Makine Öğrenmesi Ders Notları
Hazırlayan: Barış Baykant Alagöz
İnönü Üniversitesi, Bilgisayar Mühendisliği, 2023-2024 Bahar Dönemi Lisans Dersi
Sorularınız ve düzeltme önerileriniz için e-mail: baykant.alagoz@inonu.edu.tr

Öğrenme, Optimizasyon ve İhtiyaç Duyulan Matematiksel Temeller I:

Öğrenme faaliyeti bir önceki bölümde tartışıldığı üzere, aslında "tecrübeler ile işlevsel performansın iyileştirilmesi faaliyeti" olarak tanımlanabilir. Performans iyileştirme işlemi matematikte optimizasyon işlemine karşılık gelir. Mitchell'ın öğrenme tarifine dönülürse, (Tom Mitchell. Machine Learning 1997.)

"Bir makine, bir *T* görev kümesinden elde ettiği *E* tecrübe kümesi sonucunda bir *P* performans ölçütüne göre performansını iyileştirebiliyorsa makine öğrenebiliyor."

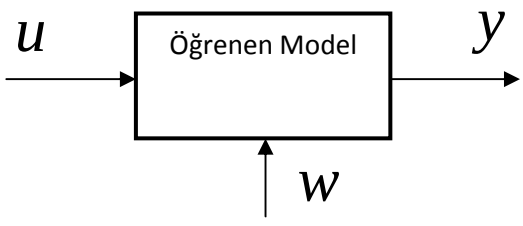
Burada bir *P* performans ölçütüne göre performansını iyileştirilmesi süreci matematiksel olarak bir optimizasyon sürecine karşılık gelir. Bu bölümden optimizasyon öğrenme ilişkisi, performans ölçütlerinin geri planı ve optimizasyona giriş yapılmaktadır.

Bir sonraki bölümde denetimli öğrenme tekniğinin matematiksel geri plan ve öğrenme mimarisi incelenecektir.

Danışmanlı Makine Öğrenmesi Tekniğinin Temel Matematiksel Altyapısı:

Önceki bölümde makine öğrenmesinin bileşenleri incelenmişti. Bu bileşenler (i) eğitim verisi,(ii) makine öğrenmesi algoritması ve (iii) Öğrenen model olarak özetlemişti.

Öğrenen model cevabı parametrik olarak değişebilen ve ayarlanabilen bir sistemin matematiksel modeldir. Doğada öğrenme kabiliyeti için elastikiyet gereklidir. Öğrenen modeller elastikiyet (değişkenlik) gösterebilen fonksiyonlardan seçilmelidir. Makine öğrenmesi açısından elastikiyet, sistem cevabının ayarlanabilir parametreler yardımı ile değişebilme kabiliyeti olarak ifade edilebilir. Algoritma tarafından öğrenme tamamlandığında, ayarlanabilen parametreler ile model cevabının tanımlanmış bir P performans ölçütüne göre istenen düzeye ulaşması sağlanır. Aşağıda öğrenen model blok diyagram olarak gösterilmiştir. u model girişlerini, y model çıkışlarını temsil eder. Burada w öğrenen sistemin elastikiyet parametreleri yani ayarlanabilen parametreleridir. Öğrenen modelin girişler için çıktısı (giriş-çıkış ilişkisi) bu parametreye bağlı şekillenir. Bu öğrenme kabiliyeti için gerekli olan elastikiyeti sağlar.



- Öğrenme hatası (Residual – artık): Öğrenme işleminin performansını (başarımını) ifade etmek için kullanılan bir ölçüttür.

$$e = d - y$$

olarak ifade edilir. Burada $y = f(u, w)$, sistemin u girdisi ve w ayarlanabilen katsayıları için çıktısıdır. d ise u giriş için arzu edilen (istenen) sistem çıktısıdır. Hatanın sıfır olması bir u giriş için öğrenen modelin d çıkışını vermeyi hatasız olarak öğrenmesi durumunu ifade eder.

$$e = d - y = d - f(u, w) = 0 \Rightarrow f(u, w) = d$$

Sıfır öğrenme hatası durumu, öğrenme işleminin bu veri için hatasız başarıldığı durumu ifade eder.

Öğrenilecek veri(bilgi, girdi) miktarının çok sayıda olduğu durumu inceleyelim. Varsayalım n girdi için istenen çıkışlar verilmiş olsun.

Girdiler: $u_1,\ u_2, .., u_n$ ve

İstenen çıktılar: $d_1, d_2, .., d_n$

Bu girdiler ile eğitim kümesi oluşturalım: $T=\{(u_1, d_1), (u_2, d_2), .., (u_n, d_n)\}$ (Bu veri kümesi bir etiketli veri kümesidir çünkü eğitim kümesinde giriş için istenen çıkış verilmiştir)

Bu eğitim kümesindeki bütün veriler için öğrenme hataları yazalım.

Öğrenme hatası:

$e_1 = d_1 - y_1 = d_1 - f(u_1, w)$,

$e_2 = d_2 - y_2 = d_2 - f(u_2, w)$,

..,

$e_n = d_n - y_n = d_n - f(u_n, w)$

Karesel Hatası ile Öğrenme:

Öğrenme performansı P yi karesel hata ölçütü ile ölçelim. T eğitim kümesinin bütün elemanları için karesel hata toplamı

$$E = \frac{1}{2}\sum_{i=1}^{n} e_1^{\,2} = \frac{1}{2}e_1^{\,2} + \frac{1}{2}e_2^{\,2} + .. + \frac{1}{2}e_n^{\,2}$$

Karesel hata fonksiyonu bir konveks ve semi pozitif definite bir fonksiyondur. Alabileceği en küçük değer sıfırdır. T eğitim kümesindeki bütün verilerin hatasız öğrenilmesi için

$$E = \frac{1}{2}e_1^{\,2} + \frac{1}{2}e_2^{\,2} + .. + \frac{1}{2}e_n^{\,2} = 0$$

olmalıdır. Bu durum hata kareleri nedeni ile ancak,

$e_1 = d_1 - y_1 = 0$, $e_2 = d_2 - y_2 = 0$,.., $e_n = d_n - y_n = 0$

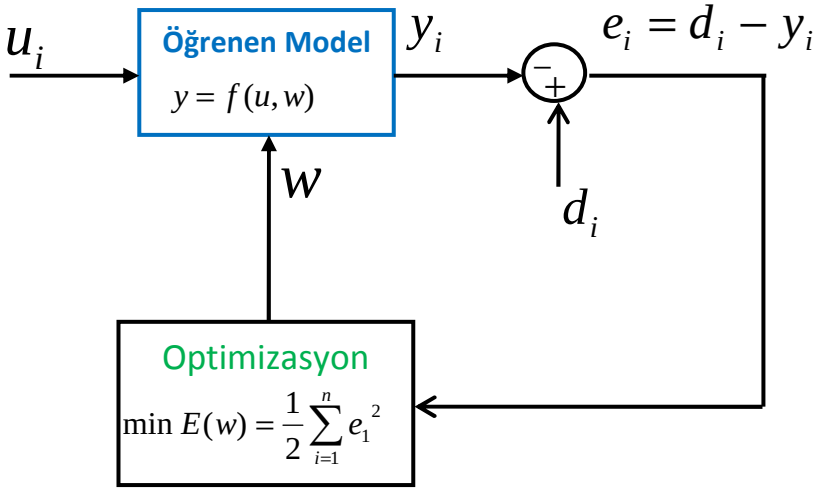
durumunda bu mümkündür.

Bu durum $y_1 = d_1$, $y_2 = d_2$,.., $y_n = d_n$ olması anlamına gelir ve öğrenme modeli, T eğitim kümesini hatasız öğrenmiş olur.

Eğer öğrenen model en genel formda $y(u) = f(u, w)$ formunda bir elastik fonksiyon ile ifade edilirse, öğrenme probleminin çözümünü sağlayan optimizasyon problemi matematiksel olarak şöyle ifade edilir.

$$\min E(w) = \frac{1}{2}\sum_{i=1}^{n} e_1^{\,2}$$

Danışmanlı Makine Öğrenme temel mimarisi aşağıdaki şekilde gösterilebilir.



Makine Öğrenmesi Dersi İçin İhtiyaç Duyulan Matematiksel Temeller:

Lineer Cebir ve Vektörleştirme:

$m \times n$ rakamdan oluşan iki boyutlu rakam topluları matrisler ile temsil edilir.

$M = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$ burada 2x3 boyutlu M matrisi görülmektedir.

Matris elemanları $a_{i,j}$ ile gösterilir. Burada i satır indisi ve j sütun indisidir.

$M = \begin{bmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{bmatrix}$ matrisinde $a_{1,1}$, $a_{1,2}$ birinci satır elemanları, $a_{2,1}$, $a_{2,2}$ ikinci satır elemanlarıdır.

$M = \begin{bmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{bmatrix}$ matrisinde $a_{1,1}$, $a_{2,1}$ birinci sütun elemanları, $a_{1,2}$, $a_{2,2}$ ikinci sütun elemanlarıdır.

$M = \begin{bmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{bmatrix}$ matrisinde $a_{1,1}$, $a_{2,2}$ diyagonal elemanlarıdır.

$M = \begin{bmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{bmatrix}$ matrisinde $a_{1,1}$, $a_{2,2}$ ters diyagonal elemanlarıdır.

Bir boyutlu rakam topluluğu vektörler ile temsil edilir.

$V = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$ => 2x1 boyutlu bir sütun vektörü

$S = \begin{bmatrix} 1 & 2 \end{bmatrix}$ => 1x2 boyutlu bir satır vektörü

Transpoze işlemi matriste satır ve sütunların karşılıklı yer değiştirmesidir. Transpoze işleminde

$\begin{bmatrix} a_{i,j} \end{bmatrix}^T = \begin{bmatrix} a_{j,i} \end{bmatrix}$ ile ifade edilir.

Transpoze işlemi satır vektörünü sütun vektörüne ve sütun vektörünü satır vektörüne dönüştürür. Transpoze işleminde $m \times n$ boyutlu bir matris $n \times m$ boyuta dönüşür. Bu nedenle boyut dönüştürmede kullanılabilir.

$V = \begin{bmatrix} 1 \\ 2 \end{bmatrix} => V^T = \begin{bmatrix} 1 & 2 \end{bmatrix}$

$M = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} => M^T = \begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix}$

```
% Matlabda 1x3 boyutlu vektör tanımlama.
V=[ 1 2 3];
% Matlabda 2x3 boyutlu matris tanımlama.
X=[ 1 2 3; 4 5 6];
% V' vektörünün transpozesi
W=V';
% X' vektörünün transpozesi
Y=X';
```

```
# Phytonda matris işlemleri numpy kütüphanesi yardımı
ile yapılabilir.
import numpy as np
# Phyton'da 1x3 satır vektörü tanımlayalım
V= np.matrix([[1, 2, 3]])
# Matlabda 2x3 boyutlu matris tanımlayalım
X= np.matrix([[1, 2, 3], [4, 5, 6]])
# V' vektörünün transpozesi
V_tr=np.transpose(V)
# X' vektörünün transpozesi
X_tr=np.transpose(X)
print('V= \n',V);print('X= \n',X);print('V_tr=
\n',V_tr);print('X_tr= \n',X_tr)
```

Transpoze işleminin bazı özellikleri:

A ve B birer matris ve c bir skaler olmak üzere

1. $(A \pm B)^T = A^T \pm B^T$

2. $(cA)^T = cA^T$

3. $(A.B)^T = B^T A^T$

4. Eğer A matrisi simetrik matris ise (Elemanları diyagonale göre simetrik olması durumu)

$A = A^T$

Transpoze işleminin bir diğer önemli kullanımı vektör yada matrisleri kareselleştirme işlemidir. Matris tersi işlemi kare matrisler (Satır ve sütun sayıları eşit matrisler) için tanımlıdır. Denklem çözümlerinde ters alma işlemi boyut uyumsuzluğu nedeni yapılamıyor ise için matrisler transpozezi ile çarpılarak kareselleştirilirler.

Herhangi bir A matris ve vektörü için AA^T matrisi kare matristir.

$V = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$ vektörünü kaseselleştiriniz.

$$VV^T = \begin{bmatrix} 1 \\ 2 \end{bmatrix}_{2x1} \begin{bmatrix} 1 & 2 \end{bmatrix}_{1x2} = \begin{bmatrix} 1 & 2 \\ 2 & 4 \end{bmatrix}_{2x2}$$

$M = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$ matirisi kare matrise dönüştürünüz.

$$MM^T = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}_{2x3} \begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix}_{3x2} = \begin{bmatrix} 14 & 32 \\ 32 & 77 \end{bmatrix}_{2x2}$$

```
% Matlabda 2x3 boyutlu matris tanımlama.
M=[ 1 2 3; 4 5 6];
% X vektörü kareselleştirilir
Y=M*M';
```

```
# Phyton'da matris kareselleştirmesi deneyelim
# Matlabda 2x3 boyutlu matris tanımlamayalım. Bu matris
kare matris değildir.
M= np.matrix([[1, 2, 3], [4, 5, 6]])
# Kareseleştirme için matrisi transpozesi ile çarpalım :
M*M_tr
W=M*np.transpose(M)
print('M= \n',M);print('Kare matris WM*M_tr= \n',W)
```

Matris Tersi:

Eğer $n \times n$ boyutlu bir A kare matrisi aşağıdaki koşulu sağlıyorsa tersinirdir.

$$AA^{-1} = I$$

Burada A^{-1} matrisi A matrisinin tersidir. Bir kare matrisin tersinin olabilmesi için determinantı sıfırdan farklı olmalıdır.

İki boyutlu matris tersi için pratik bir yol:

$$A = \begin{bmatrix} a & b \\ c & d \end{bmatrix} => A^{-1} = \frac{1}{\det(A)} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix} = \frac{1}{ad-bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$

3x3 matrisin tersinin bulunması için şu andımlar izlenir.

Örnek: $A = \begin{bmatrix} 1 & 2 \\ 1 & 4 \end{bmatrix}$ için A^{-1} hesaplayınız. $AA^{-1} = I$ olduğunu göstererek işlemin doğruluğunu sınayınız.

$$A^{-1} == \frac{1}{1.4-1.2} \begin{bmatrix} 4 & -2 \\ -1 & 1 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 4 & -2 \\ -1 & 1 \end{bmatrix} = \begin{bmatrix} 2 & -1 \\ -0.5 & 0.5 \end{bmatrix}$$

$AA^{-1} = I$ olduğunu görelim ve elde edilen sonucu doğrulayalım.

$$\begin{bmatrix} 1 & 2 \\ 1 & 4 \end{bmatrix} \cdot \begin{bmatrix} 2 & -1 \\ -0.5 & 0.5 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = I$$ birim matris olduğu için işlem doğrudur.

Daha büyük boyutlu matrislerin tersi şu formüller alabilir.

$$M^{-1} = \frac{1}{\det(M)} Adj(M)$$

Örnek: $M = \begin{bmatrix} 1 & 2 & 3 \\ 0 & 1 & 4 \\ 5 & 6 & 0 \end{bmatrix}$ 3x3 boyutlu matrisin tersini hesaplayalım.

Adım 1- Önce matrisin determinantı hesaplanır.

$$\begin{matrix} & + & - & + \\ M = & \begin{bmatrix} 1 & 2 & 3 \\ 0 & 1 & 4 \\ 5 & 6 & 0 \end{bmatrix} \end{matrix}$$

(Sadece birinci satır elemanları bir + bir –işareti ile çarpan olup, bu elemanın satır ve sütünü kapatıldığında oluşan alt matrisin determinantı yazılır)

$$\det(M) = 1\begin{vmatrix} 1 & 4 \\ 6 & 0 \end{vmatrix} - 2\begin{vmatrix} 0 & 4 \\ 5 & 0 \end{vmatrix} + 3\begin{vmatrix} 0 & 1 \\ 5 & 6 \end{vmatrix} = 1(1.0 - 6.4) - 2.(0.0 - 5.4) + 3(0.6 - 5.1) = 1$$

Adım 2- Matrisin transpozesi alınıp minörleri hesaplanır.

$$M^T = \begin{bmatrix} 1 & 0 & 5 \\ 2 & 1 & 6 \\ 3 & 4 & 0 \end{bmatrix}$$

(Her eleman satır ve sütün kapatılır. Kalan elemanlar ile oluşan altmatrisin determinantı hesaplanır.)

$$\begin{vmatrix} 1 & 6 \\ 4 & 0 \end{vmatrix} = -24, \quad \begin{vmatrix} 2 & 6 \\ 3 & 0 \end{vmatrix} = -18, \quad \begin{vmatrix} 2 & 1 \\ 3 & 4 \end{vmatrix} = 5, \quad \begin{vmatrix} 0 & 5 \\ 4 & 0 \end{vmatrix} = -20, \quad \begin{vmatrix} 1 & 5 \\ 3 & 0 \end{vmatrix} = -15, \quad \begin{vmatrix} 1 & 0 \\ 3 & 4 \end{vmatrix} = 4, \quad \begin{vmatrix} 0 & 5 \\ 1 & 6 \end{vmatrix} = -5,$$

$$\begin{vmatrix} 1 & 5 \\ 2 & 6 \end{vmatrix} = -4, \begin{vmatrix} 1 & 0 \\ 2 & 1 \end{vmatrix} = 1,$$

Adım 3- Kofaktör matrisi oluşturulur. Her elemen minörleri ile değiştirilir ve işaret değişimi (bir + ve bir – olacak şeklinde) uygulanır.

$$Adj(M)=\begin{bmatrix}-24 & -18 & 5\\ -20 & -15 & 4\\ -5 & -4 & 0\end{bmatrix}\leftrightarrow\begin{bmatrix}+ & - & +\\ - & + & -\\ + & - & +\end{bmatrix}=\begin{bmatrix}-24 & 18 & 5\\ 20 & -15 & -4\\ -5 & 4 & 1\end{bmatrix}$$

Adım 4. Aşağıdaki formüller hesapla

$$M^{-1}=\frac{1}{\det(M)}Adj(M)=\frac{1}{1}\begin{bmatrix}-24 & 18 & 5\\ 20 & -15 & -4\\ -5 & 4 & 1\end{bmatrix}=\begin{bmatrix}-24 & 18 & 5\\ 20 & -15 & -4\\ -5 & 4 & 1\end{bmatrix}$$

Diğer kolay bir yol Gauss Eliminasyon yöntemi ile matris tersi alınması. Gauss eliminasyon yöntemi bir denklem sistemlerinin nümerik olarak çözümünü sağlar. Ters alam işlemi $AA^{-1}=I$ denklem sisteminin çözümüne karşılık gelir ve Gauss eliminasyon uygulanabilir.

Satır indirgeme işlemleri: Denklemi ifade edilen bir matris üzerinde satırların toplanabilmesi, satırın bir sabitle ile çarpılabilmesi ve satırların yer değiştirebilmesi özelliklerini kullanarak matrisin denklem çözümü ifade edecek şekilde indirgemesini sağlar. Bu işlemler (satırların toplanabilmesi, satırın bir sabitle ile çarpılabilmesi ve satırların yer değiştirebilmesi) denklem sistemini bozmaz.

Nümerik ters alma işlemi bilgisayar hesaplaması için daha kolaydır. Tersi alınacak M matrisi $[M\,|\,I]$ formunda birleştiriyor. Sonra eliminasyon uygulanarak $[I\,|\,M^{-1}]$ formuna getirilir ve M^{-1} elde edilir.

$$\overset{\quad M\qquad\quad I}{\left[\begin{array}{ccc|ccc}1 & 2 & 3 & 1 & 0 & 0\\ 0 & 1 & 4 & 0 & 1 & 0\\ 5 & 6 & 0 & 0 & 0 & 1\end{array}\right]}\Rightarrow\left[\begin{array}{ccc|ccc}1 & 0 & -5 & 1 & -2 & 0\\ 0 & 1 & 4 & 0 & 1 & 0\\ 5 & 6 & 0 & 0 & 0 & 1\end{array}\right]\Rightarrow\left[\begin{array}{ccc|ccc}1 & 0 & -5 & 1 & -2 & 0\\ 0 & 1 & 4 & 0 & 1 & 0\\ 0 & 6 & 25 & -5 & 10 & 1\end{array}\right]\Rightarrow$$

$$\left[\begin{array}{ccc|ccc}1 & 0 & -5 & 1 & -2 & 0\\ 0 & 1 & 4 & 0 & 1 & 0\\ 0 & 0 & 1 & -5 & 4 & 1\end{array}\right]\Rightarrow\left[\begin{array}{ccc|ccc}1 & 0 & -5 & 1 & -2 & 0\\ 0 & 1 & 0 & 20 & -15 & -4\\ 0 & 0 & 1 & -5 & 4 & 1\end{array}\right]\Rightarrow\left[\begin{array}{ccc|ccc}1 & 0 & 0 & -24 & 18 & 5\\ 0 & 1 & 0 & 20 & -15 & -4\\ 0 & 0 & 1 & -5 & 4 & 1\end{array}\right]$$

Satır indirgemeler sonunda $\overset{\quad I\qquad\quad M^{-1}}{\left[\begin{array}{ccc|ccc}1 & 0 & 0 & -24 & 18 & 5\\ 0 & 1 & 0 & 20 & -15 & -4\\ 0 & 0 & 1 & -5 & 4 & 1\end{array}\right]}\Rightarrow M^{-1}=\begin{bmatrix}-24 & 18 & 5\\ 20 & -15 & -4\\ -5 & 4 & 1\end{bmatrix}$ elde edilir.

```
% Matlabda 2x3 boyutlu matris tanımlama.
M=[ 1 2 3; 0 1 4;5 6 0];
% Matris tersi
Y=M^-1;
% Determinentı
Y=det(M);
```

```
# Phyton'da matris tersi
# Matlabda 2x3 boyutlu matris tanımlama.
M= np.matrix([[1, 2, 3], [0, 1, 4], [5, 6, 0]])
# tersi işlemi
M_inv=np.linalg.inv(M)
# determinant
M_det=np.linalg.det(M)
print('M= \n',M);print('M_inv= \n',M_inv);print('M_det=
\n',M_det)
```

Lineer Denklem Sistemi Çözümü:

Lineer denklem sistemleri matris formunda ifade edilebilir ve lineer cebir yardımı ile çözülebilir. Örneğin, aşağıda 2 bilinmeyenli 2 denklemden oluşan bir denklem sistemi verilsin.

$m_1 x_1 + c_1 x_2 = y_1$

$m_2 x_1 + c_2 x_2 = y_2$

Denklemler matris formunda yazılırsa,

$$\begin{bmatrix} m_1 & c_1 \\ m_2 & c_2 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} \Rightarrow A = \begin{bmatrix} m_1 & c_1 \\ m_2 & c_2 \end{bmatrix}, X = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, b = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}$$

denklem sistemi matris formunda $A.X = b$ elde edilir.

Burada A matrisi her zaman kare matris çıkar ve tersi alınabilir. Eşitliğin her iki tarafı A^{-1} ile çarpılırsa,

$A^{-1}.A.X = A^{-1}.b$ (burada $A^{-1}.A = I$ olduğunu hatırlayınız)

Çözüm $X = A^{-1}.b$ olur.

Örnek. $x_1 + x_2 = 2$, $2x_1 - x_2 = 1$ denklim sistemini $X = A^{-1}.b$ ilw çözelim.

$$\begin{bmatrix} 1 & 1 \\ 2 & -1 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 2 \\ 1 \end{bmatrix} \Rightarrow A = \begin{bmatrix} 1 & 1 \\ 2 & -1 \end{bmatrix} \Rightarrow$$

$$A^{-1} == \frac{1}{1.(-1) - 1.2}\begin{bmatrix} -1 & -1 \\ -2 & 1 \end{bmatrix} = \frac{1}{-3}\begin{bmatrix} -1 & -1 \\ -2 & 1 \end{bmatrix} = \begin{bmatrix} 1/3 & 1/3 \\ 2/3 & -1/3 \end{bmatrix}$$

$X = \begin{bmatrix} 1/3 & 1/3 \\ 2/3 & -1/3 \end{bmatrix}.\begin{bmatrix} 2 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$ elde edilir. $x_1 = 1$ ve $x_2 = 1$

```
# Lineer denklem sistemini matris işlemleri ile çözeli
# Örnek: 2 bilinmeyenli 2 denklemden oluşan aşağıdaki
lineer denklem sistemini çözünüz.
# 2*x3-x2+x1=4
# x3+x2+2*x1=5
# x3+2*x2-x1=3
# Yukarıdaki denklemin çözümü x1=1, x2=1 ve x3=2
# Denklem sistemini matris formunda ifade edelim ve
pytonda çözelim.
# A=[[2, -1, 1,],[1, 1, 2],[1, 2, -1,]], b=[[4],[5],[3]]
# Çözüm:A*x=b => x=A_inv*b

A=[[2, -1, 1,],[1, 1, 2],[1, 2, -1,]]
b=np.transpose([[4, 5, 3]])
#print('x= \n', np.linalg.inv(A))
x=np.matmul(np.linalg.inv(A),b)
print('x= \n', x)
print('x3= \n',x[0]); print('x2= \n',x[1]); print('x3=
\n',x[2]);
```