Session #07

CMSC 409:
Artificial Intelligence

http://

Virginia Commonwealth University,
Fall 2023,
Dr. Milos Manic
(mmanic@vcu. edu)

1

CMSC 409:
Artificial Intelligence

Session # 07

Topics for today

- Announcements
- Previous session review
- Review
  - *Analytic Geometry in Euclidean Space with Cartesian Coordinates*
- Learning of agents
  - *Transfer function*
  - *Hamming distance in pattern matching*
- Unsupervised vs. supervised learning
  - *Representative learning rules (Hebb, Correl.)*

2

CMSC 409: Artificial Intelligence

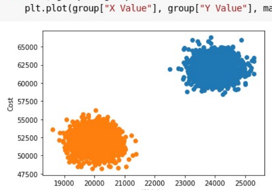
Announcements Session # 07

- Canvas
  - *New slides posted*
  - *A supplementary files posted (starts with Session_7_Ref_Slides...)*
- Office hours zoom
  - *Zoom disconnects me after 45 mins of inactivity. Feel free to chat me via zoom if that happens and I will reconnect (zoom chat welcome outside of office hours as well)!*
- Project #1
  - *Deadline Sep. 14 (noon)*
- Project #2
  - *Deadline Oct. 3 (noon)*
- Paper (optional)
  - *First draft - due Sep. 12 (noon)*
  - *Think about the topic of your paper and confirm on 1st draft deliverables (class paper instructions)*
- Subject line and signature
  - *Please use [CMSC 409] Last_Name Question*

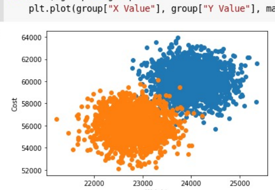
3

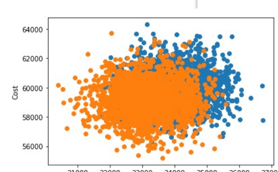
Project 1 data sets

```
import matplotlib.pyplot as plt
import numpy as np
from scipy.stats import kde
import pandas as pd

DATA_LOCATION = "AGroup.csv"
FA = "X"
SA = "Y"
TA = "Z"
datas = pd.read_csv(DATA_LOCATION)
x = datas[FA]
y = datas[SA]
z = datas[TA]
plt.xlabel('Weight')
plt.ylabel('Cost')
data = pd.DataFrame({"X Value": x, "Y Value": y, "Category": z})
groups = data.groupby("Category")

for name, group in groups:
    plt.plot(group["X Value"], group["Y Value"], marker="o", linestyle="", label=name)
```




```
import matplotlib.pyplot as plt
import numpy as np
from scipy.stats import kde
import pandas as pd

DATA_LOCATION = "BGroup.csv"
FA = "X"
SA = "Y"
TA = "Z"
datas = pd.read_csv(DATA_LOCATION)
x = datas[FA]
y = datas[SA]
z = datas[TA]
plt.xlabel('Weight')
plt.ylabel('Cost')
data = pd.DataFrame({"X Value": x, "Y Value": y, "Category": z})
groups = data.groupby("Category")

for name, group in groups:
    plt.plot(group["X Value"], group["Y Value"], marker="o", linestyle="", la
```





4

Linear Algebra Overview

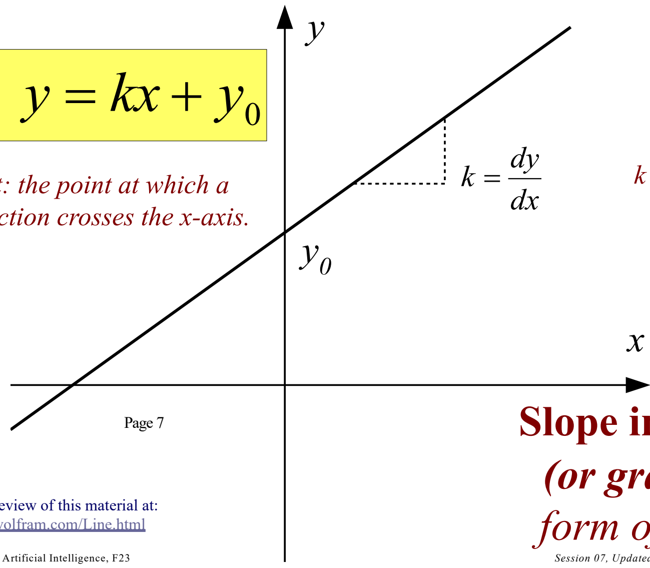
Review - Analytic Geometry in Euclidean Space with Cartesian Coordinates

- *slope intercept, line intercept, scalar form of a line, 2 & 3 point form, plane with nonzero normal vector one point and vector form*

Analytic Geometry in Euclidean Space with Cartesian Coordinates - LINE

$$y = kx + y_0$$

x-intercept: the point at which a curve or function crosses the x-axis.



k - slope

Page 7

Slope intercept *(or gradient)* *form of a line*

You can find nice review of this material at:
http://mathworld.wolfram.com/Line.html

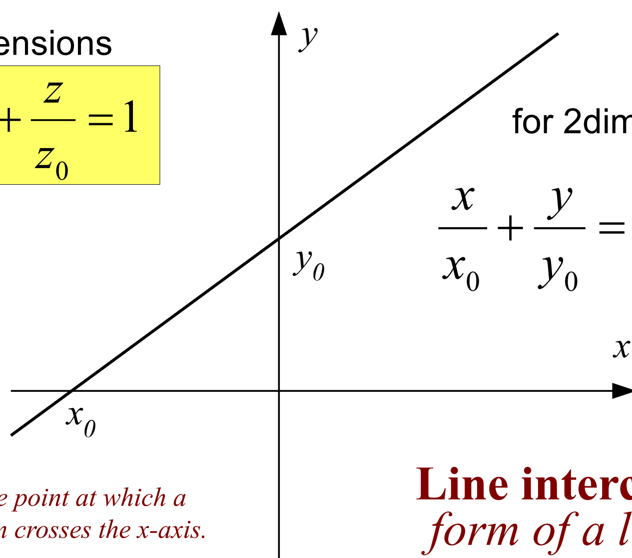
Session 07, Updated on 9/14/23 9:15:46 AM

7

Analytic Geometry in Euclidean Space with Cartesian Coordinates - LINE

for 3dimensions

$$\frac{x}{x_0} + \frac{y}{y_0} + \frac{z}{z_0} = 1$$



for 2dim.

$$\frac{x}{x_0} + \frac{y}{y_0} = 1$$

x-intercept: the point at which a curve or function crosses the x-axis.

Line intercept *form of a line*

Page 8

Session 07, Updated on 9/14/23 9:15:46 AM

8

Analytic Geometry in Euclidean Space with Cartesian Coordinates - LINE

General (scalar) form of a line:

$$ax + by + c = 0$$

…can be also written in vector form as: $\mathbf{n}^t \begin{bmatrix} x \\ y \end{bmatrix} + c = 0$

where: $\mathbf{n} = \begin{bmatrix} a \\ b \end{bmatrix}$ or: $\begin{bmatrix} a & b \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix} + c = 0$

Analytic Geometry in Euclidean Space with Cartesian Coordinates - LINE

two point form: $\dfrac{y - y_1}{x - x_1} = \dfrac{y_2 - y_1}{x_2 - x_1}$

…or in determinant form: $\begin{vmatrix} x & y & 1 \\ x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \end{vmatrix} = 0$

three point form for 3 dimensions (determinant form): $\begin{vmatrix} x & y & z & 1 \\ x_1 & y_1 & z_1 & 1 \\ x_2 & y_2 & z_2 & 1 \\ x_3 & y_3 & z_3 & 1 \end{vmatrix} = 0$

(Cramer's rules…)

(if you know your line, you know your weights….)

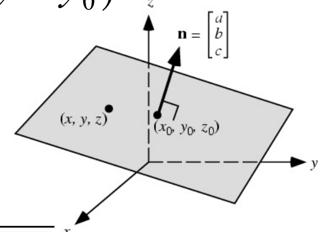
Analytic Geometry in Euclidean Space with Cartesian Coordinates

One point and vector form: $\boxed{\mathbf{n}^t(\mathbf{x}-\mathbf{x}_0)=0}$ **1st form**

… or for 2 dimensions: $n_x(x-x_0)+n_y(y-y_0)=0$

where:

n – normal vector, x_0 - point



plane if $x_0= (x_0, y_0, z_0)$

Picture taken from: http://mathworld.wolfram.com/Plane.html

Normal vector can be normalized:

$$\mathbf{r}=\frac{\mathbf{n}}{\|\mathbf{n}\|}=\frac{\mathbf{n}}{\sqrt{n_1^2+n_2^2+\cdots+n_n^2}}$$

Which gives:

$$\mathbf{r}^t(\mathbf{x}-\mathbf{x}_0)=0$$

where:

x_0 – point;

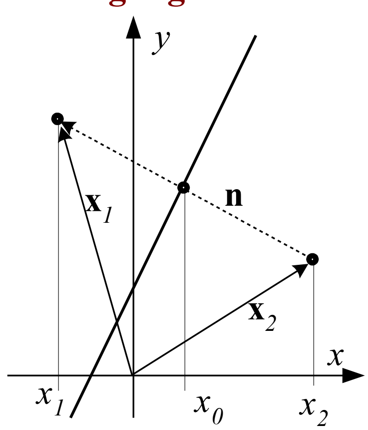
$\|n\|$ - norm of the vector

The equation of a plane with nonzero normal vector n=(a,b,c) through the point x0=(x0,y0,z0) is n·(x-x0)=0.

11

Analytic Geometry in Euclidean Space with Cartesian Coordinates

Bisecting segment between $\mathbf{x}_1$ and $\mathbf{x}_2$: $\mathbf{n}^t(\mathbf{x}-\mathbf{x}_0)=0$



where: $\mathbf{x}_0=\dfrac{\mathbf{x}_1+\mathbf{x}_2}{2}$

and: $\mathbf{n}=\mathbf{x}_1-\mathbf{x}_2$

where:

n^t – directional vector;

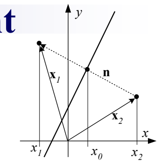
x_0 – average pattern for 2 patterns

which finally gives:

$$\boxed{(\mathbf{x}_1-\mathbf{x}_2)^t\left(\mathbf{x}-\frac{\mathbf{x}_1+\mathbf{x}_2}{2}\right)=0}$$

12

Analytic Geometry – Bisecting Segment



Bisecting segment between $\mathbf{x}_1$ and $\mathbf{x}_2$:

(our old example with 2 patterns, other way to solve)

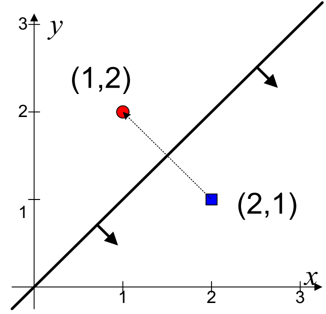
$$\boxed{\mathbf{n}^t\left(\mathbf{x}-\mathbf{x}_0\right)=0}$$ 1st form *(using one point & vector form)*

$$\mathbf{x}_0=\frac{\mathbf{x}_1+\mathbf{x}_2}{2}=\begin{bmatrix}+1.5\\+1.5\end{bmatrix} \qquad \mathbf{n}=\mathbf{x}_1-\mathbf{x}_2=\begin{bmatrix}-1\\+1\end{bmatrix}$$

$$\begin{bmatrix}-1 & +1\end{bmatrix}*\left(\begin{bmatrix}x\\y\end{bmatrix}-\begin{bmatrix}1.5\\1.5\end{bmatrix}\right)=0$$

$$\begin{bmatrix}-1 & +1\end{bmatrix}*\left(\begin{bmatrix}x-1.5\\y-1.5\end{bmatrix}\right)=0$$

$$-\left(x-1.5\right)+\left(y-1.5\right)=0$$



$$\boxed{-x+y=0}$$

*Note: above is in **vector** form.*

13

(Refresher)

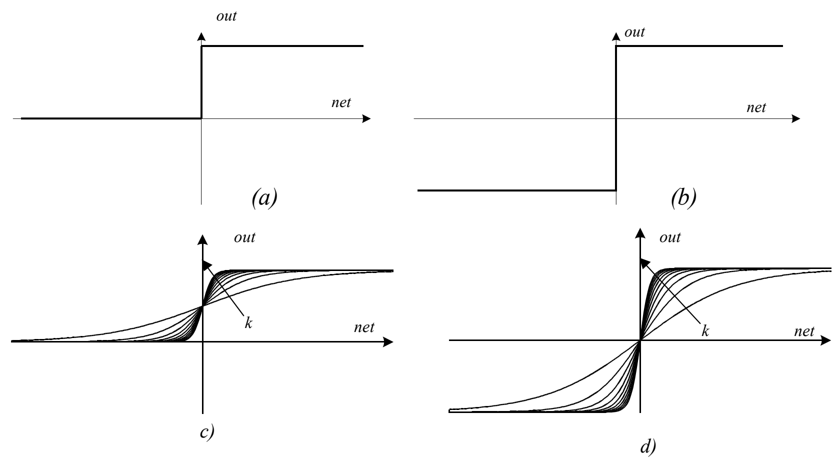
Learning of Agents

- transfer function

14

Learning Agents – transfer function

(for now, note the shape of the function only)



$$o = f(knet) = \frac{1}{1 + \exp(-knet)}$$

$$o = f(knet) = \tanh(knet) = \frac{2}{1 + \exp(-2knet)} - 1$$

15

Hamming Distance

16

Hamming Distance in Pattern Matching

For the set of binary codewords (code):

***The minimum number of bit positions in which two bit words differ is the* Hamming Distance.**

- **Used in *encoding theory, error correction & compression techniques.***
- **Error detection rate *n* (HD=t+1)**
- **Error correction rate *n* (HD=2t+1)**

- **Examples:**

0 0 0 0 0	**1 1 1 0 0**	**0 0 0 1**
0 0 1 1 1	**1 1 0 1 1**	**1 0 1 1**
$(HD = \ ?)$	$(HD = \ ?)$	$(HD = \ ?)$

Hamming Distance in Pattern Matching

Let us consider binary signals and weights such as

$\mathbf{x}$ = +1 -1 -1 +1 -1 +1 +1 -1

if weights $w_i = x_i$

$\mathbf{w}$ = +1 -1 -1 +1 -1 +1 +1 -1

then

$$net = \sum_{i=1}^{n} w_i x_i = 8 \qquad (HD = 0)$$

This is the maximum value *net* can have. For any other combinations *net* would be smaller.

Hamming Distance in Pattern Matching

For the same pattern

x = +1 -1 -1 +1 -1 +1 +1 -1

and slightly different weights $(HD = ?)$

w = +1 +1 -1 +1 -1 -1 +1 -1

$$net = \sum_{i=1}^{n} w_i x_i = 4$$

$$net = \sum_{i=1}^{n} w_i x_i = n - 2HD$$

HD *is the* ***Hamming Distance***

Single Neuron Operation

- ***Unsupervised vs. supervised learning***
- *Hebbian rule*
- *Perceptron rule*

Two types of learning..

- **Supervised**
  - Teacher provides the desired response (target patterns).
  - Teacher defines the distance between the actual o and desired d response and that distance is an error measure used for correction network weight matrix W.
- **Unsupervised**
  - Teacher embeds only the rules - desired responses are not known
  - Learning is based on observations of responses to input sequences.
  - Network itself discovers any existing regularities or properties that lead to changes in weight matrix *W*.

Single Neuron Operation

- *Unsupervised vs. supervised learning*
- ***Hebbian rule***
- ***Correlation rule***

Typical learning formulas

- Typically, weights are updated *(vector form)*:

$$\Delta \mathbf{w}_i = \alpha\, \delta\, \mathbf{x}$$

where new weight space is updated as:

$$\mathbf{w}_{i+1} = \Delta \mathbf{w}_i + \mathbf{w}_i$$

- Hebbian learning rule (unsupervised rule)
  - Requires the weight initialization at small random values prior the learning.
  - Purely feedforward, unsupervised learning.
  - Frequent input patterns have the most influence on the neuron's *W* vector and eventually produce the largest output.

Hebbian learning rule (unsupervised rule)

Hebbian learning rule

- Learning signal is simply neuron's output:

$$\delta = o$$

and weight space updated like:

$$\mathbf{w}_{i+1} = \Delta \mathbf{w}_i + \mathbf{w}_i$$

where α is a learning constant and $\Delta \mathbf{w}_i$ is:

$$\Delta \mathbf{w}_i = \alpha\, \delta\, \mathbf{x}$$

which finally results in following expression for increment:

$$\boxed{\Delta \mathbf{w}_i = \alpha\, o\, \mathbf{x}}$$

Hebb rule

What if we had desired output only?

$$\boxed{\delta = d}$$

Correlation Rule
(supervised)

Things to remember…

- **Separation line…**
  - *Is not making a decision yet, but it indicates possible weights of a neuron*
  - *Will be "making a decision" when turned into inequality (then it "selects" some space)*

- **Using Euclidean geometry**
  - *We can "design" a neuron, i.e. decide on the weights w/o running any algorithm*
  - *Useful in 2 or 3D space, more becomes difficult to visualize*

- **Adding another dimension (attribute)…**
  - *Can help distinguish behaviors…(unique feature vectors)*
  - *Cover's 95 theorem*

- **Choices (activation function, total error)…**
  - *Choose an activation function appropriate to solve the problem (hard activ. may be just fine for linearly separable patterns, etc.).*
  - *Small vs. large TE – very small TE may not be achievable for highly overlapping behaviors (Project 1&2)*