SAKARYA ÜNİVERSİTESİ

Yapay Zekaya Giriş
HAFTA 5
Makine Öğrenmesi Algoritmaları

Dr. Öğretim Üyesi Burcu ÇARKLI YAVUZ

bcarkli@sakarya.edu.tr

Makine öğrenmesi yöntemleri

- Makine öğrenmesi (ML) yöntemleri genellikle gözetimli, gözetimsiz, yarı gözetimli ve pekiştirmeli öğrenme olarak dört ana kategoriye ayrılır.
- Her bir yöntem, belirli türdeki problemleri çözmek için kullanılır ve uygulama alanlarına göre farklılık gösterir.

1. Gözetimli Öğrenme (Supervised Learning)

- Gözetimli öğrenme algoritmaları, girdi ve çıktı örnekleri arasındaki ilişkiyi modellemeye çalışır. Bu model, daha sonra yeni girdiler için çıktılar üretir.
- Çıktı değişkeninin türüne göre sınıflandırma ve regresyon görevleri olarak incelenebilir.
- **Sınıflandırma Görevleri:** Çıktı değişkeni kategorik (nominal veya ordinal) değerler alır. Bir gözetimli öğrenme modeli, verilen girdilere dayanarak hangi kategoriye ait olduklarını tahmin etmeyi amaçlar. Örneğin, e-postaları "spam" veya "spam değil" olarak sınıflandırmak veya tıbbi görüntülerdeki hücreleri "kanserli" veya "kansersiz" olarak etiketlemek sınıflandırma görevlerindendir.
- **Regresyon Görevleri:** Çıktı değişkeni sürekli (sayısal) değerlerdir. Burada amaç, girdi değişkenlerine dayanarak bir sayısal değer tahmini yapmaktır. Örneğin, bir evin özelliklerine (boyut, konum, oda sayısı vb.) dayanarak fiyatını tahmin etmek regresyon görevine bir örnektir.

1. Gözetimli Öğrenme (Supervised Learning)

➢ **Örnek algoritmalar:**

- Karar Ağaçları (Decision Trees)
- Rastgele Ormanlar (Random Forests)
- Destek Vektör Makineleri (Support Vector Machines)
- Yapay Sinir Ağları (Artificial Neural Networks)
- K-En Yakın Komşu (KNN, K-Nearest Neighbor)
- Lojistik Regresyon (Logistic Regression)
- Lineer Regresyon (Linear Regression)

Sınıflandırma Örneği: kNN

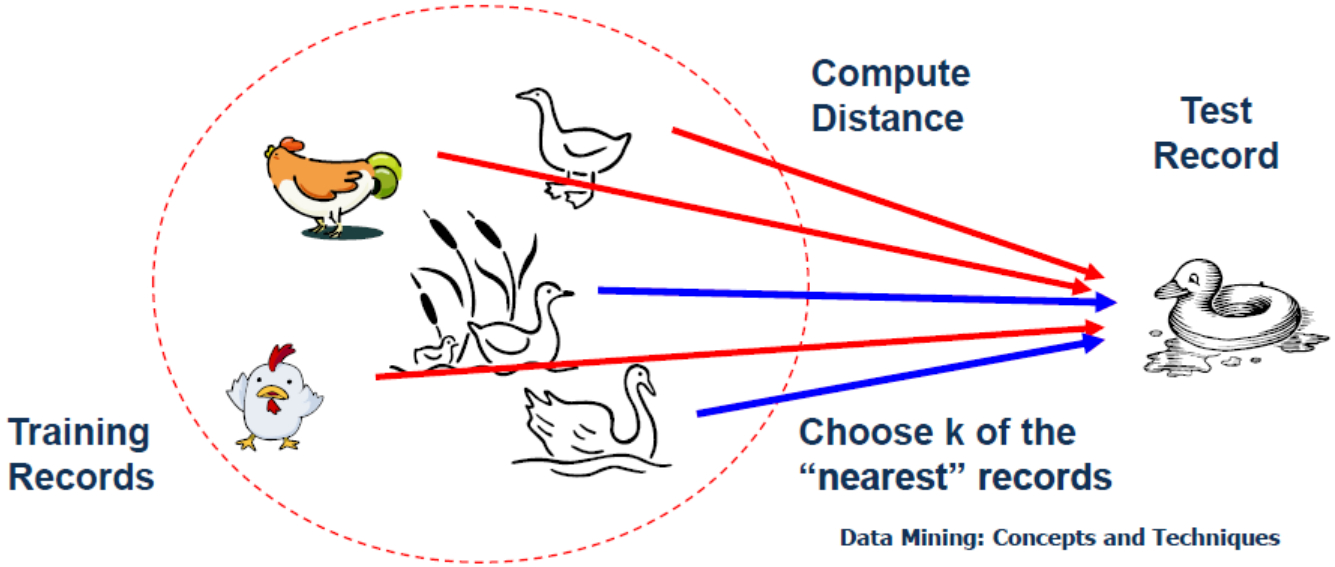
K-en yakın komşu algoritması (kNN)

- Uygulama kolaylığı nedeni ile tercih edilen, yaygın kullanıma sahip bir algoritmadır.
- Sınıfları belli olan bir veri kümesine sınıfı bilinmeyen bir veri dahil olduğunda, bu verinin kendisine en yakın k adet sınıftan frekansı fazla olan sınıfa atanması prensibine dayanır.

Sınıflandırma Örneği: kNN

Temel kNN Fikri

- Eğer ördek gibi yürüyor ve ördek sesi çıkarıyorsa muhtemelen ördektir.



Sınıflandırma Örneği: kNN

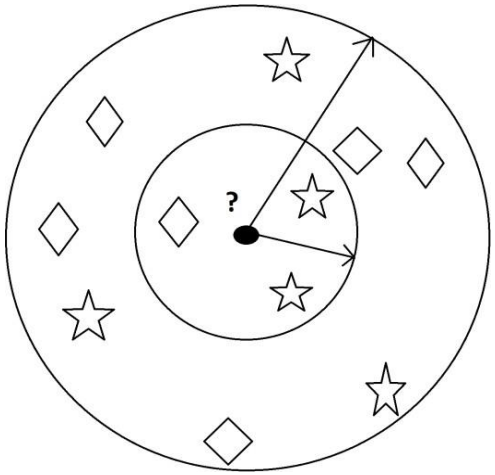
kNN ile sınıflandırma yapılırken işlem adımları:

- Öncelikle k değeri belirlenir, k değeri en yakın komşu sayısını ifade eder.
- Sınıfı belirlenecek olan gözlem değeri ile sınıfı önceden bilinen tüm değerlerin mesafeleri hesaplanır.
- En küçük mesafeye sahip k adet veri belirlenir ve bunların çoğunluğunun dahil olduğu sınıf seçilir.
- Seçilen sınıf, gözlem değerinin ait olduğu sınıf olarak kabul edilir.

Sınıflandırma Örneği: kNN

k değerinin belirlenmesi:

- kNN algoritmasında k değerinin belirlenmesi önemlidir; k çok büyük seçilirse özellikleri çok benzemeyen elemanlar aynı kümeye dahil edilebilir.
- Şekilde k=3 ve k=11 seçildiğinde farklı sınıf değerlerinin karşımıza çıktığı görülmektedir.



Sınıflandırma Örneği: kNN

X ve Y noktaları arasındaki Uzaklık hesabı:

Öklid uzaklığı: $D(x,y)= \sqrt{\sum_{k=1}^{p}(X_k - Y_k)^2}$

Manhattan uzaklığı: $D(x,y)= \sum_{k=1}^{p} |X_k - Y_k|$

Minkowski uzaklığı: $D(x,y)= \left(\sum_{k=1}^{p}(|X_k - Y_k|)^q\right)^{1/q}$

Minkowski uzaklık ölçütünde q=1 ise Manhattan, q=2 ise Öklid uzaklığı elde edilir

Hamming uzaklığı: $D(x,y)= \sum_{k=1}^{p} \delta_k, \quad \delta_k = \begin{cases} 1, X_k \neq Y_k \; ise \\ 0, X_k = Y_k \; ise \end{cases}$

Sınıflandırma Örneği: kNN

- Verilen gözlem tablosunda X ve Y niteliklerinden Z sınıfı oluşmaktadır.
- Bu gözlem değerlerine bağlı olarak yeni bir gözlem olan X=7 ve Y=3 yani (7,3) gözleminin hangi sınıfa dahil olduğunu k=4 için kNN algoritması ile bulalım.

X	Y	Z
1	3	Negatif
2	5	Pozitif
2	3	Pozitif
3	9	Negatif
4	7	Negatif
5	2	Pozitif
6	8	Pozitif
8	6	Negatif
10	6	Negatif
9	1	Negatif

Sınıflandırma Örneği: kNN

(7,3) noktasının tüm gözlem değerleri ile uzaklıklarını hesaplayalım:

$$d(1) = \sqrt{(1-7)^2 + (3-3)^2} = 6.00$$

$$d(2) = \sqrt{(2-7)^2 + (5-3)^2} = 5{,}39$$

$$d(3) = \sqrt{(2-7)^2 + (3-3)^2} = 5{,}00$$

$$\vdots$$

X	Y	Uzaklık
1	3	6
2	5	5,39
2	3	5
3	9	7,21
4	7	5
5	2	2,24
6	8	5,10
8	6	3,16
10	6	4,24
9	1	2,83

Sınıflandırma Örneği: kNN

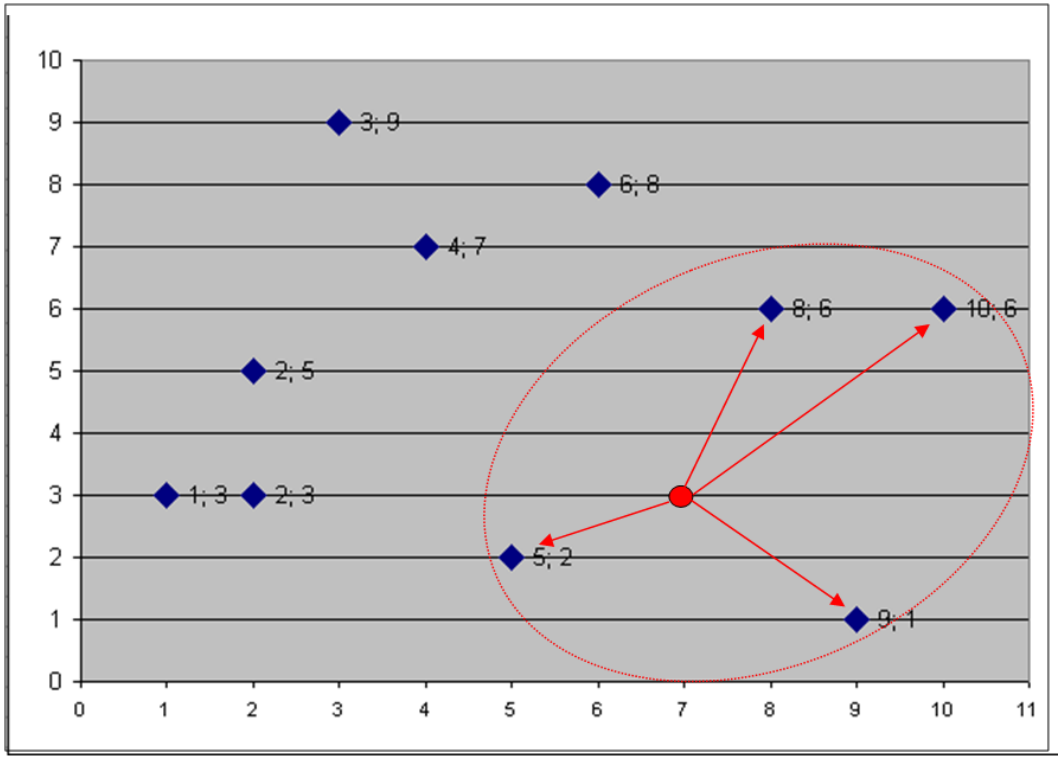
En küçük k=4 tane uzaklığı belirleyelim:

- Belirlenen dört nokta **(7,3)** noktasına en yakın değerlerdir. Gözlem değerlerin içinde **bir pozitif** ve **üç negatif** değer olduğundan **(7,3)** noktasının sınıfı **negatif** olarak belirlenir.
- **Not:** 2 adet sınıf varsa, en yakın k adet sınıf etiketinin eşit sayıda farklı sınıf içermesinden kaçınmak için tek sayıda k değeri seçiniz.

X	Y	Uzaklık	Sıralama	Z
1	3	6	9	Negatif
2	5	5,39	8	Pozitif
2	3	5	6	Pozitif
3	9	7,21	10	Negatif
4	7	5	5	Negatif
5	2	2,24	1	Pozitif
6	8	5,10	7	Pozitif
8	6	3,16	3	Negatif
10	6	4,24	4	Negatif
9	1	2,83	2	Negatif

Sınıflandırma Örneği: kNN

Komşulukları grafik ile inceleyecek olursak:



2. Gözetimsiz Öğrenme (Unsupervised Learning)

- Etiketsiz veriler üzerinde çalışır ve veri setindeki gizli yapıları veya kalıpları bulmayı amaçlar.
- Kümeleme ve boyut azaltma olarak iki alt görevde incelenebilir.

- **Kümeleme:** Hangi nesnenin hangi sınıfa ait olduğu ve grup sayısı belirsizdir. Verileri benzerliklerine göre gruplara ayırır.
  - **Kullanım Alanları:** Müşteri pazar segmentasyonu, sosyal ağ analizi, pazar araştırması.
- **Boyut Azaltma:** Veriyi daha yönetilebilir ve anlaşılır hale getirir.
  - **Kullanım Alanları:** Görsel veri sıkıştırma, gürültü azaltma, büyük veri setlerinde özellik çıkarımı.

2. Gözetimsiz Öğrenme (Unsupervised Learning)

➢**Örnek Algoritmalar:**

- K-Means Kümeleme (K-Means Clustering)
- Hiyerarşik Kümeleme (Hierarchical Clustering)
- Beklenti Maksimizasyonu (Expectation Maximization)
- Prensip Bileşen Analizi (Principal Component Analysis)

Kümeleme örneği: K-means

- Kümeleme algoritmalarının en basitidir.
- Veriyi en iyi ifade edecek K adet vektör bulmaya çalışır.
- K sayısı kullanıcı tarafından verilir.
- Uygun k sayısının belirlenmesi sorun olabilir.
- K sayısını kendi hesaplayan x-means algoritması da kullanılabilir.
- Sadece sayısal veriler ile çalışır.
- Bu metod çok geniş veritabanları üzerinde de uygulanabilir. Çünkü karmaşıklığı oldukça azdır.

Kümeleme örneği: K-means

- K-Means algoritması, veritabanındaki n tane nesnenin k adet kümeye bölümlenmesini sağlar. Kümeleme sonucu küme içi (intra-cluster) elamanlar arasındaki benzerlikler çok iken, kümeler arası (inter-cluster) elamanların benzerlikleri çok düşüktür.
- K-means kümeleme yönteminin değerlendirilmesinde en yaygın olarak karesel hata kriteri SSE (sum of squared errors) kullanılır. En düşük SSE değerine sahip kümeleme sonucu en iyi sonucu verir.

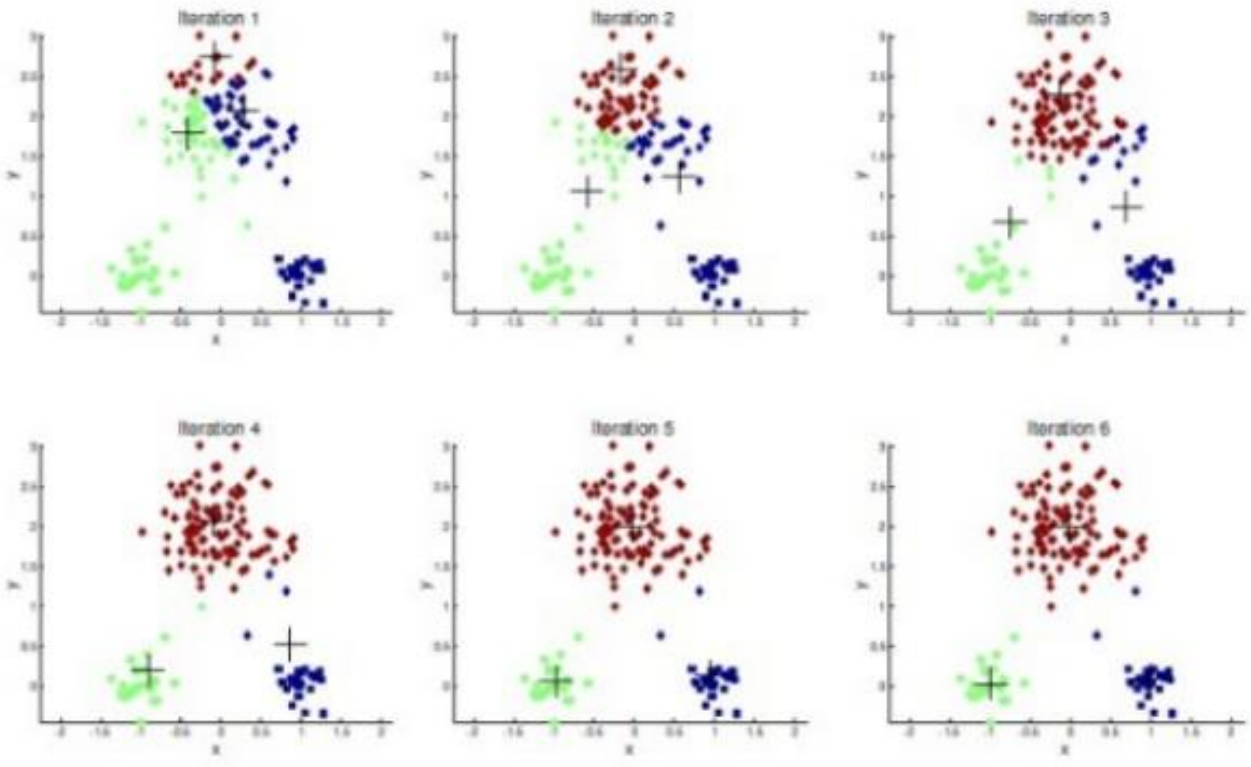
$$SSE = \sum_{i=1}^{K} \sum_{x \in C_i} dist^2(m_i, x)$$

Kümeleme örneği: K-means

K-means Algoritmasının adımları

1. *Kaç tane küme oluşturulacağı **k** sayısı ile belirlenir.*
2. *K adet nokta küme merkezi olarak atanır.*
3. *Her gözlem en yakın merkez noktanın olduğu kümeye dâhil edilir.*
4. *Her nesnenin ortalaması hesaplanır. Merkez nokta kümedeki nesnelerin ortalaması olacak şekilde güncellenir.*
5. *Nesnelerin kümelemesinde değişiklik olmayana kadar adım 2'ye geri dönülür.*

Kümeleme örneği: K-means



Kümeleme örneği: K-means

- Aşağıdaki 8 nokta için 3 küme elde ediniz

A1(2, 10)
A2(2, 5)
A3(8, 4)
A4(5, 8)
A5(7, 5)
A6(6, 4)
A7(1, 2)
A8(4, 9)

İlk küme merkezleri A1(2, 10), A4(5, 8) ve A7(1, 2) olsun.

İki nokta arasındaki uzaklık değerlerini aşağıdaki formülle hesaplayalım:

a=(x1, y1) ve *b=(x2, y2)* ;

$\rho(a, b) = |x2 - x1| + |y2 - y1|$.

Kümeleme örneği: K-means

1.İterasyon

		(2, 10)	(5, 8)	(1, 2)	
	Nokta	1.küme	2.küme	3.küme	Küme
A1	(2, 10)				
A2	(2, 5)				
A3	(8, 4)				
A4	(5, 8)				
A5	(7, 5)				
A6	(6, 4)				
A7	(1, 2)				
A8	**(4, 9)**				

Kümeleme örneği: K-means

- Her gözlem kendine en yakın merkez noktanın olduğu kümeye dahil edilir.
- (2,10) örneğinin üç küme merkezine uzaklıklarının hesaplanması:

nokta	merkez1
x1, y1	*x2, y2*
(2, 10)	(2, 10)

$\rho(a, b) = |x2 - x1| + |y2 - y1|$

$\rho(\text{nokta, merkez1}) = |x2 - x1| + |y2 - y1|$
$= |2 - 2| + |10 - 10|$
$= 0 + 0$
$= 0$

x1, y1	*x2, y2*
(2, 10)	(5, 8)

$\rho(a, b) = |x2 - x1| + |y2 - y1|$

$\rho(\text{nokta, merkez2}) = |x2 - x1| + |y2 - y1|$
$= |5 - 2| + |8 - 10|$
$= 3 + 2$
$= 5$

x1, y1	*x2, y2*
(2, 10)	(1, 2)

$\rho(a, b) = |x2 - x1| + |y2 - y1|$

$\rho(\text{nokta, merkez3}) = |x2 - x1| + |y2 - y1|$
$= |1 - 2| + |2 - 10|$
$= 1 + 8$
$= 9$

Kümeleme örneği: K-means

- Elde edilen verileri tabloya yerleştirelim.

1.İterasyon

		(2, 10)	(5, 8)	(1, 2)	
	Nokta	**1.küme**	**2.küme**	**3.küme**	Küme
A1	(2, 10)	0	5	9	**1**
A2	(2, 5)				
A3	(8, 4)				
A4	(5, 8)				
A5	(7, 5)				
A6	(6, 4)				
A7	(1, 2)				
A8	**(4, 9)**				

1.küme	2.küme	3.küme
(2, 10)		

Kümeleme örneği: K-means

- (2,5) örneğinin üç küme merkezine uzaklıklarının hesaplanması:

x1, y1 (2, 5) *x2, y2* (2, 10)

$\rho(a, b) = |x2 - x1| + |y2 - y1|$

$\rho(\text{nokta, merkez1}) = |x2 - x1| + |y2 - y1|$
$= |2 - 2| + |10 - 5|$
$= 0 + 5$
$= 5$

x1, y1 (2, 5) *x2, y2* (5, 8)

$\rho(a, b) = |x2 - x1| + |y2 - y1|$

$\rho(\text{nokta, merkez2}) = |x2 - x1| + |y2 - y1|$
$= |5 - 2| + |8 - 5|$
$= 3 + 3$
$= 6$

x1, y1 (2, 5) *x2, y2* (1, 2)

$\rho(a, b) = |x2 - x1| + |y2 - y1|$

$\rho(\text{nokta, merkez3}) = |x2 - x1| + |y2 - y1|$
$= |1 - 2| + |2 - 5|$
$= 1 + 3$
$= 4$

Kümeleme örneği: K-means

- Elde edilen verileri tabloya yerleştirelim.

1.İterasyon

		(2, 10)	(5, 8)	(1, 2)	
	Nokta	**1.küme**	**2.küme**	**3.küme**	Küme
A1	(2, 10)	0	5	9	**1**
A2	(2, 5)	5	6	4	**3**
A3	(8, 4)				
A4	(5, 8)				
A5	(7, 5)				
A6	(6, 4)				
A7	(1, 2)				
A8	**(4, 9)**				

1.küme	2.küme	3.küme
(2, 10)		**(2, 5)**

Kümeleme örneği: K-means

- Bu şekilde devam ederek tabloyu tamamlayalım.

1.İterasyon

		(2, 10)	(5, 8)	(1, 2)	
	Nokta	**1.küme**	**2.küme**	**3.küme**	Küme
A1	(2, 10)	0	5	9	**1**
A2	(2, 5)	5	6	4	**3**
A3	(8, 4)	12	7	9	**2**
A4	(5, 8)	5	0	10	**2**
A5	(7, 5)	10	5	9	**2**
A6	(6, 4)	10	5	7	**2**
A7	(1, 2)	9	10	0	**3**
A8	**(4, 9)**	**3**	**2**	**10**	**2**

1.küme	2.küme	3.küme
(2, 10)	(8, 4)	(2, 5)
	(5, 8)	(1, 2)
	(7, 5)	
	(6, 4)	
	(4, 9)	

Kümeleme örneği: K-means

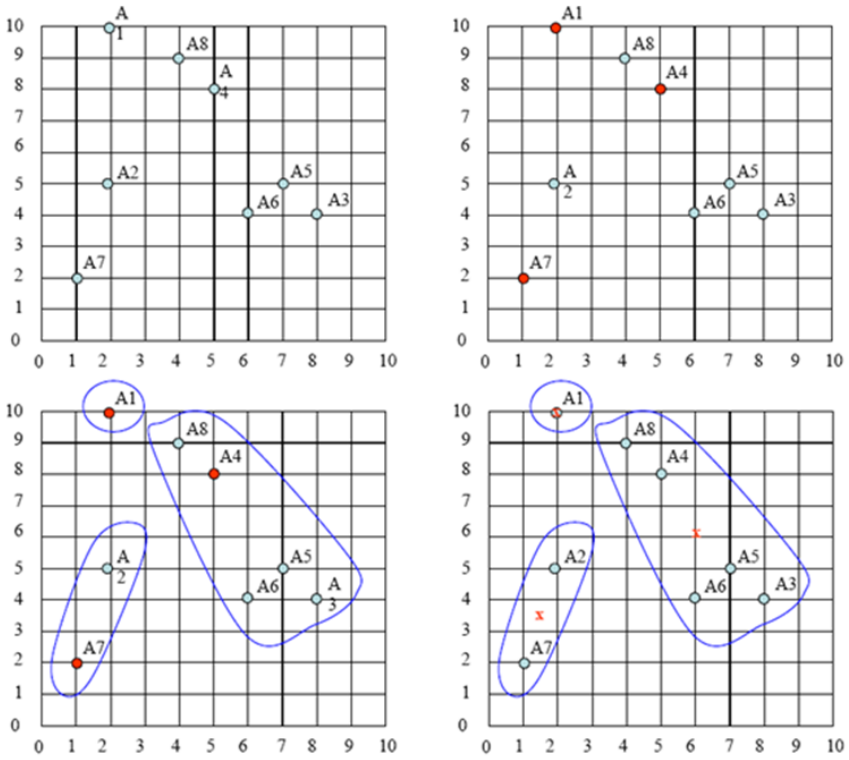
Yeni küme merkezlerini hesaplayalım:

1.küme için A1(2, 10)
2.küme için , ((8+5+7+6+4)/5, (4+8+5+4+9)/5) = (6, 6)
3.küme için , ((2+1)/2, (5+2)/2) = (1.5, 3.5)

Yeni kümeler:
1:{A1}
2:{A3,A4,A5,A6,A8}
3:{A2,A7}

Olarak elde edilmiştir.

Kümeleme örneği: K-means



3. Yarı Gözetimli Öğrenme (Semi-supervised Learning)

- Hem etiketli hem de etiketsiz verilerin bir kombinasyonunu kullanarak etiketlenmemiş verilerden de öğrenme yeteneğine sahiptirler, bu yüzden kısmen gözetimli olarak kabul edilir.
- Yarı gözetimli öğrenme, genellikle az sayıda etiketli veri noktası ve çok sayıda etiketlenmemiş veri noktası ile çalışır.

Kullanım Alanları:

- Büyük veri setlerinde etiketleme maliyetinin yüksek olması durumunda kullanışlıdır.
- Görüntü ve video analizi, web sayfası sınıflandırma, metin belgelerinin analizi.
- Bir fotoğraf uygulaması, az sayıda etiketli resim ve çok sayıda etiketlenmemiş resim kullanarak kullanıcıların fotoğraflarını otomatik olarak etiketlemeyi öğrenebilir.

3. Yarı Gözetimli Öğrenme (Semi-supervised Learning)

➢**Bazı yarı gözetimli öğrenme algoritmaları ve yöntemleri:**

- **Self-training ya da Self-labeling:**
  - Bu basit bir yaklaşımdır ve "öğretmen-student" modeli olarak da bilinir. İlk adımda, etiketli veriler kullanılarak bir model eğitilir. Daha sonra bu model, etiketlenmemiş verilere tahminler yapar ve bu tahminlerden emin olduğu örnekleri kendi eğitim setine dahil eder.
- **Co-training:**
  - Co-training, veri setinin farklı "görünümlerine" dayanır ve her bir görünüm için ayrı bir sınıflandırıcı eğitir. Bu sınıflandırıcılar daha sonra birbirlerinin eğitim veri setlerini zenginleştirmek için etiketlenmemiş verileri etiketleyebilir.
- **Çoklu örnekleme öğrenmesi (Multi-Instance Learning):**
  - Bu, örneklerin torbalar (bags) içinde gruplandığı bir durumda kullanılır. Eğer bir torba pozitif ise, içinde en az bir pozitif örnek olduğu bilinir ancak hangi örneğin pozitif olduğu bilinmez. Algoritma, torba etiketlerinden bireysel örnek etiketlerini çıkarabilmeyi öğrenir.

3. Yarı Gözetimli Öğrenme (Semi-supervised Learning)

➢ **Bazı yarı gözetimli öğrenme algoritmaları ve yöntemleri:**

- **Graflara dayalı yöntemler:**
  - Etiketlenmemiş ve etiketli veriler arasındaki ilişkileri modellemek için graf teorisi kullanılır. Örnekler bir grafın düğümleri olarak temsil edilir ve düğümler arasındaki kenarlar, örnekler arasındaki benzerlik veya ilişkileri ifade eder.
- **Yarı gözetimli Destek Vektör Makineleri (SVM):**
  - Geleneksel SVM algoritmasının genişletilmiş bir versiyonu olan bu algoritma, etiketli ve etiketlenmemiş verileri kullanarak karar sınırını optimize eder.
- **Entropy minimization:**
  - Bu yöntem, modelin etiketlenmemiş veriler üzerindeki belirsizliğini azaltacak şekilde ayarlamalar yapar, böylece model etiketlenmemiş veriler hakkında daha emin tahminlerde bulunabilir.

➢ Bu algoritmaların her biri, kendi başına etkin bir öğrenme stratejisi oluşturabilecek şekilde tasarlanmıştır ve gerçek dünya uygulamalarında, etiketleme maliyetlerini düşürmek ve mevcut etiketli verilerden en iyi şekilde yararlanmak için sıklıkla kullanılırlar.

4. Pekiştirmeli Öğrenme (Reinforcement Learning)

- Bir ajanın, ödülleri maksimize etmek için çevresiyle etkileşimde bulunarak hangi eylemlerin en iyi olduğunu öğrendiği bir ML türüdür.
- Bir dizi gözlem, eylem ve ödül/ceza üzerine kuruludur.
- Karar verme stratejisi olan bir ajan geliştirilir. Ajan, çevre üzerinde eylemler gerçekleştirir ve geri bildirim alır. Ajan, aldığı ödülleri maksimize edecek şekilde politikasını geliştirir.

4. Pekiştirmeli Öğrenme (Reinforcement Learning)

➢ **Kullanım Alanları:**

- Oyun oynamak (şah, Go, video oyunları)
- Otomatik araç kontrol sistemleri
- Robotikte yol bulma
- Online öneri sistemlerinde kullanıcı etkileşimi optimizasyonu

➢ **Örnek Algoritmalar:**

- Q-öğrenme (Q-learning)
- Derin Q Ağları (DQN)
- Politika Gradyanları
- Temporal Difference (TD) Learning
- Monte Carlo Yöntemleri

ML Modelinin Değerlendirilmesi

- Makine öğrenmesinde modelin performansını ölçmek için çeşitli metrikler ve yöntemler kullanılır.
- Performans ölçümü, modelin ne kadar iyi tahmin yapabildiğini ve gerçek dünya verileri üzerinde nasıl performans göstereceğini anlamamızı sağlar.

Hata Matrisi (Confusion Matrix)

- Sınıflandırma modellerinin performansını değerlendirmek için hedef niteliğe ait tahminler ve gerçek değerler hata matrisi kullanılarak karşılaştırılabilir.
- Modelin farklı sınıflarda nasıl performans gösterdiğini ayrıntılı olarak görmek için kullanılır.
- Dört bölümden (TP, TN, FP, FN) oluşmaktadır. Bu bölümlerdeki sayılar kullanılarak farklı değerlendirmeler yapılmaktadır.

1. Hastaya hasta demek (True Positive – TP) DOĞRU
2. Hasta olmayana hasta değil demek (True Negative – TN) DOĞRU
3. Hasta olmayana hasta demek (False Positive – FP) YANLIŞ
4. Hasta olana hasta değil demek (False Negative – FN) YANLIŞ

		TAHMİN		
		YOK	VAR	TOPLAM
GERÇEK	YOK	TN 100	FP 20	120
	VAR	FN 10	TP 200	210
	TOPLAM	110	220	

Kaynak: https://www.veribilimiokulu.com/hata-matrisini-confusion-matrix-yorumlama/

Doğruluk (Accuracy)

- Doğru tahminlerin toplam tahminlere oranıdır.
- Genel bir ölçüm olarak faydalıdır, ancak sınıf dengesizliği durumunda yanıltıcı olabilir.

Doğruluk = TN + TP / TOPLAM = 100+200 / 330 = 0,91

		TAHMİN		
		YOK	VAR	TOPLAM
GERÇEK	YOK	TN 100	FP 20	120
	VAR	FN 10	TP 200	210
	TOPLAM	110	220	

Hata Oranı (Error Rate / Misclassification Rate)

- Yanlışların toplama oranıdır.
- Aynı zamanda 1'den doğruluk oranını çıkararak da elde edilir.

```
Hata Oranı = FN + FP / TOPLAM = 10 + 20 / 330 = 0,09
Hata Oranı = 1 - Doğruluk = 1 - 0,91 = 0,09
```

		TAHMİN		
		YOK	VAR	TOPLAM
GERÇEK	YOK	TN 100	FP 20	120
	VAR	FN 10	TP 200	210
	TOPLAM	110	220	

Hata matrisinden elde edilen farklı performans ölçüm yöntemleri:

- **Kesinlik (Precision):** Hasta dediklerimizin gerçekten kaçı hasta?
- Yanlış pozitiflerin maliyetli olduğu durumlarda önemlidir (örneğin, spam tespiti).

$$precision = \frac{true\ positives}{true\ positives + false\ positives}$$

- **Recall (Duyarlılık):** Hasta olanları doğru tespit etme oranı?
- Tüm gerçek pozitif örnekleri yakalamak önemli olduğunda (örneğin, hastalık tespiti) kullanılabilir.

$$recall = \frac{true\ positives}{true\ positives\ + false\ negatives}$$

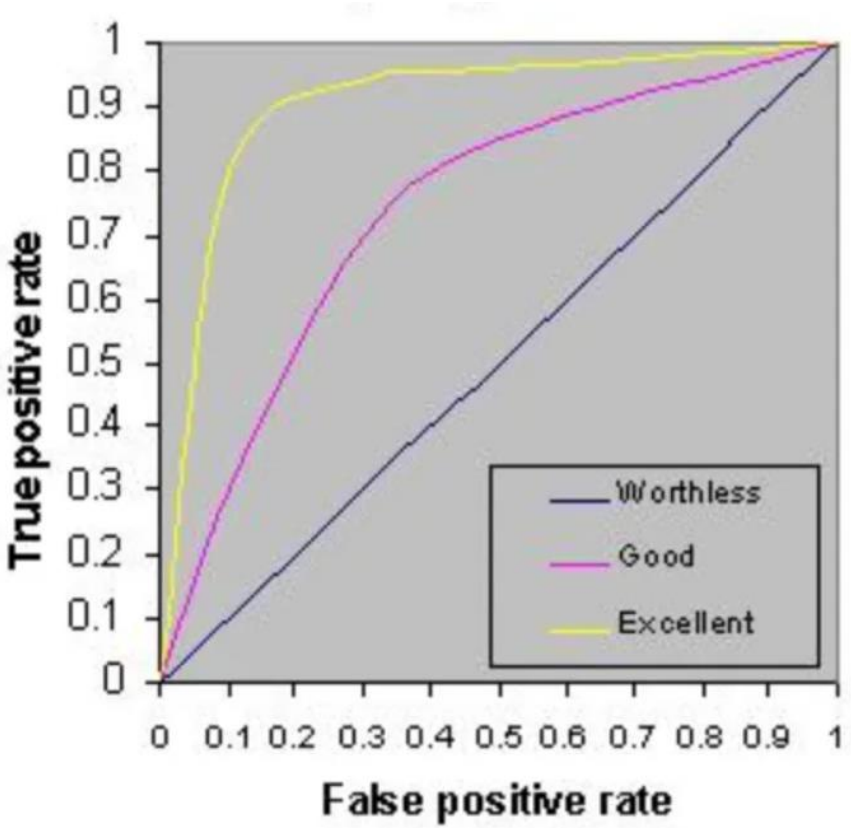
Hata matrisinden elde edilen farklı performans ölçüm yöntemleri:

- **F1 Skoru:** Precision ve recall'un harmonik ortalaması.
- Dengeli bir ölçüm sağlar ve her iki sınıfın da önemli olduğu durumlarda kullanılır.

$$F_1 = 2 * \frac{precision * recall}{precision + recall}$$

ROC Eğrisi ve AUC

- Gerçek pozitif oranı ile yanlış pozitif oranı arasındaki ilişkiyi grafik üzerinde gösterir.
- ROC bir olasılık eğrisidir ve altında kalan alan olan AUC ayrılabilirliğin derecesini veya ölçüsünü temsil eder.
- Eğrinin altında kalan arttıkça sınıflar arasında ayırt etme performansı artmaktadır. AUC değeri 0.5 olan modelin performansının kötü olduğunu ve rastgele tahminleme yaptığı söylenebilir.



Kaynak: https://medium.com/@gulcanogundur/roc-ve-auc-1fefcfc71a14

Ortalama Mutlak Hata (MAE) ve Kök Ortalama Kare Hata (RMSE)

- MAE, tahminler ile gerçek değerler arasındaki farkların mutlak değerlerinin ortalamasıdır. RMSE ise bu farkların karesinin ortalamasının kareköküdür.
- Regresyon problemlerinde yaygın olarak kullanılır.
- y_i gerçek değerler, $\hat{y}_i$ ise modelin tahmin ettiği değerlerdir. MAE, hataların büyüklüğünü ölçerken, RMSE hataların karelerini alarak büyük hatalara daha fazla ağırlık verir.

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

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

R-kare (R^2)

- Verilerin, model tarafından ne kadar iyi açıklandığının bir ölçüsüdür.
- Regresyon modellerinin açıklanabilirliğini değerlendirmek için kullanılır.
- 1'e yakın değerler, modelin verileri iyi açıkladığını belirtir.
- $\bar{y}$ tüm gözlemlerin ortalama değeridir.

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

Kümeleme algoritmalarının performansı

- **Siluet Katsayısı:** Her bir veri noktası için, kendi kümesindeki diğer noktalara olan ortalama mesafeye (yoğunluk) ve en yakın komşu kümeye olan ortalama mesafeye (ayrışma) dayanır. Değerler -1 ile 1 arasında değişir; yüksek değerler iyi kümelenmiş noktaları gösterir.
- **Davies-Bouldin İndeksi:** Küme içi benzerlik ile kümeler arası ayrımın oranını kullanarak kümelerin ne kadar iyi ayrıldığını değerlendirir. Düşük değerler daha iyi kümelenme performansını gösterir.
- **Dunn İndeksi:** Küme içi mesafelerin minimumu ile kümeler arası mesafelerin maksimumu arasındaki oranı hesaplar. Yüksek değerler daha iyi kümelenme performansını gösterir.