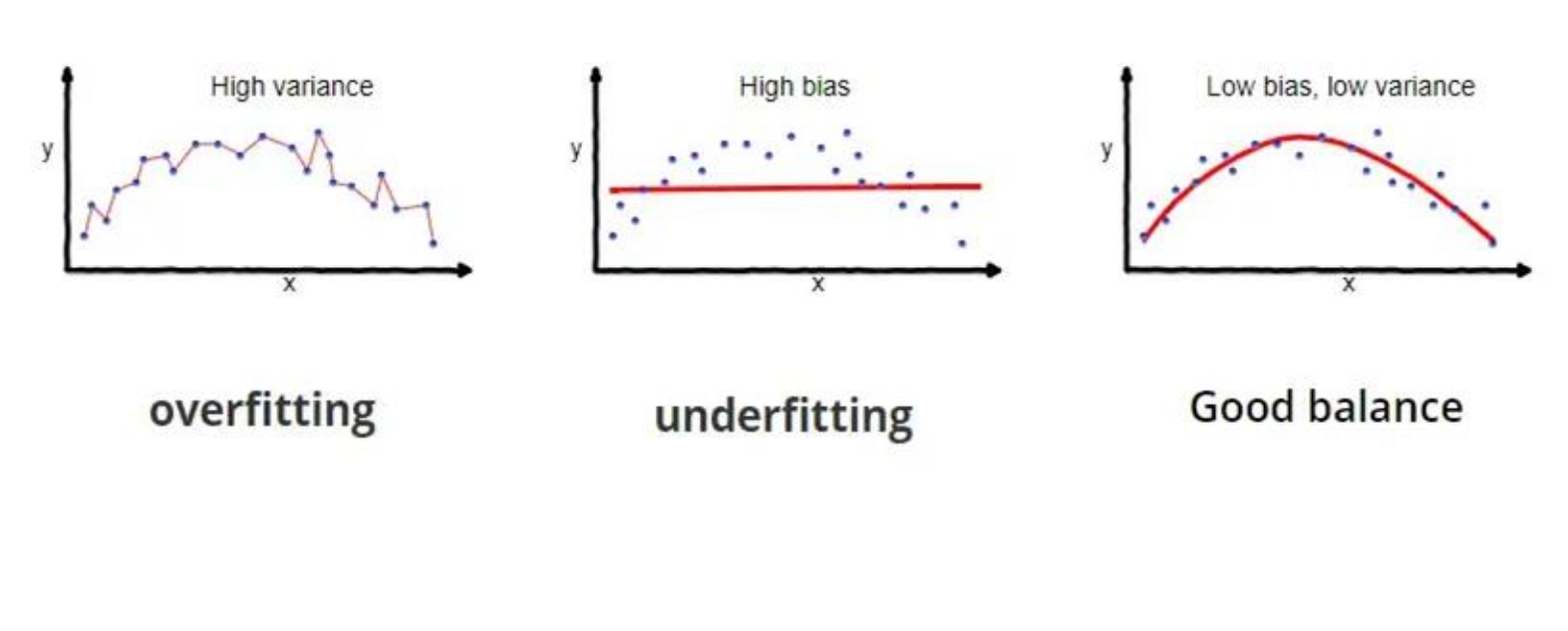
Lasso'nun mutlak değerli toplamı gösteren düz çizgileriyle OLS'in eliptik hata eğrilerinin, Beta'lardan bazılarının sıfır olduğu yerlerde kesişmesi matematiksel olarak mümkündür. Ancak bu durum Ridge için geçerli değildir. Pratikteyse bu ikisi genelde birlikte kullanılır (bknz: elastik net modelleri).

Düzenlileştiriciler (regularizers)





Düzenlileştiriciler (regularizers)



1. C. Performans Testleri ve Model Geliştirme

Modelin Performansı

Modelin ne kadar iyi?
Doğruluğu (**accuracy**) ne kadar?

- Spam sınıflandırması örneği: E-postaların %1'i spam, %99'u gerçek (spam değil).
  - Eğer model tüm e-postaları "**gerçek**" diye sınıflandırırsa, model %99 doğru görünür!
- Buna sınıf dengesizliği (**class imbalance**) denir.
  Bu yüzden daha nüanslı/ayırt edici metriklere ihtiyacımız var.

Karmaşıklık matrisi (Confusion matrix):

- **Accuracy** (Doğruluk) = (TP + TN) / (TP + TN + FP + FN)
- **Precision** (Kesinlik / Pozitif öngörü değeri) = TP / (TP + FP)
- **Recall** (Duyarlılık / Yakalama oranı) = TP / (TP + FN)
- **F1-score** = 2 × (Precision × Recall) / (Precision + Recall)

	Predicted: Spam Email	Predicted: Real Email
Actual: Spam Email	True Positive	False Negative
Actual: Real Email	False Positive	True Negative

Kısaltmalar: **TP**=Doğru Pozitif, **TN**=Doğru Negatif, **FP**=Yanlış Pozitif, **FN**=Yanlış Negatif.

Lojistik Regresyon ve ROC eğrisi

- Lojistik regresyon, sınıflandırmaya benzer bir mantıkla çalışır: her gözlem için bir olasılık tahmin eder (ör. $p > 0.5$ ise etiketi 1 yap).

- ROC eğrisi (eşik değeri değiştirerek)

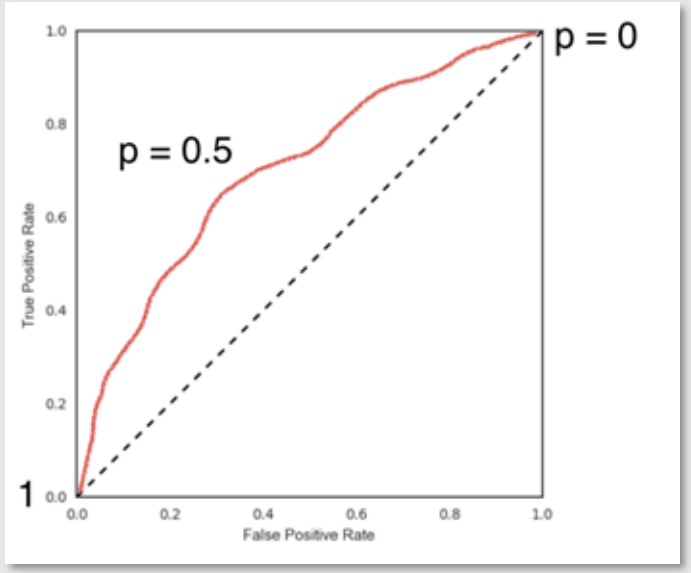
- **FPR (False Positive Rate / Yanlış Pozitif Oranı):**
0'a gitmesi gerektiği halde 1'e gidenlerin oranı
(Yani "negatifleri ne kadar yanlış alarm vererek pozitif ilan ediyorum?")
- **TPR (True Positive Rate / Doğru Pozitif Oranı, Recall / Duyarlılık):**
1'e gitmesi gerekenler içerisinden 1'e gidebilmiş olanların oranı
(Yani "pozitifleri ne kadarını yakalayabiliyorum?")

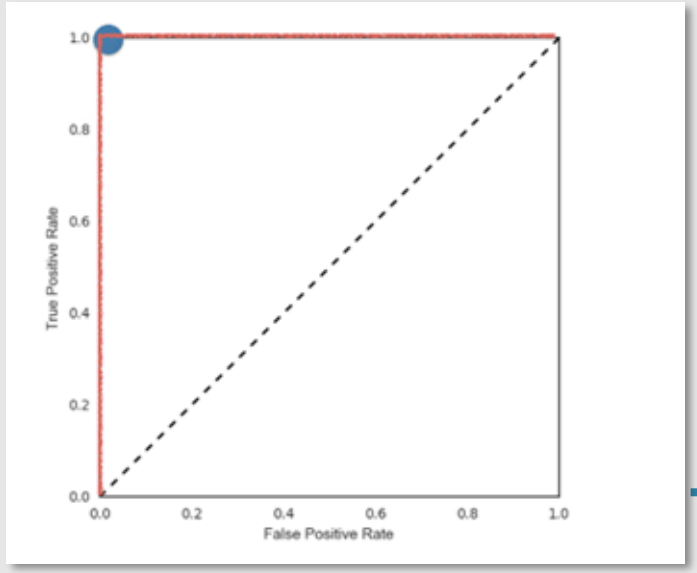
Lojistik Regresyon ve ROC eğrisi

- Lojistik regresyon, sınıflandırmaya benzer bir mantıkla çalışır: her gözlem için bir olasılık tahmin eder (ör. p > 0.5 ise etiketi 1 yap).
- ROC eğrisi (eşik değeri değiştirerek)

- **FPR (False Positive Rate / Yanlış Pozitif Oranı):**
0'a gitmesi gerektiği halde 1'e gidenlerin oranı
(Yani "negatifleri ne kadar yanlış alarm vererek pozitif ilan ediyorum?")
- **TPR (True Positive Rate / Doğru Pozitif Oranı, Recall / Duyarlılık):**
1'e gitmesi gerekenler içerisinden 1'e gidebilmiş olanların oranı
(Yani "pozitifleri ne kadarını yakalayabiliyorum?")

ROC eğrisinin altındaki alan (AUC) ne kadar büyükse, model o kadar iyidir.





Lojistik Regresyon ve ROC eğrisi

A = 0 (negative)
B = 1 (positive)

Gerçek pozitif sayısı P = 5
Gerçek negatif sayısı N = 5

Indeks:	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***10***
Gerçek sınıf:	A	A	B	A	A	B	A	B	B	B
Modelin tahmin ettiği olasılık:	*0.1*	*0.2*	*0.3*	*0.4*	*0.5*	*0.6*	*0.7*	*0.8*	*0.9*	*1*

Lojistik Regresyon ve ROC eğrisi

A = 0 (negative)
B = 1 (positive)

Gerçek pozitif sayısı P = 5
Gerçek negatif sayısı N = 5

Indeks:	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***10***
Gerçek sınıf:	A	A	B	A	A	B	A	B	B	B
Modelin tahmin ettiği olasılık:	*0.1*	*0.2*	*0.3*	*0.4*	*0.5*	*0.6*	*0.7*	*0.8*	*0.9*	*1*

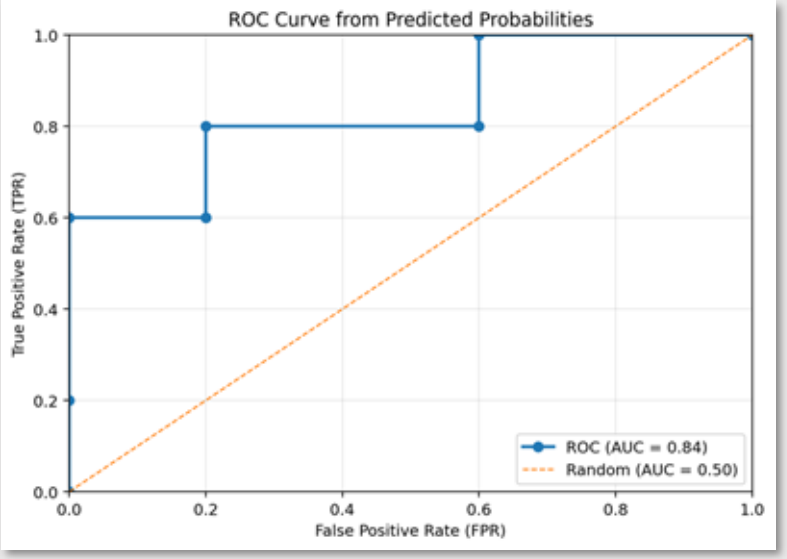
Threshold (t)	**Pozitif tahmin edilenler (score ≥ t)**	**TP**	**FP**	**TPR**	**FPR**
>1.0	–	0	0	0.00	0.00
≥1.0	10	1	0	0.20	0.00
≥0.9	9–10	2	0	0.40	0.00
≥0.8	8–10	3	0	0.60	0.00
≥0.7	7–10	3	1	0.60	0.20
≥0.6	6–10	4	1	0.80	0.20
≥0.5	5–10	4	2	0.80	0.40
≥0.4	4–10	4	3	0.80	0.60
≥0.3	3–10	5	3	1.00	0.60
≥0.2	2–10	5	4	1.00	0.80
≥0.1	1–10	5	5	1.00	1.00

Lojistik Regresyon ve ROC eğrisi

A = 0 (negative)
B = 1 (positive)

Gerçek pozitif sayısı P = 5
Gerçek negatif sayısı N = 5

Indeks:	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***10***
Gerçek sınıf:	A	A	B	A	A	B	A	B	B	B
Modelin tahmin ettiği olasılık:	*0.1*	*0.2*	*0.3*	*0.4*	*0.5*	*0.6*	*0.7*	*0.8*	*0.9*	*1*



Threshold (t)	Pozitif tahmin edilenler (score ≥ t)	TP	FP	TPR	FPR
>1.0	–	0	0	0.00	0.00
≥1.0	10	1	0	0.20	0.00
≥0.9	9–10	2	0	0.40	0.00
≥0.8	8–10	3	0	0.60	0.00
≥0.7	7–10	3	1	0.60	0.20
≥0.6	6–10	4	1	0.80	0.20
≥0.5	5–10	4	2	0.80	0.40
≥0.4	4–10	4	3	0.80	0.60
≥0.3	3–10	5	3	1.00	0.60
≥0.2	2–10	5	4	1.00	0.80
≥0.1	1–10	5	5	1.00	1.00

Hiperparametre Ayarı

- **Ridge** ve **Lasso** regresyonu / alpha seçimi
- **k-En Yakın Komşu** (k-NN) / n_neighbors seçimi
  - **k** ve **alpha** gibi parametreler hiperparametredir.

Doğru hiperparametre nasıl seçilir?

- Birçok hiperparametre değeri dene; her birini ayrı ayrı eğit; performanslarını karşılaştır; en iyi performans vereni seç.

  - **Grid search** (çapraz doğrulama ile)
  - **Randomized search** (çapraz doğrulama ile)

Ön İşleme ve Boru Hattı (Pipeline)

- **Gerçek dünyadaki veri**, en iyi modeli hemen çalıştırabilmen için **hazır gelmez**.
- **Kategorik değişkenleri** dummy (one-hot) değişkenlere çevir.
- **Eksik veriyi yönet**
  - Satırları/gözlemleri silersen: **veri kaybı**
  - O yüzden: **impute (tamamlama) yapabilirsin!**
    - Eksik olmayan değerlerin ortalamasıyla doldurma
    - Zaman boyutu varsa doğrusal (linear) imputasyon
    - Scikit-learn: Imputer

```
from sklearn.preprocessing import Imputer
imp = Imputer(missing_values='NaN', strategy='mean', axis=0)
imp.fit(X)
X = imp.transform(X)
```

Pipeline: Hepsini tek seferde yap!

Impute, normalize et, çalıştır, fit et, tahmin et, test et...

2. Gözetimsiz Öğrenme

Gözetimsiz Öğrenme

- Gözetimli öğrenme (supervised learning) belirli bir tahmin görevi için örüntüler bulurken, gözetimsiz öğrenme (unsupervised learning) belirli bir tahmin hedefi olmadan verinin içindeki örüntüleri keşfeder.

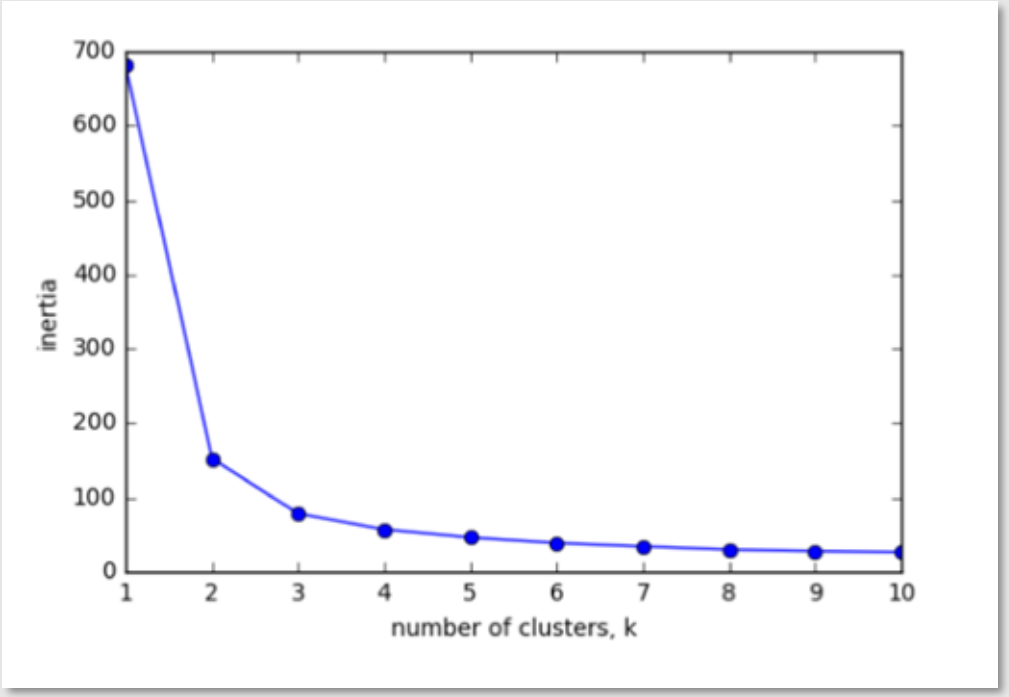
- Diyelim ki elimizde sadece özellikler (target/etiket olmadan) var:
- Gözetimsiz bir model çalıştırırız ve model, veride zaten var olan doğal kümeleri (clusters) kendisi keşfeder.

k-means kümeleme (clustering)

- kNN sınıflandırmaya benzer; ama burada etiket yoktur.
- Model etiketleri/küme atamalarını kendi kendine üretir.

Kümeleme

- **Kümelerimizin kalitesini ölçmek:** *Inertia!*
- İyi bir kümeleme, **sıkı (tight) kümelere** sahiptir (bu genelde **daha fazla küme** ve **daha düşük inertia** demektir), ama **çok fazla küme** de istemeyiz.
- Bu yüzden (sıklıkla önerildiği gibi) **dirsek (elbow) noktasını** seçeriz.



Neler var neler?

- **Naive Bayes Classification**
- **Support Vector Machines**
- **Decision Trees and Random Forests**
- **Principal Component Analysis**
- Manifold Learning
- Gaussian Mixture Models
- Kernel Density Estimation

Derin Öğrenme

- **Neural Networks**