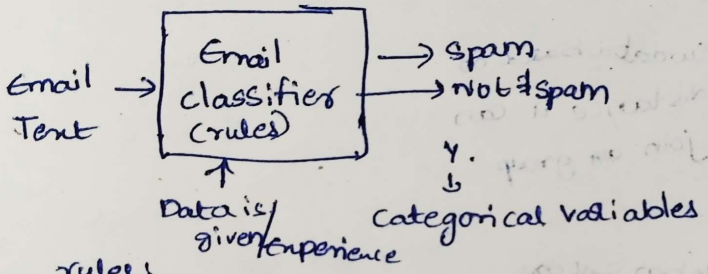
1. classification.
2. prediction
3. clustering
4. dimensionality reduction.

- pandas, Numpy, Matplotlib, Seaborn
- python
- Jupiter Notebook (Anaconda)
- Colab

Classification

Examples:

Email classification



rules:

if {
- domain email id is not repeated
- contains words like
  - discount, free, lottery, win.

then spam
}
else {
not spam.

→ Based on given data we need to write rules

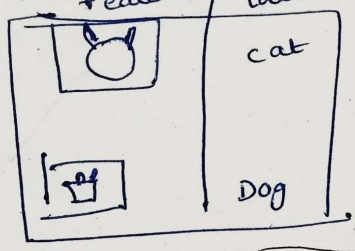
Data:-

Data can be of - supervised approach
- unsupervised approach.

(a) Supervised (x,y) : we have both x & y. Based on given data learning something if we have i/p & label is is supervised

x (Email) (feature)	Label
meeting at 10:00 am	not spam
won $ 1000.00	spam

Ex:- feature | label — Supervision.



features.

Supervision

label

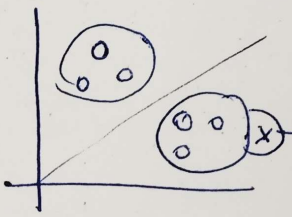
Ex:-

temp	humidity	Sunlight	wind	will it Rain tommorow

(b) UnSupervised Data:

→ have only x NO Y (X)

Ex:-1



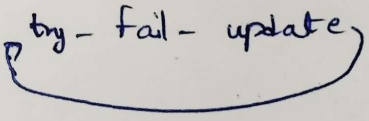
newdata based on distance it can join as group.

Ex:-2:

movie Recommendation System:

Language	Actor	Director	Genre	Songs	Budget

(c) Reinforce learning:



Learning by trying

(d) Semi supervised.

Dataset:-

(a) labelled
1) Supervised label

(b) unlabelled
1) unservised label.

Supervised learning

Splitting that data

1) Train
2) Test
3) Deployment

Train set	validation set or Development set	Test set

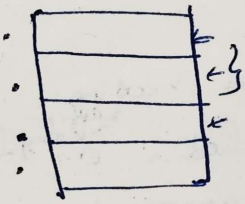
Data

Ex:- Train :- Solve examples
validation :- Solve exercise
Test :- Solve Exam Questions

: if total data is 100%, then train :- 50 or more than 50% data
validation :- 25 or 10/20
test :- 25 or 10/20.

Cross validation

- if dataset size is a small.



model: (x, y) True label

$x \rightarrow [f(x)] \rightarrow \hat{y}$ → predicted label.

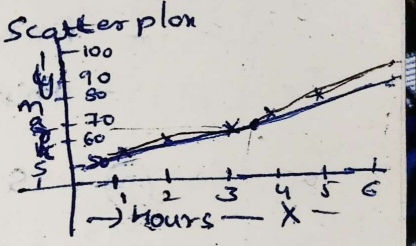
$\hat{y} = f(x)$ → model.

Regression:

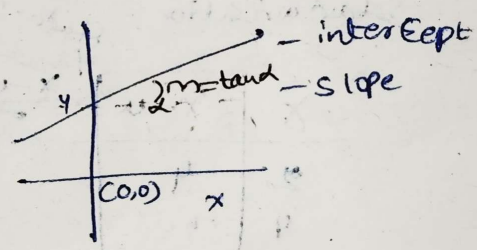
↳ simple linear Regression.

	x Hours	y marks
Train	1	50
	2	55
	3	65
	4	70
	5	80
validate	305	?

$x \rightarrow [f(x)] \rightarrow \hat{y}$



o/p i/p
$y = mx + c$
slope intercept



Data preprocessing steps

1) Read Dataset
2) Missing value
3) Encoding - Categorical data
4) Feature scaling

	nominal	ordinal
	colors	size

5) Train - val - test split

weight	weight (0-1) /100
50	0.5
60	0.6
70	0.7
80	0.8
100	1

- Missing

One Hot Encoding: (machine may give more importance based on label encoding)

label → Red - 0 (Label encoding based on number)
Green - 1
Blue - 2

So we use One Hot Encoder to not to prioritize a specific thing.

One Hot Encoder → one way of encoding

	R	G	B
Red	1	0	0
Green	0	1	0
Blue	0	0	1

(1) missing values
- fillna() - mean, median, mode
- dropna()
- imputer()

(2) Encoding - dictonary
- map
- Label Encoder
- one hot encoder (nominal)

(3) Scaling
- Standarization
- min max scaler

Standarization: Standar normal dist / normal distn

$z = \frac{x-\mu}{\sigma}$

X	X-μ
2	-4
4	-2
6	0
8	2
10	4
	0

mean = 0

variance = 1

Scaling :- converting everything in a smaller value or a specific scale based on x. we calcute

z value & Substitute all x with z so that all are in a range.

Min-Max Normalization: $x' = \frac{x - x_{min}}{x_{max} - x_{min}}$

x	$\frac{x - x_{min}}{x_{max} - x_{min}}$
2	$\frac{2-2}{8} = 0$
4	$\frac{4-2}{8} = 0.25$
8	$\frac{8-2}{8} = \frac{6}{8} = 0.75$
10	$\frac{10-2}{8} = 1$

min 2 max 10

Mean Normalization

$$x' = \frac{x - \mu}{x_{max} - x_{min}}$$

(4) ~~Split~~

Model

Data no. of study hours (x)	marks (y)	model 1 $\hat{y}$	$E_i = y - \hat{y}$	model 2 $\hat{y}$	y_i	$xiyi$	x^2
1	50	15	35	42	8	50	1
2	55	20	35	44	11	110	4
3	65	25	40	46	19	195	9
4	70	30	40	48	22	280	16
5	80	35	45	50	30	400	25
~~15~~	320		Total error = 195		Total error 90	1035	55

1) model 1, $\hat{y} = 10 + 5x$

2) model 2 : $40 + 2x$

error = $y - \hat{y}$

Training - finding the parameters that will give the minimum error

making all years positive

$$\text{error} = |y_i - \hat{y}|$$

$$\text{error} = (y_i - \hat{y}_i)^2$$ → amplifier the error.

$\hat{y} = \theta_0 + \theta_1 x$

Parameters θ_0, θ_1

$\hat{y} = 10 + 5x$

$SSE = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$

↓ Sum of Squared error.

find θ_0 and θ_1 such that error is minimum:-

(1) Using formul.

$$\theta = \bar{y} - \theta_1 \bar{x}$$

$$\theta_1 = \frac{n \Sigma x_i y_i - \Sigma x_i \Sigma y_i}{n \Sigma x_i^2 - (\Sigma x_i)^2}$$

$$\theta_1 \quad \frac{5 \cdot (1035) - (15)(320)}{5(55) - (15)^2} =$$

$$\theta = \frac{5175 - 4800}{275 - 225} = \frac{375}{50} = 7.5$$

$$\theta = 64 - (7.5)(3) = 41.5$$

$$\hat{y} = 41.5 + 7.5x$$

41.5 + 7.5

41.5
14
55.5

actual model

$\hat{y}$	ε_i
49	1
56.5	1.5
64	1
71.5	1.5
79	1

error=6

(1) model
- (has) parameter
- has

(2) Error
- (3) Train. — find parameters such that error is minimum.

What is Machine Learning

ML focusses on developing algorithms that can

- Learn from historical data
- Identify patterns and relationships with the data.
- predict outcomes or take actions on new, unseen data.

Key characteristics

- Data Driven: ML systems rely heavily on data to make decisions.
- Iterative: The more data an ML model processes, the better it gets
- Adaptative: ML Models can adapt adjust as the environment or data changes

Why is ML Important:

- ~~prey~~ personalised recommendations
- predictive maintenance in industries.
- Spam detection in emails.
- Medical diagnosis and image analysis.
- Autonomous vehicles and robotics.

ML helps business in

- Automate decision-making
- Reduce costs and improve efficiency.
- Discover insights that humans may overlook.

Types of Machine Learning

based on nature of data and tasks 3 types of ML

① Supervised Learning:

- Learning from labeled data (i/p & o/p pairs).

Ex:- predicting house prices based on size, bedrooms & location.

② Unsupervised learning:

- Learning from unlabeled data to identify patterns

Ex. Grouping Customers into segments based on spending habits

③ Reinforcement Learning:-

- Learning through trail and error to maximize rewards in an environment.
- Eg:- Teaching a robot to navigate a room by rewarding correct moves.

Types of Machine Learning problems.

(1) → Supervised :-

In supervised learning, the model learns a mapping b/w input features and their corresponding target labels. It is called supervised because the learning process is guided by labeled data.

- Goal: Train a model to make predictions on unseen data based on labeled training data.

Key concepts:

Input features (X): The independent variables

Target Labels (Y): The dependent variable (known outcomes)

Training Data: Labeled data used to train the model.

Testing Data: Data used to evaluate the model's performance.

Types:

1) Classification :- predicts a categorical label

Ex: email — spam or not spam (binary classification)

handwritten digits into classes :- (0-9)

customer ID	Age	Income	purchased
1	25	50,000	Yes
2	32	60,000	NO
3.	47	80,000	Yes
4	52	45,000	NO

2) Regression: predicts a continuous value

Ex: predicts the price of a house based on size and location

forecast the temp for next day.

house ID	Size	Bedrooms	Age	Price
1.	1500 sqft	3	10	3,00,000
2.	200	4	5	4,50,000
3.	1800	3	8	3,50,000

example Algo's

- Linear Regression
- Logistic Regression
- k-Nearest Neighbours
- Decision trees
- Random trees.

(2) Unsupervised Learning

⇒ the model learns patterns and structures from unlabeled data without any predefined labels or outcomes.

Goal: Discover hidden structures or patterns in data.

Key concepts:

- Input Features (X): Independent variables with no target labels
- Clusters: Groups of similar data points.
- Dimensionality Reduction: Simplifying data by reducing features

Types:

1. Clustering: Group data points into similar clusters.

ex: Segment customers into different groups based on their purchasing behaviour.

- Identify communities in a social nw

ex:

Customer ID	Age	Spending Score
1	22	85
2	40	15
3	35	60
4	28	77

2. Dimensionality Reduction:

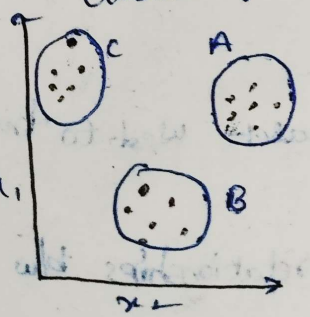
⇒ Reduce the number of features in data while retaining imp information.

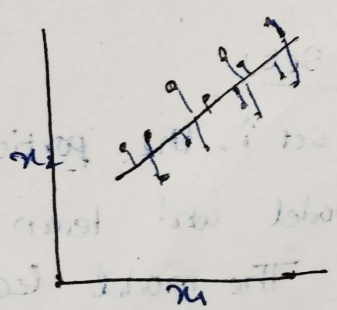
ex: Compress img data for faster preprocessing.

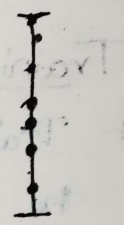
Sample ID	Petal Length	Petal width	Sepal Length	Sepal width
1	1.4	0.2	4.7	3.2
2	1.5	0.4	5.1	3.5
3	4.5	1.5	6.4	2.9

Goal:- use techniques like PCA to reduce these features into fewer dimensions while retaining important patterns.

clustering







Example Algorithms

1) K-means clustering.
2) Hierarchical clustering
3) Principal component Analysis (PCA)
4) t-SNE.

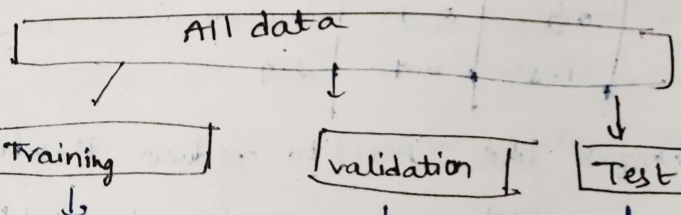
3) Reinforcement Learning:

Definition: RL is a type of ML where an agent learns to make decisions by performing actions in an environment to maximize cumulative rewards.

→ Differences b/w Supervised, unsupervised, Reinforcement Algos

Names	Definition	Example	Algos
1. Supervised	Learning from labeled data to make predictions	Spam detection, House price prediction	Linear Regression, Decision Trees.
2. Unsupervised learning.	Finding hidden patterns in unlabeled data	Customer segmentation, Anomaly detection	K-means, PCA, Hierarchical clustering.
3. Reinforcement learning	Learning by interacting with an env to maximize rewards	Game playing, Robotics, self-driving cars	Q-learning, DQN, policy Gradient

Evaluation:-



models learn the task

which model is best?

How good is this model truly?

1) Training Set:

Def:- Training set is the portion of dataset used to train the model and learn patterns.

Key Purpose:- → The model learns the relationships b/w i/p features and target labels.

- The model adjusts its parameters to minimize the error on this data.

example: from a 1k samples, 800 samples are used to train the model.

2. validation Set:

Def: The validation set is a portion of dataset used to tune the model's hyperparameters and asses performance during training.

Key purpose:

- It helps in identifying the model's ability to generalize before testing
- It is used for hyperparameter tuning (eg: adjusting learning rate, regularization strength, etc)
- prevents overfitting by providing feedback on intermediate model performance.

example: out of 1k, 100 are for reserved.

3. Test Set:

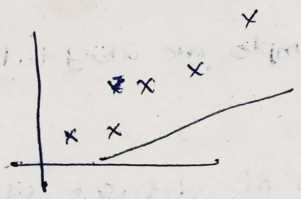
Def: ~~poor~~ portion of dataset reserved for final evaluation of model's performance on unseen data.

key purpose:

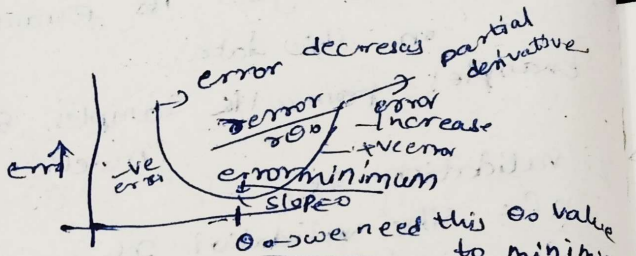
- provides an unbiased evaluation of trained model.
- Helps estimate the model's generalization performance on new data

ex:- from 1k, 100 will be for testing.

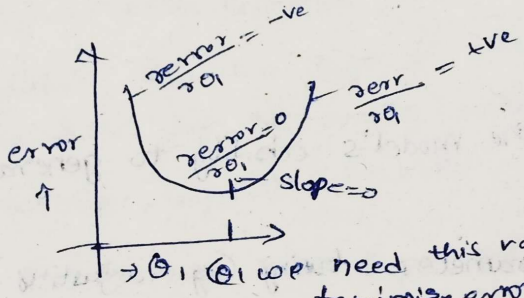
Minimizing the error

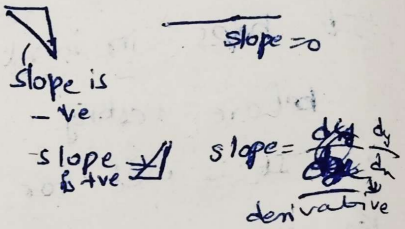


θ_0 θ_1



θ_1 is constant
θ is changing





if $\frac{dy}{dx}$ is -ve, -ve slope
if $\frac{dy}{dx}$ is +ve, +ve slope
$\frac{dy}{dx} = 0$; 0 slope.

Deriving formulas from intuition

→ $\hat{y} = \theta_0 + \theta_1 x$

find θ_0 and θ_1 such that $\frac{\partial error}{\partial \theta_0} = 0$ and $\frac{\partial error}{\partial \theta_1} = 0$

→ $SSE = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$ $\quad \hat{y} = \theta_0 + \theta_1 x$

$$\sum (y_i - \theta_0 - \theta_1 x_i)^2$$

$$\frac{\partial SSE}{\partial \theta_0} = \sum (y_i - \theta_0 - \theta_1 x_i)(-1)$$

$$= -2 \sum (y_i - \theta_0 - \theta_1 x_i)$$

$$= -2 \sum (y_i - \theta_0 - \theta_0 x_i) = 0$$

$$\sum (y_i - \theta_0 - \theta_1 x_i) = 0$$

$$\sum y_i - \sum \theta_0 - \sum \theta_1 x_i = 0$$

$$n\theta_0 = \sum y_i - \theta_1 \sum x_i$$

$$\Rightarrow \theta_0 = \frac{\sum y_i - \theta_1 \sum x_i}{n} \quad \text{—— (1) eq}$$

$$= \frac{\sum y_i}{n} - \theta_1 \frac{\sum x_i}{n}$$

$$\boxed{\theta_0 = \bar{y} - \theta_1 \bar{x}}$$ —— formula (1)

$$\frac{\partial SSE}{\partial \theta_1} = \frac{\Sigma (y_i - \theta_0 - \theta_1 x_i)^2}{\theta_1}$$

$$= 2 \Sigma (y_i - \theta_0 - \theta_0 x_i)(-x_i)$$

$$= -2 \Sigma x_i (y_i - \theta_0 - \theta_0 x_i)$$

$$-2 \Sigma x_i (y_i - \theta_0 - \theta_1 x_i) = 0$$

$$\Sigma x_i (y_i - \theta_0 - \theta_1 x_i) = 0$$

$$\Sigma x_i y_i - \theta_0 \Sigma x_i - \theta_1 \Sigma x_i^2 = 0$$

$$\Sigma x_i y_i - \left(\frac{\Sigma y_i - \theta_1 \Sigma x_i}{n}\right) \Sigma x_i - \theta_1 \Sigma x_i^2 = 0 \quad \rightarrow \text{from eq (1)}$$

$$\Sigma x_i y_i - \frac{\Sigma x_i \Sigma y_i - \theta_1 (\Sigma x_i)^2}{n} - \theta_1 \Sigma x_i^2 = 0$$

$$n \Sigma x_i y_i - \Sigma x_i \Sigma y_i + \theta_1 (\Sigma x_i)^2 - n \theta_1 \Sigma x_i^2 = 0$$

$$\Rightarrow n \Sigma x_i y_i - \Sigma x_i \Sigma y_i = -\theta_1 (\Sigma x_i)^2 + \theta_1 n \Sigma x_i^2$$

$$n \Sigma x_i y_i - \Sigma x_i \Sigma y_i = \theta_1 (n \Sigma x_i^2 - (\Sigma x_i)^2$$

$$\theta_1 = \frac{n \Sigma x_i y_i - \Sigma x_i \Sigma y_i}{n \Sigma x_i^2 - (\Sigma x_i)^2} \quad \text{— (2) formula}$$

Data Preprocessing

→ Data preprocessing is a data mining technique that involves transforming raw data into an understandable format.

→ Real world data is often incomplete, inconsitent, and lor lacking in certain behaviours or trends, and is likely to contain many errors. Data processing is a proven method of resolving such issues. Data preprocessing prepares raw for further processing

Common steps involved in Data Preprocessing:

1. Read the Dataset
2. Dealing with missing values
3. Dealing with categorical data
   - ordinal feature mapping
   - Label Encoding.

- one - Hot Encoding

4. Feature Scaling
5. Spliting dataset into training and testing sets.

1) Reading Dataset.

Pandas is a software library written for the python programming language for easy and high performance data manipulation and analysis.

```
df = pd.read_csv("weather.csv");
```

2) Dealing with Missing values

→ many real world datasets contains missing values, often encoded as blanks, NaNS or other placeholders.

Various ways to handle missing value of categorical ways.

1. Ignore observations of missing values if we are dealing with large data sets and less number of record has missing value.
2. Ignore variable, if it is not significant.
3. Replace with mean, median, mode (for numeric data)
4. Replace by most frequent value (for categorical data)
5. Missing values can be treated as a seperate category by itself (for categorical data)
6. Develop model to predict missing value.

We need to check whether dataset contains missing values or not.

```
# to check any columns contains & missing value.
df.isnull().any()
# to check count of missing values column wise
df.isnull().sum()
# to check count of missing value row wise
df.isnull().sum(axis=1);
# Deleting rows from dataframe.
x = x.drop([0,3], axis=0)
x = x.drop(1, axis=0)
```

Deleting columns from dataframe.

To delete rows and columns from DataFrames, pandas uses the drop() function.

The drop() function in Pandas be used to delete rows from a dataframe, with axis set to 0.

Imputation of missing values:

Computing the overall mean, median or mode is a very basic imputation method and it is very fast.

```
ent from sklearn.preprocessing import Imputer
    imputer = Imputer(missig_values = "NaN", strategy = "mean")
    numeric_values = x.iloc(:, 1:3)
    x.iloc[:, 1:3] = import.fit_transform(numeric_values),
```

Dealing with categorical data.

→ As we know scikit-learn models only numerical data. If any categorical values present in dataset we need to convert them into neumerics.

① nominal:

② Ordinal: - T-shirt size (XL > L > M)

Ordinal feature Mapping

we should convert the categorical string values into integers. However, since there is no convenient function that can automatically derive the correct order of the labels of our size feature, we have to define the mapping manually.

```
color = ['green', 'red', 'blue']
size = ['M', 'L', 'XL']
a = list(zip(color, size))
df1 = pd.DataFrame(data=a, colums=["color", "size"])
size_mapping = {'M': 1, 'L': 2, 'XL': 3}
df1['size'] = df1['size'].map(size_mapping)
df1
size_mapping.items();
```

Label Encoding (Nominal feature encoding)

Label Encoding is a utility class to help normalize labels such that they contain only values b/w 0 and n_classes-1. Because Scikit-learn's estimators treat class labels without any order, we can use LabelEncoder class to encode the string labels into integers.

```
# will not accept nan's
from sklearn.preprocessing import LabelEncoder
labelencoder_X = LabelEncoder()
X.iloc[:,0] = labelencoder_X.fit_transform(X.iloc[:,0])
labelencoder_Y = LabelEncoder()
Y = labelencoder_Y.fit_transform(y)
```

OneHotEncoding (Nominal feature encoding)

LabelEncoder transforms categorical feature to one new feature of integers (0 to n_categories-1)

one-hot encoding can be obtained with oneHotEncoder, which transforms each categorical feature with n_categories possible values into n_categories.

```
from sklearn.preprocessing import oneHotEncoder
onehot = oneHotEncoder(categorical_features=[0])
# make sure X has only numeric values
X = onehot.fit_transform(X).toarray().
```

Imputation with categorical data

we can handle in two ways

1. Replace by most frequent value (mode)
2. Missing values can be treated as a seperate category by itself.

```
# getting mode
md = X['outlook'].mode()
# filling missing values with mode
X['outlook'] = X['outlook'].fillna(md[0])
```

(or)

as a seperate category

x["outlook"] = x['outlook'].fillna('unknown').

Feature Scaling

→ Used to standardize the range of independent variables, also known as data normalization.

→ If one of the features has a broad range of values, the distance will be governed by particular features. Therefor, the range of all features should be normalized so that each feature contributes approximately proportionately to final distance.

→ Another reason is that gradient descent converges much faster with feature scaling that without it.

Feature scaling methods are

1. Standardization
2. Rescaling (min-max normalization)
3. Mean normalization
4. scaling to unit length.

Standardization:

→ Standardization is widely used for normalization in many ML algorithms (eg: support vector machine, logistic regression).

→ Feature Standarization makes the values of each feature in the data have zero-mean (when substracting the mean in numerator) and unit-variance.

X	$X - \mu$
2	-4
4	-2
6	0
8	2
10	4
	0

$\frac{30}{5} = 6$ $\quad \mu = \frac{30}{5} = 6$

mean = 0

variance = 1

Standardization:

Standardization is widely used for normalization

$$x' = \frac{x - \mu}{\sigma}$$

2. Min - Max Normalization

Scales the values b/w a given range (usually 0 to 1)

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}}$$

Given data : [10, 20, 30, 40, 50]

Min (x_{min}) = 10, Max (x_{max}) = 50

Original data (x)	Min-Max scaled (x')
10	0.00 $\quad \frac{10-10}{50-10} = 0$
20	0.25
30	0.50
40	0.75
50	1.00

3) Mean Normalization:

Scales the values b/w -1 and 1 using mean and range of data

formula $x' = \frac{x - \mu}{x_{max} - x_{min}}$

Given data : [10, 20, 30, 40, 50]

Mean (μ) = 30, Min = 10, Max = 50

original data (x)	Mean Normalize (x')
10	-0.67
20	-0.33
30	0.00
40	0.33
50	0.67

Implementing using scikit-learn

Standardization :

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X).
```

Min_Max Normalization.

```
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
X_scaled = scaler.fit_transform(X).
```

Mean Normalization

$X\_scaled = (X - X_{mean}(axis=0)) / (X.max(axis=0) - X.min(axis=0))$

Regression

Regression is a fundamental technique in ML used to model relationships blw variables. It is widely applied to predict outcomes based on input data.

Regression can be broadly categorized into

1. Linear Regression
2. Polynomial Regression

1. Linear Regression:-

Linear Regression assumes a straight-line relationship blw the independent variable (s) and the target variable.

The general equation for a linear regression model is

$$\hat{y} = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

where $\hat{y}$: predicted value n : number of features

x_i : i^{th} feature value θ_j : The j-th model parameter (including the bias term θ_0 and the feature weights $\theta_1, \theta_2, \ldots, \theta_n$).

Simple Linear Regression

→ when there is only one feature ($n=1$), the eq simplies to

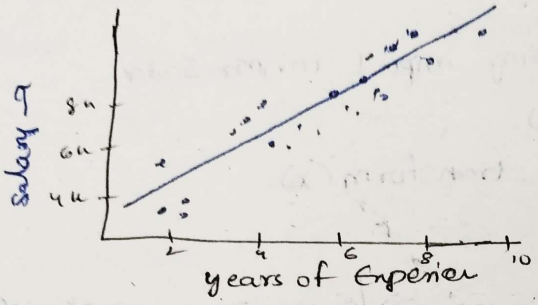
$$\hat{y} = \theta_0 + \theta_1 x_1$$

ex: a company uses experience as independent variable to predict Salary.

x_1 : Experience $\hat{y}$: predicted salary

- Goal: Understand how experience impacts salary.

Salary vs Experience



Derivation of SLR

SLR aims to model the relationship b/w one independent variable x and one dependent variable y. The r/s is

$$\hat{y} = \theta_0 + \theta_1 x$$

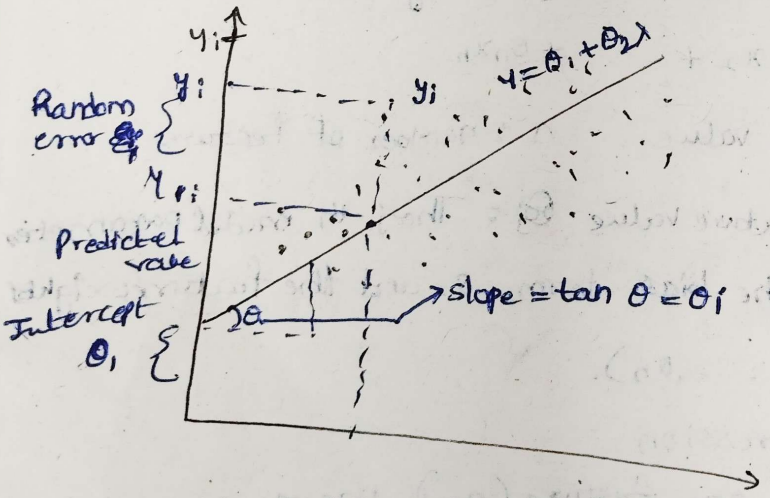
$\hat{y}$ = Predicted value x : independent variable

θ_0 : Intercept value of $\hat{y}$ when $x=0$ θ_1: Slope (rate of change of $\hat{y}$ with respect to x)

→ to determine values of slope θ_1 and intercept θ_0 that minimize the sum of squared residuals (errors).

1. Sum of Squared Residuals.

The residual (error) for each datapoint is the difference b/w the observed value y_i and the predicted value $\hat{y}_i$.

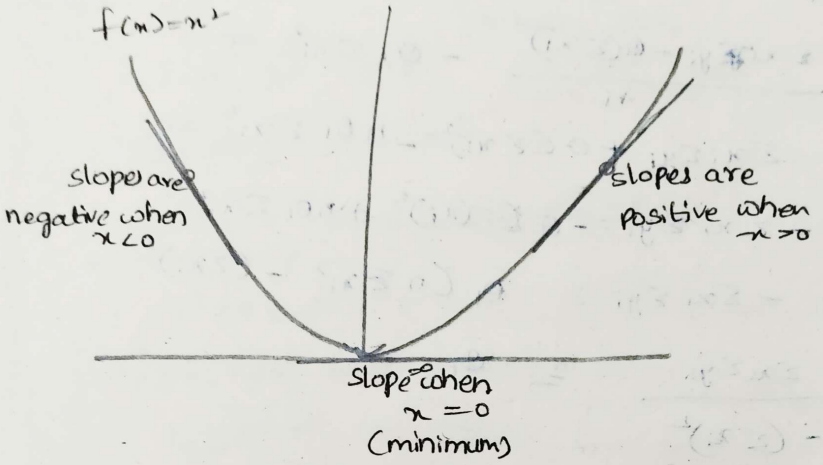


Sum of squared Errors (SSE)

$$SSE = \sum_{i=1}^{n} (y_i - (\theta_0 + \theta_1 x_i))^2$$

2. Deriving the slope (θ_1)

To minimize SSE, we differentiate SSE with respect to θ_0 and θ_1 and set the derivatives to zero.



Step 1 :- partial Derivative with Respect to θ_0.

→ Taking the derivative of SSE with respect to θ_0

$$\frac{\partial SSE}{\partial \theta_0} = \frac{\partial}{\partial \theta_0} \sum_{i=1}^{n} (y_i - (\theta_0 + \theta_1 x_i))^2$$

$$= -2 \sum_{i=1}^{n} (y_i - \theta_0 - \theta_1 x_i)$$

Set this derivative to zero.

$$\sum_{i=1}^{n} (y_i - \theta_0 - \theta_1 x_i) = 0$$

after Rearranging we get :

$$\Sigma y_i = n\theta_0 + \theta_1 \Sigma x_i$$

$$\theta = \frac{\Sigma y_i - \theta_1 \Sigma x_i}{n} \quad \text{— eq(1)}$$

$$\boxed{\theta = \bar{y} - \theta_1 \bar{x}}$$

Step 2 :- partial Derivative with Respect to θ_1,

Taking the derivative of SSE with respect to θ_1

$$\frac{\partial SSE}{\partial \theta_1} = -2 \sum_{i=1}^{n} x_i (y_i - \theta_0 - \theta_1 x_i)$$

Set this derivative to zero.

$$\sum_{i=1}^{n} x_i (y_i - \theta_0 - \theta_1 x_i) = 0$$

expanding & rearranging

$$\theta = \left(\frac{\Sigma y_i - \theta \Sigma x_i}{n}\right) \quad \text{eq (1)}$$

$$\Sigma x_i y_i = \theta_0 \Sigma x_i + \theta_1 \Sigma x_i^2$$

$$\Sigma x_i y_i - \left(\frac{\Sigma y_i - \theta \Sigma x_i}{n}\right)\Sigma x_i + \theta_1 \Sigma x_i^2 = 0$$

$$\Sigma x_i y_i - \frac{\Sigma x_i \Sigma y_i - \theta (\Sigma x_i)^2}{n} - \theta_1 \Sigma x_i^2 = 0$$

$$n\Sigma x_i y_i - \Sigma x_i \Sigma y_i + \theta (\Sigma x_i)^2 - n\theta_1 \Sigma x_i^2 = 0$$

$$n\Sigma x_i y_i - \Sigma x_i \Sigma y_i = -\theta \Sigma (x_i)^2 + n\theta_1 \Sigma x_i^2$$

$$n\Sigma x_i y_i - \Sigma x_i \Sigma y_i = \theta_1 (n\Sigma x_i^2 - (\Sigma x_i)^2$$

$$\frac{n\Sigma x_i y_i - \Sigma x_i \Sigma y_i}{n\Sigma x_i^2 - (\Sigma x_i)^2} = \theta_1$$

$$\boxed{\theta_1 = \frac{\Sigma x_i y_i - \Sigma x_i \Sigma y_i}{n\Sigma x_i^2 - (\Sigma x_i)^2}}$$

Regression:

$$\hat{y} = \theta_0 + \theta_1 x$$

$$\theta_0 = \bar{y} - \theta_1 \bar{x}$$

$$\theta_1 = \frac{n\Sigma x \Sigma y - \Sigma x y}{n\Sigma x^2 - (\Sigma x)^2}$$

$$\theta_1 = \frac{\Sigma (x_i - \bar{x})(y_i - \bar{y})}{\Sigma (x_i - \bar{x})^2}$$

X (indepet)	y (depet)
1	50
2	55
3	65
4	70
5	80

$$\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \end{bmatrix} \qquad \hat{y} = \theta_0 1 + \theta_1 x$$

$$\hat{y} = X\theta = \begin{bmatrix} 1 & 1 \\ 1 & 2 \\ 1 & 3 \\ 1 & 4 \\ 1 & 5 \end{bmatrix} \begin{bmatrix} \theta_0 \\ \theta_1 \end{bmatrix}$$

$$y = \begin{bmatrix} 50 \\ 55 \\ 65 \\ 70 \\ 80 \end{bmatrix}$$

$$X = \begin{bmatrix} 1 & 1 \\ 1 & 2 \\ 1 & 3 \\ 1 & 4 \\ 1 & 5 \end{bmatrix}$$

$$\theta = (X^T y^{-1}) X^T y$$

$$= \frac{2\times 5 \cdot 5 \times 2}{2\times 2} \quad \frac{2\times 5 \cdot 5 \times 1}{2\times 1}$$

$$2\times 1.$$

```
import numpy as np
x = np.array([[1,1],[1,2],[1,3],[1,4],[1,5]])
y = np.array([50,55,65,70,80])
theta = np.linalg.inv(x.T @ x) @ x.T @ y
print(theta)
```

$\hat{y} = X\theta$

$\therefore SSE = \sum_{i=1}^{n} (y_i - \hat{y})^2$ } both are same.

$= (y - \hat{y})^T (y - \hat{y})$

$= (y - X\theta)^T \cdot (y - X\theta)$

$= y^T y - y^T X\theta - \theta^T X^T y + \theta^T X^T X\theta$

$y^T X\theta = \theta^T X^T y$

$1\times5 \quad 5\times2 \quad 2\times1 = 1\times2 \quad 2\times5 \quad 5\times1$

$1\times1 \qquad\qquad 1\times1$

$= y^T y - 2\theta^T X^T y + \theta^T X^T X\theta$ → $2 \cdot X^T \cdot X(\theta)^2$

$\frac{\partial SSE}{\partial \theta} = \frac{y^T y - 2\theta^T X^T y + \theta^T X^T X\theta}{\partial \theta}$

$0 = 0 - 2 \cdot X^T y + 2 \cdot X^T X\theta$

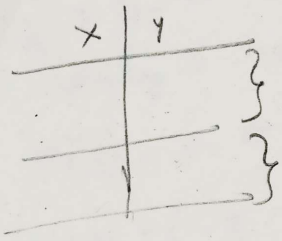
$2X^T \cdot X\theta = 2X^T y$

$X^T X\theta = X^T y$

$\theta = \frac{X^T y}{X^T X}$

$\boxed{\theta = X^T y (X^T X)^{-1}}$

Evaluation Metrics



Error
- $SSE = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$
- $MSE = \frac{SSE}{n}$
- $RMSE \Rightarrow \sqrt{mse}$

Evaluation
- Evaluation metrics
  - R^2
  - Adjusted R^2

$$R^2 = \frac{\sum_{i=1}^{n} (\hat{y}_i - \bar{y})^2}{\sum (y_i - \bar{y})^2} \rightarrow \frac{\text{variance explained by model}}{\text{variance in data}}$$

if $R^2 \rightarrow 1 \rightarrow$ Good model / perfect fit

$\rightarrow 0 \rightarrow$ bad model / not fitting.

variance: how data distributed around mean.

ex:- R^2 =

x	y	$\hat{y}$	*
	50	49	1
	55	56.5	1.5
	65	64	1
	70	71.5	1.5
	80	79	1

$\bar{y} = 64$

$(1 + (1.5)^2 + 1^2 + (1.5)^2 + 1)$

$= 1 + 2.25 + 1 + 2.25 + 1$

$= 7.5$

7.5

$(50-64)^2 + (55-64)^2 + (65-64)^2 + (70-64)^2 + (80-64)^2$

$196 + 81 + 1 + 36 + 256 = 570$

$(49-64)^2 + (56.5-64)^2 + (69-64)^2 + (71.5-64)^2 + (79-64)^2$

$225 + 56.25 + 25 + 56.25 + 225 = \frac{562.5}{570}$

$= 0.986$

It is a good model.

Adjusted R^2

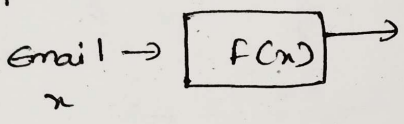
$$\text{Adjust}R^2 = 1 - \frac{(1-R^2)(n-1)}{n-k-1}$$

$n \rightarrow$ no. of data points $= 5$.
$k \rightarrow$ no. of features (inputs) $= 1$

$$= 1 - \frac{(1-(0.986)^2)(4)}{5-1-1} = 0.94$$

Classification:

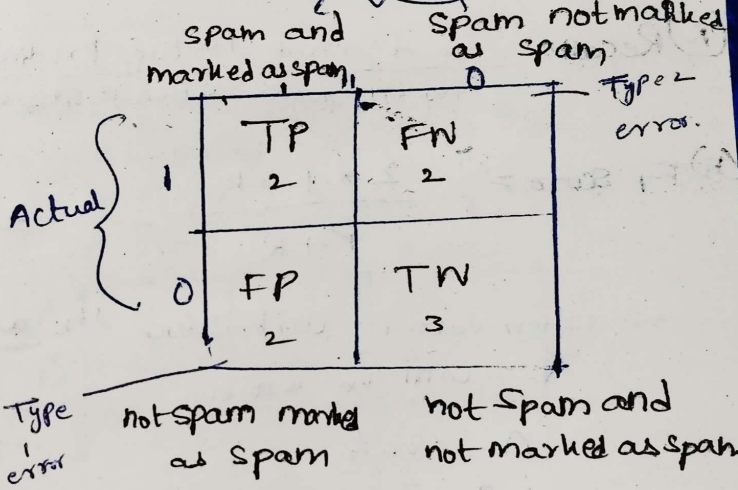
eg: 1



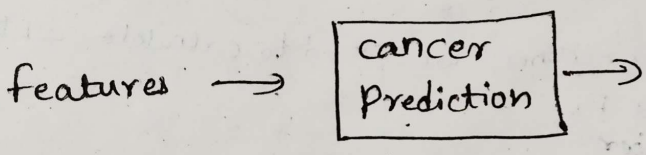
	x	y	$\hat{y}$
TN		0	0
TP		1	1
FN		1	0
FP		0	1
TN		0	0
TP		1	1
FP		0	1
FN		1	0
TN		0	0

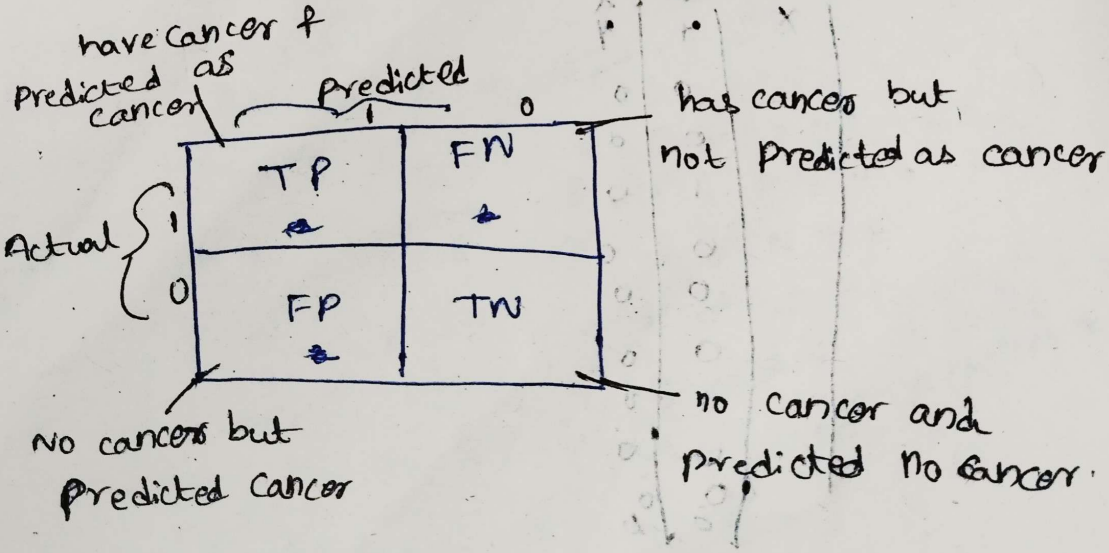
→

Confusion matrix



eg: 2





Evaluation Metrics:

- Accuracy
- Precision
- Recall
- F1 Score.

} Near to 1 → Good model
Near to 0 → Bad model (Not fitting model)

(A) Accuracy: $\frac{\text{Total correct prediction}}{\text{Total prediction}} = \frac{6}{10} = 0.6 = 60\%$

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

(P) Precision = $\frac{\text{No. of Correct Positive predictions}}{\text{no. of positive prediction}} = \frac{TP}{TP + FP} = \frac{3}{5} = 0.6 = 60\%$

(R) Recall: $\frac{\text{No. of correct Positive Prediction}}{\text{No. of actual Positive cases}} = \frac{3}{5} = \frac{TP}{TP + FN}$

$= 0.6 = 60\%$

(F) F1 Score = $\frac{2 * P * R}{P + R} = \frac{2 * 0.6 * 0.6}{0.6 + 0.6} = 0.6$

→ when data is unbalance the accuracy is high and info will be wrong.

TP 0	FN 2
FP 0	TN 8

A = 80%

P = 0

R = 0

F1 = 0

→ to get complete correctness we need to calculate all the A P R and F1.

X	cancer
4	4
0	0
0	0
1	0
0	0
0	0
0	0
1	0
0	0
0	0
0	0

Cyber

TP	FN
1	2
FP	TN
3	14

$$acc = \frac{1+14}{1+2+3+14} = \frac{15}{20} = 0.75$$

$$pre = \frac{1}{4} = 0.25$$

$$R = \frac{1}{3} = 0.3$$

$$F_1 = \frac{2 * 0.25 * 0.3}{0.25 + 0.3} = 0.27$$

Both A & F should be nearer values.

Multiclass classification

email

y	$\hat{y}$
0	0
0	0
0	2
0	0
0	2
1	1
1	1
1	1
1	0
1	1
2	1
2	1
2	2
2	2
2	2

0 — Regular Email
1 — Social Email
2 — spam Email

more than 2 classes, so multiclass.

zero	0	1	2
0	3	0	2
1	1	4	0
2	0	2	3

$$Accuracy = \frac{10}{15} = \frac{2}{3} = 0.66$$

class 0: TP = 3, TN = 9, FP = 1, FN = 2

$$P = 0.75 \quad R = 0.6 \quad F1 = \frac{2 \times 0.75 * 0.6}{0.75 + 0.6} = \frac{0.9}{6.75} = 0.13$$

class 1: TP = 4, TN = 8 (not 1, not pred 1), FP = 2, FN = 1

$$P = \frac{4}{6} = \frac{2}{3} = 0.6 \quad R = \frac{4}{5} = 0.8 \quad F1 = 2 * 0.6 * 0.8 = \frac{0.96}{0.6+0.8} = 0.68$$

TP = 3 TN = 8 FP = 2 (not 2 predicted as 2) FN = 2 (2 predicted not 2)

P = R = F =

Macro P = ——— = 0.65
3

Macro R = ——— = 0.66
3

Macro F_1 = ——— = 0.6
3

	0	1	2
0	2	1	2
1	1	3	1
2	3	0	2

y	$\hat{y}$
0	2
0	1
0	0
0	0
0	1
1	1
1	1
1	2
1	0
1	1
2	0
2	0
2	0
2	1
2	2

0 → Regular Gmail
1 → Social Gmail
2 → Spam Gmail

Accuracy = $\frac{7}{15}$

Class 0: TP = 2; TN: 6; FP: 4; FN: 3

$P = \frac{2}{6} = \frac{1}{3} = 0.3$ $R = \frac{2}{5}$ $F_1 =$

Class 1: TP: 3; TN = 9; FP: 1; FN: 2

$P = \frac{3}{4}$ $R = \frac{3}{5}$ $F_1 =$

Class 2: TP: 2; TN: 7; FP: 3; FN: 3

$P = \frac{2}{5}$ $R = \frac{2}{5}$ $F_1 =$

~~[redacted]~~

Classification Algorithms

① Decision Tree :-

$\{x, y\}$ $y \in$ {Spam, Not Spam}.

↓ discrete.

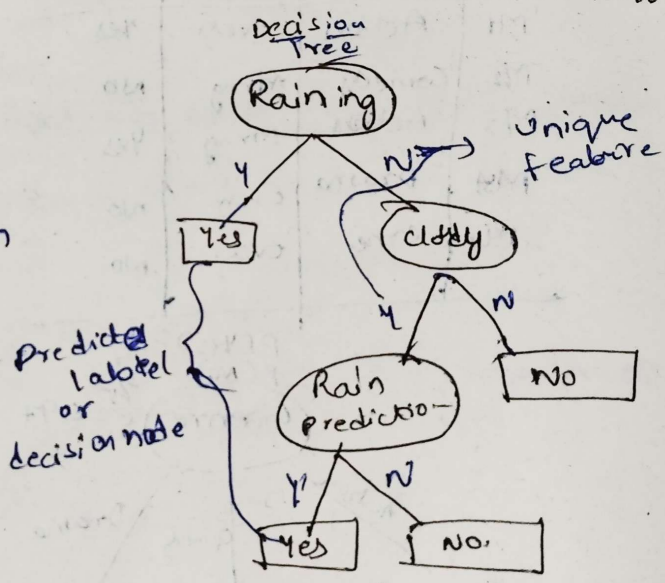
Ex. -

Cloudy, weather prediction, raining. → [will you take Umbrella] → Yes/No

x			y	$\hat{y}$
cloudy	weather prediction	raining	Umbrella ?	pred
N	N	N	N	N
Y	N	N	N	N
Y	N	Y	Y	Y
Y	Y	Y	Y	Y

Acc = 100%.

→ if raining
then Yes
elseif cloudy
if rain prediction
then Yes
else
No
else
No.



① Uncertainity and Entropy :-

→ How much certain an event is → measured by Entropy.

more Certainity ~~more~~ less Entropy.

Ex. will the students wear uniforms tomorr.

Today	C1	C2	C3	C4
	Y	Y	Y	Y
	Y	Y	Y	Y
more Certainity →	Y	Y	N	Y
	Y	N	N	N
				N

Entropy is less ← less Certainity Entropy more.

C1: $P(\text{uniform}) = \frac{4}{4} = 1$

$P(\text{Not uniform}) = \frac{0}{4} = 0$

C2: $P(U) = \frac{3}{4}$

$P(\text{Not U}) = \frac{1}{4}$

C3: $P(\text{U}) = 1/4$

$P(\text{Not U}) = 3/4$

C4: $P(U) = 2/4$

$P(\text{Not U}) = 2/4$

$$\text{Entropy} = -p_1 \log_2 p_1 - p_2 \log_2 p_2$$

$$\text{Entropy}(C_1) = -1 \log_2 1 - 0 \log_2 0$$

$$\text{Entro}(C_4) = -1/2 \log_2 1/2 - 1/2 \log 1/2 = -1 \log_2 1/2 = -(\log 1 - \log 2) = -(0 - 1) = 1$$

$$E(C_2) = -3/4 \log 3/4 - 1/4 \log 1/4 = 0.81$$

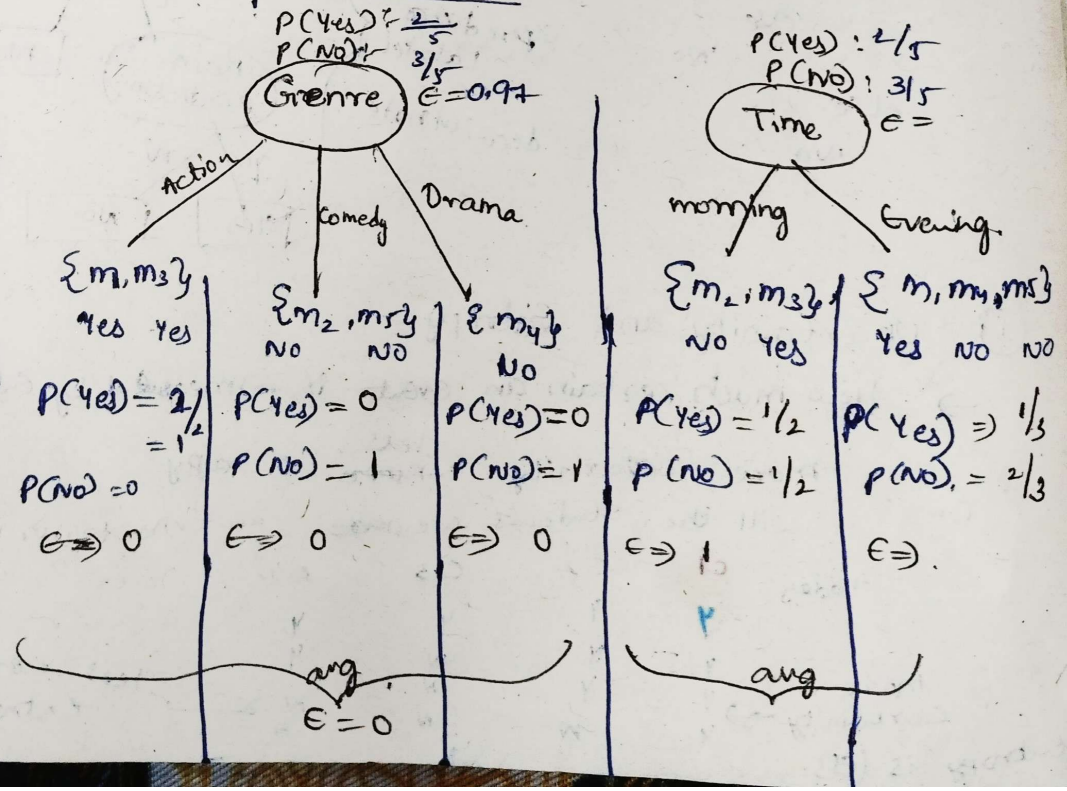
$$E(C_3) = -1/4 \log 1/4 - 3/4 \log 3/4 = 0.81$$

Uncertainity ↑ Entropy ↑

Certainity ↑ Entropy ↓

Eg: Watch Movie's ?

Movie	Genre	Time	Watch?
M1	Action	Even	Yes
M2	Comedy	Morng	NO
M3	Action	Morng	Yes
M4	Drama	Even	NO
M5	Come	Even	NO.



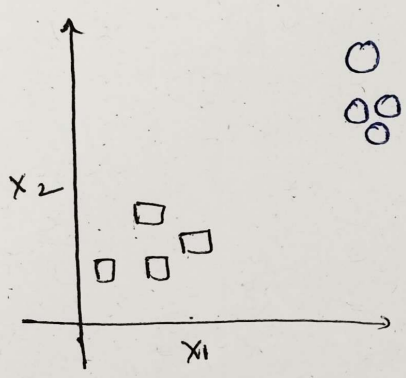
Introduction to classification

→ classification is the process of categorizing data or objects into predefined class or categories based on their features or attributes.

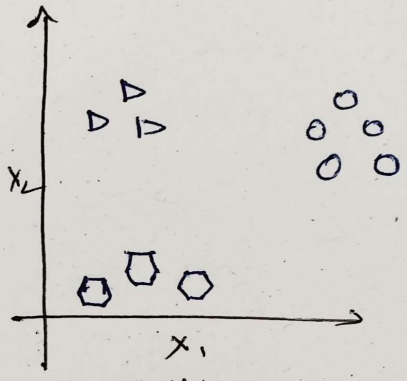
objective of classification (c)

→ main goal of c is to build a model that can accurately assign a label or category to a new observation based on its features.

ex: problem :- classify images as either "dogs" or "cats".



Binary



Multi

I. Binary classification

→ The task is to clasify input into one of two possible classes.

ex:- predict if a person has a disease based on health data

positive & Negative

II. Multiclass classification

→ The task is to classify the i/p into one of several possible classes.

shape

- circle, triangle, rectangle, sphere etc.