LEC 44

Supervised → Regression, Classification

Regression → Nonlinear

Classification → SVM, DT

1 Linear Regression

is a model which has labeled data ie Supervised learning.

It is machine learning technology used to model the relationship b/w two variables:

→ Independent variable (X)

→ Dependent variable (Y)

It assume a linear relationship b/w X & Y, meaning the changes in X result in proportional changes in Y.

$$Y = mX + C$$

where

$$m = \frac{n(\Sigma xy) - \Sigma(x)\Sigma(y)}{n(\Sigma x^2) - (\Sigma(x))^2}$$

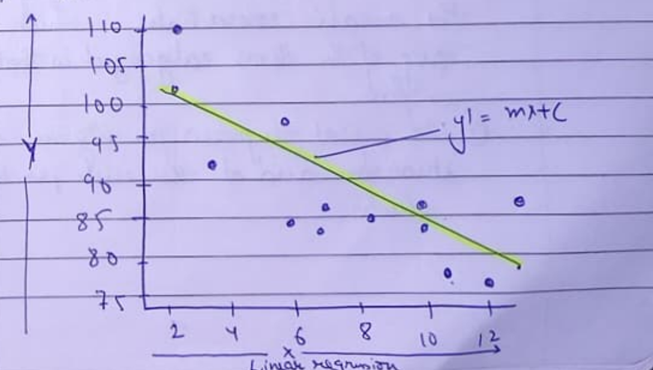
$$C = \frac{\Sigma y - m\Sigma x}{n}$$

Code Linear Regression

```
from scipy import stats
slope, intercept, _, _, _ = stats.linregress(x,y)
def myfunc(x):
    return slop * x + intercept   # mx+c

y1 []                             # y1=mx+c
for i in range (len(x)):
value = myfunc(x[i])
y1.append (value)

plt.scatter(x,y, color='blue')    # color dots blue
plt.plot (x, y1, color='yellow')  # colour line yellow
plt.xlabel ("X")                  # labels x axis
plt.ylabel ('Y')                  # labels Y axis
plt.title ("Linear regression")   # Name the graph
plt.show()
```



Exercise - Predicting Sales Based on Advertising

A company wants to predict sales (y) based on the amount spent on advertising (x in thousand of dollars)

The relationship follows

$y = a + bx$

task
(i) Compute the mean of x & y
(ii) Compute the regression coeff b
(iii) Compute the intercept a
(iv) Predict the sales if ~~70 of~~ $6,000 on advertising

Advertising $x	$y	xy ~~Sales $y~~	x^2
1	10	10	1
2	15	30	4
3	20	60	9
4	25	100	16
5	30	150	25
15	100	350	55

200
150
350

(i) Mean $x = \frac{\Sigma x}{n} = \frac{15}{5} = 3$

$y = \frac{\Sigma y}{n} = \frac{100}{5} = 20.$

(ii) $b = \cancel{\frac{n\,\Sigma xy - \Sigma x\,\Sigma y}{n(\Sigma x^2) - (\Sigma x)^2}} = \cancel{\frac{5\times(350) - (15)\times(100)}{5\times(55) - (15)^2}}$

1750 - 1500
275 - 3025
$= 0.0907$

(iii) $a = \frac{\Sigma y - m\,\Sigma x}{n} = \frac{100 - 5\times(15)}{5} = \frac{100-75}{5} = \frac{25}{5} = 5$

(ii) $a = \frac{n(\Sigma xy) - \Sigma(x)\,\Sigma(y)}{n(\Sigma x^2) - (\Sigma(x)^2)} \quad \frac{5\times(350) - 15\times(100)}{5\times 55 - (15)^2}$

$= \frac{1750 - 1500}{275 - 225}$

$= \frac{250}{50}$

$= 5.$

$y = 5 + 5x$

(iv) $y = 5 + 5\times 6 = 35.$

Ex2

$y = a + bx$

x	y	xy	x^2
50	200	10000	2500
60	250	15000	3600
80	300	24000	6400
100	350	35000	10000
120	400	48000	14400
410	1,500	1,32,000	36,900

(i) mean $\bar{x} = \dfrac{410}{5} = 82$

$\bar{y} = \dfrac{15000}{5} = 3000$

(ii) $b = \dfrac{n(\Sigma xy) - \Sigma(x)\,\Sigma(y)}{n(\Sigma x^2) - (\Sigma(x))^2}$

$= \dfrac{5\,(1,32,000) - 410 \times 1,500}{5 \times 36,900 - (410)^2}$

$= \dfrac{6,60,000 - 6,15,000}{1,84,500 - 1,68,100}$

$\dfrac{45,000}{16,400} = 2.74$

(iii) $a = \cancel{\Sigma y\ \Sigma x}\ \dfrac{\Sigma y - b\,\Sigma x}{n} = \dfrac{1500 - 2.74(410)}{5}$

$\dfrac{1500 - 1,124.99}{5}$

$= \dfrac{375.0}{5}$

$= 75$

(iv) Predict for 90 ng not hours

$y = 75 + 2.74 \times 90$

$y = 321.95$

2 K-Means Clustering

It is type of Unsupervised ML, It teaches a computer to use labeled, unclassified data and enabling the algo to operate on that data without supervision

- Unsupervised
  - Clustering
    - KMeans
    - DBScan
    - Hierarchical
      - Divisive hierarchical
      - Agglomerative
  - Association Learning

K-clustering depending on their distance from the center of the cluster

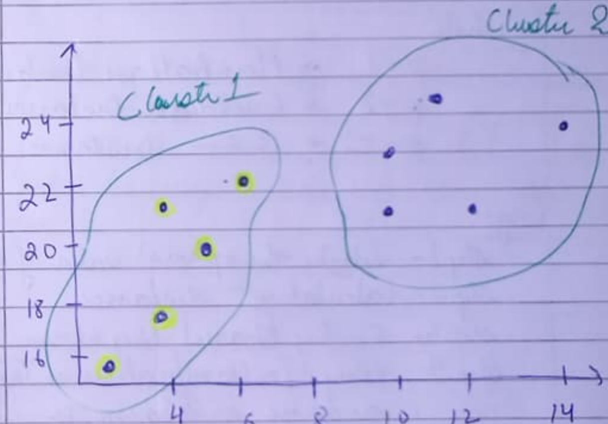
Algo

1. We randomly initialize k points, called means or cluster centroids
2. We categorize each item to its closest means ~~or~~ cluster and we update the mean's coordinates, which are the avg of the items categorized in that cluster so far
3. We repeat the process for a given no. of iterations and at the end, we have our clusters.

Code

```
from sklearn.cluster import KMeans
import numpy as np
import matplotlib.pyplot as plt
x = [4, 5, 10, 4, 3, 11, 14, 6, 10, 12]
y = [21, 19, 24, 17, 16, 25, 24, 22, 21, 21]

import Kr
Kmeans = KMeans(n_clusters = 2)   2 clusters will form
Kmeans.fit = (data)
plt.scatter(x, y, c = kmeans.labels_)
plt.show          # ploting the graph
```



K-Means

Q Apply K(=2)-Means algorithm over the data (185,72), (170,56), (168,60), (179,68), (182,72), (188,77) up to two iterations and show the clusters. Initially choose first two objects as initial centroids

Sol

Given, number of cluster to be created $k=2$ say C1, C2
no. of iterations = 2 and
The given data points can be represented in tabular form as

Instance	X	Y
1	185	72
2	170	56
3	168	60
4	179	68
5	182	72
6	188	77

also, first two objects are initial centroids
Centroid for 1st cluster $C1 = (185, 72)$
——— 2nd ——— $C2 = (170, 56)$

Iteration 1 → Now calculating similarity by using Euclidean distance measure as:

$d(c1,3) = \sqrt{(185-168)^2+(72-60)^2} = \sqrt{(17^2)+(12)^2} = \sqrt{433}$

$d(c2,3) = \sqrt{(170-168)^2+(56-60)^2} = \sqrt{2^2+(4)^2} = \sqrt{20}$

Here $d(c2,3) < d(c1,3)$

⇒ so data point 3 belongs to C2

$d(c1,4) = \sqrt{(185-179)^2+(72-68)^2} = \sqrt{52}$

$d(c2,4) = \sqrt{(170-179)^2+(56-68)^2} = \sqrt{225}$

Here $d(c1,4) < d(c2,4)$

⇒ so data point 4 belongs to C1

$d(c1,5) = \sqrt{9}$ $\quad d(c2,5) = \sqrt{400}$ Here $d(c1,5) < d(c2,5)$

⇒ so data point 5 belongs to C1

$d(c1,6) = \sqrt{34}$ $\quad d(c2,6) = \sqrt{765}$ Here $d(c1,6) < d(c2,6)$

⇒ so data point 6 belongs to C1

	X	Y	(C1) distance	(C2) distance	cluster
1	185	72			C1
2	170	56			C2
3	168	60	$\sqrt{433}$	$\sqrt{20}$	C2
4	179	68	$\sqrt{52}$	$\sqrt{225}$	C1
5	182	72	$\sqrt{9}$	$\sqrt{400}$	C1
6	188	77	$\sqrt{34}$	$\sqrt{765}$	C1

The resulting cluster after 1^{st} iteration is

1, 4, 5, 6
Cluster-1 (C1)

2, 3
Cluster-2 (C2)

Iteration 2 - Now calculating centroid for each cluster

Centroid for 1^{st} cluster $C1 = \left(\frac{185+179+182+188}{4}, \frac{72+68+72+77}{4} \right) = (183.5, 72.25)$

Centroid for 2^{nd} cluster $C2 = \left(\frac{170+168}{2}, \frac{56+60}{2} \right) = (169, 58)$

again calculating

$d(c1,1) = \sqrt{(183.5-185)^2 + (72.25-72)^2} = 1.5207$

$d(c2,1) = \sqrt{(169-185)^2 + (58-72)^2} = \cancel{2.2361}\ 21.2603$

Here $d(c1,1) < d(c2,1)$ C1

$d(c1,2) = 21.1261$
$d(c2,2) = 2.2361$
C2

$d(c1,3) = 19.7513$
$d(c2,3) = 2.2361$
C2

$d(c1,4) = 6.1897$
$d(c2,4) = 14.142$
C1

$d(c1,5) = 1.5207$
$d(c2,5) = 19.1050$
C1

$d(c1,6) = 6.5431$
$d(c2,6) = 26.8701$
C1

Now

Instance	X	Y	Distance (C1)	C2	Cluster
1	185	72	1.5207	21.2603	C1
2	170	56	21.1261	2.2361	C2
3	168	60	19.7513	2.2361	C2
4	179	68	6.1897	14.1421	C1
5	182	72	1.5207	19.105	C1
6	188	77	6.5431	26.8701	C1

{(185, 72), (179, 68)
(182, 72), (188, 77)}
C1

{(170, 56) (168, 60)}
C2

KNN

—/—/—

→ works by finding the K nearest neighbours to a given data point based on a distance metric, such as Euclidean distance

Distance measured

Minkowski Distance

$$d(x,y) = \left[\sum_{i=1}^{n} (x_i - y_i)^p\right]^{1/p}$$

while

$p=1 \rightarrow$ Manhattan Distance
$p=2 \rightarrow$ Euclidean Distance
$p=\infty \rightarrow$ Chebyshev Distance

Algo

step 1 - Selecting the optimal value of K
step 2 - Calculating distance
step 3 - Finding Nearest Neighbours
step 4 - Voting for Classification or Taking average for Regression

—/—/—

Code — standardize — mean = 0, std = 1

```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(x_train)
X_test = sc.transform(x_test)

Classifier = KNeighborsClassifier(n_neighbors = 5, metric = 'minkowski', p=2)
```

K=5

```
Classifier.fit(x_train, y_train)
y_pred = classifier.predict(x_test)
y_pred
```

array([2, 2, 0, 2, 0, 1, 0, 1, 1, 2, 0, 2, 1, 0, 0, 2, 1, 1, 1, 2, 2, 1, 1, 2, 2, 2, 0, 0, 2, 0, 2, 1, 0, 2, 2, 0])

```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
from sklearn.metrics import accuracy_score
print("Accuracy", accuracy_score(y_test, ypred))
cm
```

Output

Accuracy 0.9473684[?]1
array([[11, 0, 0],
[0, 10, 1],
[0, 1, 15]])

```
df = pd.DataFrame({'Real Values' y_test, 'Predicted V...
df
```

→ for Tabular

KNN

Q2 We have data from the questionnaires survey and objective testing with two attributes to classify whether a special paper tissue is good or not. Here is four training samples

X1 = Acid Durability (seconds)	X2 = Strength (kg/m^2)	Y = Classification
7	7	Bad
7	4	Bad
3	4	Good
1	4	Good

Now the factory produces a new paper tissue that pass laboratory test with X1 = 3 and X2 = 7. Without another expensive survey can we guess with the classification of then new tissue is?

① Determine parameter K = no. of nearest neighbors
Suppose use $\boxed{K=3}$

② Calculate distance b/w the query ~~time~~ instance and all the training samples

X1	X2	Distance to query	is included in 3NN
7	7	$\sqrt{(7-3)^2+(7-7)^2}=4$	Yes
7	4	$\sqrt{(7-3)^2+(4-7)^2}=5$	No
3	4	$\sqrt{(3-3)^2+(4-7)^2}=3$	Yes
1	4	$\sqrt{(1-3)^2+(4-7)^2}\simeq 2$	Yes

③ Majority voting

Since mans nabours are 4 as Bad
3 as good
2 as good

as majority voting then we have 2 good and 1 bad since 2>1 then we conclude that a new paper tissue that pass laboratory test with X1=3 and X2=7 is included in Good catagory.

Knn

Q1 Predicting Movie Genre

IMDb rating	Duration (min)	Genre
8.0 (Mission Impossible)	160	Action
6.2 (Gadar 2)	170	Action
7.2 (Rocky & Rani)	168	Comedy
8.2 (OMG 2)	155	Comedy

Now predict the genre of "Barbie" movie with IMDB rating 7.4 and duration 114 min

Ans

Step 1 Calculate Distance $\boxed{d(p,q) = \sqrt{(x-x_1)^2 + (y-y_1)^2}}$

Calculate the Euclidean distance b/w the new movies and each movie in the dataset $\Rightarrow x = 7.4 \quad y \Rightarrow 114$

Distance to $(8.0, 160) = \sqrt{(7.4-8.0)^2 + (114-160)^2} = \sqrt{0.36 + 2116} \simeq 46.00$

Distance to $(6.2, 170) = \sqrt{(7.4-6.2)^2 + (114-170)^2} = \sqrt{1.44 + 3136} \approx 56.01$

Distance to $(7.2, 168) = \sqrt{(7.4-7.2)^2 + (114-168)^2} = \sqrt{0.04 + 2916} \approx 54.00$

Distance to $(8.2, 155) = \sqrt{(7.4-8.2)^2 + (114-155)^2} = \sqrt{0.64 + 1681} \approx 41.00$

least distance

Step 2: Select K nearest Neighbours

Let K=3

So nearest neighbours will be 54.00, 46.00 and 41.00

C A C

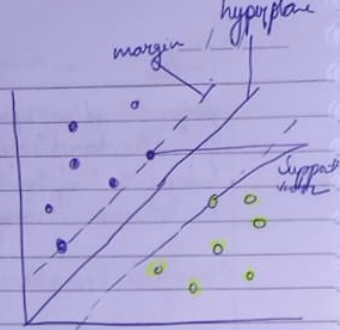
Step 3 Majority voting

Since nearest neighbours were 54.00 as comedy
46.00 as Action
41.00 as comedy

as majority voting the Barbie movies results to be COMEDY genre

SVM

Linear SVM
Non Linear SVM



Code.

```
import numpy as np                          → numpy lib
import matplotlib.pyplot as plt             // matplotlib lib
import pandas as pd                         → import pandas
from sklearn import datasets
iris = datasets.load_iris()                 // iris data
X=iris.data
y=iris.target
from sklearn.model_selection import train_test
X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.25)
```

training x,y

```
from sklearn.preprocessing import StandardScaler
sc= StandardScaler()
X_train= sc.fit_transform(X_train)
X_test= sc.transform(X_test)

from sklearn.svm import SVC                 # SVM class
classifier = SVC(kernel='linear', random_state=0)
classifier.fit(X_train, y_train)
```

__/__/__

```
y_pred = classifier.predict(X_test)
y_pred →
    array([2,0,2 2 0 2 0 0 2 0 1 1 0 2
           1,2,2,0,0,0,0,1,2,1,0
           1,2,2,2,2,2,0,1,0,1,2,1,0])
```

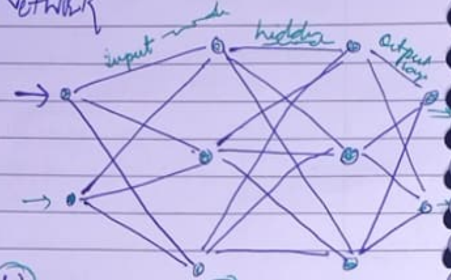
```
# Confusion matrix for the linear kernel
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
from sklearn.metrics import accuracy_score
print("Accuracy", accuracy_score(y_test, y_pred))
cm
```

Output

```
Accuracy: 0.9736842105263158
array([[14, 0, 0]
       [0, 7, 0],
       [0, 1, 16]])
```

CNN

Neural Network



1. Composed of Node layers
2. Use of Linear Regression Model
3. Data is passed b/w nodes feed forward.
4. Rely on Training Data
5. Neural Networks have Multiple Type
   - → ANN (artificial
   - → CNN (Convolutional
   - → RNN (Recurrent)

CNN (Convolutional Neural Network)

A deep learning algo for
- image processing
- computer vision
- increasingly in NLP

- inspired by the way brain processes visual info
- detect features like edges, shapes & patterns

Components

1. Input layer - receives raw pixel data (eg 28×28)
2. Convolutional layer - Applies filter that slide over the image and extract features like edges or corners

bias → neuron control
weight → network connection

3/ Activation function - add non linear to the model by apply
$$ReLU(x) = max(0, x).$$

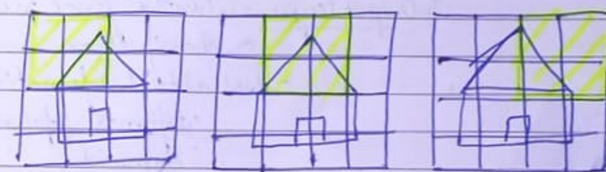
4 Pooling Layer - reduces the size of feature maps while retaining imp. info.
- helps prevent overfitting and improves efficiency

5/ Fully connected layer - Combines features from previous layers for classification

6/ Output Layer - Uses functions like softmax to give final prediction probability

Example

process



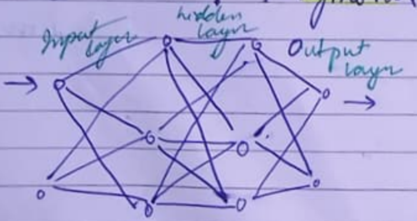
So on ...

ANN (Artificial Neural Network)

a core part of deep learning that mimics human brain neuron's work to process and learn from data

Basic Architecture

1. **Input layer** - Accepts raw features (pixels values, tabular data, etc).
2. **Hidden Layers** - performs computation using weights, biases and activation function.
   - Can have one or more hidden layers
3. **Output layer** - Gives the final prediction or classification.
   - Uses actual function like
     - Softmax (for multiclass)
     - Sigmoid



Working

- Take inputs $x_1, x_2 \dots x_n$.
- Multiplies with weight $w_1, w_2 \dots w_n$
- Adds a bias b
- Applies a activation function like ReLU, Sigmoid, Tanh

$$\boxed{\text{Output} = f(w_1 x_1 + w_2 x_2 + \dots + w_n x_n + b)}$$

RNN (Recurrent Neural Network)

- **LSTM** (Long Short Term Memo)
- **GRU** (Gate Recurrent Unit)
- **BiRNN** (Bidirectional RNN)
- **Seq2Seq** (Encoder Decod)

- only works with sequential or time series data like text, speech or stock prices

- RNN has "**MEMORY**", they can use prior outputs as inputs to influence future predictions. This make them ideal for task where order matters

$$\boxed{\text{RNNs} = \text{Neural Networks} + \text{Feedback Loops}}$$

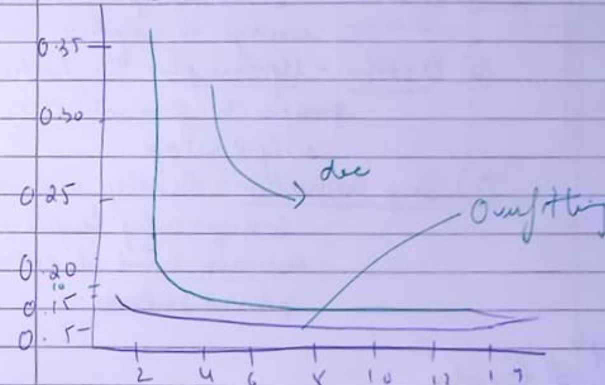
Working Each hidden state h_t depends on
- Current input x_t
- previous hidden state h_{t-1}

$x_1 \rightarrow h_1 \rightarrow y_1$
$\downarrow$
$x_2 \rightarrow h_2 \rightarrow y_2$
$\downarrow$
$x_3 \rightarrow h_3 \rightarrow y_3$
... so on

Overfitting

```
import matplotlib.pyplot as plt
acc = history.history ['accuracy']
val_acc = history ['val_accuracy']
loss = history.history ['loss']
val_loss = history.history ['val_loss']
epochs = range [1, len (acc)+1)
plt.plot (epochs, acc, 'r', label = 'Training acc')
plt.plot (epoch, val_acc, 'b' lable = 'Validation acc')
plt.title ('Training and validation accuracy')
plt.ylable ('accuracy')
plt.xlable ('epoch')
plt.legend ()
# plt.ylim ([0, 1,2])
plt.figure ()
plt.plot (epochs, loss, 'r', label = 'Training loss')
plt.plot (epochs, val_loss, 'b', label = 'Validation loss')
plt.title ('Training and validation loss')
plt.ylabel ('loss')
plt.xlabel ('epoch')
plt.legend ()
# plt.ylim ([0, 2])

plt.show
```



GAN

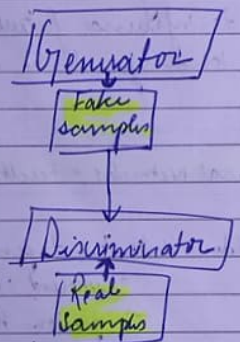
GAN (Generative Adversarial Network)

→ deep learning model used for generating new, realistic data that mimics a given data (like image, audio, video)

A GAN has 2 neural network competing with each other

① Generator (G)
- Create fake data that look real
- Takes random noise as input and outputs the data

② Discriminator (D)
- Detect wether a input is fake or real
- Acts like a critic or judge



How GAN Works?
1. The generator tries to fool the discriminator with fake data
2. The Discriminator tries to catch the Generator by distinguishing fake from real
3. Both improve over time:
   - The Generator gets better at making realistic data
   - The Discriminator gets better at detecting fakes

This is called a minimax game

Type	Best for	Key feature
ANN	General tasks	fully connected layer
CNN	Images	Convolution filter
RNN	Sequences	Memo of past inputs
LSTM/GRU	Long sequences	Handles depends
GAN	Data generation	Competing networks
Autoencoder	Compression	Encoder-decoder
Transformer	NLP & beyond	Attention mechanism
GNN	Graphs	Node/edge learn
SOM	Clustering	Unsupervised learning

UNet Architecture

It is typically used as generative especially in tasks where the output is an image similar in structure to the input such as
- → Image to image translation
- → Super-resolution
- → Semantic segmentation.

It consists of
1. Encoder - Downsamples the input image to capture context
2. Decoder - Upsamples the feature maps to generate an output image
3. Skip connections - Directly connect corresponding layers in encoder and decoder to preserve details

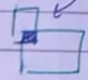
Example
```
# Encoder
e1 = Conv2D(64)(input)
e2 = Conv2D(128)(LeakyReLU(e1))
...
# Decoder
~~d1 = Conv2DTranspose(128)(ReLU(eN))~~
d1 = Concatenate()([d1, eN-1])
...
output = Conv2D(3, activation='tanh')(d_last)
```

Prediction using U-Net libraries

f1 score

```
from sklearn.metrics import f1_score
from tensorflow.keras.metrics import MeanIOU
```

IOU = Intersection over Union $\Longrightarrow \dfrac{|A \cap B|}{|A \cup B|}$



$$f1\ Score = 2 \times \frac{1}{\frac{1}{Precision} + \frac{1}{Recall}}$$

Lec 46

Q1 Find the linear Regression equation for the following two sets of data

x	2	4	6	8
y	3	7	5	10

Sol

	x	y	x^2	xy
	2	3	4	6
	4	7	16	28
	6	5	36	30
	8	10	64	30
Sum	20	25	120	144

We know $y = ax + c$

where $a = \dfrac{n\sum xy - \sum x \sum y}{n\sum x^2 - (\sum x)^2}$

$= \dfrac{4 \times 144 - 20 \times 25}{4 \times 120 - 400} = \dfrac{76}{80} = 0.95$

$b = \dfrac{\sum y \sum x^2 - \sum x \sum xy}{n\sum x^2 - (\sum x)^2} = \dfrac{25 \times 120 - 20 \times 144}{4 \times 120 - 400}$

$= \dfrac{120}{80} = 1.5$

$\boxed{y = 0.95x + 1.5}$

$y = \underset{\text{slope}}{a}x_1 + \underset{\text{slope}}{b}x_2 + c \rightarrow \text{intercept}$

Q2 Find a linear regression equation for following data

Product 1 Sales (x_1)	Product 2 Sales (x_2)	Weekly Sales (y)
1	4	1
2	5	6
3	8	8
4	2	12

$\hat{a} = [(X^T X)^{-1}$

$X = \begin{bmatrix} 1 & 1 & 4 \\ 1 & 2 & 5 \\ 1 & 3 & 8 \\ 1 & 4 & 2 \end{bmatrix} \quad y = \begin{bmatrix} 1 \\ 6 \\ 8 \\ 12 \end{bmatrix}$

$X^T X = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 \\ 4 & 5 & 8 & 2 \end{bmatrix} \begin{bmatrix} 1 & 1 & 4 \\ 1 & 2 & 5 \\ 1 & 3 & 8 \\ 1 & 4 & 2 \end{bmatrix} = \begin{bmatrix} 4 & 10 & 19 \\ 10 & 30 & 46 \\ 19 & 46 & 109 \end{bmatrix}$

$(X^T X)^{-1} = \begin{bmatrix} 3.15 & -0.59 & -0.36 \\ -0.5 & 0.2 & 0.016 \\ -0.3 & 0.016 & 0.054 \end{bmatrix}$

$((X^T X)^{-1}) X^T = \begin{bmatrix} 3.15 & -0.59 & -0.36 \\ -0.5 & 0.2 & 0.016 \\ -0.3 & 0.016 & 0.014 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 \\ 4 & 5 & 8 & 2 \end{bmatrix}$

$= \begin{bmatrix} 0.05 & 0.47 & -1.02 & 0.19 \\ -0.32 & -0.018 & 0.155 & 0.24 \\ -0.65 & 0.005 & 0.185 & -0.125 \end{bmatrix}$

_/_/_

$$\hat{a} = ((X^T X)^{-1} X^T) Y = \begin{bmatrix} -1.69 \\ 3.48 \\ -0.05 \end{bmatrix} \begin{array}{l} - a_0 \\ - a_1 \\ - a_2 \end{array}$$

then

$$\boxed{Y = 3.48 x_1 - 0.05 x_2 - 1.69}$$

Q3 Estimate the conditional probability of each attribute {color, legs, height, smelly} for the species class {H, M} using the data table given below. Also using these probabilities estimates the probability values for the new instance.

Color	Legs	height	Smelly	Species
White	3	short	Yes	M
Green	2	Tall	No	M
Green	3	Short	Yes	M
White	3	Short	Yes	M
Green	2	Short	No	H
White	2	Tall	No	H
White	2	Tall	No	H
White	2	Short	Yes	H

_/_/_

Color	M	H
White	2/4	3/4
Green	2/4	1/4

Legs	M	H
2	1/4	4/4
3	3/4	0/4

Smelly	M	H
Yes	3/4	1/4
No	1/4	3/4

Height	M	H
Tall	3/4	2/4
Short	1/4	2/4

$$P(M \mid \text{New instance}) = \frac{P(\text{New instance}/M)\, P(M)}{P(\text{New instance})}$$

$$P(M \mid \text{New instance}) = P(\text{New instance} \mid M)\, P(M)$$

$$\frac{2}{4} \times \frac{1}{4} \times \frac{3}{4} \times \frac{1}{4} \times \frac{1}{2}$$

$$= 0.0117$$

$$P(H \mid \text{New instance}) = \frac{P(\text{New instance} \mid H)\, P(H)}{P(\text{New instance})}$$

$$P(H \mid \text{New instance}) = P(\text{New instance} \mid H)\, P(H)$$

$$\frac{1}{4} \times \frac{4}{4} \times \frac{2}{4} \times \frac{3}{4} \times \frac{1}{2}$$

$$= 0.047$$

$P(H/\text{New instance}) > P(M \mid \text{New instance})$

Species class → of Class H.

Q4 An unfair coin is flipped 100 times, and 61 times heads are observed. What is the Maximum Likelihood Estimation when nothing is previously known about the coin?

Binomial distribution → P, $q = (1-p)$

success   failure

$$P(H=61 \mid P) = \binom{100}{61} p^{61} (1-p)^{39}$$

$$\frac{d\, P(H=61 \mid P)}{dp} = \binom{100}{61}\left[61 p^{60}(1-p)^{39} - 39 p^{61}(1-p)^{38}\right] = 0$$

$$\Rightarrow 61 p^{60}(1-p)^{39} - 39 p^{61}(1-p)^{38} = 0$$

$$p = 0, \frac{61}{100}, 1$$

$P(H=61 \mid P=0) = 0$

$P(H=61 \mid P=1) = 0$

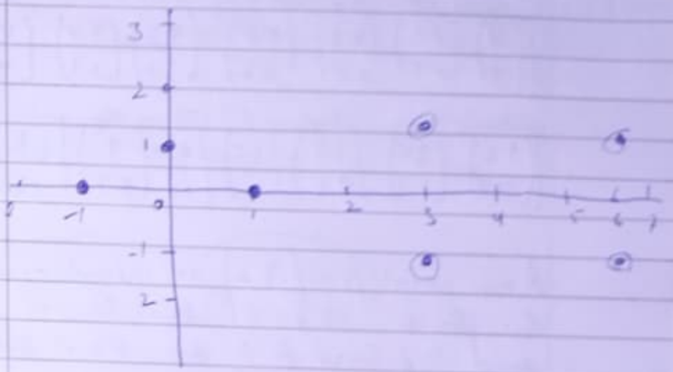
$P\left(H=61 \mid P=\frac{61}{100}\right) = \binom{100}{61}\left(\frac{61}{100}\right)^{61}\left(\frac{39}{100}\right)^{39} \neq 0$

$P = \frac{61}{100}$ → MLE is achieved

Q5 Suppose we are given the following +ve and -ve labeled data points in R^2

$\{(3,1), (3,-1), (6,1), (6,-1)\} \in$ +ve class

$\{(1,0), (0,1), (0,-1), (-1,0)\} \in$ -ve class



$S_1 = \begin{pmatrix}1\\0\end{pmatrix} \rightarrow -1$ ⊙ → +1

$S_2 = \begin{pmatrix}3\\1\end{pmatrix} \rightarrow +1$ • → -1

$S_3 = \begin{pmatrix}3\\-1\end{pmatrix} \rightarrow +1$

$a_1 \tilde{S}_1 \tilde{S}_1 + a_2 \tilde{S}_2 \tilde{S}_1 + a_3 \tilde{S}_3 \tilde{S}_1 = -1$

$a_1 \tilde{S}_1 \tilde{S}_2 + a_2 \tilde{S}_2 \tilde{S}_2 + a_3 \tilde{S}_3 \tilde{S}_2 = +1$

$a_1 \tilde{S}_1 \tilde{S}_3 + a_2 \tilde{S}_2 \tilde{S}_3 + a_3 \tilde{S}_3 \tilde{S}_3 = 1$

Lec 47

__/__/__

Q1 There are 18 points in an axis plane, such that

$$\left[\begin{array}{ccccc} x_1 & x_2 & x_3 & x_4 & x_5 \\ \begin{pmatrix} 0.8 \\ 0.8 \end{pmatrix} & \begin{pmatrix} 1 \\ 1 \end{pmatrix} & \begin{pmatrix} 1.2 \\ 0.8 \end{pmatrix} & \begin{pmatrix} 0.8 \\ 1.2 \end{pmatrix} & \begin{pmatrix} 1.2 \\ 1.2 \end{pmatrix} \end{array}\right]$$ belongs to class 1

$$\left[\begin{array}{ccccccc} x_6 & x_7 & x_8 & x_9 & x_{10} & x_{11} & x_{12} \\ \begin{pmatrix} 4 \\ 3 \end{pmatrix} & \begin{pmatrix} 3.8 \\ 2.8 \end{pmatrix} & \begin{pmatrix} 4.2 \\ 2.8 \end{pmatrix} & \begin{pmatrix} 3.8 \\ 3.2 \end{pmatrix} & \begin{pmatrix} 4.2 \\ 3.2 \end{pmatrix} & \begin{pmatrix} 4.4 \\ 2.1 \end{pmatrix} & \begin{pmatrix} 4.4 \\ 4.4 \end{pmatrix} \end{array}\right]$$ belong to class 2

$$\left[\begin{array}{cccccc} x_{13} & x_{14} & x_{15} & x_{16} & x_{17} & x_{18} \\ \begin{pmatrix} 3.2 \\ 0.4 \end{pmatrix} & \begin{pmatrix} 3.2 \\ 0.7 \end{pmatrix} & \begin{pmatrix} 3.8 \\ 0.5 \end{pmatrix} & \begin{pmatrix} 3.5 \\ 1 \end{pmatrix} & \begin{pmatrix} 4 \\ 1 \end{pmatrix} & \begin{pmatrix} 4 \\ 0.7 \end{pmatrix} \end{array}\right]$$ belong to class 3

A new point $P = \begin{pmatrix} 4.2 \\ 1.8 \end{pmatrix}$ is introduced into the plane. What class does the point P belong to? Use KNN technique with $k=5$ to obtain the result.

We have to find euclidean distance betn P & other points

Euclidean distance $d(x,y) = \sqrt{(x_1-y_1)^2+(x_2-y_2)^2}$

$d(x_1, P) = \sqrt{(0.8-4.2)^2+(0.8-1.8)^2} = 3.54$

Similarly

$d(x_1, P) = 3.54$ $\qquad d(x_2, P) = 3.30$

$d(x_3, P) = 3.16$ $\qquad d(x_4, P) = 3.45$

$d(x_5, P) = 3.06$ $\qquad d(x_6, P) = 1.21$

$d(x_7, P) = 1.07 \leftarrow$ $\qquad d(x_8, P) = 1 \leftarrow$

$d(x_9, P) = 1.45$ $\qquad d(x_{10}, P) = 1.4$

$d(x_{11}, P) = 1.02 \leftarrow$ $\qquad d(x_{12}, P) = 2.60$

$d(x_{13}, P) = 1.72$ $\qquad d(x_{14}, P) = 1.48$

$d(x_{15}, P) = 1.36$ $\qquad d(x_{16}, P) = 1.06 \leftarrow$

$d(x_{17}, P) = 0.82 \leftarrow$ $\qquad d(x_{18}, P) = 1.12$

Now since $k=5$, nearest neighbour to P are

$X_{17}, X_8, X_{11}, X_{16}, X_7$

Classes 3

Vote

3 points $\longrightarrow$ class 2

2 points $\longrightarrow$ class 3

The point P belongs to class 2

__/__/__

Q2 Use LDA for two classes C1 and C2 to cluster into two groups. $C1 = \begin{bmatrix} 1 & 2 \\ 2 & 3 \\ 3 & 3 \\ 4 & 5 \\ 5 & 5 \end{bmatrix}$ and $C_2 = \begin{bmatrix} 4 & 2 \\ 5 & 0 \\ 5 & 2 \\ 3 & 2 \\ 5 & 3 \\ 6 & 3 \end{bmatrix}$ new transformation point will be ____

$\Rightarrow \mu_1 = \begin{bmatrix} 3 \\ 3.6 \end{bmatrix} \quad \mu_2 = \begin{bmatrix} 4.67 \\ 2 \end{bmatrix}$

Scatter matrix S_i, $S_1 = \Sigma (x-\mu_1)(x-\mu_1)^T$

$$S_1 = \left[\begin{bmatrix}1\\2\end{bmatrix} - \begin{bmatrix}3\\3.6\end{bmatrix}\right]^2 + \left[\begin{bmatrix}2\\3\end{bmatrix} - \begin{bmatrix}3\\3.6\end{bmatrix}\right]^2 + \left[\begin{bmatrix}3\\3\end{bmatrix} - \begin{bmatrix}3\\3.6\end{bmatrix}\right]^2 + \left[\begin{bmatrix}4\\5\end{bmatrix} - \begin{bmatrix}3\\3.6\end{bmatrix}\right]^2 + \left[\begin{pmatrix}5\\5\end{pmatrix} - \begin{pmatrix}3\\3.6\end{pmatrix}\right]^2$$

$$= \begin{bmatrix} 10 & 8 \\ 8 & 7.2 \end{bmatrix}$$

Similarly $S_2 = \begin{bmatrix} 5.33 & 1 \\ 1 & 6 \end{bmatrix}$

Within class scalar matrix, $S_W = S_1 + S_2$

$$S_W = \begin{bmatrix} 15.33 & 9 \\ 9 & 13.20 \end{bmatrix}$$ $\longrightarrow$ We have to first inverse

__/__/__

Between-class scatter matrix $S_B = (\mu_1 - \mu_2)(\mu_1 - \mu_2)^T$

$$S_B = \left[\begin{bmatrix}3\\3.6\end{bmatrix} - \begin{bmatrix}4.67\\2\end{bmatrix}\right]\left[\begin{pmatrix}3\\3.6\end{pmatrix} - \begin{bmatrix}4.67\\2\end{bmatrix}\right]^T$$

$$= \begin{bmatrix} 2.79 & -2.67 \\ -2.67 & 2.56 \end{bmatrix}$$

We know $w = S_W^{-1}(\mu_1 - \mu_2)$

$$= \frac{1}{15.33 \times 13.20 - 81} \begin{bmatrix} 13.20 & -9 \\ -9 & 15.33 \end{bmatrix} \begin{bmatrix} -1.67 \\ 1.6 \end{bmatrix}$$

$$= \begin{bmatrix} 0.109 & -0.074 \\ -0.074 & 0.126 \end{bmatrix} \begin{bmatrix} -1.67 \\ 1.6 \end{bmatrix}$$

$$= \begin{bmatrix} -0.3 \\ 0.326 \end{bmatrix}$$

_/_/_

Q4 In the third iteration of AdaBoost, the weight assigned to a misclassified data point is 0.4. If the initial weight for all data points is 1, what was the misclassification rate of this data point at the end of the second iteration?

$D_{t+1}(n) = D_t(n) \exp(\alpha_t)$ for misclassification

where $\alpha_t = \frac{1}{2} \ln \frac{(1-\epsilon_t)}{\epsilon_t}$ — (1)

$0.4 = 1 \exp(\alpha_t) \Rightarrow \alpha_t = \ln(0.4)$

Substituting α_t in eq^n (1) $\rightarrow$

$$\ln(0.4) = \frac{1}{2} \ln \frac{(1-\epsilon_t)}{\epsilon_t}$$

$$\Rightarrow -1.832 = \ln \frac{1-\epsilon_t}{\epsilon_t}$$

$$0.16 = \frac{1-\epsilon_t}{\epsilon_t}$$

$$\boxed{\epsilon_t = 0.862.}$$

Lec 48

_/_/_

Q Kmean clustering on the data set *
$X = \{31, 43, 30, 58, 62, 34, 92, 65, 33, 90\}$
with $k=3$. The set of initial data labels are provided in Table given below.

iteration	31	43	30	58	62	34	92	65	33	90
0	1	1	1	2	2	2	3	3	3	1

- Find the set of final data labels for the initial label
- Find the final cluster u_j ($j = 1, 2, 3$) (till converge) for the table given above

itr.	31	43	30	58	62	34	92	65	33	90	u_1	u_2	u_3
0	1	1	1	2	2	2	3	3	3	1	48.5	51.33	63.33
1	1	1	1	3	3	1	3	3	1	3	34.2	51.33	73.4
2	1	2	1	2	2	1	3	2	1	3	32	54.33	84.33
3	1	1	1	2	2	1	3	2	1	3	34.2	61.67	91
4	1	1	1	2	2	1	3	2	1	3	34.2	61.67	91

final ~~data~~ cluster $\rightarrow$ 34.2, 61.67, 91

- perform single single hierarchical agglom clustering on the set of points $S = \{-5, 2, 5, 7, -10, 10, 0\}$. Also draw the dendrogram.

Step 1 - Initialize Cluster

[-5] [2] [5] [7] [-1] [10] [0]

Step 2 - Calculate pairwise distance

	-5	2	5	7	-1	10	0
-5	0	7	10	12	4	15	5
2	7	0	3	5	3	8	2
5	10	3	0	2	6	5	5
7	12	5	2	0	8	3	7
-1	4	3	6	8	0	11	1
10	15	8	5	3	11	0	10
0	5	2	5	7	1	10	0

Step 3 - Merging

$S = \{-5, 2, 5, 7, -1, 10, 0\}$

$= -5, 2, 5, 7, (-1, 0), 10$

$= -5\ (2, -1, 0), (5, 7)\ 10$

$= -5\ (2, -1, 0, 5, 7, 10)$

Dendrogram

=