

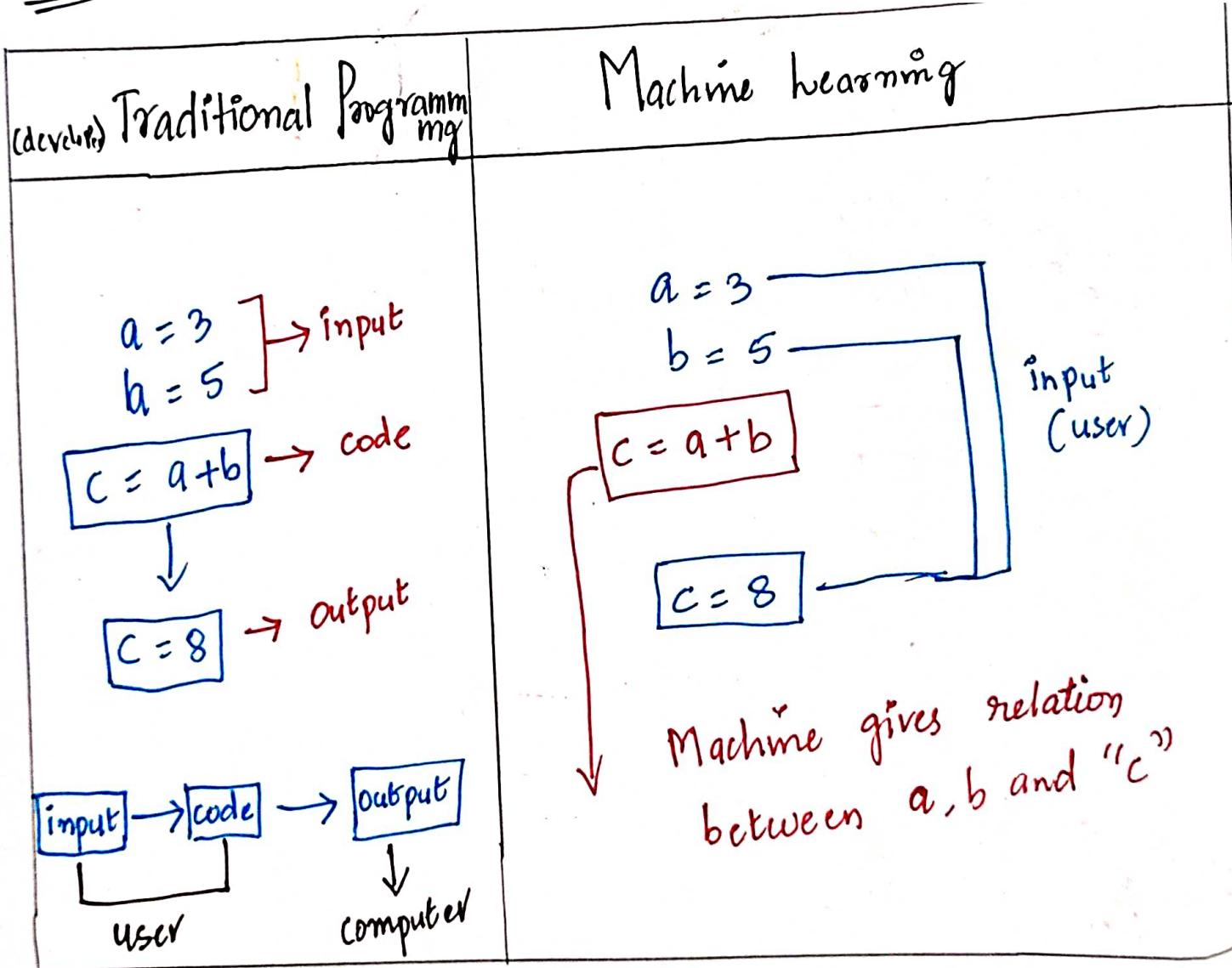
Part 01

⇒ Machine learning

Ques:- What is Machine Learning?

Ques:- What is Machine learning?
Ans :- Machine learning is The subField of Computer
Science that gives "Computer Ability to learn"
without being Emplitcally Programmed"

Example:-



input	output
X	Y
1	10
2	20
3	30
4	40
5	50
6	?

$\rightarrow g$ (Prediction)
 $"y = 10x"$

input have more columns

x_1	x_2	y

$$y = f[x_1, x_2]$$

$$ax_1 + bx_2 = y$$

MODELLING

Q. How I can say it is 6?

Ans :- * Identifying The between "x" and "y"

$$\therefore y = 10x \Rightarrow \text{MODEL}$$

* Substitute "6" in "x" value

$$y = 10[6] = 60$$

$$\therefore y = 60.$$

PREDICTION

Machine learning

* Machine
 * Identifying

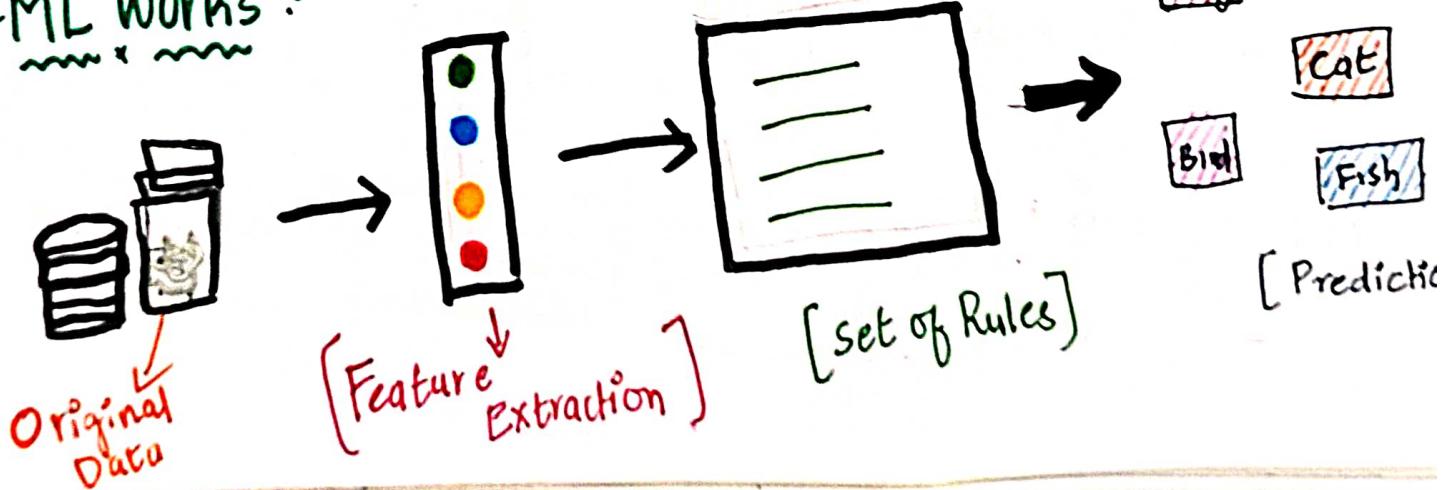
Output

learn The given Data.

The relation between input and
 is called Machine Learning.

* Prediction

* ML WORKS :-



⇒ Major Machine Learning Techniques

- * **Regression / Estimation**
 - Predicting **Continuous Values**
- * **Classification**
 - Predicting **Discrete Values** / **Item class** / **category of a case**
- * **clustering**
 - (No prediction)
- Finding the **structure of data**; **Summarization**
- * **Associations**
 - Associating **Frequent Co-occurring items / Events**
Ex: Buying Brush/paste.
- * **Anomaly detection**
 - Discovering **abnormal** **Unusual Cases**
- * **Sequence Mining**
 - Predicting next Events; **Click stream** (**Markov Model, HMM**)

* Dimensional Reduction

- Reducing The size of data [PCA]

* Recommendation systems

- Recommending Items.

Ques What is Supervised Learning?

Ans:- We "Teach The model" then with that knowledge,
it can Predict Unknown (or) Future instances.

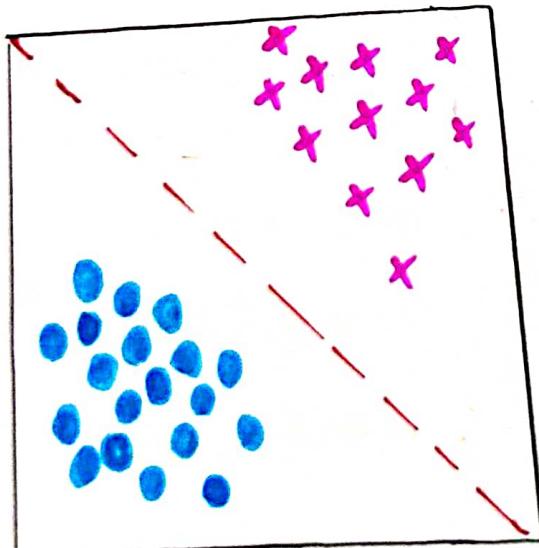
→ Where Ever The Output Variable is Available
is called as "Labeled data".

"In Supervised Learning Model, "The Algorithm Learns on a
Label dataset , To generate reasonable predictions For The

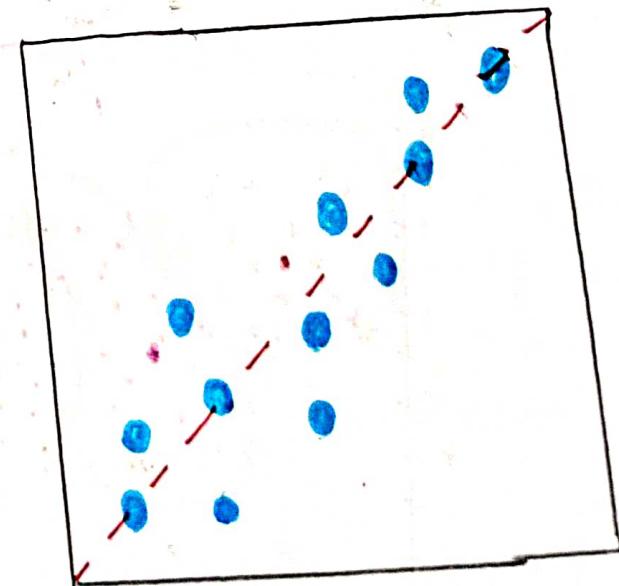
response to new data.

1. Regression [Continuous Data]

2. Classification [Discrete Data]



* Classification



* Regression

Ques :- What is UnSupervised Learning?

Ans :- UnSupervised Learning, we have

Unlabeled Data

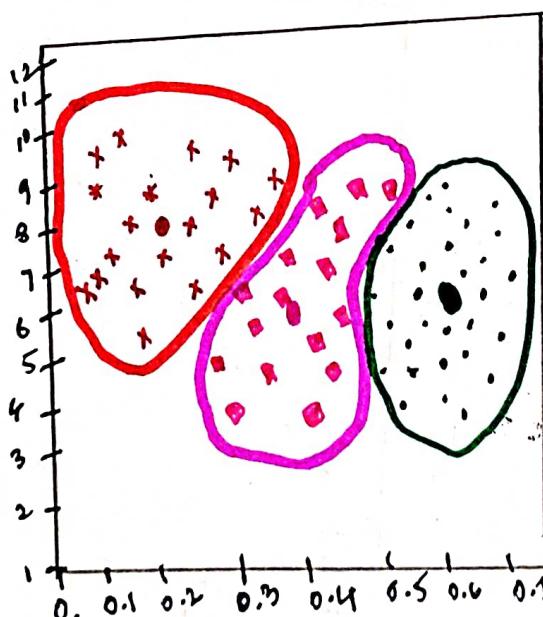
i.e. **No Output Feature Available.**

"**Unlabeled Data**" that the algorithm tries to make sense of by **Extracting Features**, **Co-occurrence** and **Underlying patterns on its own.**

1. clustering
2. Anomaly Detection
3. Association
4. Auto Encoders

Ques :- what is clustering?

"clustering" is **grouping of data points** (or) **Objects that are somehow similar by**



* clustering

* Supervised Learning

- * Regression
 - classifies labeled data
- * Classification
 - Predicts trends using previous Labeled data
- * Has more Evaluation methods than Unsupervised Learning.
- * Controlled Environment

* Unsupervised Learning

- * Clustering
 - finds pattern and grouping from Unlabeled data.
- * Has Fewer Evaluation Methods than Supervised Learning
- * Less Controlled Environment

Dt: 6/4/22
10:30pm

6/4/22
3:30 pm

Sklearn

↓
Scikit Learn

Feature Scaling

From Sklearn.preprocessing import StandardScaler

df["Age-sc"] = sc.fit_transform(df[["Age"]])

Using Pandas
input, we giving as Data Frame

Out :- We get DataFrame.

missing values

From sklearn.impute

df["CLMAGE"] = pd.DataFrame(mean_imputer.fit_transform(df[["CLMAGE"]]))

import SimpleImputer
using Same Pandas.

Out :- We get array.

Input, we giving Same Data Frame

Example :-

User Defined Functions

```
def add(a,b)
```

$$c = a + b$$

```
return float(c)
```

→ add(3,4)

Out: 7.0

→ Float value

```
def add(a,b)
```

$$c = a + b$$

```
return c
```

→ add(3,4)

Out: 7

```
def mains(a,b)
```

$$c = a + b$$

$$d = a * b$$

$$e = a \% b$$

$$f = a - b$$

```
print("sum": c)
```

```
print("mult": d)
```

```
print("div": e)
```

```
print("sub": f)
```

→ Saving in "Jack.py"

From Jack import maths

↓
module

↓
Function

* Because, function is created in a such a way That
answer should be float. / But Both The Functions are same
↓ input

Only output is different.

* Same,
Simple Imputer, Standard Scaler and More Functions

From sk.learn are InBuilt Function.

* We Can't Remember Every function. So, We Take The

and see The Output Variable Type.

Help function

If Out is Array. We change (or) Convert into

Data Frame , Just writing [Pd. DataFrame] before The

Function. Download Part 02 & Part 03

<https://t.me/AIMLDeepThaught/665>

→ Estimator API

Modelling

```
# import 'ML' Algorithm  
# model = MLAlgo()  
# model.fit(x-train, y-train)
```

Prediction

```
# yPred = model.predict(x-test)
```

Evaluation

```
# Comparison (y pred, y test)
```

Accuracy

Ex:-

X	Y
1	10
2	20
3	30
4	40
5	50
6	?

model = Algorithm [$y = 10x$]

yPred # model.predict [x-test]

$$y = 10x \quad [x=6]$$

$$y = 10 \times 6$$

$$\text{output} = 60$$

Relation : $y = 10(x)$



ML Algorithm [model]

Students



Learn



write The Exam



Evaluate



grade / percentage

Download Part 02 & Part 03

<https://t.me/AIMLDeepThaught/665>

* MODEL Evaluation * → Train-test-Split

* Train and Test on Same Dataset

* Train / Test Split

EX:-

Case 1

* In a class Sir given + Interview question

Train → 100 Q & A



Test → 30 Q

80% marks (%)

You're a good student, But
Only for this class Only.

* If whatever The Data I have
given [100 Q/A]. if we ask
30 Q/A from the same dataset.
it gives better performance, But.
Only for that dataset it
works, not for any new
dataset (Q/A)

Case 2

100 Q/A

70
[Training]

30
[Testing]

↓
For These 70 (Q/A) Given To Machine, not
to understand, But given Only For
Understand Concepts Only. not For reading data

Now, We ask new 30 (Q/A), not from
70 (Q/A) that we have given, and
Evaluate Accuracy For Training and For
Testing, Separately.

Example

	input						y (Output)
	x_1	x_2	x_3	x_4	x_5	x_6	y
0	2.0	3.0	3	8	4	3	196
1	8.0	4.8	2	22	11	9	121
2	3.0	3.2	1	44	22	16	136
3	4.8	3.8	8	2	118	22	225
4	9.0	5.2	4	3	22	11	224
5	6.8	4.4	3	44	114	18	230
6	8.0	3.2	8	8	55	16	255
7	2.8	8.9	2	3	88	14	264
8	8.0	10.10	5	2	63	12	288

If This is input, what is out (Predict by model)
Even Though we have "y" value we consider it, But Machine has Predict same Value.

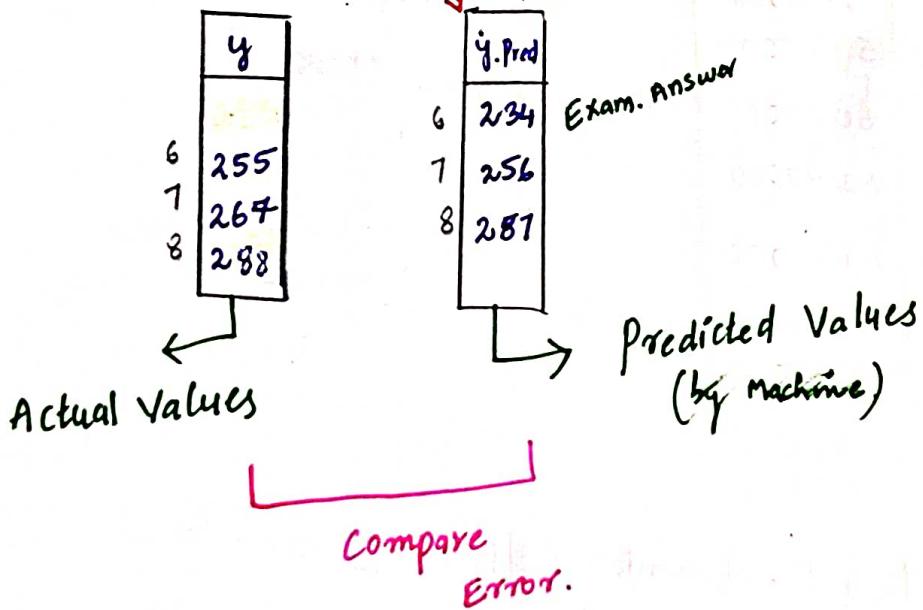
* In Testing, we give input data that we are separated First.

* In Testing, we give input data that we are separated First.
and check with Accuracy or "y" value (output)

* In Testing, we don't take "y" variable (output) because To

Check Given data, Model is Working (or) not.

MODEL



Download Part 02 & Part 03

<https://t.me/AIMLDeepThaught/665>

Code :-

```
# Import numpy as np
# Import Pandas as pd
# Import matplotlib.pyplot as plt
%matplotlib inline
```

```
# df = pd.read_csv ("homeprices.csv")
```

df

Out :-

	town	area	Price
0	Chennai	2600	5500000
1	Chennai	3000	5650000
2	Chennai	3200	6100000
3	Chennai	3600	6800000
4	Bangalore	2800	5850000
5	Bangalore	3300	6150000
6	Bangalore	2600	6500000
7	Bangalore	3600	7100000
8	Hyderabad	2600	5750000
9	Hyderabad	2900	6000000
10	Hyderabad	3100	6200000
11	Hyderabad	3600	6950000

Here we use same dataset of Encoding. and we divided them into train / test. it will easy to understand by taking same Data set for further steps.

creating dummies

```
# df_dum = pd.get_dummies (df, dropfirst=True)
```

```
# df_dum
```

Download Part 02 & Part 03

Out <https://t.me/AIMLDeepThaught/665>

Now, this
Changed
dataset, ←
Is we Consider
Further to
Train/test

Area	Price	town-chennai	town-Hyderabad
2600	5500000	1	0
3000	5650000	1	0
3200	6100000	1	0
3600	6800000	1	0
2600	5850000	0	0
2800	6150000	0	0
3300	7100000	0	0
3600	5750000	0	0
2600	6000000	0	1
2900	6500000	0	1
3100	6200000	0	1
3600	6950000	0	1

Here Based on Area (sft) and town, we want predict The price

Independent Variable

So... area, town-chennai, town-Hyderabad are → Input variable

price is Output variable (y)

Dependent Variable

Because, it's ←
Depending on "x" values To give output.

Option To Consider x & y Variable

x = df_dum.drop("Price", axis = "columns")

y = df_dum.Price

check "x" → Input Variables
(or)

X
Independent
Variables

Here we split the data into Train / test.

stores in "x" and "y".

check "y" → Output Variables
(or)
Dependent Variables

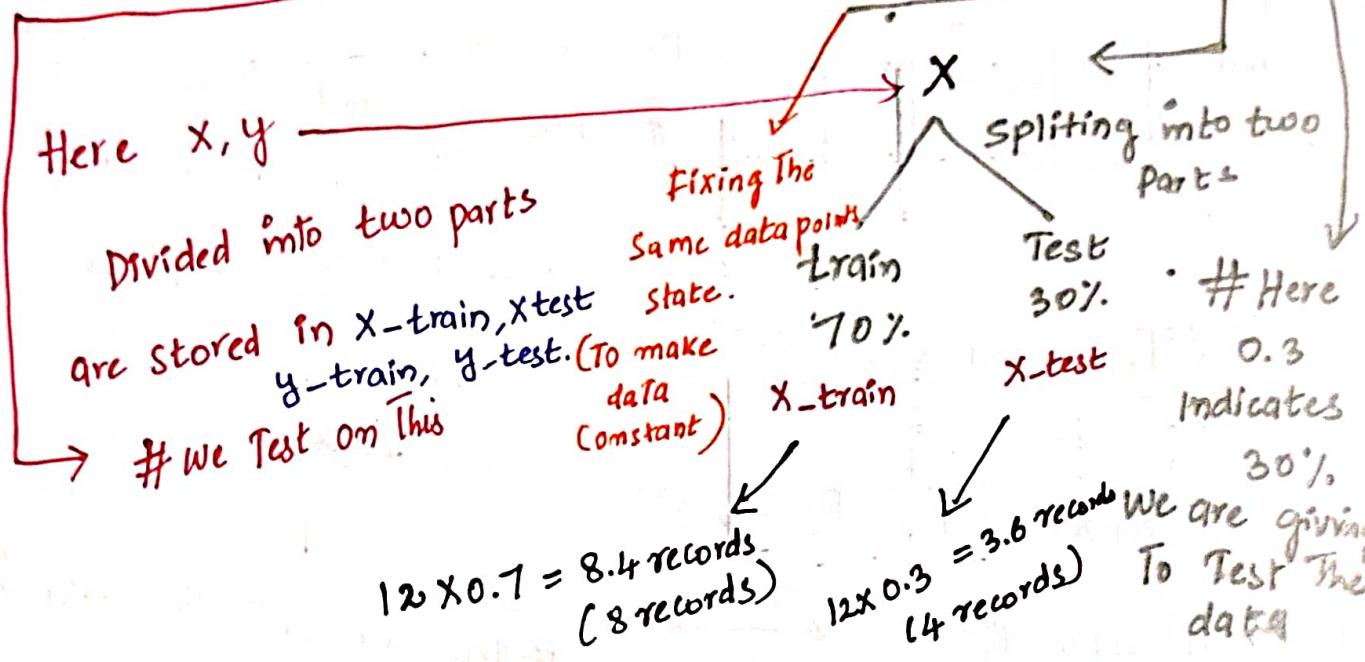
y

Out:	Area	town-chennai	town-Hyderabad
0	2600	1	0
1	3000	1	0
2	3200	1	0
3	2600	1	0
4	2800	0	0
5	3300	0	0
6	3600	0	0
7	2800	0	0
8	2600	0	1
9	2900	0	1
10	3100	0	1
11	3600	0	1

Out:	Price
0	5500000
1	5650000
2	6100000
3	6800000
4	5850000
5	6150000
6	7100000
7	5750000
8	6000000
9	6500000
10	6200000
11	6950000

from Sklearn.model_selection import train_test_split
 ↓ ↓
 Library Module

$x\text{-train}, x\text{-test}, y\text{-train}, y\text{-test}$ = train-test-split ($x, y, \text{test_size}=0.3$, Random_state = 29)



"ML" Algorithm 1

* Same Data

"ML" Algorithm 2

* Same Data

In Order To Compare which algorithm is working well and giving high Accuracy.

We Have To Same data For Every algorithm and compare with all algorithm

code:

```
# X_train, Xtest, y_train, ytest = train_test_split(x, y, test_size=0.3,
Random_state = 29)
```

X-train

[Out] :-

Area	town-chennai	town-hyderabad
7 3600	0	0
6 3300	0	0
1 3000	1	0
0 2600	1	0
8 2500	0	1
2 3200	1	0
3 3600	1	0
5 2800	0	0

X-test

[Out] :-

area	town-chennai	town-hyderabad
9 29	0	1
4 26	0	0
10 31	0	1
11 36	0	1

y-train

[Out] :-

Price
7 7100000
6 6500000
1 5650000
0 5500000
8 5750000
2 6100000
3 6800000
5 6150000

This Heading
is not
mentioned
in my pc
→ Only
These
Values.

y-test

[Out] :-

Price
9 6000000
4 5850000
10 6200000
11 69500000

✓
✓
✓
✓
✓

4:00 AM

Download Part 02 & Part 03
<https://t.me/AIMLDeepThought/665>

Dt: 7/4/22
12:30 PM

Step 3 :- Prediction

Train

$$\text{MODEL} = 10x + 5$$

Substituting x values in -

$$\Rightarrow 10(1) + 5 = 15$$

$$\Rightarrow 10(5) + 5 = 55$$

$$\Rightarrow 10(9) + 5 = 95$$

$$\Rightarrow 10(36) + 5 = 365$$

$$\Rightarrow 10(12) + 5 = 125$$

$$\Rightarrow 10(4) + 5 = 45$$

$$\Rightarrow 10(14) + 5 = 145$$

Example :-

Past Data

Input

	X	Y
0	1	10
1	5	15
2	9	36
3	14	45
4	12	12
5	4	88
6	78	56
7	2	44
8	14	68
9	36	52

Based on
The data
model is
formed (Equ)

If we change
some data (1, 2 records)
our model also
changes.

X	Y	y-Pred-train
1		15
5		55
9		95
36		365
12		125
4		45
14		145

Test

X	Y	y-Pred-test
2		25
14		145
78		785

$$\Rightarrow 10(2) + 5 = 25$$

$$\Rightarrow 10(14) + 5 = 145$$

$$\Rightarrow 10(78) + 5 = 785$$

Step 1 :- Split The Data For Train/test

	X	Y
0	1	10
1	5	15
2	9	36
3	36	52
4	12	12
5	4	88
6	14	68

	X	Y
7	2	44
3	14	45
6	78	56

Test

Train

70%

Selected
Randomly

30%

Step 4 : Evaluation

# Train	Y	y-Pred-train
	10	15
	15	55
	36	95
	52	365
	12	125
	88	45
	68	145

difference b/w
Actual "Y" - Predicted "Y"
= ERROR

$$\begin{aligned} \text{Ex: } & 10 - 15 = -5 \\ & 15 - 55 = 40 \\ & 36 - 95 = 59 \end{aligned}$$

Train_Error

Step 2 :- Modelling

$$\text{MODEL} = 10x + 5$$

Test

Y	y-Pred-test
44	25
45	145
56	785

Test_Error

* Machine Learning Algo steps

→ MODELLING

1. Finding The relation between X & Y of given data

$x\text{-train}$, $y\text{-train}$

relation → "MODEL"

Ex: $y = f(x)$
 $y = ax^2 + bx + c$
 $y = mx + c$
 $y = 10x + 5$ (Equation)

→ PREDICTION

Substituting

the values of "x" in the given Equation and

MODEL calculating

"y" values.

→ EVALUATION

Calculating

the error between Predicted Values and

Actual Values

Good Model

Train-Error = 5% Accuracy = 95%] [±5]

Test-Error = 5% Accuracy = 95%] ↓
with difference

⇒ train accuracy ≈ Test accuracy
with deviation of ±5%. # Good Model.

Overfitting

- train Accuracy = 90%
→ Train Error = 10%.
- test Accuracy = 60%
→ Test Error = 40%.

[Model] [Performs good in training]

But,
Gives [Bad, in testing]
results

train Accuracy > test accuracy.

Underfitting

- train accuracy = 60%
→ Train Error = 40%.
- test accuracy = 90%
→ Test Error = 10%.

[Model] [Performs bad in training]

Data. But

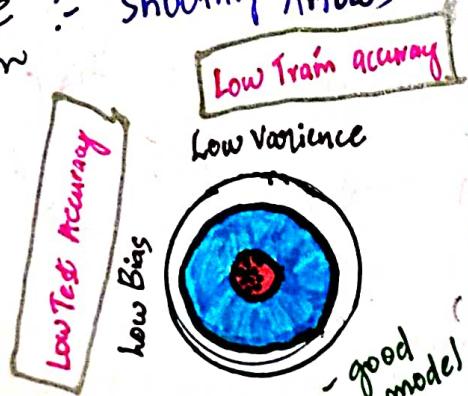
performs [Good in Testing data]

train accuracy < test accuracy.

⇒ Bias/Variance Trade-off

→ difference

Example :- Shooting Arrows



High Variance / High Train accuracy

- Overfitting



- Bad model

Variance = Distance b/w Data Points / Train Error

Bias = close To Target / Test Error

Overfitting
~~~~~\*~~~~~

Variance = train  
Bias = test

→ High variance = High train Accuracy

→ Low bias = Low test Accuracy

# Underfitting  
~~~~~\*~~~~~

→ High bias = High test accuracy

→ Low variance = Low train accuracy

good model
~~~~~\*~~~~~

→ Low Bias = Low Test accuracy

→ Low Variance = Low Train accuracy

✓  
14/22  
5:00 pm

Download Part 02 & Part 03

<https://t.me/AIMLDeepThaught/665>

# Q/A.

## 1. Linear Regression

Over View  
Machine learning

it is used to predict real values.  
(Or) Predicting output variable is "Continuous Data".  
a relationship is established between independent and dependent variable by fitting a best fit line.  
(or) Assuming relation between  $x$  &  $y$  is "Linear".  
The best fit line is known as "regression line".

represented as slope intercept form

$$y = mx + b$$

dependent variable      independent variable

## 2. Logistic Regression:

it is a classification algorithm used to estimate "Discrete Data Values" based on given independent variables.

Set of independent variables lies between [0, and 1] or [Yes/No]. Uses probability.

The sigmoid function.

$$\frac{e^y}{1+e^y}$$

### 3. Decision tree

it is **Supervised learning algorithm**

Used for

**classification**

**problems**

which **works**

for **both**

**dependent variables.**

**Categorical** and **continuous**

In this **algorithm**

"population"

**split**

into **two (or) More homogenous sets.**

Given **Conditions.**

it uses **different Techniques** like

- \* Gini impurity
- \* Information Gain
- \* Chi-Square Entropy

### 4. Random Forest

A **collection** of **Decision trees** is called

as "**Random Forest**". To classify a **new object** based

on its **attributes**, **Each Tree** is **classified** and

Take **majority of votes** for **that class**.

**classifies them.** (**Bagging Technique**)

applicable for both

**Classification**

& **Regression**

## 5. K-Nearest Neighbors

K-Nearest Neighbors is a simple algorithm that stores all available cases and classifies New Cases by a majority vote of its "K-Neighbors". It is measured by distance function.

- \* Manhattan Distance  $|x_2 - x_1| + |y_2 - y_1|$
- \* Euclidean Distance  $\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$

## 6. Naive Bayes Theorem (algorithm)

Conditional probability is based on it is supervised learning algorithm. Which "Bayes theorem" and used for solving classification problems. Bayes theorem is used to

determine probability of a hypothesis with likelihood.

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

\* Naive: it assumes that each input variable is independent.

## 7. SVM (Support Vector Machine)

it is Supervised Machine learning algorithm

Used for both classification & regression

Main objective of SVM is to find a hyperplane in a "N-dimensional space" that distinctly classifies data points.

(or) SVM to find best line in two dimensions for in more than two dimensions

The best hyperplane in our space into classes.

Order to help us separate

\* Linear Separable

\* Non-linear Separable (or) Kernel SVM

## 8. Boosting Techniques

Ensemble learning

(or) boosting Techniques

it is combination of "weak learners". that implies

combination of estimators (models) with an

applied coefficient could act as effective Ensemble

Estimator

summation  
of t=1 to T

Boosting formula :

$$F_T(x) = \sum_{t=1}^T f_t(x) = \alpha_t h(x)$$

learning rate

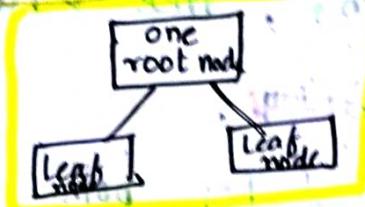
Each model

## 9. AdaBoost Algorithm

Ada Boost (adaptive Boosting) Ensemble method.  
it assigns weights to observations which are incorrectly predicted. Next model works to predict these values correctly before.

Each "weak Learner" is called as "Stump".

- Depth = 1



consists of  
1 root nodes  
2 leaf nodes

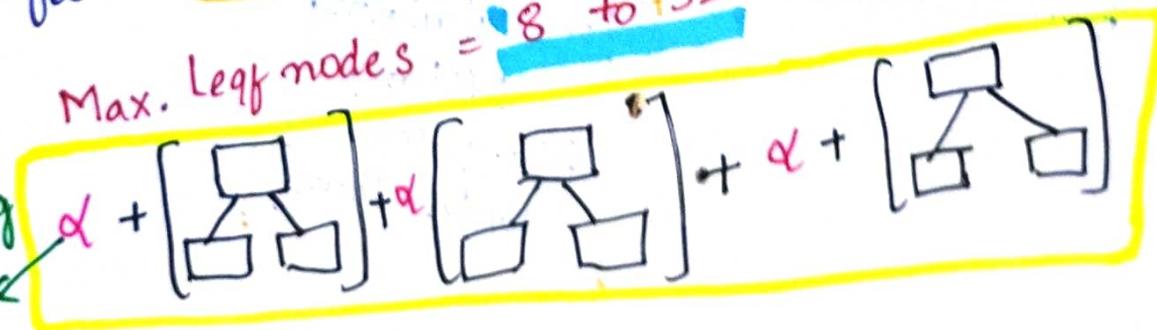
## 10. Gradient Boost

Gradient boost is used to predict both continuous and categorical variables. It is same as boosting techniques. Here we calculate the errors and minimize loss function (single training)

and sequential order.

and trees are adjusted multiple times after

Max. Leaf nodes = 8 to 32



## 11. XGBoost

XGBoost, also called as "Regularized Boosting". Like  $L_1$  &  $L_2$  regularization, it is 10 times faster than Gradient boost. It implements "parallel processing" which reduces overfitting, handles missing data, implements "cross-validation", tree pruning using depth-first approach, and high flexibility. It can be used for both classification and regression problems.

## 12. K-Means algorithm

It is an unsupervised learning algorithm. (or) Unlabel data. K-means algorithm identifies 'k' no. of centroids, and then allocates every data point to nearest cluster, while keeping centroids small as possible, it uses "Distance formula". Same as K-nearest Neighbours.

For calculating distance b/w points & cluster them.

### 13. Hierarchical clustering

it combines Data points which are very close by distance. That Groups similar objects into Groups called clusters. This clustering involves creating clusters that are predetermined ordering.

from Top to bottom.



These are two of hierarchical clustering

- \* Agglomerative (from top to bottom)
- \* Divisive, (from bottom to top)

### DB-Scan clustering

DB-Scan (Density Based spatial Clustering of

application with noise")

DB-Scan clustering locates regions of "high density"

and Separates Outliers.

DB Scan or is to find the neighborhoods of data certain Density threshold.

Points

Exceeds

two parameters

\* R (radius of neighborhood)

\* m (min. no. of neighbours)

Download Part 02 & Part 03

<https://t.me/AIMLDeepThought/665>

## 15. Gradient descent algorithm

[Gradient Descent] is an "Optimization" algorithm for finding a Local minima (minimum Point of a differentiable function.] (or) [it Gradually reducing weights and making Errors as Minimum Value].  
(or) [Ex: By applying different "x" values and identifying the perfect "x" value, till we get error as "0". The zero is called gradient "descent"] minimize Cost Function

$$y = mx + b$$

Update rule  
formula

## 6. Stochastic Gradient Descent

"Global minima" and it has both "Local Minima" and "Local maxima".  
Entire data We divide into parts (or) batches.  
Each part we are identifying. The "local minima" and from that overall identify The "Global minima" is called "Stochastic Gradient Descent".

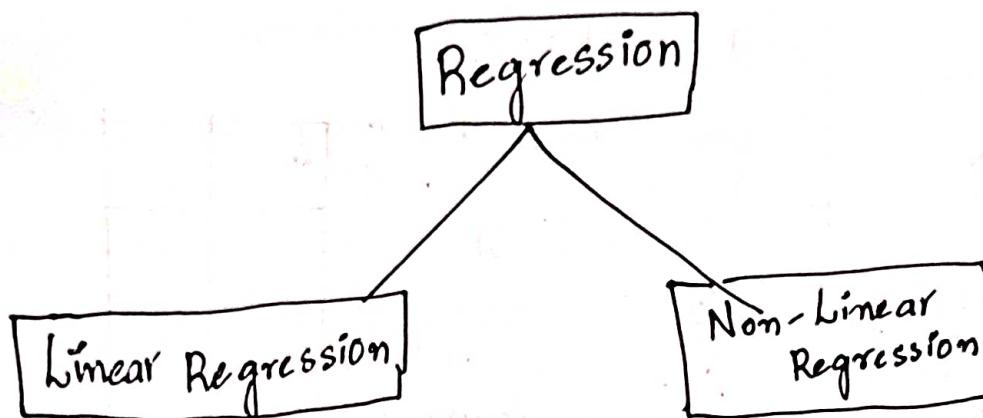
Ex:- State ranker (local Minima)

All india ranker (Global Minima)

7/4/22  
9:30pm

## Regression

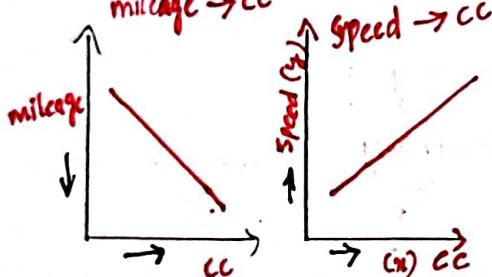
Predicting Output which is Continuous



(Assuming the relation between  $x$  &  $y$ )

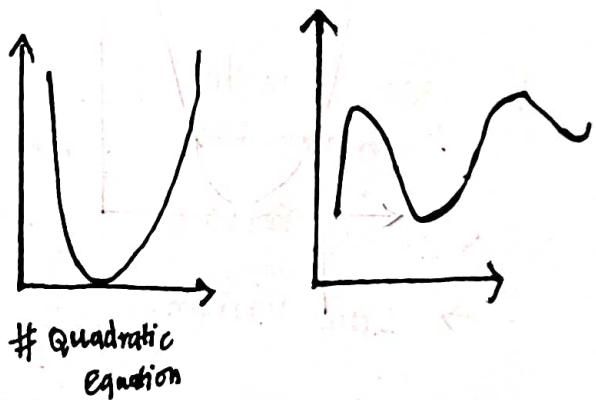
is Linear")

Ex:- In The Form of  $(y = mx + c)$



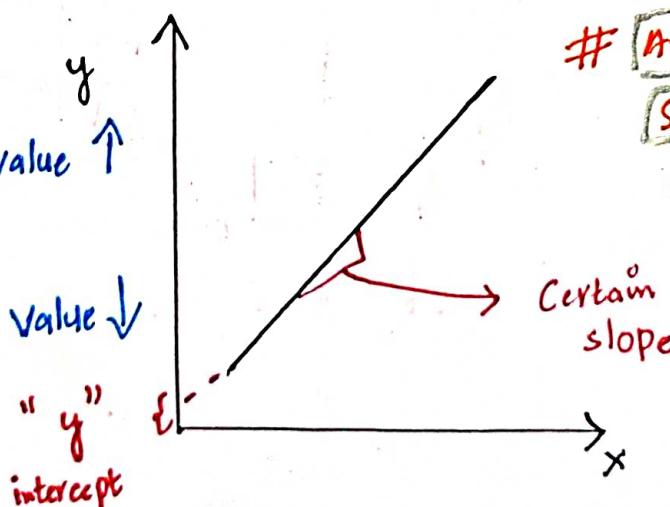
\* mileage decreases ↑ \* cc increases ↑  
\* cc increases ↑ \* speed increases ↑

Any line other than straight line is Non-Linear



#  $x$  value ↑,  $y$  value ↑  
(+ve slope)

#  $x$  value ↑,  $y$  value ↓  
(-ve slope)



# Any relation in terms of Straight Line, it is called as "Linear Regression"

Ex:-

| X | Y    |
|---|------|
| 1 | 15   |
| 2 | 25   |
| 3 | 35   |
| 4 | 45   |
| 5 | 55   |
| 6 | ? 35 |

# In Early Ages, They took "Avg" values.

Like

$$\frac{15 + 25 + 35 + 45 + 55}{5} = 35$$

7 35

8 35

→ They say, Avg Line is Best Fit Line;

## \* Best Fittants

# tips (waiter)

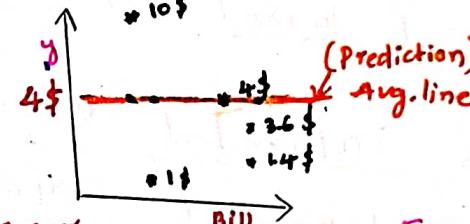
- 1. 10 \$ - Bill amount (not given)
- 2. 1 \$ - so, For only One Variable / column - (Avg)
- 3. 4 \$ -
- 4. 3.6 \$ -
- 5. 1.4 \$ -
- 6 → Avg 4 \$

7 → 4 \$ Every time Answer is 4 \$ (because of Avg line)

# For Single Variable

Best Fit line

Average line



For x value [Predict Va

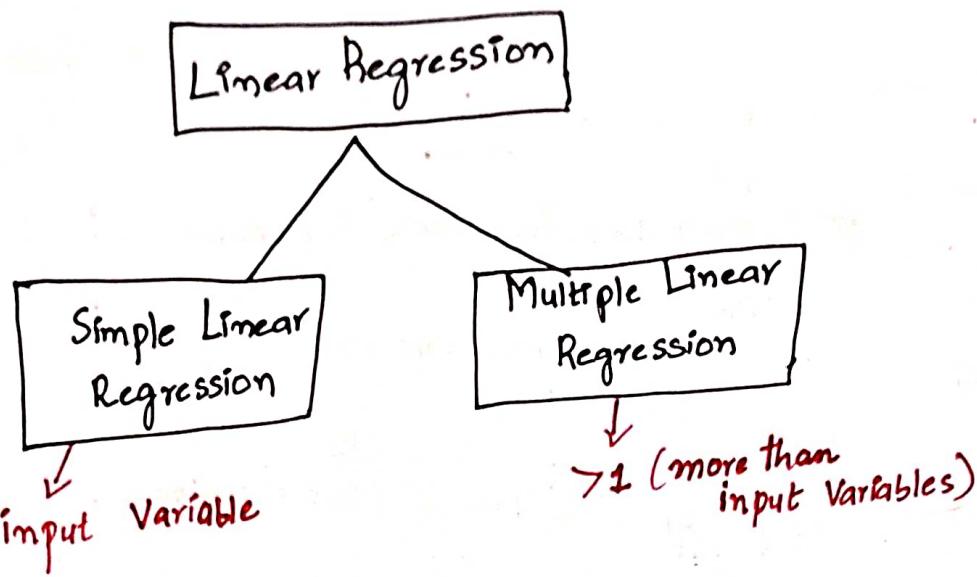
H<sub>0</sub>: Average line is best fit line

H<sub>1</sub>: Regression line > Avg. line.

| Y   | avg.line<br>Y-Prediction | Error $\delta$ | Error $\delta^2$ |
|-----|--------------------------|----------------|------------------|
| 10  | 4                        | 6              | 36               |
| 1   | 4                        | -3             | 9                |
| 4   | 4                        | 0              | 0                |
| 3.6 | 4                        | -0.4           | 0.16             |
| 1.4 | 4                        | -2.6           | 6.76             |
|     |                          | 0              | 51.92            |

Download Part 02 & Part 03

<https://t.me/AIMLDeepThaught/665>



## # Simple Linear Regression :

- In Simple Linear regression, We predict the value of One variable "Y" based on Other variable "X".
- "X" is independent variable (or) input variable (or) exploratory variable  
"y" is dependent variable (or) output variable (or) response variable

Ex:- 

|                |         |         |                                               |
|----------------|---------|---------|-----------------------------------------------|
| Weight         | CC      | Speed   | Mileage                                       |
| indepn<br>dvar | in. var | in. var | → O/P<br>(it depends on weight,<br>CC, Speed) |

Que :- Why it is Simple ?

Because, it Examines

Relationship

between

two

Ans :-

Variables Only.

| X | Y |
|---|---|
|   |   |

$$y = x$$

Que :- Why linear ?

independent

Variable increases

(or Decreases)

Ans :- When the

dependent Variable

increases (or Decreases) in

The dependent Variable

a linear Fashion.

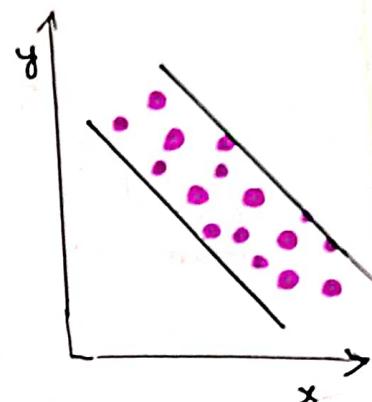
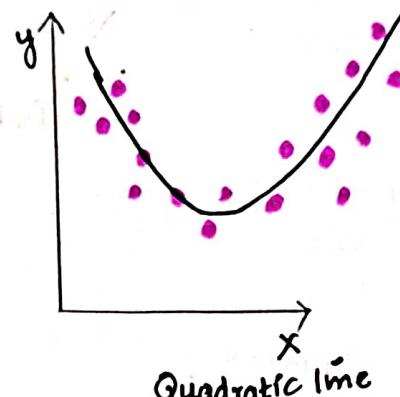
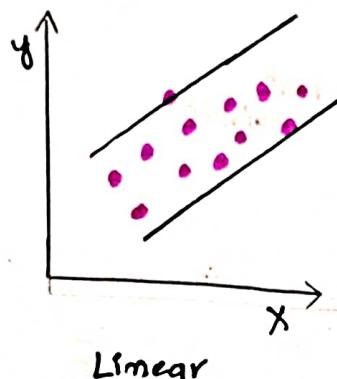
Que: How to check Linear Regression ?

(or) not

Ans:-

By scatter plot

Linearity  
direction



As "x" value increasing  
"y" value also increasing

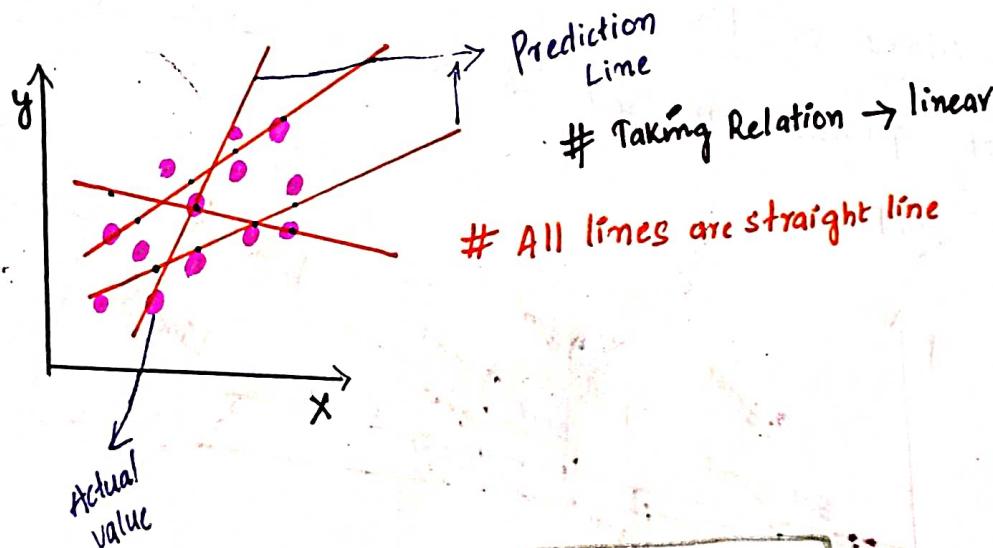
"Positive Direction"

As "x" value increasing  
"y" value also decreases

"Negative direction"

Que:- Which Line to be Considered ?

Ans:-



Que:- Different notations ? But Same Concept ?

Ans:-

$$y = mx + c \quad \begin{matrix} \text{slope} \\ \text{"y" intercept} \end{matrix}$$

$$y = an + b \quad \begin{matrix} \text{slope} \\ \text{"y" intercept} \end{matrix}$$

$$y = b_0 + b_1 n \quad \begin{matrix} \text{slope} \\ \text{"y" intercept} \end{matrix}$$

$$y = \beta_0 + \beta_1 n \quad \begin{matrix} \text{slope} \\ \text{"y" intercept} \end{matrix}$$

$$\quad \begin{matrix} \text{slope} \\ \text{"y" intercept} \end{matrix}$$

Ques: What is Standard notation?

Ans:

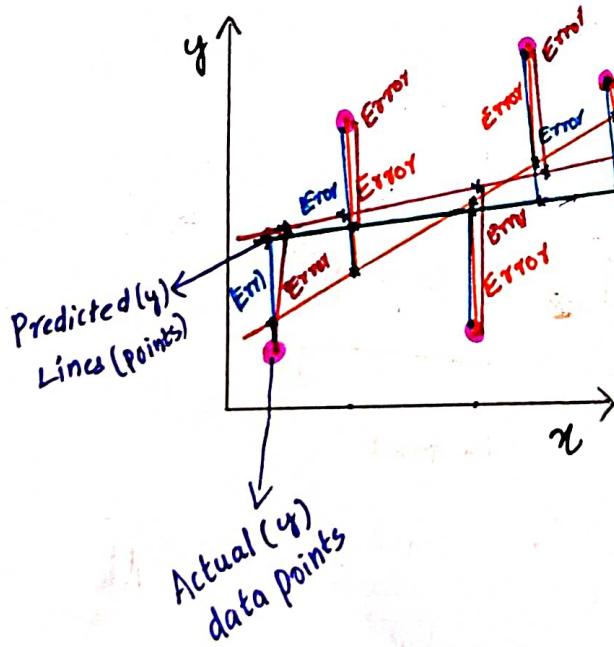
$$y = \beta_0 + \beta_1 x_1$$

Annotations:

- (dependent variable)
- "y intercept" value
- Constant coefficient
- Independent variable
- slope value

# Ordinary Least squares :

- Least squares fitting is a way to find The best fit Curve
- Least squares fitting is a way to find The best fit Curve
  - Line for a set of points
  - Sum of Squares of Residuals Errors are used to estimate the best fit curve (or) Line
  - Least squares method is used to obtain The coefficient of "m" and "b"



# best fit line?  
Where The Sum of Squares of Error is Minimum

# distance b/w Actual data point - Predicted data point [Error]

# Square The errors [each error do square]  
# after Squaring Error, Then Sum Error

Que:- What is SSE?

Ans:- SSE  $\Rightarrow$  Some of squares of Error

Denoted by  $\sum [y - \hat{y}]^2$  squaring them  
= SSE<sub>min</sub>

Sum  
Actual value  
Predicted value

$\sum [y - \hat{y}]^2$   $\xrightarrow{\min}$  This value Should be Minimum  
\* you Have identify the line such a way  $\sum [y - \hat{y}]^2$   $\xrightarrow{\min}$

Que :- How to identify The Exact line?

Ans:- Explanation:- OF Calculus

Minimum Point :  $y = x^n$  (what is Least "y" value)

$y = 0$  # if 0.00001 also it get squared  
# so, "0" is Least y value

how it is identified

$\frac{dy}{dx} \text{ at } x=0$   $\Rightarrow$   $\frac{dy}{dx} \text{ at } x=0 \Rightarrow \frac{d(x^n)}{dx} \text{ at } x=0 \Rightarrow 2x \Rightarrow 2(0) = 0$

Ex:-  $y = 4x^2 + 5 \rightarrow 4x^2 = 8$

$\frac{dy}{dx} \text{ at } x=0 \Rightarrow 8x + 0 \Rightarrow 8(0) + 0 \Rightarrow 0$

\*  $\frac{d}{dx}(4x^2) = 2 \times 4 = [8x]$

\*  $\frac{d}{dx}(5) = 0$

$$\# \text{ Minimum Value} = \frac{dy}{dx} \text{ at } x=0$$

$$\# SSE_{\min} = \epsilon [y - \hat{y}]^2$$

So, in order to get minimum value for S.S.E

calculate

$$\frac{d [\epsilon [y - \hat{y}]^2]}{dx} \text{ at } x=0$$

$$\hat{y} = \beta_0 + \beta_1 x$$

$$\frac{d}{dx} \epsilon (y - \beta_0 + \beta_1 x)$$

Expand This Equation ↑

$$\beta_1 = \frac{\sum_{i=1}^n (x_i - \bar{x})(y - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2}$$

$$\beta_0 = \bar{y} - \beta_1 \bar{x}$$

"y" Mean      "x" Mean

Download Part 02 & Part 03  
<https://t.me/AIMLDeepThaught/665>

Example :

$H_0: \text{Avg. line} \leq \text{Regression Line}$   
 $H_1: \text{Regression line} < \text{Avg. line}$

| X  | Y  |
|----|----|
| 4  | 11 |
| 6  | 4  |
| 7  | 6  |
| 8  | 5  |
| 12 | 8  |

# identifying best fit line for these values

By using

$$y = \beta_0 + \beta_1 x \quad [\text{Linear line}]$$

not in  $y = \log x, e^x, ax^n, \beta x^{-n}$

For Average Fit line

For  $2+5x$  Equation

For  $5+4x$  Equation

| X  | Y  | y-Pred avg | Error | Error <sup>2</sup> |
|----|----|------------|-------|--------------------|
| 4  | 11 | 6.8        | 4.2   | 17.64              |
| 6  | 4  | 6.8        | -2.8  | 7.84               |
| 7  | 6  | 6.8        | -0.8  | 0.64               |
| 8  | 5  | 6.8        | -1.8  | 3.24               |
| 12 | 8  | 6.8        | 1.2   | 1.44               |
|    |    | 30.8       |       |                    |

Random Value  
For  $2+5x$  (Eq)

SSE

| X  | Y  | y-Pred | Error | Error <sup>2</sup> |
|----|----|--------|-------|--------------------|
| 4  | 11 | 22     | -11   | 121                |
| 6  | 4  | 32     | -28   | 784                |
| 7  | 6  | 37     | -31   | 961                |
| 8  | 5  | 42     | -37   | 1369               |
| 12 | 8  | 62     | -54   | 2916               |
|    |    | 6151   |       |                    |

For  $5+4x$  Equation

SSE

| X  | Y  | y-Pred | Error | Error <sup>2</sup> |
|----|----|--------|-------|--------------------|
| 4  | 11 | 21     | -10   | 100                |
| 6  | 4  | 29     | -25   | 625                |
| 7  | 6  | 33     | -27   | 729                |
| 8  | 5  | 37     | -32   | 1024               |
| 12 | 8  | 53     | -45   | 2025               |
|    |    | 4503   |       |                    |

SSE

$$\text{* Avg} = \frac{11+4+6+5+8}{5} = 6.8$$

$$\begin{aligned} \text{* Error} &\Rightarrow 11 - 6.8 = 4.2 \\ (\text{Actual} - \text{Predicted}) &= 4 - 6.8 = -2.8 \\ y &\quad y_{\text{pred}} = 6 - 6.8 = -0.8 \\ &\quad 5 - 6.8 = -1.8 \\ &\quad 8 - 6.8 = 1.2 \end{aligned}$$

$$\begin{aligned} \text{* Error}^2 &= (4.2)^2 = 17.64 \\ (2.8)^2 &= 7.84 \\ (0.8)^2 &= 0.64 \\ (1.8)^2 &= 3.24 \\ (1.2)^2 &= 1.44 \end{aligned}$$

$$\begin{aligned} \text{* Some of square of Error} &= 17 + 7.84 + 0.64 + 3.24 + 1.44 \\ &= 30.8 \end{aligned}$$

For,  $2+5x$

$$\begin{aligned} \Rightarrow y_{\text{pred}} &\Rightarrow 2+5(4) = 22 \\ &\quad 2+5(6) = 32 \\ &\quad 2+5(7) = 37 \\ &\quad 2+5(8) = 42 \\ &\quad 2+12(12) = 62 \end{aligned}$$

⇒ Error

$$(y - \hat{y}) = 11 - 22 = -11$$

$$4 - 32 = -28$$

$$6 - 37 = -31$$

$$5 - 42 = -37$$

$$8 - 62 = -54$$

$$\begin{aligned} \Rightarrow \text{Error}^2 &= (-11)^2 = 121 \\ &\quad (-28)^2 = 784 \\ &\quad (-31)^2 = 961 \\ &\quad (-37)^2 = 1369 \\ &\quad (-54)^2 = 2916 \end{aligned}$$

$$= 6151$$

For,  $5+4x$

$$\begin{aligned} \Rightarrow y_{\text{pred}} &\Rightarrow 5+4(4) = 29 \\ &\quad 5+4(6) = 33 \\ &\quad 5+4(7) = 37 \\ &\quad 5+4(8) = 41 \\ &\quad 5+4(12) = 53 \end{aligned}$$

$$\begin{aligned} \text{* Error} &= 21 - 29 \\ &= 4 - 29 \\ &= 6 - 33 \\ &= 5 - 37 \\ &= 8 - 53 \end{aligned}$$

$$\begin{aligned} \text{* Error}^2 &= (10)^2 = 100 \\ (25)^2 &= 625 \\ (27)^2 &= 729 \\ (32)^2 &= 1024 \\ (45)^2 &= 2025 \end{aligned}$$

$$\begin{aligned} \text{SSE} &= 100 + 625 + 729 + 1024 + 2025 \\ &= 2025 \end{aligned}$$

$$\Rightarrow SSE = 121 + 784 + 961 + 1369 + 2916$$

# # For Regression line

$$y = \beta_0 + \beta_1 x$$

| X  | Y  | $x - \bar{x}_{\text{mean}}$ | $y - \bar{y}$ | $\bar{x} * \bar{y}$ | $(x - \bar{x})^2$ |
|----|----|-----------------------------|---------------|---------------------|-------------------|
| 4  | 11 | -3.4                        | 4.2           | -14.28              | 11.56             |
| 6  | 4  | -1.4                        | -2.8          | 3.92                | 1.96              |
| 7  | 6  | -0.4                        | -0.8          | 0.32                | 0.16              |
| 8  | 5  | 0.6                         | -1.8          | -1.08               | 0.36              |
| 12 | 8  | 4.6                         | 1.2           | 5.52                | 21.16             |
|    |    |                             |               | $\Sigma E = -5.6$   | $E(35.2)$         |

$$\therefore \beta_0 = \bar{y} - \beta_1 (\bar{x})$$

$$\beta_1 = \frac{\Sigma (x - \bar{x})(y - \bar{y})}{\Sigma (x - \bar{x})^2}$$

X Avg: 7.4  
Y Avg: 6.8

\* Avg. of X =  $\frac{4+6+7+8+12}{5} = 7.4$

\* Avg. of Y =  $\frac{11+4+6+5+8}{5} = 6.8$

\*  $x - \bar{x}_{(\text{mean})} = 4 - 7.4 = -3.4$   
 $6 - 7.4 = -1.4$   
 $7 - 7.4 = -0.4$   
 $8 - 7.4 = 0.6$   
 $12 - 7.4 = 4.6$

\*  $y - \bar{y}_{(\text{mean})} = 11 - 6.8 = 4.2$   
 $4 - 6.8 = -2.8$   
 $6 - 6.8 = -0.8$   
 $5 - 6.8 = -1.8$   
 $8 - 6.8 = 1.2$

\*  $\bar{x}_{\text{mean}} * \bar{y}_{\text{mean}} =$   
 $-3.4 * 4.2 = -14.28$   
 $-1.4 * -2.8 = 3.92$   
 $-0.4 * -0.8 = 0.32$   
 $+0.6 * -1.8 = -1.08$   
 $4.6 * 1.2 = 5.52$

\*  $\Sigma \bar{x}_{\text{mean}} * \bar{y}_{\text{mean}}$

$$= -14.28 + 3.92 + 0.32 - 1.08 + 5.52 = -5.6$$

\*  $(x - \bar{x})^2$

$$\begin{aligned} (3.4)^2 &= 11.56 \\ (-1.4)^2 &= 1.96 \\ (-0.4)^2 &= 0.16 \\ (0.6)^2 &= 0.36 \\ (4.6)^2 &= 21.16 \end{aligned}$$

\*  $\Sigma (x - \bar{x})^2$

$$= 11.56 + 1.96 + 0.16 + 0.36 + 21.16 = 35.2$$

$$\beta_1 = \frac{\Sigma (x - \bar{x})(y - \bar{y})}{\Sigma (x - \bar{x})^2} \Rightarrow \frac{-5.6}{35.2} = 0.15909$$

Slope

$$\beta_0 = \bar{y} - \beta_1 (\bar{x}) = 6.8 - (0.159)(7.4)$$

$$6.8 - (0.159) = 7.977$$

Now, The equation is

Intercept

$$y = \beta_0 + \beta_1 x$$

0.15909 [n]

$$y = 7.977$$

# Here

$$y = 7.977273 - 0.159[x]$$

Equation.

| X  | Y  | y <sub>pred</sub> | Err   | Error <sup>2</sup> |
|----|----|-------------------|-------|--------------------|
| 4  | 11 | 7.34              | 3.66  | 13.39              |
| 6  | 4  | 7.02              | -3.02 | 9.12               |
| 7  | 6  | 6.86              | -0.86 | 0.73               |
| 8  | 5  | 6.70              | -1.7  | 2.89               |
| 12 | 8  | 6.06              | 1.94  | 3.76               |
|    |    |                   | 29.89 | SSE                |

$$\text{Equation: } 7.977 - 0.159[x]$$

$$7.977 - 0.159(4) = 7.34$$

$$7.977 - 0.159(6) = 7.02$$

$$7.977 - 0.159(7) = 6.86$$

$$7.977 - 0.159(8) = 6.70$$

$$7.977 - 0.159(12) = 6.06$$

# Comparing with  
Avg. line and  
Other Equations

$$y = 7.977 - 0.159[x]$$

Best fit line.

[SSE] min

Error

$$11 - 7.34 = 3.66$$

$$4 - 7.02 = -3.02$$

$$6 - 6.86 = -0.86$$

$$5 - 6.70 = -1.7$$

$$8 - 6.06 = 1.94$$

Error<sup>2</sup>

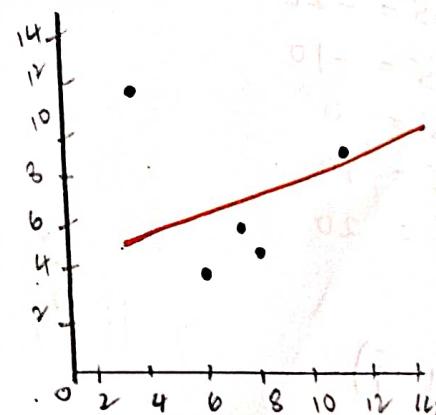
$$(3.66)^2 = 13.39$$

$$(-3.02)^2 = 9.12$$

$$(-0.86)^2 = 0.73$$

$$(-1.7)^2 = 2.89$$

$$(1.94)^2 = 3.76$$



$$\begin{aligned} \text{SSE} &= 13.39 + 9.12 + 0.73 + 2.89 + 3.76 \\ &= 29.89 \end{aligned}$$

For, Average  $\rightarrow$

For,  $10x+5$

| X | Y  | $\bar{y}$ | Error | $c_{xy} \times 10^2$ |
|---|----|-----------|-------|----------------------|
| 1 | 15 | 35        | +20   | 400                  |
| 2 | 25 | 35        | +10   | 100                  |
| 3 | 35 | 35        | 0     | 0                    |
| 4 | 45 | 35        | -10   | 100                  |
| 5 | 55 | 35        | -20   | 400                  |

EX [For Intercept]

$$\therefore R^2 = 1 - \frac{0_{\text{reg}}}{1000_{\text{tot}}}$$

$$R^2 = 1$$

Maximum value

| X | Y  | $x-\bar{x}$ | $y-\bar{y}$ | $(x-\bar{x})(y-\bar{y})$ | $(x-\bar{x})^2$ | $\bar{y}$ | Error | Error |
|---|----|-------------|-------------|--------------------------|-----------------|-----------|-------|-------|
| 1 | 15 | -2          | -20         | 40                       | -4              | 15        | 0     | 0     |
| 2 | 25 | -1          | -10         | 10                       | -1              | 25        | 0     | 0     |
| 3 | 35 | 0           | 0           | 0                        | 0               | 35        | 0     | 0     |
| 4 | 45 | 1           | 10          | 10                       | 1               | 45        | 0     | 0     |
| 5 | 55 | 2           | 20          | 40                       | 4               | 55        | 0     | 0     |

$$\bar{x} = 3 \quad \bar{y} = 35$$

$$X \text{ Avg} = \frac{1+2+3+4+5}{5} = 3$$

$$Y \text{ Avg} = \frac{15+25+35+45+55}{5} = 35$$

$$* x-\bar{x} = 1-3 = -2$$

$$2-3 = -1$$

$$3-3 = 0$$

$$4-3 = 1$$

$$5-3 = 2$$

$$* y-\bar{y} = 15-35 = -20$$

$$25-35 = -10$$

$$35-35 = 0$$

$$45-35 = 10$$

$$55-35 = 20$$

$$* (x-\bar{x})(y-\bar{y})$$

$$= -2 \times (-20) = 40$$

$$-1 \times (-10) = 10$$

$$1 \times 10 = 10$$

$$2 \times 20 = 40$$

$$* \varepsilon(x-\bar{x})(y-\bar{y})$$

$$= 40+10+0+10+40 = 100$$

$$* (x-\bar{x})^2 \# \text{ For squaring don't consider } (+, -)$$

$$= (-2)^2 = +4$$

$$(-1)^2 = +1$$

$$(1)^2 = 1$$

$$(2)^2 = 4$$

$$* \varepsilon(x-\bar{x})^2 = -4-1+1+4 = 10$$

$$\beta_0 = \bar{y} - \beta_1 \bar{x}$$

$$\beta_1 = \frac{\varepsilon(x-\bar{x})(y-\bar{y})}{\varepsilon(x-\bar{x})^2}$$

$$\# \beta_1 = \frac{10}{10} = 10 \rightarrow \text{slope}$$

$$\beta_0 = 35 - 10 \cdot 3 = 5 \rightarrow y \text{ intercept}$$

Equation is

$$y = \beta_0 + \beta_1 x$$

$$y = 5 + 10x$$

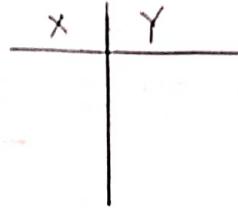
✓  
X 8/4/22  
3:30 am

8/4/22  
3:00 PM

## \* BIVARIATE STATISTICS \*

⇒ Simple linear regression :-

it is **2 Sample "T" test**



$H_0$  : Avg. line  $\geq$  Regression line  
 $H_1$  : Regression line  $>$  Avg. line.

Q: How can this be proved statistically?

A: Where

(SSE)<sub>Regression line</sub>

< (SSE)<sub>Average line</sub>

Ex:- 120

30 <

⇒ Multiple linear regression :-

| $x_1$ | $x_2$ | $x_3$ | $x_4$ | $y$ |
|-------|-------|-------|-------|-----|
|       |       |       |       |     |
|       |       |       |       |     |
|       |       |       |       |     |
|       |       |       |       |     |

Where we have multiple "x" values and only single "y". We use **"ANOVA"** Statistical Test ✓ applied when output variable is "Continuous".

Q: What is the hypothesis Test apply for Regression?

A: **Anova Test**

( $> 2$  samples) Apply for More than 2 samples

A:

if we apply Only for 2 samples

it is **"2 sample T Test"**

\* **"Correlation"** is high

gives "good" results

→ **Linear Regression**

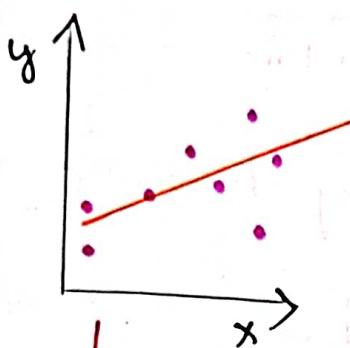
" Relation between two variables  
Modulus of "r" any value  
 $|r| = \pm$

$|r| \geq 0.8$  (strong) correlation  
 $0.5 < |r| < 0.8$  (moderate)  
 $|r| < 0.5$  (weak)

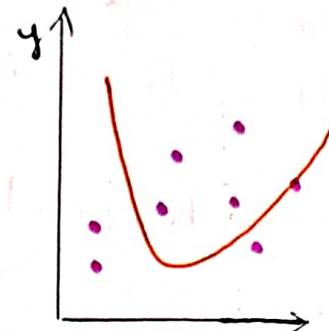
# The value of "One Variable", is a function of "Other Variable"

# The value of  $y$ , is a function of  $x$ :

$$y = f(x)$$



$$\therefore y = mx + c$$

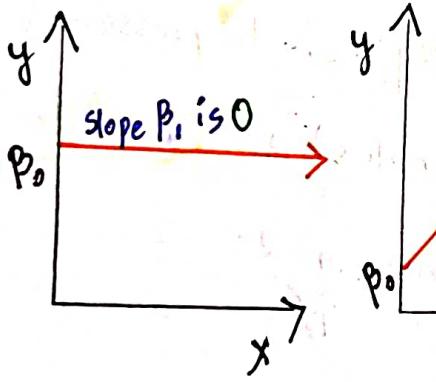


$$y = ax^2 + bx + c$$

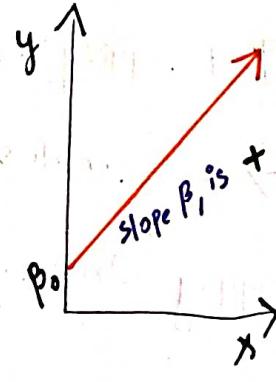
# The value of dependent variable, is function of independent Variable

$$\Rightarrow E(y) = \beta_0 + \beta_1 x \rightarrow \text{"slope"} \quad \downarrow \text{"y" intercept}$$

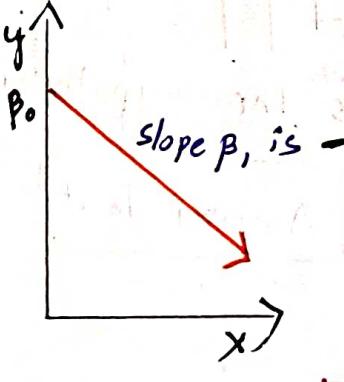
Estimated value



$$E(y) = \beta_0 + 0(x)$$



$$E(y) = \beta_0 + \beta_1 x$$



$$E(y) = \beta_0 - \beta_1 x$$

Download Part 02 & Part 03

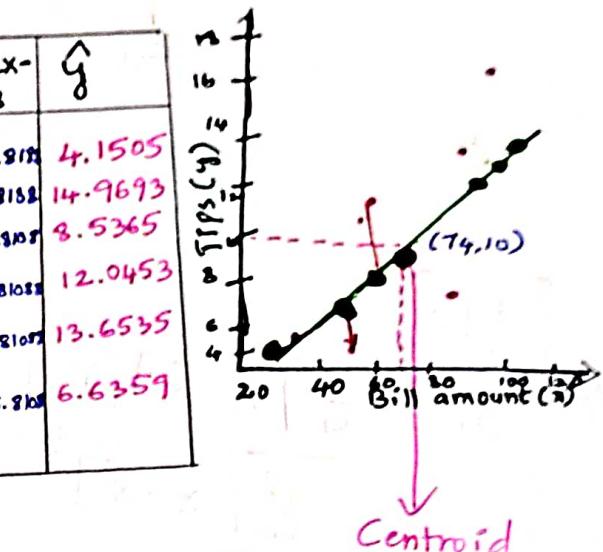
<https://t.me/AIMLDeepThaught/665>

## \* Calculations

| X              | Y  | $X - \bar{X}$  | $Y - \bar{Y}$ | $(X - \bar{X})(Y - \bar{Y})$ | $(X - \bar{X})^2$ | $y = 0.1462x - 0.8188$ | $\hat{y}$ |
|----------------|----|----------------|---------------|------------------------------|-------------------|------------------------|-----------|
| 34             | 5  | -40            | -5            | 200                          | 1600              | $0.1462[34] - 0.8188$  | 4.1505    |
| 108            | 17 | 34             | 7             | 238                          | 1156              | $0.1462[108] - 0.8188$ | 14.9693   |
| 64             | 11 | -10            | 1             | -10                          | 100               | $0.1462[64] - 0.8188$  | 3.5365    |
| 88             | 8  | 14             | -2            | -28                          | 196               | $0.1462[88] - 0.8188$  | 12.0453   |
| 99             | 14 | 25             | 4             | 100                          | 625               | $0.1462[99] - 0.8188$  | 13.6535   |
| 51             | 5  | -23            | -5            | 115                          | 529               | $0.1462[51] - 0.8188$  | 6.6359    |
|                |    |                |               |                              |                   |                        |           |
| $\bar{X} = 74$ |    | $\bar{Y} = 10$ |               | $\sum E = 615$               | $\sum F = 4206$   |                        |           |

$$y = 0.1462[x] - 0.8188$$

Equation



The line should pass through centroid.

$$\text{Avg. of } X := \frac{34 + 108 + 64 + 88 + 99 + 51}{6} = \bar{X} = 74$$

$$\text{Avg of } Y := \frac{5 + 17 + 11 + 8 + 14 + 5}{6} = \bar{Y} = 10$$

How This Equation is formed?

Descriptive Statistics / Centroid

$$\bar{X}, \bar{Y} = 74, 10$$

$$\text{Variance } (X) = \frac{\sum (x - \bar{x})^2}{(n-1)} \rightarrow s^2$$

\* Line  $[\hat{y} = b_0 + b_1 x]$  min

Variance(X) can be written as.

$$\frac{\sum (x - \bar{x})(x - \bar{x})}{(n-1)}$$

$$b_1 = \frac{\sum [x_i - \bar{x}][y_i - \bar{y}]}{\sum [x_i - \bar{x}]^2}$$

Covariance (X, Y) written as

$$\frac{\sum (x - \bar{x})(y - \bar{y})}{(n-1)}$$

Substituting & Equating

$$\text{cov}(X, Y) = \frac{\sum [x - \bar{x}][y - \bar{y}]}{n-1}$$

Slope =

$$\frac{\sum [x - \bar{x}][y - \bar{y}]}{\sum [x - \bar{x}]^2}$$

(S.S.E)

\* Sum of Squares of Error =  $\sum [y - \hat{y}]^2$

$$= \sum [y - (\beta_0 + \beta_1 x)]^2$$

$$= \sum [y - \bar{y} - \beta_1 x]^2$$

↓ should be minimum

# Slope  $b_1 = \frac{\sum [(x - \bar{x})(y - \bar{y})]}{\sum [x - \bar{x}]^2}$

$$b_1 = \frac{615}{4206}$$

$$b_1 = 0.1462$$

# y intercept  $\Rightarrow b_0 = \bar{y} - b_1 \bar{x}$

$$b_0 = 10 - 0.1462 [74]$$

$$b_0 = -0.8188$$

Final Equation  $\Rightarrow y = b_0 + b_1 x$

$$y = -0.8188 + 0.1462 x$$

$$y = \beta_0 + \beta_1 x$$

(or)

$$y = 0.1462 x + (-0.8188)$$

\*\*\* "Accuracy" called as  
"Coefficient of Determination"

$$R^2 = 1 - \frac{SSE_{\text{reg}}}{SST_{\text{Avg}}}$$

"Determining The variation of "y"  
with respect To variation of "x"  
→ As "x" value changes, how  
"y" value changes.

(or)  
"y" variable Explained by

"x" variable.

Ques: What is Max. value of  $R^2$   
Ans:  $R^2 = 1$  [Max. value]

$$(SSE)_{\text{Reg}} = 0$$

$$R^2 = -\infty$$

$\therefore R^2$  (Wors mod)

\*\*\*\* Range of  $R^2$ .

$[\alpha, 1]$

Infinite

# "Regression Squared Error"

| $\hat{y}$ | Error<br>$y - \hat{y}$ | Square Error<br>$(y - \hat{y})^2$ |
|-----------|------------------------|-----------------------------------|
| 4.1505    | (5.4 - 4.1505)         | (0.8495) <sup>2</sup>             |
| 14.9693   | 2.0307                 | 4.1237                            |
| 8.5365    | 2.4635                 | 6.0688                            |
| 12.0453   | -4.0453                | 16.3645                           |
| 13.6535   | 0.3465                 | 0.1201                            |
| 6.6359    | -1.6359                | 2.6762                            |

$$SSE \Rightarrow \sum = 30.075$$

- \* Avg. line = 120
  - \* Regression line = 30.075
- difference between these are  $\frac{30}{120} \Rightarrow 75\%$ .

Coefficient of Determination:

(Or)

$$R^2 = 1 - \frac{SSE(\text{Reg})}{SST(\text{Avg})}$$

$\sum [y - \hat{y}]^2$   
SSE [Sum of squares of Error (Regression line)]  
SST [Sum of squares of Error (Average line)]

$$R^2 = 1 - \frac{\sum [y - \hat{y}]^2_{\text{reg}}}{\sum [y - \bar{y}]^2_{\text{Avg}}}$$

$$R^2 = 1 - \frac{30}{120^4}$$

$$R^2 = \frac{4-1}{4} = \frac{3}{4} \Rightarrow 0.75 \Rightarrow 75\% \text{ accuracy}$$

enf  
8/21/22  
4:30 PM

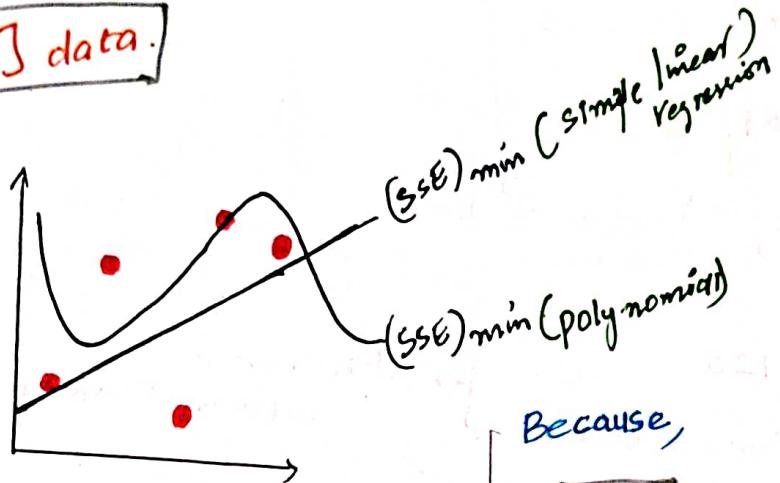
Download Part 02 & Part 02  
<https://t.me/AIMLDeepThaught/665>

Dr. - 9/4/22  
3:00 PM

## Evaluation Metrics in Regression

After Model Fitting, we would like to assess The Performance of model by Comparing model Predictions to actual

[True] data:



1. **MAE** [Mean Absolute Error]
 

bad

$\downarrow$   
differentiation  
 $\downarrow$   
squaring of units

$$\frac{\sum |y - \hat{y}|}{n}$$
2. **MSE** [Mean Squared Error]
 
$$\frac{\sum [y - \hat{y}]^2}{n}$$

Because,

$$\text{MAE} \rightarrow \frac{d[|x|]}{dx}$$

$$\frac{d[|x|]}{dx} \underset{\text{at } x=0}{\rightarrow} \frac{x}{|x|} = \frac{0}{0} \Rightarrow \infty$$

↓  
Infinite

$$\text{MSE} \rightarrow \frac{dx^2}{dx}$$

$$\frac{dx^2}{dx} \underset{\text{at } x=0}{=} 2x$$

Example milage of a car

| $y$  | $\hat{y}$ | $y^2$ |
|------|-----------|-------|
| 18.2 | 18.9      | 0.7   |

Here  
18.2 Kmph  
18.9 Kmph

$$\text{Error} = (0.7)^2 = 0.49 \frac{\text{Km}^2}{\text{h}^2}$$

If we square it

Squaring  
Solves the  
problem of  
units

$$\frac{d[\ ]}{dx} \text{ at } x=0$$

$x$  is only notation

$x^2$ ,  $z^2$ ,  $ab^2$ ,  $\frac{dc}{dy}$

Ques: What are Evaluation Metrics used for Regression?

Ans: We have 1. Mean Absolute Error (MAE)  
and 2. Mean square error [MSE].

Ques  $\Rightarrow$  Mean absolute Error (MAE) Where, we are going to use?

Ans: MAE is "Absolute Values of Mean" is called mean absolute Error.

But, problem with  $\frac{d}{dx}$  of  $(\text{Mod})|x|$  is infinite

is  $\frac{x}{|x|}$  and at  $x=0$ , The answer is "MAE"

So, we are not going to consider

Ques  $\Rightarrow$  Mean Square Error (MSE) with calculating of  $\frac{d}{dx}$  of  $x^n$

Ans: Problem with MSE is we will get some value of  $2x$ . But, by squaring the Error. Units are also squaring up it gives wrong prediction. So, we not going to consider "MSE".

And Solution For This is :-

$$\text{RMSE} = \sqrt{\frac{\sum (y - \hat{y})^2}{N}}$$

Same as in Variance

We take Standard deviation

"Root Mean Square Error"

Q4c :- What is "Root Mean Square Error" RMSE?

A :- "RMSE" represents standard deviation of residuals [Errors] difference between model predictions and True values (Training data)

\* RMSE can be easily interpreted compared To MSE because RMSE Units match units of Output.

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n [y - \hat{y}]^2}$$

(or)

$$\sqrt{\frac{\sum [y - \hat{y}]^2}{n}}$$

Example

Squaring of Errors

Converting negative values to positive values by adding 11

| X | $\hat{Y}$ | Error | MAE | MSE  |
|---|-----------|-------|-----|------|
| 1 | 1.4       | 0.4   | 0.4 | 0.16 |
| 2 | 2.2       | 0.2   | 0.2 | 0.04 |
| 3 | 3.8       | 0.8   | 0.8 | 0.64 |
| 4 | 4.1       | 0.1   | 0.1 | 0.01 |
| 5 | 5.6       | 0.6   | 0.6 | 0.36 |

$$\frac{\sum E}{n} = 0.42$$

$$\frac{\sum E^2}{n} = 0.242$$

Root Mean Square Error  
 $= \sqrt{0.242}$

⇒ Mean percentage Error [MPE]

$$MPE = \frac{100\%}{n} \sum_{i=1}^n (y_i - \hat{y}_i) / y_i$$

In this we calculate Percentage of Error. But we don't consider "Positive" or "Negative" value, without absolute operation.

⇒ Mean absolute Percentage Error [MAPE]

In This "MAPE" we convert Every value To positive by applying "absolute" to the Equation.

$$MAPE = \frac{100\%}{n} \sum_{i=1}^n |y_i - \hat{y}_i| / y_i$$

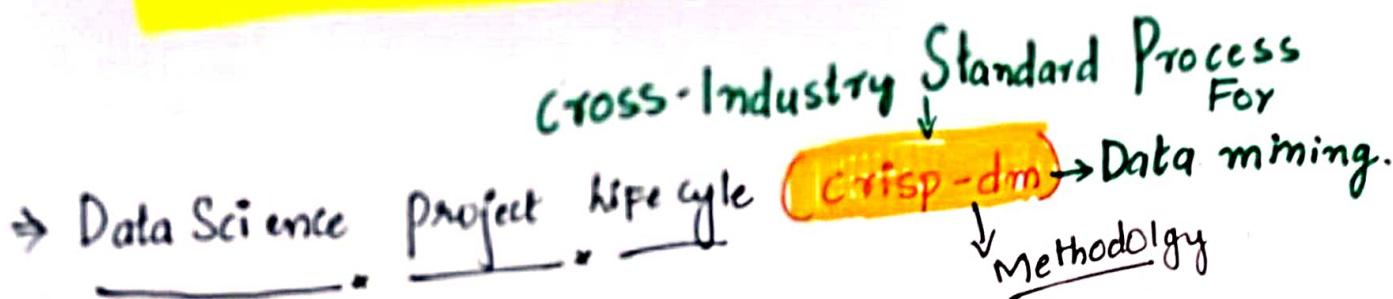
9/4/22  
5:10pm

Download Part 02 & Part 03  
<https://t.me/AIMLDeepThaught/665>

• Join me on LinkedIn for the latest updates on Machine Learning:  
<https://www.linkedin.com/groups/7436898/>

• Join my WhatsApp Channel for the latest updates on Machine Learning:  
<https://www.whatsapp.com/channel/0029VavNSDO9mrGWYirxz40G>

# \* "Simple Linear Regression" \*



1. Business problem Understanding

2. Data Understanding

3. Data Preprocessing

4. Modelling

5. Evaluation

6. Presentation

Data collection [Data Eng.]

Data variables (research domain Expert)

Dataset understanding

EDA

Data cleaning

Data Wrangling

Train / test split

Apply Different Algorithms

Accuracy checking

(or) If not go back and start again

Visualizations

(Tableau, Power BI)

⇒ Basic Libraries for all Machine Learning

# Import numpy as np

# Import Pandas as pd

# Import matplotlib.pyplot as plt

%matplotlib inline

# Import Seaborn as sns

## Step 1: Business Problem Understanding

- \* Is there a relationship between total advertising Spend and Sales?
- \* Our next ad campaign will have a total spend of \$200k, how many units do we expect to sell as a result of this?

## Step 2.1 Data collection

# df = pd.read\_csv ("Advertising.csv")

# df.head()

Out

| TV    | Radio | Newspaper | Sales |
|-------|-------|-----------|-------|
| 230.1 | 37.8  | 69.2      | 22.1  |
| 44.5  | 39.3  | 45.1      | 10.4  |
| 17.2  | 45.9  | 69.3      | 9.3   |
| 151.5 | 41.3  | 59.5      | 18.5  |
| 180.8 | 10.8  | 58.4      | 12.9  |

## Step 2.2 Data Understanding

The sample data displays Sales (in thousands of units) for a particular product as a function of advertising budgets (in thousand of dollars) for TV, radio and Newspaper media.

## Independent Variables

- TV : Advertising dollars spent on TV for a Single product in a Given Market (in thousands of dollars) → EX:  $230.1 \times 1000$  already done scaling
- Radio : Advertising dollars Spent on Radio
- Newspaper : Advertising dollars Spent on news paper

## Target Variable

- Sales : Sales of a Single product in a given market (in thousands of widgets) → EX:  $22.1 \times 1000$  = 22100 products sold

## Step - 2.3 Dataset Understanding

# df.info()

| Out | column    |
|-----|-----------|
| 0   | TV        |
| 1   | Radio     |
| 2   | Newspaper |
| 3   | Sales     |

|   | Non Null Count | Dtype    |
|---|----------------|----------|
| 0 | 200 non null   | float 64 |
| 1 | 200 non null   | float 64 |
| 2 | 200 non null   | float 64 |
| 3 | 200 non null   | float 64 |

(200,4)

Que :- If some one was to spend a total of \$200, what would the expected sales be?

X :- We have simplified all Features

This quite entails bit by Combining into "total spend"

We add new column, total or 3 column creating new one

# df[total-spend] = df["TV"] + df["radio"] + df["newspaper"]

# df.head()

| Out | TV    | Radio | newspaper | Sales | total-Spend |
|-----|-------|-------|-----------|-------|-------------|
| 0   | 230.1 | 37.8  | 69.2      | 22.1  | 337.1       |
| 1   | 44.5  | 39.3  | 45.1      | 10.4  | 128.9       |
| 2   | 17.2  | 45.9  | 69.3      | 9.3   | 132.4       |
| 3   | 151.5 | 41.3  | 58.5      | 18.5  | 251.3       |
| 4   | 180.8 | 10.8  | 58.4      | 12.9  | 250.0       |

We drop 3 columns, consider only two columns For Calculating

# df.drop (columns = ["TV", "radio", "newspaper"], inplace=True)

# df.head()

Download Part 02 & Part 03

<https://t.me/AIMLDeepThaught/665>

Out

| Sales | Total-Spend |
|-------|-------------|
| 22.1  | 337.1       |
| 10.4  | 128.9       |
| 9.3   | 132.4       |
| 18.5  | 251.3       |
| 12.9  | 250.0       |

Exploratory Data Analysis [EDA]

Step - 3.1

On The basis of this data. How should you spend advertising money in future? These general questions might lead you to more specific questions:

1. Is there a relationship between ads and Sales?

Is that correlation?

How strong is that correlation?

Given ad spending, can Sales be predicted?

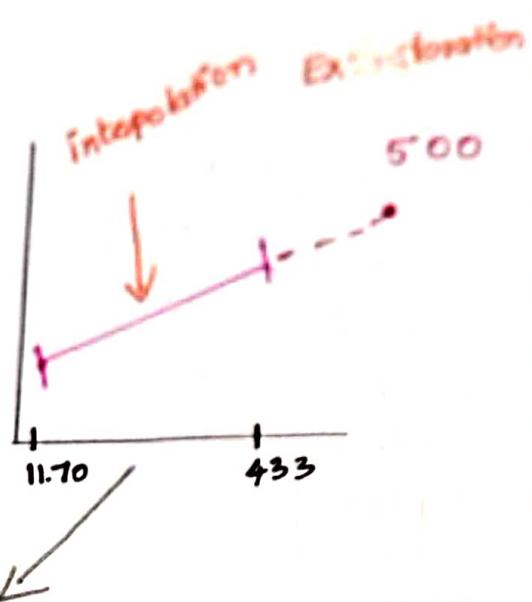
Given ad spending,

## # df.describe()

[Out]:

|       | Sales      | total-spend |
|-------|------------|-------------|
| Count | 200.000000 | 200.000000  |
| Mean  | 14.022500  | 200.860500  |
| Std   | 5.217457   | 92.985181   |
| min   | 1.600000   | 11.700000   |
| 25%   | 10.375000  | 123.550000  |
| 50%   | 12.900000  | 207.350000  |
| 75%   | 17.400000  | 281.125000  |
| max   | 27.000000  | 433.600000  |

How close  
The values  
in dataset  
To the mean  
  
Mean and median  
are close.  
They normal  
distribution



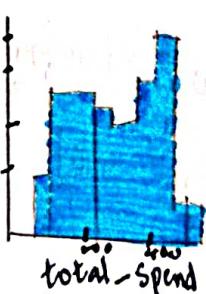
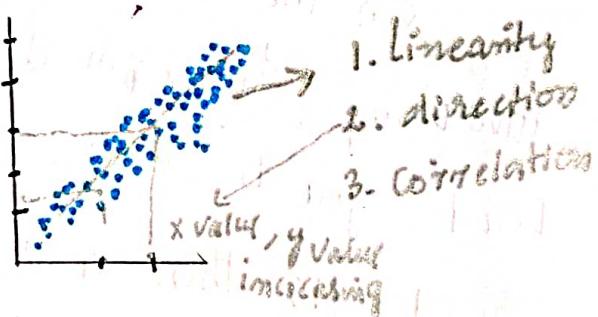
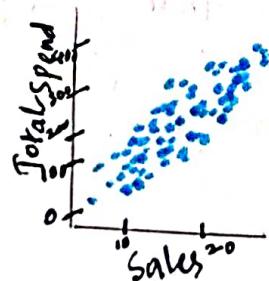
- \* **Interpolation** : Estimation of value with in two known values in sequence of values.
- \* **Extrapolation** : Prediction of value outside of known values. it is called Extrapolation.



## # sns.pairplot(df)

## # plt.show()

[Out]:



# df.corr()

| out         | Sales    | total_spend |
|-------------|----------|-------------|
| Sales       | 1.000000 | 0.867712    |
| total-spend | 0.867712 | 1.000000    |

Step 3.2 Data cleaning

# df.isnull().sum()

out sales 0  
total-spend 0

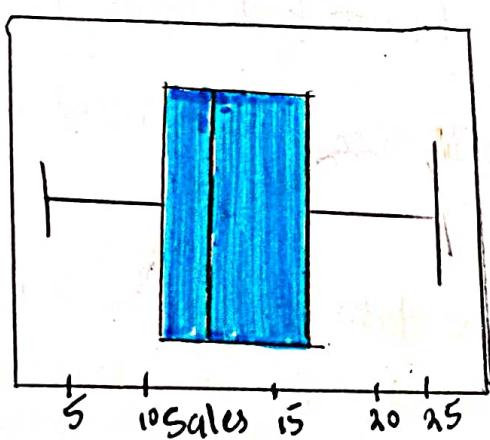
Step 3.3 Data Wrangling

Download Part 02 & Part 03

\* Outliers <https://t.me/AIMLDeepThaught/665>

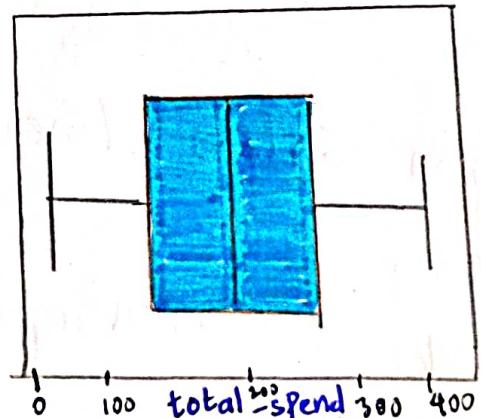
# sns.boxplot(df.sales)

# plt.show()



# sns.boxplot(df.total\_spend)

# plt.show()



\* Feature transformation

# df.sales.skew()

[Out]: 0.407

# df.total\_spend.skew()

[Out]: 0.049

Step

3.

Train Test split

# x = df.drop(columns = "sale")

# y = df["sales"]

from sklearn.model\_selection

import train\_test\_split

# x-train, x-test, y-train, y-test = train\_test\_split(x, y, test\_size = 0.3, random\_state = )

Step-4 Modelling :- If doing with data preprocessing directly  
working on modelling is "Base line model" (0%) raw model.

# from sklearn.linear\_model

import LinearRegression

# Storing  
# model = LinearRegression()

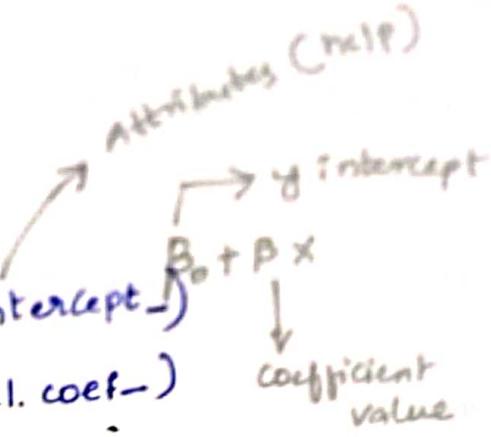
# model.fit(x-train, y-train)

[Out]: Linear Regression()

# Print("Model Intercept : ", model.intercept\_)

print("Model Coefficient : ", model.coef\_)

[Out]: Model Intercept : 4.33176  
Model Coefficient : 0.0480



$$\hat{y} = 4.33176 + 0.0480(x)$$

→ Predictions

# spend = 200

# predicted - sales = 4.33176 + 0.0480 \* spend

# Predicted - sales

[Out]: 13.93860

If we have multiple equations, we can write this

# model.predict([[200]])

input should  
given in  
2 dimensional

[Out]: array([13.93860])

Predicting on x-train, x-test

|            | x | y  | $\hat{y}$ |
|------------|---|----|-----------|
| x<br>train | 1 | 10 | 14        |
|            | 2 | 15 | 12        |
|            | 3 | 20 | 13        |
|            | 4 | 30 | 14        |
| x<br>test  | 5 | 50 | 33        |

→ Train accuracy

→ Test accuracy

# train-Predictions = model.predict(x-train)

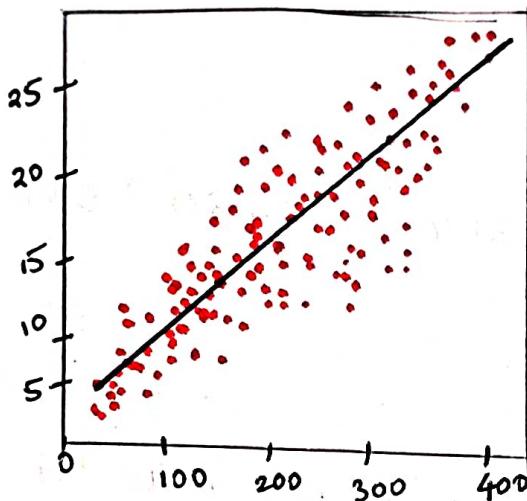
# test-Predictions = model.predict(x-test)

Only stores No output

Download Part 02 & Part 03  
<https://t.me/AIMLDeepThaught/665>

→ plotting the Least Squares line

```
# plt.scatter(x-train, y-train, color = "red")
# train-predictions = model.predict(x-train)
# plt.plot(x-train, train-predictions, color = "black")
# plt.show()
```



Step 5

```
⇒ Evaluation metrics
from sklearn.metrics import mean_absolute_error
# print("MAE For Test data:", mean_absolute_error(y-test,
test-predictions))
# print("MAE For Train data:", mean_absolute_error(
y-train, train-predictions))
```

Out: MAE For Train data : 1.97168  
MAE For Test data : 1.90653

```
from sklearn.metrics import mean_squared_error
```

```
# print ("MSE for test data : ", mean_squared_error(y-test,  
test_predictions))
```

```
# print ("MSE for train data : ", mean_squared_error(y-train,  
train_predictions))
```

**out**: MSE for Test data : 6.82220

MSE for Train data : 6.40720

RMSE (Just write `np.sqrt`).  $\rightarrow$  `np.sqrt`

```
# print ("RMSE for test data", np.sqrt(mean_squared_error  
(y-test, test_predictions)))
```

```
# print ("RMSE for train data", np.sqrt(mean_squared_error  
(y-train, train_predictions)))
```

**out**: RMSE for Test data : 2.611935

RMSE for Train data : 2.53124

```
from sklearn.metrics import r2_score
```

```
# print ("R2 for test data", r2_score(y-test, test-  
predictions))
```

```
# print ("R2 for train data", r2_score(y-train, train-  
predictions))
```

**out**: R2 for Test data : 0.74001

R2 for Train data : 0.76534

Another way for R<sup>2</sup>  
# model.score(x\_train, y\_train)

[Out]: 0.76534

# model.score(x\_test, y\_test)

[Out]: 0.740017

# cross-validation  $\Rightarrow$  K-Fold Cross Validation  
from sklearn.model\_selection import cross\_val\_score

# scores = cross\_val\_score(model, x, y, cv=5)  
Here K = 5

# print(scores)

# scores.mean()

[Out]: [0.74964192, 0.79455226, 0.76417134, 0.74872042,  
0.65980565]

$\Rightarrow 0.74337 \rightarrow$  should be equal to test square  
Then it is good model.

Linear Regression assumptions

- L  $\rightarrow$  Linearity
- I  $\rightarrow$  Independent
- N  $\rightarrow$  Normality
- E  $\rightarrow$  Equal Variance

any  
10/14/22  
3:30 AM

## → CHECK LIST ←

### Assumptions of linear regression

1. check whether model has overfitting (if) underfitting problem
2. Is test Accuracy = Cross Validation Score
3. check assumptions (if it is linear Regression)  
check model meets business problem requirements
4. Finally, save the model and share to deployment team.
5. purely Assumption of The Researcher

Ques:- Is model has overfitting (if) underfitting problem?

Ans :- It's good model.

Ques :- Is test accuracy = Cross Validation Score?

Ans :- Applied K-fold cross validation and it is equal in test accuracy and CV score.

Ques :- Check Assumptions (if it is linear regression)

Ans :- Line Assumptions.

1. Linearity \* of Errors

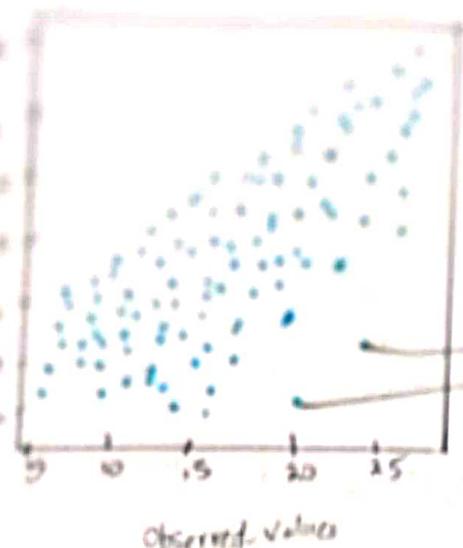
# test\_res =  $y_{\text{test}} - \text{test\_Predictions}$ .  
(↑ Saved in TM)

# plt.scatter(y\_test, test\_res)

plt.xlabel("Observed-values")

plt.ylabel("fitted-values")

plt.show()

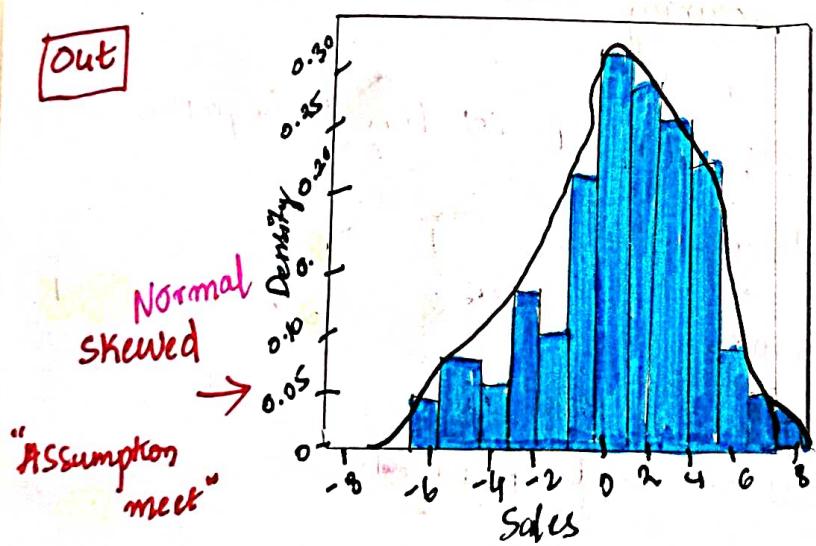


We can ignore  
The Edge points  
upto  $\pm 5\%$  error

## 2. Normality of Errors

# `sns.distplot(test_res, bins=15, kde=True)`  
`plt.show()`

Out



Normal  
Skewed  
→  
"Assumption  
meet"

↓ p. of positive Errors  
↓ p. of negative Errors  
↓ 1.6% color = red

(Homoscedasticity)

## 3. Equal variance of Error

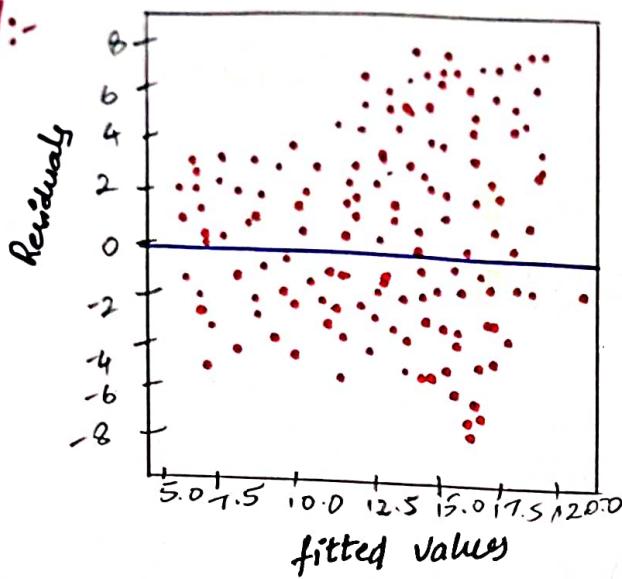
# `plt.scatter(test_predictions, test_res, c="blue")`

# `plt.axhline(y=0, color = "blue")`

↓ ax = axis → horizontal line

```
# plt.xlabel ("fitted values")
# plt.ylabel ("Residuals")
# plt.show()
```

[out]:-



For model

$H_0$ : Augline best fit line  
 $H_1$ : regression line best line

Anova test  
 $P \leq \alpha$   
 $P \geq \beta$

4. Variable significance

```
# import statsmodels.formula.api as Smf
```

# m = Smf.ols ("y~x", data=df).fit ()

# m. summary ()

OLS Regression Results.

[out]

Dep. Variable : y

R-squared : 0.753

Adj. R-squared : 0.752

F. static : 603.4

prob.(F.static) :  $5.06 \times 10^{-62}$

| p t   | 0.025 | 0.975 |
|-------|-------|-------|
| 0.000 | 3.378 | 5.908 |
| 0.000 | 0.045 | 0.053 |

Confidence Interval  
 $\text{Std.Error} = 6/5n$

Model : OLS

method : Least squares

coffs std.error t

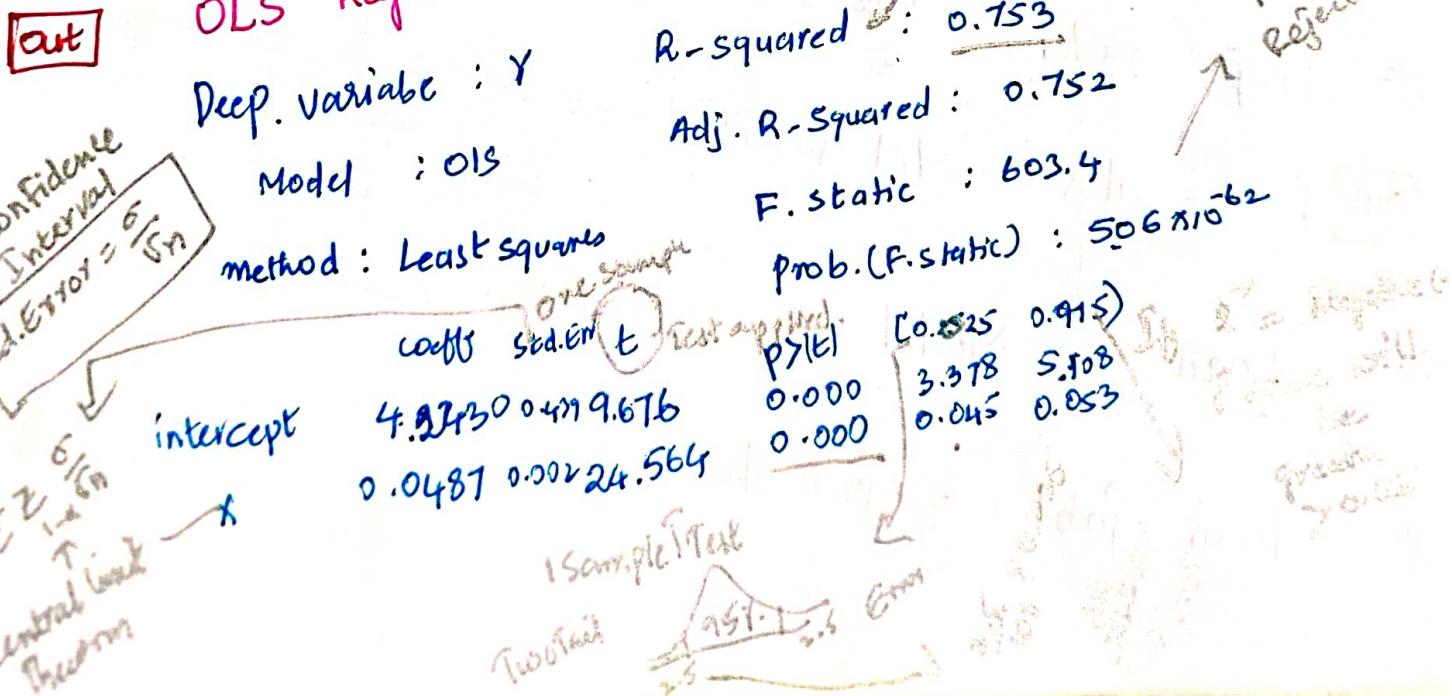
intercept

4.923 0.0479 9.676

0.0487 0.00224 5.564

1-sample T-test  
 Two-tail

t-statistic  
 p-value  
 degrees of freedom



When doing  $\hat{y}$  going to fail in regression?  
When sum of square Coefficient Average line  
When sum of square the sum of square of regression  
↳ **Last Point** the sum of square of regression  
 $(SST)_{avg}$  it going to fail in this situation.

$$(SST)_{avg} < (SSE)_{reg}$$

### Step G Final Inferences

```
# model.predict([[200]])
```

**Out** array ([13.9898])

→ Save a Model  
from joblib import dump  
import "Sales-model.joblib")

```
# dump(model,
```

**Out**: ["sales-model.joblib"]

→ saves in working directory

→ Load a Model → from client side

```
# from joblib import load
```

```
# loaded-model = load ("sales-model.joblib")
```

```
# loaded-model.predict ([[200]])
```

[out] array ([13.989])

```
# loaded-model.predict ([[500]]) # check with multiple values.
```

[out]: array ([28.3488])

~~27/4/22  
5.00pm~~

~~with 74% accuracy.~~

## Download Part 02 & Part 03

<https://t.me/AIMLDeepThaught/665>

• Join me on LinkedIn for the latest updates on Machine Learning:

<https://www.linkedin.com/groups/7436898/>

• Join my WhatsApp Channel for the latest updates on Machine Learning:

<https://www.whatsapp.com/channel/0029VavNSDO9mrGWYirxz40G>

Date :- 11/04/22  
10:30 pm

## Multiple linear Regression

## # Multiple linear Regression :-

Examines Relationship between

(or) more than two

## Variables

- \* Each independent variable has its own corresponding coefficient.

Coefficient:

$$y = b_0 + b_1 * x_1 + b_2 * x_2 + \dots + b_n * x_n$$

dependent variable

Independent Variables.

⇒ Example :

| TV | Bad<br>to | nP | Sales |
|----|-----------|----|-------|
|    |           |    |       |

$$\rightarrow \hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3$$

↓ ↓ ↓  
 TV + paper newspaper

TV Radio news  
paper

y-intercept :  $y$

Coefficient :-  $\beta_1, \beta_2, \beta_3$

Same Example : But different method

### Step-1

#### Bussiness Problem Understanding.

- \* What is The relation b/w each advertising channel [TV, Radio, Newspaper] and Sales?  $y = f(x_1, x_2, x_3)$
- \* Previously, We Explored is There a relationship between Total advertising Spend and Sales? as well as predicting the total sales for some value of Total spend.

### Step: 2.1

#### Data collection

```
# df = pd.read_csv("Advertising.csv")
# df.head()
```

Out

| TV    | Radio | Newspaper | Sales |
|-------|-------|-----------|-------|
| 230.1 | 37.8  | 69.2      | 22.1  |
| 44.5  | 39.3  | 45.1      | 10.4  |
| 17.2  | 45.9  | 69.3      | 9.3   |
| 151.5 | 41.3  | 58.5      | 18.5  |
| 180.8 | 10.8  | 58.4      | 12.9  |

### Step 2.2

#### Data Understanding

Same as Simple linear Regression Dataset.

### Step 2.3

#### Data Set Understanding

```
# df.info()
```

Step-3.2

## Exploratory Data Analysis [EDA]

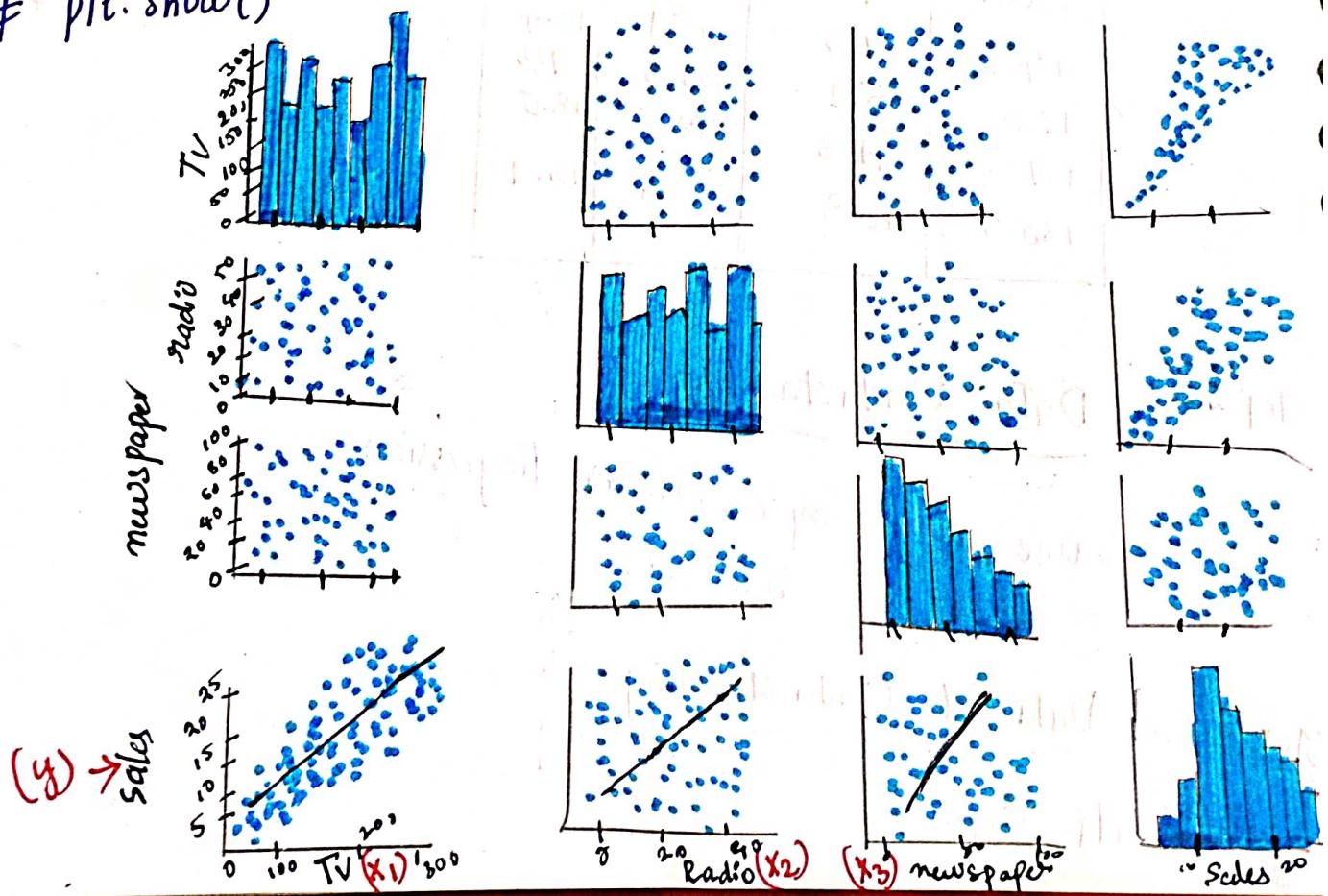
# df.describe()

out:

|       | TV         | Radio      | newspaper  | Sales      |
|-------|------------|------------|------------|------------|
| Count | 200.000000 | 200.000000 | 200.000000 | 200.000000 |
| Mean  | 147.042500 | 23.264000  | 30.554000  | 14.024500  |
| Std   | 85.854236  | 14.846809  | 21.778621  | 5.217457   |
| min   | 0.700000   | 0.000000   | 0.300000   | 1.600000   |
| 25%   | 74.375000  | 9.975000   | 12.750000  | 10.375000  |
| 50%   | 149.750000 | 22.900000  | 25.750000  | 12.900000  |
| 75%   | 218.825000 | 36.525000  | 45.100000  | 17.400000  |
| max%  | 296.400000 | 49.600000  | 114.000000 | 27.000000  |

# sns.pairplot(df)

# plt.show()



→ By Observing The Scatter plot, we made an assumption of relation between  $y$  and  $[x_1 + x_2 + x_3]$  is Linear (symmetrical Matrix)

# df. corr(C) →

- i)  $y$  (vs)  $x_1$
- ii)  $y$  (vs)  $x_2$
- iii)  $y$  (vs)  $x_3$

relation → strong → High Accuracy

$x_1$  (vs)  $x_2$  → low  
 $x_1$  (vs)  $x_3$  ↓  
 $x_2$  (vs)  $x_3$  ↓

If strong, collinearity problem

|            | TV       | Radio    | News Paper | Sales    |
|------------|----------|----------|------------|----------|
| TV         | 1.000000 | 0.054809 | 0.056648   | 0.782224 |
| Radio      | 0.054809 | 1.000000 | 0.354104   | 0.576223 |
| News paper | 0.056648 | 0.354104 | 1.000000   | 0.228299 |
| Sales      | 0.782224 | 0.576223 | 0.228299   | 1.000000 |

dependent  
Independent variable

- ⇒ The relation between "y" and "x" be high.
- ⇒ The higher The value - stronger The correlation
- ⇒ The relation between any two independent variables should be "Low"

\*\*\*\* if the correlation between any two (2) independent variables is strong. then it is called "collinearity problem".

Should be "weak"

Example

\* independent Variable

\* dependent Variable

[student 2]

[Teacher]

should be weak

should be strong

Surf 12/4/22  
3:30 PM

Dt: 12/4/21  
1:00PM

## Step: 3.2

### Data Cleaning

Same as Simple Linear

## Step 3.3

### Data Wrangling

not Same

## Step 3.4

### Train - Test Split

#  $x = df. drop [columns = "Sales"]$

#  $y = df. ["Sales"]$

from sklearn.model\_selection import train-test-split

#  $x\_train, x\_test, y\_train, y\_test = train\_test\_split (x, y, test\_size = 0.3, RandomState = 29)$

## Step - 4

### Modelling

Here,  $y = \beta_0 + \beta_1 \times [TV] + \beta_2 \times [radio] + \beta_3 \times [Newspaper]$

from sklearn.linear\_model import Linear Regression

# model = Linear Regression()

# model.fit (x\_train, y\_train)

# : Linear Regression()

Out

```
# model.coef
```

```
[Out]: array([0.04422917, 0.18181641, 0.0075874])  
           $\beta_1$        $\beta_0$        $\beta_2$ 
```

```
# model.intercept
```

```
[Out]: 2.974097357  
         $\beta_0$ 
```

→ Predictions → Predicting On "X"

```
# train-predictions = model.predict(x-train)
```

```
# test-predictions = model.predict(x-test)
```

Step: 5

Evaluations → on "y"

```
from sklearn.metrics import mean_squared_error
```

```
# test_RMSE = np.sqrt(mean_squared_error(y-test, test-predictions))
```

```
# train_RMSE = np.sqrt(mean_squared_error(y-train, train-predictions))
```

```
# print(train_RMSE, test_RMSE)
```

```
[Out]: 1.64082, 1.7805
```

# model.score (xtrain, y-train) (train R<sup>n</sup>)

[out]: 0.88817

# model.score (x-test, y-test) (test R<sup>n</sup>)

[out]: 0.905258

⇒ checklist

Que: Is model has Underfitting (or) Overfitting problem?

Ans:- Good model

Que: Is Test Accuracy = Cross Validation Score

Ans:

from sklearn.model\_selection import cross\_val\_score

# Scores = cross\_val\_score (model, x, y, cv=5)

# print (Scores)

# Scores.mean()

[0.87865198, 0.9176312, 0.92933, 0.81443, 0.895]

[out]

0.887106349

compare with xtest,ytest accuracy.

Download Part 02 & Part 03

<https://t.me/AIMLDeepThought/665>

### 3. Check Assumptions

\* Linearity of errors

#  $\text{test\_res} = y_{\text{test}} - \text{test\_predictions}$

$x_{\text{test}}$

Residuals

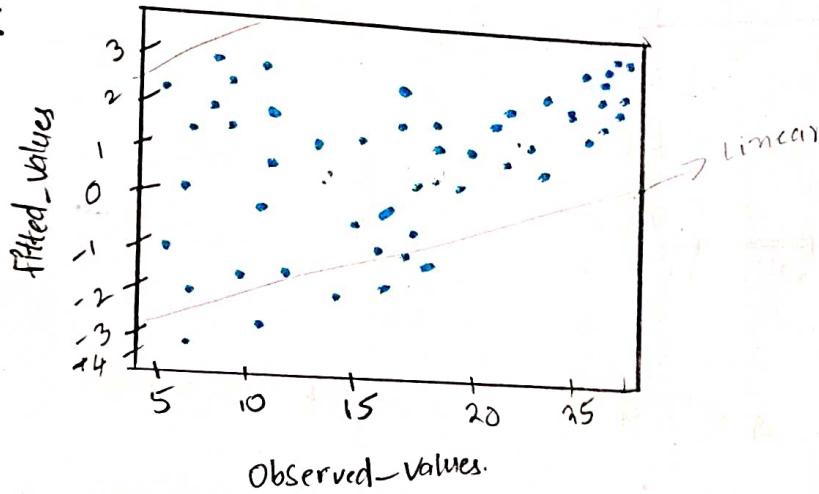
# plt. Scatter ( $y_{\text{test}}$ , test\_res)

# plt. xlabel ("Observed-values")

# plt. ylabel ("fitted-values")

# plt. show()

out:



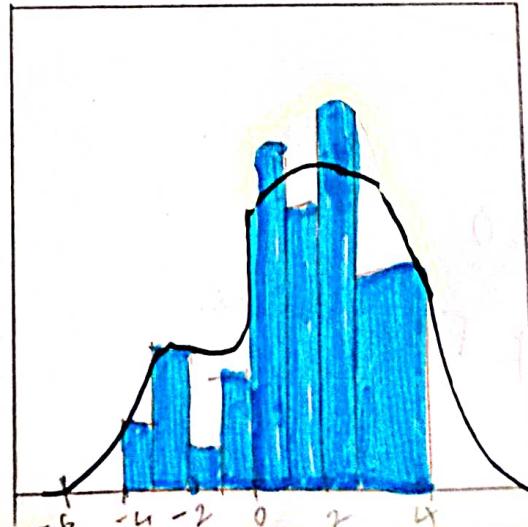
\* Normality of errors

# sns.distplot(test\_res, bins=15, kde=True)

# plt.show()

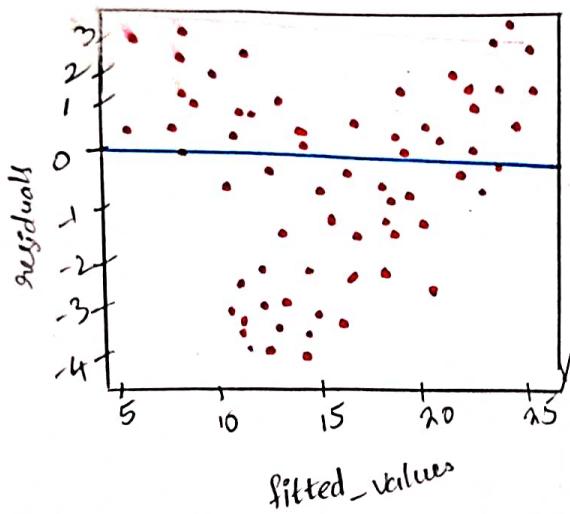
out:

Left skewed



### 3. Equal Variance of Errors (Homoscedasticity) X

```
# plt.scatter(test_predictions, test_res, c="r")  
# plt.axhline(y=0, color="blue")  
# plt.xlabel("fitted-values")  
# plt.ylabel("residuals")  
# plt.show()
```



\* ASSumptions Failed,  $R^2 = 86\%$ . [we reject the model]

⇒ Every checklist should satisfy  $\star \star \star \star$  condition. Than, Only we consider the model.

### 4. Variable Significance

$$H_0 : \beta_1 = 0$$

$$H_1 : \beta_1 \neq 0$$

⇒ Hypothesis Testing For Variables.

```
import statsmodels.formula.api smf
```

```
# model 2 = Smf. OLS("y ~ x", data=df).fit()
```

```
# model 2 . summary ()
```

[out]: OLS Regression Results.

For Model: Accept  $H_0$ :  
↓  
Reject  $H_0$ .

Dependent Variable : y

R-squared : 0.897

Model : OLS

Adj. R-squared : 0.896

method : least squares

F statistic : 570.3

Prob (F-statistic) :  $1.58 \times 10^{-96}$

|                | coeff   | std err | t      | p >  t | [0.025] | 0.975 |
|----------------|---------|---------|--------|--------|---------|-------|
| Intercept      | 2.9389  | 0.312   | 9.422  | 0.000  | 2.324   | 3.554 |
| $\beta_1$ x(0) | 0.0458  | 0.001   | 32.809 | 0.000  | 0.043   | 0.049 |
| $\beta_2$ x(1) | 0.1885  | 0.009   | 21.893 | 0.000  | 0.472   | 0.206 |
| $\beta_3$ x(2) | -0.0010 | 0.006   | -0.177 | 0.860  | -0.013  | 0.011 |

Variable 1

$H_0: \beta_1 = 0$

$H_1: \beta_1 \neq 0$

Variable wise

Variable 2

$H_0: \beta_2 = 0$

$H_1: \beta_2 \neq 0$

Variable 3

$H_0: \beta_3 = 0$

$H_1: \beta_3 \neq 0$

$P < 0.05 \rightarrow P_{low}$

Null, go  $\rightarrow$  Reject  $H_0$

$P > 0.05$

⇒ Checking whether data has any influence value using influence index plots

Download Part 02 & Part 03

<https://t.me/AIMLDeepThaught/665>

1) \* Influential index plots, For Making "P" value equal  $< 0.05$

import statsmodels.api as sm

# sm. graphics.influence\_plot(model1)

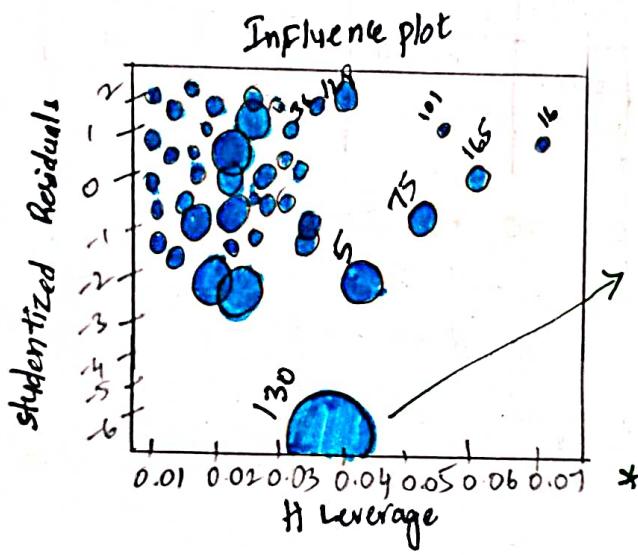
Why?

$$\beta_3 = 0,$$

$$\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3$$

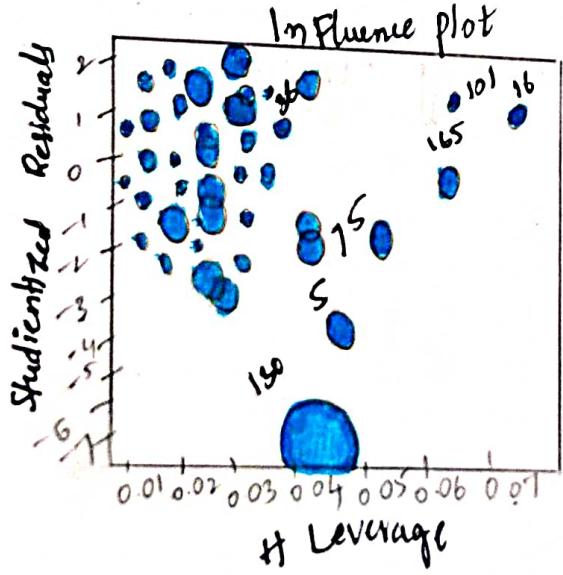
By dropping, we are losing  $\frac{1}{3}$  of data = 33%.  
In that case, we drop some records, that are influencing more.  
 $Ex: - \frac{1}{200} = 0.05\%$ .

But:



Index 130 is showing high influence. So, we exclude that entire row.

studentized Residuals = Residuals / standard deviation of residuals.



# df\_new = df.drop(df.index[[130]], axis=0) row

# df\_new.

Out

|     | TV    | Radio | News Paper | Sales |
|-----|-------|-------|------------|-------|
| 0   | 230.1 | 37.8  | 69.2       | 22.1  |
| 1   | 44.5  | 39.3  | 45.1       | 10.4  |
| 199 | 232.1 | 8.6   | 8.7        | 13.4  |

199 x 4 columns

Once again Rebuild model

# model 2 = smf.ols (formula = "Sales ~ TV + radio + newspaper", data = df\_new).fit()

# model2. summary()

Out : OLS Regression MODEL

|                 | coeff  | std. err | t      | P> t  |
|-----------------|--------|----------|--------|-------|
| Intercept       | 3.0931 | 0.290    | 10.654 | 0.000 |
| TV X(0)         | 0.0448 | 0.001    | 34.425 | 0.000 |
| Radio X(1)      | 0.1939 | 0.008    | 24.130 | 0.000 |
| news paper X(2) | -0.043 | 0.005    | -0.777 | 0.438 |

This is one method of reducing the P value.

→ Reduced  
From  
0.86 to  
0.43  
By Reducing.

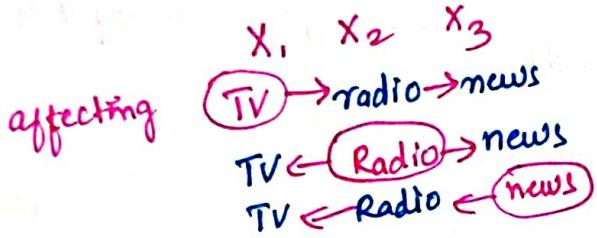
Download Part 02 & Part 03  
<https://t.me/AIMLDeepThaught/665>

2) Variance inflation Factor [VIF]

Variance inflation Factor [VIF] measures ratio between the variance for a given regression coefficient with only that variable in the model versus the variance for given regression coefficient with all variables in the model.

1 independent variable

influence on other independent variable is called as ("VIF")



$$V.I.F = \frac{1}{1-R^2}$$

#  $r_{sq-TV} = \text{smf.ols}(\text{"TV ~ radio + newspaper"}, \text{data=df})$   
• fit()

#  $r_{sq-TV}.\text{summary}()$

Out : OLS Regression Results

R-squared = 0.005

$$V.I.F = \frac{1}{1-0.005^2} \Rightarrow V.I.F = 1.0000$$

Calculating VIF's values of independent variables

#  $r_{sq-TV} = \text{smf.ols}(\text{"TV ~ radio + newspaper"}, \text{data=df}).$   
• fit(), r.squared

#  $vif-TV = 1/(1 - r_{sq-TV})$

```
# rsq_radio = Smf.ols ("radio ~ TV + newspaper", data=df)  
    .fit().rsquared
```

```
# vif_radio = 1/(1-rsq_radio)
```

```
# r sq_newspaper = Smf.ols ("newspaper ~ radio + TV",  
    data=df).fit().rsquared
```

```
# vif_newspaper = 1/(1-rsq_newspaper)
```

Storing VIF values in a DataFrame

```
# d1 = { "variables": [ "TV", "radio", "newspaper" ],  
        "VIF": [ vif_TV, vif_radio, vif_newspaper ] }
```

```
# vif_frame = pd. Data Frame (d1)
```

```
# vif_frame
```

Out

|   | variable  | VIF      |
|---|-----------|----------|
| 0 | TV        | 1.004611 |
| 1 | Radio     | 1.144952 |
| 2 | newspaper | 1.145187 |

# If The VIF model  $\rightarrow$  (greater) 4. For  
any independent variable, drop  
variable

27/04/22

4:40 pm

3) Dt: 16/9/22

⇒ AV plot [Added Variable plot)

# partial differentiation instead of normal differentiation.

$$[\text{SSE}]_{\min} \quad [\mathbb{E}(y - \hat{y})^2]_{\min}$$

$$\frac{\partial}{\partial x} \text{ at } x=0$$

#

What is partial differentiation?

$\sum [y - \hat{y}]^2 \rightarrow$  it should be Minimum

$$\left\{ \left[ y - [\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3] \right]^2 \right\}_{\text{minimum}}$$

$$\left\{ \left[ y - \beta_0 - \beta_1 x_1 - \beta_2 x_2 + \beta_3 x_3 \right]^2 \right\}_{\text{minimum}}$$

In Simple linear Regression?

$$\frac{\partial \left[ \mathbb{E}[y - \beta_0 - \beta_1 x_1]^2 \right]}{\partial x_1} \text{ at } x_1=0 \rightarrow \begin{array}{l} \text{He we have only} \\ \text{One variable.} \end{array}$$

In Multiple linear Regression?

$$\frac{\partial}{\partial x_1} \left\{ \left[ y - \beta_0 - \beta_1 x_1 - \beta_2 x_2 - \beta_3 x_3 \right]^2 \right\}$$

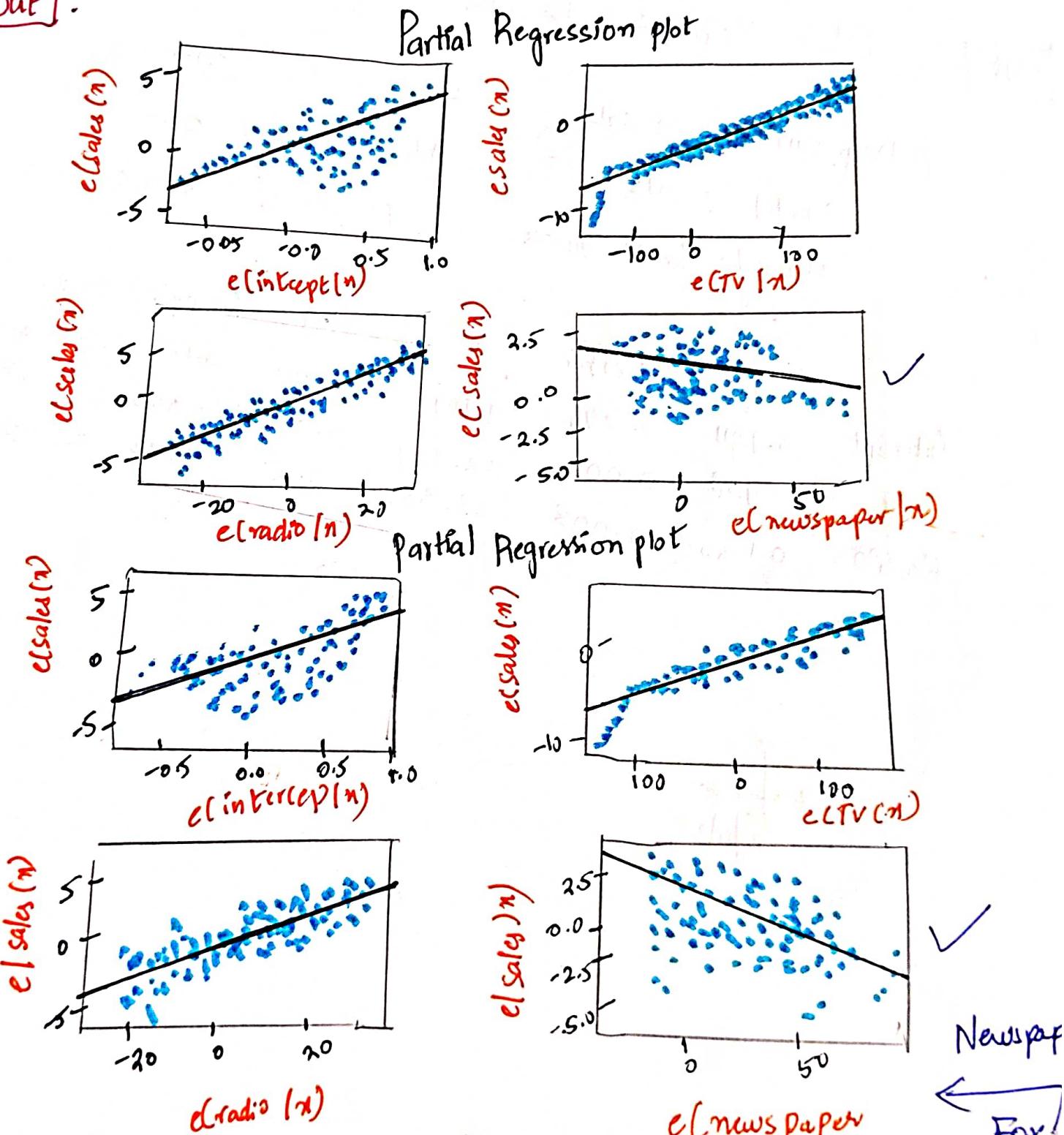
constant    calculate    constant  
constant    constant    constant

↓  
when we calculate with  $\frac{\partial}{\partial x_1} \frac{\partial}{\partial x_1}$ .  
When we calculate with one variable other variable will be constant.

→ it is apply Simple Linear Regression on  $x_1$  and  $x_2$   
 Simple Linear Reg on  $x_3$ . it is Applying individually  
 S.L.R on Each and Every individual variable.

# Sm. graphics . plot - partregress- grid (1m)  
Partregress -

**Out**:



# Added variable plot is not showing any significance For

⇒ Final model including "TV" and "Radio" Only

# final\_model = smf.ols (formula = "Sales ~ TV + radio",  
data = df). fit()

# final\_model.summary()

| Out                   | OLS Regression Results |                           |                   |
|-----------------------|------------------------|---------------------------|-------------------|
| Dep. Variable : sales |                        | R-squared : 0.897 = 90.1. |                   |
| Model : OLS           |                        | Adj. R-squared : 0.896    |                   |
| Method: Least Squares |                        | F-statistics = 859.6      |                   |
|                       |                        |                           |                   |
|                       | coeff                  | std. error                | t                 |
| Intercept             | 2.9211                 | 0.294                     | 9.919             |
| TV                    | 0.0458                 | 0.001                     | 32.909            |
| Radio                 | 0.1880                 | 0.008                     | 23.382            |
|                       |                        |                           | p-value           |
|                       |                        |                           | (0.025) (0.975)   |
|                       |                        |                           | 3.540 0.048       |
|                       |                        |                           | 0.043 0.172 0.204 |

?  
enf  
16/11/22

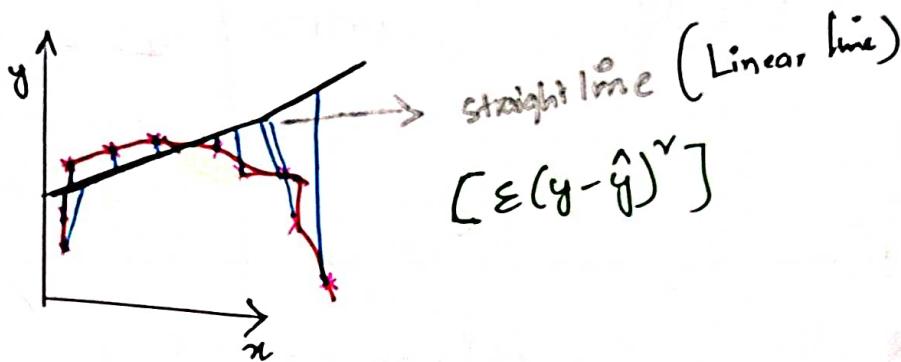
Download Part 02 & Part 03  
<https://t.me/AIMLDeepThaught/665>

Dr. Ishant 1:00 PM

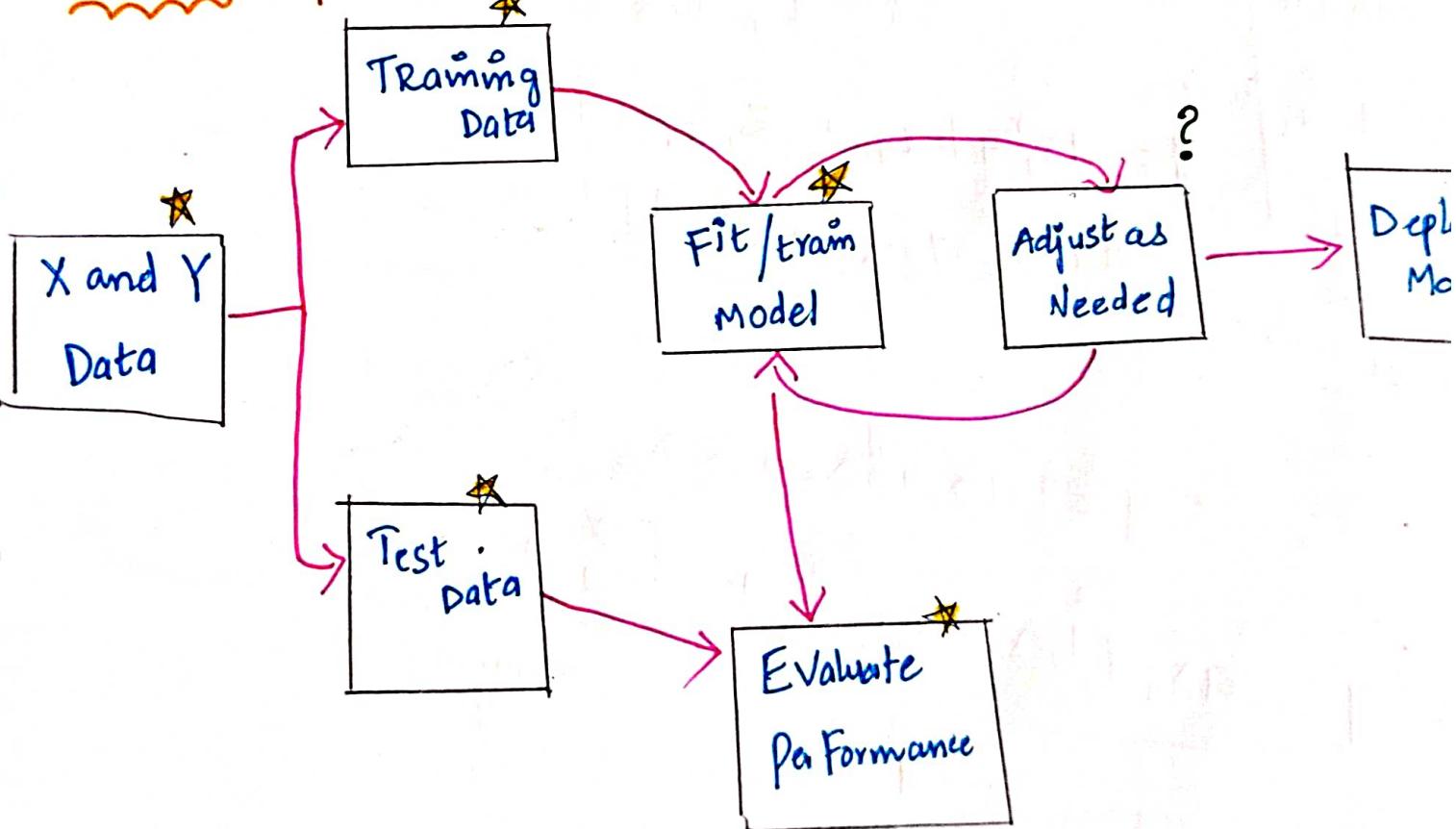
## Polynomial Regression

Any Regression problem.

$$[SSE]_{\min} \Rightarrow [\varepsilon[y - \hat{y}]^2]_{\min}$$



Supervised Machine learning Process



S.L.R

M.L.R

**1 Variable**

$$y = ax + b$$

# Linear

**2 Variables**

$$y = \frac{ax + b}{\text{1st variable}} + cx + d$$

$$y = ax_1 + bx_2 + c$$

it is came from joining  
Two equations

$$= ax_1 + b + cx_2 + d$$

always

intercept  
Have  
One value Only

**3 Variable**

$$y = ax_1 + bx_2 + cx_3 + d$$

it is came from  
Joining Three equations

$$y = ax_1 + b$$

$$y = +cx_2 + d$$

$$y = +ex_3 + f$$

#  
Quadratic  
1 Variable

$$y = ax^2 + bx + c$$

Here  
we write  
intercept value  
as "Z"

$$y = ax^2 + bx_1 + dx_2 + ex_3 + Z$$

it is came from  
Joining two  
Equations

$$y = ax^2 + bx_1 + Z_1, c_1 +$$

$$+ dx_2^2 + ex_3 + Z_2, f_2$$

$$* y = ax_1 + bx_2 + cx_3 + Z$$

$$* y = ax_1 + bx_2 + cx_3 + dx_1^2 + ex_2^2 + fx_3^2 + Z$$

$$+ gx_1x_2 + hx_2x_3 + ix_3x_1$$

Combinations of variable [Iteration terms]

original data

Squaring original Data

Combination (or)  
Iteration terms

| TV | Radio | Newspaper | Sales | $TV^2$ | $Radio^2$ | $Newspaper^2$ | $TV \cdot Radio$ | $Radio \cdot Newspaper$ | $Newspaper^2 \cdot TV$ |
|----|-------|-----------|-------|--------|-----------|---------------|------------------|-------------------------|------------------------|
| -  | -     | -         | -     | -      | -         | -             | -                | -                       | -                      |
| -  | -     | -         | -     | -      | -         | -             | -                | -                       | -                      |
| -  | -     | -         | -     | -      | -         | -             | -                | -                       | -                      |

\* Quadratic equation:  $x_1^2 + x_2^2 = x_1 + x_2$  ?

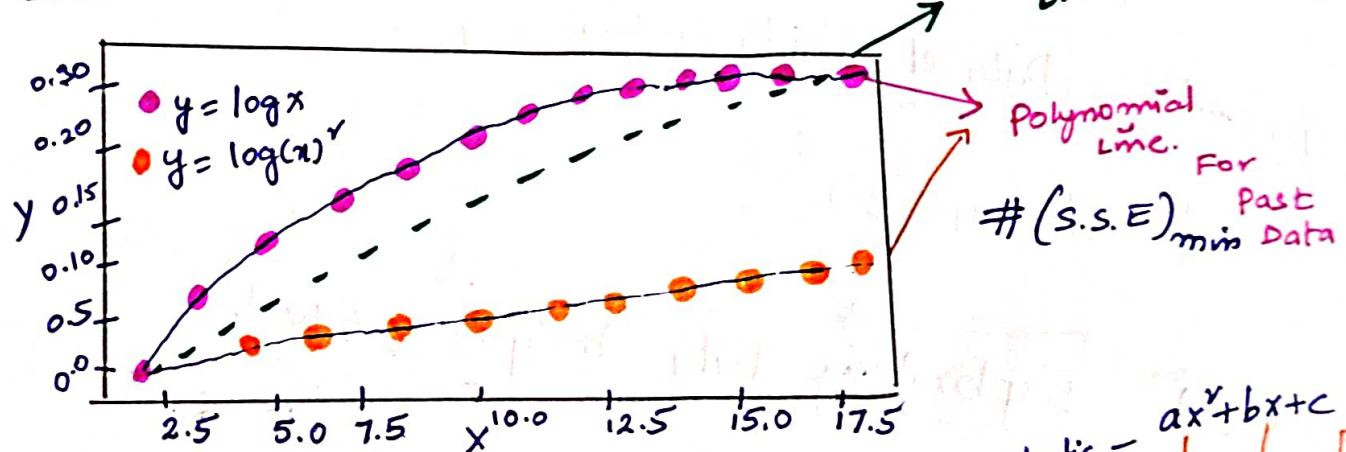
degree 2 ↗

$$a^m \cdot a^n = a^{m+n}$$

Polynomial Regression

Multiply

if we add two variables  
degree 1 ↗



linear line

Polynomial line.

#(S.S.E) for Past Data

#quadratic -  $ax^2 + bx + c$

If we want to calculate combinations with squaring.  
Degree 2 =  $TV^2 (or) x_1^2 + x_2^2 + x_3^2 + x_1 x_2 + x_2 x_3 + x_3 x_1 + \dots$

Degree 2

$$TV^2 (or) x_1^2 + x_2^2 + x_3^2 + x_1 x_2 + x_2 x_3 + x_3 x_1 + \dots$$

Degree 3 =  $x_1^3 + x_2^3 + x_3^3 + x_1 x_2^2 + x_2 x_3^2 + x_3 x_1^2 + x_1^2 x_2 + x_2^2 x_3 + x_3^2 x_1 + x_1 + x_2 + x_3$

$$x_1^3 + x_2^3 + x_3^3 + x_1 x_2^2 + x_2 x_3^2 + x_3 x_1^2 + x_1^2 x_2 + x_2^2 x_3 + x_3^2 x_1 + x_1 + x_2 + x_3$$

# cubic Equation :

$$a x^3 + b x^2 + c x + d$$

**STEP 1**

Problem Understanding

Download Part 02 & Part 03

<https://t.me/AIMLDeepThought/665>

**STEP 2**

Data Collection

```
# df = pd.read_csv("Advertising.csv")
```

```
# df.head()
```

**STEP 3**

Data Understanding

```
# df.shape
```

**STEP 4**

Dataset Understanding

```
# df.info()
```

**STEP 5**

Exploratory Data Analysis [EDA]

```
# df.describe()
```

```
# sns.pairplot(df)
```

**STEP 6**

Data Cleaning

```
# df.isnull().sum()
```

**STEP 7**

Data Wrangling

```
# x = df.drop("sales", axis=1)
```

```
# y = df["Sales"]
```

new Features  
Adding to original data

Polynomial Regression with Scikit-Learn

```
from sklearn.preprocessing import PolynomialFeatures
```

```
# polynomial_converter = PolynomialFeatures(degree=2, include_bias=False)
```

$$\begin{aligned} & ax_1^2 + bx_2^2 + cx_3^2 \\ & dx_1x_2 + ex_2x_3 + fx_3x_1 \\ & gx_1 + hx_2 + ix_3 \end{aligned}$$

don't include "z" value

```
# x_poly = polynomial_converter.fit_transform
```

↓ calculate  
↓ convert  
(x)

```
# x_poly.shape
```

Out (200, 9)  
rows ↓  
columns → changed 3 to 9

```
from sklearn.model_selection import train_test_split
```

```
# x_train, x_test, y_train, y_test = train_test_split
```

[x\_poly, y, test\_size=0.3,

Random\_state=29]

200

140 x 9 = train  
60 x 9 = test



## MODEL Fitting On Polynomial Data.

from sklearn.linear\_model import LinearRegression

# model = LinearRegression()

# model.fit(x-train, y-train)

### Predictions

# train-pred = model.predict(x-train)

# test-pred = model.predict(x-test)



### Evaluation

# model.score(x-train, y-train) [ $R^2$ ]

[out] 0.986

# model.score(x-test, y-test) [ $R^2$ ]

[out] 0.980

Download Part 02 & Part 03  
<https://t.me/AIMLDeepThought/665>

## # = cross-validation

```
from sklearn.model_selection import cross_val_score
```

```
# scores = cross_val_score (model, X_poly, y, cv=5)
```

```
# print (scores)
```

```
# scores.mean()
```

```
[Out] [0.987, 0.989, 0.991, 0.958, 0.993]
```

```
mean : 0.984
```

## # RMSE

```
from sklearn.metrics import mean_squared_error
```

```
# test-RMSE = np.sqrt (mean_squared_error (y-test, test-pred))
```

```
# train-RMSE = np.sqrt (mean_squared_error (y-train, train-pred))
```

```
# print (train-RMSE, test RMSE)
```

```
0.5950, 0.7233
```

Earlier,

Multiple Linear Regression

\* RMSE - 1.94

Polynomial Regression

\* RMSE - 0.72

```
[Out]
```

Download Part 02 & Part 03  
<https://t.me/AIMLDeepThaught/665>

## # Applying loop for knowing better Accuracy From First to Last.

```
• from sklearn.preprocessing import PolynomialFeatures  
• from sklearn.model_selection import train_test_split  
• from sklearn.linear_model import LinearRegression
```

```
# train_rmse_errors = []
```

```
# test_rmse_errors = []
```

for d in range [1, 10] :

```
# polynomial_Converter = PolynomialFeatures(degree=d, include_bias=False)
```

```
# x_poly = polynomial_Converter.fit_transform(x)
```

```
# x_train, x_test, y_train, y_test = train_test_split(x_poly, y, test_size=0.3, RandomState=29)
```

```
# model = LinearRegression()
```

```
# model.fit(x_train, y_train)
```

```
# train_pred = model.predict(x_train)
```

```
# test_pred = model.predict(x_test)
```

```
# train_RMSE = np.sqrt(mean_squared_error(y_train, train_pred))
```

```
# train_rmse_errors.append(train_RMSE)
```

```
# test_RMSE = np.sqrt(mean_squared_error(y_test, test_pred))
```

```
# test_rmse_errors.append(test_RMSE)
```

## # train\_rmse\_errors

**Out**: 0 [1.7345 d=1  
1 0.5879 d=2  
2 0.4339 d=3  
3 0.3517 d=4  
4 0.2509 d=5  
5 0.19704 d=6  
6 0.14214 d=7  
7 0.14180 d=8  
8 0.16654 d=9

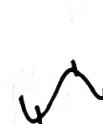
degree

No. of bends

 = quadratic

 = cubic

 = degree of 4

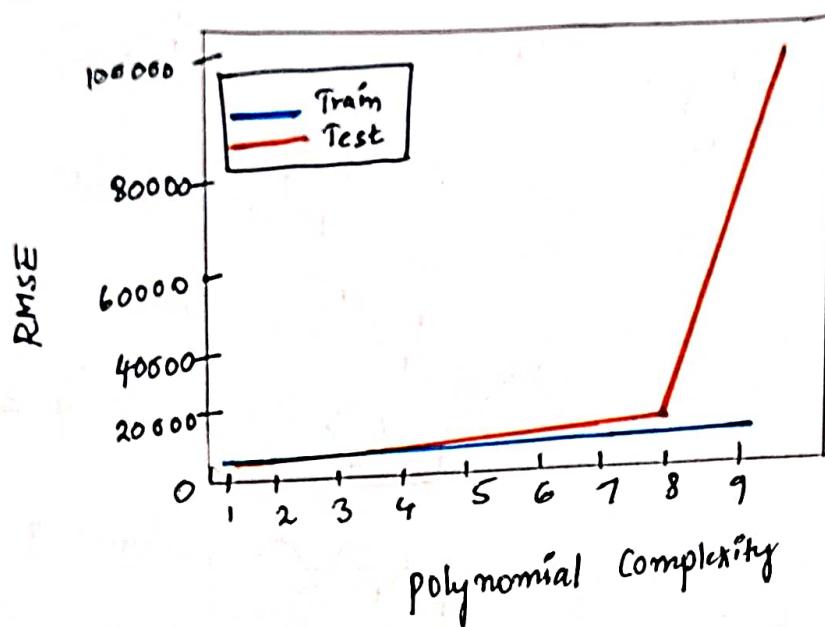
 = degree of 7

## # test\_rmse\_errors

**Out**: 0 1.5161 d=1  
1 0.6646 d=2  
2 0.5803 d=3  
3 0.5077 d=4  
4 0.5758 d=5  
5 0.4926 d=6  
6 0.381404 d=7  
7 0.4449.599 d=8  
8 0.959124 d=9

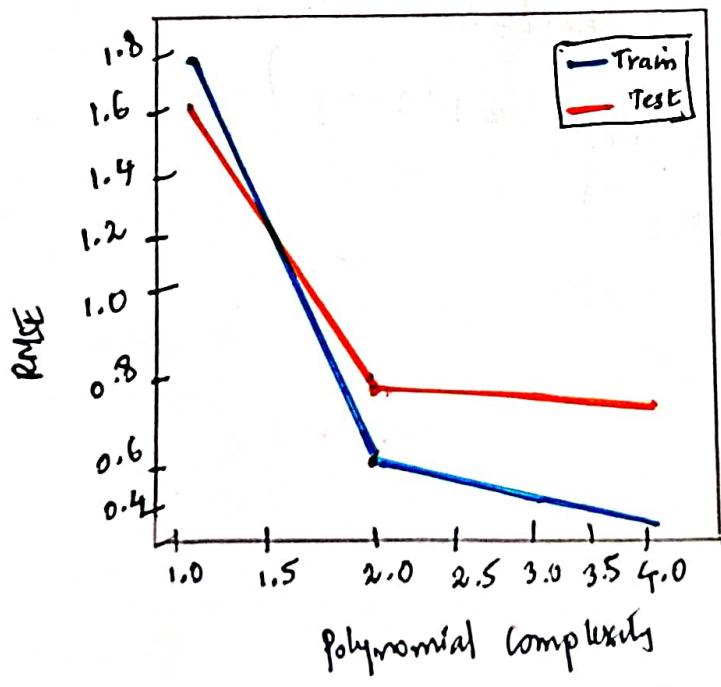
# plt.plot (range(1,10), train\_rmse\_errors, Label = "TRAIN")  
# plt.plot (range(1,10), test\_rmse\_errors, Label = "TEST")  
# plt.xlabel ("Polynomial complexity")  
# plt.ylabel ("RMSE")  
# plt.legend ()  
# plt.show ()

**Out** :



```
# plt.plot(range(1,5), train_rmse_errors[:4], label = "TRAIN")
# plt.plot(range(1,5), test_rmse_errors [:4], label = "TEST")
# plt.plot(range(1,5), test_rmse_errors [:4])
# plt.xlabel ("Polynomial complexity")
# plt.ylabel ("RMSE")
# plt.legend()
# plt.show()
```

**Out** :



5/5

## Finalizing Model choice

```
# final_poly_Converter = polynomial Features (degree = 2,  
include_bias = False)
```

```
# final_model = Linear Regression()
```

```
# final_model.fit(final_poly_Converter.fit_transform(X), y)
```

**out**: Linear Regression()

\* Saving MODEL and CONVERTER

```
from joblib import dump
```

```
# dump(final_model, "Sales-poly-model.joblib")  
out ("Sales-poly-model.joblib")
```

```
# dump(final_poly_Converter, "Poly-converter.joblib")
```

**out**: [{"poly-converter.joblib"}]

# Deployment & predictions:

Client wants to spend 149K on "TV", 22K on "radio"  
12K on "Newspaper" Ads. How many units could we  
expect to sell as a result?

```
from joblib import load
```

```
# loaded_poly = load ("poly-converter.joblib")
# loaded_model = load ("Sales-poly-model.joblib")
-
# Campaign_poly = loaded_poly.transform ([[149, 22, 12]])
-
# final_model.predict(campaign_poly)
[OUT]: array([14.5114])
```

anuj  
16/04/22  
9:25 pm

Download Part 02 & Part 03  
<https://t.me/AIMLDeepThaught/665>

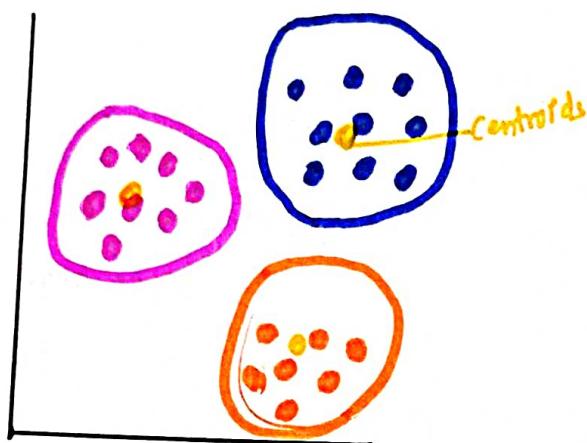
## \* DBScan Clustering



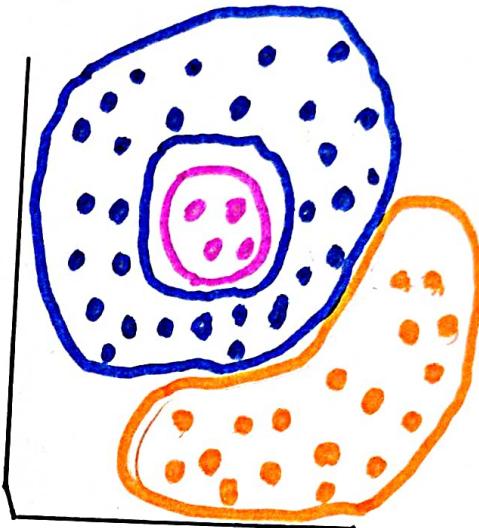
Density Based spatial Clustering of application with noise.

### # Density - Based clustering

\* Spherical - Shape clusters



\* Arbitrary shape clusters



- K-means assigns all points to a cluster Even if they do not belong in any

- Density based clustering Locates regions of "high density", and separates outliers.  
\*\*\* Based on Given density of values it is identifying the clusters

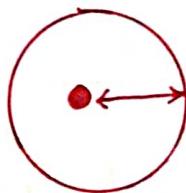
## \* What is DBSCAN?

### \* DBSCAN

- it is one of the most common clustering algorithms
- works based on density of objects.

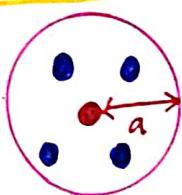
### \* R (radius of neighborhood)

- Radius (R) that if includes enough number of points within, we call it a dense area.

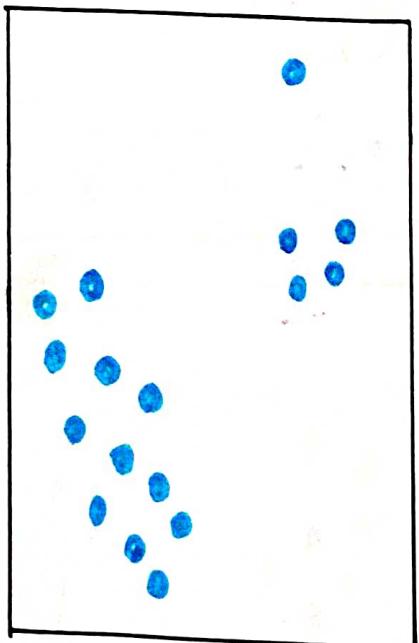


### \* M (min number of neighbors)

- The minimum number of data points we want in a neighborhood to define a cluster.



Ex:-



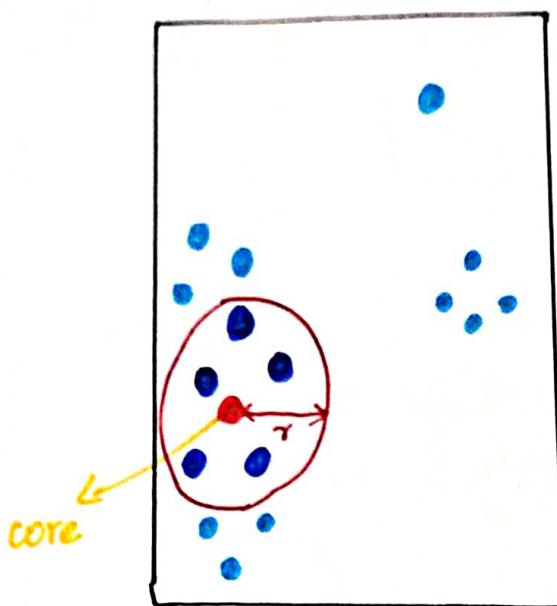
Each point is either :

- core Point
- border Point
- Outlier Point

$$\therefore R = 2 \text{ unit}, M = 6$$

\* DBScan algorithm :- Core point

↓  
which is satisfy minimum no. of neighbours.

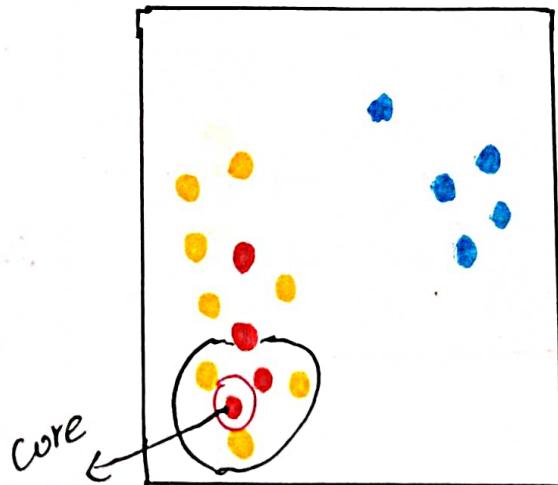
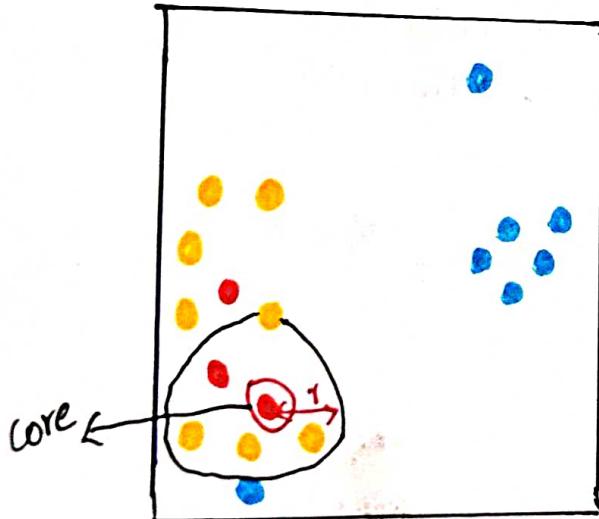
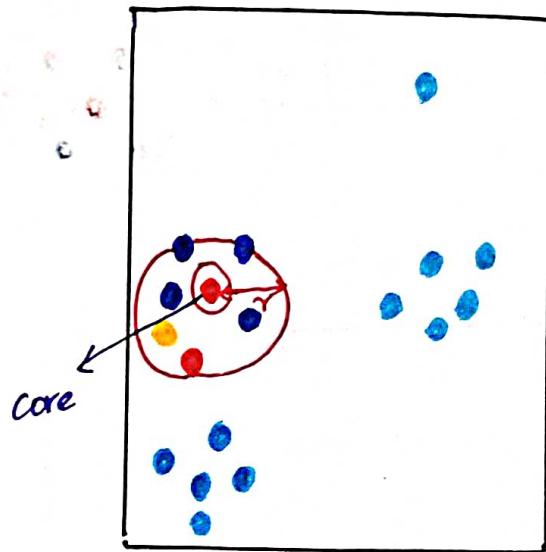
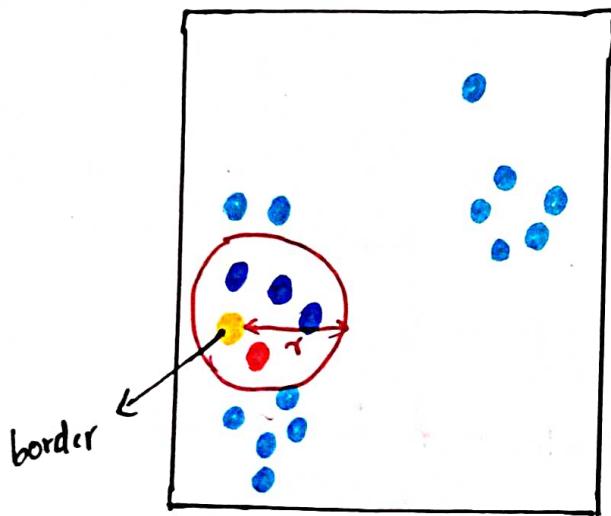


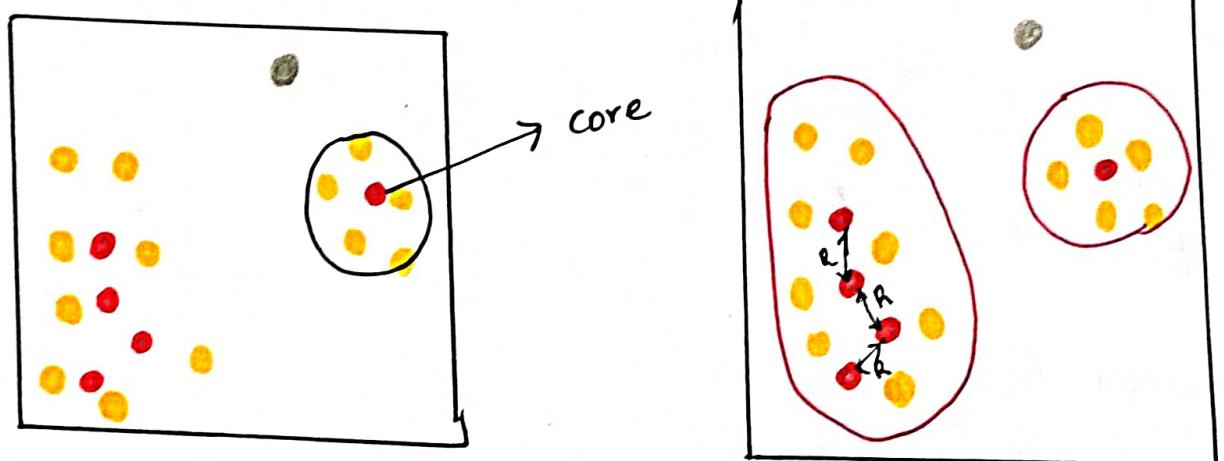
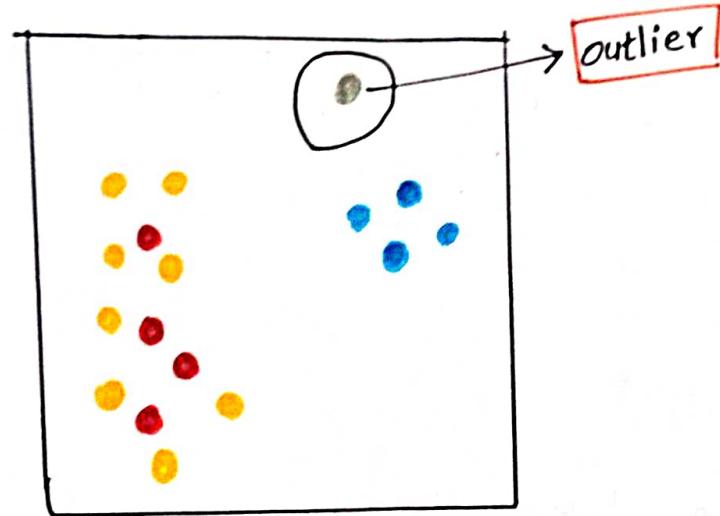
$$R = 2 \text{ unit}, M = 6$$

↓  
minimum.

\* DBScan algorithm : border point

$$R = 2 \text{ unit}, M = 6$$





CODE : Same data & Same steps till modelling

MODEL: DBSCAN

```
from sklearn.cluster import DBSCAN
# dbs = DBSCAN (eps=5, min_samples=5)
    ↳ Radius.
```

Predict

```
# y-dbs = dbs.fit_predict(x)
```

```
# y-dbs
```

[out]: array([-1, 0, -1, 0, -1, 0, -1, -1, 0, 0, -1  
          , -1, -1, -1, -1, -1, -1, -1, -1, -1, -1]).

```
# np.unique(y-dps)
```

[out]: array([-1, 0, 1, 2, 3, 4]).

### Visualising the clusters

```
# cluster 1
```

```
# plt.scatter(x[y-dbs == -1, 0], x[y-dbs == -1, 1],  
              s=100, c="red", label="cluster 1")
```

```
# cluster 5
```

```
# plt.scatter(x[y-dbs == 0, 0], x[y-dbs == 0, 1],  
              s=100, c="magenta", label="cluster 5")
```

```
# cluster 2
```

```
# plt.scatter(x[y-dbs == 1, 0], x[y-dbs == 1, 1],  
              s=100, c="blue", label="cluster 2")
```

```
# cluster 3
```

```
# plt.scatter(x[y-dbs == 2, 0], x[y-dbs == 2, 1],  
              s=100, c="green", label="cluster 3")
```

```
# cluster 4
```

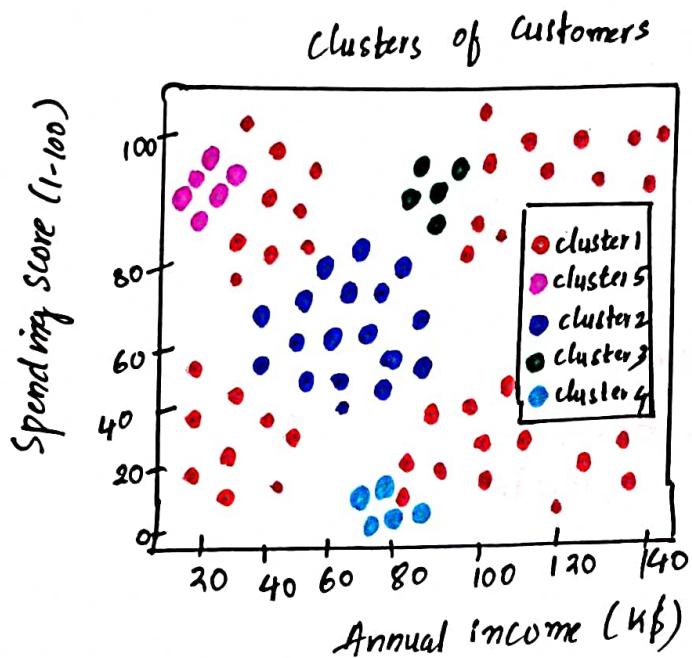
```
# plt.scatter(x[y-dbs == 3, 0], x[y-dbs == 3, 1],  
              s=100, c="cyan", label="cluster 4")
```

```

# plt.title ("clusters of customers")
# plt.xlabel ("Annual income (k$)")
# plt.ylabel ("spending score (1-100)")
# plt.legend()
# plt.show()

```

**Out:**



n  
 04 05/22  
 5:00pm

Download Part 02 & Part 03  
<https://t.me/AIMLDeepThaught/665>

## \* Hierarchical Clustering [HC]

↓  
Combine data points which are very close by distance

Ex:-

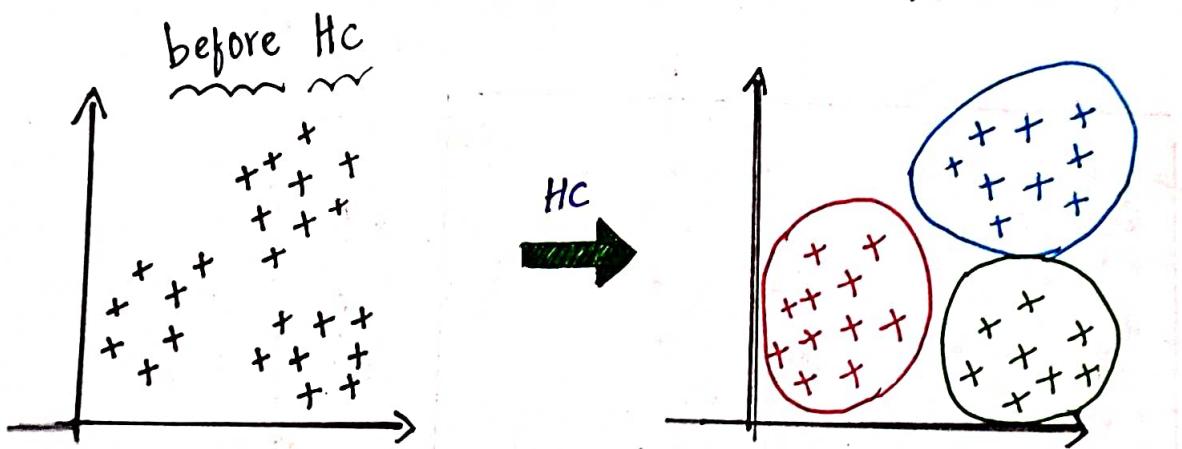


⇒ Hierarchy

- \* Hierarchical clustering algorithms build a hierarchy of clusters where "Each node is a cluster" consists of clusters of its daughter nodes.

What "HC" does ?

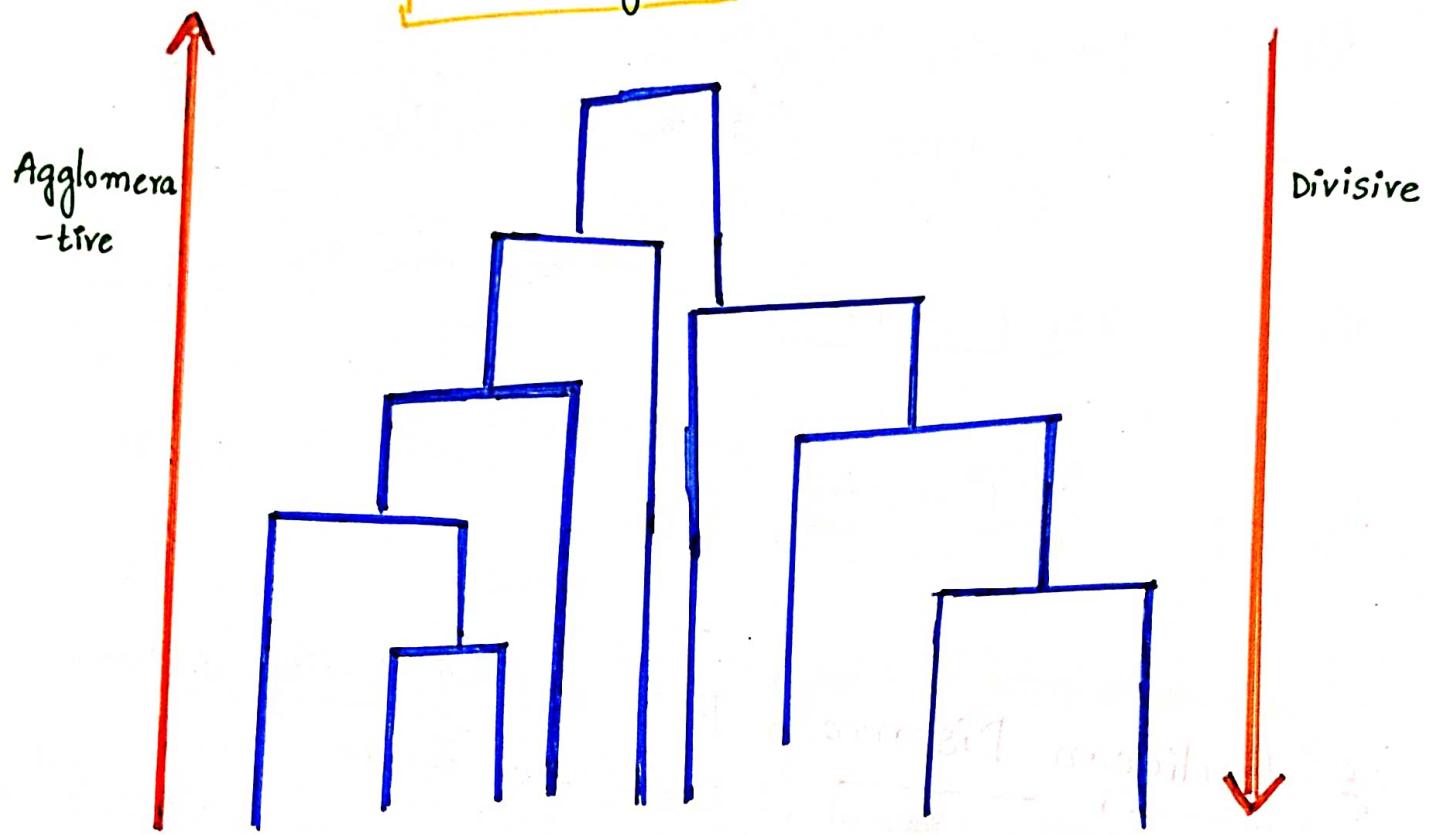
after HC



# Same as k-means but different process.

## \* Hierarchical clustering

# Dendrogram



\* They are two types clustering.

\* Agglomerative (From bottom to top)

\* Divisive (From Top to bottom)

Most of  
The time, we  
use, This  
Technique.

⇒ Steps of Hierarchical clustering of (Agglomerative)

Step : 1

Make Each data point a single point cluster  
(# that forms N clusters)

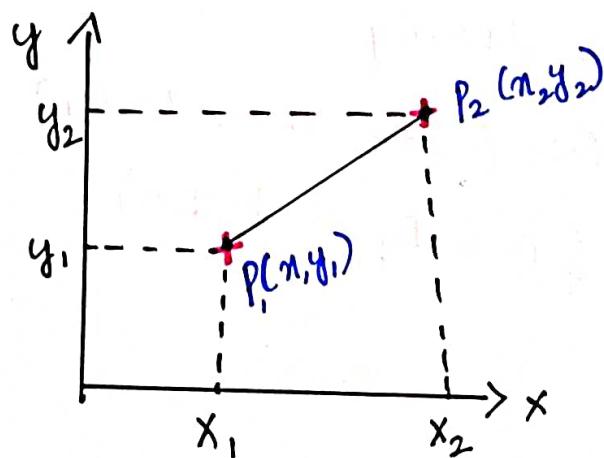
Step:2 Take the two closest data points and make them one cluster (# that forms  $N-1$  clusters)

Step:3 Take the two closest clusters and make them one cluster (# that forms  $N-2$  clusters)

Step:4 Repeat Step 3, until there is only one cluster

Step 5: Final model.

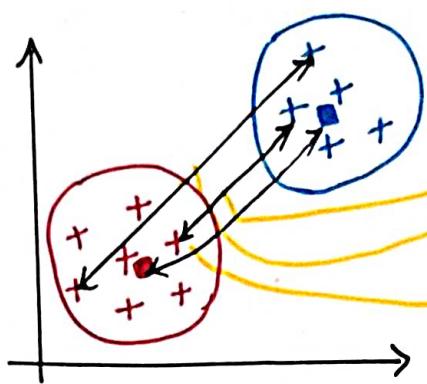
# Euclidean Distance :- it is used to identify distance between two points.



\* Euclidean distance between  $P_1$  and  $P_2$

$$= \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

## Distance between clusters

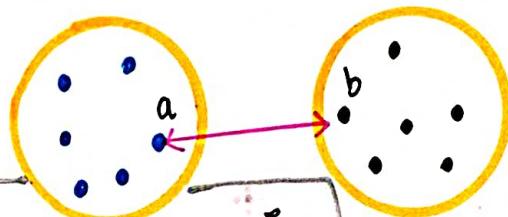


# Distance b/w two clusters.

- option 1 : closest points
- option 2 : furthest points
- option 3 : Average Distance
- option 4 : Distance between Centroids

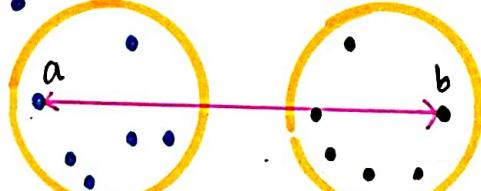
### \* Single-linkage clustering

- Minimum distance between clusters.



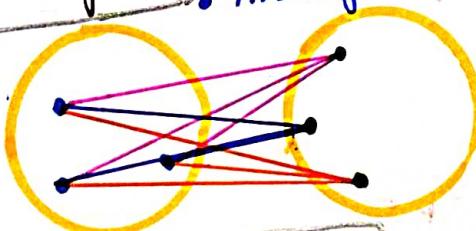
### \* Complete-linkage clustering

- Maximum distance between clusters



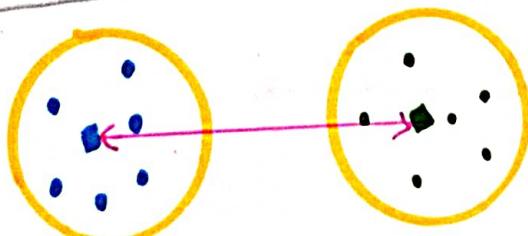
### \* Average linkage clustering

- Average distance between clusters



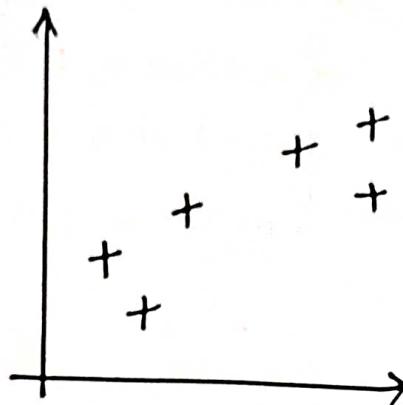
### \* Centroid Linkage clustering

- distance between cluster Centroids

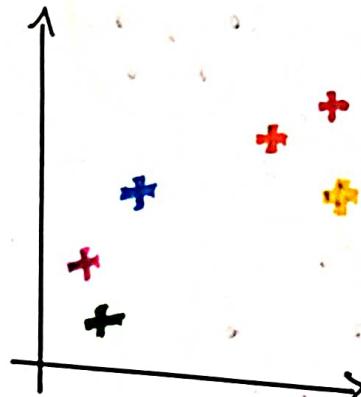


## # Agglomerative HC

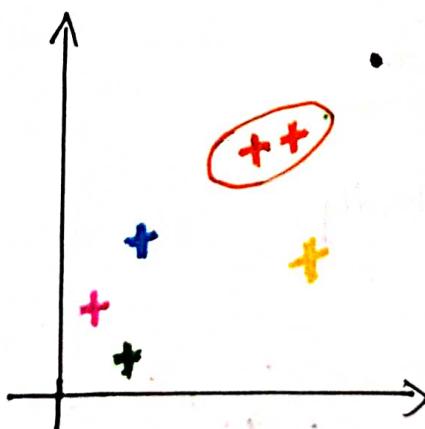
Consider the following dataset of  $N=6$  data points



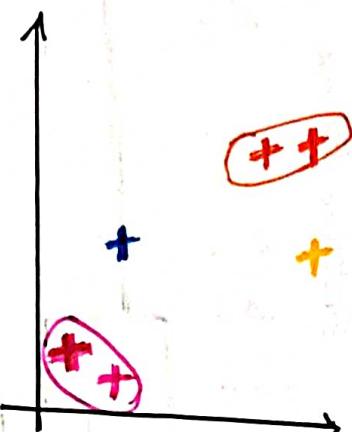
**Step : 1** (Make Each data point a single point cluster)  
that forms 6 clusters)



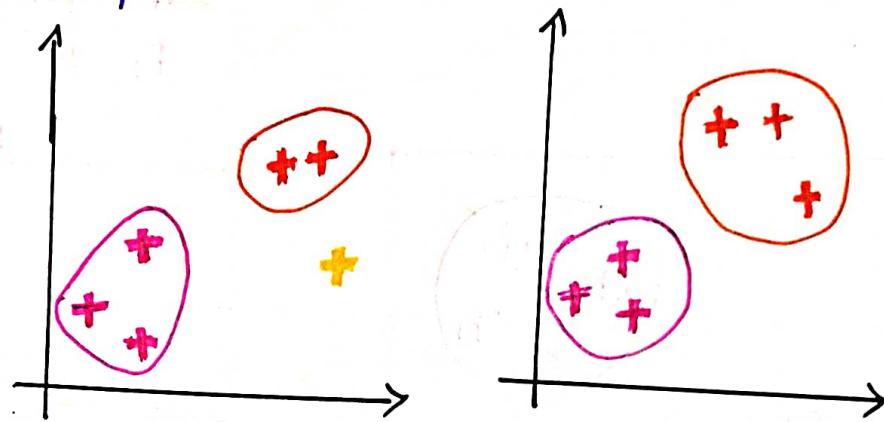
**Step : 2** (Take the two closest data points and make them  
one cluster → that forms 5 clusters)



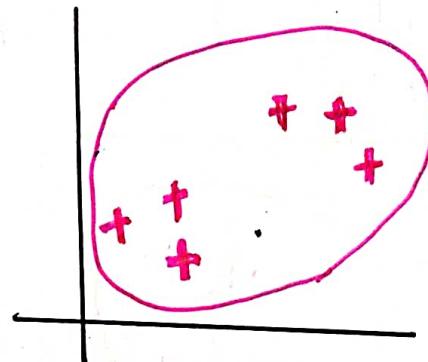
**Step 3 :** Take the two closest clusters and make them one cluster  $\rightarrow$  That forms 4 clusters ( $n-2$ )



**Step 4 :** Repeat step 3. until there is only one cluster.

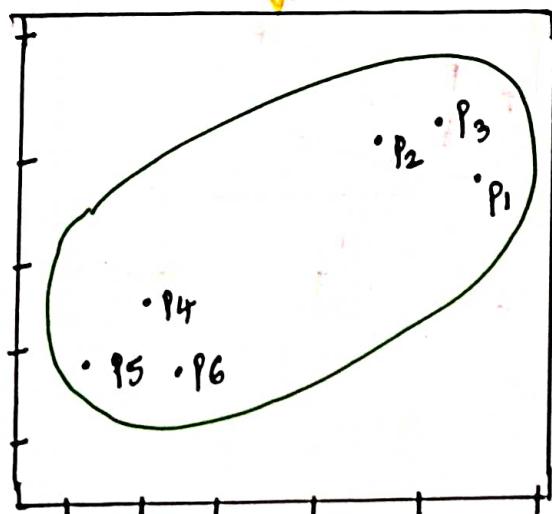
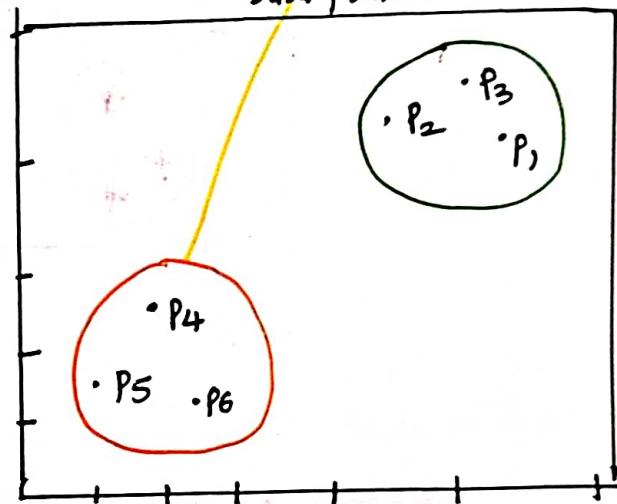
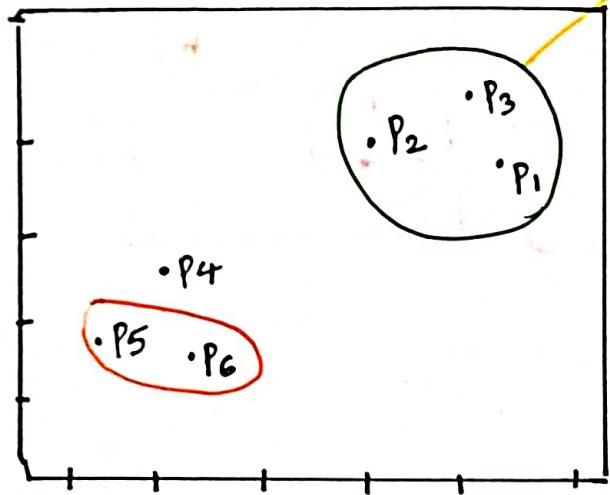
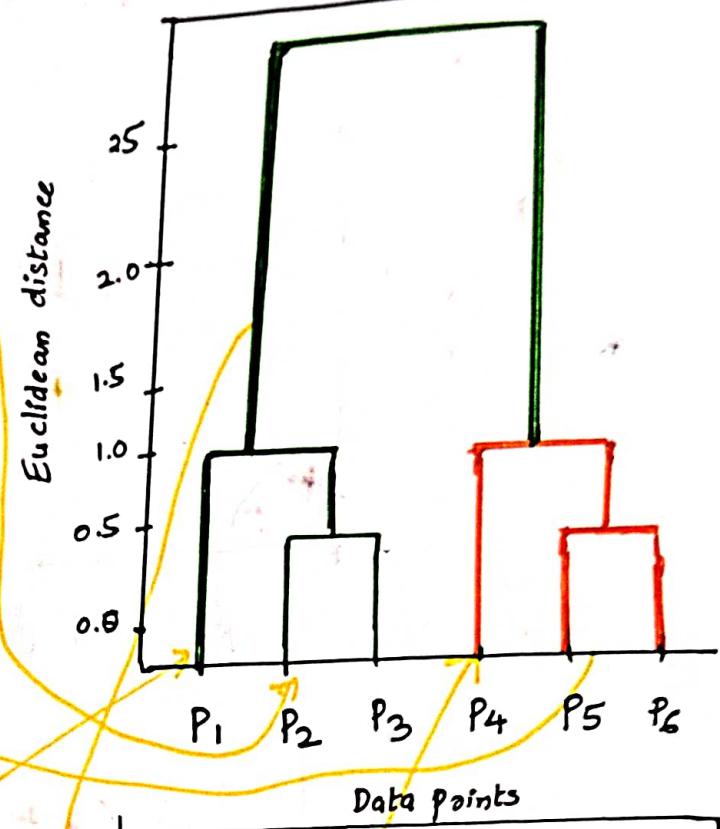
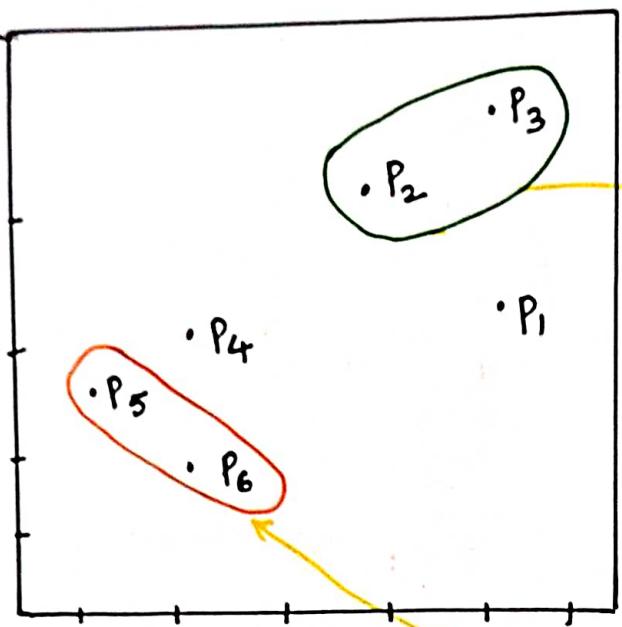


#final :



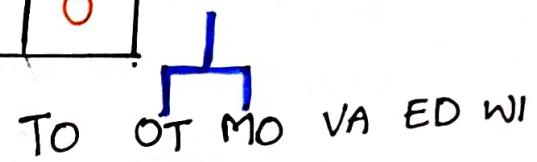
Download Part 02 & Part 03  
<https://t.me/AIMLDeepThaught/665>

# # How DO Dendograms Work ?



Example

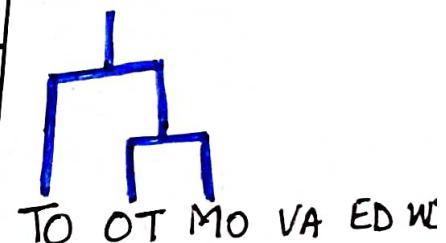
|    | TO | OT   | VA   | MO   | WI   | ED   |
|----|----|------|------|------|------|------|
| TO | 0  | 351  | 3363 | 505  | 1510 | 2699 |
| OT | 0  | 3543 | 167  |      | 1676 | 2840 |
| VA |    | 0    | 3690 | 1867 | 819  |      |
| MO |    |      | 0    | 1824 | 2976 |      |
| WI |    |      |      | 0    | 1195 |      |
| ED |    |      |      |      | 0    |      |



# - OT and MO (combined)

# Distances are recalculated again.

|       | TO | OT/MO | VA   | WI   | ED   |
|-------|----|-------|------|------|------|
| TO    |    | 351   | 3363 | 1510 | 2699 |
| OT/MO |    |       | 3543 | 1676 | 2840 |
| VA    |    |       |      | 1867 | 819  |
| WI    |    |       |      |      | 1195 |
| ED    |    |       |      |      |      |



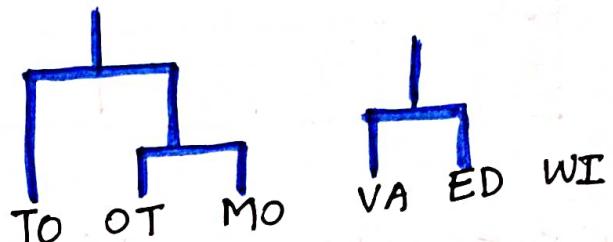
# OT/MO and To (combined)

# Distances are recalculated again, to make as one cluster.

|          | To/OT/MO | VA   | WI   | ED   |
|----------|----------|------|------|------|
| To/OT/MO |          | 3543 | 1676 | 2840 |
| VA       |          |      | 1867 | 819  |
| WI       |          |      |      | 1195 |
| ED       |          |      |      |      |

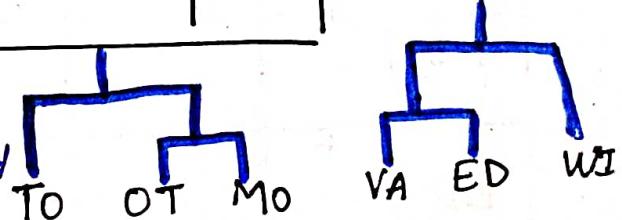
# ED and VA (combined)

# Distances are recalculated again



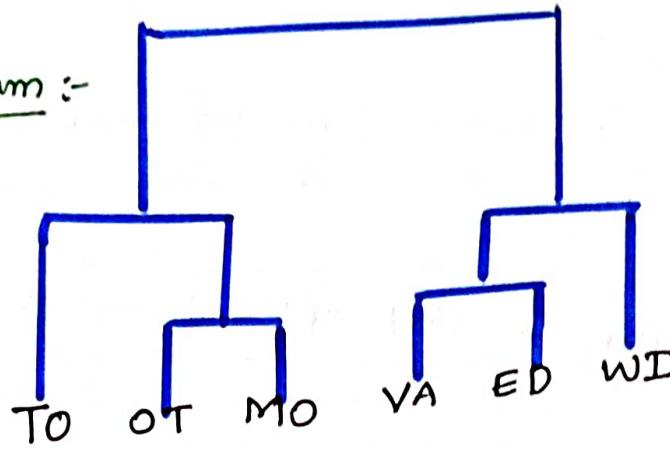
|          | To/OT/MO | VA/ED | WI             |
|----------|----------|-------|----------------|
| To/OT/MO |          | 2840  | 1676           |
| VA/ED    |          |       | 1667           |
|          |          |       | # value wrong. |

# WI and VA/ED (combined)  
# Distances are recalculated again

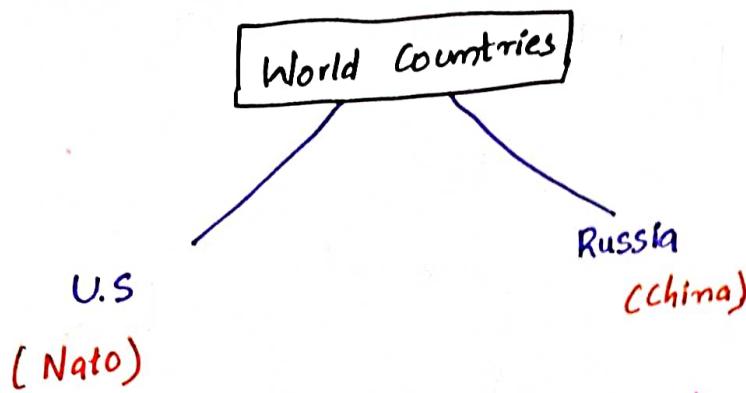


|          | To/OT/MO | VA/ED/WI |
|----------|----------|----------|
| To/OT/MO |          | 1676     |
| VA/ED/WI |          |          |

# Dendrogram :-



Ex:-



- # if U.S attack russia. it is going to identify which country is going to support russia.
- # which country is directly participate in war.
- # Which are very close to each other . it is going to identify.
- # this is hierarchical clustering .

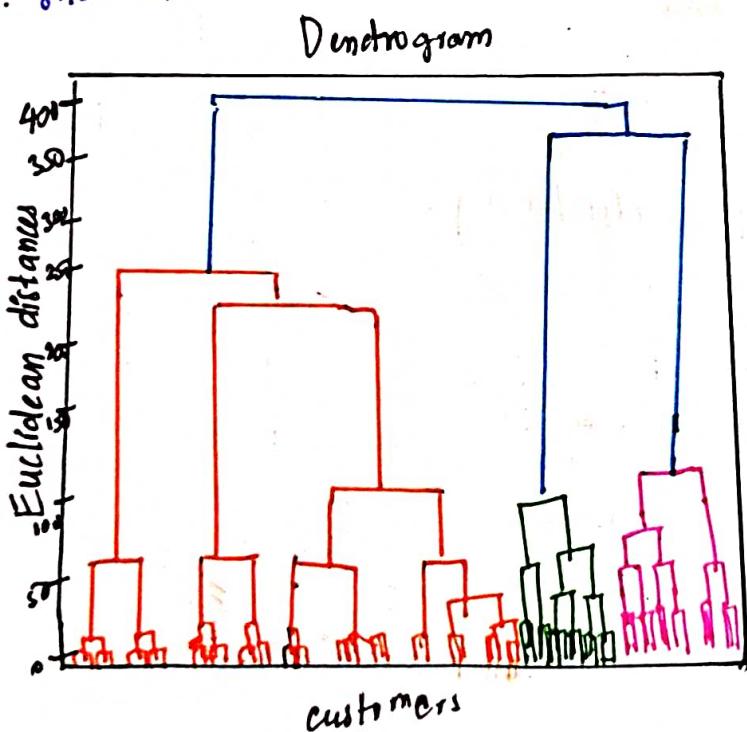
Download Part 02 & Part 03  
<https://t.me/AIMLDeepThought/665>

Code : Same data & Same steps till modelling

Using the dendrogram to find optimal no. of clusters.

import scipy.cluster.hierarchy as sch

```
# dendrogram = sch.dendrogram(sch.linkage(x,  
method = "ward"))  
# plt.title("Dendrogram")  
# plt.xlabel("Customers")  
# plt.ylabel("Euclidean distances")  
# plt.show()
```

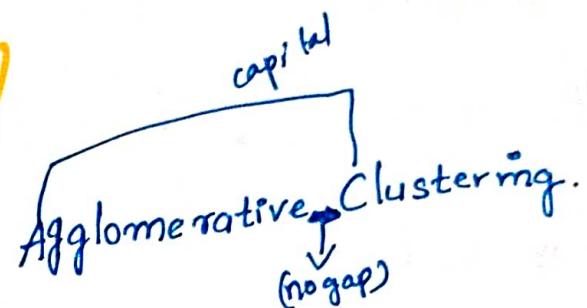


Q: How to identify optimal clusters during dendograms?

A:- biggest disadvantage with "HC" is we can't identify "Optimal no. of clusters". only in K-means we identify.

### Hierarchical clustering model :

from sklearn.cluster import



# hc = AgglomerativeClustering (n\_clusters = 5, affinity = "Euclidean", linkage = "ward")

### Predict

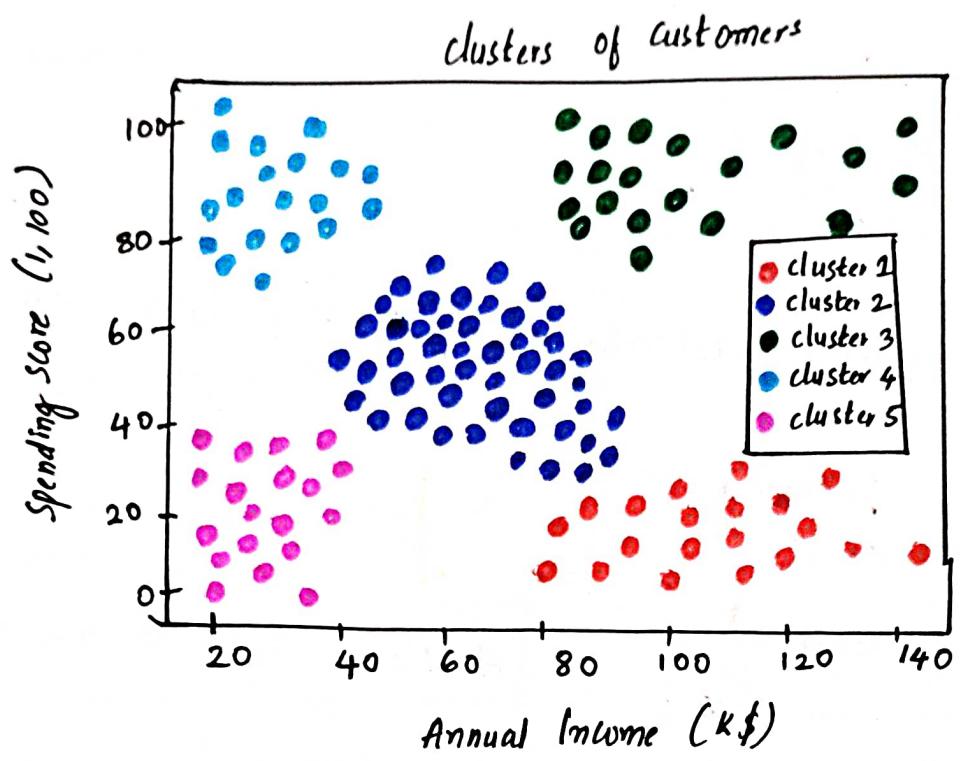
# y\_hc = hc.fit\_predict(x)

# y\_hc

[out]: array ([4, 3, 4, 3, 4, 3, ...,  
..., 1, 1, 1, 1, 1, 1, ...,  
..., 0, 2])

## # Visualising clusters

```
# cluster 1  
# plt.scatter(x[y-hc == 0,0], x[y-hc == 0,1], s=100,  
#               c = "red", label = "cluster 1")  
  
# cluster 2  
# plt.scatter(x[y-hc == 1,0], x[y-hc == 1,1], s=100,  
#               c = "blue", label = "cluster 2")  
  
# cluster 3  
# plt.scatter(x[y-hc == 2,0], x[y-hc == 2,1], s=100,  
#               c = "green", label = "cluster 3")  
  
# cluster 4  
# plt.scatter(x[y-hc == 3,0], x[y-hc == 3,1], s=100,  
#               c = "cyan", label = "cluster 4")  
  
# cluster 5  
# plt.scatter(x[y-hc == 4,0], x[y-hc == 4,1], s=100,  
#               c = "magenta", label = "cluster 5")  
  
# plt.title("clusters of customers")  
# plt.xlabel("Annual Income (K$)")  
# plt.ylabel("Spending Score (1-100)")  
# plt.show()  
# plt.legend()
```



?  
runny  
04/05/22

5:00 Am.

Download Part 02 & Part 03  
<https://t.me/AIMLDeepThought/665>

Dt: 3/05/22  
3: 30 AM

## \* UnSupervised Machine Learning:

# We don't have any output columns,  
Variable /, Feature  
↓ (we group them)

### Clustering :-

# Grouping of similar objects

1. K-means Clustering

2. Hierarchical clustering

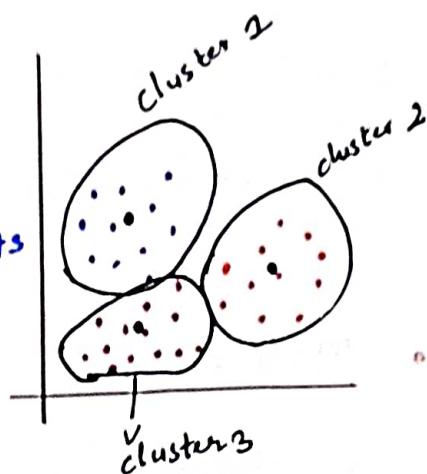
3. DB Scan clustering

# Unlabel data ←

Download Part 02 & Part 03  
<https://t.me/AIMLDeepThaught/665>

## 1. What is a cluster?

A:- A group of Objects that are "similar to other Objects" in the cluster, and dissimilar to data points in other clusters.



# groups :- segments

## 2. Clustering applications ?

### • Retail / marketing :

- \* Identifying buying patterns of Customers
- \* Recommending new books (or) movies to new customers.

### • Banking :

- \* Fraud detection in credit card use.
- \* Identifying clusters of customers (e.g. loyal)

### • Insurance :

- \* Fraud detection in claims analysis
- \* Insurance risk of customers.

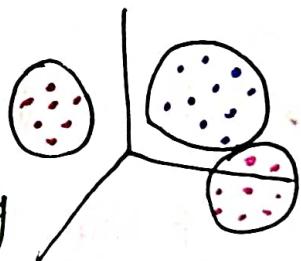
- Publication :
  - \* auto - categorizing news based on their content
  - \* Recommending similar news articles
- Medicine :
  - \* characterizing patient behaviour
- Biology :
  - \* clustering genetic markers to identify family ties.

3.

### clustering algorithms

- \* Partitioned-based clustering
  - Relatively Efficient

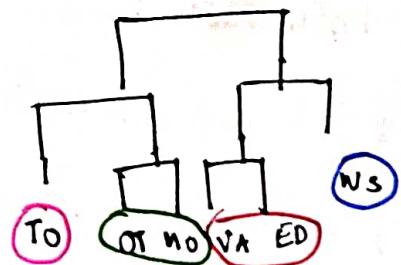
Eg:- K-means, K-median, Fuzzy C-means.



### Hierarchical clustering

- Produces trees of clusters

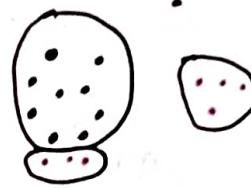
Eg:- Agglomerative, Divisive



## \* Density-based clustering

- Produces arbitrary shaped clusters

eg:- DBSCAN



## 1. K-Means.

- \* Partitioning clustering
- \* K-means divides the data into non-overlapping subsets (clusters) without any cluster internal structure
- \* records within a cluster are very similar

- \* records across different clusters are very different

Ex :- covid zones

- \* Determine the similarity ( $r$ )  
dissimilarity

[intra-cluster distances are minimized.]

- intra - within
- inter - outside

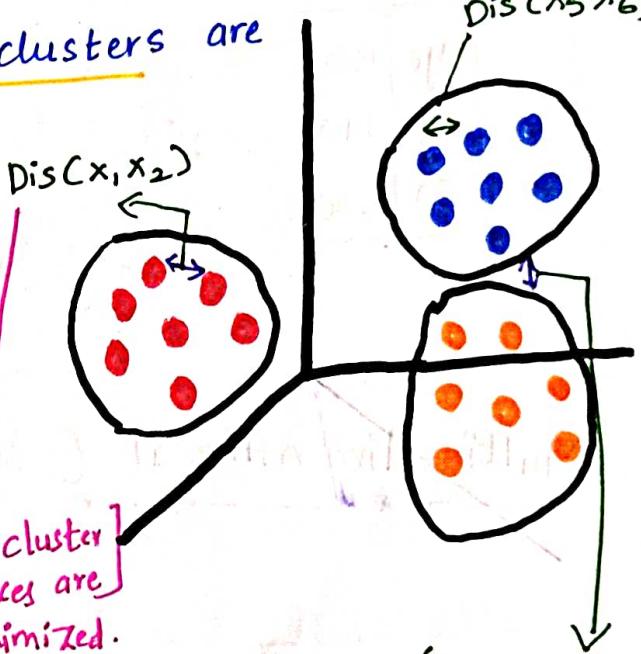
$\text{Dis}(C_2, C_3)$

[inter-clustering distances are maximized]

→ Aim:-

- \* Between two clusters, The distance should be maximum - Better The model

- \* Within cluster, The data point should be closely to each other - Better The model



## \* One-dimensionality (similarity / distance)

|   | Age |
|---|-----|
| 1 | 54  |
| 2 | 50  |

We, calculate Distance.

$$\sqrt{(54 - 50)^2} = 4.$$

$$Dis(x, x_2) = \sqrt{\sum_{i=0}^n (x_{1i} - x_{2i})^2}$$

# 3 dimensional - 3 columns

$$\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}$$
 (Euclidean)

## \* two-dimensionality (similarity / distance)

| Age | Income |
|-----|--------|
| 54  | 190    |
| 50  | 200    |

$$Dis(x, x_2) = \sqrt{(x_1 - x_2)^2}$$

$$= \sqrt{(54 - 50)^2 + (190 - 200)^2}$$

$$= 10.77$$

## \* Multi-dimensional (similarity / distance)

| Age | income | Educational |
|-----|--------|-------------|
| 54  | 190    | 3           |
| 50  | 200    | 8           |

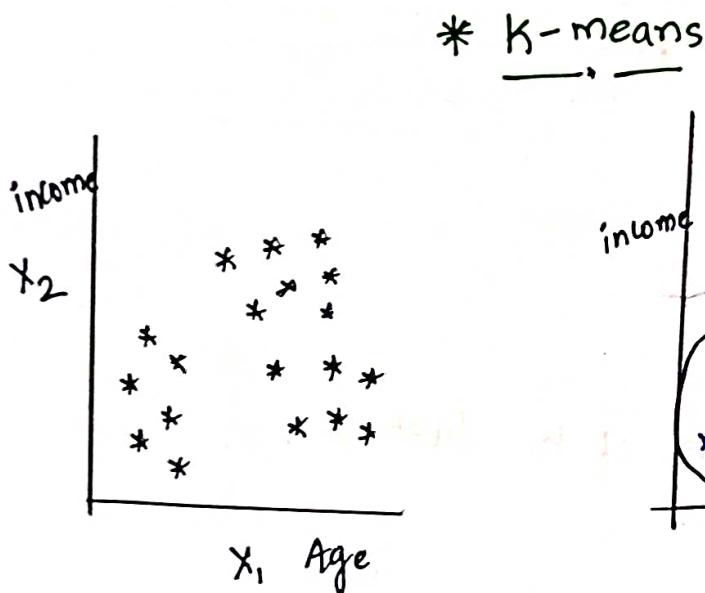
$$Dis(x, x_2) = \sqrt{(x_{1i} - x_{2i})^2}$$

$$= \sqrt{(54 - 50)^2 + (190 - 200)^2 + (3 - 8)^2}$$

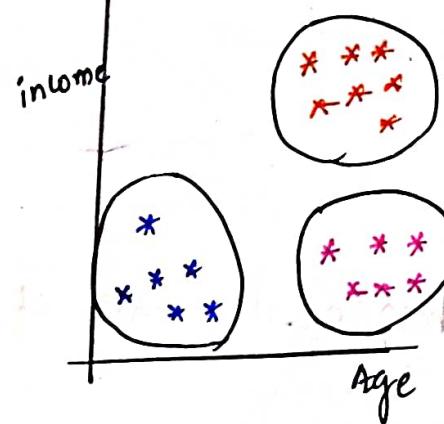
$$= 11.87$$

Q:- what is the difference b/w "K-means" & KNN?

A: Both use **Distance** formulas to group them.  
but, K-means is **clustering**, KNN-  
**classification**.  
Non supervised  
(only group)  
supervised  
(O/P) Predict



(normal scatterplot)



(# K-means)  
clustering.

\* Steps to K-means :

Step : 1 (choose The no. of "K" of clusters)

Step : 2 (Select at random "K" points, The Centroids)  
(not necessarily from your dataset)

Step : 3 (assign Each data point to closest Centroid  
→ (that forms "K" clusters)

↓  
Step: 4 (Compute & place new Centroid of Each cluster)

↓  
Step: 5 (Re assign Each data point to the new closest Centroid. if any reassignment took place. go to Step 4. Otherwise go to FIN

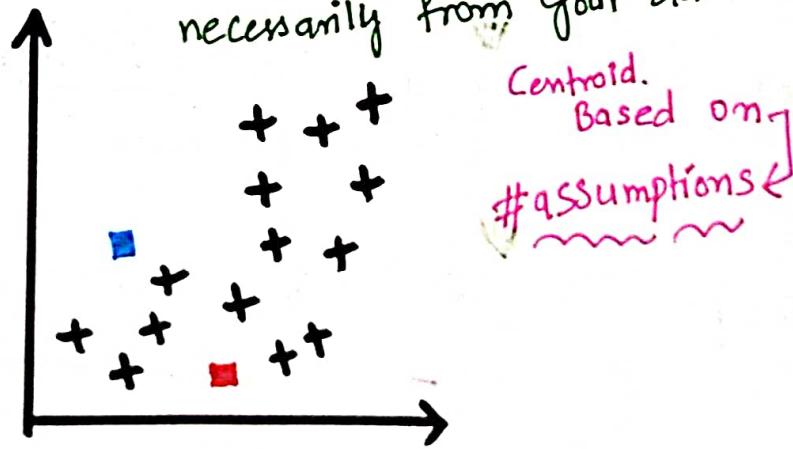
\* K-value is not Fixed... →

Your Model is Ready

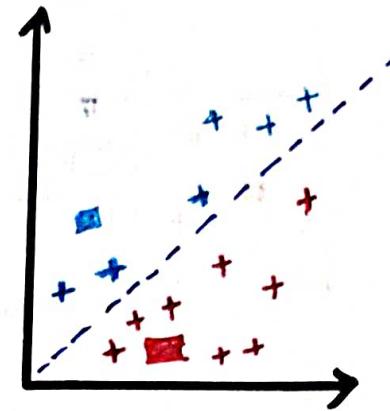
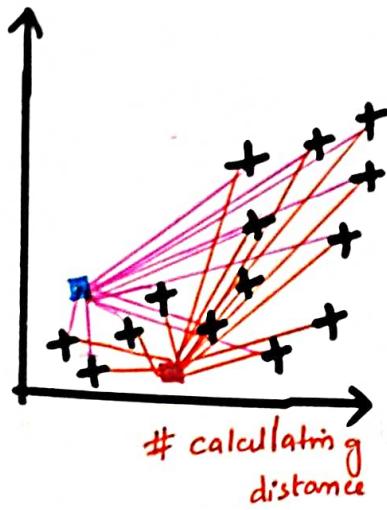
⇒ Step: 1: Choose the no. of K-clusters : 2



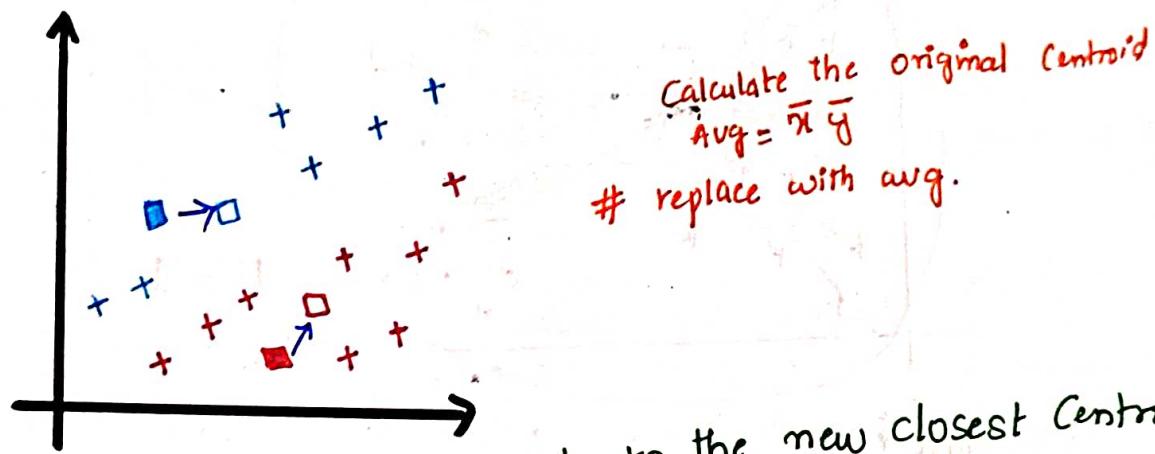
⇒ Step: 2 Select at random K-points, The centroids (not necessarily from your dataset)



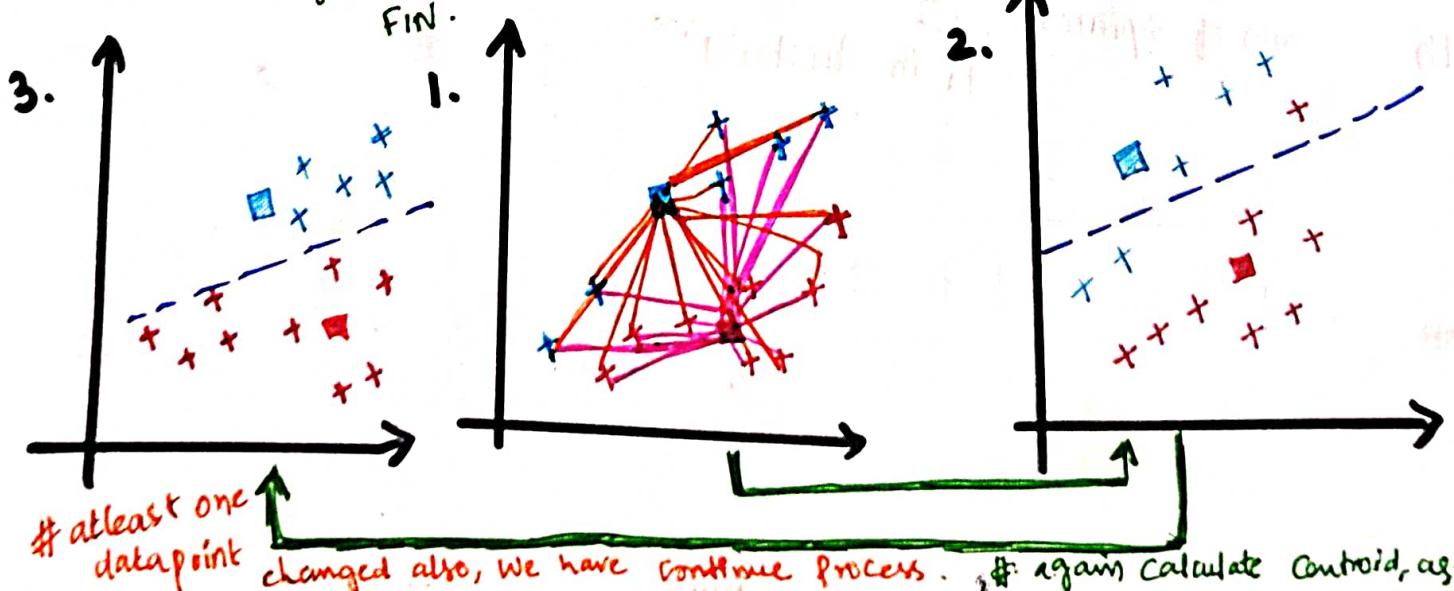
Step:-3 (assign Each data point to the closest Centroid  
that forms K-clusters)



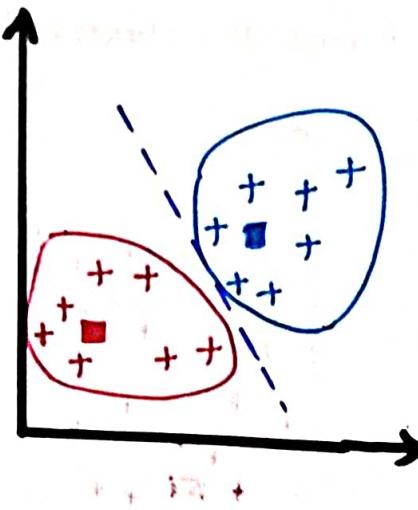
Step:4 Compute and replace The new centroid of Each other.



Step:5 Resign each data point to the new closest Centroid.  
if any reassignment took place, go to step 4. Otherwise go to

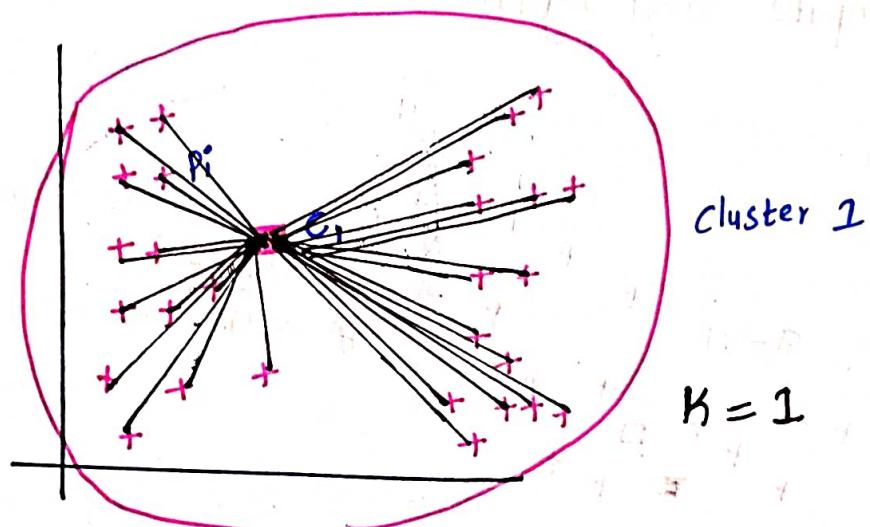


FINAL : Model is Ready.



Que :- How to choose right number of clusters?

A :-

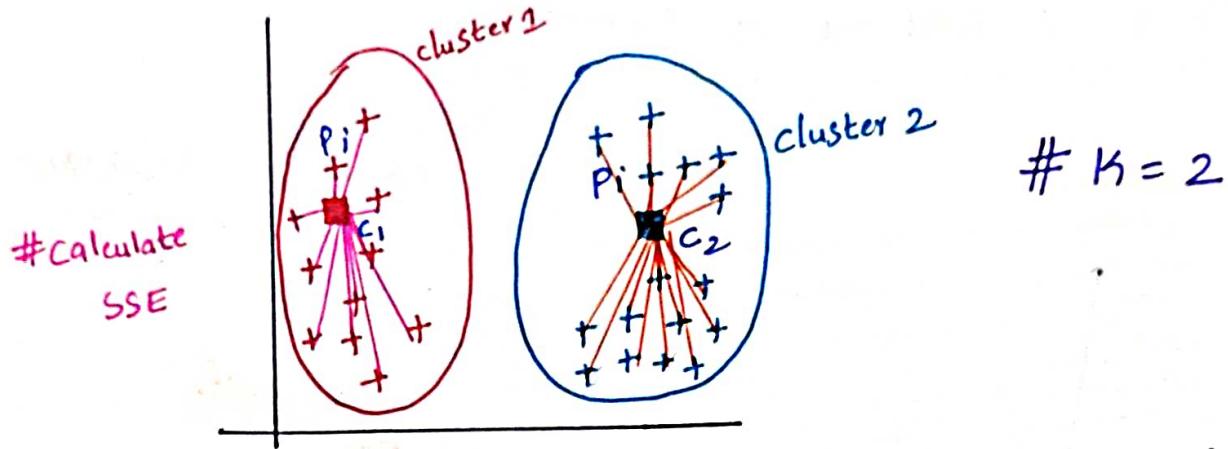


\* WCSS  $\rightarrow$  (with cluster Sum of Squares)

$$\text{Sum of Squares} = \sum_{P_i \text{ in cluster } 1} \text{distance}(P_i, c_1)^2$$

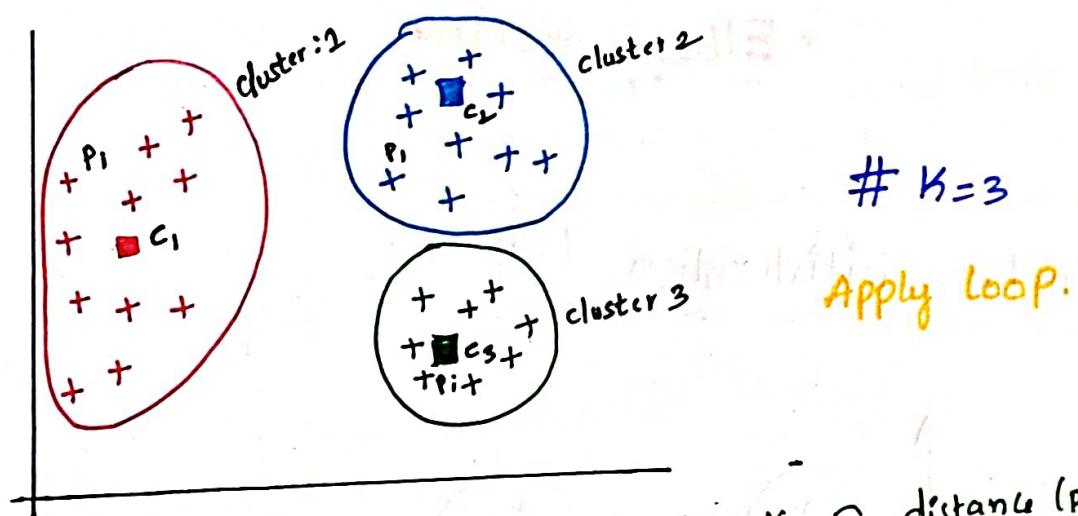
$$* \text{WCSS} = \text{distance}(P_i, c_1)^2$$

#  $K=1$



$$* \text{WCSS} = \sum_{\text{Pi in cluster 1}} \text{distance}(\text{Pi}, c_1)^2 + \sum_{\text{Pi in cluster 2}} \text{distance}(\text{Pi}, c_2)^2$$

(SSE) + (SSE)

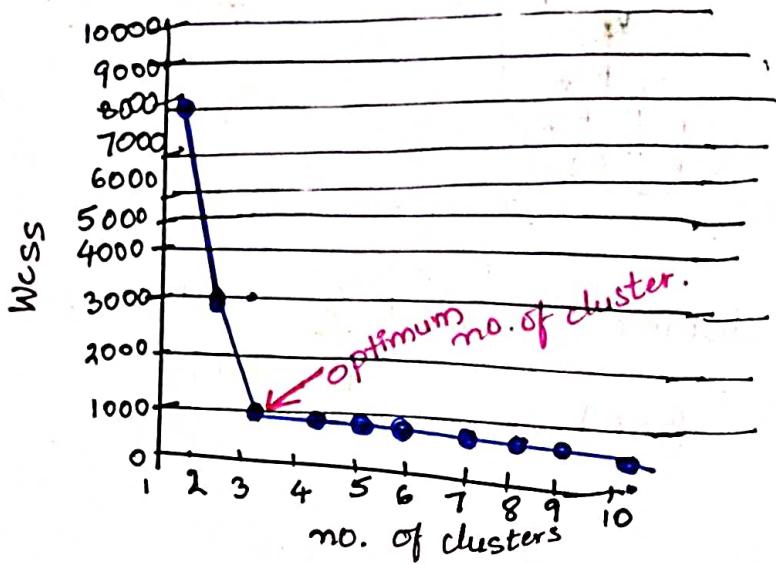


$$\text{WCSS} = \sum_{\text{Pi in cluster 1}} \text{distance}(\text{Pi}, c_1)^2 + \sum_{\text{Pi in cluster 2}} \text{distance}(\text{Pi}, c_2)^2 + \sum_{\text{Pi in cluster 3}} \text{distance}(\text{Pi}, c_3)^2$$

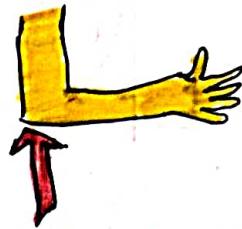
~~03/05/22~~  
03/05/22  
6:00 AM.

Download Part 02 & Part 03  
<https://t.me/AIMLDeepThaught/665>

## \* choosing The right no. of clusters

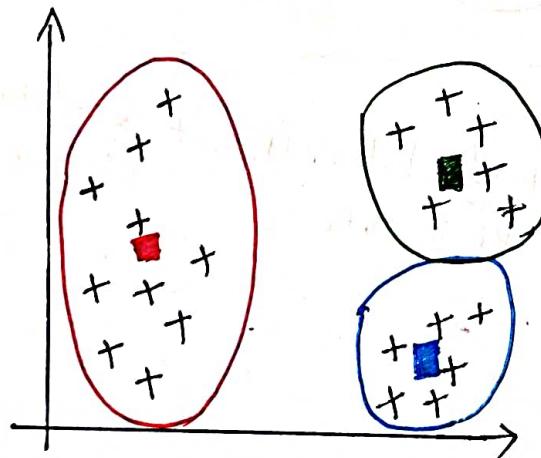


## \* Elbow Technique



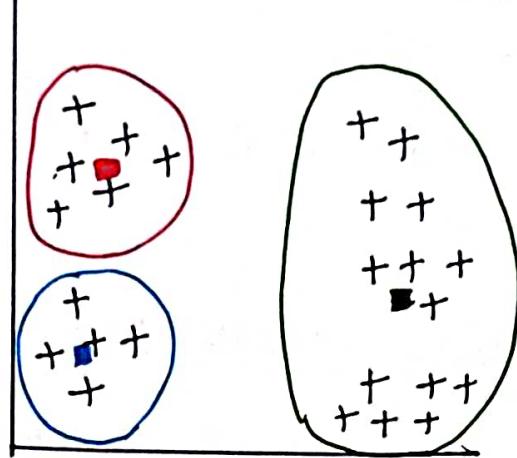
## \* Elbow method

## \* Random initialization trap



# randomly

Ques: what would happen if we had a bad random  
initialisation?



# different types of Clusters But, Data Set is common.

# Based on random initialization, it is going to change. clusters

If you choose wrong initialization. Total problem

Gets wrong. it is the problem in K-means.

A: Solution for this is K-means ++

⇒ The Random initialization Trap

One major drawback of K-means clustering is

Random initialisation of Centroids. The formation of clusters is closely bound by the initial position of a centroid.

The random positioning of Centroids can completely alter clusters and can result in random formation.

\* The Solution is K-means ++

\* K-means ++ is an algorithm that is used to initialise the K-means algorithm. Instead of "arithmetic mean", it takes "Weighted mean". For calculating centroid.

## K-means ++

1. Choose one Centroid uniformly at random from among the data points.
2. For each data point say  $n$ , Compute  $D(n)$ , which is distance between  $n$  and nearest Centroid that has already been chosen.
3. Choose one new data point at random as a new Centroid, using weighted probability distribution where a point " $n$ " is chosen with probability proportional to  $D(n)^2$ .
4. Repeat steps 2 and 3 until  $K$  Centres have been chosen.
5. Proceed with Standard K-means clustering.

Ex:-

| Age | Edu    | Exp |
|-----|--------|-----|
| 21  | B.Tech | 0   |
| 24  | B.Tech | 2   |
| 22  | B.Tech | 0   |
| 28  | M.Tech | 3   |

group

Encoding

→ grouping same Element (or) similar  
 Multiple feature overall  
 calculate distance formula

CODE :-

```
# df = pd.read_csv("Mall-Customers.csv")  
# df.head()
```

**Out :-**

| Customer ID | Genre  | Age | Annual income (K\$) | Spending score (1-100) |
|-------------|--------|-----|---------------------|------------------------|
| 1           | Male   | 19  | 15                  | 39                     |
| 2           | Male   | 21  | 15                  | 81                     |
| 3           | Female | 20  | 16                  | 6                      |
| 4           | Female | 23  | 16                  | 77                     |
| 5           | Female | 21  | 17                  | 40                     |

Download Part 02 & Part 03

<https://t.me/AIMLDeepThaught/665>

**Out :-** (200, 5)

# df.info()

**Out :-** 200 entries , 0 to 199  
column Non null count Dtype  
customerID 200 non null int64  
Genre 200 non null object  
Age " int64  
Annual Income " "  
Spending Score "

# df.isnull().sum()

**Out :-** 0.

# to understand, we take only two columns.

#  $x = df. iloc[:, [3, 4]]. values$

## MODEL

from sklearn.cluster import KMeans.

# using the elbow method to find the optimal no. of clusters.

# WCSS = []

for K in range (1, 21):

Kmeans = KMeans (n\_clusters = K, init = "K-means++")

Kmeans. fit (x)

WCSS . append (Kmeans. inertia\_)

- # WCSS values is called inertia

# WCSS

[out]: [ 269981. 2800 ... # K=1  
1811363. 5959 , # K=2  
106348. 373062,  
73679. 789039

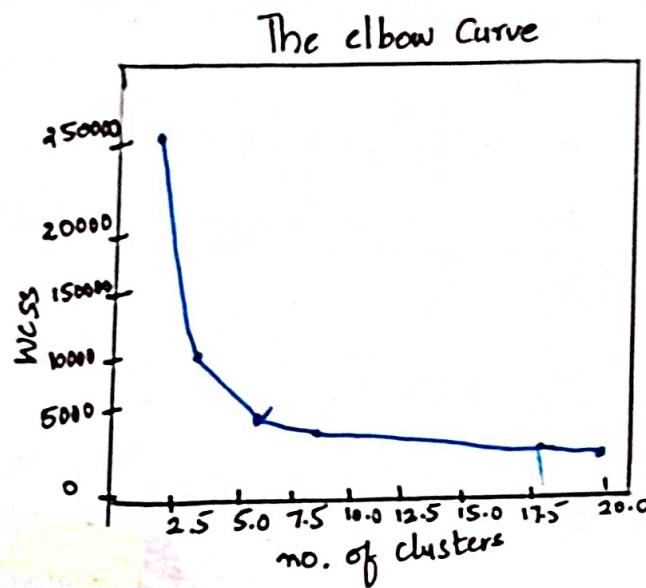
Download Part 02 & Part 03

<https://t.me/AIMLDeepThaught/665>

8504. 24137246]

# Elbow curve

```
# plt.plot(range(1,21), wcss)
# plt.title('The elbow method')
# plt.xlabel("no. of clusters")
# plt.ylabel("wcss")
# plt.show()
```



fit The K-means Model on the data.

```
# Kmeans = KMeans(n_clusters = 5, init = "K-means++",
random_state = 42)
```

# predict

```
# y_Kmeans = Kmeans.fit_predict(x)
```

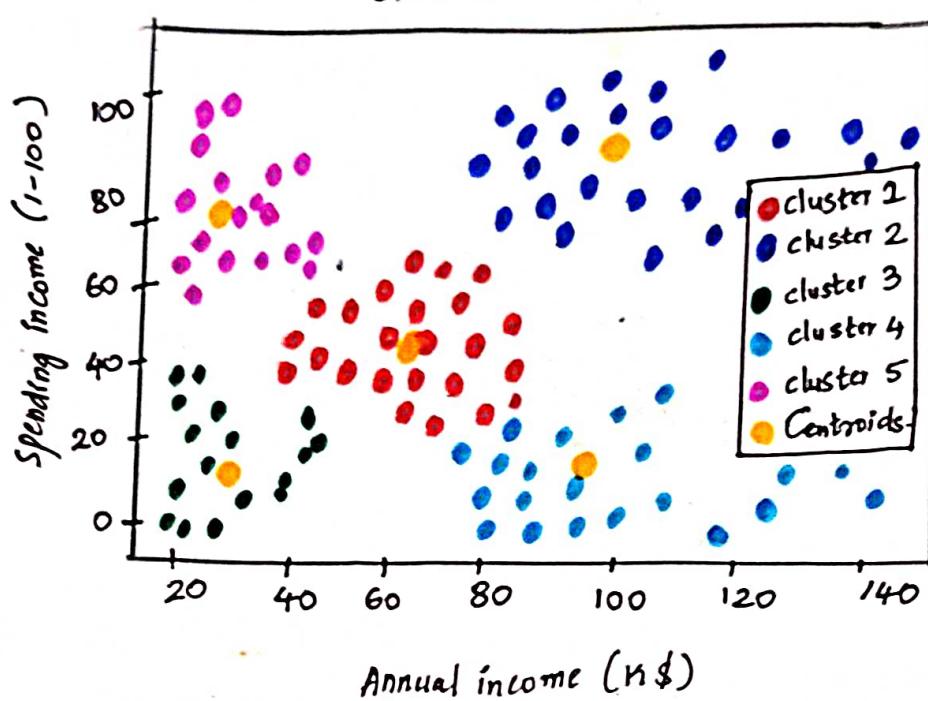
Download Part 02 & Part 03  
<https://t.me/AIMLDeepThaught/665>

## Visualising The clusters

```
# cluster 1  
# plt.scatter(x[y_kmeans == 0, 0], x[y_kmeans == 0, 1],  
#               s=100, c="red", label="cluster 1")  
  
# cluster 2  
# plt.scatter(x[y_kmeans == 1, 0], x[y_kmeans == 1, 1],  
#               s=100, c="blue", label="cluster 2")  
  
# cluster 3  
# plt.scatter(x[y_kmeans == 2, 0], x[y_kmeans == 2, 1],  
#               s=100, c="green", label="cluster 3")  
  
# cluster 4  
# plt.scatter(x[y_kmeans == 3, 0], x[y_kmeans == 3, 1],  
#               s=100, c="cyan", label="cluster 4")  
  
# cluster 5  
# plt.scatter(x[y_kmeans == 4, 0], x[y_kmeans == 4, 1],  
#               s=100, c="magenta", label="cluster 5")  
  
# centroid  
# plt.scatter(kmeans.cluster_centers_[:, 0],  
#             kmeans.cluster_centers_[:, 1],  
#             s=100, c="yellow", label="centroids")  
  
# plt.title("clusters of customers")  
# plt.xlabel("Annual income ('K $')")  
# plt.ylabel("spending score (1-100)")  
# plt.legend()  
# plt.show()
```

Out :-

clusters of customers.



## Download Part 02 & Part 03

# original data <https://t.me/AIMLDeepThaught/665>

```
# a = df.iloc[:, 3]
```

```
# b = df.iloc[:, 4]
```

```
# plt.scatter(a, b)
```

Join me on LinkedIn for the latest  
updates on Machine Learning:  
<https://www.linkedin.com/groups/7436898/>

Out :

