Makine Öğrenmesi Nedir?

Makine öğrenmesi, bilgisayar sistemlerinin veri kullanarak örüntüleri (patterns) tanımasını, tahmin yapmasını ve karar vermesini sağlayan bir yapay zekâ (AI) alt dalıdır. Bu alanda geliştirilen algoritmalar, veriler arasındaki ilişkileri öğrenir ve bu bilgiyi gelecekte benzer durumlar için otomatik karar verme veya tahmin yapmada kullanır. Örneğin, Netflix'in film öneri sistemi ya da e-postalardaki spam filtreleri makine öğrenmesi sayesinde çalışır.

Makineler Nasıl Öğrenir: Yapay zekâ algoritmaları, büyük miktarda veriye dayalı olarak desenleri tanımlar, kararlar alır ve sonuçları tahmin ederler. İnsanların öğrenme sürecine benzer şekilde, yapay zekâ modelleri de hatalarından öğrenir ve deneyimleriyle daha iyi hale gelir.

Makinelerde Öğrenme Kavramı: Bilgisayar sistemlerinin doğrudan programlanmadan, verilerden ve deneyimlerden öğrenebilmesini ifade eder. Makine, bir algoritma aracılığıyla geçmiş verileri analiz eder, bu analizden bir "öğrenme modeli" çıkarır ve gelecekteki yeni veriler için bu modeli kullanarak karar verir. Örneğin, bir e-ticaret sitesinin müşterilere ürün önermesi veya bir bankanın kredi başvurusunu değerlendirmesi, makinelerin öğrenme kavramına dayalıdır.

Makine Öğrenmesi Terminolojisi:

- ***Veri Kümesi (Dataset):*** Makine öğrenmesinde kullanılan ham veri topluluğudur. Veri kümesi genellikle girdiler (features) ve çıktılar (labels) içerir. Örneğin, ev fiyatı tahmini için veri kümesi; oda sayısı, metrekare, konum gibi özellikler ve evin fiyatı etiketinden oluşabilir.
- ***Öznitelik (Features):*** Modelin öğrenebilmesi için kullanılan girdi değerleridir. Her bir özellik, veri kümesinde bir sütunu temsil eder. Örneğin, bir öğrencinin sınav performansını tahmin etmek için "ders çalışma süresi" ve "uyku süresi" birer özniteliktir.
- ***Etiketler (Labels):*** Modelin tahmin etmeye çalıştığı değerlerdir. Yani özniteliklere karşılık gelen çıktı veya hedef değişkenidir. Örneğin, "ders çalışma süresi" ve "uyku süresi" öznitelikse, "sınav notu" etikettir.
- ***Model:*** Makine öğrenmesi algoritmasının, veriden öğrendiklerini temsil eden matematiksel yapıdır. Eğitim sürecinde verilerden örüntüleri öğrenir ve daha sonra yeni veriler üzerinde tahmin yapar.
- ***Eğitim (Training):*** Modelin, veri kümesi üzerinden örüntüleri öğrenme sürecidir. Eğitim sırasında model, tahminlerini yapar ve hata oranına göre parametrelerini günceller.
- ***Doğrulama (Validation):*** Modelin performansını ölçmek ve aşırı öğrenmeyi (overfitting) önlemek için kullanılan aşamadır. Eğitim sırasında modelin gerçek dünyaya ne kadar genellenebilir olduğunu görmek için doğrulama verileri kullanılır.
- ***Test***: Model eğitildikten sonra, daha önce hiç görmediği verilerle son performansının ölçüldüğü aşamadır. Bu aşama, modelin gerçek dünyada ne kadar başarılı olacağını gösterir.
- ***Sınıflandırma (Classification):*** Bir veriyi kategorilerden birine atayan makine öğrenmesi problemidir. Örnek: E-postaları "Spam" veya "Spam değil" olarak sınıflandırmak.
- ***Regresyon (Regression):*** Sürekli (sayısal) değerlerin tahmin edilmesi problemidir.
  Örnek: Bir evin fiyatını tahmin etmek.
- ***Öğrenme Algoritmaları (Learning Algorithms):*** Verilerden örüntüleri çıkarmak için kullanılan matematiksel yöntemlerdir. Örnek algoritmalar:
  - Doğrusal Regresyon (Linear Regression)
  - Karar Ağaçları (Decision Trees)
  - Destek Vektör Makineleri (SVM)
  - K-En Yakın Komşu (KNN)

- ***Öznitelik Mühendisliği (Feature Engineering):*** Veri kümesindeki özellikleri seçme, dönüştürme veya yeni özellikler oluşturma sürecidir. Amaç, modelin daha iyi öğrenmesini sağlamaktır. Örneğin, "doğum yılı" verisinden "yaş" özelliğini oluşturmak bir öznitelik mühendisliğidir.
- ***Hiperparametreler (Hyperparameters):*** Modelin eğitim sürecini kontrol eden ayar değerleridir. Modelin kendisi tarafından öğrenilmez; kullanıcı tarafından belirlenir.
  Örnek: Öğrenme oranı (learning rate), katman sayısı, karar ağacının derinliği gibi.
- ***Aşırı Öğrenme (Overfitting):*** Modelin eğitim verisini ezberlemesi, yeni verilerde kötü sonuç vermesidir.
- ***Eksik Öğrenme (Underfitting):*** Modelin veri içindeki örüntüleri öğrenememesi hem eğitim hem testte kötü performans göstermesidir.
- ***Cross-Validation(Çarpraz Doğrulama):*** Modelin doğruluğunu daha güvenilir ölçmek için veri kümesinin birden fazla alt parçaya bölünerek test edilmesi yöntemidir.
  En yaygın olanı k-fold cross-validation yöntemidir.
- ***Derin Öğrenme (Deep Learning):*** Makine öğrenmesinin bir alt alanıdır. Yapay sinir ağları kullanarak büyük miktarda veriden karmaşık örüntüleri öğrenir. Görsel tanıma, konuşma tanıma ve doğal dil işleme (NLP) gibi alanlarda çok başarılı sonuçlar verir.

Makine Öğrenmesi Yöntemleri:

1. Supervised Learning (Gözetimli Öğrenme): Model etiketlenmiş veri kullanarak eğitilir. Yani her girdiye karşılık bir doğru çıktı (label) bulunur. Model, bu eşleşmeleri öğrenir ve yeni veriler için doğru tahmini yapmaya çalışır. Amaç: Girdi → Çıktı ilişkisini öğrenmek. Örnekler:

- Bir e-postanın "spam" olup olmadığını tahmin etmek.
- Ev fiyatlarını tahmin etmek (metrekare, konum, oda sayısına göre).
- Hastalık teşhisi yapmak (belirtilere göre hastalık sınıflandırmak).

<u>*Yaygın Algoritmalar*</u>: * Linear Regression) * Decision Trees * Destek Vektör Makineleri (SVM)

2. Unsupervised Learning (Gözetimsiz Öğrenme): Veriler etiketlenmemiştir. Model, veriler arasındaki gizli yapıları, grupları veya ilişkileri kendi başına keşfetmeye çalışır. Amaç: Verilerdeki örüntüleri veya benzerlikleri bulmak. Örnekler:

- Müşterileri alışveriş alışkanlıklarına göre gruplamak (müşteri segmentasyonu).
- Benzer haberleri veya belgeleri kümelere ayırmak.
- Görsel verilerde benzer objeleri tespit etmek.

<u>*Yaygın Algoritmalar:*</u> *K-Means Clustering * Hiyerarşik Clustering * PCA (Temel Bileşen Analizi)

3. Reinforcement Learning (Pekiştirmeli Öğrenme): Bu yöntemde model, bir ortamla etkileşime girerek deneme-yanılma yoluyla öğrenir. Her yaptığı eylemin sonucunda bir ödül (reward) veya ceza (penalty) alır. Amaç, toplam ödülü maksimize edecek en iyi stratejiyi (policy) öğrenmektir. Amaç: Deneyimlerden öğrenerek en iyi kararı vermek. Örnekler:

- Oyun oynayan yapay zekâlar (örneğin, AlphaGo'nun Go oyununu öğrenmesi).
- Otonom araçların sürüş kararlarını öğrenmesi.
- Robotların bir hedefe ulaşmayı öğrenmesi.

<u>*Yaygın Algoritmalar*</u>: * Q-Learning * Deep Q-Networks (DQN) *Policy Gradient Yöntemleri

Makine Öğrenmesi Süreci:

***Verilerin Toplanması, Düzenlenmesi ve Bölünmesi*:** Makine öğrenmesi sürecinin ilk adımı veridir. Modelin başarısı, büyük ölçüde kullanılan verinin kalitesine bağlıdır. Bu aşamada:

1. *Veri toplanır* (sensörler, veri tabanları, API'ler, web siteleri vb.).
2. *Veri temizlenir* (eksik, hatalı veya gereksiz bilgiler çıkarılır).
3. *Veri dönüştürülür* (sayısal forma getirilir, normalleştirilir).
4. *Veri bölünür*: Eğitim (%70), doğrulama (%15) ve test (%15) setlerine ayrılır.

Örnek: Bir banka, kredi başvurularını analiz etmek için müşteri verilerini toplar; eksik gelir bilgilerini temizler, ardından veriyi eğitim/doğrulama/test olarak böler.

Makine Öğrenmesi Modelinin Seçilmesi: Bu adımda, problemin türüne uygun algoritma seçilir:

- Eğer sonuç sınıf (örneğin: "hasta / sağlıklı") ise → Sınıflandırma algoritması
- Eğer sonuç sayısal değer ise → Regresyon algoritması
- Eğer veri etiketsiz ise → Kümeleme veya boyut indirgeme algoritması

Örnek: Evin fiyatını tahmin etmek için Doğrusal Regresyon (Linear Regression); e-postayı "spam veya değil" olarak sınıflandırmak için Naive Bayes modeli seçilebilir.

Modelin Train Edilmesi (Eğitilmesi): Bu aşamada model, eğitim verisi ile örüntüleri öğrenir. Algoritma, her adımda tahmin yapar, hatayı hesaplar ve parametrelerini günceller. Amaç, hatayı (loss) mümkün olduğunca azaltmaktır.

Örnek: Bir sinir ağı, binlerce e-posta üzerinde eğitilerek "spam" ve "değil" arasındaki farkları öğrenir.

Sonuçların Değerlendirilmesi: Model eğitildikten sonra, doğrulama ve test verileri ile performansı ölçülür. Değerlendirmede kullanılan bazı metrikler:

- Accuracy (Doğruluk)
- Precision (Kesinlik)
- Recall (Duyarlılık)
- F1-Score
- RMSE (Kök Ortalama Kare Hatası)

Hiperparametre Ayarlaması: Modelin performansını artırmak için, kullanıcının dışarıdan belirlediği ayar değerleri optimize edilir. Bu değerlere hiperparametreler denir (örneğin: öğrenme oranı, katman sayısı, karar ağacı derinliği). Bu işlem genellikle Grid Search veya Random Search yöntemleriyle yapılır. Amaç: Modelin doğruluğunu ve genelleme yeteneğini artırmak.

Modelin Hizmete Sunulması (Deployment): Eğitilmiş ve test edilmiş model, artık gerçek dünyada kullanılabilir hale getirilir. Bu adımda model bir uygulamaya, web servisine veya sisteme entegre edilir. Örnek:

- Bir bankanın kredi onaylama sisteminde modelin kullanılması.
- Netflix'in öneri modelinin kullanıcılara film önerileri sunması.
- Bir web sitesinde anlık fiyat tahmini yapılması.

Makine Öğrenmesi Uygulamaları:

- ***Tıp ve Sağlık*:** Makine öğrenmesi, hastalık teşhisi, ilaç geliştirme ve tıbbi görüntü analizi gibi alanlarda doktorlara destek sağlar. Örneğin, kanser hücrelerini erken tespit etmekte kullanılır.
- ***Finans:*** Banka ve sigorta sistemlerinde kredi riski analizi, dolandırıcılık tespiti ve hisse senedi fiyat tahmini gibi işlemlerde kullanılır.
- ***Perakende ve Pazarlama:*** Müşteri davranışlarını analiz ederek kişisel öneriler sunar, satış tahminleri ve stok yönetimi yapar.
- ***Ulaşım ve Lojistik:*** Rota optimizasyonu, trafik tahmini ve otonom araç kontrolü gibi uygulamalarda kullanılır. Bu sayede zaman ve yakıt tasarrufu sağlanır.
- ***Güvenlik ve Siber Güvenlik:*** Sistemler, ağ trafiğindeki anormallikleri tespit ederek siber saldırıları önceden fark eder. Ayrıca kimlik doğrulama ve veri koruma alanlarında da kullanılır.
- ***Sınır Güvenliği:*** Yüz tanıma, davranış analizi ve belge doğrulama sistemlerinde kullanılarak sınır geçişlerinde güvenliği artırır.
- ***Robotik:*** Robotların çevresini algılayıp kendi kararlarını verebilmesi için makine öğrenmesi kullanılır. Örneğin, üretim robotlarının hatasız montaj yapmasını sağlar.
- ***Enerji Planlaması ve Verimlilik:*** Enerji tüketimini tahmin eder, üretimi optimize eder ve yenilenebilir enerji kaynaklarının verimli kullanılmasını sağlar.

Makine Öğrenmesi Araçları:

TensorFlow: Google tarafından geliştirilmiş açık kaynaklı bir derin öğrenme kütüphanesidir. Sinir ağları, görüntü işleme ve doğal dil işleme (NLP) gibi alanlarda yaygın olarak kullanılır.

PyTorch: Facebook tarafından geliştirilen esnek ve kullanıcı dostu bir derin öğrenme kütüphanesidir. Araştırma ve deneysel projelerde hızlı prototipleme için tercih edilir.

Keras: TensorFlow üzerine kurulmuş, derin öğrenme modellerini kolayca oluşturmayı sağlayan yüksek seviyeli bir kütüphanedir. Başlangıç seviyesi kullanıcılar için oldukça uygundur.

NumPy: Sayısal hesaplamalar için kullanılan güçlü bir Python kütüphanesidir. Makine öğrenmesinde matris işlemleri ve veri manipülasyonu için temel yapı taşını oluşturur.

Scikit-Learn: Makine öğrenmesinde sınıflandırma, regresyon, kümeleme ve model değerlendirme gibi işlemleri kolaylaştıran popüler bir kütüphanedir.

Pandas: Veri analizi ve veri temizleme işlemleri için kullanılan Python kütüphanesidir. Veri çerçeveleri (DataFrame) sayesinde veriyi kolayca düzenlemeyi sağlar.

Matplotlib: Veri görselleştirme için kullanılan bir kütüphanedir. Grafikler, histogramlar ve çizgi grafikleri oluşturarak modellerin sonuçlarını analiz etmeye yardımcı olur.

Theano: Matematiksel hesaplamalar ve derin öğrenme modelleri için kullanılan bir kütüphanedir. GPU desteği sayesinde işlemleri hızlandırır; ancak günümüzde TensorFlow ve PyTorch tarafından büyük ölçüde yerini almıştır.

Veri Tipleri:

- ***Kategorik:*** Sayısal olmayan, belirli grupları veya kategorileri temsil eden verilerdir. Model bu verileri etiket olarak kullanır, ancak aralarında matematiksel işlem yapılmaz.
    - ***Nominal (Adlandırılmış Veriler):*** Kategoriler arasında doğal bir sıralama yoktur sadece isim veya etiket belirtir. Örnek: Renkler (Kırmızı, Yeşil, Mavi), Cinsiyet (Kadın, Erkek), Şehir (Ankara, İzmir, Adana).
    - ***Ordinal (Sıralı Veriler):*** Kategoriler arasında doğal bir sıralama vardır, ancak aralarındaki fark ölçülemez. Örnek: Eğitim Seviyesi (İlkokul <Ortaokul <Lise <Üniversite), Memnuniyet Düzeyi (Az – Orta – Çok).
- ***Sayısal:*** Sayısal veriler, ölçülebilen ve üzerinde matematiksel işlem yapılabilen verilerdir.
    - ***Interval (Eşit Aralıklı Veriler):*** Değerler arasında eşit aralıklar vardır, fakat mutlak sıfır noktası yoktur (0, "hiçlik" anlamına gelmez). Örnek: Sıcaklık (Celsius), Zaman (Saat cinsinden).
    - ***Ratio (Oranlı Veriler):*** Değerler arasında hem eşit aralıklar vardır hem de mutlak sıfır noktası bulunur (0 = hiçlik). Bu yüzden oranlar karşılaştırılabilir (örneğin, 10 metre = 2 × 5 metre). Örnek: Uzunluk, Ağırlık, Yaş, Gelir, Hız.

Veri Ön İşleme (Data Preprocessing)*:* Veri ön işleme, ham verileri makine öğrenmesi için uygun hâle getirme sürecidir. Gerçek hayatta veriler genellikle eksik, hatalı veya farklı formatlardadır. Bu nedenle modelin doğru öğrenebilmesi için veriler önce temizlenip düzenlenir. Veri ön işleme süreçleri:

- ***Veri temizleme (Data Cleaning):*** Veri setindeki eksik, hatalı veya tutarsız verilerin tespit edilerek düzeltilmesi veya kaldırılması işlemidir. Bu adım modelin güvenilirliğini artırır. Örnek: Bir tablo içinde yaş sütununda boş değerler varsa ortalama yaşla doldurmak ya da o satırı silmek. Veri temizleme işlemi genellikle şu nedenlerle gereklidir:
  * Doğruluk *Tutarlılık *Eksik Veri Problemi *Aykırı Değerler *Standartlaştırma
    <u>Veri Temizleme Yöntemleri:</u>
    - ***Kayıp Veri Problemi:*** Bir veri setinde eksik veya bozuk verilerin tespit edilmesi.
    - ***Tutarsız Veri Düzeltme*:** Veri setindeki tutarsızlıkları tespit etme ve düzeltme işlemi.
    - ***Yinelenen Verilerin Silinmesi:*** Veri setindeki tekrarlayan verilerin tespit edilmesi ve bunların temizlenmesi işlemidir.
    - ***Gereksiz Sütunların Silinmesi:*** Veri setinde analiz veya modelleme işlemleri için gerekli olmayan sütunların çıkarılması veya kaldırılması işlemidir.
- ***Veri standardizasyonu (Data Standardization*):** Farklı ölçekteki sayısal verilerin ortak bir ölçeğe getirilmesidir. Amaç, modelin bazı özelliklere gereğinden fazla önem vermesini önlemektir. Örnek: Bir veri setinde "yaş" 1–100 arası, "gelir" 1000–100000 arası olabilir. Bu değerler, aynı ölçekte (örneğin 0–1) normalize edilir.
- ***Öznitelik seçimi (Feature Selection):*** Modelin performansını artırmak için önemsiz veya gereksiz özelliklerin çıkarılmasıdır. Sadece tahmini etkileyen önemli öznitelikler tutulur. Örnek: Ev fiyatı tahmininde "evin rengi" gereksiz bir öznitelik olabilir, bu nedenle kaldırılır.
- ***Veri dönüşümü (Data Transformation):*** Verilerin model tarafından daha kolay işlenebilmesi için format veya tür değiştirme işlemidir. Kategorik veriler genellikle sayısal forma dönüştürülür. Örnek: "Cinsiyet" verisini → Erkek = 0, Kadın = 1 şeklinde dönüştürmek.
- ***Aykırı değerlerin işlenmesi (Outlier Handling):*** Veri kümesinde diğerlerinden çok farklı (uç) değerlerin tespit edilip uygun şekilde işlenmesidir. Bu değerler modelin öğrenmesini bozabileceği için çıkarılabilir veya sınırlandırılabilir. Örnek: Bir maaş verisinde 1.000.000 TL gibi aşırı yüksek bir değer varsa bu aykırı değerdir; ortalamayı bozduğu için temizlenebilir.

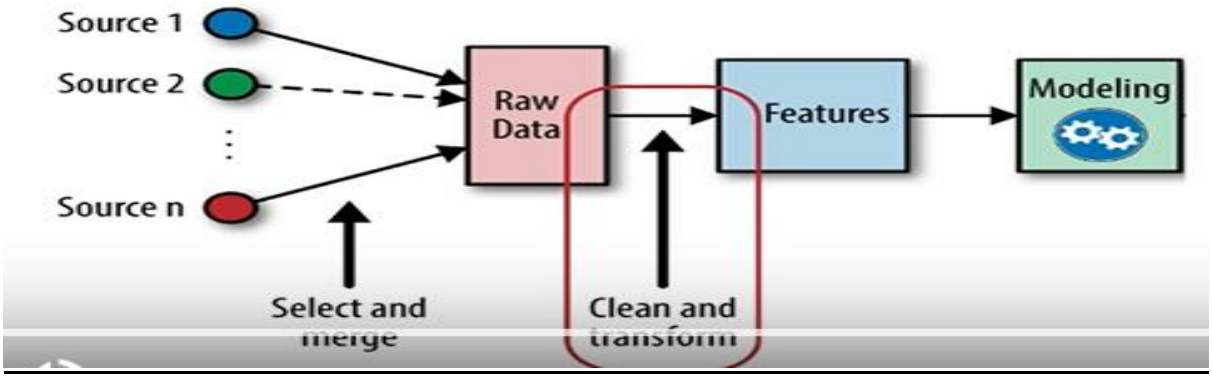
Veri Kodlaması: Veri kodlaması, kategorik verileri sayısal değerlere dönüştürme işlemidir. Bu dönüşüm, makine öğrenimi algoritmaları gibi birçok analiz yöntemi için gereklidir, çünkü bu algoritmalar genellikle sayısal verilerle çalışır. Veri kodlamasının 2 ana türü vardır:

- ***Nominal Kodlama:*** Kategorik veriler arasında herhangi bir sıralama veya önem farkı olmadığında kullanılır. Örneğin, renkler (kırmızı, mavi, yeşil) nominal veridir. Bu tür veriler genellikle One-Hot Encoding gibi yöntemlerle kodlanır.
- ***Ordinal Kodlama:*** Kategorik veriler arasında doğal bir sıralama veya derecelendirme olduğunda kullanılır. Örneğin, eğitim düzeyi (lise <lisans <yüksek lisans <doktora) ordinal veridir. Bu veriler genellikle 0, 1, 2, … gibi artan sayısal değerlerle kodlanır.

Öznitelik (Feature) Nedir?

Öznitelik, bir veri örneğini tanımlayan veya açıklayan ölçülebilir veya gözlemlenebilir bir özelliktir. Veri setinde her bir gözlemin bir parçasını oluşturan bu özellikler genellikle veri setindeki sütunlara karşılık gelir. Örneğin, bir öğrencinin veri setinde yaşı, not ortalaması veya cinsiyeti birer özniteliktir.

Öznitelik Mühendisliği (Feature Engineering): Öznitelik mühendisliği, veri bilimi ve makine öğreniminde, mevcut öznitelikleri optimize etmek veya yeni öznitelikler oluşturmak için yapılan işlemleri ifade eder. Bu süreç, modelin performansını artırmak ve veriden daha anlamlı bilgiler çıkarmak için kritik öneme sahiptir.



Bu görsel, makine öğrenmesi sürecinde verilerin hazırlanma aşamalarını gösteriyor:

- Source 1, Source 2, Source n: Verilerin toplandığı farklı kaynakları temsil eder. Bunlar veritabanları, sensörler, dosyalar, API'ler veya web siteleri olabilir.
- *Select and merge (Seçme ve birleştirme):* Farklı kaynaklardan gelen veriler seçilip birleştirilir. Bu aşamada hangi verilerin kullanılacağına karar verilir.
- *Raw Data (Ham Veri):* Henüz işlenmemiş, doğrudan kaynaklardan gelen veridir. Bu veriler eksik, hatalı veya gereksiz bilgiler içerebilir.
- *Clean and transform (Temizleme ve dönüştürme):* Ham veriler bu aşamada temizlenir (eksik veya hatalı veriler düzeltilir) ve modele uygun hale getirilir (örneğin sayısal formata dönüştürülür).
- *Features (Öznitelikler):* Temizlenmiş ve anlamlı hale getirilmiş verilerdir. Bu veriler, modelin öğrenmesi için kullanılır.
- *Modeling (Modelleme):* Son aşamada, bu öznitelikler kullanılarak makine öğrenmesi modeli eğitilir ve tahmin veya sınıflandırma işlemleri yapılır.

Özetle: Farklı kaynaklardan alınan ham veriler temizlenip dönüştürülür, anlamlı özellikler çıkarılır ve bu özellikler modelleme aşamasında kullanılır.

Öznitelik Mühendisliğinin Önemi: Öznitelik mühendisliği, modelin başarısı ve verinin verimli kullanımı açısından kritik bir süreçtir. Başlıca faydaları şunlardır:

- Model performansını artırma
- Boyut azaltma
- Hesaplama maliyetini azaltma

Öznitelik Seçimi (Feature Selection): Öznitelik seçimi, veri setinde bulunan öznitelikler arasından en önemli veya en bilgilendirici olanları seçme veya belirleme sürecidir. Bu süreçte, veri setinde gereksiz, tekrarlayan veya düşük etkili özniteliklerin tanımlanması ve çıkarılması hedeflenir. Amacı, modelin performansını artırmak, karmaşıklığı azaltmak ve aşırı uyumu engellemektir.

Öznitelik Seçimi Yöntemleri:

- ***İnformasyon tabanlı yöntemler:*** Her bir özniteliğin hedef değişkenle olan bilgi ilişkisini ölçer. Genellikle Bilgi Kazancı (Information Gain), Karşılıklı Bilgi (Mutual Information) gibi metrikler kullanılır. Amaç, en fazla bilgi sağlayan öznitelikleri seçmektir.
  Örnek: Karar ağaçlarında bilgi kazancı ölçütüyle öznitelik seçimi yapılır.
- ***Wrapper yöntemler:*** Modeli bir kara kutu (black box) gibi kullanır. Farklı öznitelik kombinasyonlarıyla modeli eğitir ve performansa göre en iyi seti seçer. Daha doğru sonuç verir ama hesaplama açısından pahalıdır.
  Örnek: Forward Selection, Backward Elimination, Recursive Feature Elimination (RFE)
- ***Gömülü yöntemler:*** Model eğitimi sırasında özellik seçimini otomatik olarak yapar. Bazı modeller (örneğin LASSO, Decision Tree) özniteliklerin önemini içsel olarak hesaplar. Hem performans hem hız açısından dengelidir.
  Örnek: LASSO regresyonunda katsayısı sıfır olan öznitelikler elenir.
- ***Filter yöntemler:*** Modelden bağımsız çalışır. İstatistiksel ölçülerle (örneğin korelasyon) özniteliklerin hedefle olan ilişkisi değerlendirilir. Hızlıdır ama modelin yapısını dikkate almaz.
  Örnek: Korelasyonu düşük olan öznitelikleri filtrelemek.

***Öznitelik Çıkarma*:** Öznitelik çıkarma (feature extraction), veri setindeki özniteliklerin temsilini değiştirme veya dönüştürme sürecidir. Bu süreçte, var olan özniteliklerin birleştirilmesi, dönüştürülmesi veya yeni özniteliklerin oluşturulması yoluyla verinin temsilinin iyileştirilmesi amaçlanır. Öznitelik çıkarma, veri boyutunu azaltarak, gereksiz bilgiyi çıkararak ve verinin daha anlamlı bir şekilde temsil edilmesini sağlayarak model performansını artırabilir.

Öznitelik Çıkarma Yöntemleri:

- ***Principle Component Analysis (PCA):*** Yüksek boyutlu veriyi, en fazla varyansı açıklayan daha az sayıda bileşene indirger. Bu bileşenler, orijinal özniteliklerin lineer kombinasyonlarıdır. Amaç, bilgi kaybını en aza indirerek boyutu küçültmektir.
  Örnek: 100 özellikli veriyi, verinin çoğu bilgisini koruyarak 10 bileşene indirgemek.
- ***Linear Discriminant Analysis (LDA):*** Sınıflar arasındaki farkı maksimum, sınıf içi farkı minimum yapan yeni öznitelikler oluşturur. PCA'dan farkı: sınıf etiketlerini dikkate alır (yani denetimli bir yöntemdir). Özellikle sınıflandırma problemlerinde boyut indirgeme için kullanılır. Örnek: İki farklı sınıfı en iyi ayıran doğrusal eksenleri bulmak.
- ***Autoencoders:*** Yapay sinir ağlarına dayalı bir boyut indirgeme yöntemidir. Girdi verisini sıkıştırarak kodlar (encoder) ve ardından yeniden oluşturur (decoder). Gizli katmandaki temsil, verinin en anlamlı özelliklerini taşır.
  Örnek: Görsel veriyi düşük boyutlu temsile indirip, gürültüsüz yeniden oluşturmak.

Veri Normalizasyonu ve Standardizasyonu: Veri normalizasyonu ve standardizasyonu, veri özniteliklerinin farklı ölçeklere veya dağılımlara sahip olması durumunda bunları belirli bir standart forma dönüştürmek için kullanılan yöntemlerdir. Bu işlemler, makine öğrenimi modellerinin daha doğru ve hızlı öğrenmesini sağlar.

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

Öncelikle, Matematik notlarını [0, 1] aralığına normalizasyon yapalım:

$$X_{norm} = \frac{X - 70}{90 - 70} = \frac{X - 70}{20}$$

Öğrenci	Matematik Notu	Normalizasyon
1	85	$\frac{85-70}{20} = 0.75$
2	70	$\frac{70-70}{20} = 0.0$
3	90	$\frac{90-70}{20} = 1.0$
4	75	$\frac{75-70}{20} = 0.25$
5	80	$\frac{80-70}{20} = 0.5$

$$\text{Mean} = \frac{\sum X}{N} = \frac{85 + 70 + 90 + 75 + 80}{5} = 80$$

$$\text{Std} = \sqrt{\frac{\sum (X - \text{Mean})^2}{N}}$$

$$= \sqrt{\frac{(85-80)^2 + (70-80)^2 + (90-80)^2 + (75-80)^2 + (80-80)^2}{5}} \approx 6.71$$

Şimdi, Matematik notlarını standart normal dağılıma dönüştürelim:

$$X_{stand} = \frac{X - \text{Mean}}{\text{Std}} = \frac{X - 80}{6.71}$$

Öğrenci	Matematik Notu	Standardizasyon
1	85	$\frac{85-80}{6.71} \approx 0.75$
2	70	$\frac{70-80}{6.71} \approx -1.49$
3	90	$\frac{90-80}{6.71} \approx 1.49$
4	75	$\frac{75-80}{6.71} \approx -0.75$
5	80	$\frac{80-80}{6.71} \approx 0.0$

Veri Bölme (Data Splitting): Veri bölme, bir veri setini eğitim, doğrulama ve test kümeleri olarak üçe veya daha fazla parçaya ayırma işlemidir. Bu işlem, makine öğrenimi modelinin eğitilmesi, doğrulanması ve değerlendirilmesi için yapılır. Genellikle, veri setinin büyük bir kısmı modelin eğitimi için kullanılırken, geri kalan kısmı modelin doğrulanması ve test edilmesi için kullanılır.

Veri bölmenin önemi:

- Model performansının doğru değerlendirilmesi
- Aşırı uyumun (overfitting) önlenmesi
- Parametre ayarı ve model seçiminin yapılması

Dengesiz Veri İşleme: Dengesiz veri, farklı sınıflara ait örneklerin sayısal olarak büyük farklılıklar gösterdiği durumlarda ortaya çıkar. Örneğin, bir sınıfın örnek sayısı diğerlerine göre çok daha fazlaysa, bu dengesiz veri problemi olarak kabul edilir. Bu tür durumlarda model, çoğunluk sınıfına aşırı odaklanabilir ve azınlık sınıfını doğru tanımakta zorlanabilir. Sonuç olarak modelin performansı düşer ve özellikle nadir olayları (örneğin sahtekârlık, hastalık tespiti) yakalama başarısı azalır.

<u>Dengesiz Veri İşleme Yöntemleri:</u>

- ***Oversampling (Aşırı Örnekleme):*** Azınlık sınıfındaki örnekleri artırarak veri setini dengeler.
- ***Undersampling (Yetersiz Örnekleme):*** Çoğunluk sınıfındaki örnekleri azaltarak denge sağlar.
- ***Synthetic Sampling (Sentetik Örnekleme):*** Yeni örnekler oluşturulur (ör. SMOTE yöntemi).

Aykırı Değer (Outlier): Aykırı değer, genel veri trendinden önemli ölçüde farklı olan ve genellikle diğer veri noktalarından uzakta bulunan bir veri noktasıdır. Makine öğreniminde aykırı değerler genellikle modelin yanlış eğitilmesine veya yanıltılmasına neden olabilir.

Aykırı değer tespiti önemi:

- Model performansını etkileyebilir.
- Modelin hassasiyetini artırır.
- Modelin yorumlanabilirliğini artırır.

<u>Aykırı değer tespit yöntemleri:</u>

- ***Standart sapma yaklaşımı:*** Veri değerlerinin ortalamadan ne kadar uzaklaştığını ölçer. Belirli bir standart sapma eşiğini aşan değerler aykırı olarak kabul edilir.
- ***Box Plot (Kutu Grafiği) yöntemi:*** İstatistiksel olarak çeyrek değerleri (Q1, Q3) ve IQR (Interquartile Range) kullanır. Aykırı değerler genellikle şu aralığın dışındakilerdir: [Q1- 1.5×IQR, Q3 + 1.5×IQR]
- ***Z-Skoru yaklaşımı:*** Her veri noktasının ortalamadan kaç standart sapma uzakta olduğunu hesaplar. Z-skoru genellikle ±3'ten büyük veya küçük olan değerler aykırı kabul edilir. Formül: $\mathbf{Z = \frac{(x - ortalama)}{standart\ sapma}}$
- ***LOF (Local Outlier Factor):*** Veri noktalarının komşularına göre yoğunluğunu karşılaştırır. Bir veri noktası, çevresindeki diğer noktalara göre çok daha seyrek bir bölgede bulunuyorsa, aykırı olarak değerlendirilir.

Makine Öğrenmesi için Temel Matematik

Lineer Cebir: Makine öğrenmesinin temel taşlarından biridir. Özellikle matrisler vektörler ve matris işlemleri makine öğrenmesi algoritmalarının pek çoğunda kullanılır. Örneğin, model parametrelerinin matris formunda ifade edilmesi, özellik vektörlerinin işlenmesi ve boyut indirgeme teknikleri gibi.

Skaler (Scalar): Skaler, büyüklüğü(magnitude) belirten ve yalnızca bir sayıdan oluşan matematiksel bir nesnedir. Yani, sadece büyüklüğü vardır ve bir yöne veya konuma sahip değildir. Bir otomobilin sürati, bir kişinin yaşı veya oda sıcaklığı birer skaler değerdir.

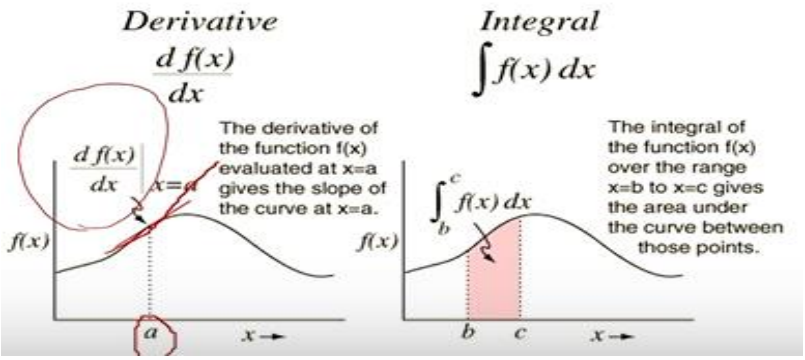
Vektör (Vector): Vektör hem büyüklüğü hem de yönü belirten matematiksel bir nesnedir. Başlangıç noktasından bir bitiş noktasına yönelen bir ok olarak düşünülebilir. Bir otomobilin hızı örneğin, 50 km/saat doğuya bir vektördür. Bir kuvvetin uygulandığı yön ve şiddeti de bir vektördür.

Matrisler: Matrisler, sayılar kümesinin satır ve sütunlarla düzenlenmiş bir düzlem tablosudur.

Lineer Denklemler: Lineer denklem, içinde yalnızca birinci dereceden terimlerin bulunduğu bir denklem sistemidir. Bu denklemler, bilinmeyen değişkenlerin doğrusal olarak ifade edildiği denklemlerdir. Makine öğrenmesi bağlamında, lineer denklemler genellikle modelin temelini oluşturur. Özellikle regresyon analizi ve sınıflandırma gibi temel makine öğrenmesi problemlerinde lineer modeller sıklıkla kullanılır.

Türev: Türev, bir fonksiyonun belirli bir noktasındaki anlık değişim oranını ifade eder. Özellikle gradyan inişi gibi optimizasyon teknikleri, bir fonksiyonun türevini kullanarak en uygun parametre değerlerini bulmaya çalışır. Modelin kayıp fonksiyonunun (loss function) türevi, parametre güncellemelerinde bir rehber olarak kullanılır.

İntegral: İntegral, bir fonksiyonun belirli bir aralıktaki alanını ifade eder. İntegral, bir fonksiyonun toplamını almak için kullanılır ve belirli aralıktaki sonsuz küçük parçaların toplamını hesaplamak için integral işareti kullanılır.



Olasılık: Olasılık, belirsizlik altında olayların olasılığını ve bu olayların matematiksel modellerini inceleyen bir alandır. Bir olayın gerçekleşme olasılığını sayısal olarak ifade eder.

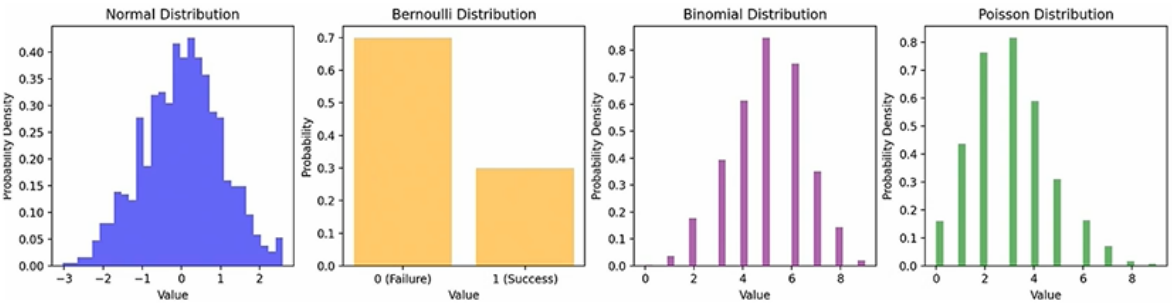
İstatistik: Veri toplama, analiz etme, yorumlama ve sonuç çıkarma süreçlerini inceleyen bir alandır. Bu süreçler, veri setlerinden anlamlı bilgiler çıkarılmasına ve karar verme süreçlerine destek sağlar.

Olasılık Temel Kavramları:

- ***Olay (Event):*** Bir deneyin sonucunda meydana gelebilecek belirli bir durumu ifade eder.
- ***Örnek Uzay (Sample Space):*** Bir deneyde oluşabilecek tüm olası sonuçların kümesidir.
- ***Koşullu Olasılık (Conditional Probability):*** Bir olayın diğer bir olayın gerçekleşme durumuna bağlı olarak gerçekleşme olasılığını ifade eder.

Olasılıksal Dağılımlar:

- ***Normal Dağılım (Gauss Dağılımı):*** Verilerin çoğunun ortalama etrafında toplandığı, çan eğrisi şeklinde bir dağılımdır.
- ***Bernoulli Dağılımı:*** Sadece iki olasılıklı (örneğin "başarılı/başarısız", "1/0") deneylerin olasılığını tanımlar.
- ***Binom Dağılımı:*** Birden fazla bağımsız Bernoulli denemesinin sonucundaki başarı sayısının dağılımını gösterir.
- ***Poisson Dağılımı:*** Belirli bir zaman veya alanda nadiren gerçekleşen olayların sayısını modellemek için kullanılır.



Temel İstatistiksel Özellikler:

* Ortalama (Mean) * Medyan * Mod * Varyans *Standart Sapma (Standart Deviation)

Gözetimli Öğrenme

Gözetimli öğrenme, eğitim verilerinde girdi ve çıktı arasındaki ilişkiyi öğrenmeyi amaçlayan bir makine öğrenmesi yaklaşımıdır. Bu tür öğrenme genellikle "etiketlenmiş veri" ile gerçekleştirilir, yani her girdiye karşılık gelen bir çıktı etiketi bulunur. Gözetimli öğrenmenin ana adımları şunlardır:

- Veri Toplama ve Hazırlığı
- Model Seçimi ve Eğitimi
- Model Değerlendirmesi
- Model Ayarlaması ve Hiperparametre Optimizasyonu

Gözetimli öğrenme, sınıflandırma ve regresyon gibi çeşitli problemleri çözmek için kullanılabilir. Örnek uygulamalar arasında hastalık teşhisi, müşteri tercihlerini tahmin etme, Pazar segmentasyonu ve hisse senedi fiyatlarını tahmin etme bulunmaktadır.

Gözetimli Öğrenme Algoritmaları:

- ***Sınıflandırma:***
  - K- En Yakın Komşu (KNN)
  - Karar Ağaçları
  - Rasgele Orman
  - Logistic Regresyon
  - Destek Vektör Makinesi
  - Naive Bayes
- ***Regresyon:***
  - Lineer Regresyon
  - Çoklu Lineer Regresyon
  - Polinom Regresyon

KNN (K-Nearest Neighbors):

K-Nearest Neighbors (KNN), makine öğrenmesinde sınıflandırma ve regresyon problemlerini çözmek için kullanılan basit, sezgisel ve etkili bir algoritmadır. Temel mantığı, yeni bir veri noktasını kendisine en yakın K adet komşu veri noktasıyla karşılaştırarak karar vermektir.

- **Sınıflandırmada:** Yeni veriyi, en yakın komşuların çoğunlukta olduğu sınıfa atar.
- **Regresyonda:** Komşuların ortalama değerini kullanarak tahmin yapar.

KNN Akış Şeması:

1. Komşu sayısı K belirlenir. Karar verirken kaç komşunun dikkate alınacağını belirler.
2. Tüm veri noktalarının mesafeleri hesaplanır Yeni veri noktasına olan uzaklıklar ölçülür.
3. En yakın K komşu seçilir. Hesaplanan mesafelere göre en yakın K nokta belirlenir.
4. Komşuların sınıf etiketleri sayılır. Her sınıf için kaç komşunun bulunduğu sayılır.
5. Yeni veri noktası, çoğunluk sınıfına atanır. En çok komşuya sahip sınıf, yeni veri için tahmin edilen sınıf olur.

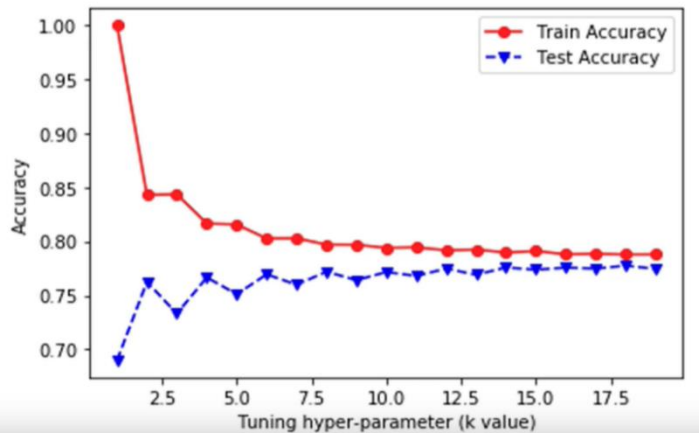
Distance: KNN algoritmasında "yakınlık" kavramı mesafe hesaplama yöntemlerine göre belirlenir:

- ***Euclidean distance:*** En yaygın kullanılan mesafe türüdür. İki nokta arasındaki "doğrusal uzaklığı" ölçer.
- ***Manhattan distance:*** Koordinatlar arasındaki farkların mutlak değerlerinin toplamıdır. "Şehir blok mesafesi" olarak da bilinir.
- ***Minkowski distance:*** Euclidean ve Manhattan mesafelerinin genelleştirilmiş hâlidir. p parametresiyle esneklik sağlar.
- ***Hamming distance:*** İki veri noktasındaki farklı bit sayısını ölçer. Genellikle kategorik veya ikili (binary) verilerde kullanılır.

Hiperparametre Optimizasyonu:

Elbow method, KNN algoritmalarında K değerini seçmek için kullanılan bir tekniktir. Bu yöntem, farklı K değerlerini deneyerek modelin performansını değerlendirir ve optimum K değerini belirlemeye çalışır. Elbow methodu, K değerinin artmasıyla birlikte modelin performansındaki azalışın hızının azalması ve eğrinin dirsek gibi bükülmesi prensibine dayanır. Bu dirsek noktası, K değerinin en iyi performansı sağladığı noktayı temsil eder.

- K değeri küçük olursa → model gürültüye duyarlı olur (overfitting).
- K değeri büyük olursa → model genelleme yapar ama ayrıntıları kaçırabilir (underfitting).



Bu grafik, KNN algoritmasında k değerinin doğruluğa etkisini gösterir.

Kırmızı çizgi (Train Accuracy): k küçükken eğitim doğruluğu çok yüksek → overfitting.

Mavi çizgi (Test Accuracy): k arttıkça test doğruluğu yükselir, sonra sabitlenir → çok büyük k'da underfitting olur.

- En uygun k, iki doğrunun dengede olduğu noktadır (yaklaşık k = 5–7).

Uygulama Alanları:

- **Sağlık:** Hastalık teşhisi veya ilaç öneri sistemlerinde kullanılır.
- **E-Ticaret:** Müşteri benzerliklerine göre ürün tavsiyeleri yapılır.
- **Finans:** Kredi risk analizi veya dolandırıcılık tespitinde kullanılır.

Zayıf ve Güçlü Yönleri:

Avantajları	Dezavantajları
Uygulaması kolay ve birçok durumda yüksek doğruluk sağlar.	Özellik sayısı arttıkça mesafe hesaplamaları anlamını yitirir (curse of dimensionality).
Eğitme aşaması gerekmez; sadece K değeri belirlenir.	Gürültülü veya dengesiz veriler sonucu etkileyebilir.
Verinin tamamı saklanır ve kararlar doğrudan veri üzerinden verilir.	Her tahmin için tüm veriyle mesafe hesaplaması gerekir, bu da büyük veri kümelerinde yavaştır.

Decision Trees (Karar Ağaçları):

Karar ağaçları, ağaç yapısı şeklindeki bir dizi karar kuralıyla veri kümesini bölerek hedef değişkenin tahmin edilmesini sağlar. Karar ağacı hem sınıflandırma hem de regresyon görevleri için kullanılan parametrik olmayan bir denetimli öğrenme algoritmasıdır. Bu algoritma, kök düğüm, dallar, iç düğümler ve yaprak düğümlerinden oluşan hiyerarşik bir ağaç yapısına sahiptir. Amaç, veriyi belirli koşullara göre alt gruplara ayırarak hedef değişkenin sınıfını (sınıflandırma) veya sayısal değerini (regresyon) tahmin etmektir.

Karar Ağaçları Türleri:

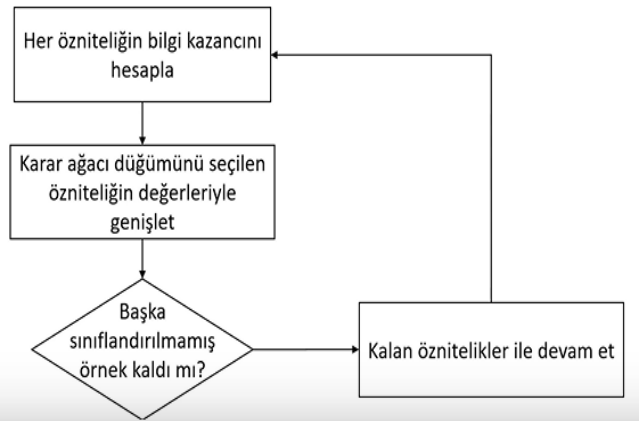
- ***ID3 (Iterative Dichotomiser 3):*** İlk geliştirilen karar ağacı algoritmalarından biridir. Entropi ve bilgi kazancı (Information Gain) ölçütlerini kullanarak en iyi bölünmeyi seçer. Kategorik (nitel) verilerle çalışmak için uygundur, ancak sürekli verilerde yetersiz kalır. Küçük ve orta ölçekli veri setleri için tercih edilir.

- ***C4.5*** ID3'ün geliştirilmiş versiyonudur. Karar ağaçlarında bölme noktalarını değerlendirmek için bilgi kazancı veya kazanç oranlarını kullanabilir. Hem sürekli hem de kategorik verilerle çalışabilir. Kazanç oranı (Gain Ratio) kullanarak daha dengeli bölünmeler yapar. Ağaç budama (pruning) işlemi sayesinde aşırı öğrenmeyi (overfitting) azaltır.

- ***CART (Classification and Regression Trees):*** Sınıflandırma ve regresyon problemlerinde kullanılabilir. Bölünme kararlarını verirken Gini impurity (Gini karışıklığı) ölçütünü kullanır. Gini karışıklığı, rastgele seçilen bir özelliğin ne sıklıkla yanlış sınıflandırıldığını ölçer. Gini karışıklığını kullanarak değerlendirirken, daha düşük bir değer daha idealdir. Rastgele ormanlar (Random Forests) ve Gradient Boosting gibi topluluk yöntemlerinin (ensemble) temelini oluşturur. Kayıp verilerle başa çıkabilir ve büyük veri setlerinde oldukça etkilidir.

	ID3	**C4.5**	**CART**
Veri Tipi	Kategorik verilerle çalışır.	Sürekli ve kategorik verilerle çalışabilir.	Sürekli ve kategorik verilerle çalışabilir.
Veri Boyutu	Küçük ve orta ölçekli veri setlerinde etkilidir.	Orta ve büyük ölçekli veri setlerinde kullanılabilir.	Büyük veri setleri için uygundur.
Hız	Hesaplaması yavaştır çünkü bilgi kazancı her düğümde hesaplanır.	Orta düzeyde hızlıdır.	Hızlıdır ve büyük veri setlerinde iyi performans gösterir.
Boosting	Desteklemez.	Desteklemez.	Destekler (Random Forest, Gradient Boosting gibi yöntemlerle birlikte kullanılabilir).
Kayıp Veri	Kayıp verilere karşı hassastır.	Orta düzeyde dayanıklıdır.	Kayıp verilerin üstesinden gelebilir.
Algoritma	Entropi ve Bilgi Kazancı kullanır.	Bilgi kazancı ve Kazanç Oranı kullanır.	Gini Impurity (Gini Karışıklığı) kullanır.

ID3: Bu akış şeması, karar ağacı oluşturma sürecini gösterir.



1. Bilgi kazancı hesapla: Her özelliğin hedefi ne kadar iyi ayırdığını ölç.

2. En iyi özelliği seç: En yüksek bilgi kazancına sahip özelliği düğüm olarak ekle.

3. Veriyi böl: Seçilen özelliğin değerlerine göre veriyi dallara ayır.

4. Kontrol et: Sınıflandırılmamış örnek kaldı mı?

- Kalmadıysa: Yaprak düğüm oluştur.
- Kaldıysa: Kalan özelliklerle işlemi tekrarla.

Bu döngü, tüm örnekler sınıflandırılana kadar devam eder.

Bilgi Kazancı (Information Gain):

Bir özelliğin bölünme noktasının ne kadar iyi olduğunu belirlemek için kullanılır. Amacı, bir özelliğin seçilmesinin, veri kümesinin ne kadar iyi sınıflandırılacağına dair belirsizliği azaltıp azaltmadığını ölçmektir. Yani, bir özellik seçildiğinde, yeni oluşan alt kümelerin daha homojen olup olmadığını değerlendirir. Bilgi kazancı ne kadar yüksekse, bir özellik o kadar iyi bir bölünme kriteri olarak kabul edilir.

$$Gain(S, A) = Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

Açıklaması:

- **Entropy(S):** Başlangıç veri kümesinin belirsizliğini ölçer.
- **S_v:** Özellik **A**'nın belirli bir değeri **v** için oluşan alt kümedir.
- $|S_v| / |S|$: Alt kümenin toplam veriye oranıdır.
- Entropy (S_v)**:** O alt kümenin belirsizliğini ölçer.

Formül, başlangıçtaki entropiden (belirsizlikten), bölünme sonrası alt kümelerin ağırlıklı entropi ortalamasını çıkarır. Sonuç, özellik A'nın bilgi kazancını verir. Bilgi kazancı yüksekse, özellik veri kümesini daha iyi ayırır ve karar ağacında düğüm olarak seçilir.

Entropi:

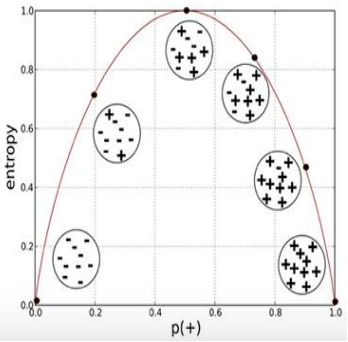
Karar ağaçları (decision trees) içinde entropi, belirli bir düğümdeki veri noktalarının sınıflarının ne kadar karışık olduğunu ölçen bir metriktir. Daha düşük bir entropi değeri, bir düğümün daha homojen sınıflara sahip olduğunu yani veri noktalarının çoğunluğunun aynı sınıfa ait olduğunu gösterir. Daha yüksek bir entropi değeri ise, bir düğümün daha karışık sınıflara sahip olduğunu, yani veri noktalarının farklı sınıflara daha eşit olarak dağıldığını gösterir.

Entropi

$$E(S) = \sum_{i=1}^{c} - p_i \log_2 p_i$$

Play Golf	
Yes	No
9	5

Entropy(PlayGolf) = Entropy (5,9)
= Entropy (0.36, 0.64)
= - (0.36 $\log_2$ 0.36) - (0.64 $\log_2$ 0.64)
= 0.94



Burada:

Eğri, **p (+) = 0,5** civarında en yüksek olur (entropi ≈ 1). Bu, sınıflar yarı yarıya karışıksa belirsizliğin maksimum olduğunu gösterir. Eğer p (+) ≈ 0 veya p (+) ≈ 1 ise, entropi ≈ 0 olur.
Bu durumda veri tamamen saf (pure) demektir; örneğin tüm örnekler "Yes" ise artık belirsizlik yoktur.

Özetle, bilgi kazancını (information gain) hesaplamak için kullanılır. Karar ağacında, hangi özelliğin veriyi en iyi ayırdığını bulmak için önce her özelliğin entropisi hesaplanır.

- **Düşük entropi → saf (homojen) grup, Yüksek entropi → karışık (heterojen) grup**

AKADEMİ

ID3 Durum Çalışması

$$Gain(S, A) = Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$E(S) = \sum_{i=1}^{c} - p_i \log_2 p_i$$

Outlook	Temperature	Humidity	Wind	Played football(yes/no)
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Overcast	Hot	High	Weak	Yes
Rain	Mild	High	Weak	Yes
Rain	Cool	Normal	Weak	Yes
Rain	Cool	Normal	Strong	No
Overcast	Cool	Normal	Strong	Yes
Sunny	Mild	High	Weak	No
Sunny	Cool	Normal	Weak	Yes
Rain	Mild	Normal	Weak	Yes
Sunny	Mild	Normal	Strong	Yes
Overcast	Mild	High	Strong	Yes
Overcast	Hot	Normal	Weak	Yes
Rain	Mild	High	Strong	No

CART algoritması, karar ağacını oluşturmak için şu adımları izler:

1. Her özelliğin Gini impurity değerini hesaplar.
2. En düşük Gini değerine sahip özelliği seçer.
3. Veriyi bu özelliğe göre dallara ayırır.
4. Tüm örnekler sınıflanana kadar kalan özelliklerle aynı işlemi tekrarlar.

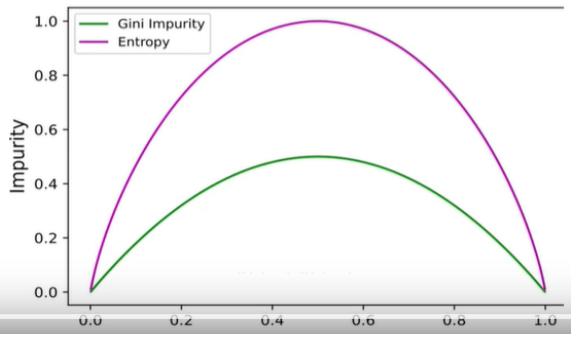
Amaç: Veriyi en saf (en az karışık) alt gruplara ayırmaktır.

CART Algoritması Nedir?

CART (Classification and Regression Tree), yani Sınıflandırma ve Regresyon Ağacı, karar ağaçları oluşturmak için kullanılan popüler bir algoritmadır.
Amaç: Veriyi en saf (en az karışık) alt gruplara ayırmaktır.

Gini Impurity: Gini belirsizliği, bir başka karar ağacı algoritması olan CART (Sınıflandırma ve Regresyon Ağaçları) tarafından kullanılan bir ölçümdür. Bir düğümdeki veri noktalarının farklı sınıflara dağılımının ne kadar karışık olduğunu ölçer. Gini karışıklığı, 0 ile 0,5 arasında bir değer alır, 0 en saf durumu (yani tüm veri noktalarının aynı sınıfa ait olduğu durumu), 0.5 ise en karışık durumu ifade eder. Bir düğümün Gini karışıklığı ne kadar düşükse, o kadar homojen sınıflara sahip olur ve böylece daha iyi bir bölünme kriteri olarak kabul edilir.

$$Gini\ Impurity = 1 - \sum p_i^2$$



Özetle:

Kavram	Açıklama
Amaç	Veriyi en saf alt gruplara ayırmak
Kullanılan Ölçüt	Gini impurity
Gini Değeri	0 → saf, 0.5 → karışık
Seçim Kuralı	En düşük Gini değerine sahip özellik seçilir
Sonuç	Her bölünmede veriler daha homojen hale gelir

CART Durum Çalışması

Weekend	Weather	Parents	Money	Decision
W1	Sunny	Yes	Rich	Cinema
W2	Sunny	No	Rich	Tennis
W3	Windy	Yes	Rich	Cinema
W4	Rainy	Yes	Poor	Cinema
W5	Rainy	No	Rich	Stay In
W6	Rainy	Yes	Poor	Cinema
W7	Windy	No	Poor	Cinema
W8	Windy	No	Rich	Shopping
W9	Windy	Yes	Rich	Cinema
W10	Sunny	No	Rich	Tennis

Karar Ağaçları Hiperparametreleri:

- ***Maksimum Derinlik (max_depth):*** Ağacın alabileceği en fazla seviye sayısını belirler. Derinlik arttıkça model karmaşıklaşır ve aşırı uyum (overfitting) riski artar.
- ***Minimum Bölme Boyutu (min_samples_split):*** Bir düğümün dallanabilmesi için gerekli minimum örnek sayısını belirler. Bu değeri artırmak, ağacın çok küçük alt gruplara bölünmesini engeller.
- ***Minimum Yaprak Boyutu (min_samples_leaf):*** Yaprak düğümünde bulunması gereken en az örnek sayısını belirler. Küçük değerler daha fazla yaprak oluşturur, büyük değerler modeli sadeleştirir.
- ***Maksimum Özellikler (max_features):*** Her bölünmede dikkate alınacak maksimum özellik (değişken) sayısını belirler. Bu parametre, rastgelelik ekleyerek aşırı uyumu azaltmaya yardımcı olur.
- ***Bölünme Kriteri (criterion):*** Dallanma sırasında hangi ölçütün kullanılacağını belirler. Sınıflandırma için genellikle "gini" veya "entropy", regresyon için "mse" (mean squared error) kullanılır.

Not: Bu hiperparametrelerin uygun değerleri genellikle Kapsamlı Arama (Grid Search) veya Çapraz Doğrulama (Cross Validation) yöntemleriyle belirlenir. Böylece modelin hem doğruluğu hem de genelleme başarımı optimize edilir.

Uygulama Alanları:

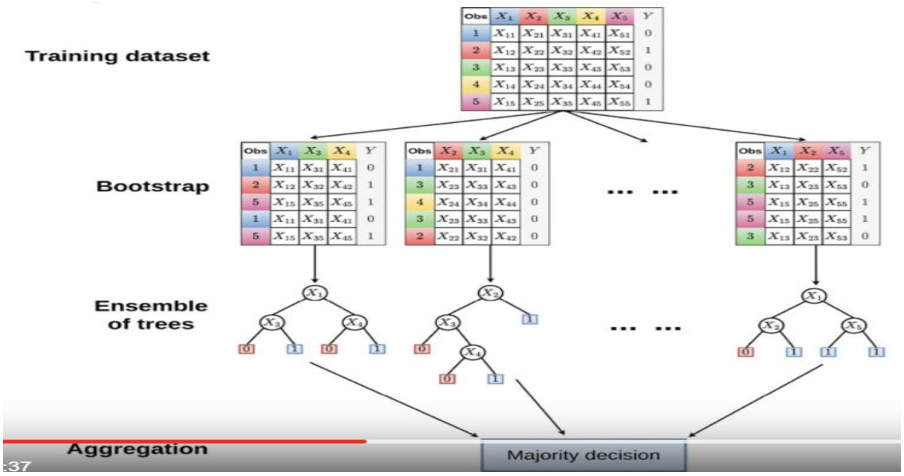
- *Sınıflandırma Problemleri:* Örnekleri kategorilere ayırmak için kullanılır. (Örnek: E-posta → spam / spam değil)
- *Regresyon Problemleri:* Sürekli bir değerin tahmini için kullanılır. (Örnek: Ev fiyatı tahmini)
- *Pazarlama ve Satış Analizi:* Müşteri segmentasyonu, ürün önerisi ve satış tahminlerinde kullanılır.
- *Risk Değerlendirmesi*: Bankacılık veya sigortacılıkta kredi riski, dolandırıcılık tespiti gibi alanlarda uygulanır.
- *Sağlık ve Tıp:* Hastalık tahmini, teşhis destek sistemleri ve hasta risk analizi için kullanılır.
- *Endüstriyel Üretim ve Kalite Kontrol:* Üretim hatalarının nedenlerini belirlemek ve kaliteyi artırmak için tercih edilir.

Avantajları	Dezavantajları
Anlaşılabilirlik = Karar ağaçları görsel ve mantıksal olarak kolay yorumlanabilir. "Eğer... ise..." yapısı insan mantığına uygundur.	Aşırı Uyum (Overfitting): Ağaç çok derinleştiğinde eğitim verisine aşırı uyar ve yeni verilere genelleme yapamaz.
Değişken Önemi = Hangi özelliklerin tahmin üzerinde daha etkili olduğunu belirleyebilir.	Duyarlılık = Küçük veri değişiklikleri bile ağacın yapısını büyük ölçüde değiştirebilir.
Çoklu Çıkışlı Modellere Uygunluk = Aynı anda birden fazla sınıf veya çıktıyı tahmin edebilir.	Dengesiz Veri Kümeleri = Bir sınıfın çok baskın olduğu veri kümelerinde dengesiz sonuçlar üretebilir.
Veri Ön İşleme İhtiyacının Azalması = Normalizasyon veya ölçeklendirme gibi ön işlemler genellikle gerekmez.	

Rastgele Orman (Random Forest):

Rastgele Ormanlar, birden fazla karar ağacının bir araya gelerek oluşturduğu bir topluluk öğrenme (ensemble learning) yöntemidir. Her bir karar ağacı, eğitim verisinden rastgele seçilen alt örneklem (bootstrap sample) ve/veya rastgele seçilen özelliklerin alt kümesi (random subset of features) ile eğitilir. Sonuçta, her ağacın çıktısı birleştirilerek nihai tahmin elde edilir.

Rastgele Orman Türleri: *Sınıflandırma için Rastgele Ormanlar *Regresyon için Rastgele Ormanlar



Çalışma Mantığı

1. **Bootstrap Örnekleme:** Eğitim verisinden rastgele örneklem alınır.
2. **Ağaçların Eğitilmesi:** Her karar ağacı bu örneklem üzerinde, rastgele seçilen özelliklerin alt kümesini kullanarak eğitilir.
3. **Tahminlerin Birleştirilmesi (Aggregation):**

- Sınıflandırma için: Çoğunluk oylaması (majority voting) yapılır.
- Regresyon için: Ağaçların tahminlerinin ortalaması alınır.

Rastgele Orman Hiperparametreleri:

- Ağaç sayısı (n_estimators): Oluşturulacak karar ağacı sayısıdır. Ağaç sayısı arttıkça doğruluk artabilir ancak eğitim süresi uzar.
- Özelliklerin sayısı (max_features): Her ağacın bölünme noktasında kullanabileceği maksimum özellik sayısıdır. Modelin çeşitliliğini artırır.
- Ağaç derinliği (max_depth): Her bir karar ağacının maksimum derinliğini belirler. Ağaç çok derinse overfitting riski artabilir.
- Örnekleme oranı (bootstrap): Verinin örneklenme biçimini kontrol eder. True ise bootstrap yöntemiyle örnekleme yapılır.

Zayıf ve Güçlü Yönleri:

Zayıf Yönler	Güçlü Yönler
Eğitim süreci tek bir karar ağacına göre daha uzun olabilir.	Overfitting'e karşı dirençlidir, genelleme yeteneği yüksektir.
Çok yüksek veya çok düşük boyutlu veri setlerinde aynı performansı göstermeyebilir.	Değişkenlerin önem derecesini belirleme yeteneğine sahiptir.
Modelin yapısı karmaşık olduğundan yorumlanması zor olabilir.	Eksik veriye karşı dayanıklıdır ve yüksek doğruluk sağlar.

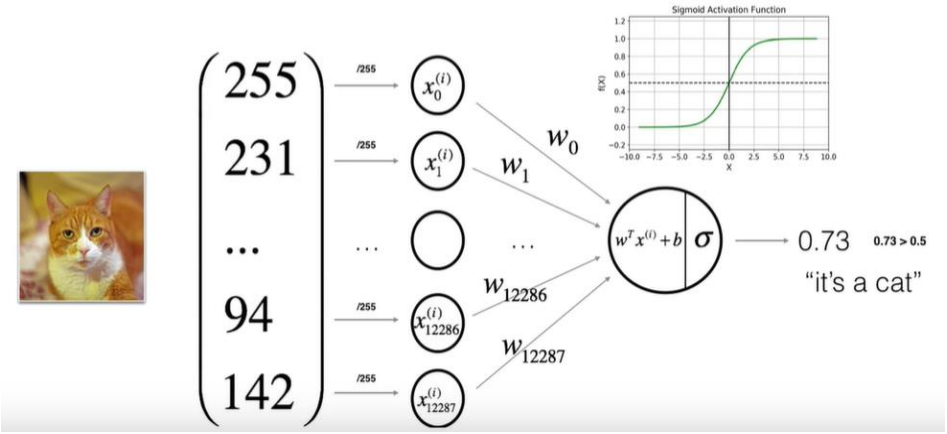
Lojistik Regresyon:

Lojistik regresyon, istatistik ve makine öğrenmesi alanlarında **sınıflandırma** problemlerini çözmek için kullanılan bir yöntemdir. Genellikle bağımlı değişkenin kategorik olduğu durumlarda kullanılır. Amaç, bir olayın gerçekleşme olasılığını tahmin etmektir. Örneğin:

- Bir müşterinin bir ürünü satın alıp almayacağı,
- Bir e-postanın spam olup olmadığı,
- Bir hastalığa sahip olma olasılığı gibi durumlar, lojistik regresyon ile modellenebilir.

Lojistik Regresyon Tahmini:

Lojistik regresyonda tahmin işlemi, ileri yayılım (forward propagation) süreciyle gerçekleştirilir.



1. Girdi (Input): Bu resim, bilgisayar tarafından sayısal değerlere (piksel değerleri) dönüştürülüyor. Her pikselin 0–255 arası bir değeri vardır (örneğin parlaklık veya renk yoğunluğu). Görseldeki sütun (255, 231, ..., 94, 142) bu piksel değerlerini temsil ediyor.

2. Normalizasyon (Normalization): Her piksel değeri 255'e bölünerek 0–1 aralığına ölçekleniyor: Bu işlem, modelin daha verimli öğrenmesini ve hesaplamaların daha stabil olmasını sağlar.

3. Ağırlıklarla Çarpma (Weighted Sum): Her girdi değeri bir ağırlık (weight) değeriyle çarpılır. Ardından tüm bu ağırlıklı değerler toplanır ve bir bias (b) eklenir: [**z = w^T* x + b]**

4. Aktivasyon Fonksiyonu (Sigmoid): Lojistik regresyonda kullanılan aktivasyon fonksiyonu sigmoid fonksiyonudur (σ). Sigmoid fonksiyonu giriş olarak aldığı (z) değerini 0 ile 1 arasında bir olasılığa dönüştürür: [**σ(z) = 1/ (1 + e^(-z))**]

Sağ üstteki grafik, sigmoid fonksiyonunun şeklini gösteriyor.

- Küçük (z) değerlerinde sonuç 0'a yakın, Büyük (z) değerlerinde sonuç 1'e yakın olur.

5. Çıktı (Output) — Tahmin

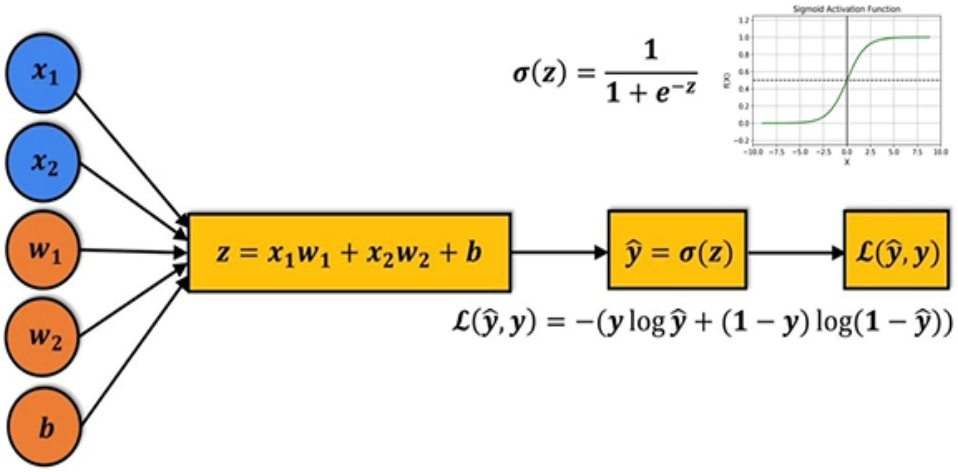
Hesaplanan olasılık değeri, örneğin: [σ(z) = 0.73] olarak çıkıyor. Bu, modelin "görüntüde bir kedi olma olasılığı %73" demesi anlamına gelir. Eğer bu değer 0.5'ten büyükse, model sonucu:

"It's a cat" (Bu bir kedi) olarak sınıflandırır. (Eğer 0.5'ten küçük olsaydı "not a cat" denecekti.)

Lojistik Regresyon Eğitimi:

Lojistik regresyon modeli, **geri yayılım (backpropagation)** yöntemiyle eğitilir. Bu süreçte model, hatasını minimize edecek şekilde ağırlık ve bias değerlerini günceller. Eğitim adımları şu şekildedir:

- ***Ağırlık(w) ve Biasların(b) Başlatılması:*** Ağırlıklar ve bias değerleri genellikle küçük rastgele sayılarla başlatılır. Bu, modelin simetrik olmayan bir şekilde öğrenmesini sağlar.
- ***İleri Yayılım (Forward Propagation):*** Girdi verileri modele uygulanır ve mevcut ağırlık (w) ile bias (b) değerleri kullanılarak tahmin edilen değer ($\hat{y}$) hesaplanır.
- ***Loss Fonksiyonunun Hesaplanması:*** Gerçek değer (y) ile modelin tahmini ($\hat{y}$) karşılaştırılır ve aradaki fark kayıp (loss) olarak ölçülür.
- ***Geri Yayılım (Backward Propagation):*** Kayıp fonksiyonunun türevi alınarak her bir ağırlığın kayıptaki etkisi ($\frac{\partial L}{\partial w}$) hesaplanır. Bu bilgi, modelin hangi yönde değişmesi gerektiğini belirlemek için kullanılır.
- ***Ağırlıkların Güncellenmesi:*** Hata azaltılacak şekilde ağırlıklar yeniden ayarlanır:
  $w = w - \alpha \frac{\partial L}{\partial w}$
  Burada α öğrenme oranıdır (learning rate). Bu süreç, model hatası kabul edilebilir düzeye inene kadar tekrar edilir.



Bu görsel, lojistik regresyonda ileri yayılım (forward propagation) ve kayıp (loss) hesaplama sürecini gösterir. Adım adım:

1. Girdiler (Inputs): Görselin solunda iki adet giriş (özellik) var:

- x_1, x_2: Özellik (feature) değerleri — örn. görüntü pikselleri, yaş, tansiyon vb.
- w_1, w_2: Bu özelliklere karşılık gelen ağırlıklar (weights). Modelin her özelliğe verdiği göreli önemi belirler.
- b: Bias (sapma) terimi. Karar sınırını kaydırarak modeli esnekleştirir; tüm örneklere sabit bir katkı ekler.

2. Lineer Kombinasyon (Z Hesabı): Girdiler ağırlıklarla çarpılıp toplanır ve bias eklenir:

$$z = x_1w_1 + x_2w_2 + b \text{(vektörel olarak } z = \mathbf{w}^T\mathbf{x} + b\text{)}$$

Bu işlemde amaç, giriş verilerini tek bir sayısal değere (z) dönüştürmektir. Bu işlem "linear combination" (doğrusal birleşim) olarak adlandırılır.

3. Aktivasyon Fonksiyonu (Sigmoid): Lineer çıktı z, sigmoid aktivasyonuna sokulur: $\boldsymbol{\sigma(z) = \frac{1}{1+e^{-z}}}$
Bu, z'yi 0 ile 1 arasında bir olasılık değeri olan ŷ(y şapka) şeklinde verir: $\boldsymbol{\hat{y} = \sigma(z)}$

- $\hat{y}$ modelin "pozitif sınıfa ait olma olasılığı" tahminidir. Karar eşiği genelde 0.5'tir: $\hat{y} \geq 0.5 \Rightarrow$sınıf 1, aksi halde sınıf 0. Ayrıca $z = 0$ doğrusal karar sınırını temsil eder.

4. **Kayıp Fonksiyonu (Loss Function):** Modelin tahmini $\hat{y}$ile gerçek etiket $y \in \{0,1\}$ karşılaştırılır. Tek örnek için binary cross-entropy (log loss) şöyle yazılır:
$\boldsymbol{\mathcal{L}(\hat{y}, y) = -[y \log(\hat{y}) + (1-y) \log(1-\hat{y})]}$

- Eğer $y = 1$: $\mathcal{L} = -\log(\hat{y})$. (Doğruysa loss küçük, yanlışsa büyük.)
- Eğer $y = 0$: $\mathcal{L} = -\log(1-\hat{y})$.

5. İleri Yayılım ve Sonrası

Forward propagation: Modelin tahmin yapma aşamasıdır (girdi → z→ ŷ→ loss). Bu loss değeri, geri yayılım (backpropagation) aşamasında kullanılarak ağırlıkların etkisi ($\partial\mathcal{L}/\partial w$) hesaplanır ve ağırlıklar güncellenir. Böylece model, her iterasyonda hatasını azaltarak daha iyi tahmin yapmayı öğrenir.

Lojistik Regresyon Hiperparametreleri:

- **Penalty (Cezalandırma):** Ağırlıkların büyüklüğünü sınırlayarak overfitting riskini azaltır.
- **C (Cezalandırma katsayısı):** Regülarizasyon gücünü belirler. Küçük C değeri → daha güçlü cezalandırma.
- **Solver (Çözücü):** Optimizasyonun nasıl yapılacağını belirler (örneğin: 'liblinear', 'saga').
- **Max Iterations:** Modelin eğitim sırasında geçeceği maksimum adım sayısıdır.
- **Class Weight:** Dengesiz veri kümelerinde azınlık sınıflarına daha fazla önem verir.

Uygulama Alanları:

- **Tıp ve Sağlık:** Hastalık teşhisi, risk analizi, hasta sonuçlarının tahmini
- **Finans:** Kredi geri ödeme olasılığı, dolandırıcılık (fraud) tespiti
- **İnsan Kaynakları:** İşe alım tahmini, çalışan memnuniyeti veya ayrılma olasılığı
- **Pazarlama:** Müşteri terk (churn) tahmini, satın alma davranışı analizi

Zayıf ve Güçlü Yönleri:

Özellikler	Zayıf Yönler	Güçlü Yönler
Yapı	Lineer bir modeldir; karmaşık, doğrusal olmayan ilişkileri iyi modelleyemez.	Basit, hızlı ve yüksek yorumlanabilirliğe sahiptir.
Overfitting Riski	Büyük özellik setleriyle ve yüksek boyutlu veri setleriyle eğitildiğinde aşırı uydurmaya eğilimlidir.	Regülarizasyon (L1, L2) kullanılarak bu durum kontrol altına alınabilir.
Çıktılar	Sınıflandırma problemlerine odaklanır, regresyon problemleri için uygun değildir.	Sınıf olasılıklarını ve sınıf tahminlerini sağlar.
Veri Ölçeği	Özelliklerin ölçeklenmesi yapılmazsa dengesiz sonuçlar üretebilir.	Normalize edilmiş verilerde oldukça stabil ve güvenilir sonuçlar verir.

Support Vector Machines (Destek Vektör Makineleri):

Destek Vektör Makinesi (SVM), sınıflandırma ve regresyon problemlerini çözmek için kullanılan güçlü bir makine öğrenimi algoritmasıdır. SVM'nin temel amacı, veri kümesindeki farklı sınıfları en iyi şekilde ayıran hiperdüzlemi (decision boundary) bulmaktır. Bu hiperdüzlem, sınıflar arasındaki marjini (margin) maksimize edecek şekilde seçilir. SVM, özellikle doğrusal olmayan (non-linear) veri kümeleri için de etkilidir. Çünkü Kernel (çekirdek) yöntemi sayesinde veriler daha yüksek boyutlu bir uzaya dönüştürülerek doğrusal olarak ayrılabilir hale getirilebilir.

Destek Vektör Makineleri Hiperparametreleri:

1. C (Ceza Parametresi): Modelin hataya ne kadar tolerans göstereceğini belirler.

- Küçük C değeri → daha geniş margin, ancak bazı sınıflandırma hatalarına izin verilir (daha basit model).
- Büyük C değeri → hataya karşı daha duyarlı model, margin daralır (aşırı uyum riski artar).

2. Kernel (Çekirdek) Tipi: Doğrusal olmayan verileri daha yüksek boyutlu uzaylara dönüştürmek için kullanılan fonksiyon türüdür. En yaygın kernel tipleri:

- **Linear (Doğrusal):** Veriler doğrusal olarak ayrılabiliyorsa kullanılır.
- **Polynomial (Polinom):** Veriler polinom biçiminde bir sınırla ayrılabiliyorsa uygundur.
- **RBF (Radial Basis Function):** En yaygın kullanılan çekirdektir, karmaşık ve doğrusal olmayan ilişkileri iyi yakalar.
- **Sigmoid:** Genellikle sinir ağlarındaki aktivasyon fonksiyonlarına benzer dönüşüm yapar.

Uygulama Alanları:

- **Tıp:** Hastalık teşhisi, genetik veri analizi, tümör sınıflandırma.
- **Finans:** Kredi riski analizi, dolandırıcılık tespiti, borsa tahmini.
- **Görüntü İşleme:** Nesne tanıma, yüz tanıma, el yazısı karakter tanıma.
- **Biyoinformatik:** Protein sınıflandırma, DNA dizilimi analizi.
- **Metin ve Dil İşleme:** Spam e-posta tespiti, duygu analizi, belge sınıflandırma.

Zayıf ve Güçlü Yönleri:

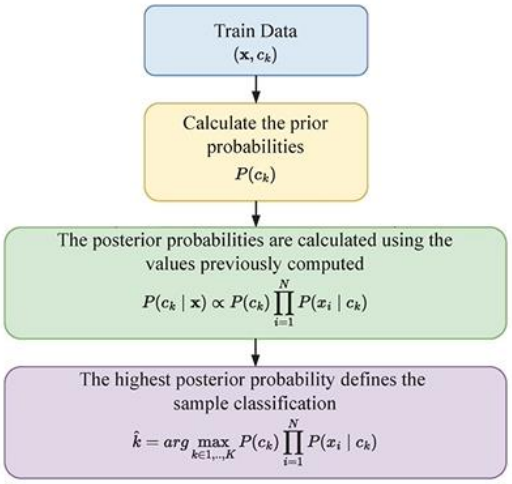
Zayıf Yönler	Güçlü Yönler
Büyük veri setlerinde eğitim süresi uzar ve yüksek hesaplama maliyeti gerektirir.	Doğrusal ve doğrusal olmayan sınıflandırma problemlerinde oldukça etkilidir.
Gürültülü (noisy) ve örtüşen sınıflı verilerde performans düşebilir.	Kernel yöntemi sayesinde karmaşık veri yapılarıyla başa çıkabilir.
Parametre (özellikle C ve kernel seçimi) ayarları doğru yapılmazsa aşırı uyum (overfitting) görülebilir.	Az sayıda veriyle bile yüksek doğruluk oranı elde edebilir.

Naive Bayes:

Naive Bayes, olasılık temelli bir sınıflandırma algoritmasıdır ve makine öğreniminde özellikle metin sınıflandırma ve duygu analizi gibi görevlerde yaygın olarak kullanılır. Bu yöntem, Bayes Teoremi'ne dayanır ve her bir özelliğin (feature) sınıf etiketinden bağımsız olduğunu varsayar. Bu varsayım nedeniyle "naive" (saf/basit) olarak adlandırılır.

Naive Bayes Akış Şeması

$$P(A|B) = \frac{P(B|A)\ P(A)}{P(B)}$$



Uygulama Alanları:

- **Metin Sınıflandırma:** Spam e-posta tespiti, haber kategorilendirme.
- **Duygu Analizi:** Sosyal medya gönderilerinde olumlu/olumsuz duygu belirleme.
- **Tıbbi Teşhis:** Belirli hastalıkların olasılık tabanlı tahmini.
- **Pazarlama:** Müşteri segmentasyonu, satın alma eğilimi tahmini.
- **Biyoinformatik:** Gen veya protein sınıflandırması.

Zayıf ve Güçlü Yönleri:

Özellikler	Zayıf Yönler	Güçlü Yönler
Hiperparametreler	Daha az hiperparametreye sahip olması, bu nedenle daha az esneklik sağlar.	Basit, hızlı ve düşük veri gereksinimiyle yüksek performans sağlar.
Öznitelik Bağımsızlığı	Öznitelikler arasındaki bağımsızlık varsayımı gerçek dünyada her zaman geçerli değildir.	Metin sınıflandırma ve spam filtreleme gibi birçok uygulama alanında başarılıdır.
Hassasiyet	Genellikle çok karmaşık yapıları yakalayamaz, bu nedenle yüksek hassasiyet gerektiren problemlerde performansı düşük olabilir.	Veri seti dengesizliğine ve gürültülü veriye karşı dayanıklıdır.

Durum Çalışması

Day	Outlook	Temp	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Outlook	Yes	No
Sunny	2/9	3/5
Overcast	4/9	0/5
Rainy	3/9	3/5

Temp	Yes	No
Hot	2/9	3/5
Mild	4/9	2/5
Cool	3/9	2/5

Humidity	Yes	No
High	3/9	4/5
Normal	6/9	1/5

Wind	Yes	No
Strong	3/9	3/5
Weak	6/9	2/5

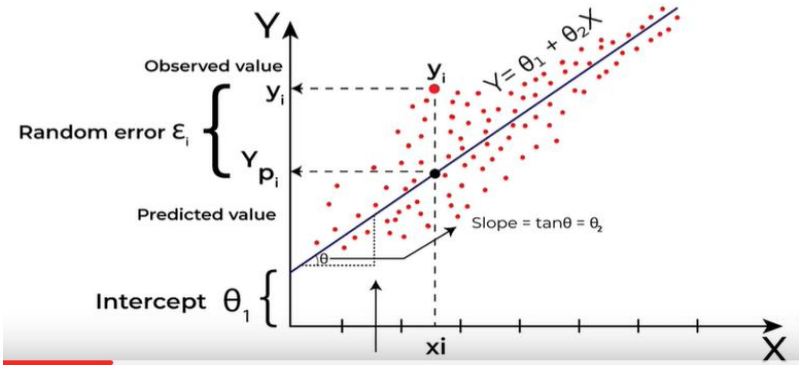
Lineer Regresyon:

Lineer regresyon, bağımlı değişken (y) ile bir veya daha fazla bağımsız değişken (X) arasındaki doğrusal ilişkiyi modelleyen istatistiksel bir yöntemdir. Amaç, veri noktalarına en iyi uyan bir doğruyu (regresyon doğrusunu) bulmaktır. Bu doğru, şu genel formülle ifade edilir:

$$\boldsymbol{y = b_0 + b_1 x_1 + b_2 x_2 + \ldots + b_n x_n + \epsilon}$$

$\boldsymbol{y}$: Bağımlı değişken, $\boldsymbol{x_i}$: Bağımsız değişkenler, $\boldsymbol{b_i}$**:** Katsayılar, $\boldsymbol{\epsilon}$**:** Hata terimidir.

* Modelin temel amacı, hata karelerinin toplamını (SSE- Sum of Squared Errors) minimize etmektir.



Lineer Regresyon Türleri:

1. Basit Lineer Regresyon: Yalnızca bir bağımsız değişken ile bir bağımlı değişken arasındaki doğrusal ilişkiyi inceler. Örnek: Reklam harcamasına göre satış tahmini.

2. Çoklu Lineer Regresyon: Birden fazla bağımsız değişkenin, bir bağımlı değişken üzerindeki etkisini analiz eder. Örnek: Fiyat tahmini (oda sayısı, metrekare, konum gibi birden fazla faktörle).

3. Polinom Regresyon: Değişkenler arasındaki ilişki doğrusal değilse, veriye eğri uydurmak için polinom terimleri eklenir. Örnek: Yaş ile gelir arasındaki doğrusal olmayan ilişki.

4. Ridge ve Lasso Regresyonu (Regularization Yöntemleri): Aşırı öğrenmeyi (overfitting) önlemek için katsayılara ceza ekler.

Uygulama Alanları:

- **Ekonomi:** Hisse senedi fiyat tahmini, gelir-gider analizi, ekonomik büyüme tahminleri.
- **Pazarlama:** Reklam harcamasının satışa etkisi, müşteri davranışı analizi.
- **Sağlık Bilimleri:** Yaşa, kiloya veya tedaviye bağlı sağlık sonuçlarını tahmin etme
- **Mühendislik:** Üretim maliyeti tahmini, enerji tüketim analizi.

Zayıf ve Güçlü Yönleri:

Zayıf Yönler	Güçlü Yönler
Doğrusal olmayan ilişkileri yakalamakta zorlanabilir.	Basit, anlaşılır ve yüksek yorumlanabilirliğe sahiptir.
Aykırı değerlere (outlier) karşı hassastır, bu da modeli bozabilir.	Hızlı çalışır ve kısa eğitim süresine sahiptir.
Değişkenler arasında çoklu doğrusal ilişki (multicollinearity) varsa sonuçlar güvenilir olmayabilir.	Özellikle doğrusal ilişki gösteren verilerde yüksek doğruluk sağlar.

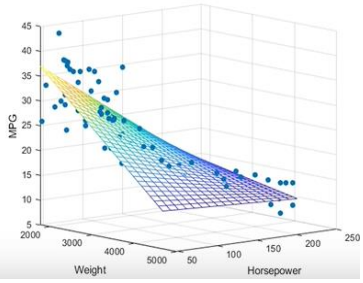
Çoklu Lineer Regresyon

- Çoklu Lineer Regresyon, birden fazla bağımsız değişkenin kullanıldığı bir regresyon türüdür.

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_n X_n$$

Burada:

- Y, bağımlı değişken,
- $X_1, X_2, \ldots, X_n$, bağımsız değişkenler,
- $\beta_0, \beta_1, \beta_2, \ldots, \beta_n$, regresyon katsayıları



Örnek olarak, bir evin satış fiyatını tahmin etmek için çoklu lineer regresyon kullanalım. Bağımlı değişkenimiz ev fiyatı olsun. Bağımsız değişkenlerimiz ise aşağıdaki gibi faktörler olsun:

- Ev alanı
- Oda sayısı
- Banyo sayısı
- Mahalle güvenlik puanı

$$EvFiyatı = \beta_0 + \beta_1 * EvAlanı + \beta_2 * OdaSayısı + \beta_3 * BanyoSayısı + \beta_4 * MahalleGüvenlikPuanı + \varepsilon$$

Uygulama Alanları:

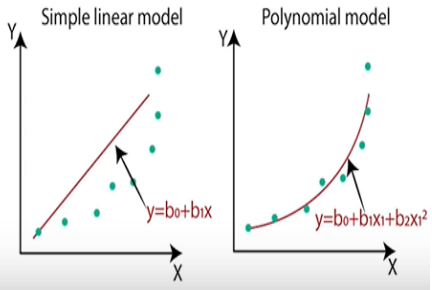
- **Pazarlama:** Satış tahminleri, reklam etkisi analizi
- **Finans:** Hisse senedi fiyat tahmini, kredi risk analizi
- **Sağlık Bilimleri:** Hastalık risk faktörlerinin analizi
- **Eğitim:** Öğrenci başarısını etkileyen faktörlerin incelenmesi

Polinom Regresyon:

$$Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \ldots + \beta_n X^n$$

Burada:

- Y, bağımlı değişken,
- X, bağımsız değişken,
- $\beta_0, \beta_1, \beta_2, \ldots, \beta_n$, regresyon katsayıları,
- n, polinomun derecesi (2, 3, 4, vb.).



Lineer regresyonun genişletilmiş bir versiyonudur ve doğrusal olmayan ilişkileri modellemek için kullanılır. Polinom regresyonunda bağımsız değişkenlerin doğrusal birleşimleri yerine polinom terimleri kullanılır.

Polinom Regresyon Hiperparametreleri:

Polinom regresyonun hiperparametreleri, genellikle polinomun derecesini belirlemek için seçilir. Polinom regresyonunda hiperparametre seçimini etkileyen bazı faktörler şunlardır:

- **Modelin Performansı:** Derece arttıkça model karmaşıklaşır ve veriye daha iyi uyum sağlayabilir; ancak aşırı uyum (overfitting) riski de artar.
- **Aşırı Uyum ve Aşırı Basitlik:** Düşük derece → verinin karmaşıklığını yakalayamaz. Yüksek derece → modeli veriye fazla uydurur, genelleme gücü düşer.
- **Bilgi Kaybı:** Derece seçimi yapılırken, modelin hem veriye uyum sağlaması hem de yeni veriler üzerinde doğru tahminler yapabilmesi için denge kurulmalıdır.

Uygulama Alanları:

- **Fizik ve Mühendislik:** Fiziksel değişkenler arasındaki doğrusal olmayan ilişkileri modelleme.
- **Finans ve Ekonomi**: Faiz oranı, gelir değişimlerinin zamanla eğrisel etkilerini tahmin etme.
- **İklim Bilimi:** Sıcaklık, nem, karbon emisyonu faktörlerin uzun vadeli değişimini inceleme.
- **Tıp ve Sağlık Bilimleri:** Hastalık ilerleme eğrileri, ilaç dozuna bağlı tepki modelleri.
- **Eğitim ve Öğrenme Analizi:** Öğrenme hızının zaman içindeki değişimini modelleme.

Gözetimsiz Öğrenme

Gözetimsiz öğrenme, etiketlenmemiş veri kümesi üzerinde yapılan bir makine öğrenmesi yaklaşımıdır. Bu yöntemde, veri kümesindeki yapıyı anlamak, gizli desenleri keşfetmek ve veri noktaları arasındaki ilişkileri belirlemek amaçlanır. Gözetimsiz öğrenme genellikle veri keşfi, veri ön işleme ve özellik çıkarımı (feature extraction) gibi aşamalarda kullanılır.

Gözetimsiz Öğrenme Kullanım Alanları:

- **Kümeleme (Clustering):** Veri noktalarını benzer özelliklerine göre gruplandırmak.
- **Boyut Azaltma (Dimensionality Reduction):** Yüksek boyutlu veriyi daha az boyutla temsil ederek hesaplama yükünü azaltmak (örneğin: PCA, t-SNE).
- **Değişken Keşfi (Feature Discovery):** Verideki önemli özellikleri otomatik olarak ortaya çıkarmak.
- **Yoğunluk Tahmini (Density Estimation):** Veri noktalarının dağılımını modelleyerek olasılık yoğunluğunu tahmin etmek.
- **Anomalilerin Tespiti (Anomaly Detection):** Verideki normal örüntüye uymayan olağandışı (anormal) noktaları belirlemek.

Kümeleme (Clustering): Kümeleme, veri noktalarını benzerliklerine göre gruplandırma işlemidir. Her küme, birbirine benzeyen verilerden oluşur. Bu yöntem, veri içindeki doğal yapıyı ortaya çıkarmak için kullanılır. Başlıca kümeleme algoritmaları:

- K-Means Kümeleme
- Hiyerarşik Kümeleme
- DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

K-Means Clustering (K-Ortalamalı Kümeleme):

K-Means, en yaygın kullanılan gözetimsiz öğrenme algoritmalarından biridir. Amaç, veri kümesini önceden belirlenmiş K adet kümeye ayırmaktır. Her küme, küme merkezine (centroid) olan uzaklık temel alınarak oluşturulur.

K-Means Algoritmasının Adımları:

1. Küme merkezlerini başlat: Rastgele K adet merkez seçilir.
2. Veri noktalarını kümelere ata: Her veri noktası, en yakın merkeze ait kümeye atanır.
3. Küme merkezlerini yeniden hesapla: Her kümedeki noktaların ortalaması alınarak yeni merkez belirlenir.
4. Yeniden atama yap: Noktalar yeni merkezlere göre tekrar kümelenir.
5. Yakınsama kontrolü: Merkezler değişmiyorsa algoritma durur.

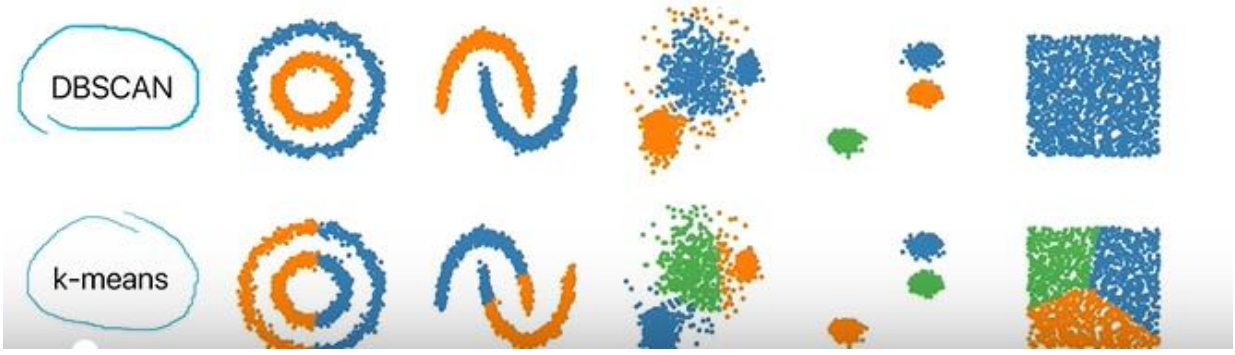
Hiyerarşik Kümeleme

Hiyerarşik kümeleme, verileri bir ağaç benzeri yapı (dendrogram) içinde gruplandırır. Başlangıçta her veri noktası tek başına bir küme olarak kabul edilir, ardından kümeler aralarındaki benzerliğe göre birleştirilir (veya bölünür). Hiyerarşik kümeleme, 2 ana türde gerçekleştirilir:

- Agglomerative (Birleştirici): Her veri noktası kendi kümesindedir; benzer kümeler adım adım birleştirilir.
- Divisive (Bölücü): Tüm veri tek bir kümede başlar; ardından adım adım bölünür**.**

DBSCAN (Gürültülü Uygulamaların Yoğunluğa Dayalı Mekansal Kümelemesi)

DBSCAN (Density-Based Spatial Clustering of Applications with Noise), gözetimsiz bir kümeleme algoritmasıdır. Özellikle belirsiz şekilli kümelere sahip veya gürültü içeren veri setlerini analiz etmek için kullanılır. DBSCAN, veri noktaları arasındaki yoğunluk ve mesafe ilişkilerine dayanarak kümeler oluşturur. Yoğun bölgelerdeki noktalar aynı kümeye ait kabul edilirken, düşük yoğunluklu bölgelerdeki noktalar gürültü (noise) olarak değerlendirilir. Bu algoritma, önceden belirli bir küme sayısına ihtiyaç duymaz ve karmaşık yapılı veri setlerinde etkili sonuçlar verebilir. Ayrıca, aykırı değerlere karşı dayanıklı olması DBSCAN'in en önemli avantajlarından biridir.



Bu görsel, DBSCAN ve K-Means algoritmalarının farklı veri kümeleri üzerindeki kümeleme performanslarını karşılaştırıyor

- **DBSCAN:** DBSCAN, veri noktalarının yoğunluklarına göre kümeler oluşturur. Bu yüzden karmaşık veya eğri şekilli kümeleri (örneğin spiral veya iç içe halkalar) başarıyla ayırabilir. Ayrıca gürültü (noise) noktalarını da tespit edebilir.
- **K-Means:** K-Means kümeleri küresel (dairesel) yapıda varsayar ve her kümeyi merkeze göre ayırır. Bu nedenle spiral gibi düzensiz şekilli verilerde başarısız olur; kümeleri yanlış şekilde böler. Ancak düzenli ve ayrık veri kümelerinde iyi sonuç verir.

📊 **Özet:**

- DBSCAN → Karmaşık şekiller ve gürültülü verilerde güçlü.
- K-Means → Basit ve düzgün ayrılmış verilerde etkili.

Pekiştirmeli Öğrenme (Reinforcement Learning)

Pekiştirmeli öğrenme, bir yapay zekâ (AI) modelinin ortamla etkileşime girerek belirli bir görevi nasıl gerçekleştireceğini deneme-yanılma yoluyla öğrenmesini sağlayan bir öğrenme türüdür. Amaç, modelin (ajanın) toplam ödülü maksimize edecek davranışı öğrenmesidir.

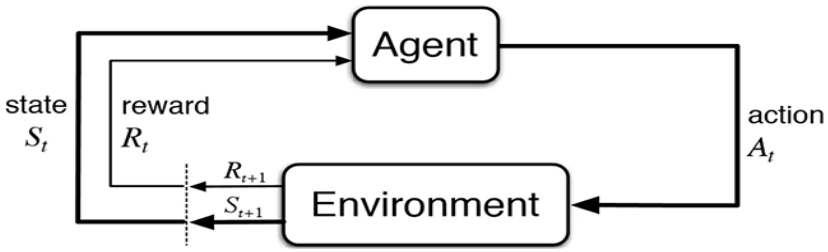
Pekiştirmeli Öğrenme Türleri:

- **Model Tabanlı Pekiştirmeli Öğrenme:** Ajan, ortamın nasıl davrandığını modellemeyi öğrenir ve bu modele göre eylem planı oluşturur.
- **Model Tabanlı Olmayan Öğrenme:** Ortamın modelini öğrenmez, doğrudan deneyimlerden (ödül ve durum değişimlerinden) öğrenir.
- **Değer Tabanlı Öğrenme:** Her durum veya eylem için bir "değer" hesaplanır (Q-learning).
- **Politika Tabanlı Öğrenme:** Ajan, doğrudan hangi durumda hangi eylemi yapacağını belirleyen bir politika öğrenir.

Pekiştirmeli Öğrenme Terimleri:

- **Agent (Ajan):** Ortamda karar verip eylem yapan yapay zekâ birimi.
- **Environment (Ortam):** Ajanın etkileşimde bulunduğu dış dünya.
- **State (Durum – S_t):** Ajanın bulunduğu mevcut ortamın tanımı (örneğin konum, hız, sıcaklık).
- **Action (Eylem – A_t):** Ajanın belirli bir durumda yaptığı hareket veya seçim.
- **Reward (Ödül – R_t):** Ortamın, ajanın yaptığı eyleme karşı verdiği geri bildirim.
    - İyi eylem → yüksek ödül
    - Kötü eylem → düşük ya da negatif ödül
- **Policy (Politika):** Ajanın her durumda hangi eylemi seçeceğini belirleyen strateji.
- **Value:** Belirli bir durumun veya eylemin gelecekteki ödül açısından ne kadar "iyi" olduğunu gösteren ölçü.
- **Q-Value:** Belirli bir durum-eylem çiftinin beklenen toplam ödülünü gösterir.

Pekiştirmeli Öğrenme Nasıl Çalışır?



Süreç döngüsü:

1. Ajan, ortamın mevcut durumunu (S_t) gözlemler.
2. Bu duruma göre bir eylem (A_t) seçer.
3. Ortam, bu eyleme tepki verir ve yeni durumu (S_{t+1}) ile ödülü (R_{t+1}) gönderir.
4. Ajan, aldığı ödüle göre stratejisini günceller.

🎯 **Amaç**: Ajanın uzun vadede toplam ödülü maksimize edecek bir strateji (policy) öğrenmesidir.

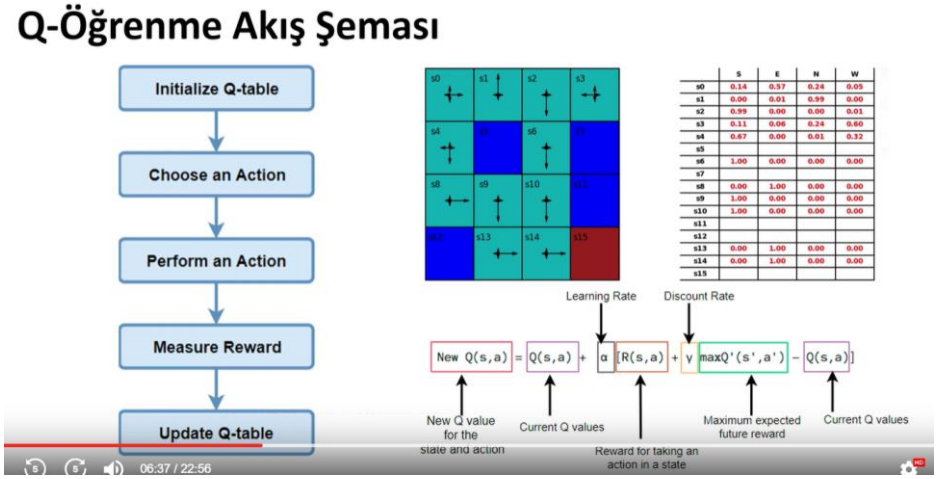
Pekiştirmeli Öğrenme Uygulamaları:

- **Robotik:** Robotların çevreleriyle etkileşime girerek yürüme, nesne tutma veya yön bulma gibi görevleri deneme-yanılma yoluyla öğrenmesini sağlar.
- **Kontrol:** Otonom araçlar, enerji sistemleri veya üretim hatları gibi dinamik süreçlerin en verimli şekilde yönetilmesinde kullanılır.
- **Oyun Oynama:** Satranç, Go, Atari gibi oyunlarda stratejik kararlar alabilen ve insanı yenebilen yapay zekâ ajanları geliştirmede kullanılır.
- **İşletme:** Tedarik zinciri yönetimi, kaynak planlama veya müşteri etkileşimleri gibi süreçlerde verimlilik ve kâr optimizasyonu sağlar.
- **Üretim:** Otomasyon sistemlerinin üretim hızını, kalite kontrolünü ve kaynak kullanımını optimize etmesine yardımcı olur.
- **Finans:** Portföy yönetimi, alım-satım stratejileri ve risk analizlerinde dinamik karar verme süreçlerini geliştirmek için kullanılır.

Pekiştirmeli Öğrenme Algoritmaları: 1. Q-Learning- 2. Deep Q-Learning

1. Q-Learning:

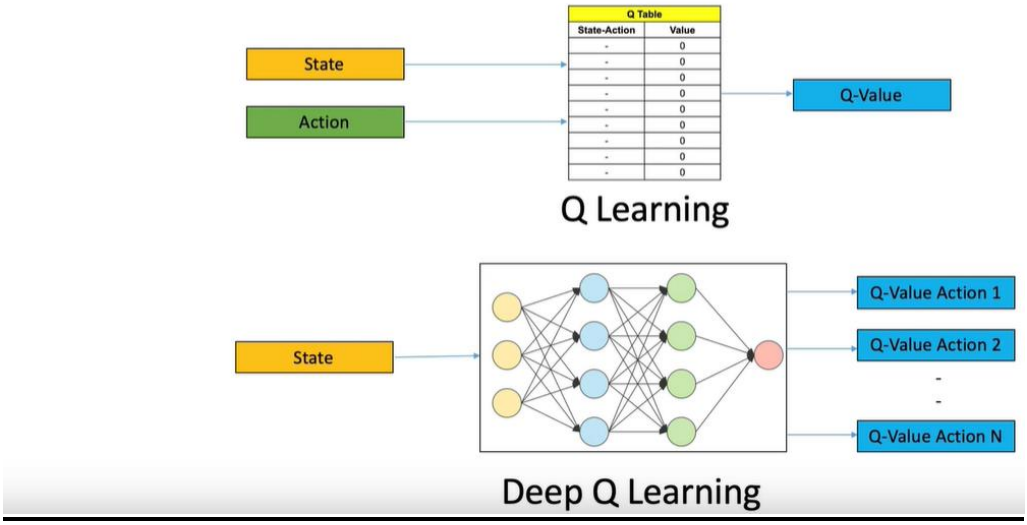
Q-Learning, pekiştirmeli öğrenme alanında kullanılan temel bir öğrenme tekniğidir. Bu yöntem, bir ajanın bulunduğu ortamda etkileşimde bulunarak deneyimlerinden öğrenmesini ve en uygun stratejiyi geliştirmesini sağlar. Temel olarak ajan, ortamda çeşitli eylemler gerçekleştirir, bu eylemlerin sonuçlarını gözlemler elde ettiği ödüller doğrultusunda gelecekte en uygun eylemleri seçmeyi öğrenir.



Q-Learning Avantajları ve Dezavantajları:

Avantajlar	*Dezavantajlar*
Model bağımsızdır: Ortamın dinamiklerini (geçiş olasılıkları, ödül fonksiyonları) bilmeye gerek yok.	Keşif ve exploitation dengesi doğru kurulmazsa öğrenme süreci verimsiz olabilir.
Ajan, doğrudan en iyi politikayı öğrenir, belirli bir politikaya bağlı kalmaz.	Boyut problemi: Durum ve eylem uzayları büyüdükçe Q-tablosunun boyutu artar.
Farklı ortamlara ve görev türlerine kolayca uyarlanabilir, esnektir.	Bazı durumlarda tahmini Q-değerleri gerçekte olduğundan yüksek olabilir.
Gerçek ortamda etkileşim gerekmeden önceden toplanmış verilerle öğrenme mümkündür.	Öğrenme süreci uzun sürebilir ve optimal performansa ulaşmak zaman alabilir.

Q-Öğrenme ve Derin Q-Öğrenme



Q-Learning, her durum-eylem çifti için bir Q-tablosu oluşturarak öğrenme yaparken, Deep Q-Learning bu tabloyu bir yapay sinir ağı (neural network) ile tahmin eder.

- Q-Learning, küçük ve sınırlı durum uzaylarında etkili çalışır ancak büyük veya sürekli durum uzaylarında ölçeklenemez.
- Deep Q-Learning ise derin öğrenme mimarileri sayesinde karmaşık, yüksek boyutlu ortamlarda daha başarılıdır.
- DQN, görüntü, ses gibi ham verilerden doğrudan öğrenebilirken Q-Learning bunu yapamaz. Ancak DQN daha fazla hesaplama gücü ve veri gerektirir.

Sonuç olarak, Q-Learning basit ortamlar için uygunken, Deep Q-Learning daha genel ve güçlü bir yaklaşım sunar.

Pekiştirmeli Öğrenme Yöntemleri:

- **Monte Carlo Methods:** Bir problemdeki olası durumları ve sonuçları rastgele örnekleyerek değer tahminleri yapan yöntemdir.
- **SARSA (State-Action-Reward-State-Action):** Öğrenme sırasında ajan, mevcut durum ve aksiyonu, aldığı ödül ve sonraki durumdaki aksiyonu dikkate alarak Q-değerlerini günceller.
- **Policy Gradient:** Politika parametrelerini ödülü maksimize edecek şekilde gradyan yöntemiyle güncellemektir.
- **Actor-Critic:** Actor, aksiyonları seçerken kritik tarafından verilen geri bildirimle öğrenir.
- **TD-Learning:** Monte Carlo gibi tüm bölümü beklemeden, her adımda ödül ve bir sonraki durumun değerini kullanarak tahminleri günceller.
- **Thompson Sampling:** Her aksiyonun ödül dağılımını olasılıksal olarak tahmin eder ve rastgele seçim yaparken belirsizlikten faydalanır.
- **UCB:** Her aksiyonun tahmini ödülünü ve belirsizlik sınırını kullanarak en yüksek üst güven sınırına sahip aksiyonu seçer.

Boyut İndirgeme Nedir?

Makine öğrenmesinde boyut indirgeme (dimensionality reduction), veri setindeki özellik (değişken) sayısını azaltarak modelin performansını artırmak, hesaplama maliyetini düşürmek veya veriyi daha anlaşılır hale getirmek amacıyla kullanılan bir tekniktir. Boyut indirgeme iki temel şekilde yapılabilir:

1. Öznitelik Seçimi (Feature Selection): Var olan özellikler arasından en anlamlı ve etkili olanları seçmek anlamına gelir. Gereksiz, tekrarlı veya bilgi değeri düşük özellikler çıkarılır.
Örneğin: Korelasyonu yüksek iki değişkenden biri çıkarılabilir.

2. Öznitelik Çıkarma (Feature Extraction): Mevcut özelliklerden yeni özellikler türetilir. Bu işlem genellikle matematiksel dönüşümlerle yapılır. Amaç, orijinal bilgiyi koruyarak boyutu azaltmaktır.
Örneğin: PCA veya LDA teknikleri bu gruba girer.

Temel Bileşen Analizi (PCA- Principal Component Analysis)

PCA, çok boyutlu veri setlerindeki değişkenliği azaltmak için kullanılan bir denetimsiz boyut indirgeme tekniğidir.Veri setindeki değişkenler arasındaki korelasyonu dikkate alarak, bu değişkenlerin birleşiminden oluşan yeni, bağımsız değişkenler (temel bileşenler) oluşturur. Amaç, verideki bilgiyi kaybetmeden daha az sayıda boyutla veriyi temsil etmektir.

PCA Adımları:

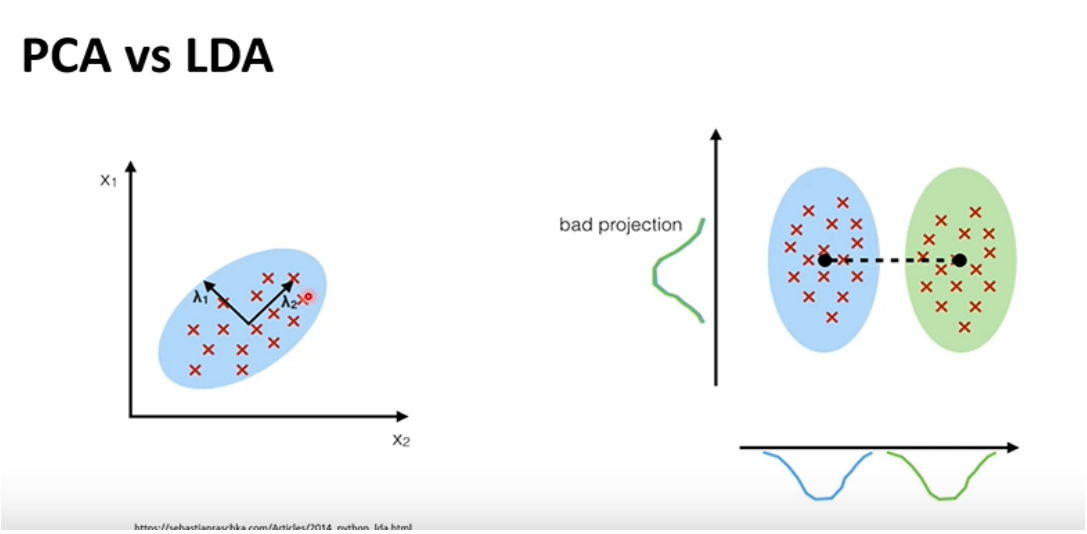
5. **Veri Normalizasyonu:** Tüm özelliklerin aynı ölçeğe getirilmesi.
6. **Kovaryans Matrisinin Hesaplanması:** Değişkenler arasındaki ilişkiyi (korelasyonu) ölçer.
7. **Özdeğer ve Özvektörlerin Hesaplanması:** Verideki en fazla varyansı açıklayan yönleri bulmak için kullanılır.
8. **Özdeğerlerin Sıralanması:** En büyük özdeğere sahip bileşenler en fazla bilgiyi taşır.
9. **Temel Bileşenlerin Seçimi:** İstenilen varyans oranını açıklayan bileşenler seçilir.
10. **Yeni Veri Setinin Oluşturulması:** Veriler seçilen bileşenler doğrultusunda yeniden ifade edilir.

Doğrusal Diskriminant Analizi (LDA):

LDA, özellikle sınıflandırma problemlerinde kullanılan denetimli bir boyut indirgeme yöntemidir. Amaç, sınıflar arasındaki farkı en iyi şekilde ayıran yeni bir uzay oluşturmaktır.

LDA Adımları:

1. **Sınıf Ortalama Vektörlerinin Hesaplanması:** Her sınıf için ortalama değer belirlenir.
2. **Sınıf İçi ve Sınıflar Arası Dağılımın Hesaplanması:** Her sınıfın kendi içindeki varyansı ve sınıflar arasındaki fark ölçülür.
3. **Sınıflar Arasındaki Ayrımın Maksimize Edilmesi:** En iyi ayırımı yapan doğrultular bulunur.
4. **Veri Setinin Dönüştürülmesi:** Veriler bu yeni doğrultulara göre yeniden ifade edilir.



Özellik	PCA (Temel Bileşen Analizi)	LDA (Doğrusal Diskriminant Analizi)
Amaç	Değişkenliği azaltmak ve veri setini daha az boyutta ifade etmek	Sınıflar arasındaki ayrımı maksimize etmek.
Denetimli/Denetimsiz	Denetimsiz	Denetimli
Kullanım Alanları	Veri görselleştirmesi, veri sıkıştırma, gürültü azaltma	Sınıflandırma problemleri, sınıf ayrımını iyileştirme
Hesaplama	Veri setinin kovaryans matrisinin özdeğer ve özvektörleri	Sınıf ortalamaları ve kovaryans matrislerinin hesaplanması
Sonuçlar	Değişkenler arasındaki ilişkileri yakalar	Sınıflar arasındaki farkı en iyi şekilde yakalar

t-Dağıtılmış Stokastik Komşu Yerleştirme (t-SNE- t-Distributed Stochastic Neighbor Embedding)

t-SNE, özellikle yüksek boyutlu veri setlerinin görselleştirilmesi için kullanılan bir nonlineer (doğrusal olmayan) boyut indirgeme tekniğidir. Veri setindeki noktalar arasındaki benzerlikleri koruyarak, veriyi genellikle 2 veya 3 boyuta indirger ve görsel olarak kümelenme yapısını ortaya çıkarır.

t-SNE Adımları:

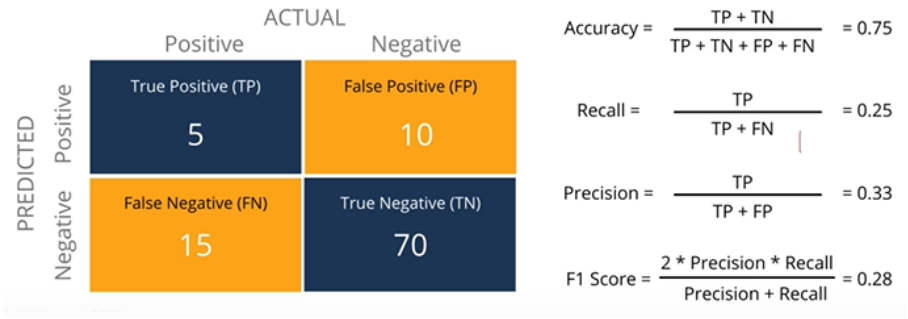
1. **Yüksek Boyutlu Uzayda Benzerlik Ölçümü:** Noktalar arasındaki olasılıksal benzerlikler hesaplanır.
2. **Düşük Boyutlu Uzaya Dönüştürme:** Bu benzerlikler, düşük boyutlu uzayda olabildiğince korunarak yeni bir yerleştirme yapılır.

Not: t-SNE genellikle veri keşfi ve görselleştirme amaçlı kullanılır; model eğitimi için doğrudan kullanılmaz.

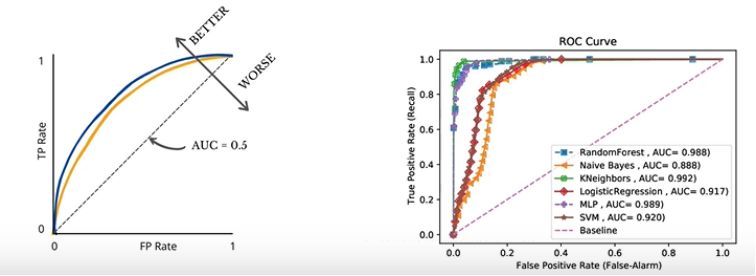
Değerlendirme Metriği Nedir?

Makine öğrenmesi modellerinin başarısını ölçmek için çeşitli değerlendirme metrikleri kullanılır. Bu metrikler, modelin tahmin performansını nicel olarak ifade eder. Problem türüne göre kullanılan metrikler farklılık gösterir: classification ve regression problemleri için farklı metrikler tercih edilir.

<u>Sınıflandırma için Değerlendirme Metrikleri:</u>



- ***Doğruluk (Accuracy):*** Modelin doğru tahmin ettiği örneklerin, toplam örnek sayısına oranıdır.
- ***Kesinlik (Precision):*** Pozitif olarak tahmin edilen örneklerin ne kadarının gerçekten pozitif olduğunu gösterir.
- ***Duyarlılık (Recall):*** Gerçek pozitif örneklerin ne kadarının doğru şekilde tahmin edildiğini ifade eder.
- ***F1-Skoru (F1-Score):*** Precision ve Recall değerlerinin harmonik ortalamasıdır; dengesiz veri kümelerinde özellikle önemlidir.
- ***ROC Skoru (ROC-AUC):*** "Receiver Operating Characteristic" eğrisi altında kalan alanı temsil eder. Modelin pozitif ve negatif sınıfları ayırt etme yeteneğini ölçer.



- ***Karmaşıklık Matrisi (Confusion Matrix):*** Gerçek ve tahmin edilen sınıfların karşılaştırıldığı bir tablodur; modelin hangi sınıflarda hata yaptığını görselleştirir.

<u>Regresyon için Değerlendirme Metrikleri:</u>

- ***Ortalama Mutlak Hata (Mean Absolute Error- MAE):*** Gerçek ve tahmin edilen değerler arasındaki ortalama mutlak farkı gösterir.
- ***Ortalama Kare Hata (Mean Squared Error- MSE):*** Gerçek ve tahmin edilen değerler arasındaki farkların karelerinin ortalamasıdır.
- ***Kök Ortalama Kare Hata (Root Mean Squared Error- RMSE):*** MSE'nin kareköküdür; hata değerini orijinal birimlerde gösterir.
- ***R-Kare (R-Squared):*** Bağımsız değişkenlerin, bağımlı değişkendeki toplam varyansın ne kadarını açıkladığını gösterir. Modelin veriyle ne kadar iyi uyum sağladığını ifade eder.

Model Seçimi Nedir?

Makine öğrenmesinde model seçimi, belirli bir problem için en uygun modelin belirlenmesi sürecidir. Bu süreç; veri setinin özellikleri, problemin doğası ve hedeflenen çıktılar göz önünde bulundurularak gerçekleştirilir. Model seçimi, makine öğrenmesi çalışmalarında kritik bir adımdır çünkü:

- Modelin performansını doğrudan etkiler.
- Genelleme yeteneğini belirler.
- Kullanılan hesaplama kaynaklarını optimize eder.
- Modelin yorumlanabilirlik ve açıklanabilirlik düzeyini belirler.
- Modelin uygulanabilirliğini artırır.

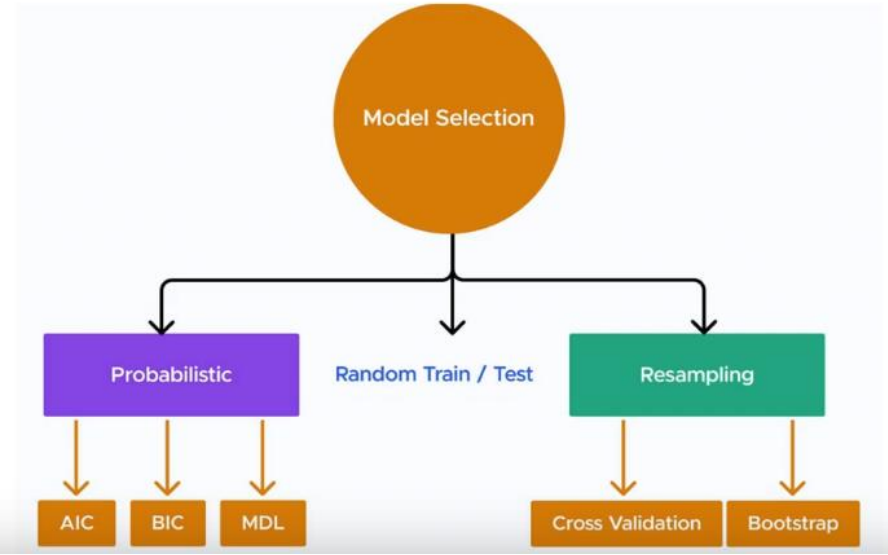
Model Seçimi Kriterleri:

En uygun modeli seçmek, yalnızca istatistiksel yöntemlerin değil, aynı zamanda alan bilgisi ve uzman değerlendirmesinin de bir kombinasyonunu gerektirir. Basit ya da karmaşık, doğrusal ya da doğrusal olmayan, geleneksel ya da derin öğrenme tabanlı modeller arasındaki seçim şu faktörlere bağlıdır:

- Problem Türü (sınıflandırma, regresyon, kümeleme vb.)
- Veri Erişilebilirliği ve Kalitesi
- Bilgisayar Kaynakları (CPU/GPU, bellek)
- Modelin Açıklanabilirliği ve Yorumlanabilirliği
- Ölçeklenebilirlik (Scalability)
- Çoklu Girişleri İşleme Yeteneği
- Zaman Kısıtları (Eğitim/Test süresi)
- Alan Bilgisi (Domain Expertise)

Model Seçme Yöntemleri:

Model seçimi süreci, çeşitli istatistiksel ve deneysel yaklaşımlar kullanılarak yürütülür. Bu yöntemler modelin performansını, genelleme yeteneğini ve karmaşıklığını dengeli biçimde değerlendirir.



1. Olasılıksal: Olasılıksal yöntemler, modelin veri kümesine ne kadar iyi uyduğunu ve ne kadar karmaşık olduğunu değerlendirmek için kullanılır. Bu yöntemler genellikle En Büyük Olabilirlik Tahmini (Maximum Likelihood Estimation – MLE) temeline dayanır ve model performansını hem uyum hem de karmaşıklık açısından değerlendirir. Başlıca olasılıksal model seçim kriterleri:

- ***Akaike Bilgi Kriteri (AIC – Akaike Information Criterion):*** Modelin veriyle ne kadar iyi uyum sağladığını ve ne kadar karmaşık olduğunu dengelemeyi amaçlar.
- ***Minimum Açıklama Uzunluğu (MDL – Minimum Description Length):*** Veri setinin kodlanması ve modelin karmaşıklığı arasındaki toplam bilgi miktarını (bit sayısını) en aza indirmeyi hedefler.
- ***Bayesyen Bilgi Kriteri (BIC – Bayesian Information Criterion):*** AIC'ye benzer şekilde modelin karmaşıklığı ile uyumu arasında denge kurar, ancak model parametre sayısını daha fazla cezalandırır.

2. Rastgele Bölünme: Bu yöntem, veri setini rastgele veya zamana dayalı olarak eğitim, doğrulama ve test kümelerine böler.

- Rastgele bölme, modelin farklı veri alt kümelerinde performansını değerlendirerek genelleme yeteneğini ölçer.
- Zaman tabanlı bölme ise zaman bileşeni içeren verilerde (örneğin hava durumu, borsa verisi gibi) geleceğe yönelik tahminlerde tutarlılığı sağlar.

3. Örnekleme: Örnekleme yöntemleri, modelin eğitim almadığı veri örnekleri üzerinde performansını görmek için veri örneklerini yeniden düzenlemenin basit yöntemleridir. Başka bir deyişle, örnekleme, modelin genelleme yeteneğini belirlememizi sağlar. Örnekleme yöntemleri:

a. Bootstrap*:* Bu teknik, bir veri setinden rastgele örneklemler çekmek suretiyle tekrarlanan örnekleme yaparak, popülasyon üzerinde istatistiksel sonuçların tahmin edilmesini sağlar. Bootstrap yöntemi, genellikle küçük veri setleri veya sınırlı veriye sahip durumlarda parametrik olmayan istatistiklerin güven aralıklarını ve standart hatalarını hesaplamak için kullanılır.

b. Çapraz Doğrulama

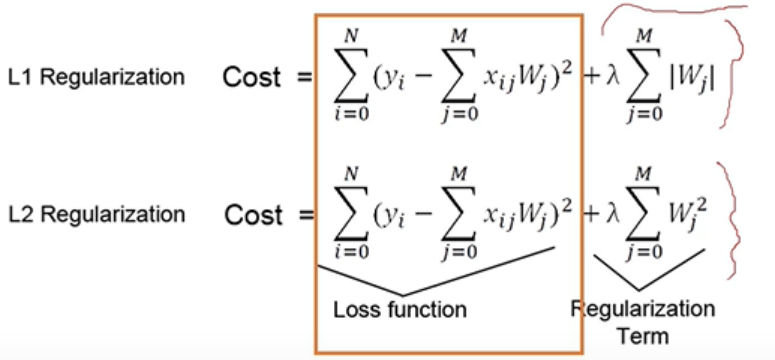
- ***K-Fold Çapraz Doğrulama:*** Veri seti k eşit parçaya bölünür (katman). Ardından, her bir katman sırayla test seti olarak kullanılırken, diğer katmanlar birleştirilerek eğitim seti oluşturulur. Bu işlem n kere tekrarlanır ve her bir katmanın test seti üzerindeki performansı değerlendirilir. Sonuçlar genellikle ortalamalar alınarak rapor edilir.
- ***Leave-One-Out (LOO) Çapraz Doğrulama:*** Her bir veri noktası sırayla test seti olarak ayrılırken, geri kalan veri noktaları eğitim seti olarak kullanılır. Bu işlem veri noktalarının tamamı için tekrarlanır ve her bir veri noktasının test seti üzerindeki performansı değerlendirilir.

Regularizasyon:

Makine öğrenmesinde regularizasyon, bir modelin aşırı uyuma (overfitting) eğilimini azaltmak veya önlemek için kullanılan bir tekniktir. Regularizasyon, modelin karmaşıklığını kontrol altına alarak genelleme yeteneğini artırır. Bu yöntem, modelin parametrelerine ceza (penalty) terimleri ekleyerek öğrenme sürecini dengeler.

L1 Regularizasyon (Lasso Regularization): L1 regularizasyonu, model katsayılarının mutlak değerlerinin toplamını ceza terimi olarak ekler (L1 normu). Bu yöntem, bazı katsayıların tamamen sıfıra inmesine neden olur; dolayısıyla model öznitelik seçimi (feature selection) yapabilir. Sonuç olarak, daha sade ve genelleme kabiliyeti yüksek bir model elde edilir.

L2 Regularizasyon (Ridge Regularization): L2 regularizasyonu, model katsayılarının karelerinin toplamını ceza terimi olarak ekler (L2 normu). Bu teknik, büyük katsayıların değerini küçülterek modelin aşırı uyum göstermesini engeller. Ancak katsayıları tamamen sıfıra indirmez; dolayısıyla tüm öznitelikler modelde kalır, sadece etkileri azaltılır.



Özellik	L1 Regülarizasyonu (Lasso)	L2 Regülarizasyonu (Ridge)
Öznitelik Seçimi	Gereksiz öznitelikleri tamamen ortadan kaldırabilir.	Tüm öznitelikleri korur, ancak etkilerini küçültür.
Robustluk (Dayanıklılık)	Aykırı değerlere karşı daha dayanıklıdır.	Aykırı değerlere karşı daha hassastır.
Çözüm Hızı	Daha hızlı sonuç verebilir, ancak veri boyutu arttıkça kararsızlaşabilir.	Daha kararlı sonuçlar verir, ancak biraz daha yavaştır.

ElasticNet Regularizasyon:

ElasticNet, L1 ve L2 regularizasyonlarının avantajlarını birleştiren hibrit bir yöntemdir. Hem L1 cezasını (özellik seçimi sağlar) hem de L2 cezasını (katsayıları küçültür) bir arada kullanır. Bu sayede, tek başına L1 veya L2'nin eksik kaldığı durumlarda daha esnek ve dengeli bir regularizasyon sağlar.

$$\hat{\beta} = \arg\min \left\{ \sum_{i=1}^{n}(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij})^2 + \lambda_1 \sum_{j=1}^{p} |\beta_j| + \lambda_2 \sum_{j=1}^{p} \beta_j^2 \right\}$$

Regularizasyon Tekniği	Farklar	Avantajlar	Dezavantajlar
L1 Regularization	Katsayıları sıfıra indirme eğilimindedir. Sezgisel olarak öznitelik seçimi sağlar.	Sparse modeller oluşturur, yani birçok öznitelikten oluşan bir model yerine sadece önemli öznitelikleri içeren bir model oluşturur.	Modelde bazı katsayılar tamamen sıfıra düşer, dolayısıyla özniteliklerin tamamen atılmasına neden olur.
L2 Regularization	Katsayıları küçültme eğilimindedir.	Modelin aşırı uyumu azaltır ve genelleme yeteneğini artırır.	Model karmaşıklığını azaltırken sıfıra yaklaştırır, ancak tam olarak sıfıra indirmez. Bu nedenle öznitelik seçimi konusunda L1 kadar etkili değildir.
ElasticNet	Hem L1 hem de L2 regularizasyonlarını birleştirir.	L1 ve L2'nin avantajlarını birleştirir, bu nedenle daha esnek bir regularizasyon sağlar.	Hem L1 hem de L2'nin dezavantajlarını birleştirir, bu nedenle hesaplama maliyeti daha yüksek olabilir.

Hiperparametre Nedir?

Hiperparametreler, makine öğrenmesi algoritmalarında modelin davranışını, karmaşıklığını ve performansını doğrudan etkileyen ayarlanabilir dış parametrelerdir. Modelin eğitimi sırasında öğrenilmezler; bunun yerine, eğitim öncesinde belirlenirler. Örneğin, öğrenme oranı (learning rate), karar ağacının derinliği veya k-en yakın komşu (k-NN) algoritmasındaki komşu sayısı birer hiperparametredir.

Hiperparametre Ayarlamanın Önemi

Hiperparametrelerin doğru seçimi, bir modelin başarısını önemli ölçüde etkiler. Doğru ayarlama ile:

- **Aşırı uyum (overfitting)** önlenir.
- **Model performansı** artırılır.
- **Modelin stabilitesi** ve genelleme yeteneği korunur.
- **Zaman ve hesaplama kaynakları** daha verimli kullanılır.

Hiperparametre Ayarlama (Tuning) Yöntemleri:

1. **Manuel Arama (Manual Search):** Deneyim ve sezgisel bilgiye dayanarak farklı hiperparametre kombinasyonlarının denenmesidir. Küçük veri kümeleri ve basit modeller için uygundur, ancak büyük modellerde zaman alıcı olabilir.
2. **Kafes Arama (Grid Search):** Belirlenen hiperparametre değerlerinin tüm olası kombinasyonlarını sistematik olarak dener. En iyi sonucu veren kombinasyonu bulmayı amaçlar; ancak yüksek boyutlu veri ve parametre sayısında **hesaplama maliyeti** yüksektir.
3. **Rastgele Arama (Random Search):** Belirli bir aralıktan hiperparametre değerlerini **rastgele** seçerek dener. Grid Search'e göre daha az kombinasyon dener, bu sayede **daha hızlıdır** ve genellikle benzer doğrulukta sonuçlar verir.