# Machine Learning.

## Machine Learning

↳ The word ML is first coined in 1950's when Artificial intelligence pioneer Arthur Samuel built the 1st self-learning system for playing checkers.

- ML is a branch of AI, that enables computers to "self-learn" from training data and improve overtime, w/o being explicitly programmed.

- ML algorithms are able to detect ~~data and~~ pattern in data and learn from them, in order to make their own predictions.

- In short, ML algorithms and models learn through experience.

- ML, on the other hand, is an automated process that enables machines to solve problems with little or no human i/p, and take actions based on past observations.

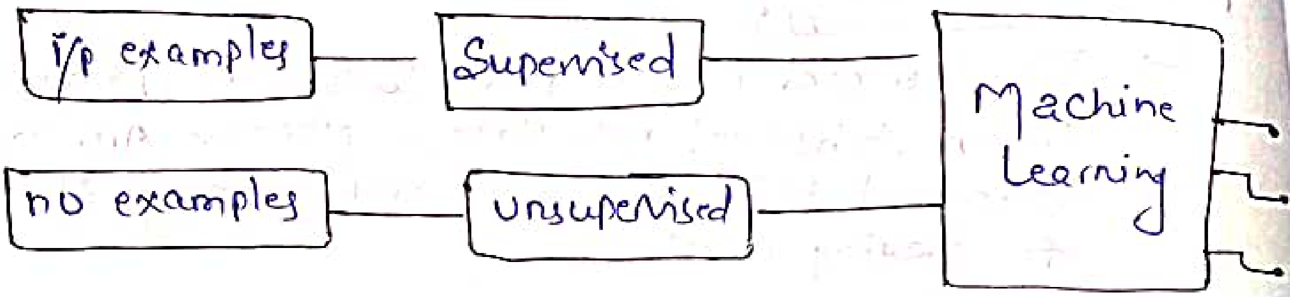
- While AI and ML often used interchangeably, they are two different concepts.

- AI is the broader concept, machines making decisions, learning new skills, solving problems in a similar way to humans

- Whereas ML is subset of AI that enables intelligent systems to autonomously learn new things from data.

- ML can be put to work on massive amount of data and can perform much more accurately than humans.

# Types of ML




(1) Supervised ML

- Supervised learning algorithm and supervised learning models make prediction based on labeled training data.
- Each training sample includes an i/p and a desired o/p.
- An educated guess when determining the labels for unseen data
- Most Common & popular approach to ML

⇒ Two types of supervised learning tasks: Classification and regression.

ⓐ Classification in Supervised ML

↳ Support vector machines (SVM)
↳ Naive Bayes etc

- o/p value is a category with a finite no. of options.

For ex: free pre-trained sentiment analysis model, you can automatically classify data as +ve, −ve, or neutral.

ⓑ Regression in Supervised ML

- In regression tasks, the expected result is a continuous number
- This model is used to predict quantity

e.x → probability an event will happen.

## (2) Unsupervised ML

- Unsupervised learning algorithms uncover insights and relationship in unlabeled data.
- In this case, models are fed i/p data but the desired outcomes are unknown.
- Models are not trained with the "right answer", so they must find patterns on their own.

e.g → Clustering

## Semi-Supervised learning

- training data is split into two
  - ↳ Small amount of labeled data
  - ↳ Large " " unlabeled "
- Provides more ~~accur~~ accurate results than regular supervised learning
- = Work on large datasets
- More cost-effective than supervised learning

## Reinforcement Learning (RL)

↳ Reinforced ML models attempt to determine the best possible path they should take ~~to~~ in a given situation.

↳ Do through trial & error

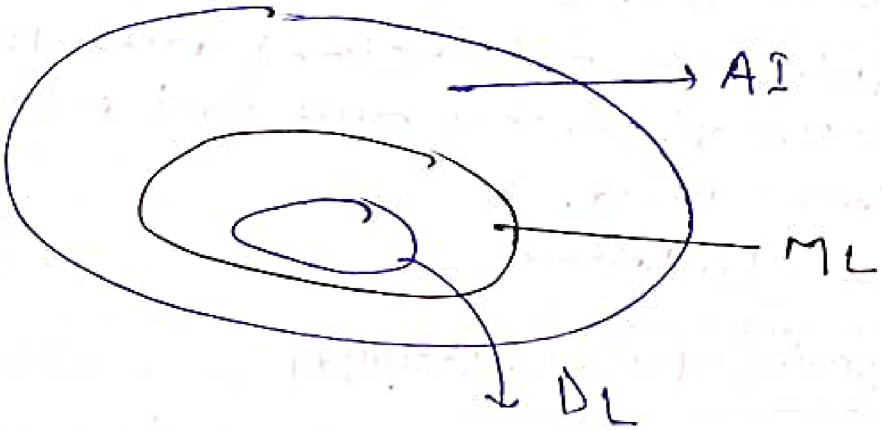
↳ Since there is no training data, machines learn from their own mistakes

↳ Used in robotics & gaming

## Deep Learning

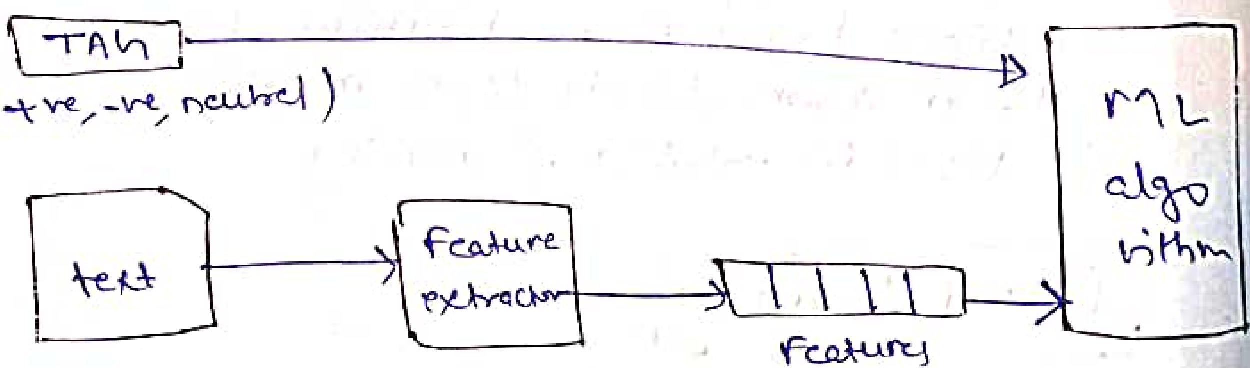
- DL models can be supervised, semi-supervised, or unsupervised, or combination of any or all of the three.
- Advanced ML algorithms

• Used by tech giants, like- Google, Microsoft and Amazon to run entire system and power things.
like — self driving cars, smart assistants.

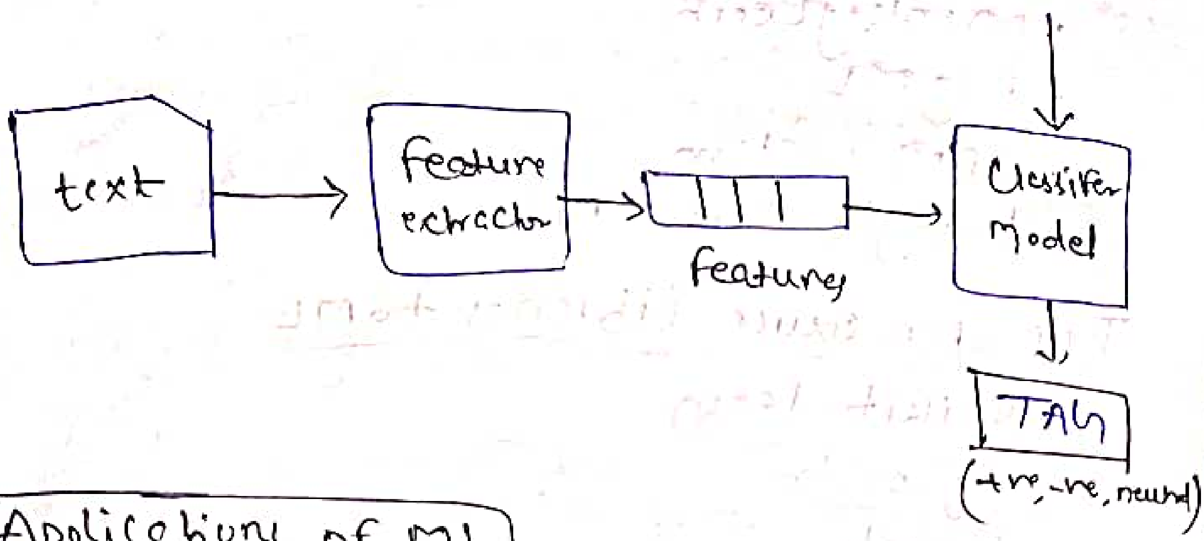



• DL based on Artificial Neural Network (ANN)
• A type of computer system that emulates the way the human brain works.
• It's kind of human brain that evolves with age and experience
• DL is common in image recognition, speech recognition and ~~neutral~~ NLP.
• DL models usually perform better than other ML algorithms.

## (a) Training



(b) Prediction -




# Applications of ML

- Social Media Monitoring
- Customer Service & Customer satisfaction
- Image Recognition
- Virtual Assistants
- Product Recommendation
- Stock Market Trading
- Medical Diagnosis

## Social Media Monitoring

Using ML you can monitor mentions of your brand on social media and immediately identify, if customers require urgent attention. By detecting mentions from angry customers, in real-time, you can automatically tag customers feedback and respond right away

NLP → Natural language Processing gives ~~the~~ machines the ability to break down spoken or written language much like human would to process "natural" language. So machine learning can handle text from practically any source.

## Top SaaS ML tools.

- MonkeyLearn
- BigML
- IBM Watson
- Google Cloud ML

## Top open source libraries to ML

- Scikit-learn
- PyTorch
- Kaggle
- NLTK
- Tensorflow

## # Instance-based learning

Sometimes called memory-based learning

- Instance-based learning is a family of learning algorithms that, instead of performing explicit generalization, compares new problem instances with instances seen in training, which have been stored in memory

e.g → K-nearest neighbor, decision tree

## # Model-based learnings:

- ML models that are parameterized with a certain no. of parameters that do not change as the size of training data changes.

- If you don't assume any distribution with a fixed no. of parameters over your data for ex: In K-~~neighbor~~ nearest neighbor, or in a decision tree, where the no. of parameters grows with the size of training data then you are not model-based or non parametric.

# Differen b/w Instance-based 4 Model-based learning:

| Model-based. | Instance-based. |
| --- | --- |
| • The goal is learn a generalizable mode. | • Don't try to learn generalizable model |
| • that can be used to prediction on new data. This means that the model is trained on a dataset and then tested on separate unseen dataset to evaluate its performance | • Memorise the training example |
| | • Their performance on new data is not reliable |
| • Model based learning can be more scalable | • B/C memorising the training examples they can be very slow and memory intensive. |
| • Don't have to store all of the training examples. | • Store all the training examples in memory |
| • Often produces model that are easier to interpret | • Store the examples and use them as basis of prediction. |
| • Required more efforts | • less effort required |

# Challenges in ML

- Lack of training data
- Poor quality of data
- Data Over fitting → overgeneralisation
- Data under fitting → model is too simple or misses parameters that is should have included in order to produce a clear and unbiased result.
- Irrelevant features

# Machine learning Life Cycle :

The ML life cycle is the cyclical process that data science project follow. It defines each step that an organisation should follow to take advantage of ML and AI to derive practical business value.

- There are (5) major steps in ML life cycle

Define Project objective → Acquire & Explore Data → Model Data → Interpret & Communicate → Implement document & maintain

(1) Define Project objective

- Specify business problem
- Acquire subject matter expertise
- Define unit of analysis and predict target
- Prioritize model criteria
- Consider risk and success criteria
- Decide weather to continue

(2) Acquire & explore Data

- Find appropriate data
- Merge data into single table
- Conduct exploratory data analysis
- Find and remove any target leakage
- Feature engineering

(3) Model Data

- variable selection
- Build candidate models
- Model validation & selection

(4) Interpret & Communicate

- Interpret model
- Communicate model insights

(5) Implement, document & maintain

- Set up batch of API prediction system
- Document modeling process for reproducibility
- Create model monitoring and maintenance plan

# Univariate Analysis

- Univariate analysis is a type of data visualization where we visualize only a single variable at a time.
- Univariate analysis helps us to analyse the distribution of variable present in the data so that we can perform further analysis.

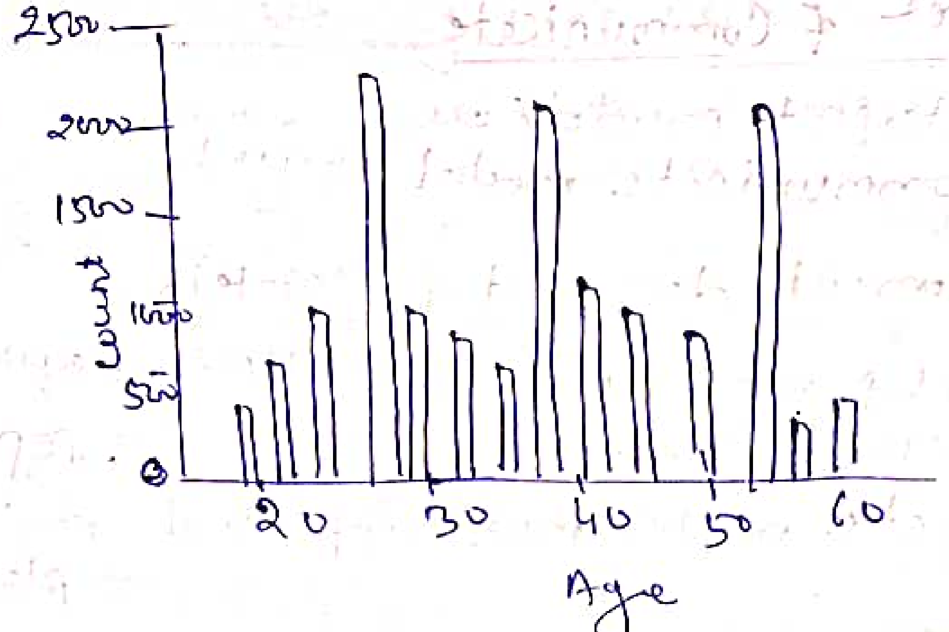
```
import pandas as pd
import seaborn as sns
data = pd.read_csv ('Employee_dataset.csv')
print (data.head())
```

- O/P :-

⇒ He we'll be performing univariate on numerical variables using the histogram function.

```
sns.histplot(data['age'])
```

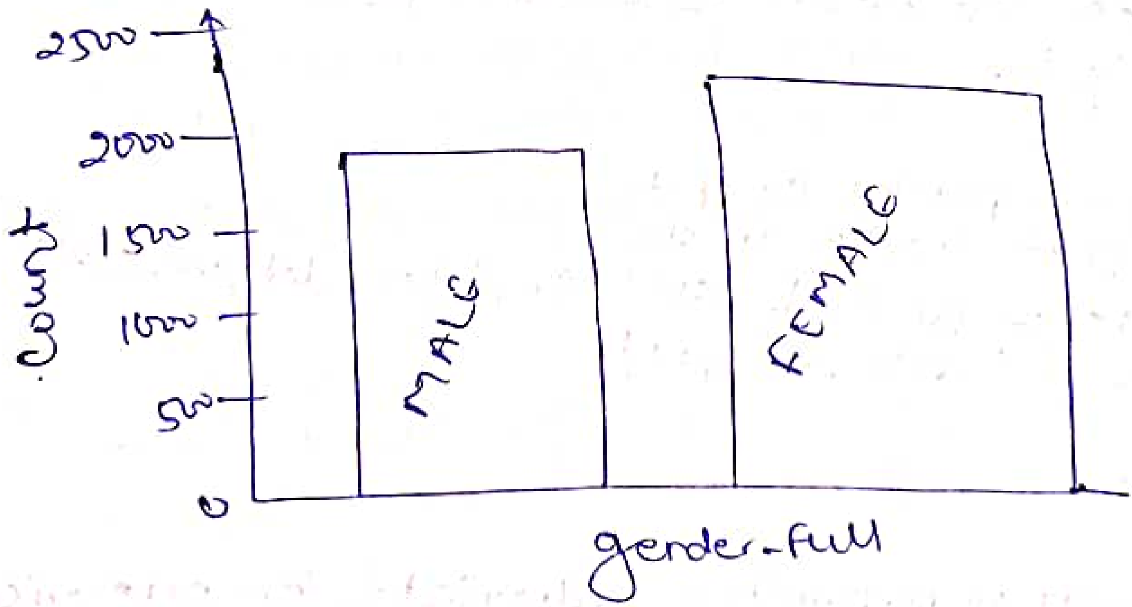
Output: <AxesSubplot: xlabel='age', ylabel='Count'>




=> Univariate analysis of categorical data.
We'll be using the count plot function from the seeborn library

sns.countplot(data['gender-full'])

__output:-__ < Axessubplot : xlabel='gender-full', ylabel='count'>



# Bivariate analysis

- Bivariate analysis is the simultaneous analysis of two variables.
- It explores the concept of the relationship b/w two variable
  - Weather there exists an association and the strength of this association or
  - weather there are differences b/w two variables and the significance of these differences

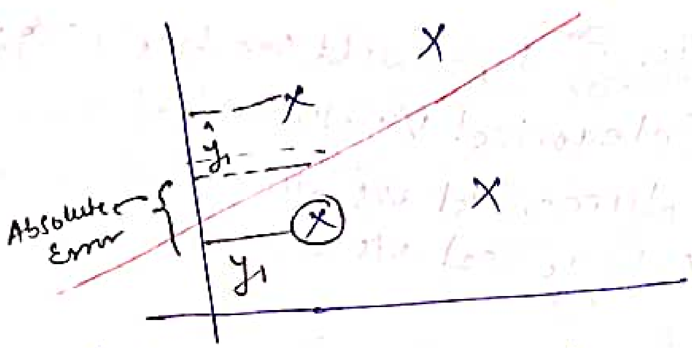
- The main (3) type will see here are:
  1. Categorical v/s Numerical
  2. Numerical v/s Numerical
  3. Categorical v/s categorical

# Regression Metrics.

(1) Mean Absolute Error (MAE)
(2) Mean Square Error (MSE)
(3) Root Mean Sq. Error (RMSE)
(4) $R_2$ Score
(5) Adjusted $R_2$ Score

## (1) Mean Absolute Error (MAE)

x | y
CGPA | package (lpa)



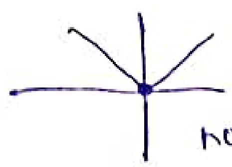

$$\frac{|y_1-\hat{y}_1| + |y_2-\hat{y}_2| + \cdots + |y_n-\hat{y}_n|}{n}$$

or $$\boxed{MAE = \frac{\sum_{i=1}^{n} |y_i-\hat{y}_i|}{n}}$$

**Advantage**

1. Same unit
2. Robust to outliers.

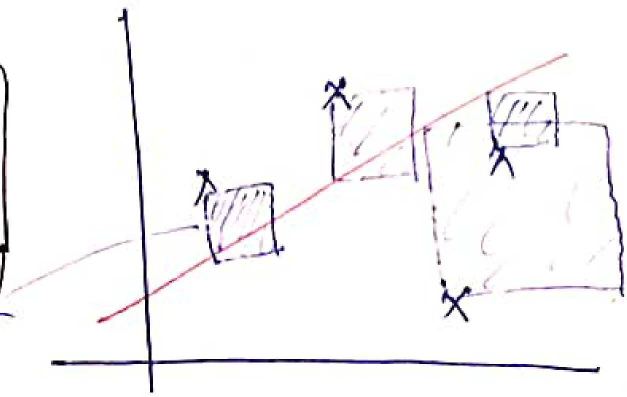
**Disadvantage**



not differentiable at '0' to get optimize

## (2) Mean Square Error (MSE)

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



$(y_1-\hat{y}_1)^2$

**Advantage**
↳ Can use as a loss function

**disadvantage**
- unit different
→ y – lpa
  mse – (lpa)²
- Not Robust to outliers

## (3.) Root Mean square Error (RMSE)

$$\boxed{Rmse = \sqrt{mse}}$$

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

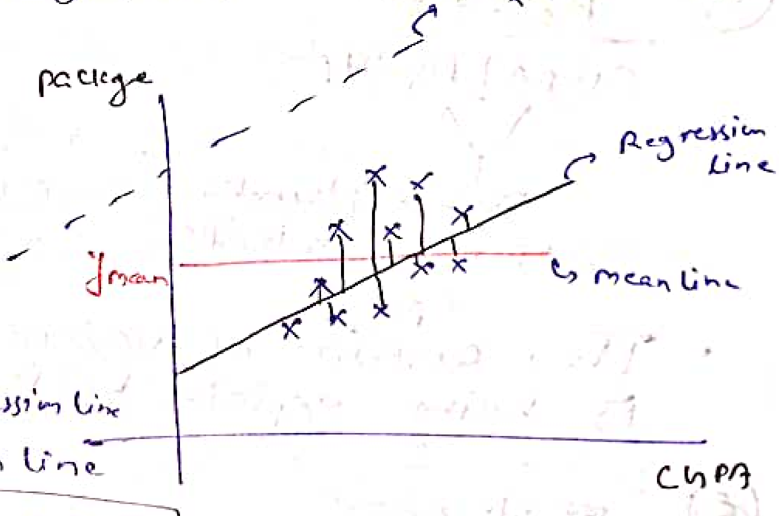
**Advantage**
- unit same
  y 4 ⊕L Func

**disadv**
† Not robust outliers

## (4.) R2 score

Dataset:
100 students → CGPA | package (lpa)

Regression line then R2 = -ve




$$\boxed{R2 = 1 - \frac{SSR}{SSM}}$$

SSR = sum of sq. Error in regression line
SSM = " " " " mean line

or

$$\boxed{R2 = 1 - \frac{\left[\sum_{i=1}^{n} (y_i - \hat{y}_i)^2\right]_{Reg}}{\left[\sum_{i=1}^{n} (y_i - \bar{y}_i)^2\right]_{mean}}}$$

Let say.

$R_2 = 0$ , i.e $\frac{\left[\sum_{i=1}^{n} (y_i - \hat{y}_i)^2\right]_{Reg}}{\left[\sum_{i=1}^{n} (y_i - \bar{y}_i)\right]_m} = 1$ , that mean Reg line & mean line is same
∴ CGPA does not contribute in it.

- $R_2 = 1$ $\quad \frac{\sum_{n=i}^{n} (y_i - \hat{y}_i)^2}{\sum_{n=i}^{n} (y_i - \bar{y}_i)} = 0 \longrightarrow = 0.$ that mean

that means Regression line is the perfect line passes through all data point, not do any mistakes

- ensure your $R_2$ tending to 1 rather zero.
- $R_2 = -ve.$ , SSR > SSm
  Regression line doing more mistakes as compare to mean line.
  ↳ data is highly non linear.

(ex)

$R_2 = 0.82$

CGPA | package means CGPA explain 80% of variance in lpa

(e.g) $R_2 = 0.80.$

CGPA | IQ | LPA

80% explaination of variance in lpa

20% can't explain by CGPA | IQ.

(R2)

- This amount of variance in the output column is being explain by the i/p column.

(5) (Disadvantage)

↳ no. of i/p column ↑es, explaination ↑es

(Disadvantage)

↳ no. of i/p column ↑es which is irrelevant, explaination ↑es which meaningless

(eg) → temp

## (5.) Adjusted R2 score

$$\boxed{R^2_{adj} = 1 - \left[\frac{(1-R_2)(n-1)}{(n-1-K)}\right]}$$

$R_2 \rightarrow$ R2 score

$n \rightarrow$ no. of rows (how many students data you have)

$K$ = independent column (i/p column)

suppose, only CGPA, $K=1$

CGPA, IQ, then $K=2$

CGPA, IQ, temp, $K=3$.

$$R^2_{adj} = 1 - \left[\frac{\overbrace{(1-R_2)}^{\text{Const or slightly}\uparrow\text{es}}\overbrace{(n-1)}^{\text{Const}}}{(n-1-\underset{\uparrow}{K})}\right] \rightarrow \text{over}\uparrow\text{es.}$$

Suppose, we add one irrelevent column i.e Temp, $K\uparrow$es, $(n-1-K)\downarrow$, $(n-1)$ const, $(1-R_2)$ const or less $\uparrow$es

$\hookrightarrow \downarrow$es $R^2_{adj}\downarrow$es.

$\therefore$ overall $\frac{(1-R_2)(n-1)}{(n-1-K)}$ $\uparrow$es

$\therefore$ $R^2_{adj}\downarrow$es, which shows $\downarrow$es in explaination

Suppose, we add one very relevant column, i.e IQ.

$K\uparrow$es, $(n-1-K)\downarrow$es, $(n-1)$ const, $R_2\downarrow\downarrow$es, $(1-R_2)\downarrow\downarrow$es

$\therefore$ overall $\frac{(1-R_2)(n-1)}{(n-1-K)}$ $\downarrow$es. $R^2_{adj}\uparrow$es $\rightarrow$ which is good.

- multiple linear regression $\longrightarrow$ $R^2_{adj}$ good.

# Linear Regression

↳ Supervised ML algorithm

1. Simple linear Regression
2. multiple " "
3. polynomial " "

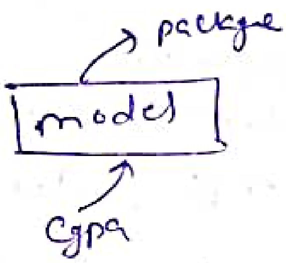
# Simple Linear Regression

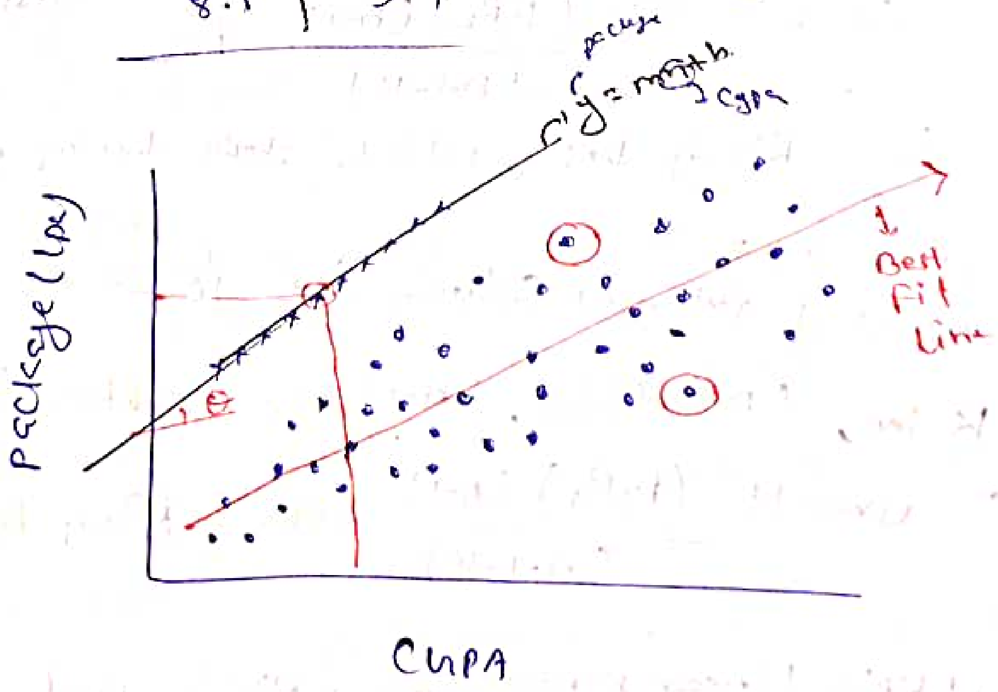
1 i/p Col , 1 o/p Col

| CGPA | package (lpa) |
|---|---|
| 6.66 | 3.01 |

Example

| Cgpa | Package (lpa) |
|---|---|
| 7.1 | 3.5 |
| 4.7 | 1.2 |
| 8.9 | 4.2 |
| 8.1 | 3.9 |




$\boxed{y = mx + b}$ → $\boxed{\text{package} = m \times \text{cgpa} + b}$

m = ? → weightage
b = ?

$\boxed{package = m \times exp + b}$ → offset

| exp | package |
|---|---|

exp = 0

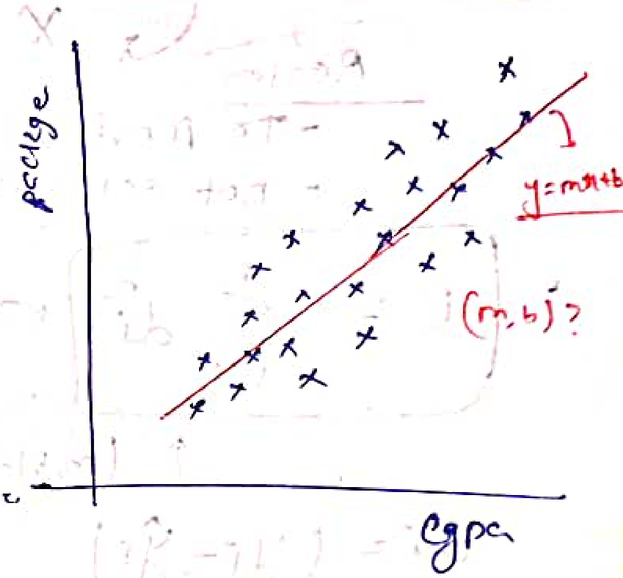
package = 0 X → package = b.

⇒ How to find m and b.

- Closed form solution →
- Non Closed

$ax^2 + bx + c = 0$

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$




① Closed form solⁿ

→ OLS → direct formula

② Non-Closed form solⁿ

→ (Gradient Descent)

$$\boxed{b = \bar{y} - m\bar{x}}$$

y → package
x → cgpa

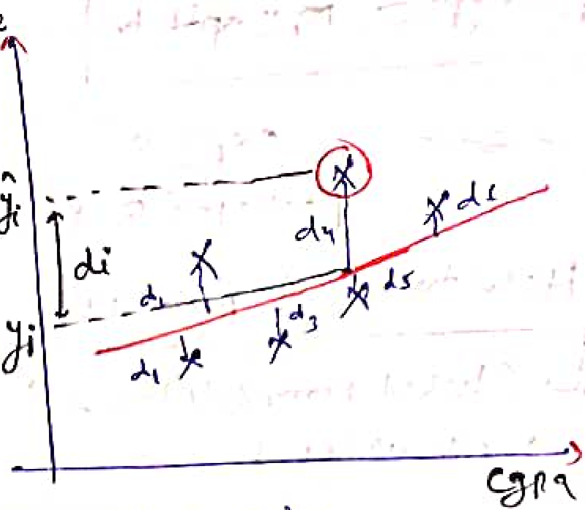
$$\boxed{m = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n} (x_i - \bar{x})^2}}$$

$x_i$ = current row

$\bar{x}$, $\bar{y}$ → mean value

$$E = d_1 + d_2 + d_3 + \dots + d_n$$

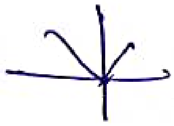
$$E = d_1^2 + d_2^2 + d_3^2 + \dots + d_n^2$$




$$E = |d_1| + |d_2| + \dots |d_n|$$ X

<u>Reason</u>

- To penalise the far distance point
- not easily differentiable



$$\therefore \boxed{E = \sum_{i=1}^{n} d_i^2}$$ ← Error function

↑ $(m, b)$

$$d_i = (y_i - \hat{y}_i) \quad , \therefore \boxed{E = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2}$$

↳ Total error

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

↳ avg. error

$$\boxed{\hat{y}_i = m x_i + b}$$

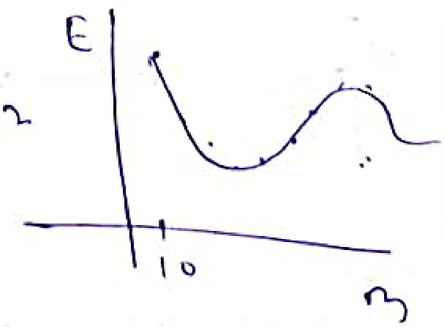
$$\therefore \boxed{E(m,b) = \sum_{i=1}^{n} (y_i - m x_i - b)^2} \rightarrow \text{minimum}$$

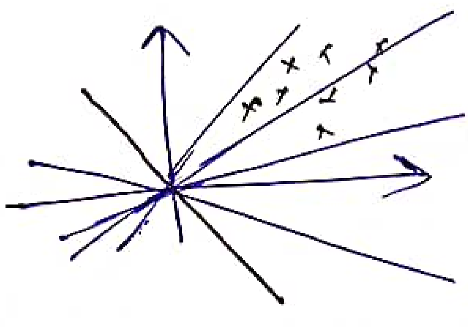
error function

$y = f(x)$

suppose $\boxed{b = 0}$

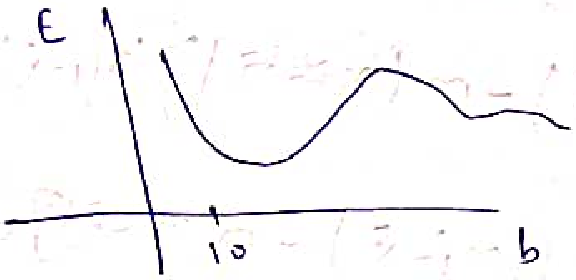
$$E(m) = \sum_{i=1}^{n} (y_i - m x_i)^2$$

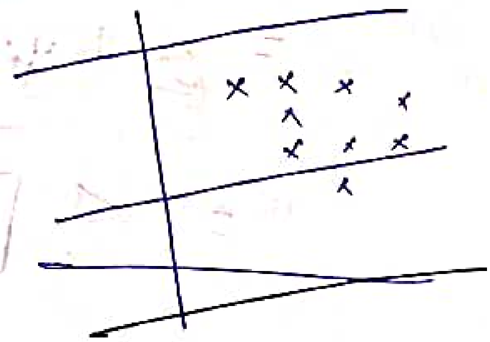


suppose $m = 1$

$$\therefore \quad E(b) = \sum_{i=1}^{n} (y_i - x_i - b)^2$$



Now consider both, $m, b$.

$$\frac{\partial E}{\partial b} = \frac{\partial}{\partial b} \sum_{i=1}^{n} (y_i - m x_i - b)^2 = 0$$

$$= \sum \frac{\partial}{\partial b} (y_i - m x_i - b)^2 = 0$$

$$= \sum 2 (y_i - m x_i - b) \times (-1) = 0$$

$$= \sum (y_i - m x_i - b) = 0$$

$$\frac{\sum y_i}{n} - \frac{\sum m x_i}{n} - \frac{\sum b}{n} = \frac{0}{n}$$

$\rightarrow \frac{b + b + b + \cdots n \text{ times}}{n} = \frac{nb}{n}$

$$\bar{y} - m\bar{x} - \frac{nb}{n} = 0$$

$$\bar{y} - m\bar{x} = b$$

$$\boxed{b = \bar{y} - m\bar{x}} \quad \text{---} (1)$$

$$E = \sum (y_i - m x_i + b)$$

$$\text{or } E = \sum (y_i - m x_i - \bar{y} + m\bar{x})^2 \quad \text{--- (from (1))}$$

$$\frac{\partial E}{\partial m} = \sum \frac{\partial}{\partial m} (y_i - m x_i - \bar{y} + m\bar{x})^2$$

$$= \sum 2(y_i - mx_i - \bar{y} + m\bar{x})(-x_i + \bar{x}) = 0$$

$$= \sum (y_i - mx_i - \bar{y} + m\bar{x})(x_i - \bar{x}) = 0$$

$$= \sum [(y_i - \bar{y}) - m(x_i - \bar{x})](x_i - \bar{x}) = 0$$

$$= \sum [(y_i - \bar{y})(x_i - \bar{x}) - m(x_i - \bar{x})^2] = 0$$

$$= \sum (y_i - \bar{y})(x_i - \bar{x}) = m \sum (x_i - \bar{x})^2$$

$$\boxed{m = \frac{\sum_{i=1} (x_i - \bar{x})(y_i - \bar{y})}{\sum (x_i - \bar{x})^2}}$$

# Multi Linear Regression

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

→ cgpa | gender | iq | lpa

$y = mx + b$

$\boxed{y = \beta_0 + \beta_1 x_1 + \beta_2 x_2}$ ← $y = mx_1 + nx_2 + b$

3D.

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n$$

or

$$\boxed{y = \beta_0 + \sum_{i=1}^{n} \beta_i x_i}$$ → $n = 1$

$y = \beta_0 + \beta_1 x_1$

↳ 3D. $x_1 = cgpa$, $x_2 = iq$, $y = lpa$

$$\boxed{lpa = \beta_0 + \beta_1 \times cgpa + \beta_2 \times iq}$$

## Mathematical Formulation

cgpa | iq | gender | lpa

4D $x_1$ $x_2$ $x_3$ $y$ → actual

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

pre

2D.

$\cancel{y = mx + b}$

$y = \beta_0 + \beta_1 x$

100 students.

(100, 4)

$$\hat{Y} = \begin{bmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \hat{y}_3 \\ \vdots \\ \hat{y}_{100} \end{bmatrix} = \begin{bmatrix} \beta_0 & \beta_1 x_{11} & \beta_2 x_{12} & \beta_3 x_{13} \\ \beta_0 & \beta_1 x_{21} & \beta_2 x_{22} & \beta_3 x_{23} \\ \vdots \\ \beta_0 & \beta_1 x_{1001} & \beta_2 x_{1002} & \beta_3 x_{1003} \end{bmatrix}$$

Suppose 100 rows ⇒ $\boxed{n \text{ rows}}$, 3 cols → $\boxed{m \text{ colms}}$

$$\therefore \hat{Y} = \begin{bmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \vdots \\ \hat{y}_n \end{bmatrix} = \begin{bmatrix} \beta_0 & \beta_1 x_{11} & \beta_2 x_{12} & \beta_3 x_{13} & \cdots & \beta_m x_{1m} \\ \beta_0 & \beta_1 x_{21} & \beta_2 x_{22} & \beta_3 x_{23} & \cdots & \beta_m x_{2m} \\ \vdots \\ \beta_0 & \beta_1 x_{n1} & \beta_2 x_{n2} & \beta_3 x_{n3} & \cdots & \beta_m x_{nm} \end{bmatrix}$$

or

$$= \begin{bmatrix} 1 & x_{11} & x_{12} & x_{13} & \cdots & x_{1m} \\ 1 & x_{21} & x_{22} & x_{23} & \cdots & x_{2m} \\ \vdots \\ 1 & x_{n1} & x_{n2} & x_{n3} & \cdots & x_{nm} \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_m \end{bmatrix}$$

or

$$\boxed{\hat{Y} = X\beta} \longrightarrow (1)$$

$X$ = x_train

$Y$ = y_train

$$\boxed{\beta = (X^T X)^{-1} X^T Y}$$

# Gradient Descent

| gpe | package |
|---|---|
| — | — |
| — | — |
| — | — |
| — | — |

OLS, $\hat{y}_i = m x_i + b$

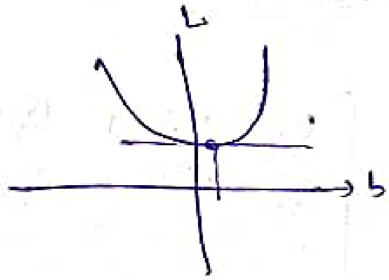
$$L = \sum_{i=1}^{4} (y_i - \hat{y}_i)^2$$

$$L = \sum_{i=1}^{4} (y_i - m x_i - b)^2$$

$$\underset{\min}{\tilde{L}} = \sum_{i=1}^{4} (y_i - 78.35 x_i - b)^2$$

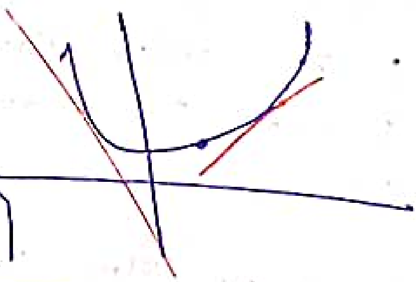
$m = 78.35$

$L \to b^2$




- Step-1: select a random $b_{min}$

$$b = -10$$



$\boxed{b_{new} = b_{old} - slope}$

$\boxed{b_{new} = b_{old} - \eta\, slope}$ $\eta = 0.01$ ↳ generally

- When stop
  1. diff $b_{old}$, $b_{new}$
     $b_{new} - b_{old} = 0.0001$

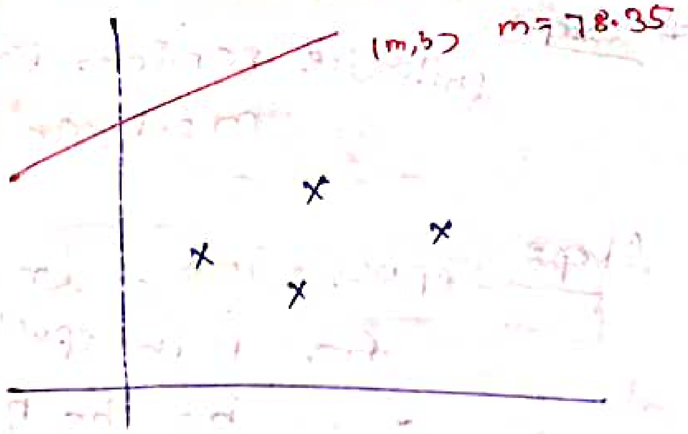
Example.

Step-1.
Start with a random $b = b$

for i in epochs:

$$\boxed{\eta = 0.01}$$

$$\boxed{b_{new} = b_{old} - \eta \times slope}$$




$$L = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$ let $b = 0$.

$$\therefore \frac{dL}{db} = \frac{d}{db}\left( \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \right) = 2\sum_{i=1}^{n} ($$

$$\frac{d}{db}\sum_{i=1}^{n} (y_i - mx_i - b)^2 = 2\sum_{i=1}^{n} (y_i - mx_i - b)(-1)$$

$$= -2\sum_{i=1}^{n} (y_i - mx_i - b)$$

$$= -2\sum_{i=1}^{n} (y_i - 78.35 * x_i - 0)$$

slope ($b=0$) $i = 1$

$$b_{new} = b_{old} - \eta\, slope_{b=old}$$

$b_{new}$

# Step-1:

Initialise random values for m & b

m = 1 and b = 0

Step 2 epocs = 100 , lr = 0.01

for i in epochs

b = b - $\eta$ slope

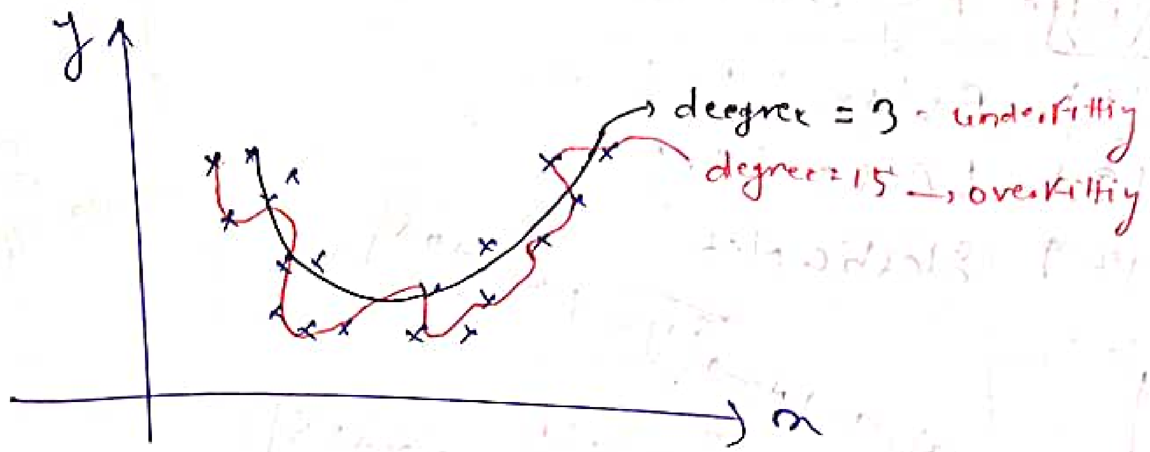
m = m - $\eta$ slope

# Polynomial Regression

$\boxed{y = \beta_0 + \beta_1 x}$ → simple Linear regression

$$\boxed{y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \cdots + \beta_n x_n}$$

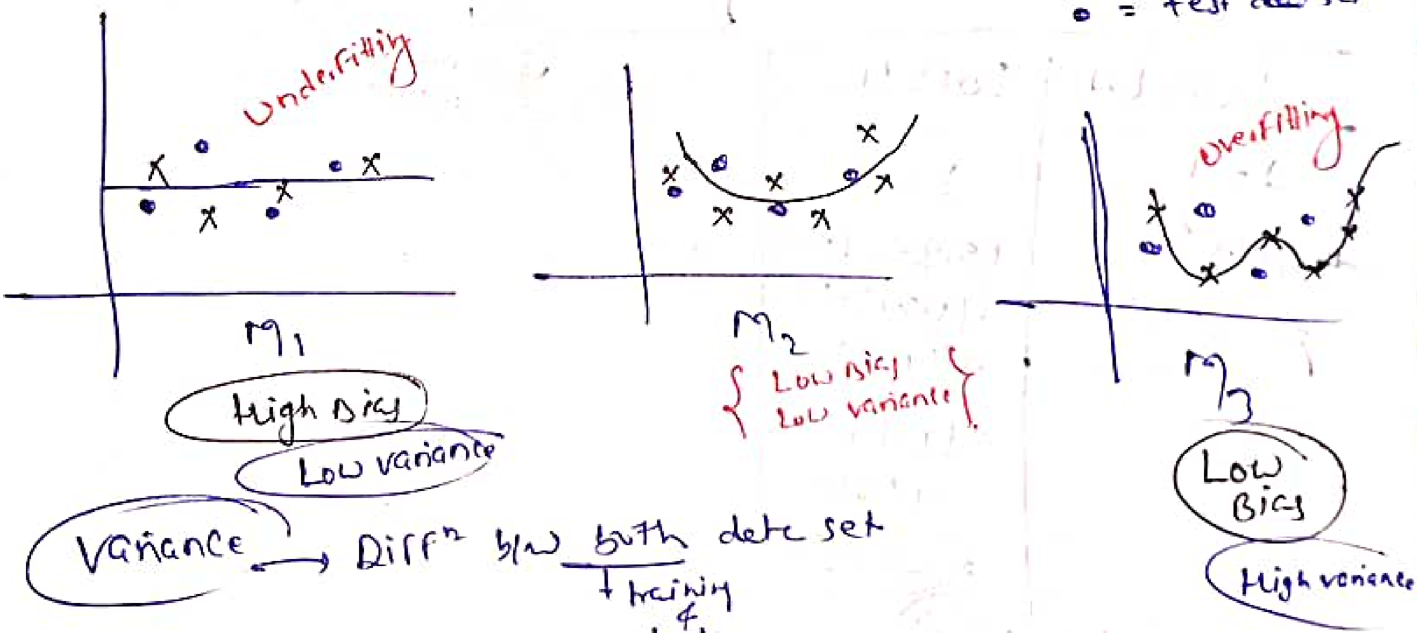
↳ multiple Linear Regression




# Bias Variance Trade Off

↳ Bias ↳ inability of a ML model to duplicate/capture in the training data

x = training set
• = test set




Variance → Diff^n b/w both data set (training & test)

• Regularization

↳ ~~to add in ML model to overcome the~~

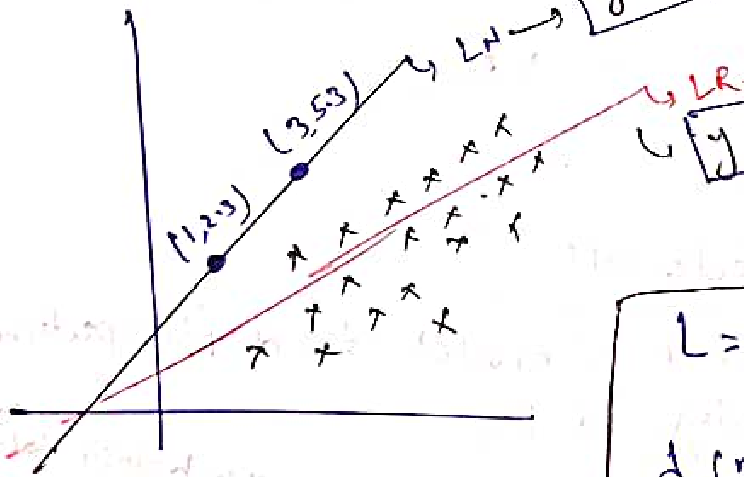
induce some added information in ML model so that overcome ~~the~~ or reduce the overfitting

# Ridge Regularization

(1) Ridge ($L_2$)
(2) LASSO ($L_1$)
(3) Elastic Net

LN → $y = 1.5x + 0.8$

LR → $y = 0.9x + 1.5$




$$L = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \lambda (m^2)$$

become zero for LN b/c lines pass from all training data set no mistake no error

$\lambda (m^2)$ ↳ $L^2$ norm

$\hat{y}_i = 0.9x + 1.5$

| Loss LN | Loss LR |
|---|---|
| $\lambda = 1$<br>$0 + (1.5)^2$<br>$= 2.25$ | $\lambda = 1$<br>$(2.3 - 0.9 - 1.5)^2 + (5.3 - 2.7 - 1.5)^2 + (0.9)^2$<br>$= (0.1)^2 + (1.1)^2 + (0.9)^2$<br>$= 2.03$ |

LR < LN

Model automatically choose LR line

$$L = \sum_{i=1}^{n} (y_i - m x_i - \bar{y} - m\bar{x})^2 + \lambda m^2$$

$\bar{y} \rightarrow$ y-mean
$\bar{x} \rightarrow$ x-mean

$$\frac{\partial L}{\partial m} = 0$$

$$\boxed{m = \frac{\sum_{i=1}^{n} (y_i - \bar{y})(x_i - \bar{x})}{\sum_{i=1}^{n} (x_i - \bar{x})^2 + \lambda}}$$

$\rightarrow$ Rigde Regression

$\lambda \rightarrow$ hyperparameter

equl $\lambda = 0$

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

$\rightarrow$ simple Regression

$\Rightarrow$ ~~Rig~~ Ridge Regression For nD data

$$\boxed{W = (X^T X + \lambda I)^{-1} X^T Y}$$

$$W = (X^T X)^{-1} X^T Y$$

## Ridge

$\lambda(w_1^2+w_2^2+\dots+w_n^2)$

$$\boxed{L = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \lambda ||w||^2}$$

↳ overfitting reduce

## Lasso

$$\boxed{L = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \lambda ||w||}$$

## Elastic Net Regression → Ridge + Lasso

$$\boxed{L = \sum (y_i - \hat{y}_i)^2 + a||w||^2 + b||w||}$$

By default
$\lambda = 1$
$l_1$ ratio $= 0.5$
$\therefore a = 0.5,\ b = 0.5$

$$\left\{ \begin{aligned} \lambda &= a+b \\ l_1 \text{ ratio} &= \frac{a}{a+b} \end{aligned} \right\}$$

Ex

Q. Find a quadratic regression model for the following data.

| x | y |
|---|---|
| 3 | 2.5 |
| 4 | 3.2 |
| 5 | 3.8 |
| 6 | 6.5 |
| 7 | 11.5 |

Solution

Let the quadratic polynomial regression model be

$$y = a + a_1 x + a_2 x^2$$

The value of $a, a_1, a_2$ are calculated using

$$\sum y_i = n a_0 + a_1 (\sum x_i) + a_2 (\sum (x_i^2)) \quad (1)$$

$$\sum y_i x_i = a_0 (\sum x_i) + a_1 (\sum x_i^2) + a_2 (\sum x_i^3) \quad (2)$$

$$\sum y_i x_i^2 = a_0 (\sum x_i^2) + a_1 (\sum x_i^3) + a_2 (\sum x_i^4) \quad (3)$$

| | $x$ | $y$ | $x^2$ | $x^3$ | $x^4$ | $y \times x$ | $y \times x^2$ |
|---|---|---|---|---|---|---|---|
| | 3 | 2.5 | 9 | 27 | 81 | 7.5 | 22.5 |
| | 4 | 3.2 | 16 | 64 | 256 | 12.8 | 51.2 |
| | 5 | 3.8 | 25 | 125 | 625 | 19.0 | 95 |
| | 6 | 6.5 | 36 | 216 | 1296 | 39 | 234 |
| | 7 | 11.5 | 49 | 343 | 2401 | 80.5 | 563.5 |
| $\Sigma$ | 25 | 27.5 | 135 | 775 | 4659 | 158.8 | 966.2 |

$$27.5 = 5a_0 + 25a_1 + 135a_2$$

$$158.8 = 25a_0 + 135a_1 + 775a_2$$

$$966.2 = 135a_0 + 775a_1 + 4659a_2$$

$a_0 = 12.42857$

$a_1 = -5.51286$

$a_2 = 0.76429$

$\therefore$ $\boxed{y = 12.42857 - 5.51286x + 0.76429x^2}$ Ans

Step-1 → Dataset

no. of features (n) = 2

no. of samples (N) = 4

Step 2 → Computation of mean of variables

$\bar{x}$ - $\bar{y}$

4, 8    8.5

Step 3 Computation of covariance matrix

$(x,x)$ $(x,y)$, $(y,x)$, $(y,y)$

(1) $Co(x,x) = \frac{1}{N-1} \sum_{K=1}^{N} (x_{iK}-\bar{x}_i)^2 (x_{K}$

$$= \boxed{\frac{1}{N-1} \sum_{i=1}^{N} (x_i-\bar{x})(y_i-\bar{y})}$$

Step. Covariance matrix $(n \times n)$

$$S = \begin{bmatrix} Cov(x,x) & Cov(x,y) \\ Cov(y,x) & Cov(y,y) \end{bmatrix}$$

Step-4 Eigen value Eigen vector, normalize eigen vector

(1) Eigen value

$\det(S - \lambda I) = 0$

$I =$

$\lambda_1, \lambda_2$

$\boxed{\lambda_1 > \lambda_2}$

(ii) Eigen vector of $\lambda_1$

$$(S - \lambda_1 I)\, v_1 = 0 \qquad v_1 = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}.$$

$(u_1, u_2)$

(iv)-

(iii) Normalisation of eigen vector $v_1$

$e_1 =$

$\lambda_2 \to e_2 = ?$

Step-5 Derive new dataset

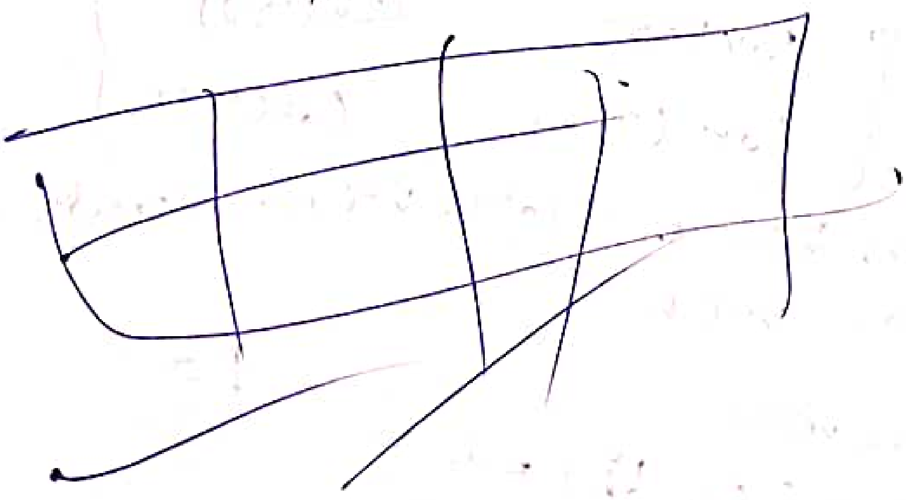
$$P_{11} = e_1^T \begin{bmatrix} 4-8 \\ 11-8.5 \end{bmatrix}$$

$$= e_1^T \begin{bmatrix} x_i - \bar{x} \\ y_i - \bar{y} \end{bmatrix}$$

$P_{12}$

$P_{13}$

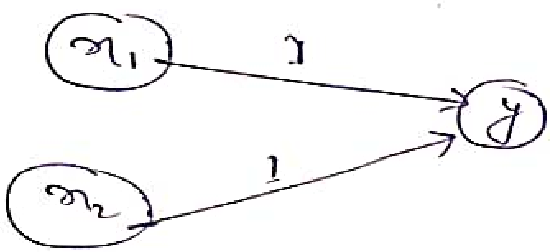
$P_{14}$



Ex-3.1. Mc culloch-pitts.

AND function.

| $x_1$ | $x_2$ | $y$ |
|---|---|---|
| 1 | 1 | 1 |
| 1 | 0 | 0 |
| 0 | 1 | 0 |
| 0 | 0 | 0 |




output Y is

$$Y = f(y_{in})$$

The net input

$$\boxed{y_{in} = \sum_i \text{weights} * \text{input}}$$

$w_1 = w_2 = 1$

$$y_{in} = 1 \times x_1 + 1 \times x_2$$

$$y_{in} = x_1 + x_2$$

from this the activation of output neuron can be performed

$$Y = f(y_{in}) = \begin{cases} 1 & \text{if } y_{in} \geq 2 \\ 0 & \text{if } y_{in} < 2 \end{cases}$$

Now. present the input

(i) $x_1 = x_2 = 1$, $y_{in} = x_1 + x_2 = 1 + 1 = 2$

$y = f(y_{in}) = 1$ since $y_{in} = 2$

(ii) $x_1 = 1,\ x_2 = 0$, $y_{in} = x_1 + x_2 = 0+0 = 1$

$y = f(y_{in}) = 0$ since $y_{in} = 1 < 2$

This is same when $x_1 = 0$ and $x_2 = 1$

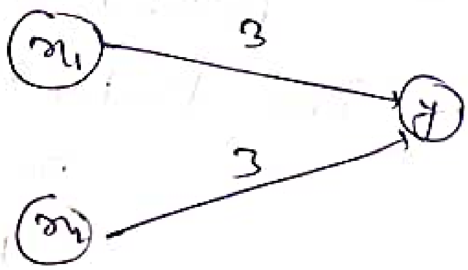
(iii) $x_1 = 0,\ x_2 = 0,\ y_{in} = x_1 + x_2 = 0+0 = 0$

Hence $y = f(y_{in}) = 0$ since $y_{in} = 0 < 2$

Ex-3.2. Generate OR function using McCulloch Pitts neuron model.

Sol$^n$ The truth table for OR function

| $x_1$ | $x_2$ | $y$ |
|---|---|---|
| 1 | 1 | 1 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 0 | 0 | 0 |




The threshold for the unit is 3.

The net input is calculated as

$$y_{in} = 3x_1 + 3x_2$$

output given by

$$y = f(y_{in}) = \begin{cases} 1 & \text{if } y_{in} \geq 3 \\ 0 & \text{if } y_{in} < 3 \end{cases}$$

→ Presenting the inputs

(i) $x_1 = x_2 = 1$, $y_{in} = 3x_1 + 3x_2 = 1+1 = 2$

$= 3\times1 + 3\times1 = 6 >$ threshold $3$

Hence $y = 1$

(ii) $x_1 = 1, x_2 = 0$

$y_{in} = 3x_1 + 3x_2$

$= 3 \times 1 + 3 \times 0 = 3 =$ threshold

Applying activation formulae

$y = f(y_{in}) = 1$

This is also the case when $x_1 = 0$ & $x_2 = 1$

(iii) $x_1 = x_2 = 0$

$y_{in} = 3x_1 + 3x_2 = 3 \times 0 + 3 \times 0 = 0 <$ threshold

Hence output $y = 0$

Ex-3.3. Realise NOT function using McCulloch Pitts neuron model.

The truth table for NOT function

| $x$ | $y$ |
|---|---|
| 1 | 0 |
| 0 | 1 |

$X \xrightarrow{1} Y$

Threshold for unit $y$ is 1

net input

$y_{in} = x.w$

$w = 1, y_{in} = x$

Output activation

$$y = f(y_{in}) = \begin{cases} 1 & \text{if } y_{in} < 1 \\ 0 & \text{if } y \geq 1 \end{cases}$$

Presenting the input

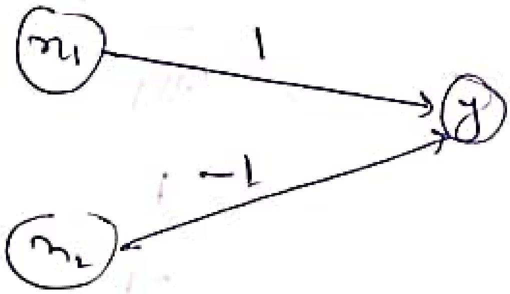
(i) $x_1 = 1$, $y_{in} = 1$, Applying activation $y = f(y_{in}) = 0$

(ii) $x_1 = 0$, $y_{in} = 0$, " " $y = f(y_{in}) = 1$

Ex-3.4. Generate the output of ANDNOT function using McCulloch-Pitts neuron.

Truth table of AND-NOT

| $x_1$ | $x_2$ | $y$ |
|---|---|---|
| 1 | 1 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 0 |
| 0 | 0 | 0 |




Threshold 1.

net input $y_{in} = x_1 w_1 + x_2 w_2$

$= x_1 \times 1 + x_2 \times (-1)$

$y_{in} = x_1 - x_2$

output activation

$$y = f(y_{in}) = \begin{cases} 1 & \text{if } y_{in} \geq 1 \\ 0 & \text{if } y_{in} < 1 \end{cases}$$

Presenting the input

(i) $x_1 = x_2 = 1$ $\quad y_{in} = x_1 - x_2 = 1 - 1 = 0 < 1$

Hence $y = f(y_{in}) = 0$

(ii) $x_1 = 1, x_2 = 0$, $\quad y_{in} = x_1 - x_2 = 1 - 0 = 1 \geq 1$

$y = f(y_{in}) = 1$

(iii) $x_1 = 0, x_2 = 1$ $\quad y_{in} = x_1 - x_2 = 0 - 1 = -1 < 1$

$y = f(y_{in}) = 0$

(iv) $x_1 = x_2 = 0$, $\quad y_{in} = x_1 - x_2 = 0 = 0 < 1$

$y = f(y_{in}) = 0$

Thus AND NOT function is realized.

# PERCEPTRON

**Ex-4.1.** Develop a perceptron for the AND function with bipolar inputs and targets

**Soln**

| input | | | Target |
|---|---|---|---|
| $x_1$ | $x_2$ | $b$ | $t$ |
| 1 | 1 | 1 | 1 |
| -1 | 1 | 1 | -1 |
| 1 | -1 | 1 | -1 |
| -1 | -1 | 1 | -1 |

Step1: Initial weights $w_1 = w_2 = 0$ and $b = 0$, $\alpha = 1$, $\theta = 0$

Step2: Begin computation

Step3: For input $(1,1)$ : 1, do Steps 4-6

Step4: Set activations of inputs

$x_i = (1,1)$

Step5: Calculate the net input

$$\boxed{y_{in} = b + \sum x_i w_i} = 0 + 1\times 0 + 1 \times 0 = 0$$

Applying the activation,

$$y = f(y_{in}) = \begin{cases} 1 & \text{if } y_{in} > 0 \\ 0 & \text{if } y_{in} = 0 \\ -1 & \text{if } y_{in} < -0 \end{cases}$$

Therefore $y = 0$

$$\boxed{\alpha = 1}$$

Step 6: $t = 1$ and $y = 0$

Since $t \neq y$, the new weights are

$$w_i(new) = w_i(old) + \alpha t x_i$$

$$w_1(new) = w_1(old) + \alpha t x_1$$

$$= 0 + 1 \times 1 \times 1 = 1$$

$$\boxed{\alpha = 1}$$

$$w_2(new) = w_2(o) + \alpha t x_2$$

$$= 0 + 1 \times 1 \times 1 = 1.$$

$$b(new) = b(old) + \alpha t$$

$$b(n) = b(o) + \alpha t = 0 + 1 \times 1 = 1$$

The new weights and bias are [1 1 1].

| Input | | | Net | output | Target | Weight Changes | | | Weights | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $x_1$ | $x_2$ | b | $y_{in}$ | y | t | $\Delta w_1$ | $\Delta w_2$ | $\Delta b$ | $w_1$ | $w_2$ | b |
| | | | | | | | | | (0 | 0 | 0) |
| ~~1~~ | ~~1~~ | ~~1~~ | ~~0~~ | ~~0~~ | ~~1~~ | ~~1~~ | ~~1~~ | ~~1~~ | ~~1~~ | ~~1~~ | ~~1~~ |
| ~~-1~~ | ~~1~~ | ~~1~~ | ~~1~~ | ~~1~~ | ~~-1~~ | | | | ~~2~~ | | |
| ~~1~~ | ~~-1~~ | ~~1~~ | | | | | | | | | |
| ~~-1~~ | ~~-1~~ | ~~1~~ | | | | | | | | | |

A = (0 - 1).

| Input | | b | Net | output | Target | Weight change | | | New weights | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $x_1$ | $x_2$ | b | $y_{in}$ | y | t | $\Delta w_1$ | $\Delta w_2$ | $\Delta b$ | $w_1$ | $w_2$ | b |
| | | | | | | | | | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| -1 | 1 | 1 | 1 | 1 | -1 | 1 | -1 | -1 | 2 | 0 | 0 |
| 1 | -1 | 1 | 2 | 1 | -1 | -1 | 1 | -1 | 1 | 1 | -1 |
| -1 | -1 | 1 | -3 | -1 | -1 | 0 | 0 | 0 | 1 | 1 | -1 |

This complete one epoch of the training
The final weight after the first epoch is completed are, $w_1=1, w_2=1, b=-1$

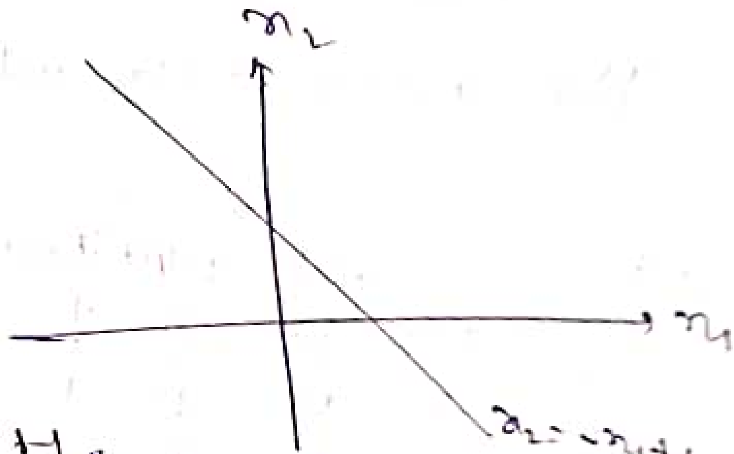
We know that

$$b + x_1 w_1 + x_2 w_2 = 0$$

$$\boxed{x_2 = -x_1 \frac{w_1}{w_2} - \frac{b}{w_2}}$$

$$x_2 = -x_1 \frac{1}{1} - \frac{(-1)}{1}$$

$$x_2 = -x + 1$$




$x_2 = -x+1$ is the separating line equation

Ex-4.2 Develop a perceptron for the AND function with binary inputs and bipolar targets without bias up to 2 epochs. (Take first with (0,0) and next without (0,0)).

Soln

$w_1 = w_2 = 0$

$\alpha = 1, \ \theta = 0.$

(a) With (0,0) and without bias. $\boxed{\alpha = 1}$

net input $y_{in} = \sum x_i w_i$

$$y = f(y_{in}) = \begin{cases} 1 & \text{if } y_{in} > 0 \\ 0 & \text{if } y_{in} = 0 \\ -1 & \text{if } y_{in} < 0 \end{cases}$$

The weight change $\boxed{\Delta w_i = \alpha t x_i}$

$\boxed{w(new) = w(old) + \Delta w}$

Epoch-1.

| Input | | Net | O/P | Target | Weight Change | | Weights | |
|---|---|---|---|---|---|---|---|---|
| $x_1$ | $x_2$ | $y_{in}$ | $y$ | $t$ | $\Delta w_1$ | $\Delta w_2$ | $w_1$ (0 | $w_2$ 0) |
| 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| 1 | 0 | 1 | 1 | -1 | -1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | -1 | 0 | -1 | 0 | 0 |
| 0 | 0 | 0 | 0 | -1 | 0 | 0 | 0 | 0 |

The separating line for 1st & 2nd i/p are $x_1 + x_2 = 0$ and $x_2 = 0$ respectively

Epoch: Initial weights used are the final weights from the previous iteration

| Input | | Net | O/P | Target | Weight change | | weights | |
|---|---|---|---|---|---|---|---|---|
| $x_1$ | $x_2$ | $y_{in}$ | $y$ | $t$ | $\Delta w_1$ | $\Delta w_2$ | $w_1$ | $w_2$ |
| | | | | | | | (0 | 0) |
| 1 | 1 | 0 | 0 | 1 | $\cancel{0}$ 1 | 1 | 1 | 1 |
| 1 | 0 | 1 | 1 | -1 | -1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | -1 | 0 | -1 | 0 | 0 |
| 0 | 0 | 0 | 0 | -1 | 0 | 0 | 0 | 0 |

for 1st i/p : $x_1+x_2=0$
for 2nd i/p : $x_2=0$.

(b) Without bias and (1, 0).

Epoch 1:

Ex-4.3 · Sol$^n$

Initial weights are assumed to be zero and the learning rate is 1

if $y \neq t$ weight change

$\boxed{\Delta w = \alpha t x_i}$ $\boxed{\Delta b = \alpha t}$

$$\boxed{\begin{aligned} W_{new} &= W_{old} + \Delta W \\ b_{new} &= b_{old} + \Delta b \end{aligned}}$$

if $t = y$, no weight change

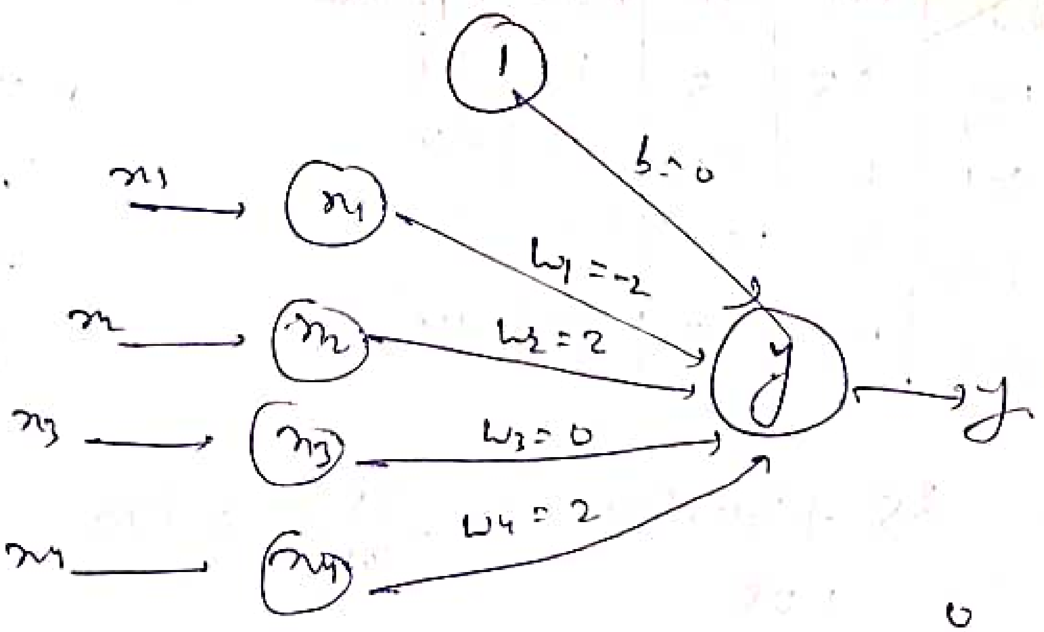
$$\boxed{y_{in} = b + \sum_i x_i w_i}$$

$$y = f(y_{in})$$

$$\boxed{y_{in} = b + x_1 w_1 + x_2 w_2 + x_3 w_3 + x_4 w_4}$$

$$y = f(y_{in}) = \begin{cases} 1 & \text{if } y_{in} > 0 \\ 0 & \text{if } y_{in} = 0 \\ 1 & \text{if } y_{in} < 0 \end{cases}$$

| Input | | | | | Net | Output | Target | Weight Changes | | | | | Weights | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $x_1$ | $x_2$ | $x_3$ | $x_4$ | b | $y_{in}$ | y | t | $\Delta w_1$ | $\Delta w_2$ | $\Delta w_3$ | $\Delta w_4$ | $\Delta b$ | $w_1$ | $w_2$ | $w_3$ | $w_4$ | |
| | | | | | | | | | | | | | 0 | 0 | 0 | 0 | 0 |
| Epoch-1 | | | | | | | | | | | | | | | | | |
| 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| -1 | 1 | -1 | -1 | 1 | -1 | -1 | 1 | -1 | 1 | -1 | -1 | 1 | 0 | 2 | 0 | 0 | 2 |
| 1 | 1 | 1 | -1 | 1 | 4 | 1 | -1 | -1 | -1 | -1 | 1 | -1 | -1 | 1 | -1 | 1 | 1 |
| -1 | -1 | -1 | 1 | 1 | 1 | 1 | -1 | -1 | 1 | 1 | -1 | -1 | -2 | 2 | 0 | 0 | 0 |
| | | | | | | | | | | | | | initial val | | | | |
| Epoch-2 | | | | | | | | | | | | | | | | | |
| 1 | 1 | 1 | 1 | | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | -1 | 3 | 1 | 1 | 1 |
| -1 | 1 | -1 | -1 | | 3 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | ~~-1~~ | 3 | ~~0~~ | ~~0~~ | ~~1~~ |
| 1 | 1 | 1 | -1 | | | | -1 | | | | | | | | | | |
| 1 | -1 | -1 | 1 | | | | -1 | | | | | | | | | | |




|  |  | Predicted 0 | 1 |
|---|---|---|---|
| Actual | 0 | TN | FP |
|  | 1 | FN | TP |

## Classification

(A) $\text{Accuracy} = \dfrac{T_P + T_N}{T_P + T_N + F_P + F_N} = \dfrac{\text{No. of correctly classified}}{\text{Total no. of prediction}}$

(P) $\text{Precision} = \dfrac{T_P}{T_P + F_P}$ = what

(R) $\text{Recall} = \dfrac{T_P}{T_P + F_N}$

$F_1 \text{ score} = \dfrac{2PR}{P+R}$

P ↑ R ↓ ———— P ↓ R ↑

Ex

| Actual \ Predicted | Dog | Cat | Rabbit | Σ prediction |
|---|---|---|---|---|
| Dog | 25 | 5 | 10 | 40 |
| Cat | 0 | 30 | 4 | 34 |
| Rabbit | 4 | 10 | 20 | 34 |
| Σ actual | 29 | 45 | 34 | 108 |

$\frac{T_P}{T_P+F_P}$

$\frac{T_P}{T_P+F_N}$

$$A = \frac{25+30+20}{108} = \frac{75}{108} = 0.694$$

$$(\text{Precision})_{Dog} = \frac{25}{29} = 0.862$$

$$P_{Cat} = \frac{30}{45} = 0.66, \quad P_{Rabbit} = \frac{20}{34} = 0.588$$

$$\text{Macro Precision} = \frac{0.862+0.66+0.588}{3} = 0.7$$

$$\text{Weighted Precision} = \frac{40}{108} \times 0.862 + \frac{34}{108} \times 0.66 + \frac{34}{108} \times 0.588$$
$$= 0.7114$$

$$(\text{Recall})_{Dog} = \frac{25}{40} = 0.625$$

$$R_{Cat} = \frac{30}{34} = 0.883, \quad R_{rabbit} = \frac{20}{34} = 0.588$$

$$\text{macro Recall} = \frac{0.625+0.883+0.588}{3} = 0.698$$

$$\text{Weighted Recall} = \frac{29}{108} \times 0.625 + \frac{45}{108} \times 0.883 + \frac{34}{108} \times 0.588$$
$$= 0.7204$$

## → PCA Problem

Q:1 Given the following data, use PCA to reduce the dimension from 2 to 1.

| Features | Example 1 | Example 2 | Example 3 | Example 4 |
|---|---|---|---|---|
| $x$ | 4 | 8 | 13 | 7 |
| $y$ | 11 | 4 | 5 | 14 |

Ans). Step 1: Dataset

No. of features, $n = 2$
No. of samples, $N = 4$

Step 2: Computation of mean of variables

$$\bar{x} = \frac{4+8+13+7}{4} = 8$$

$$\bar{y} = \frac{11+4+5+14}{4} = 8.5$$

Step 3: Computation of covariance matrix.

Ordered pairs are

$x, y$

$(x,x)$, $(x,y)$, $(y,x)$, $(y,y)$

variable $n$
order pairs $= n^2$

1) Covariance of all ordered pairs

$$Cov(x,x) = \frac{1}{N-1}\sum_{k=1}^{N}(x_{ik}-\bar{x}_i)(x_{jk}-\bar{x}_j)$$

$$= \frac{1}{4-1}\left[(4-8)^2 + (8-8)^2 + (13-8)^2 + (7-8)^2\right]$$

$$= 14$$

$$\boxed{Cov(x,x) = \frac{1}{N-1}\sum_{K=1}^{N}(x_i - \bar{x})^2}$$

$$Cov(x,y) = \frac{1}{4-1}\left[(4-8)(11-8.5) + (8-8)(4-8.5) + (13-8)(5-8.5) + (7-8)(14-8.5)\right]$$

$$= -11$$

$$Cov(y,x) = Cov(x,y) = -11.$$

$$Cov(y,y) = \frac{1}{4-1}\left[(11-8.5)^2 + (4-8.5)^2 + (5-8.5)^2 + (14-8.5)^2\right]$$

$$= 23.$$

Covariance matrix size $= n \times n$
i.e $2 \times 2$

$$S = \begin{bmatrix} Cov(x,x) & Cov(x,y) \\ Cov(y,x) & Cov(y,y) \end{bmatrix}$$

$$= \begin{bmatrix} 14 & -11 \\ -11 & 23 \end{bmatrix}$$

Step4: Eigen value, Eigen vector, Normalized eigen vector.

i) Eigen value

$I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

$\det(S - \lambda I) = 0$.

$\lambda I = \begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix}$

$$\det\left(\begin{bmatrix} 14-\lambda & -11 \\ -11 & 23-\lambda \end{bmatrix}\right) = 0$$

$\Rightarrow (14-\lambda)(23-\lambda) - (-11)\times(-11) = 0$.

$\Rightarrow \lambda^2 - 37\lambda + 201 = 0$.

$\lambda = 30.3849,\ 6.6151$

$\lambda_1 > \lambda_2$

$\lambda_1 = 30.3849 \Rightarrow$ first principal component

$\lambda_2 = 6.6151$

ii) Eigen vector of $\lambda_1$.

$U_1$ = Eigen vector of $\lambda_1$

$(S - \lambda_1 I)U_1 = 0$

$U_1 = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$

$$\begin{bmatrix} 14-\lambda_1 & -11 \\ -11 & 23-\lambda_1 \end{bmatrix}\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = 0$$

$$\begin{bmatrix} (14-\lambda_1)u_1 - 11u_2 \\ -11u_1 + (23-\lambda_1)u_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$\Rightarrow (14-\lambda_1)u_1 - 11u_2 = 0$ —— (i)

$\Rightarrow -11u_1 + (23-\lambda)u_2 = 0$ —— (ii)

$$\frac{u_1}{11} = \frac{u_2}{14-\lambda_1} = t$$

when $t=1$

$$u_1 = 11$$

$$u_2 = 14-\lambda_1$$

Eigen vector $v_1$ of $\lambda_1 = \begin{bmatrix} 11 \\ 14-\lambda_1 \end{bmatrix}$

$$= \begin{bmatrix} 11 \\ 14-30.3841 \end{bmatrix} = \begin{bmatrix} 11 \\ -16.3849 \end{bmatrix}$$

iii) Normalize the eigen vector $v_1$

$$e_1 = \begin{bmatrix} 11 \\ -16.3849 \end{bmatrix} \rightarrow \begin{bmatrix} 11/\sqrt{11^2+(-16.3849)^2} \\ \dfrac{-16.3849}{\sqrt{11^2+(-16.3849)^2}} \end{bmatrix}$$

$$= \begin{bmatrix} 0.5574 \\ -0.8303 \end{bmatrix}$$

$\lambda_2$ $\quad e_2 = \begin{bmatrix} 0.8303 \\ 0.5574 \end{bmatrix}$

## Step5: Derive new data set

| First principal component $PC_1$ | Ex1 | Ex2 | Ex3 | Ex-4 |
|---|---|---|---|---|
| | $P_{11}$ | $P_{12}$ | $P_{13}$ | $P_{14}$ |

$$P_{11} = e_1^T \begin{bmatrix} 4-8 \\ 11-8.5 \end{bmatrix}$$

$$= \begin{bmatrix} 0.5574 & -0.8303 \end{bmatrix} \begin{bmatrix} -4 \\ 2.5 \end{bmatrix}$$

1×1

$$= -4.3052$$

$$P_{12} = \begin{bmatrix} 0.5574 & -0.8303 \end{bmatrix} \begin{bmatrix} 8-8 \\ 4-8.5 \end{bmatrix}$$

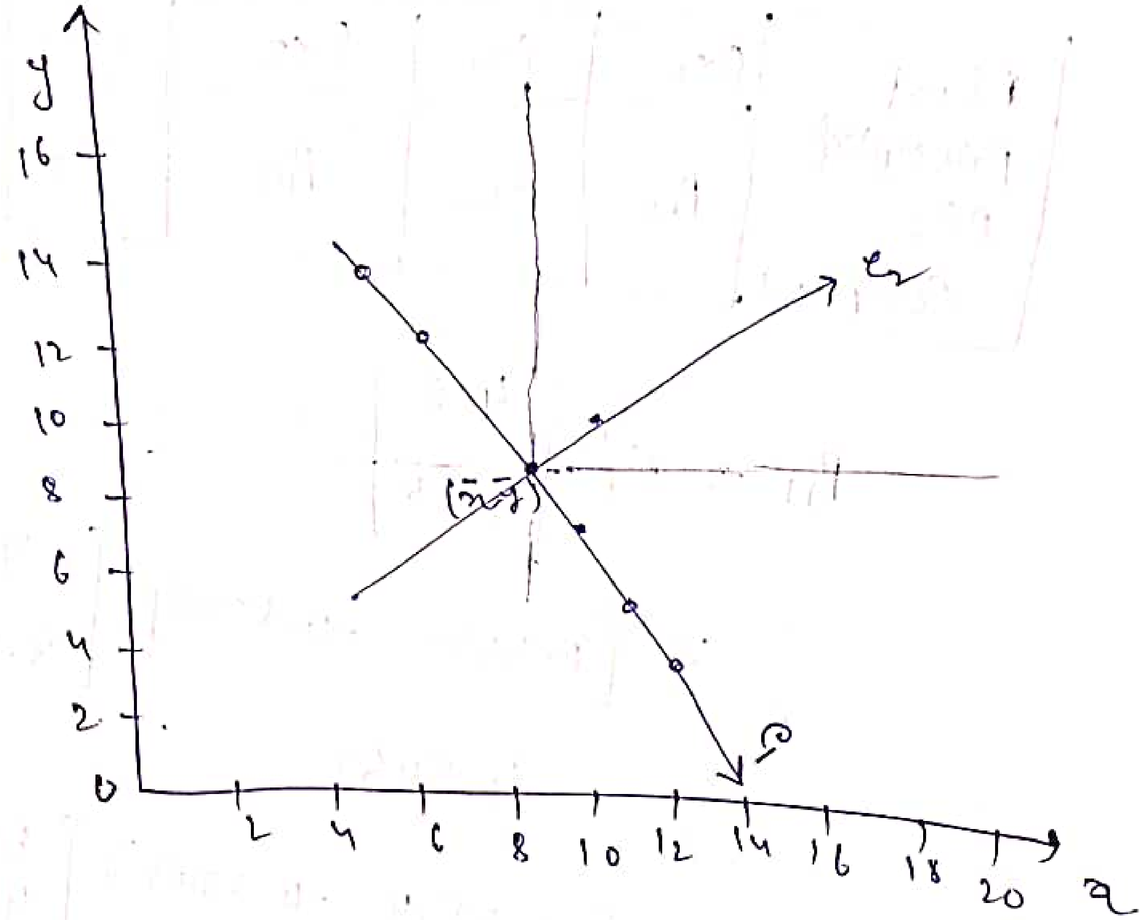
$$= 3.7361$$

$$P_{13} = 5.6928$$

$$P_{17} = -5.1238$$

| | Ex-1 | Ex-2 | Ex-3 | Ex-4 |
|---|---|---|---|---|
| $PC_1$ | -4.3052 | 3.7361 | 5.6928 | -5.1238 |

Coordinate system for principal Components



# LDA

Q.1: Let's trace a 2-D dataset

$C_1 \Rightarrow X_1 = (x_1, x_2) = \{(4,1), (2,4), (2,3), (3,6), (4,4)\}$

$C_2 \Rightarrow X_2 = (x_1, x_2) = \{(9,10), (6,8), (9,5), (8,7), (10,8)\}$

Sol

Step 1: Compute within-class scatter matrix ($S_W$)

$$\boxed{S_W = S_1 + S_2}$$

$S_1$ is the covariance matrix for the class $C_1$ and $S_2$ is for $C_2$

So, lets now find the covariance matrices of each class

$$S_1 = \sum_{x \in w_1} (x - \mu_1)(x - \mu_1)^T$$

$\mu_1$ is the mean of class $C_1$, which is computed by & $X_1$

$$\mu_1 = \left\{ \frac{4+2+2+\cancel{2}+3+4}{5}, \frac{1+4+3+6+4}{5} \right.$$

$$\mu_1 = [3.00 \quad 3.60$$

Similarly, $\mu_2 = [8.4 \quad 7.60]$

$$S_1 = \sum_{x \in w_1} (x - \mu_1)(x - \mu_1)^T \qquad \mu_1 = [3, 3.6]$$

$$(X_1 - \mu_1) = \begin{bmatrix} 1 & -1 & -1 & 0 & 1 \\ -2.6 & 0.4 & -0.6 & 2.4 & 0.4 \end{bmatrix}$$

Now, for each $x$, we are going to calculate $(x-\mu_1)(x-\mu_1)^T$. So, we will have "5" such matrices

We will go one by one

$$\begin{bmatrix} 1 \\ -2.6 \end{bmatrix} \begin{bmatrix} 1 & -2.6 \end{bmatrix} = \begin{bmatrix} 1 & -2.6 \\ -2.6 & 6.76 \end{bmatrix} \quad \text{— (1)}$$

↳ First matrix

Similarly for rest we get

$$\begin{bmatrix} -1 \\ 0.4 \end{bmatrix} \begin{bmatrix} -1 & 0.4 \end{bmatrix} = \begin{bmatrix} 1 & -0.4 \\ -0.4 & 0.16 \end{bmatrix} \quad \text{— (2)}$$

$$\begin{bmatrix} -1 \\ -0.6 \end{bmatrix} \begin{bmatrix} -1 & -0.6 \end{bmatrix} = \begin{bmatrix} 1 & 0.6 \\ 0.6 & 0.36 \end{bmatrix} \quad \text{— (3)}$$

$$\begin{bmatrix} 0 \\ 2.4 \end{bmatrix} \begin{bmatrix} 0 & 2.4 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 6.76 \end{bmatrix} \quad \text{— (4)}$$

$$\begin{bmatrix} 1 \\ 0.4 \end{bmatrix} \begin{bmatrix} 1 & 0.4 \end{bmatrix} = \begin{bmatrix} 1 & 0.4 \\ 0.4 & 0.16 \end{bmatrix} \quad \text{— (5)}$$

See Sarjana notes

Adding (1) + (2) + (3) + (4) + (5) and taking avg we get Covariance matrix $S_1$

$$S_1 = \begin{bmatrix} 0.8 & -0.4 \\ -0.4 & 2.6 \end{bmatrix}$$

Similarly for the class 2, the covariance matrix is given by

$$S_2 = \begin{bmatrix} 1.84 & -0.04 \\ -0.04 & 2.64 \end{bmatrix} \quad \& \quad \mu_2 = \begin{bmatrix} 8.4 & 7.6 \end{bmatrix}$$

$$S_W = S_1 + S_2$$

$$S_W = \begin{bmatrix} 2.64 & -0.44 \\ -0.44 & 6.28 \end{bmatrix}$$

Step2: Compute B/w Class scatter matrix ($S_B$)

$$S_B = (\mu_1 - \mu_2)(\mu_1 - \mu_2)^T$$

$$= \begin{pmatrix} -5.4 \\ -4 \end{pmatrix} \begin{pmatrix} -5.4 & -4 \end{pmatrix} = \begin{pmatrix} 29.16 & 21.6 \\ 21.6 & 16.00 \end{pmatrix}$$

Step3: Find the best LDA projection vector

Similar to PCA
we find this using eigen vectors having largest eigen value

$$S_W^{-1} S_B V = \lambda \overset{\text{Projection vector}}{V} \quad \text{--- (a)}$$

$$|S_W^{-1} S_B - \lambda I| = 0$$

$$\begin{vmatrix} 11.89-\lambda & 8.81 \\ 5.08 & 3.76-\lambda \end{vmatrix} = 0$$

Solving $\lambda$, we get $\lambda = 15.65$

Substituting in $\lambda$ in (a)

$$\begin{bmatrix} 11.89 & 8.81 \\ 5.08 & 3.76 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = 15.65 \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}$$

we get $\begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \begin{bmatrix} 0.91 \\ 0.39 \end{bmatrix}$

or we get directly solve

$$\begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = S_W^{-1} (\mu_1 - \mu_2)$$

$$= \begin{bmatrix} 0.1921 & -0.032 \\ -0.031 & 0.38 \end{bmatrix} \begin{bmatrix} -5.4 \\ -4 \end{bmatrix}$$

$= [-0.91 \quad -0.39]^T$

Note:

$S_W^{-1}$ is found by using the formula

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}^{-1} = \frac{1}{ad-bc}\begin{bmatrix} d & -b \\ -c & a \end{bmatrix}.$$

So,

$$S_W = \begin{bmatrix} 2.64 & -0.44 \\ -0.44 & 5.28 \end{bmatrix}$$

$$S_W^{-1} = \frac{1}{13.74}\begin{bmatrix} 5.28 & 0.44 \\ 0.44 & 2.64 \end{bmatrix}$$

$$= \begin{bmatrix} 0.384 & 0.032 \\ 0.032 & 0.192 \end{bmatrix}$$

Step 4: Dimension Reduction

$y = W^T x$ → input data sample

↳ projection vector