⇒ Machine learning Introduction

Que :- What is Machine Learning ?

Ans :- Machine learning is The SubField of computer Science that gives "Computer Ability to learn without being Explitcitly Programmed"

Example :-

(developer) Traditional Programming	Machine learning
a = 3 b = 5 → input c = a+b → code ↓ c = 8 → output input → code → output ↓ user computer	a = 3 b = 5 c = a+b c = 8 input (user) Machine gives relation between a, b and "c"

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

→ 6 (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"

$\therefore y = 10x \Rightarrow$ MODEL

* Substitute "6" in "x" value

$y = 10[6] = 60$

$\therefore y = 60.$

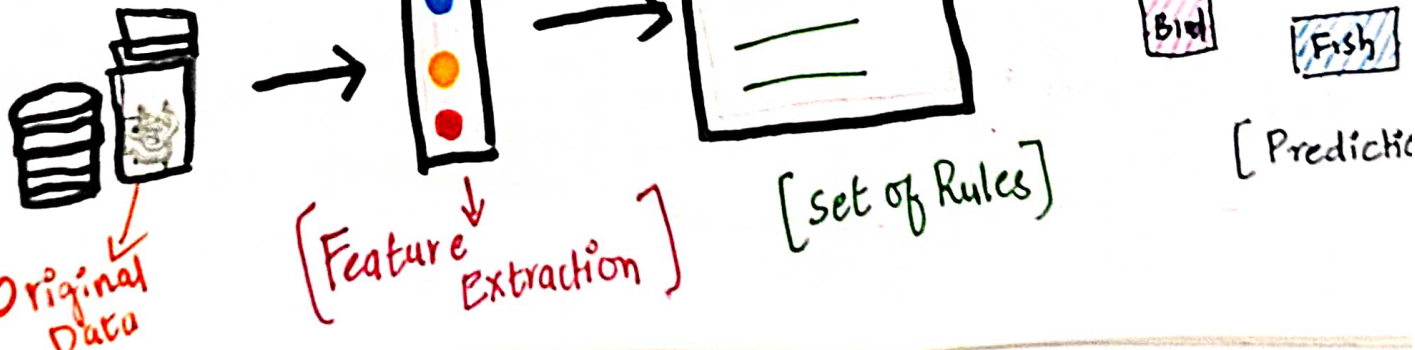
PREDICTIO

Machine learning

* Machine learn The given Data.
* Identifying The relation between Input and Output is called Machine Learning.

* Prediction

*ML Works :-



⇒ Major Machine Learning Techniques

* **Regression / Estimation**
  - Predicting **Continous 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** & **Unsual Cases**

* **Sequence Minning**
  - **Predicting next Events**; **Click stream** (Markov Model, HMM)

* Dimensional Reduction
  - Reducing The size of data [PCA]
* Recommendation systems
  - Recommending Items.

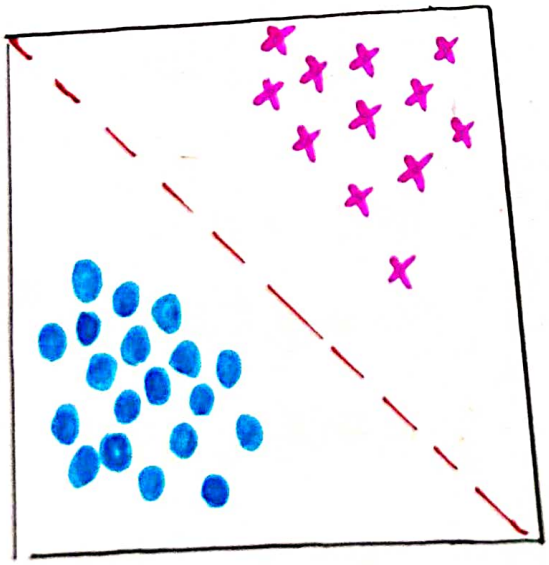
Que What is Supervised Learning ?

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

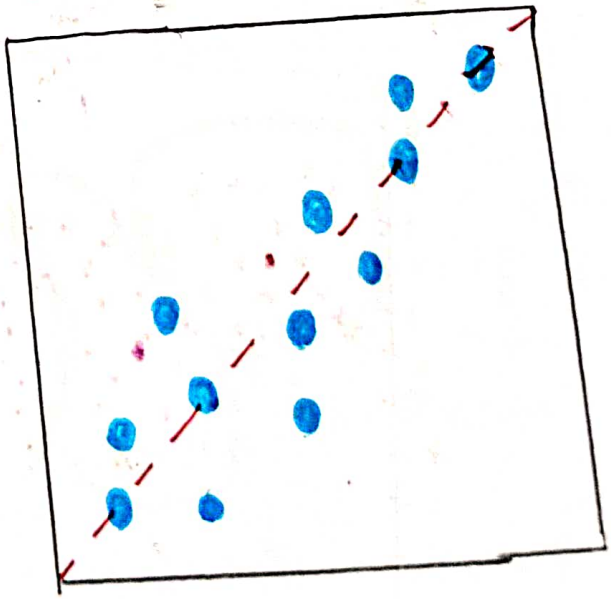
* Where Ever The Output Variable is Available is called as "Labeled data"

"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

Que :- What is UnSupervised Learning ?

Ans :- UnSupervised Learning, We have **Unlabed Data**

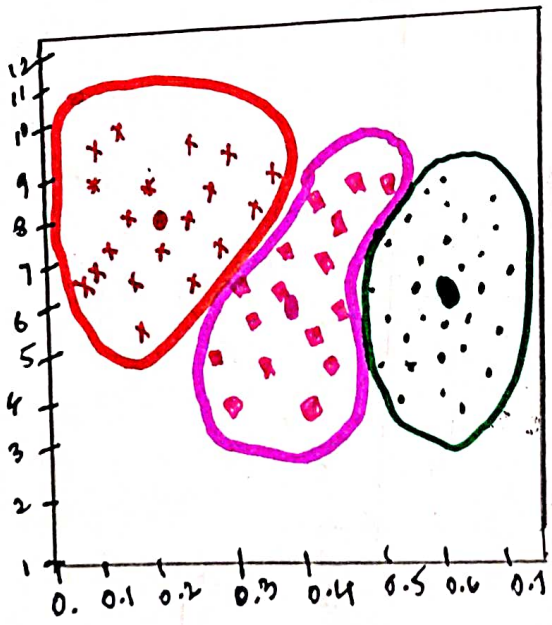
ie. **No Output Feature Available.**

"UnLabed Data" that the algorithm tries to make Sense of by "Extracting Features, Co-occurence and **Underlying patterns on its Own.**

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

Que : What is clustering ?

Ans: "clustering" is **grouping of data points** (or) **Objects that are Somehow Similar by**



* clustering

* Supervised Learning	* Unsupervised Learning
* Regression - Classifies labeled data * classification - Predicts trends using previous labeled data * Has more Evaluation methods than The Unsupervised Learning. * Controlled Environment	* 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 import SimpleImputer

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

Using Simple Imputer Pandas.

Data Frame

input, we giving Same

Out :- We get array.

Example :- User Defined Functions

```
def add(a,b)
    c = a+b
    return float(c)
```

→ dd (3,4)

Out : 7.0 → Float value

```
def add (a,b)
    c = a+b
    return c
```

→ add (3,4)

Out : 7

```
def maths (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
```

Jack → module, maths → Function

```
maths(3,4)
op:
Sum : 7
Sub = -1
Mul = 12
div = 0.75
```

* 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 Help function and see The Output Variable Type.

* If Out is Array. We change (or) Convert into Data Frame, Just writting [Pd.DataFrame] before The Function.

⇒ Estimator API

Modelling

import 'ML' Algorithm

model = MLAlgo ( )

model.fit [x_train, y_train)

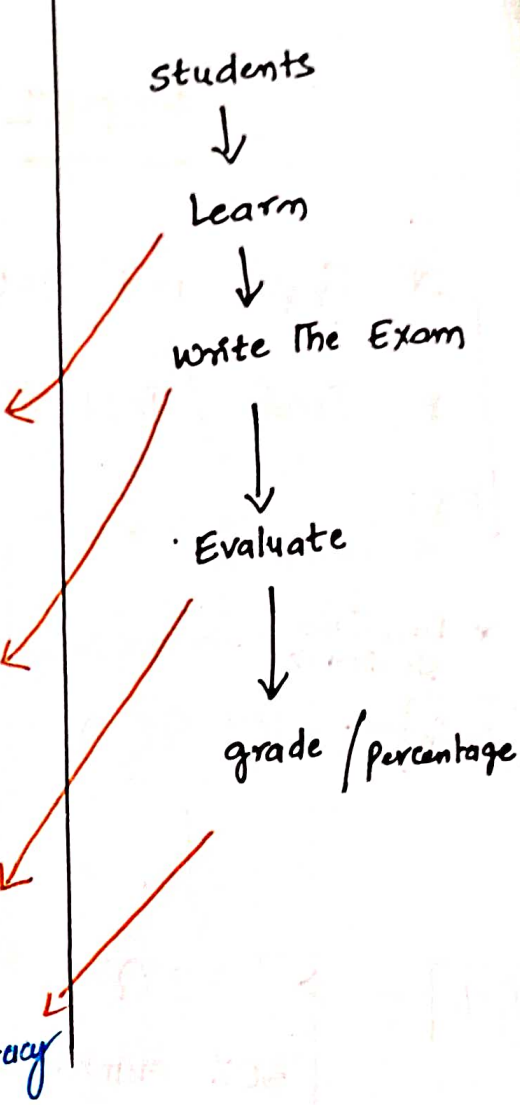
Prediction

ypred = model.predict (x_test)

Evaluation

Comparsion (y pred, y test)

Accuracy



Ex:-

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

Relation : $y = 10(x)$

ML. Algorithm [model]

model = Algorithm [y = 10x]

ypred # model. predict [x_test]

6

$y = 10x$ [x = 6]

$y = 10 \times 6$

Output = 60

* 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 (%)

Your a good student, But Only for this class Only.

* If what ever 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

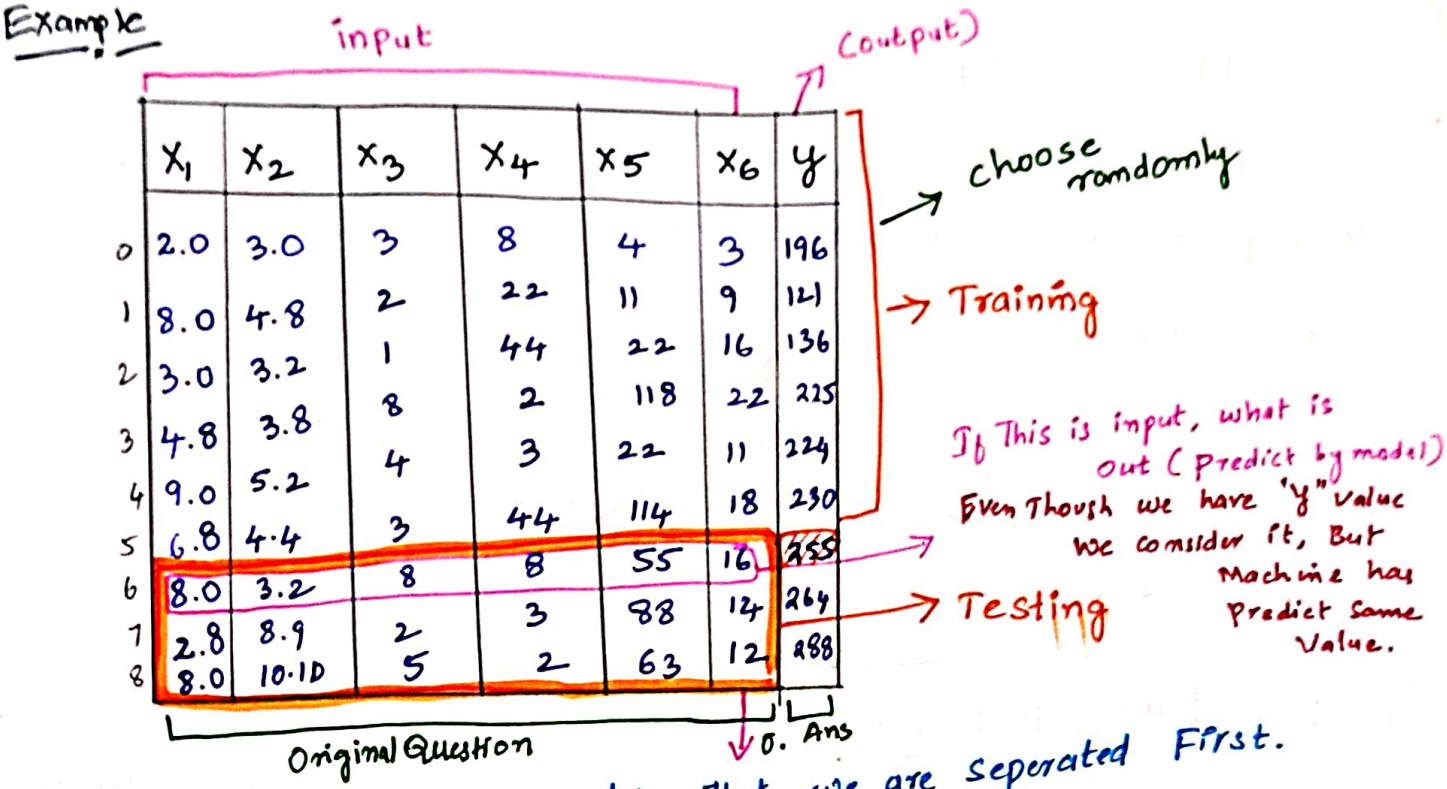
70 [Traning] 30 [Testing]

↓

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 Traning and For Testing, Seperately.

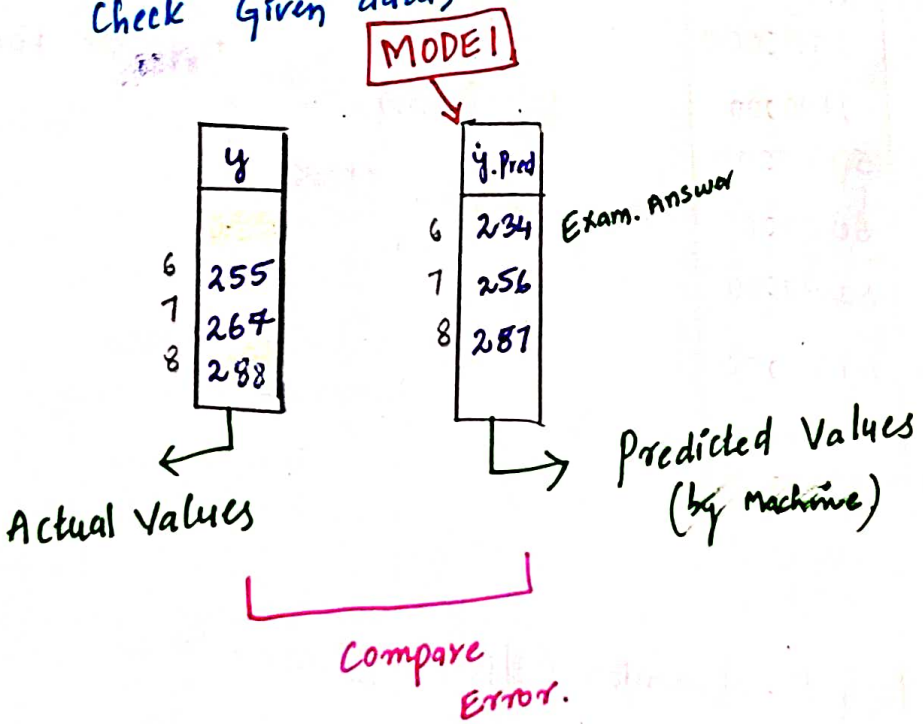
Example



	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

* In Testing, we give input data That we are seperated First.
and check with Accuracy of "y" value (output)

* In Testing, we don't take "y" Variable (output) because To Check Given data, Model is Working (or) not.



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 Dataset For Further steps.

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

Out

Area	Price	town_Chennai	town_Hydrabad
2600	5500000	1	0
3000	5650000	1	0
3200	6100000	1	0
3600	6900000	1	0
2600	5850000	0	0
2900	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

Now, this Changed data set, is we Consider Further to Train/test

square Foot

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

Independent Variables

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

price is Output Variable (y)

Dependent Variable

Because, it is Depending on "x" values To give Output.

option To Consider x & y Variable

```
x = df_dum.drop("price", axis = "columns")
y = df_dum.price
```

#Here we split The data into Train/test.

stores in "x" and "y".

check "x" → Input Variables (or) Independent Variables

```
x
```

Out:

	area	town_chennai	town_Hydrabad
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

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

```
y
```

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

Liberary — Module — (Function)

x_train, x_test, y_train, y_test = train_test_split (x, y, test_size = 0.3, Random_state = 29)

y → train y_train, Test y_test

Here x, y → Divided into two parts → are stored in x_train, x_test, y_train, y_test.

We Test on this → x

Spliting into two parts

Fixing The Same data point state. (To make data Constant)

Train 70% → x_train → 12 X 0.7 = 8.4 records (8 records)

Test 30% → x_test → 12 X 0.3 = 3.6 records (4 records)

Here 0.3 indicates 30%, we are giving To Test The data

"ML" Algorithm 1	"ML" Algorithm 2
* Same Data	* 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	2600	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

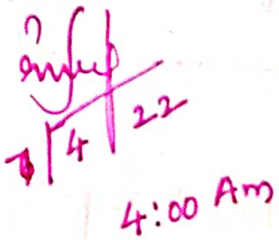
→ This Heading is not mentioned in my pe

→ Only These Values.

y_test

Out :-

	Price
9	6000000
4	5850000
10	6200000
11	69500000



Dt:- 7/4/22
12:30pm

Example :- # past Data

Input ← X | Y → Output

	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.

Step1 :- Split The Data For Train/test

	X_train		y_train
0	1	10	
1	5	15	
2	9	36	
9	36	52	
4	12	12	
5	4	88	
8	14	68	

	X_test		y_test
7	2	44	
3	14	45	
6	78	56	

Train 70% — Selected Randomly

Test 30%

Step2 :- Modelling

MODEL = 10x + 5

Step3 :- Prediction.

Train

MODEL = 10x + 5 ← Substituting x values in

X	Y	y_Pred_train
1		15
5		55
9		95
36		365
12		125
4		45
14		145

⇒ 10(1) + 5 = 15
⇒ 10(5) + 5 = 55
⇒ 10(9) + 5 = 95
⇒ 10(36) + 5 = 365
⇒ 10(12) + 5 = 125
⇒ 10(4) + 5 = 45
⇒ 10(14) + 5 = 145

Test

X	Y	y_Pred_test
2		25
14		145
78		785

⇒ 10(2) + 5 = 25
⇒ 10(14) + 5 = 145
⇒ 10(78) + 5 = 785

Step4 : Evaluation

Train

Y	y_Pred_train
10	15
15	55
36	95
52	365
12	125
88	45
68	145

difference blw Actual "Y" - Predicted "Y" = Error (anything)

Ex:- 10 − 15 = −5
15 − 55 = 40
36 − 95 = 59
⋮

Train_Error

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_train , Y_train

#relation → "MODEL"

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

⇒ PREDICTION

Substuting the **Values of "x"** in the given Equation and MODEL

MODEL & Calculating **"y" Values.**

⇒ Evaluation

Calculating the **error between** Predicted Values and Actual Values.

Good Model

#Train_Error = 5 % Accuracy = 95 %

#Test_Error = 5 % Accuracy = 95 %

[±5] with difference

⇒ train accuracy ≅ Test accuracy # Good Model.

with deviation of ±5 %.

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%.

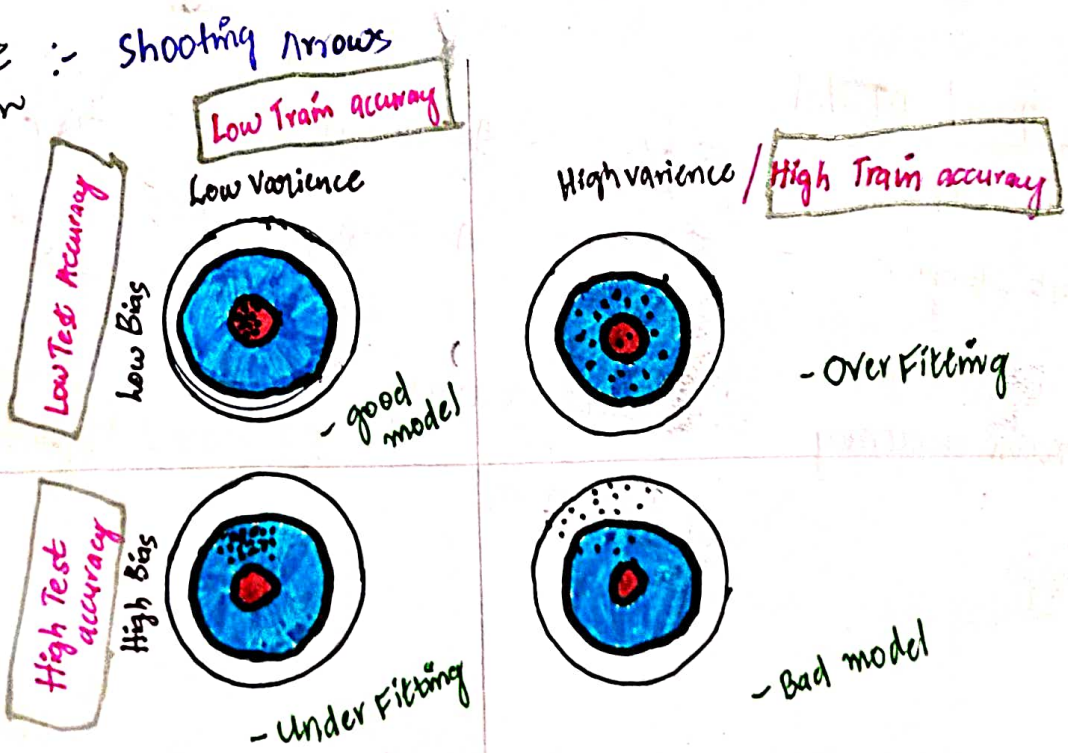
→ test accuracy = 90 %.
→ Test Error = 10 %.

[Model Performs bad in training Data. But Performs Good in Testing data]

train accuracy < test accuracy.

⇒ Bias / variance Trade-off → difference

Example :- Shooting Arrows



Variance = Distance b/w Data points / Train Error

Bias = close To Target / Test Error

***

Varience = train
Bias = test

Overfitting

→ High varience = 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

1/4/22
5:00 pm