* **Makine Öğrenmesi (Machine Learning)**: Verilerden örüntüler veya kurallar öğrenerek tahmin veya karar verme yapabilen algoritmalar bütünüdür.
* **Matris Manipülasyonu**: Matrisler üzerinde yapılan işlemlerdir (toplama, çarpma, transpoz alma, tersini alma gibi). Makine öğrenmesinde veri yapıları ve model hesaplamaları genellikle matrisler aracılığıyla ifade edilir.
* **Özdeğer (Eigenvalue)** ve **Özvektör (Eigenvector)**:
  + Bir matris A için, Av=λv =eşitliğini sağlayan λ skalarına **özdeğer**, v vektörüne ise **özvektör** denir.
  + Özvektör, uygulanan dönüşümde yalnızca ölçeklenen ama yönü değişmeyen vektördür.

**Makine Öğrenmesi ile İlişkisi**

**1. Matris Manipülasyonu**

Makine öğrenmesi algoritmalarının çoğu, özellikle **çok boyutlu veriler** (feature vectors) ile çalışır. Bu veriler genellikle matris biçiminde saklanır ve işlenir.

* **Örnek:**
  + Lojistik regresyonda, tahminler y=σ(XW+b) şeklinde matris işlemleriyle hesaplanır.
  + Sinir ağlarında ileri ve geri yayılım (forward/backward propagation) tamamen matris çarpımları ile yapılır.

**2. Özdeğerler ve Özvektörler**

Özdeğerler ve özvektörler, özellikle **boyut indirgeme** ve **veri sıkıştırma** tekniklerinde önemli rol oynar.

* **Principal Component Analysis (PCA)**:
  + Yüksek boyutlu verilerde en fazla varyansa sahip yönleri bulmak için kullanılır.
  + Adımlar:
    1. Veriyi normalize et.
    2. Kovaryans matrisini hesapla.
    3. Kovaryans matrisinin özdeğerlerini ve özvektörlerini bul.
    4. En büyük özdeğere sahip özvektörler, en anlamlı yönlerdir (principal components).
    5. Veri bu yeni uzaya yansıtılır → boyut indirgenmiş veri elde edilir.
* **Spektral Kümeleme (Spectral Clustering)**:
  + Özellikle doğrusal olmayan kümeleme problemlerinde kullanılır.
  + Adımlar:
    1. Benzerlik grafı oluşturulur.
    2. Laplasyen matrisi oluşturulur.
    3. Laplasyenin özvektörleri alınır → bu vektörler yeni bir uzayda veri temsili sağlar.
* **Yapay Sinir Ağlarının Ağırlık Analizi**:
  + Eğitim sırasında ağırlık matrislerinin özdeğer dağılımı, ağın öğrenme kapasitesini ve genelleme yeteneğini analiz etmekte kullanılır.

**Kullanıldığı Yöntemler ve Yaklaşımlar**

| **Yöntem/Yaklaşım** | **Kullanım Alanı** | **Özdeğer/Özvektör İlişkisi** |
| --- | --- | --- |
| **PCA** | Boyut indirgeme, görselleştirme | Kovaryans matrisinin özdeğer analizi |
| **LDA (Linear Discriminant Analysis)** | Sınıflandırma, boyut indirgeme | Scatter matrislerinin özdeğerleri |
| **Spektral Kümeleme** | Kümeleme (clustering) | Laplasyen matrisinin özvektörleri |
| **Kernel PCA** | Doğrusal olmayan boyut indirgeme | Kernel matrislerinin özdeğerleri |
| **Singular Value Decomposition (SVD)** | Veri sıkıştırma, öneri sistemleri | Özdeğer analiziyle ilişkili |

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**numpy.linalg.eig Fonksiyonu**

**Fonksiyonun Amacı**

numpy.linalg.eig fonksiyonu, bir kare matrisin özdeğerlerini ve sağ özvektörlerini hesaplar. Yani, verilen bir kare matris A için, aşağıdaki denklemi sağlayan λ(özdeğer) ve v(özvektör) çiftlerini bulur:

Av=λvA

**Söz Dizimi**

numpy.linalg.eig(a)

**Parametreler:**

* a: Kare bir matris (boyutları M x M )

**Dönüş Değeri:**

* w: Matrisin özdeğerlerini içeren bir dizi
* v: Matrisin sağ özvektörlerini içeren bir dizi; her sütun, w[i] özdeğerine karşılık gelen özvektördür​

**Notlar:**

* Elde edilen özdeğerler, genellikle karmaşık sayılar olabilir.
* Özvektörler, birim uzunlukta (normalize edilmiş) olacak şekilde döndürülür.​

**🧪 Örnek Kullanım**

import numpy as np

A = np.array([[1, -2],

[1, 3]])

eigenvalues, eigenvectors = np.linalg.eig(A)

print("Özdeğerler:", eigenvalues)

print("Özvektörler:\n", eigenvectors)

**Çıktı:**

Özdeğerler: [2.+1.j 2.-1.j]

Özvektörler:

[[ 0.81649658+0.j 0.81649658-0.j ]

[-0.40824829-0.40824829j -0.40824829+0.40824829j]]

Bu örnekte, matrisin karmaşık özdeğerleri ve bunlara karşılık gelen özvektörleri hesaplanmıştır.​

**Kaynak Kod İncelemesi**

numpy.linalg.eig fonksiyonu, NumPy'nin linalg modülünde tanımlanmıştır.

**Önemli Dosyalar**

* linalg.py: Yüksek seviyeli lineer cebir fonksiyonlarının tanımlandığı dosya.
* lapack\_lite: LAPACK (Linear Algebra PACKage) kütüphanesinin hafif bir versiyonu; düşük seviyeli lineer cebir işlemleri için kullanılır.
* umath\_linalg.cpp: C++ ile yazılmış olup, performans kritik lineer cebir işlemlerini içerir.​

eig fonksiyonu, genellikle LAPACK kütüphanesindeki dgeev (gerçek matrisler için) veya zgeev (karmaşık matrisler için) rutinlerini çağırarak özdeğer ve özvektör hesaplamalarını gerçekleştirir. Bu sayede, yüksek performanslı ve güvenilir sonuçlar elde edilir.​

**Ek Bilgiler**

* numpy.linalg.eig fonksiyonu, genellikle PCA (Principal Component Analysis), LDA (Linear Discriminant Analysis) gibi boyut indirgeme tekniklerinde kullanılır.
* Eğer matrisiniz simetrik veya Hermitian ise, numpy.linalg.eigh fonksiyonunu kullanmanız daha uygundur; bu fonksiyon, simetrik matrisler için optimize edilmiştir.

Kaynaklar: <https://www.veribilimiokulu.com/makine-ogrenmesine-cok-degiskenli-istatistiksel-yaklasimlar-temel-bilesenler-analizi/2/>

<https://tr.d2l.ai/chapter_appendix-mathematics-for-deep-learning/eigendecomposition.html>

<https://chatgpt.com/>

<https://gemini.google.com/>