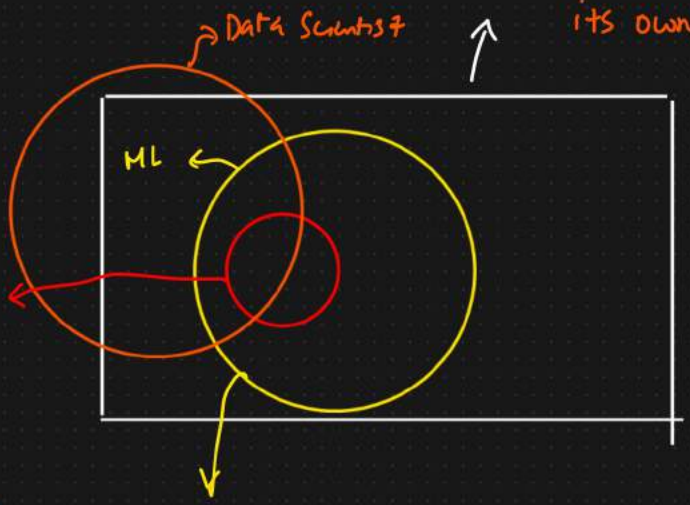
AI Vs ML Vs DL Vs DS

AI → Smart Application that can perform its own task without any human intervention



Eg: Self Driving car
Robots
Alexa

Deep Learning
Mimic the human Brain
{ Multi Layered NN }.

Eg: Object Detection
Image recognition
CHATBOTS
Recommendation Systems

① Statistics
② Linear Algebra
③ Calculus
④ Probability

It provide stats tool to analyze, visualize, predictive models, forecasting

Eg: Amazon.in → Product → Recommendation System
Netflix → Movies

PW SKILLS

Supervised, Unsupervised, Semi- Supervised, Reinforcement Learning

Types of ML

1. Supervised ML → CLASSIFICATION, REGRESSION
2. Unsupervised ML
3. Semi Supervised
4. Reinforcement Learning

1. Supervised ML → Classification, Regression

i. Dataset → O/p feature of the Dataset

CLASSIFICATION

O/p features → Dependent feature

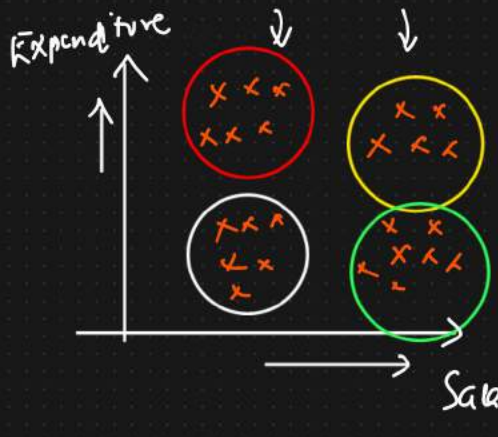
No. of hours played	No. of study hours	Pass/Fail
8	2	Fail
7	3	Fail
6	4	Fail
5	5	Pass
4	6	Pass

Regression → O/p → continuous Value

Size of house	No. of Room	Price of the House ⇑ o/p
—	—	⇒ continuous value

② Unsupervised ML → No O/P ⇒ Clusters → Group of Similar DATA

Eg: Customer Segmentation

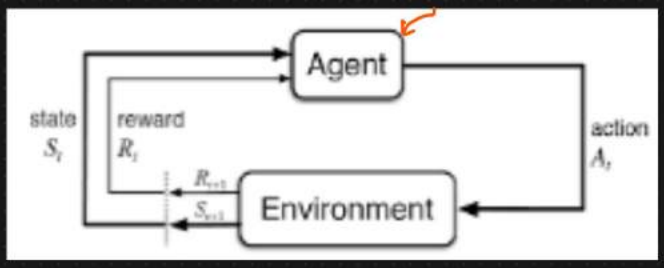


Salary Expenditure

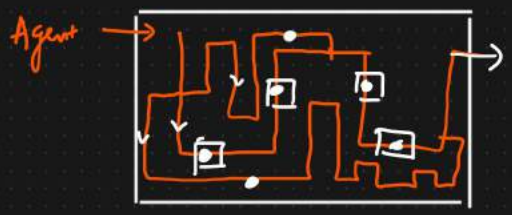
MALL → Watches → Discount Coupon

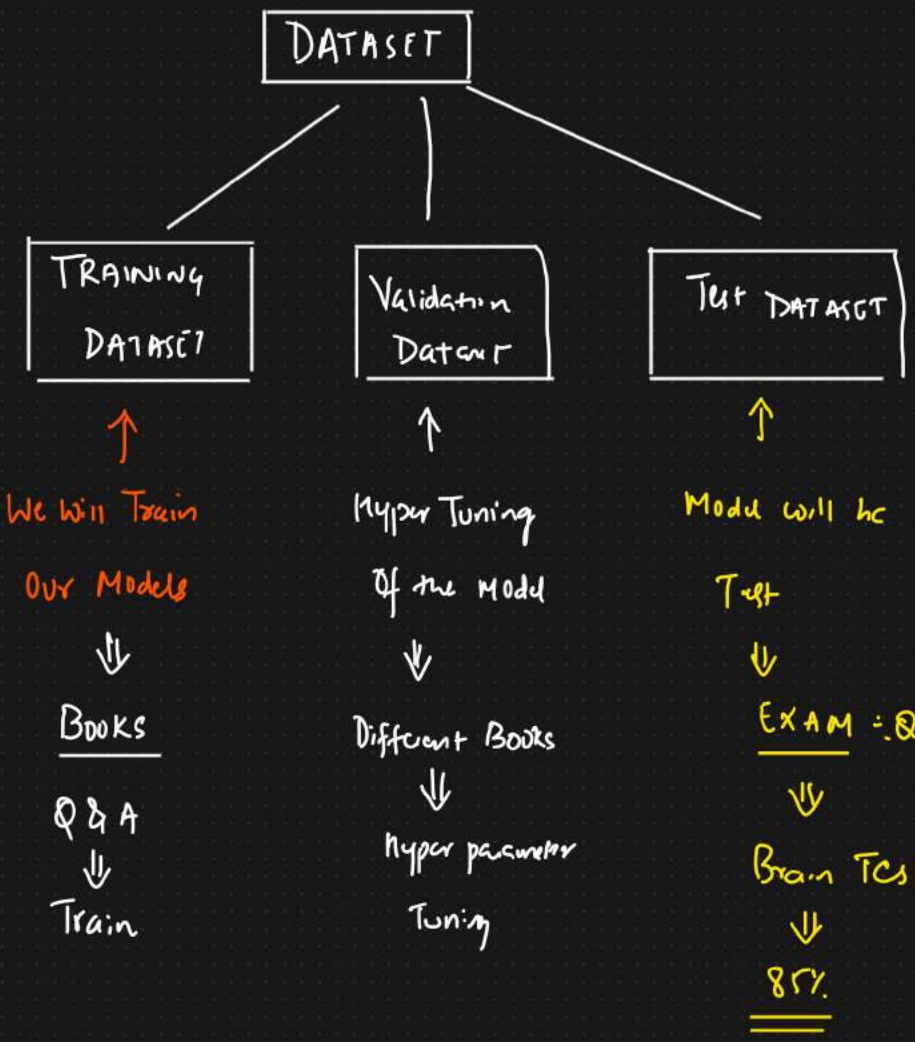
③ Semi Supervised : Supervised + Unsupervised

④ Reinforcement Learning :



Reinforcement learning is an area of machine learning concerned with how intelligent agents ought to take actions in an environment in order to maximize the notion of cumulative reward. Reinforcement learning is one of three basic machine learning paradigms, alongside supervised learning and unsupervised learning.




DATASET
TRAINING DATASET
Validation Dataset
Test DATASET
We will Train Our Models
Hyper Tuning Of the Model
Model will be Test
Books
Q & A
Train
Different Books
Hyper parameter Tuning
EXAM : Question paper
Brain Test
85%

Eg: Exams

Machine Learning

① Model Performance — Accuracy ↑↑ → High

② Overfitting, Underfitting

③ Bias Vs Variance

① Overfitting Underfitting

DATASET

① Books → Train → Model is Trained → Accuracy ↑↑ → 95%
Exam → Test → Model is Tested → Accuracy ↓↓ → 60%

} Overfitting ⇓ { Low Bias, High Variance }

② Train → Accuracy ↓↓ → 55%
Test → Accuracy ↓↓ → 50%

} Underfitting ⇓ { High Bias, High Variance }

Generalized Model ↑

Train → Acc ↑↑
Test → Acc ↑↑

⇓

{ Low Bias, Low Variance }

Feature Extraction

Feature Extraction is process of selecting and extracting the most importance features from raw data.

ML Application → 1000 features

⇓

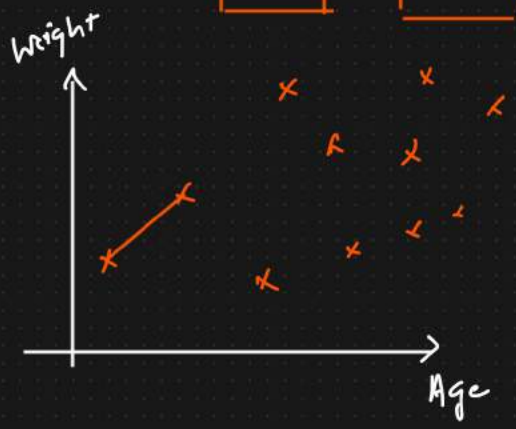
Most Important features

⇓

Machine learning Algo.

① Feature Scaling

(y) Age	(kg) Weight	(cm) Height	BMI
32	70	140cm	–
28	75	160cm	–
35	80	155cm	–



Normalize or Standardize

⇓

Z-score = $\boxed{\mu=0 \quad \sigma=1}$

$$\boxed{Z_{score} = \frac{x_i - \bar{x}}{\mu}}$$

⇓

Standardization

Normalization [Min Max Scaler]

(0,1)

Unit Vector

② Feature Selection → We just pick the most Important feature

500 features → Top 10 features

↓

ML Model Train

Correlation

↓

Feature Selection

① Filter Method ② Embedded Method

③ PCA {Principal Component Analysis}.

Feature Scaling

① Standardization → Z-score

② Min Max Scaling {Normalization}

③ Unit vector.

① Standardization

Age
24
25
26
27
28
29

$$Z\text{-score} = \frac{x_i - \bar{x}}{\sigma}$$

$\mu = 0, \sigma = 1$

Age' → $\mu = 0$ $\sigma = 1$

② Normalization [Min Max Scaler] → 0 to 1

Age
24
25
26
27
28
30

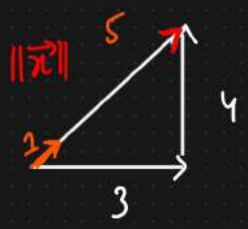
TRANSFORMATION ⟹

$$x_{scaled} = \frac{x_i - x_{min}}{x_{max} - x_{min}}$$

Age' [0-1]

③ Unit Vector - Magnitude of 1

$\vec{x} = (3, 4)$



$\|\vec{x}\| = \sqrt{(3)^2 + (4)^2} = \sqrt{25}$

$= 5/1$

$\hat{u} = \left(\frac{3}{\|\vec{x}\|}, \frac{4}{\|\vec{x}\|}\right) = \left(\frac{3}{5}, \frac{4}{5}\right)$

$\|\hat{u}\| = \sqrt{\left(\frac{3}{5}\right)^2 + \left(\frac{4}{5}\right)^2} = \sqrt{\frac{9+16}{25}}$

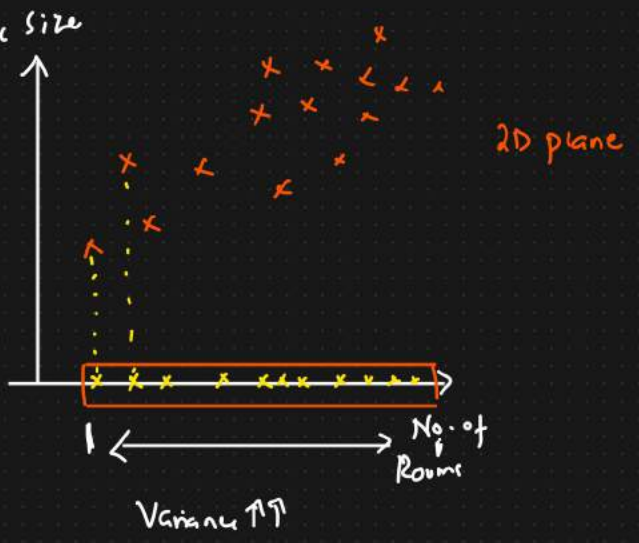
$= \sqrt{25/25}$

$= \underline{1}$

PCA {Principal Component Analysis}.

Dataset

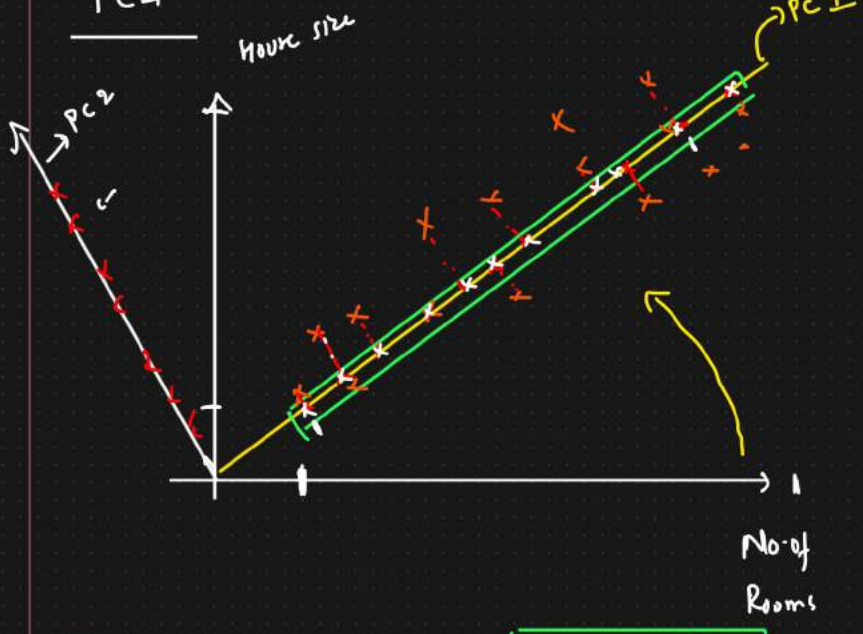
No. of Rooms	House size	Price
2	900	40 lakhs
–	–	–
–	–	–



No. of Rooms Price Machine Learning Model

Problem : Data Loss

PCA



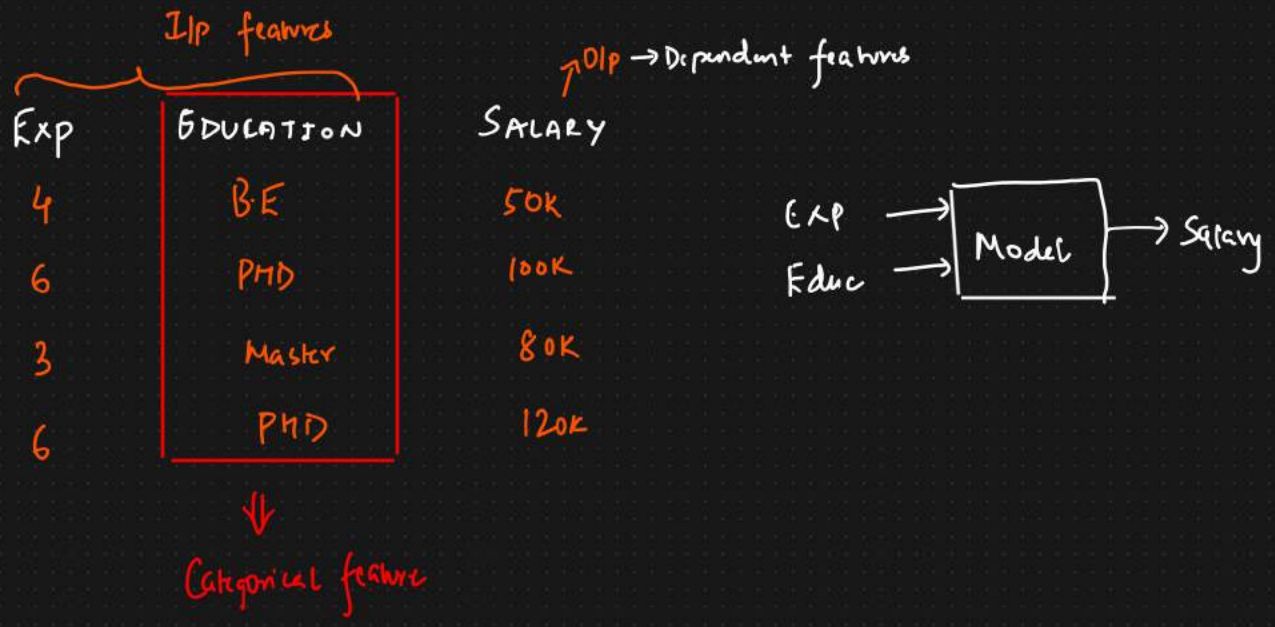
100D → 2D

2D → 1D

3D → 2D

No. of Room House size [PC1 Price] ⇒ Model Train

DATA ENCODING



Data Encoding Arm → Categorical feature → Numerical
⇓
ML Model
⇓
TRAIN.

ⓚ Types of Data Encoding

① Nominal / OHE (One Hot Encoding)
② ORDINAL And LABEL ENCODING
③ Target Guided Ordinal Encoding

⇒ Categorical
⇓
Numerical

Nominal / One Hot Encoding Techniques

Nominal encoding is a technique used to transform categorical variables that have no intrinsic ordering into numerical values that can be used in machine learning models. One common method for nominal encoding is one-hot encoding, which creates a binary vector for each category in the variable.

House Price (DATASET)

No. of Rooms	House Size	Location	Price
		Bangalore	
		Noida	
		Delhi	
		Bangalore	
		Delhi	
		Noida	

Location → OHE

Bangalore	Delhi	Noida
[1	0	0]
0	0	1
0	1	0
1	0	0
0	1	0

Disadvantage [10 location]

① Sparse matrix [overfitting]

② 1000 features [1000 categories]

⇓

It will increase the number of the features

PW SKILLS

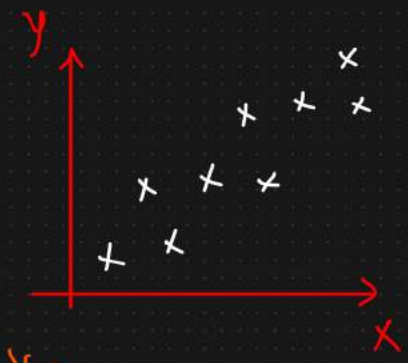
Covariance And Correlation

[Relationship between X and Y]

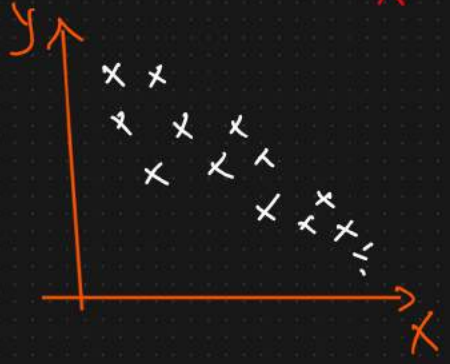
X	Y
2	3
4	5
6	7
8	9

X↑	Y↑
X↓	Y↑
X↓	Y↓
X↑	Y↓

↓↑ Size of House → Price of House ↑↓ O/P



X↑	Y↑
X↓	Y↓



X↑	Y↓
X↓	Y↑

Covariance

$$Cov(x,y) = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{n-1}$$

$$Var(x) = \frac{\sum_{i=1}^{n} (x_i - \bar{x})^2}{n-1} = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(x_i - \bar{x})}{n-1} = Cov(x,x) \Rightarrow \text{Spread}$$

x_i → Datapoints of x

$\bar{x}$ → Sample mean of x

y_i → Datapoints of y

$\bar{y}$ → Sample mean of y

Cov(x,y)

x↑	y↑
x↓	y↓

+ve Covariance

x↑	y↓
x↓	y↑

-ve Covariance

X	y
→ 2	3
→ 4	5
→ 6	7
$\bar{x} = 4$	$\bar{y} = 5$

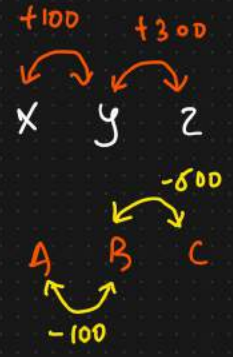
$$Cov(x,y) = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{n-1}$$

$$= \frac{[(2-4)(3-4) + (4-4)(5-5) + (6-4)(7-5)]}{n-1}$$

$$= \frac{4+0+4}{2} = \frac{8}{2} = 4 \text{ tve value}$$

⇓

Positive Covariance



X & y are having a positive Covariance

Advantages

① Relationship between X and y

+ve or -ve value

Disadvantages

① Covariance does not have a specific limit value

② Pearson Correlation Coefficient [-1 to 1]

$$\rho_{x,y} = \frac{Cov(x,y)}{\sigma_x \cdot \sigma_y}$$

① The more the value towards +1 the more +ve Correlated it is (x, y)

② The more the value towards -1 the more -ve Correlated it is (x, y)

③ Spearman Rank Correlation [-1 to 1]

$$r_s = \frac{Cov(R(x), R(y))}{\sigma(R(x)) * \sigma(R(y))}$$

X	y	R(x)	R(y)
1	2	5	5
3	4	4	4
5	6	3	3
7	8	2	1
0	7	6	2
8	1	1	6

Feature Selection

+ve	+ve	+ve	≈0	-ve	
Size of House ↑	No. of Rooms ↑	Location ↑	No. of people staying	Haunted	Price ↑

Simple Linear Regression

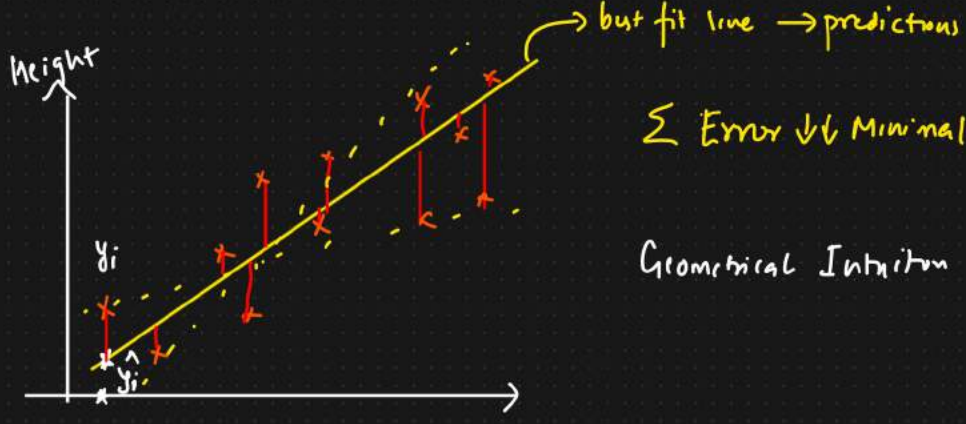
Supervised ML
- → Regression — o/p or Dependent feature → Continuous
- → Classification — o/p " → Categorical

Dataset

Independent feature: Weight	Dependent feature: Height (o/p)
74	170
80	180
75	175.5
—	—
—	—
—	—

TRAIN DATASET → Model → Acc ↑↑

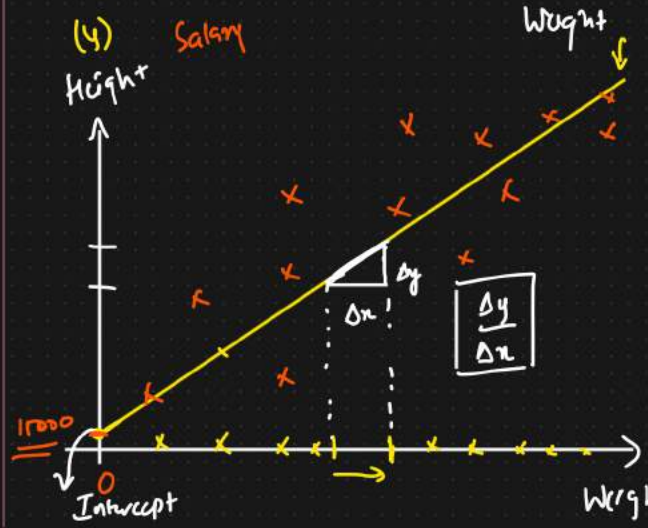
new Weight → Model → prediction: Height



→ best fit line → predictions

Σ Error ↓↓ Minimal

Geometrical Intuition



$\hat{y} = mx + c$

$\hat{y} = \beta_0 + \beta_1 x_1$

$h_\theta(x) = \theta_0 + \theta_1 x_1$

$\theta_0 \Rightarrow$ Intercept

$\theta_1 \Rightarrow$ Slope or coefficient

Simple Linear Regression ⇑ Model ↓ Hypothesis Testing

θ_0, θ_1 → change

Error ↓↓

Experience $\Rightarrow$ x_1 $\Rightarrow$ Datapoint

Cost function [Error].

$$J(\theta_0, \theta_1) = \frac{1}{n} \sum_{i=1}^{n} (\underbrace{y_i}_{\text{Actual}} - \underbrace{h_\theta(x)_i}_{\text{predicted}})^2 \quad \text{[Mean Squared Error]}$$

$h_\theta(x)_i = \hat{y}_i$

n = no. of datapoints

y_i = Actual value

$h_\theta(x)$ = predicted value

Final Aim [In order to get best fit line]

Minimize $J(\theta_0, \theta_1) = \frac{1}{n} \sum_{i=1}^{n} (y_i - h_\theta(x)_i)^2$ ↓↓

θ_0, θ_1

Minimization Error ↓↓

θ_0

$h_\theta(x) = \theta_0 + \theta_1 x_1$

Lets consider $\theta_0 = 0$

$h_\theta(x) = \theta_1 x_1$

Dataset

x	y	$h_\theta(x)$
1	1	
2	2	
3	3	

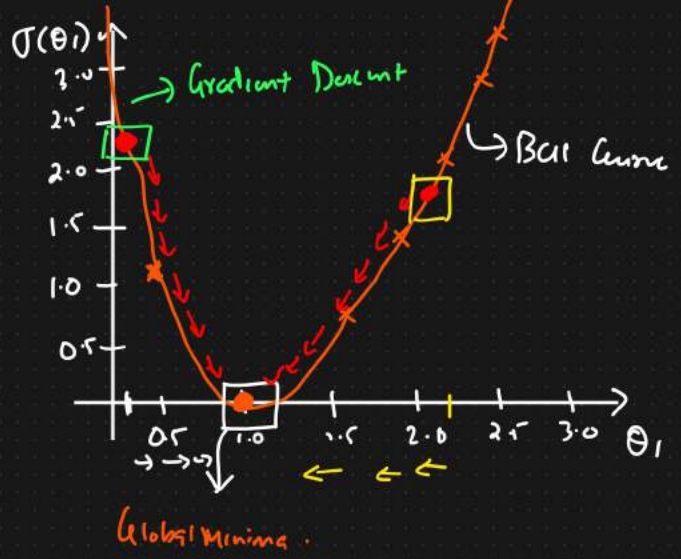
		Let $\theta_1 = 1$	Let $\theta_1 = 0.5$	Let $\theta_1 = 0$
predicted	$x=1$	$h_\theta(x) = 1(1) = 1$	$h_\theta(x) = 0.5$	$h_\theta(x) = 0$
	$x=2$	$h_\theta(x) = 1(2) = 2$	$h_\theta(x) = 1.0$	$h_\theta(x) = 0$
	$x=3$	$h_\theta(x) = 1(3) = 3$	$h_\theta(x) = 1.5$	$h_\theta(x) = 0$

Cost fn $\theta_1 = 1$

$$J(\theta_1) = \frac{1}{n}\sum_{i=1}^{n} (y_i - h_\theta(x)_i)^2$$

$$= \frac{1}{3}\left[(1-1)^2 + (2-2)^2 + (3-3)^2\right].$$

$$J(1) = 0$$



Cost fn $\theta_1 = 0.5$

$$J(\theta_1) = \frac{1}{3}\left[(1-0.5)^2 + (2-1)^2 + (3-1.5)^2\right]$$

$$J(\theta_1) = \underline{\underline{1.16}}$$

Cost fn $\theta_1 = 0$

$$J(\theta_1) = \frac{1}{3}\left[(1-0)^2 + (2-0)^2 + (3-0)^2\right]$$

$$J(\theta_1) = 4.66$$

Convergance Algorithm {Optimize the change of θ_0, θ_1 to Global Minima}.

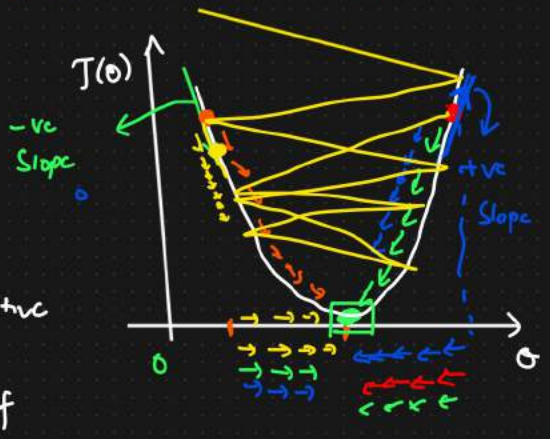
Repeat until Convergance

{ $j = 0, 1$

$\theta_j : \theta_j - \alpha \left[\frac{\partial}{\partial \theta_j} J(\theta_j)\right]$ ⇒ Derivative ⇓ Slope of

}

$\alpha = 0.01$



Learning Rate.

a point

① $\theta_1 = \theta_1 - \alpha(-ve)$

$\theta_{1new} = \theta_{1old} + (value)$

$\theta_{1new} >> \theta_{1old}$

Learning Rate => Speed of Convergence

② $\theta_{1new} = \theta_{1old} - \alpha(+ve)$

$\theta_{1new} = \theta_{1old} - (value)$

$\theta_{1new} << \theta_{1old}$

Conclusion

Repeat until Convergence

{

$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$

}

$J(\theta_0, \theta_1) = \frac{1}{n} \sum_{i=1}^{n} (y_i - h_\theta(x)_i)^2$

⇓

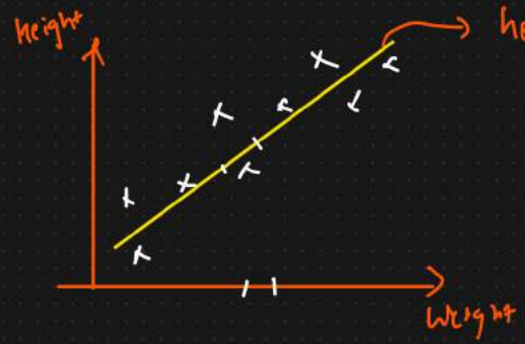
Mean Square Error

Multiple Linear Regression

Dataset

I/P → Weight

O/P → Height



$h_\theta(x) = \theta_0 + \theta_1 x_1$

$\theta_0 \rightarrow$ Intercept

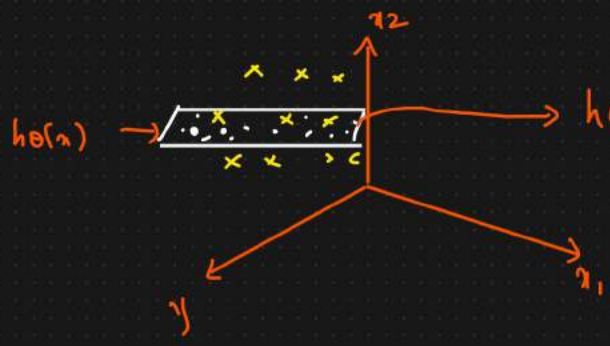
$\theta_1 \rightarrow$ Coefficient or Slope

House Pricing Dataset

x_1 — No. of Rooms

x_2 — Size of Room

y — Price ← Dependent feature



$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2$

$\theta_1, \theta_2 \Rightarrow$ Slope or coefficient

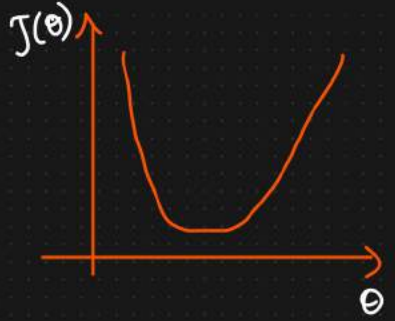
$\theta_0 \Rightarrow$ Intercept

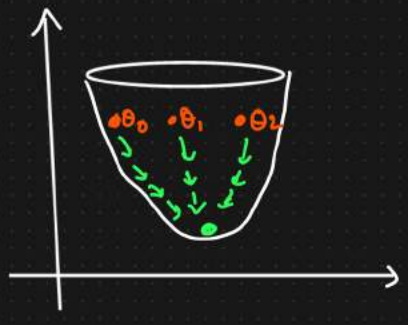
Simple Linear Regression

$$h_\theta(x) = \theta_0 + \theta_1 x_1$$

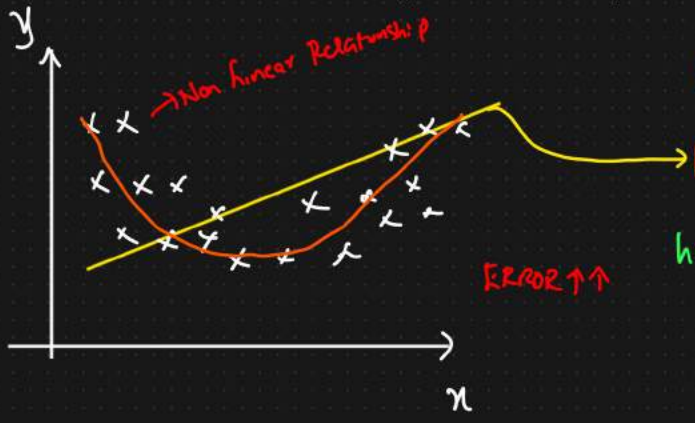
Generic Equation Multiple Linear Regression

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \cdots + \theta_n x_n$$





Polynomial Regression



Simple linear Regression

$h_\theta(x) = \theta_0 + \theta_1 x_1$

$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$

↓

Multiple linear Regression

Polynomial Regression

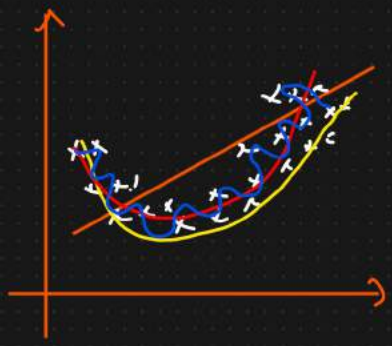
- → Simple Polynomial Regr
- → Multiple Polynomial Regr

⇓

polynomial Degrees

$h_\theta(x) = \theta_0 + \theta_1 x_1$

Simple Polynomial Regression



polynomial degree = 0 → $h_\theta(x) = \theta_0 \times 1 = \theta_0 x_1^0$ ⇒ Constant

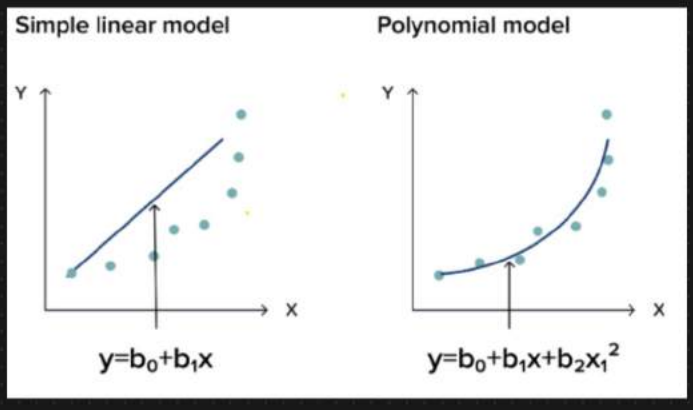
polynomial degree = 1 → $h_\theta(x) = \theta_0 x_1^0 + \theta_1 x_1^1$ ⇒ Simple linear Regression

polynomial degree = 2 → $h_\theta(x) = \theta_0 x_1^0 + \theta_1 x_1^1 + \theta_2 x_1^2$

$h_\theta(x) = \theta_0 x_1^0 + \theta_1 x_1^1 + \theta_2 x_1^2 + \theta_3 x_1^3$

polynomial degree = n

$$h_\theta(x) = \theta_0 x_1^0 + \theta_1 x_1^1 + \theta_2 x_1^2 + \cdots + \theta_n x_1^n$$



① Multiple Polynomial Regression

$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3$ { Multiple Linear Regression }

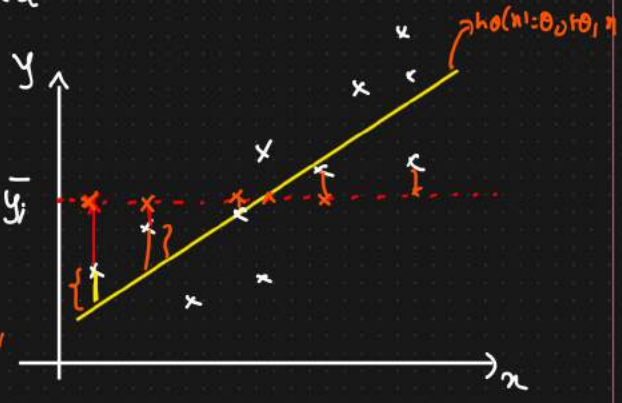
Polynomial Degree = 2

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \theta_4 x_1^2 + \theta_5 x_2^2 + \theta_6 x_3^2$$

Performance Metrics Used In Regression

① R Squared ② Adjusted R Squared

① R Squared



$$R_{squared} = 1 - \frac{SS_{Res}}{SS_{Total}}$$

SS_{Res} → Error

SS_{Total} → {Average of y_i i.e.} → Error

SS_{Res} = Sum of Square Residual {Error}

SS_{Total} = Sum of Square Total

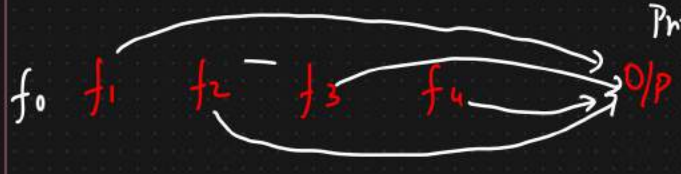
$$R\ Squared = 1 - \frac{\sum_{i=1}^{n} (y_i - h_\theta(x))^2}{\sum_{i=1}^{n} (y_i - \bar{y}_i)^2}$$

SS_{Res} ⇒ Small

$\sum (y_i - \bar{y}_i)^2$ ⇒ Big

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

R Squared ranges between 0 to 1



R squared = 75% = 0.75

R Squared = 80%

R Squared = 90% ↑↑↑

R squared = 92% 77

② Adjusted R squared

R-squared = 85% = 0.85

R-squared = 90% = 0.90

R-squared = 91% = 0.91

$$\text{Adjusted R Square} = 1 - \frac{(1-R^2)(N-1)}{N-p-1}$$

N = no. of datapoints

R^2 = R squared

P = No. of Independent features.

$R^2 = 0.8$ $N = 11$ $p = 2$ $p = 3$ → feature

$$\text{Adjusted R squared} = 1 - \left[\frac{(0.2)(10)}{11-2-1}\right] = 0.75$$

R^2 >> Adjusted R^2

$R^2 = 80\%$ Adjusted $R^2 = 75\%$

$p = 3$ $R^2 = 85\%$ Adjusted $R^2 = 78\%$

p=4 ↑ R^2 = 87% Adjusted R^2 = 76% ↓↓

⇓

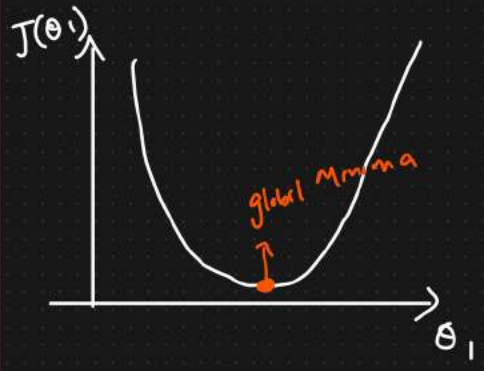
Independent feature is not that important

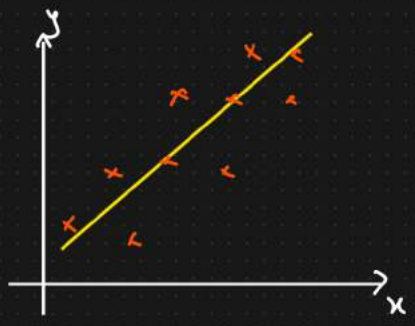
MSE, MAE, RMSE [Cost function]

① Mean Squared Error (MSE) ✓

② Mean Absolute Error (MAE)

③ Root Mean Squared Error (RMSE)





$$J(\theta_0,\theta_1) = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - h_\theta(x)_i\right)^2$$

⇓

Mean Squared Error

① Mean Squared Error (MSE)

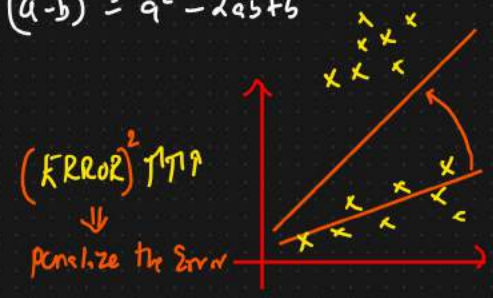
$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - h_\theta(x)_i\right)^2$$

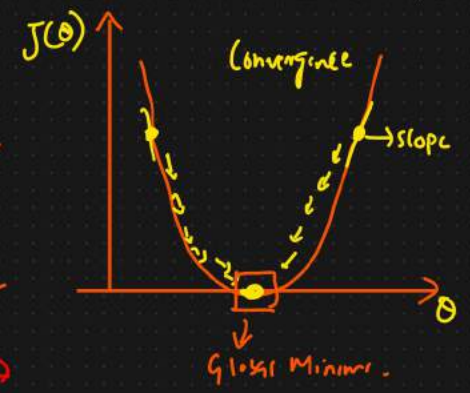
$$\boxed{MSE = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - \hat{y}_i\right)^2} \rightarrow \text{Quadratic Equation}$$

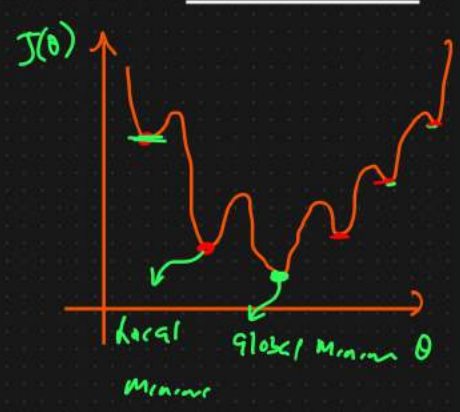
⇓ → Convex function

Non Convex function

$$(a-b)^2 = a^2 - 2ab + b^2$$







Advantage

① Equation is differentiable

② It has only one local or global minima.

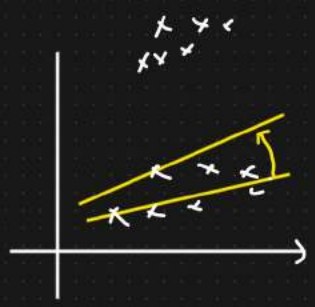
Disadvantage

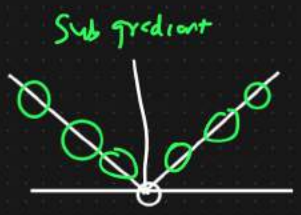
① Not Robust to outliers

② It is not in the same unit

② Mean Absolute Error (MAE)

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





Advantage

1. Robust to outliers
2. It will be in the same unit

Disadvantage

1. Convergence usually takes more time

③ RMSE (Root Mean Squared Error)

$$RMSE = \sqrt{MSE}$$

$$= \sqrt{\frac{1}{n} \sum_{i=1}^{n} (y_i - h_\theta(x)_i)^2}$$

Advantages

1. Same Unit
2. Differentiable

Disadvantage

1. Not Robust to outliers.

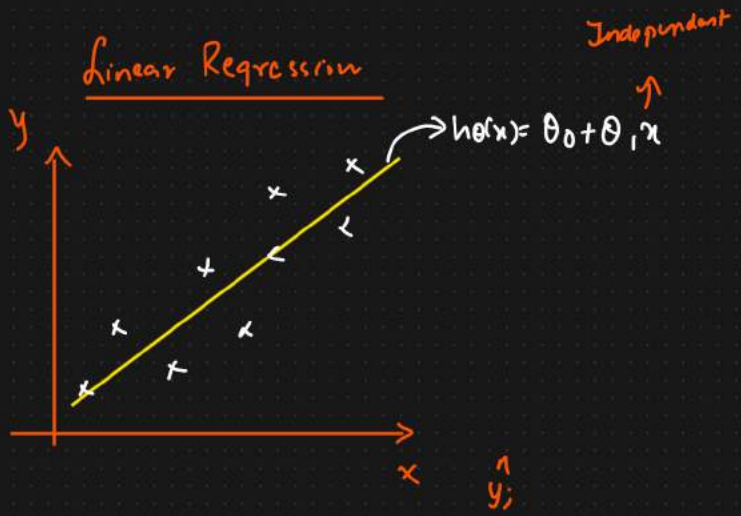
Note: Linear Regression

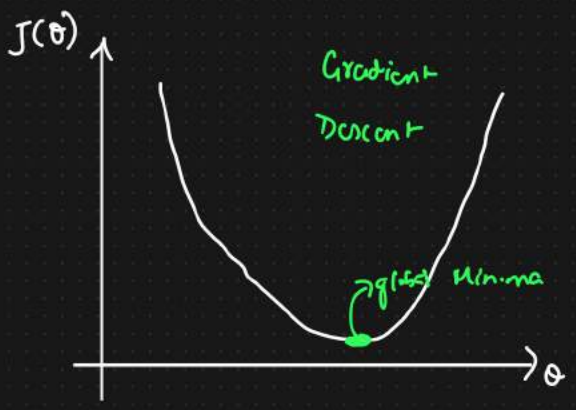
Performance Metrics = R^2 and Adjusted R^2 ⇒ Acc of model

Cost function → Error → MSE, MAE, RMSE

Ridge / Lasso And Elasticnet Regression

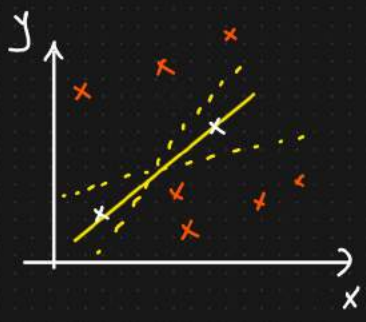
Linear Regression





Cost fn : $\frac{1}{n} \sum_{i=1}^{n} (y_i - h_\theta(x)_i)^2$ [Mean Squared Error].

① Ridge Regression (L2 Regularization) ⟶ Reducing Overfitting



Error = 0

Error ↑↑ Test DATA

DATASET

TRAIN TEST

Overfitting

TRAIN → Acc ↑↑ → R^2

TEST → Acc ↓↓ → R^2

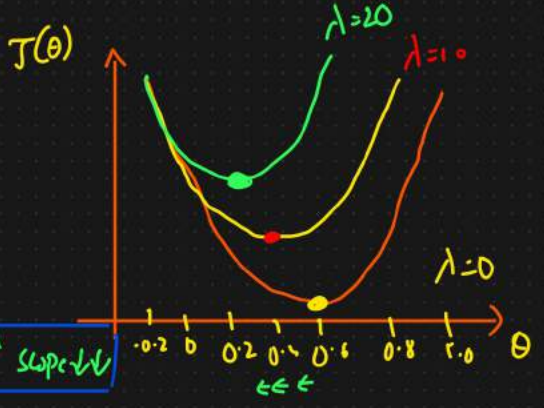
$h_\theta(x) = \theta_0 + \theta_1 x_i$

$\lambda = 1$

Cost fn = $\frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$ + $\lambda \sum_{i=1}^{n} (\text{slope})^2$

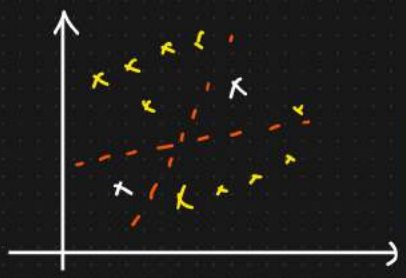
$\Downarrow$ 0

Hyperparameter



$\lambda \uparrow\uparrow$ slope ↓↓

$= 0 + 1 [(\theta_1)^2]$ ⇐ Punalize the cost function



$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3$$

$$\text{Cost fn} = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y})^2 + \lambda \left[(\theta_1)^2 + (\theta_2)^2 + (\theta_3)^2 \right]$$

② Lasso Regression (L1 Regularization) → Feature Selection

$$\text{Cost fn} : \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \lambda \sum_{i=1}^{n} |\text{slope}|$$

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3$$

$$= 0.52 + 0.65 x_1 + 1.5 x_2 + 0.2 x_3$$

feature remove

feature selection

prediction.

$x_2 \rightarrow$ 1 unit

$y \rightarrow 1.5 x_2$

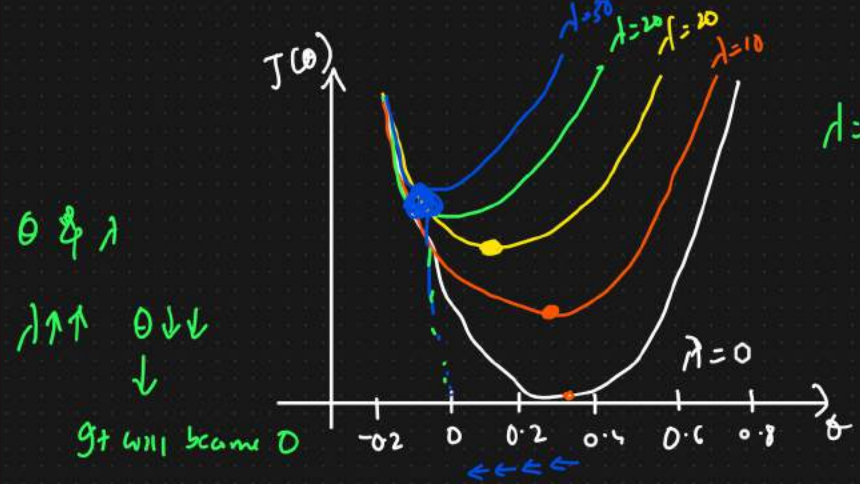
$x_3 \rightarrow$ 1 unit

$y \rightarrow 0.2 x_3$

$\lambda = 1$

$$\text{Cost fn} : \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \lambda \sum_{i=1}^{n} |\text{slope}|$$

$$= \text{Error} + 1 \left[|\theta_1| + |\theta_2| + |\theta_3| \right]$$



λ : Hyperparameter

θ ↓ λ

λ ↑↑ θ ↓↓

↓

gt will became 0

③ ElasticNet Regression

- Reducing Overfitting → Ridge
- Feature Selection → Lasso

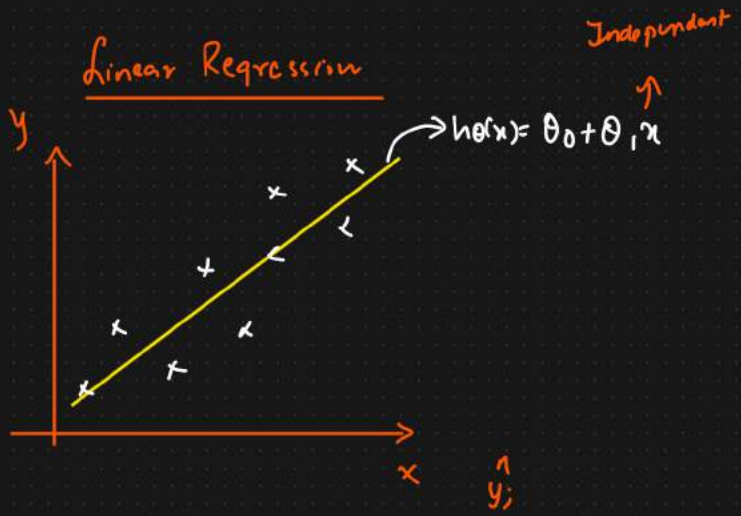
$$\text{Cost fn} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 + \lambda_1 \sum_{i=1}^{n}(\text{slope})^2 + \lambda_2 \sum_{i=1}^{n}|\text{slope}|.$$

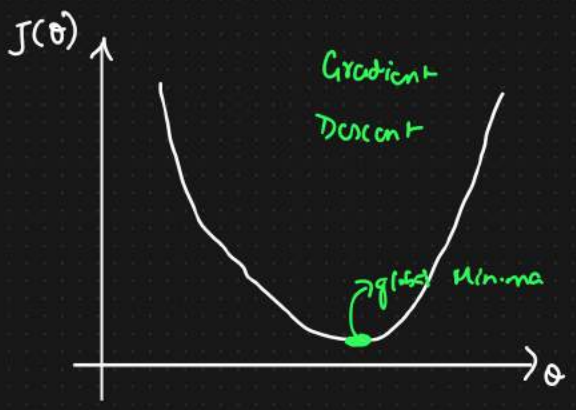
MSE + Reducing overfitting + Feature selection

λ_1, λ_2 {Hyperparameter Tuning}.

Ridge / Lasso And Elasticnet Regression

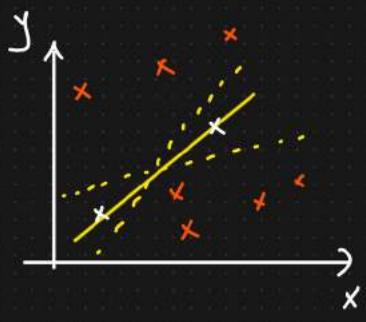
Linear Regression





Cost fn : $\frac{1}{n} \sum_{i=1}^{n} (y_i - h_\theta(x)_i)^2$ [Mean Squared Error].

① Ridge Regression (L2 Regularization) ⟶ Reducing Overfitting



Error = 0

Error ↑↑ Test DATA

DATASET

TRAIN TEST

Overfitting

TRAIN → Acc ↑↑ → R^2

TEST → Acc ↓↓ → R^2

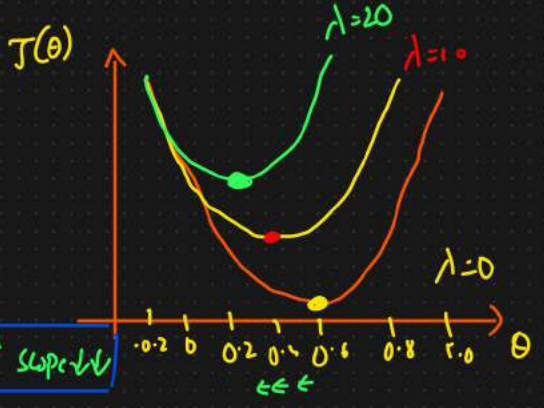
$h_\theta(x) = \theta_0 + \theta_1 x_i$

$\lambda = 1$

Cost fn = $\frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$ + $\lambda \sum_{i=1}^{n} (\text{slope})^2$

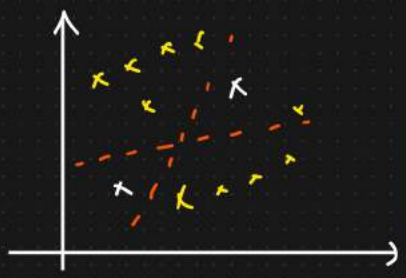
$\Downarrow$ 0

Hyperparameter



$\lambda \uparrow\uparrow$ slope ↓↓

$= 0 + 1 [(\theta_1)^2]$ ⇐ Punalize the cost function



$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3$$

$$\text{Cost fn} = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y})^2 + \lambda \left[(\theta_1)^2 + (\theta_2)^2 + (\theta_3)^2 \right]$$

② Lasso Regression (L1 Regularization) → Feature Selection

$$\text{Cost fn} : \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \lambda \sum_{i=1}^{n} |\text{slope}|$$

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3$$

$$= 0.52 + 0.65 x_1 + 1.5 x_2 + 0.2 x_3$$

feature remove

feature selection

prediction.

$x_2 \rightarrow$ 1 unit

$y \rightarrow 1.5 x_2$

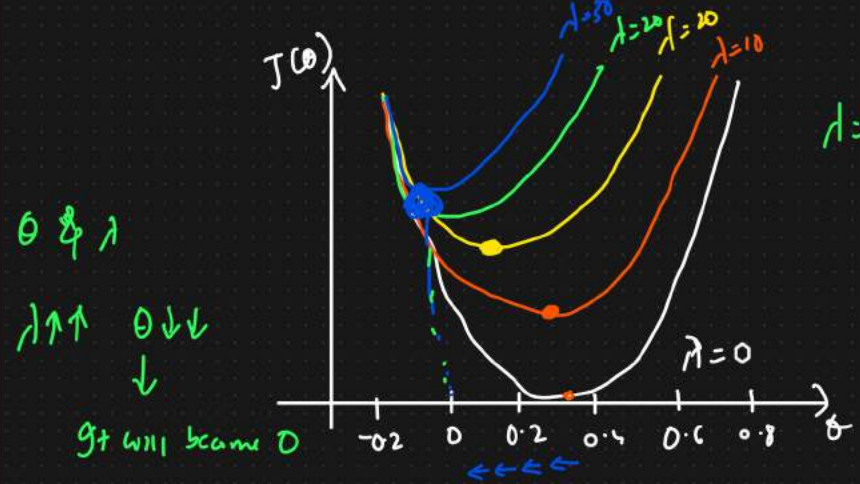
$x_3 \rightarrow$ 1 unit

$y \rightarrow 0.2 x_3$

$\lambda = 1$

$$\text{Cost fn} : \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \lambda \sum_{i=1}^{n} |\text{slope}|$$

$$= \text{Error} + 1 \left[|\theta_1| + |\theta_2| + |\theta_3| \right]$$



λ : Hyperparameter

θ ↓ λ

λ ↑↑ θ ↓↓

↓

gt will became 0

③ ElasticNet Regression

- Reducing Overfitting → Ridge
- Feature Selection → Lasso

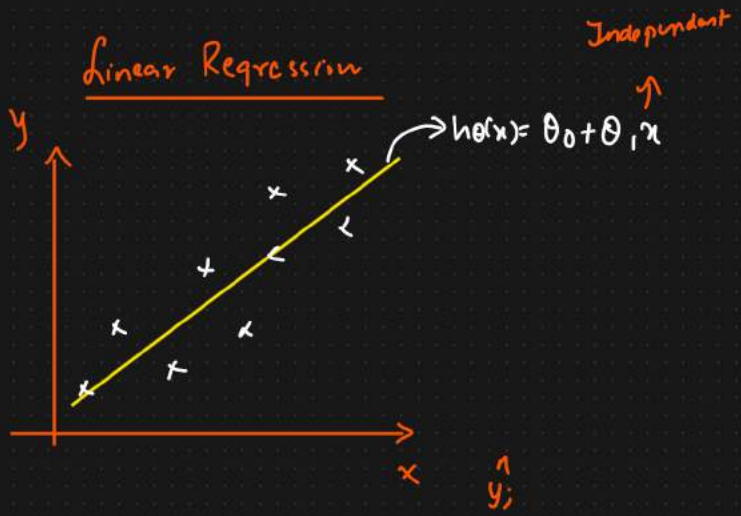
$$\text{Cost fn} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 + \lambda_1 \sum_{i=1}^{n}(\text{slope})^2 + \lambda_2 \sum_{i=1}^{n}|\text{slope}|.$$

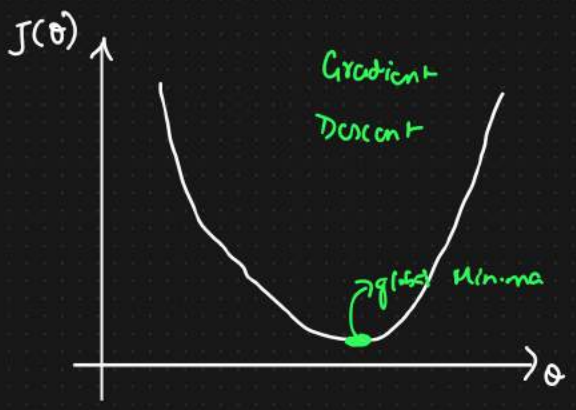
MSE + Reducing overfitting + Feature selection

λ_1, λ_2 {Hyperparameter Tuning}.

Ridge / Lasso And Elasticnet Regression

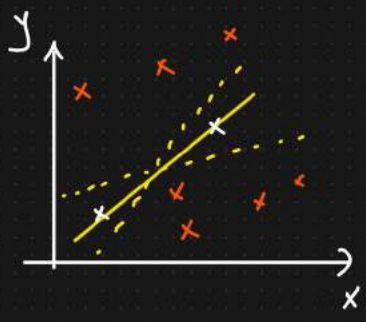
Linear Regression





Cost fn : $\frac{1}{n} \sum_{i=1}^{n} (y_i - h_\theta(x)_i)^2$ [Mean Squared Error].

① Ridge Regression (L2 Regularization) ⟶ Reducing Overfitting



Error = 0

Error ↑↑ Test DATA

DATASET

TRAIN TEST

Overfitting

TRAIN → Acc ↑↑ → R^2

TEST → Acc ↓↓ → R^2

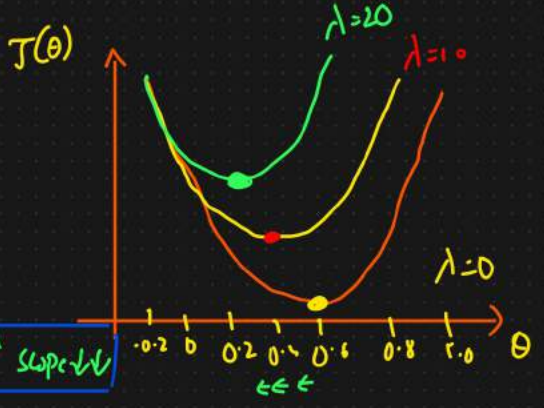
$h_\theta(x) = \theta_0 + \theta_1 x_i$

$\lambda = 1$

Cost fn = $\frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$ + $\lambda \sum_{i=1}^{n} (\text{slope})^2$

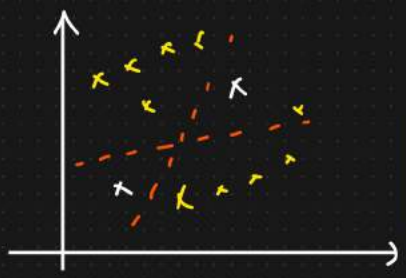
$\Downarrow$ 0

Hyperparameter



$\lambda \uparrow\uparrow$ slope ↓↓

$= 0 + 1 [(\theta_1)^2]$ ⇐ Punalize the cost function



$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3$$

$$\text{Cost fn} = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y})^2 + \lambda \left[(\theta_1)^2 + (\theta_2)^2 + (\theta_3)^2 \right]$$

② Lasso Regression (L1 Regularization) → Feature Selection

$$\text{Cost fn} : \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \lambda \sum_{i=1}^{n} |\text{slope}|$$

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3$$

$$= 0.52 + 0.65 x_1 + 1.5 x_2 + 0.2 x_3$$

feature remove

feature selection

prediction.

$x_2 \rightarrow$ 1 unit

$y \rightarrow 1.5 x_2$

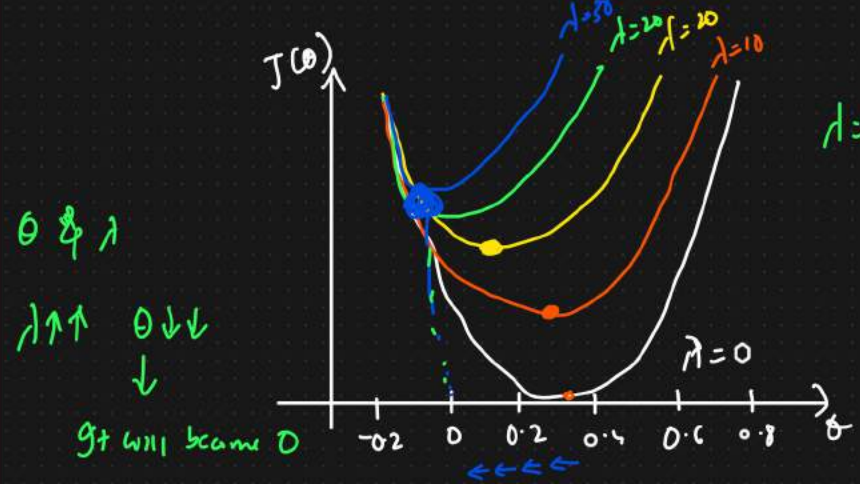
$x_3 \rightarrow$ 1 unit

$y \rightarrow 0.2 x_3$

$\lambda = 1$

$$\text{Cost fn} : \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \lambda \sum_{i=1}^{n} |\text{slope}|$$

$$= \text{Error} + 1 \left[|\theta_1| + |\theta_2| + |\theta_3| \right]$$



λ : Hyperparameter

θ ↓ λ

λ ↑↑ θ ↓↓

↓

gt will became 0

③ ElasticNet Regression

- Reducing Overfitting → Ridge
- Feature Selection → Lasso

$$\text{Cost fn} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 + \lambda_1 \sum_{i=1}^{n}(\text{slope})^2 + \lambda_2 \sum_{i=1}^{n}|\text{slope}|.$$

MSE + Reducing overfitting + Feature selection

λ_1, λ_2 {Hyperparameter Tuning}.

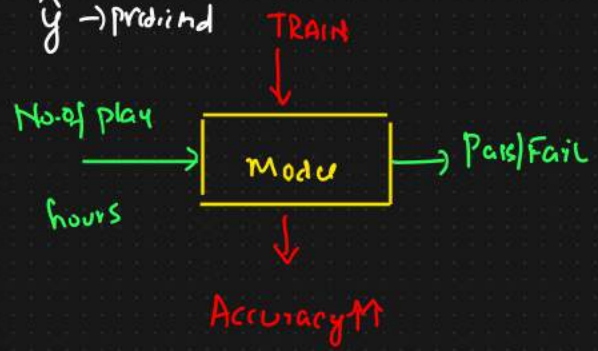
Logistic Regression

To solve classification problem → Binary classification ←
→ Multiclass classification

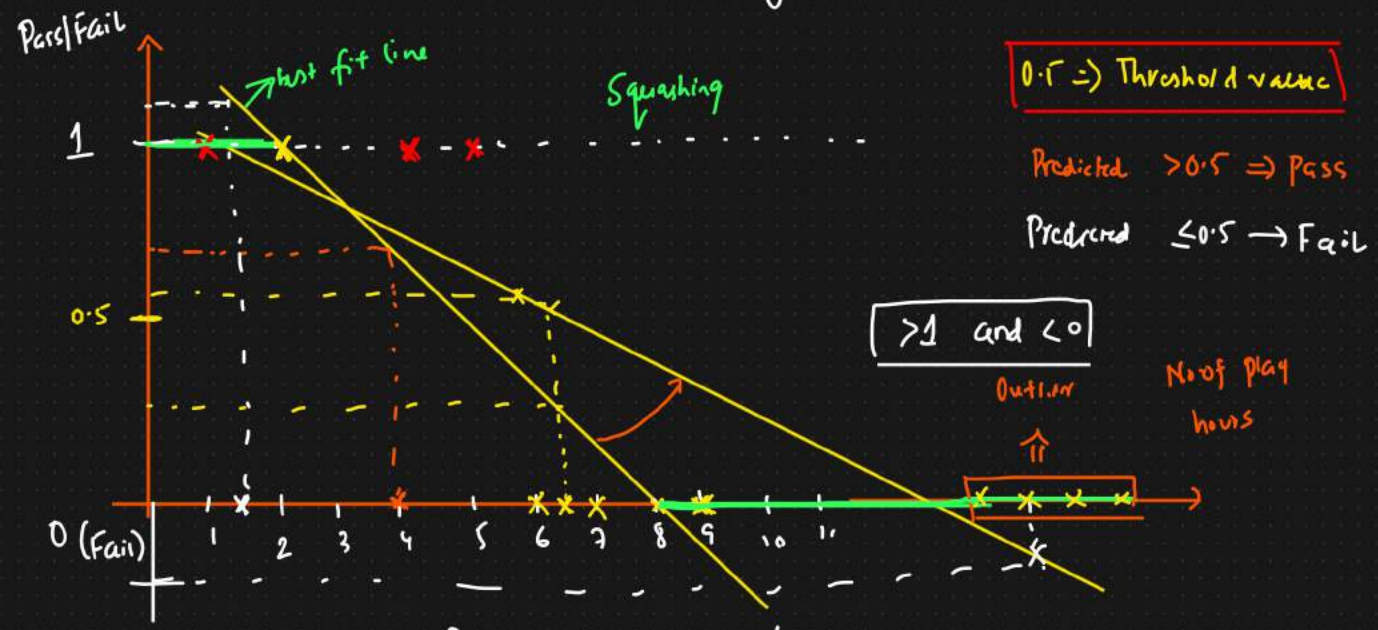
Dataset

Independent feature ↓ — Dependent or o/p feature ↓

No. of play hours	Pass/Fail {y}. $\hat{y}$ → predicted
9	Fail 0
8	Fail 0
7	Fail 0
6	Fail 0
5	Pass 1
4	Pass 1
1	Pass 1
2	Fail 0



Can we solve this classification problem using Regression?



Why we cannot use linear Regression for classification

① Best fit line changes because of outliers → prediction goes wrong

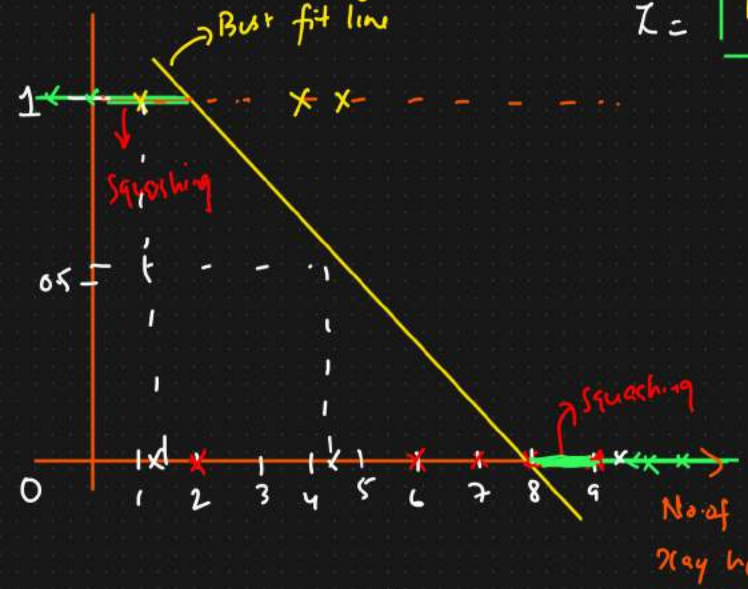
② The outcome comes >1 and < 0 also

To solve this problem we use: Logistic Regression

⇓

0 to 1 ⇒ Squashing Technique

How Logistic Regression Solves Classification Problem



$z =$ $h_\theta(x) = \theta_0 + \theta_1 x_1$ → Best fit line

⇓

[Sigmoid Activation function]

⇓

0 to 1

$$\sigma = \frac{1}{1+e^{-z}} \Rightarrow \text{0 to 1}$$

$$h_\theta(x) = \sigma(\underbrace{\theta_0 + \theta_1 x_1}_{z})$$

$$h_\theta(x) = \frac{1}{1+e^{-z}} \Rightarrow z = \theta_0 + \theta_1 x_1 \Rightarrow h_\theta(x) = \frac{1}{1+e^{-(\theta_0+\theta_1 x_1)}}$$

↳ Logistic Regression

Linear Regression Cost function

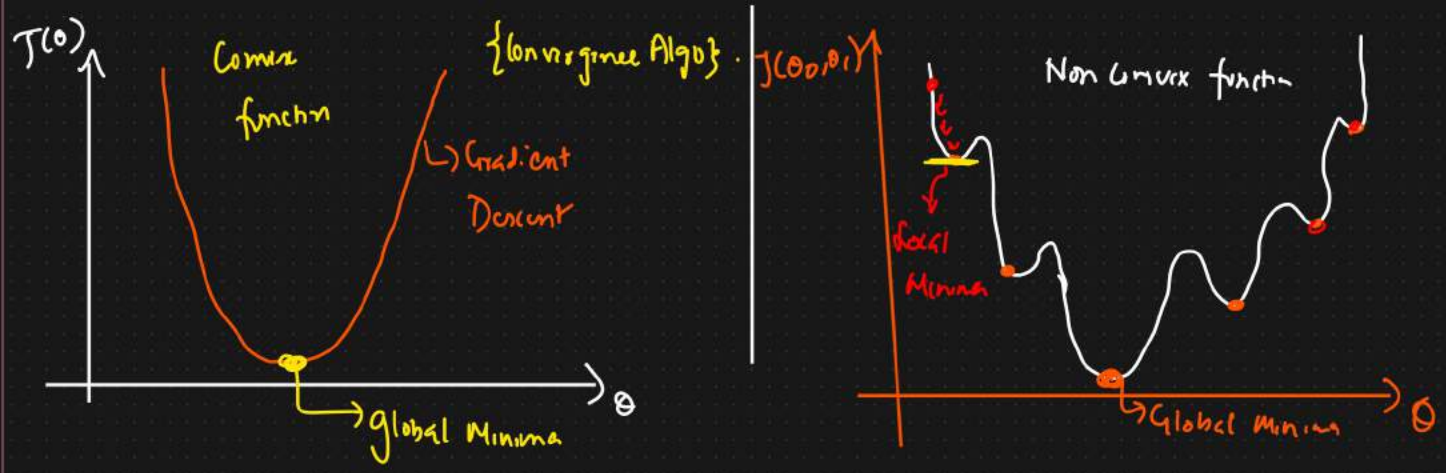
$$J(\theta_0,\theta_1) = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - h_\theta(x)_i\right)^2$$

$$h_\theta(x) = \theta_0 + \theta_1 x_1$$

Logistic Regression Cost function

$$J(\theta_0,\theta_1) = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - h_\theta(x)_i\right)^2$$

$$h_\theta(x) = \frac{1}{1+e^{-z}} = \frac{1}{1+e^{-(\theta_0+\theta_1 x_1)}}$$



Log Loss

$$J(\theta_0,\theta_1) = \begin{cases} -\log(h_\theta(x)) & \text{if } y=1 \\ -\log(1-h_\theta(x)) & \text{if } y=0 \end{cases} \qquad h_\theta(x) = \frac{1}{1+e^{-z}} = \frac{1}{1+e^{-(\theta_0+\theta_1 x_1)}}$$

$$J(\theta_0,\theta_1) = -y\log(h_\theta(x)) - (1-y)\log(1-h_\theta(x)). \Rightarrow \text{Convex function}$$

if $y=1$

$= -\log(h_\theta(x)) \Rightarrow y=1$

if $y=0 \Rightarrow -\log(1-h_\theta(x))$

Final Aim :

Minimize Cost function $J(\theta_0,\theta_1)$ by changing θ_0 & θ_1

Convergence Algorithm

Repeat

{

$j = 0, 1$

$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta} J(\theta_0, \theta_1)$

}

Logistic Regression With Regularization Parameters

Cost function

$$J(\theta_0,\theta_1) = -y\log(h_\theta(x)) - (1-y)\log(1-h_\theta(x))$$

$$h_\theta(x) = \frac{1}{1+e^{-(\theta_0+\theta_1 x_1)}}$$

$$J(\theta_0,\theta_1) = \begin{cases} -\log(h_\theta(x)) & \text{if } y=1 \\ -\log(1-h_\theta(x)) & \text{if } y=0 \end{cases}$$

Reduce Overfitting ⇓

$$J(\theta_0,\theta_1) = -y\log(h_\theta(x)) - (1-y)\log(1-h_\theta(x)) + \lambda_2 \text{ Regularization}$$

feature selection ↙

$$J(\theta_0,\theta_1) = -y\log(h_\theta(x)) - (1-y)\log(1-h_\theta(x)) + \lambda_1 \text{ Regularization}$$

$$J(\theta_0,\theta_1) = -y\log(h_\theta(x)) - (1-y)\log(1-h_\theta(x)) + \lambda_2 \text{ Reg} + \lambda_1 \text{ Reg.}$$

- **λ_2 Regularization** ⇒ Reduce Overfitting

$$J(\theta_0,\theta_1) = -y\log(h_\theta(x)) - (1-y)\log(1-h_\theta(x)) + \lambda \sum_{i=1}^{n} (\text{slope})^2$$

λ_1 Regularization ⇒ Feature selection

λ ⇒ Hyperparameter

$$J(\theta_0, \theta_1) = -y \log(h_\theta(x)) - (1-y) \log(1-h_\theta(x)) + \lambda \sum_{i=1}^{n} |\text{slope}_i|$$

- ElasticNet

$$J(\theta_0, \theta_1) = -y \log(h_\theta(x)) - (1-y) \log(1-h_\theta(x)) + \lambda_1 \sum_{i=1}^{n} (\text{slope}_i)^2 + \lambda_2 \sum_{i=1}^{n} |\text{slope}_i|$$

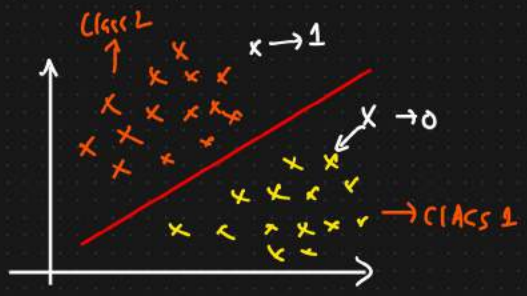
C & λ Relationship

$$C = \frac{1}{\lambda} \Rightarrow \lambda = \frac{1}{C}$$

Performance Metrics, Accuracy, Precision, Recall And F-Beta

Topics to be covered

1. Confusion Matrix ✓
2. Accuracy ✓
3. Precision ✓
4. Recall ✓
5. F-Beta Score



Dataset		Actual	Predicted	
x_1	x_2	y	$\hat{y}$	← Model Prediction
–	–	0	1	→ Wrong Prediction
→ –	–	1	1	→ Correct Prediction
→ –	–	0	0 →	" "
–	–	1	1 →	" "
–	–	1	1 →	" "
–	–	0	1 →	Wrong Prediction
–	–	1	0	

① Confusion Matrix

y ↑ Actual values

	1	0
1	3	2
0	1	1

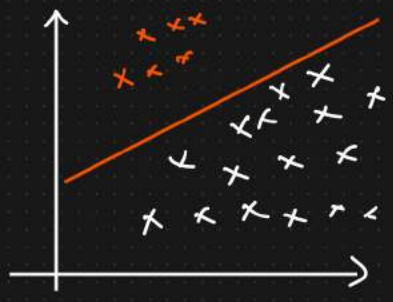
Predicted value → $\hat{y}$ (rows) / Actual (columns)

	1	0
1	TP	FP
0	FN	TN

Predicted

$$\text{Model Acc.} = \frac{TP + TN}{TP + FP + FN + TN}$$

$$= \frac{3+1}{3+2+1+1} = \frac{4}{7} = 0.571 \approx 57.1\%$$



② DATASET → Imbalanced dataset

1000 datapoints → 900 → 1, 100 → 0 ⇒ Imbalanced dataset

Dumb Model → o/p → 1 ⇒

Accuracy = 90%

Imbalanced Dataset –

In this scenario we cannot use Accuracy performance

② Precision = $\frac{TP}{TP+FP}$ } Out of all the actual values how many are correctly predicted

	1	0
1	TP	FP
0	FN	TN

FP → Important

FP ↓↓

③ Recall = $\frac{TP}{TP+FN}$ ⇒ FN ↓↓

⇒ Out of all the predicted values how many are correctly predicted with actual values

	1	0
1	TP	FP
0	FN	TN

Usecase 1 ⇒ Spam classification

Mail → (predict) → Spam / Not a Spam

	1	0
1	TP	FP
0	FN	TN

Mail → Spam
Model → Spam
} Good Scenario

FP ↓↓↓

0 ⇐ Mail → Not a Spam
1 ⇐ Model → Spam
} ⇐ Blunder

FP is Important

1 ⇐ Mail → Spam
0 ⇐ Model → Not a Spam
} ⇒ FN

PRECISION PERFORMANCE METRICS.

Usecase 2 ⇒ FN is Important

To predict whether a person has diabetes or not

	Diabetes	No Diabetes → Actual
Diabetes	TP	FP
No Diabetes	FN ↓↓	TN

① Actual → Diabetes
Model → Diabetes
} Good

② Actual → Diabetes
Model → No. Diabetes
} Blunder
⇒ FN ↓↓↓ ⇒ Important

③ Actual → No Diabetes
Model → Diabetes
} FP ⇒ Wrong Prediction

④ Actual → No Diabetes
Model → No Diabetes
} Correct

⇓

RECALL

Assignment : ① Tomorrow the stock will crash or not

Reduce FP ↓↓ or FN ↓↓

④ F-Beta Score $= (1+\beta^2) \frac{\text{Precison} * \text{Recall}}{\text{Precision} + \text{Recall}}$

① If FP & FN are both important

$\beta = 1$

Harmonic Mean

$$\text{F1 Score} = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

② If FP is more important than FN

$\beta = 0.5$

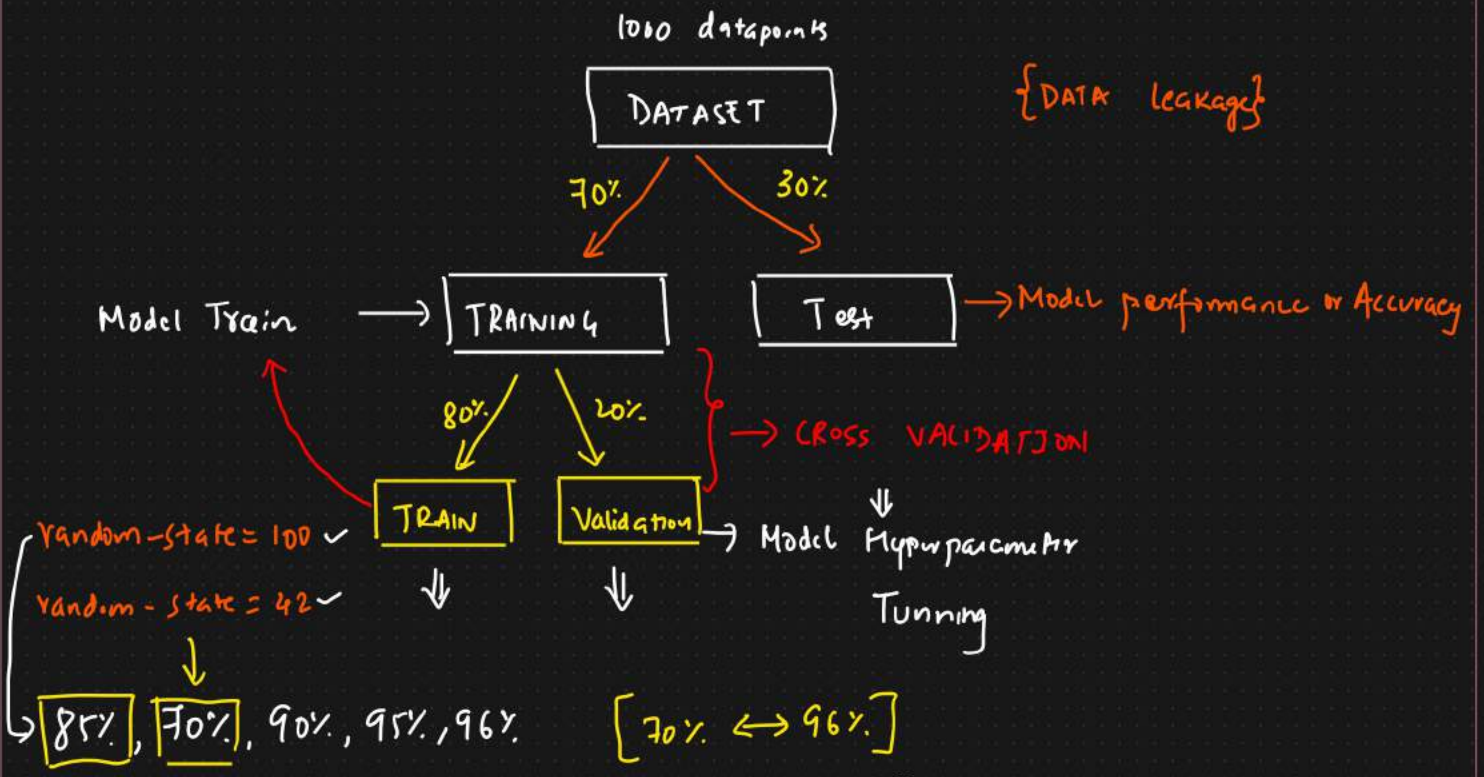
$$\text{F 0.5 Score} = (1+0.25) \frac{P*R}{P+R}$$

③ If FN >> FP

$\beta = 2$

$$\text{F2 Score} = (1+4) \frac{P*R}{P+R}$$

CROSS VALIDATION AND ITS TYPES



CROSS VALIDATION CV = 5

Exp1	TRAIN	VALIDATE	Acc1
Exp2	TRAIN	VALIDATE	Acc2
Exp3	TRAIN	VALIDATE	Acc3

[Acc1, Acc2, Acc3]

[80%, 85% 78%]

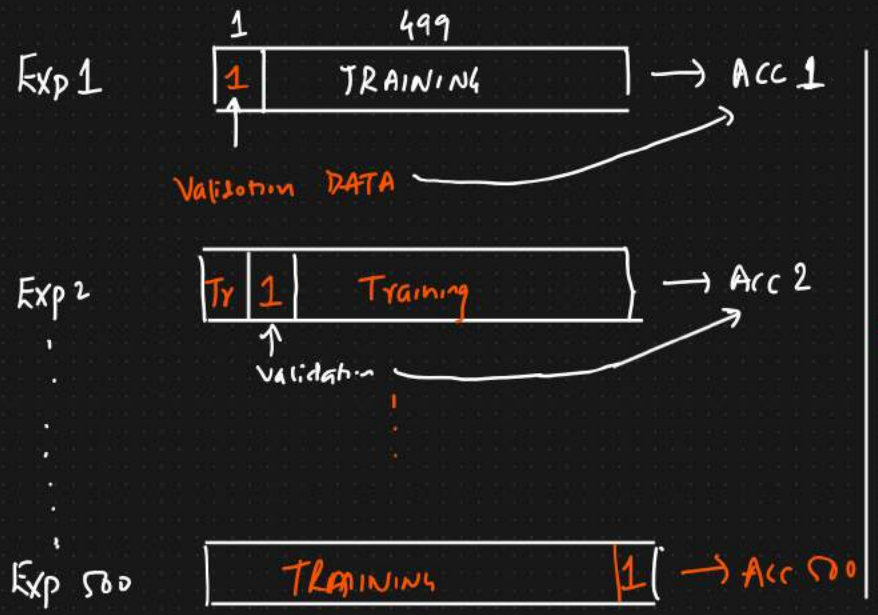
⇓

Final Accu = Mean of this Accuracy

Types of Cross Validation

① Leave One Out Cross Validation (LOO CV)

TRAINING DATA → 500 Records



TRAINING → Model TRAIN

VALIDATION → Model Predict

⇒ Mean of All these Accuracies

Disadvantage

① Time Complexity is huge for training Big datasets

② Model Overfit ⟶ TRAINING Acc ↑↑
New data ⟶ Validation Acc ↓↓

② Leave p out CV

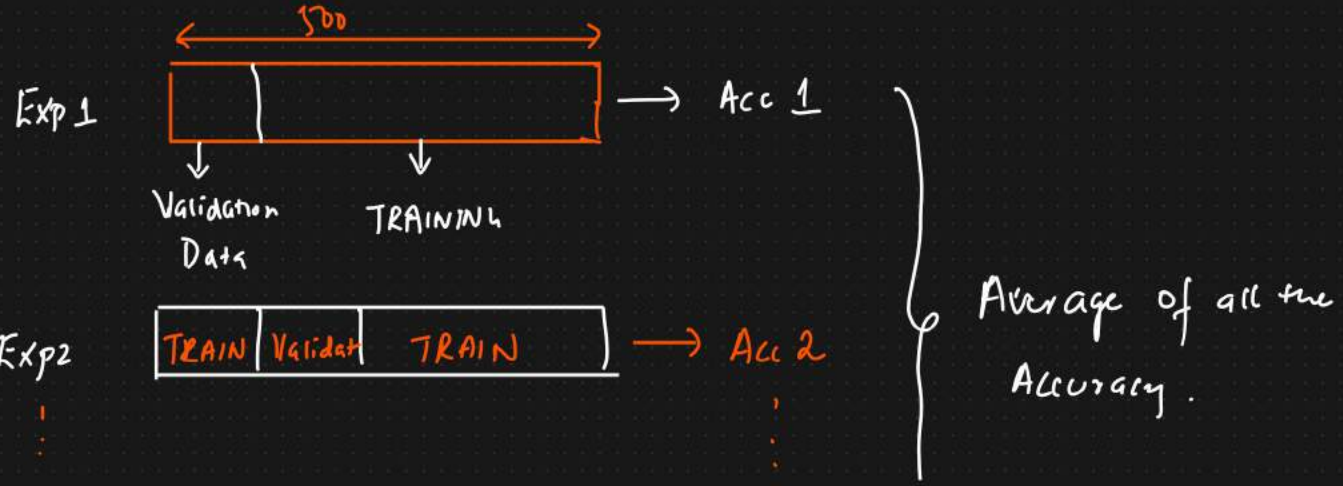
p = 10, 20, 30, 40 ⟶ Hyperparameter

③ K Fold Cross Validation

K = 5 n = 500

CV = 5

Validation size = $\frac{500}{5}$ = 100



Average of all the Accuracy.

Exp 5 | TRAIN | Validation | → Acc 5

f1 f2 y

4 — — 0
— — 0
0
0
— — 1
1
0
1

④ Stratified K Fold CV

{Imbalanced DATASET}.

{50's 1, 50's 0}

K=5 n=500 validation = 100

No of 0's & 1's should equal

n=500

{350 → 1, 150 → 0}

100

{50's 0, 50's 1}

Exp 1 | 100 | TRAINING DATA |

↓ Validation

Exp 2 | | 100 | |

↓ validation

⋮

⑤ Time Series CV

Amazon review Sentiment Analysis

Product A

JAN → DEC

TRAINING | VALIDATION

Day 1 Day 2 Day 3 Day 4 | - - - - - Day N

Time Series Application

Hyperparameter Tuning

↳ Finding the best parameters while training the model

① GridSearchCV

② RandomizedSearchCV

1) GridSearchCV [Grid Search + CROSS VALIDATION] [CV=5]

Logistic Regression

penalty{'l1', 'l2', 'elasticnet', None},

solver{'lbfgs', 'liblinear', 'newton-cg', 'newton-cholesky', 'sag', 'saga'}

CV ⇒ K fold CV

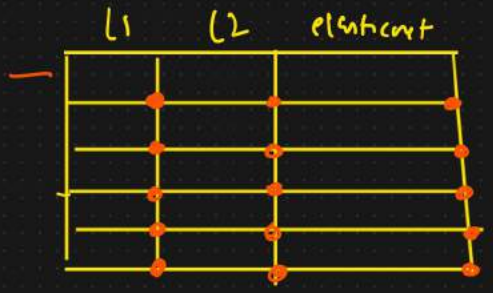
K=5

TRAIN AND VALIDATION = 5

Average Accuracy ⇒

model = LogisticRegression(elasticnet, newton-cg)

model.predict(new_data)



Increase the Model Performance or Accuracy

Disadvantage

① Time Complexity increases for Training the Model

② RandomizedSearchCV ←

n_iter = 10 + CV=5

10 different combination + CV=5

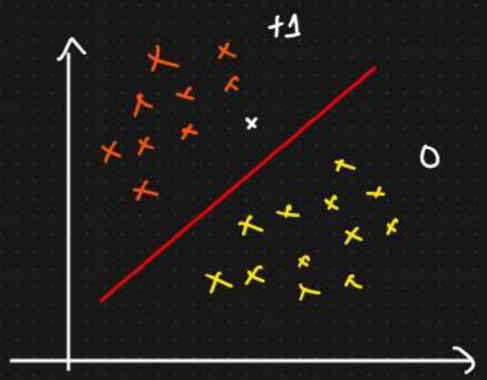
Advantage

① Time Complexity Decrease

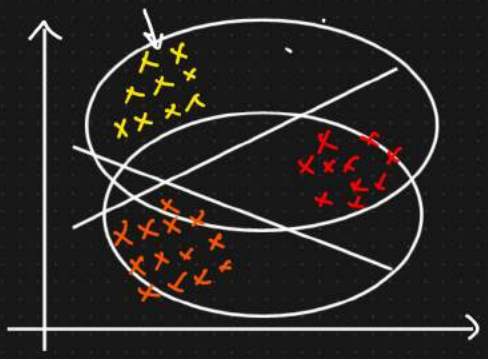
Logistic Regression For Multiclass Classification

① One Versus Rest

② Multinomial



f1	f2	f3	O/p
–	–	–	1
–	–	–	0
–	–	–	2
–	–	–	2
–	–	–	0
–	–	–	1



OVR

M1 ⇓ Binary classification

M2 ⇓ Logistic Binary Classifier

M3 ⇓ Binary Classification

f1	f2	f3	O/p	O1	O2	O3
–	–	–	1	0	1	0
–	–	–	0	1	0	0
–	–	–	2	0	0	1
–	–	–	2	0	0	1
–	–	–	0	1	0	0
–	–	–	1	0	1	0

I/p feature — O/p

M1 ⇒ f1, f2, f3 — O1 → Binary classification

M2 ⇒ f1, f2, f3 — O2 → Binary classification " "

M3 ⇒ f1 f2 f3 — O3 → " "

New Data point = [M1 M2 M3] [0.20, 0.35, 0.45] ⇒ 1

⇓ ⇓

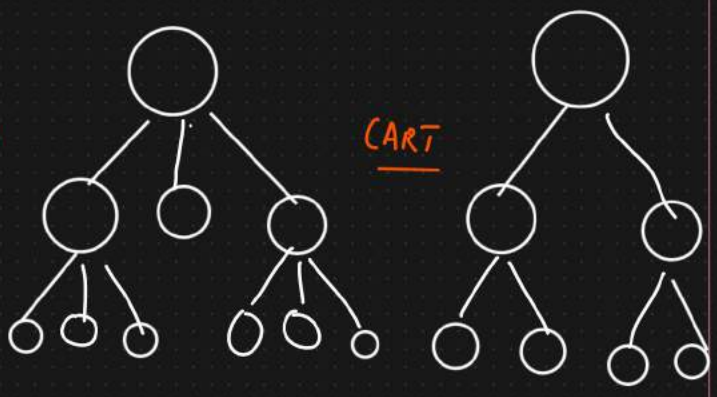
03 ← ap

Multinomial Probabilities.

Decision Tree

① Decision Tree classifier [classification]

② Decision Tree Regressor [Regression]



Decision Tree Classifier =>

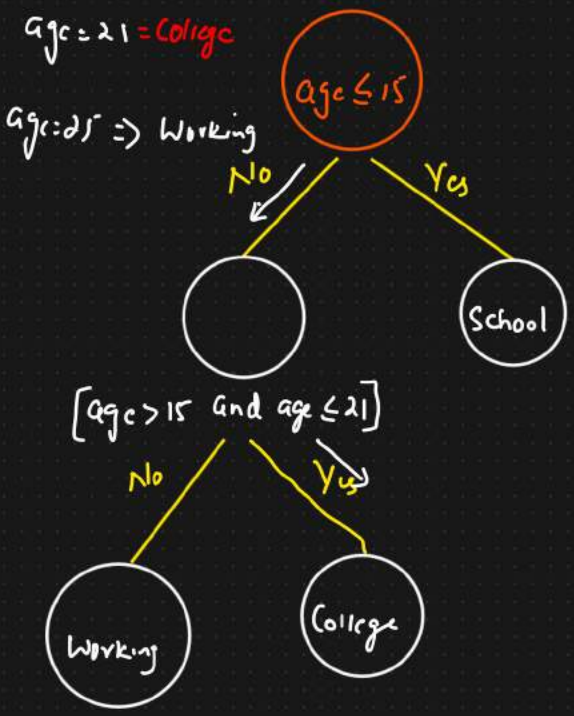
Two types

① ID3 [Iterative Dichotomiser 3]

② CART ✓ [Classification And Regression Tree]

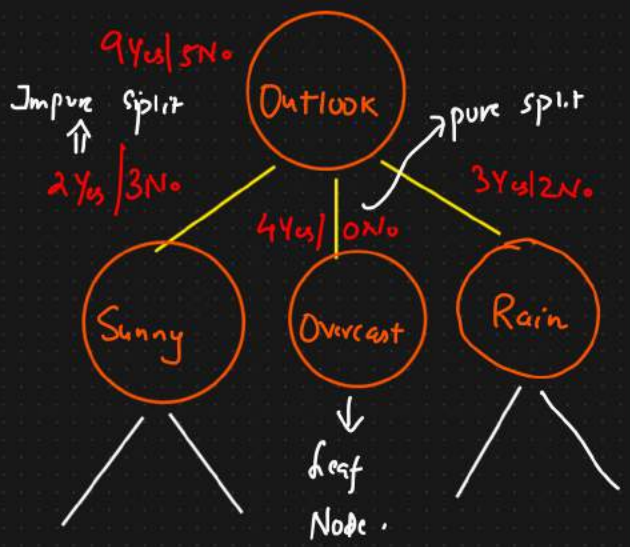
age = 14

```
if (age ≤ 15) :
    Print ("School").
clif (age > 15 and age ≤ 21):
    Print ("College")
else:
    Print ("Working")
```



Dataset → Predict Play Tennis OR Not

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No



① Purity check : Pure Split or Impure Split

- Entropy
- Gini Impurity

} Measure of purity. ✓

② What feature you need to select to start the split ? → Information Gain

Binary classification

① Entropy

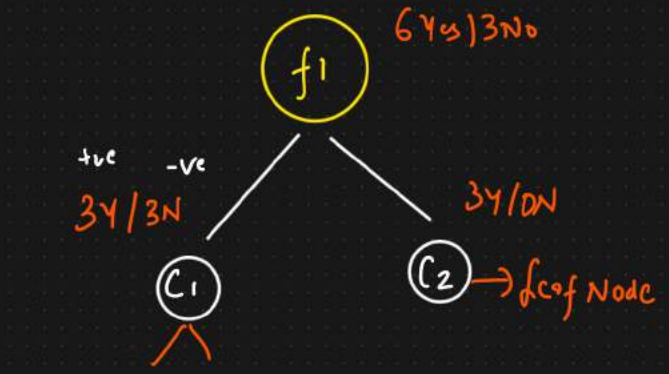
$$H(S) = -P_+ \log_2 P_+ - P_- \log_2 P_-$$

P_+ = probability of positive category

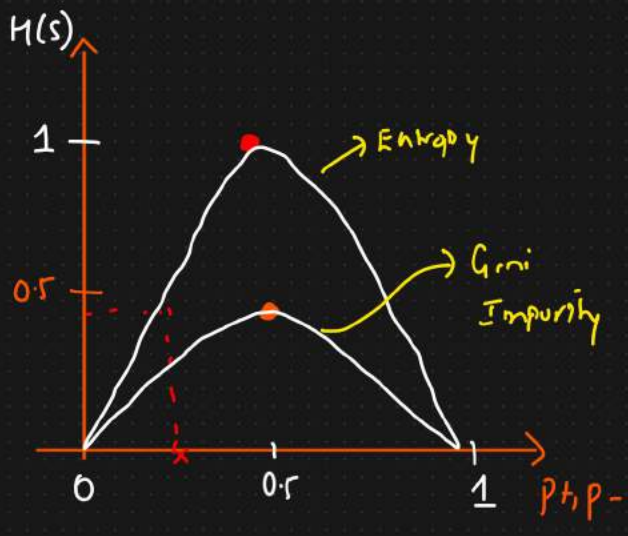
P_- = probability of negative category

② Gini Impurity

$$G.I = 1 - \sum_{i=1}^{n} (p)^2$$



$$H(C_1) = -P_+ \log_2 P_+ - P_- \log_2 P_-$$
$$= -\frac{3}{6} \log_2 \left(\frac{3}{6}\right) - \left(\frac{3}{6}\right) \log_2 \left(\frac{3}{6}\right)$$



$= -\frac{1}{2}\log_2(1/2) - (1/2)\log_2(1/2)$.

$= 1$ ⇒ Impure Split

$H(C_2) = -\frac{3}{3}\log_2(3/3) - 0/3\log_2(0/3)$

$= 0$ ⇒ Pure Split

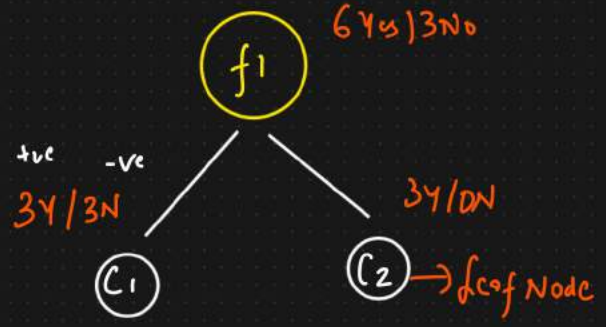
Multiclass

C_1 C_2 C_3
Yes | No | May Be

$H(S) = -P_{C_1}\log_2 P_{C_1} - P_{C_2}\log_2 P_{C_2} - P_{C_3}\log_2 C_3$

② Gini Impurity

$G.I = 1 - \sum_{i=1}^{n} (p)^2$



$= 1 - [(p_+)^2 + (p_-)^2]$

$= 1 - \left[\left(\frac{3}{6}\right)^2 + (3/6)^2\right]$

$= 1 - \left[\frac{1}{4} + \frac{1}{4}\right]$

$= 1/2 = 0.5$ ⇒ Impure Split

$1 - [(3/3)^2 + (0/3)^2]$

$= 1 - 1$

$= 0/1$ ⇒ Pure Split

② What feature you need to select to start the split? → Information Gain

f1 f2 f3 O/P

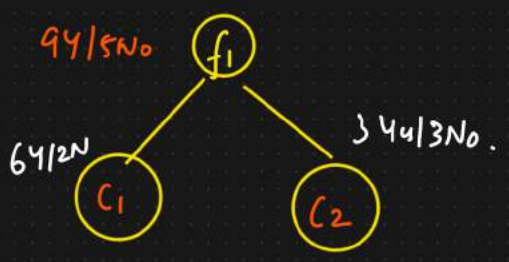
f1 f2

Information Gain

Entropy of the root node

$$Gain(S, f_i) = \boxed{H(S)} - \sum_{v \in val} \frac{|S_v|}{|S|} H(S_v).$$

f1	f2	f3	O/P



$$H(S) = -p_+ \log_2 p_+ - p_- \log_2 p_-$$

$$= -\frac{9}{14} \log_2 \frac{9}{14} - \frac{5}{14} \log_2 5/14$$

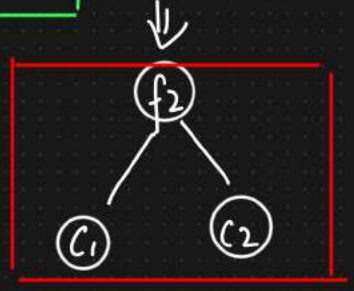
$$\approx 0.94 //$$

$$H(C_1) = -\frac{6}{8} \log_2 (6/8) - \frac{2}{8} \log_2 \frac{2}{8} \approx \boxed{0.81}$$

$$H(C_2) = \underline{1} //.$$

$$Gain(S, f_1) = 0.94 - \left[\frac{8}{14} * 0.81 + \frac{6}{14} * \underline{1}\right]$$

$$\boxed{Gain(S, f_1) = 0.049}$$



$$\underline{Gain(S, f_2) = 0.051} > \boxed{Gain(S, f_1) = 0.049}$$

Information gain is more when we split using f2.

Entropy Vs Gini Impurity

Whenever dataset is small → Entropy
dataset is huge → Gini Impurity

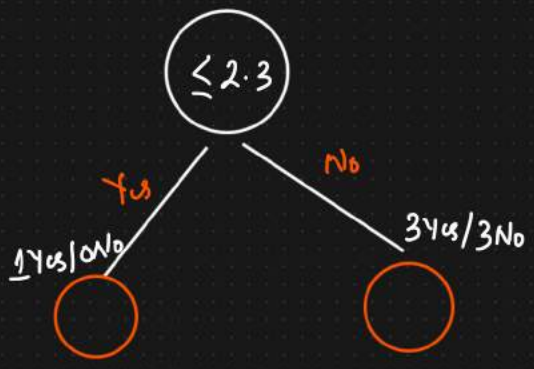
Decision Tree For Numerical Split

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No

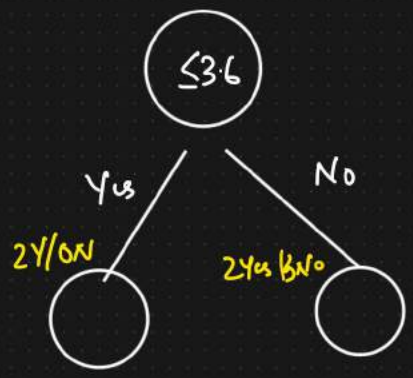
① Sorting the feature

f2	f1	O/P
-	2.3	Yes
-	3.6	Yes
-	4	No
-	5.2	No
-	6.7	Yes
-	7.8	No
-	9.0	Yes

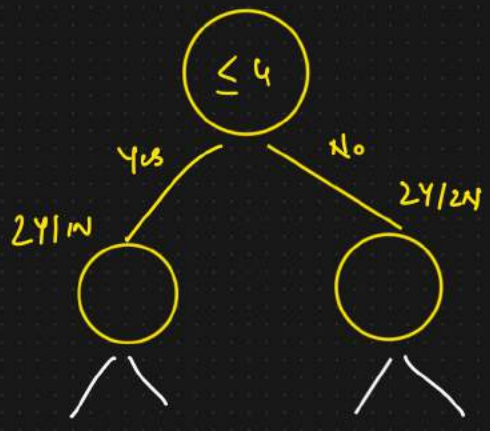
① Threshold = 2.3



② Threshold = 3.6



③ Threshold = 4

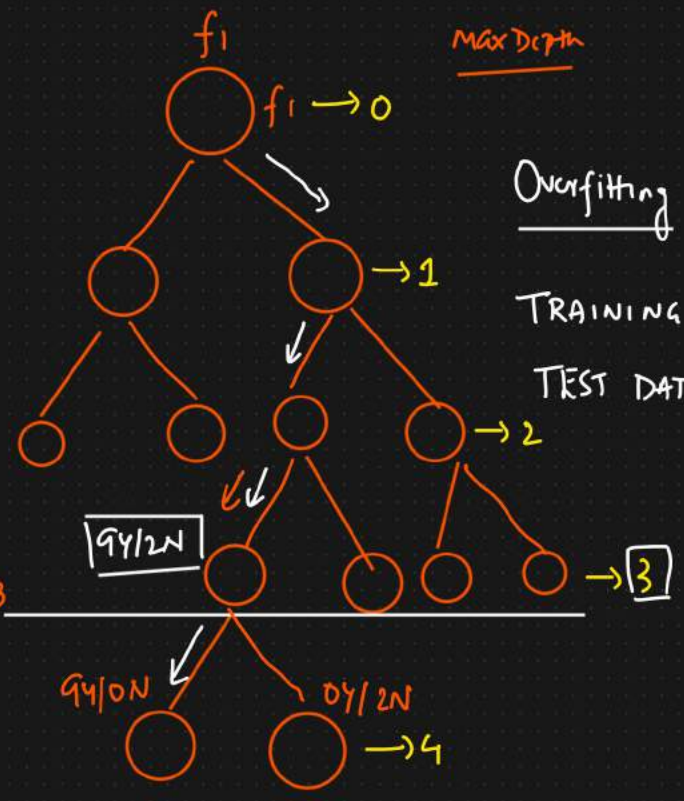


Millions of Records

Time Complexity ↑↑

Decision Tree Post Pruning And Pre Pruning [Reduce Overfitting]

Traning Data



Max Depth

① Post Pruning

② Pre Pruning

To Reduce Overfitting.

Overfitting

TRAINING ACC ↑↑

TEST DATA ACC ↓↓

Small Dataset Yes

① Post Pruning

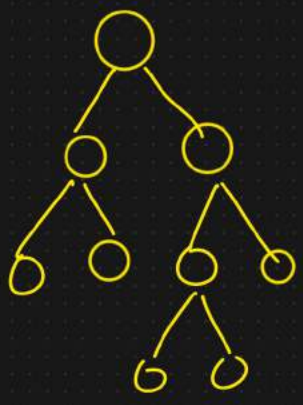
① Decision Tree Construct

② Prune the Tree

② Pre pruning = Hyperparameter Tuning

⇓

Large Dataset



max depth

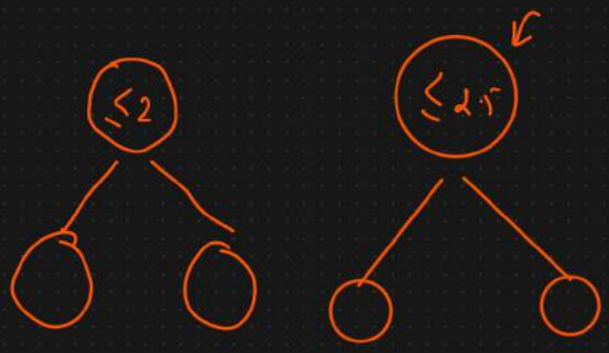
Criterion

Decision Tree Regressor

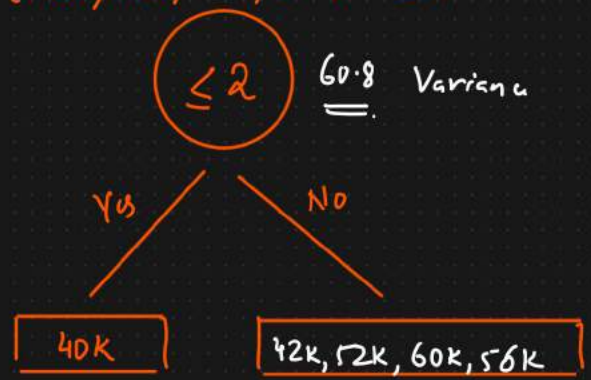
Dataset

Exp	Career Gap	Salary
→ 2	Yes	40K
→ 2.5	Yes	42K
3	No	52K
4	No	60K
4.5	Yes	56K
		50K = $\bar{y}$

QP



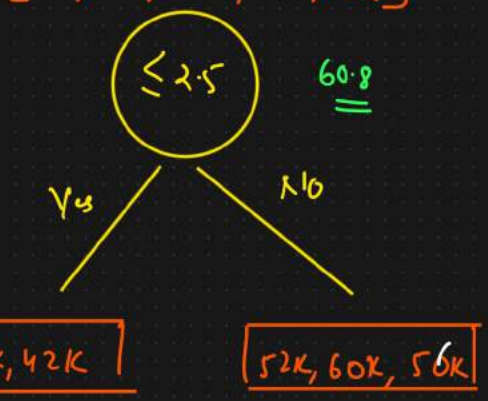
[40K, 42K, 52K, 60K, 50K]



100 (Variance)

Variance (51)

[40K, 42K, 52K, 60K, 50K]



Variance 82

Variance Reduction

$$\text{Variance} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y})^2$$ [Mean Squared Error].

$$\text{Variance (Root)} = \frac{1}{5}\left[(40-50)^2 + (42-50)^2 + (52-50)^2 + (60-50)^2 + (56-50)^2\right]$$

$$= \frac{1}{5}\left[100 + 64 + 4 + 100 + 36\right]$$

$$\text{Var(left)} = \frac{1}{n}\sum_{i=1}^{n}(y-\bar{y})^2$$

$$= \frac{1}{2}\left[(40-50)^2 + (42-50)^2\right]$$

$$= \frac{1}{2}\left[100 + 64\right]$$

$$= \frac{164}{2} = 82//.$$

$$\text{Var (Right)} = \frac{1}{3}\left[4 + 100 + 36\right]$$

$= 60.8$

$= \frac{140}{3} = 46.66$

$$\text{Variance (left)} = \frac{1}{n}\sum_{i=1}^{n}(y-\bar{y})^2$$

$$= \frac{1}{1}\left[(40-50)^2\right]$$

$$\text{Variance (left)} = 100$$

$$\text{Variance Reduction} = 60.9 - \left[\frac{2}{5} * 82 + \frac{3}{5} * 46.66\right]$$

$$= 0.004$$

$$\text{Variance (Right)} = \frac{1}{4}\left[(42-50)^2 + (52-50)^2 + (60-50)^2 + (56-50)^2\right]$$

$$= \frac{1}{4}\left[64 + 4 + 100 + 36\right]$$

$$= 51$$

$$Wi(L) = \frac{1}{5} \quad Wi(R) = 4/5$$

$$\text{Variance Reduction} = \text{Var(Root)} - \sum Wi\,\text{Var(child)}$$

$$= 60.8 - \left[\frac{1}{5} * 100 + \frac{4}{5} * 51\right]$$

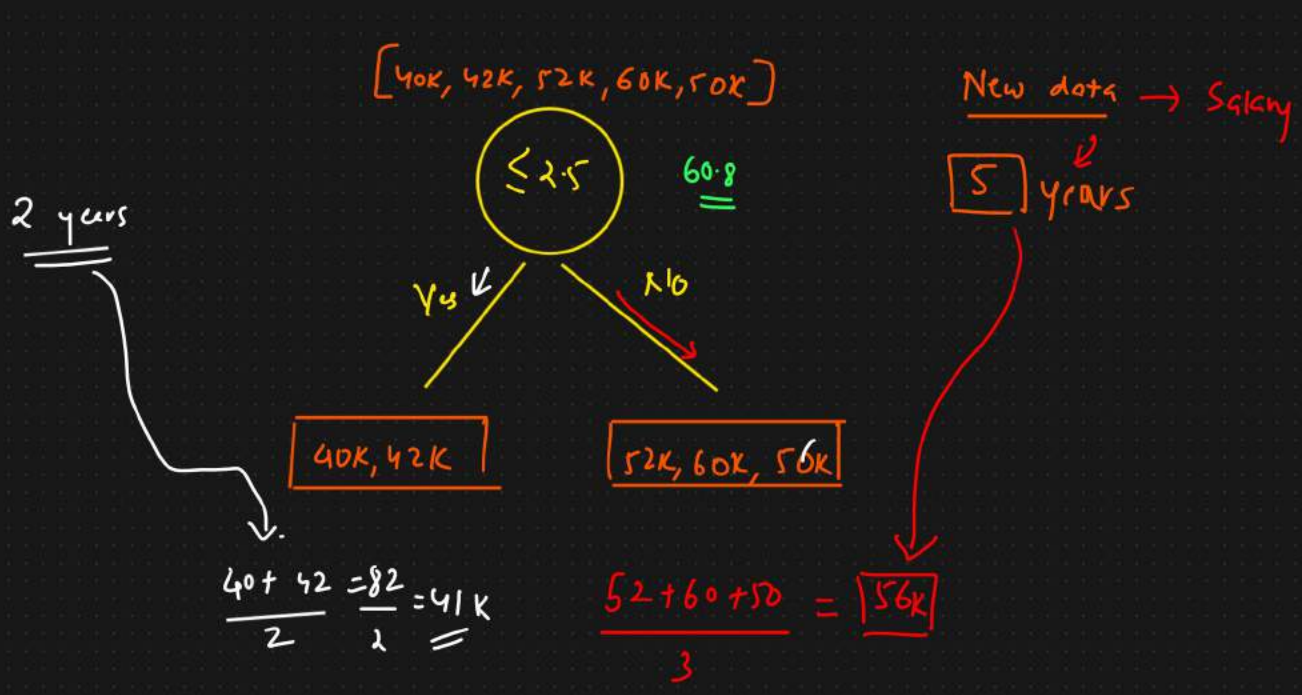
$$= 60.8 - 20 - 40.8$$

Variance Reduction	= 0

○

0.004

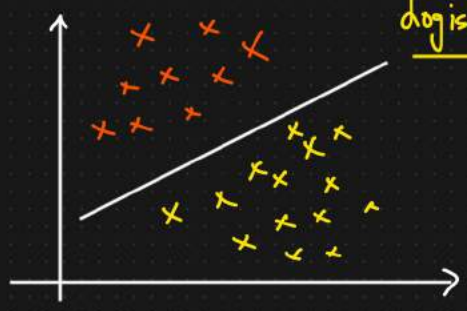
Variance Reduction (Left Split) < VR (Right Split)

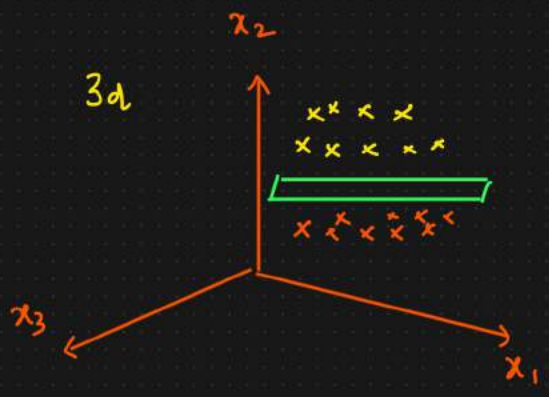

[40K, 42K, 52K, 60K, 50K]
≤ 2.5
60.8
Yes
No
2 years
New data → Salary
5 years
40K, 42K
52K, 60K, 56K
40 + 42 = 82 = 41K
2 2
52 + 60 + 50 = 56K
3

Support Vector Machines ML Algorithm

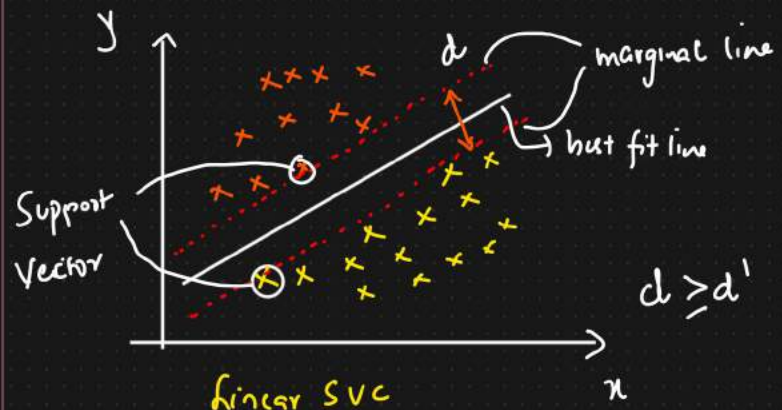
① SVC (Support Vector Classifier) ⟶ classification

② SVR (Support Vector Regressor) ⟶ Regression





① Support Vector Classifier (SVC)

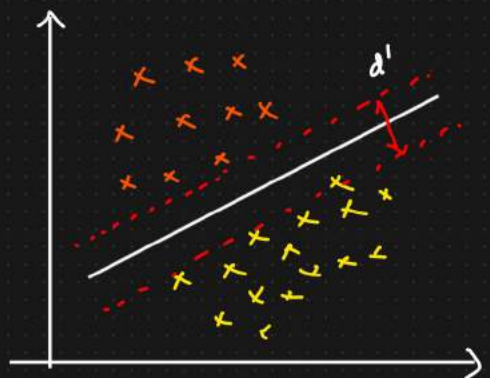


$d \geq d'$

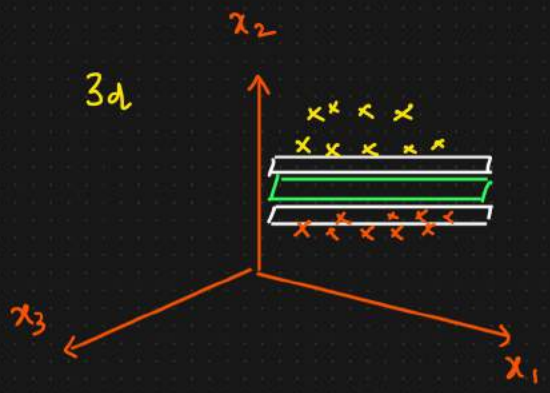
linear SVC

distance is maximum

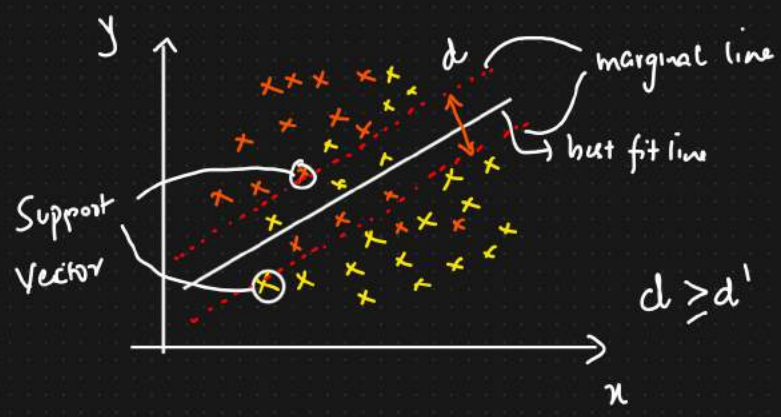
d = marginal plane distance



if $d' \geq d$



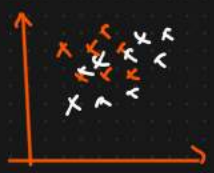
Soft Margin And Hard Margin In SVC



Hard Margin = None of the datapoints are misclassified

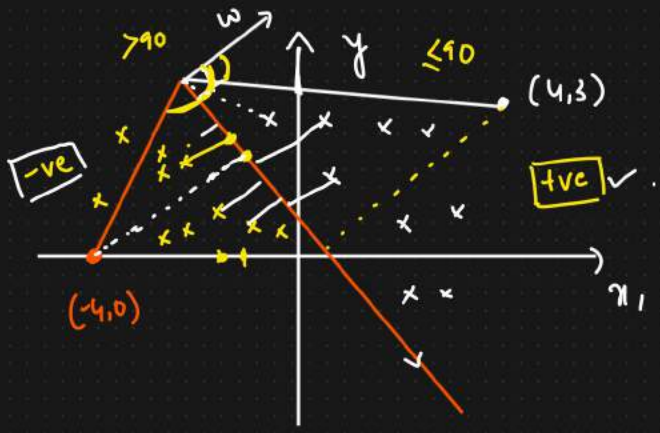
⇓

Impossible



② Soft Margin : Some data point are misclassified [Error]

③ Support Vector Machines (SVC) Maths Intuition



$w^T x + b = 0$

Equation of a straight line

$\hat{y} = mx + c \iff ax + by + c = 0$

$h_\theta(x) = \theta_0 + \theta_1 x_1$

$by = -ax - c$

$y = \frac{-a}{b} x - \frac{c}{b}$

$y = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3$

$y = m x + c$

$y = b + [w_1 x_1 + w_2 x_2 + w_3 x_3]$

$$w = \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} \quad x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

$$w^T = [w_1\ w_2\ w_3] \cdot x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

$$w^T x = [w_1 x_1 + w_2 x_2 + w_3 x_3]$$

$y = w^T x + b \Rightarrow y = mx + c$

$ax + by + c = 0$

$$w^T x + b = 0$$

Marginal plane in SVC

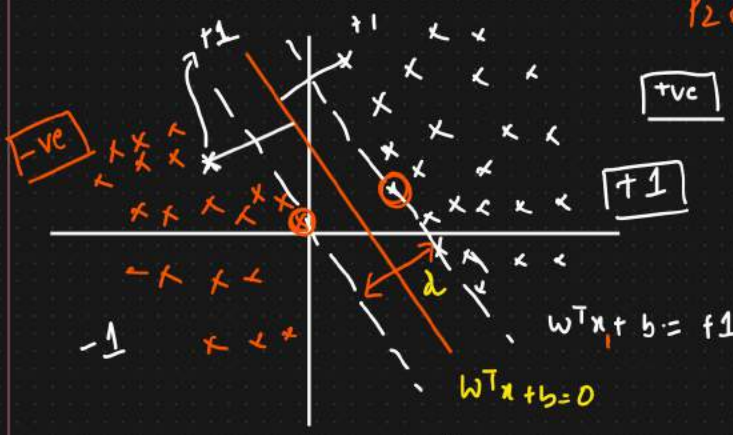
$P_1 \Leftarrow x_1 \Rightarrow (x_1, x_2)$

$P_2 \Leftarrow x_2 \Rightarrow (x_1, x_2)$

$w^T x_1 + b = +1$

$w^T x_2 + b = -1$

(-) (-) (+)



$\vec{w} \Leftarrow \frac{w^T (x_1 - x_2)}{\|w\|} = \frac{2}{\|w\|}$ ⇒ distance between Marginal plane.

⇓

Maximize

Cost function

Maximize $_{w,b}$ $\frac{2}{\|w\|}$ ⇒ Distance between Marginal plane

Constraint such that

$$y_i \begin{cases} +1 & \text{if } w^T x + b \geq 1 \\ -1 & \text{if } w^T x + b \leq -1 \end{cases}$$

⇓

For all correctly classified data points

$$y_i * (w^T x + b) \geq 1.$$

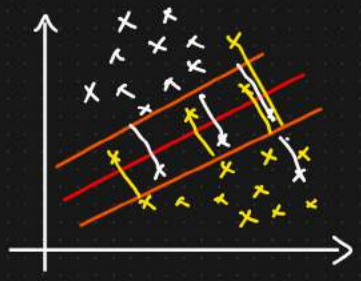
Modified Cost function of SVC

$$\underset{w,b}{\text{Maximize}} \quad \frac{2}{\|w\|} \Rightarrow \boxed{\underset{w,b}{\text{Minimize}} \quad \frac{\|w\|}{2}}$$

Constraint such that

$$y_i \begin{cases} +1 & \text{if } w^T x + b \geq 1 \\ -1 & \text{if } w^T x + b \leq -1 \end{cases}$$

$\boxed{C_i = 5}$



Cost function of Soft Margin SVC

$$\text{Cost fn} = \underset{w,b}{\text{Min}} \frac{\|w\|}{2} + \boxed{C_i \sum_{i=1}^{n} \xi_i} \Rightarrow \text{Hinge loss}$$

C_i: hyperparameter

ξ_i: Summation of the distance of incorrect data points to the marginal plane

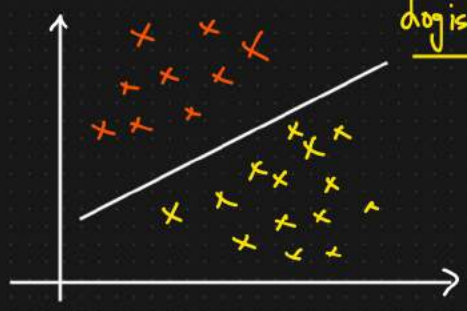
$\boxed{C = \frac{1}{\lambda}}$

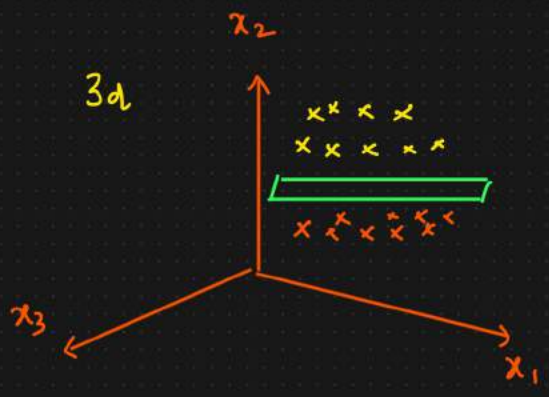
{How many points we can consider for misclassification}

Support Vector Machines ML Algorithm

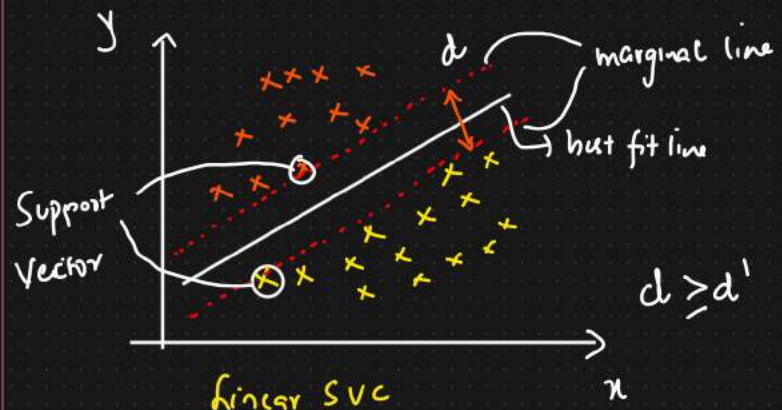
① SVC (Support Vector Classifier) ⟶ classification

② SVR (Support Vector Regressor) ⟶ Regression





① Support Vector Classifier (SVC)

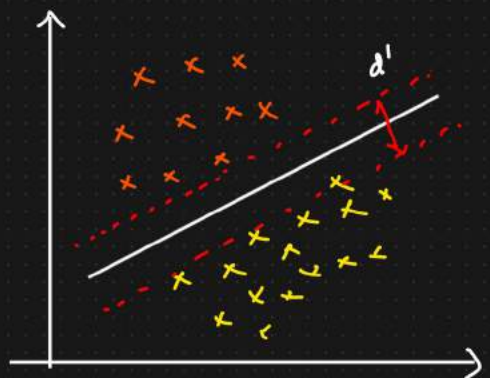


$d \geq d'$

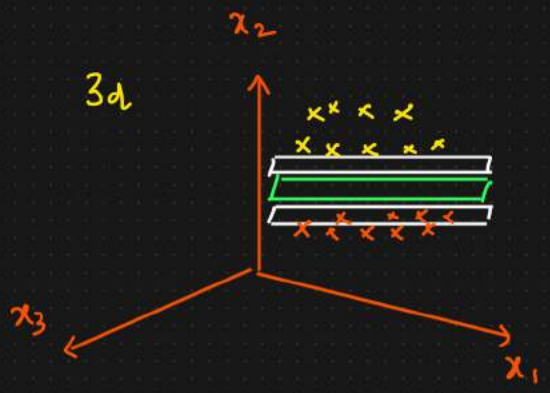
linear SVC

distance is maximum

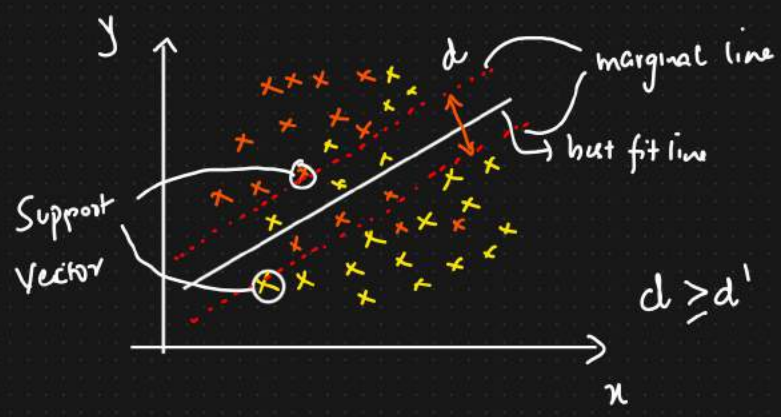
d = marginal plane distance



if $d' \geq d$



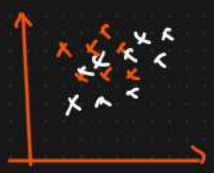
Soft Margin And Hard Margin In SVC



Hard Margin = None of the datapoints are misclassified

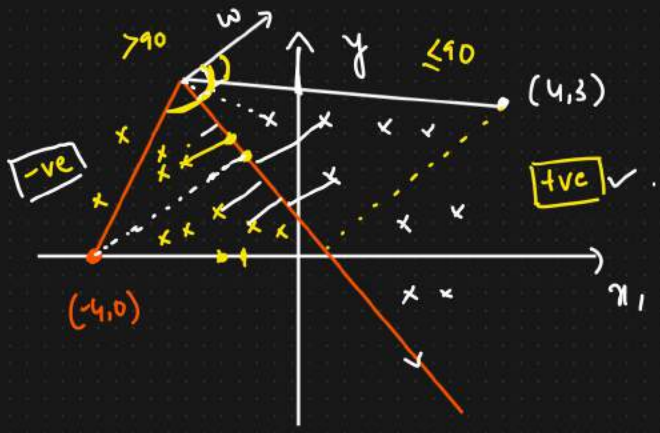
⇓

Impossible



② Soft Margin : Some data point are misclassified [Error]

③ Support Vector Machines (SVC) Maths Intuition



$w^T x + b = 0$

Equation of a straight line

$\hat{y} = mx + c \iff ax + by + c = 0$

$h_\theta(x) = \theta_0 + \theta_1 x_1$

$by = -ax - c$

$y = \frac{-a}{b} x - \frac{c}{b}$

$y = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3$

$y = m x + c$

$y = b + [w_1 x_1 + w_2 x_2 + w_3 x_3]$

$$w = \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} \quad x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

$$w^T = [w_1\ w_2\ w_3] \cdot x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

$$w^T x = [w_1 x_1 + w_2 x_2 + w_3 x_3]$$

$y = w^T x + b \Rightarrow y = mx + c$

$ax + by + c = 0$

$$w^T x + b = 0$$

Marginal plane in SVC

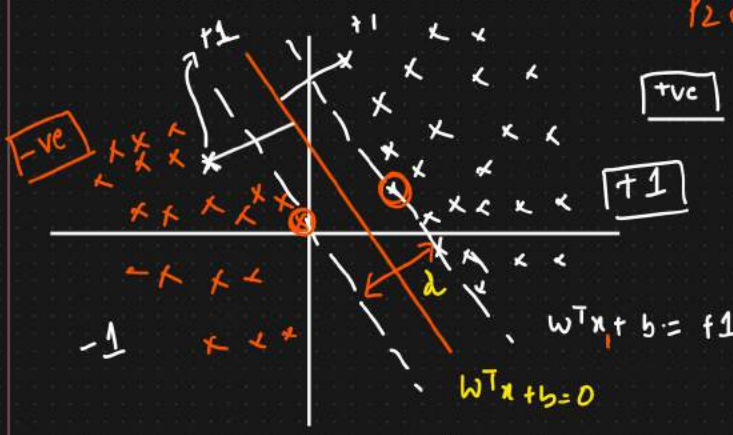
$P_1 \Leftarrow x_1 \Rightarrow (x_1, x_2)$

$P_2 \Leftarrow x_2 \Rightarrow (x_1, x_2)$

$w^T x_1 + b = +1$

$w^T x_2 + b = -1$

(-) (-) (+)



$\vec{w} \Leftarrow \frac{w^T (x_1 - x_2)}{\|w\|} = \frac{2}{\|w\|}$ ⇒ distance between Marginal plane.

⇓

Maximize

Cost function

Maximize $_{w,b}$ $\frac{2}{\|w\|}$ ⇒ Distance between Marginal plane

Constraint such that

$$y_i \begin{cases} +1 & \text{if } w^T x + b \geq 1 \\ -1 & \text{if } w^T x + b \leq -1 \end{cases}$$

⇓

For all correctly classified data points

$$y_i * (w^T x + b) \geq 1.$$

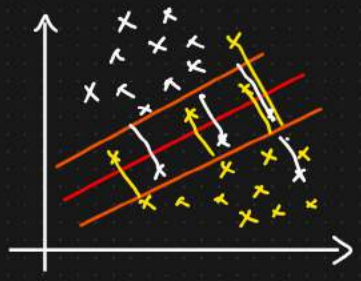
Modified Cost function of SVC

$$\underset{w,b}{\text{Maximize}} \quad \frac{2}{\|w\|} \Rightarrow \boxed{\underset{w,b}{\text{Minimize}} \quad \frac{\|w\|}{2}}$$

Constraint such that

$$y_i \begin{cases} +1 & \text{if } w^T x + b \geq 1 \\ -1 & \text{if } w^T x + b \leq -1 \end{cases}$$

$\boxed{C_i = 5}$



Cost function of Soft Margin SVC

$$\text{Cost fn} = \underset{w,b}{\text{Min}} \frac{\|w\|}{2} + \boxed{C_i \sum_{i=1}^{n} \xi_i} \Rightarrow \text{Hinge loss}$$

C_i → hyperparameter

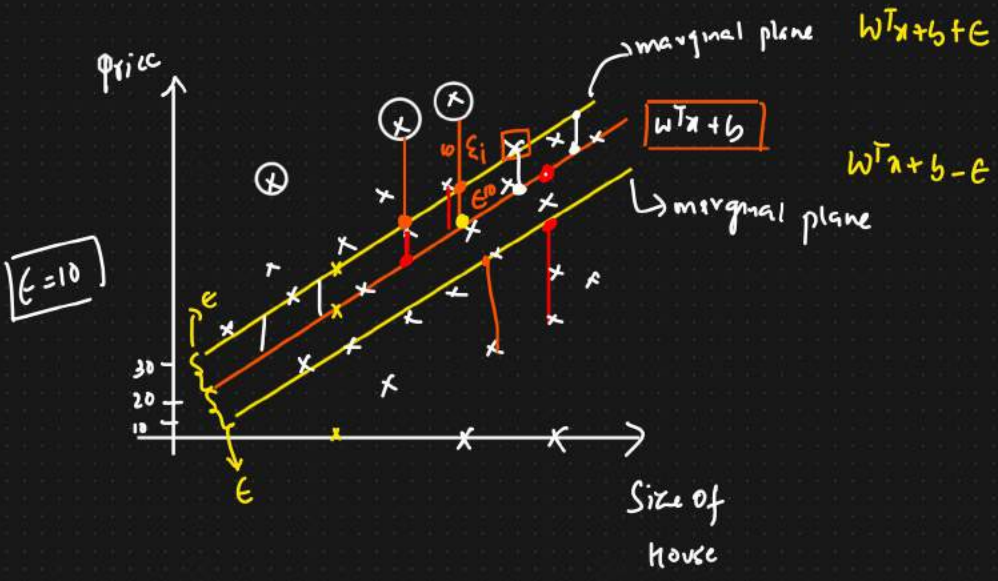
ξ_i → Summation of the distance of incorrect data points to the marginal plane

$\boxed{C = \frac{1}{\lambda}}$

{How many points we can consider for misclassification}

Support Vector Regressor (SVR)

ϵ = Marginal Error



$\xi_i = 10$, $\epsilon = 10$

≤ 20

Cost fn

$$\min_{w,b} \frac{||w||}{2} + \boxed{C \sum_{i=1}^{n} \xi_i} \Rightarrow \text{Hinge loss}$$

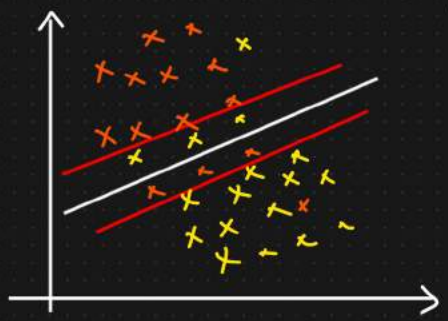
Constraint

$$|y_i - w^T x_i| \leq \epsilon + \xi_i$$

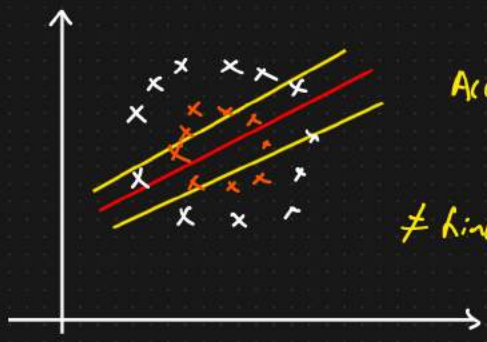
ϵ Marginal Error

ξ_i = Error above the Margin

SVM Kernels



linear SVC

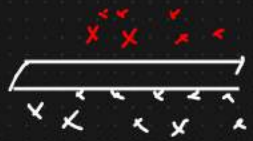


Accuracy = 50%

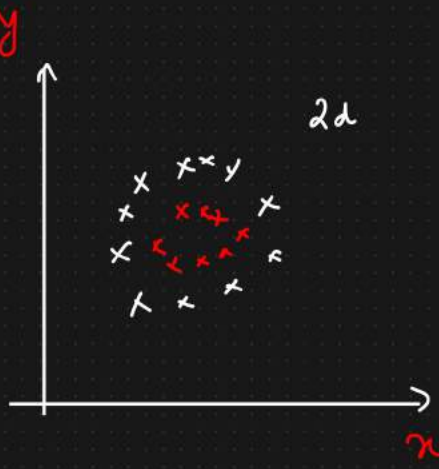
≠ linear SVC

⇓

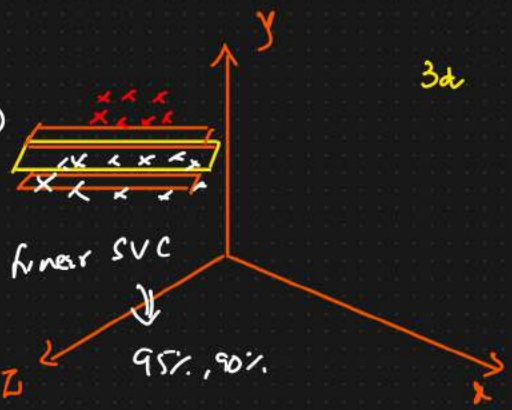
SVM Kernels



⇓



⇒ TRANSFORMATION ⇒



⇓

Mathematical Formula

$z = f(x, y)$

↳ Mathematical formula.

Dataset

X	y	$z = x^2$
2	Yes	4
3	No	9
4	Yes	16
–	Yes	–
–		–

1 dimension

⇒

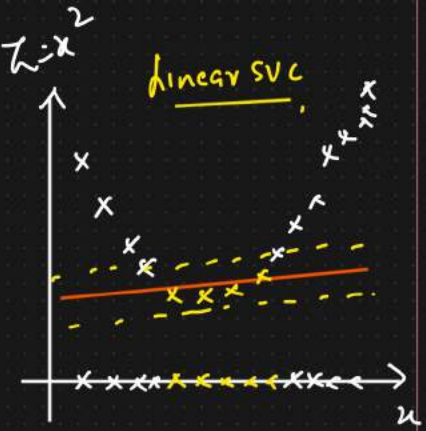
1d - 2d

SVM Kernel

⇓

Data Transformation

$z = x^2$

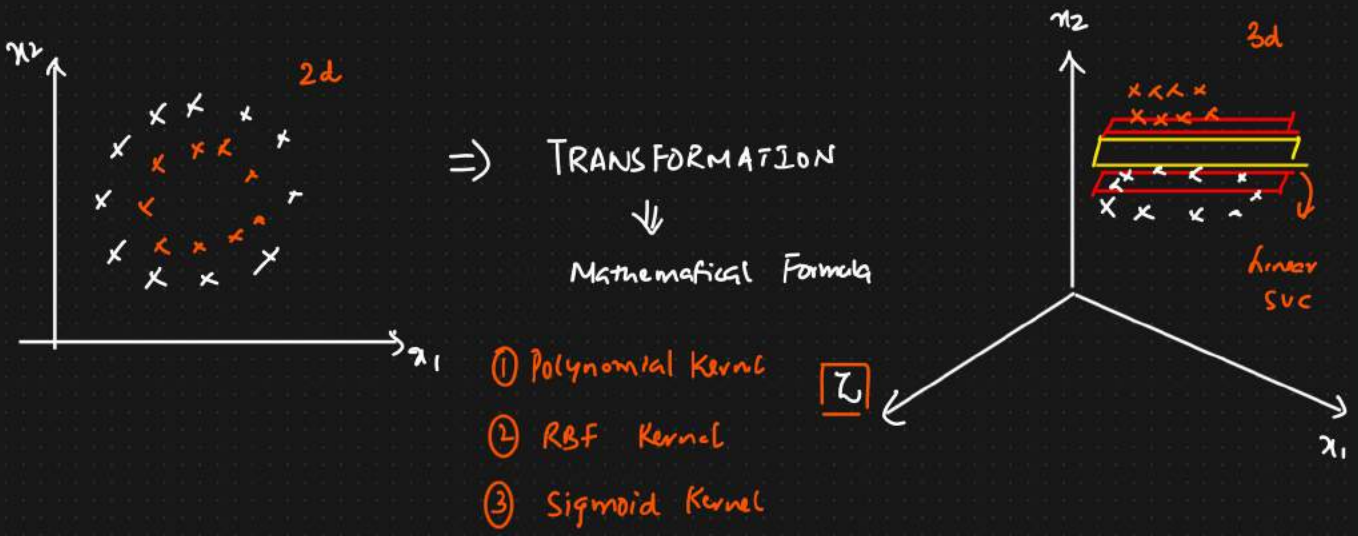


① Polynomical Kernel
② RBF Kernel
③ Sigmoid Kernel

⇒ Transformation ⇒ Mathematical Formula

SVM Kernels

1. Polynomial Kernel
2. RBF Kernel
3. Sigmoid Kernel



1. Polynomial Kernel

$$x = [x_1, x_2]$$

$$y = [x_1, x_2]$$

$$f(x,y) = (x^T y + c)^d \qquad c = 1$$

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \cdot [x_1, x_2]$$

2d

x_1 x_2 y $= \begin{bmatrix} x_1^2 & x_1 x_2 \\ x_1 x_2 & x_2^2 \end{bmatrix}$

⇓ Transformation

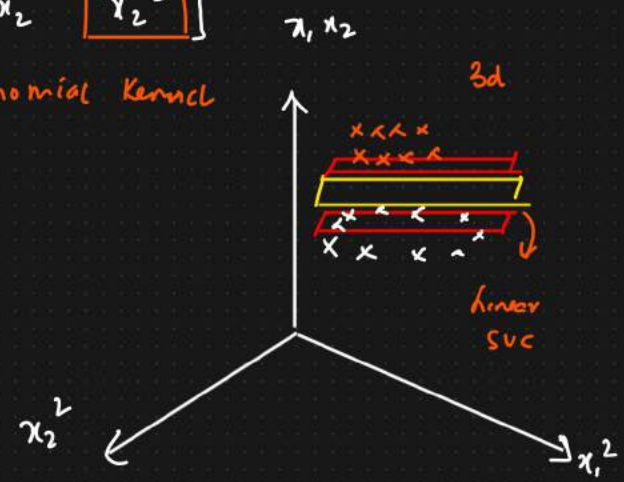
3d

x_1^2 $x_1 x_2$ x_2^2 y

\- \- \-

\- \-

⇒ Polynomial Kernel



② RBF Kernel

↓

Formula

③ Sigmoid Kernel

↓

Formula

Naive Bayes Algorithm (Classification)

① Probability [Independent And Dependent Events]

② Bayes theorem

③ Naive Bayes Math Intuition.

① Probability

Independent Events

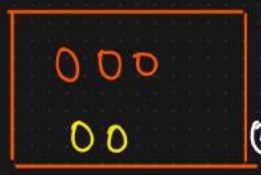
Rolling a Dice {1,2,3,4,5,6}

$Pr(1) = \frac{1}{6}$ $Pr(2) = \frac{1}{6}$ $Pr(3) = \frac{1}{6}$.

= =

Dependent Events

① What is the probability of first removing a orange marble and then a yellow marble?



① → P(O) = 3/5 → 1st Event

② → P(Y) = 2/4 → 2nd Event

⇓

$\boxed{P(Y/O) = 2/4}$ => Conditional Probability

$Pr(O \text{ and } Y) = P(O) * \boxed{P(Y/O)}$ => Conditional Probability

$= 3/5 * 2/4 = \boxed{\frac{3}{10}}$

⇓

$$Pr(A \text{ and } B) = Pr(A) * P(B/A)$$

Bayes Theorem

$$Pr(A \text{ and } B) = Pr(B \text{ and } A)$$

$$Pr(A) * Pr(B/A) = Pr(B) * Pr(A/B)$$

Bayes Theorem

$$Pr(A/B) = \frac{Pr(A) * Pr(B/A)}{Pr(B)}$$

P(A/B) = Probability of Event A given B has occured

P(A) = Probability of Event A

P(B) = Probability of Event B

P(B/A) = Probability of Event B given A has occured.

DATASET

x_1	x_2	x_3	↓ Predict O/P ⇒ Y
–	–	–	Yes
–	–	–	No
–	–	–	Yes
–	–	–	No
–	–	–	Yes

$$Pr(Y/(x_1, x_2, x_3)) = \frac{Pr(Y) * P(x_1, x_2, x_3/Y)}{Pr(x_1, x_2, x_3)}$$

(A = Y, B = (x_1, x_2, x_3))

$$Pr(A/B) = \frac{Pr(A) * Pr(B/A)}{Pr(B)}$$

$$Pr(Y/(x_1, x_2, x_3)) = \frac{Pr(Y) * P(x_1, x_2, x_3/Y)}{Pr(x_1, x_2, x_3)}$$

$$= \frac{Pr(Y) * Pr(x_1/Y) * Pr(x_2/Y) * Pr(x_3/Y)}{Pr(x_1) * Pr(x_2) * Pr(x_3)}$$

DATASET

x_1	x_2	x_3	↓ Pred. O/P ⇒
–	–	–	Yes
–	–	–	No
–	–	–	Yes
–	–	–	No
–	–	–	Yes

$$Pr(y=Yes/(x_1, x_2, x_3)) = \frac{Pr(Yes) * Pr(x_1/Yes) * P(x_2/Yes) * Pr(x_3/Yes)}{\cancel{Pr(x_1) * Pr(x_2) * P(x_3)}} = 0.60 \Rightarrow 1$$

Remove

$$Pr(y=No/(x_1, x_2, x_3)) = \frac{Pr(No) * Pr(x_1/No) * Pr(x_2/No) * Pr(x_3/No)}{\cancel{Pr(x_1) * Pr(x_2) * Pr(3)}} = 0.40$$

Lets Solve This Problem

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Outlook

	Yes	No	P(E/Yes)	P(E/No)
Sunny	2	3	2/9	3/5
Overcast	4	0	4/9	0/5
Rain	3	2	3/9	2/5

(Sunny, Hot) = O/P

Temperature

	Yes	No	P(E/Yes)	P(E/No)
Hot	2	2	2/9	2/5
Mild	4	2	4/9	2/5
Cool	3	1	3/9	1/5

Yes/No

Yes	9	P(Y) = 9/14
No	5	Pr(No) = 5/14

$$Pr(Yes/(Sunny, Hot)) = \frac{P(Yes) * Pr(Sunny/Yes) * Pr(Hot/Yes)}{\cancel{Pr(Sunny) * Pr(Hot)}}$$

$$= 9/14 * 2/9 * 2/9$$

$$= \frac{2}{63} = \boxed{0.031}$$ //.

$$Pr(No/(Sunny, Hot)) = \frac{Pr(No) * Pr(Sunny/No) * Pr(Hot/No)}{\text{Constant}}$$

$$= 5/14 * 3/5 * 2/5$$

$= 0.085$

Finally

$$Pr(Yes|(Sunny, hot) = \frac{0.031}{0.031 + 0.085} = 0.27 = 27\%$$

$$Pr(No|(Sunny, hot) = \frac{0.085}{0.031 + 0.085} = 0.73 = 73\%$$

NEW DATA [Sunny, Hot] $\Rightarrow$ 73% $\Rightarrow$ NO $\Rightarrow$ 0

27% $\Rightarrow$ Yes

$\Downarrow$

Person will Not play

Variants of Naive Baye's

1. Bernoulli Naive Bayes
2. Multinomial Naive Bayes
3. Gaussian Naive Bayes

1. Bernoulli Naive Bayes

Whenever your features are following a Bernoulli Distribution, then we use Bernoulli Naive Baye's

Bernoulli ⟶ 0, 1

$P(\text{Success}) = 1 = p$

$P(\text{Fail}) = 0 = 1 - p$

Dataset:

f1	f2	f3	O/P
Yes	Pass	Male	Yes
No	Fail	Female	No
Yes	Pass	Male	Yes
No	Pass	Female	No
Yes	Pass		

2. Multinomial Naive Bayes ⇒ I/P = Text

Dataset: Sentiment Analysis

I/P

Review Message	O/P
The product is really good	Positive
The product is bad	Negative

⇓

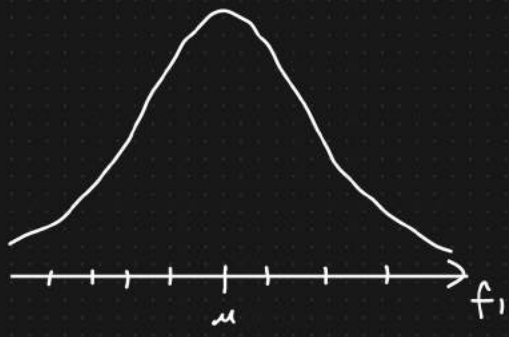
Numerical Values => Natural Language Processing

1. BOW
2. Tf-IDF
3. Word2Vec

3. Gaussian Naive Bayes

If the features are following Gaussian Distribution then we use Gaussian Naive Baye's Algorithm to solve Classification problem

IRIS Dataset



[Continuous features]

Age	Height	Weight	Yes/No
25	170	78	
28	160	75	
32	170		
34	140		

Ensemble Technique

① What is Ensemble?

Combining Multiple Models ⇒ TRAIN ⇒ Predictions

Two Types

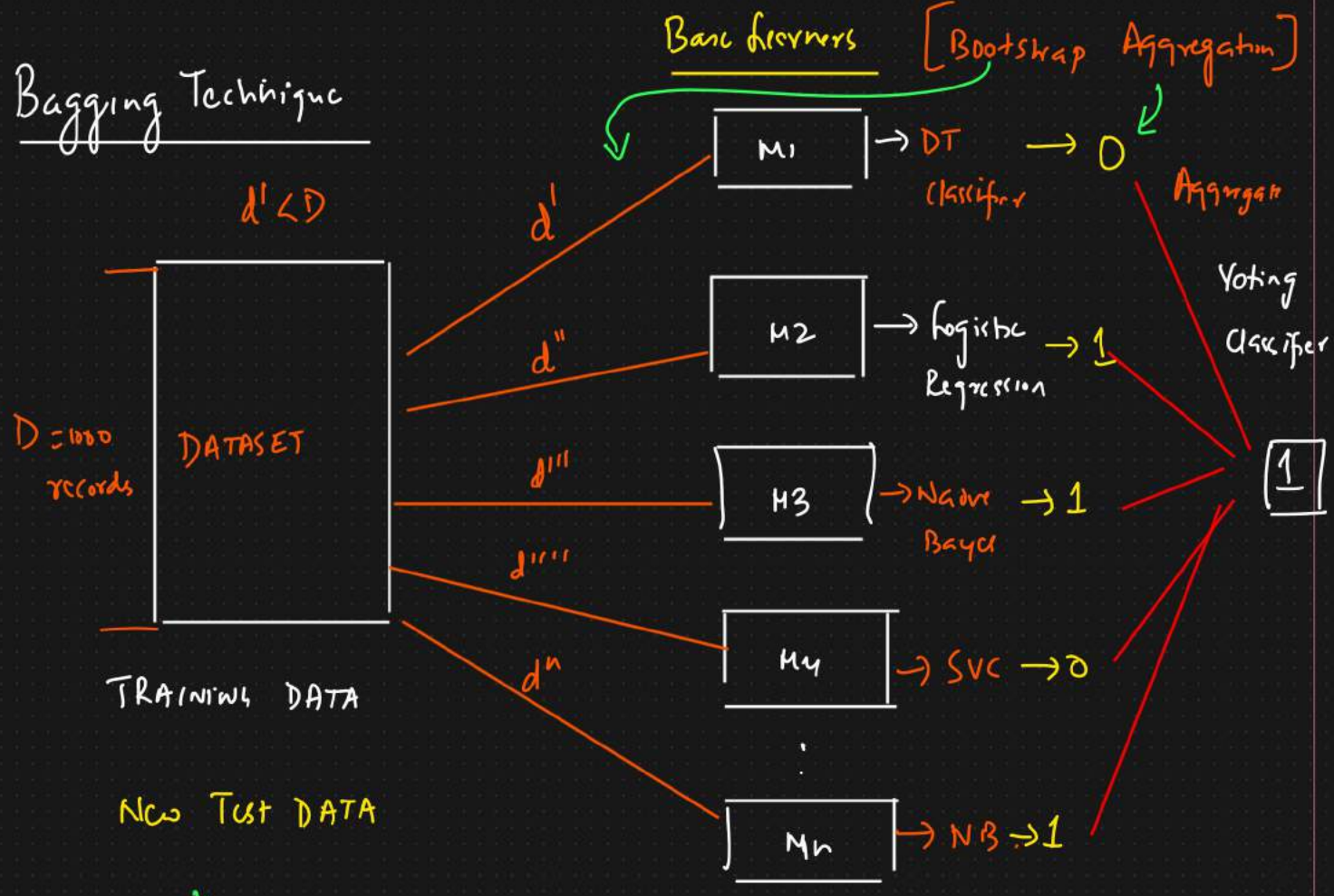
① Bagging ↙

① Random Forest Classifier And Regressor

② Boosting.

① AdaBoost
② GRADIENT BOOST
③ Xgboost

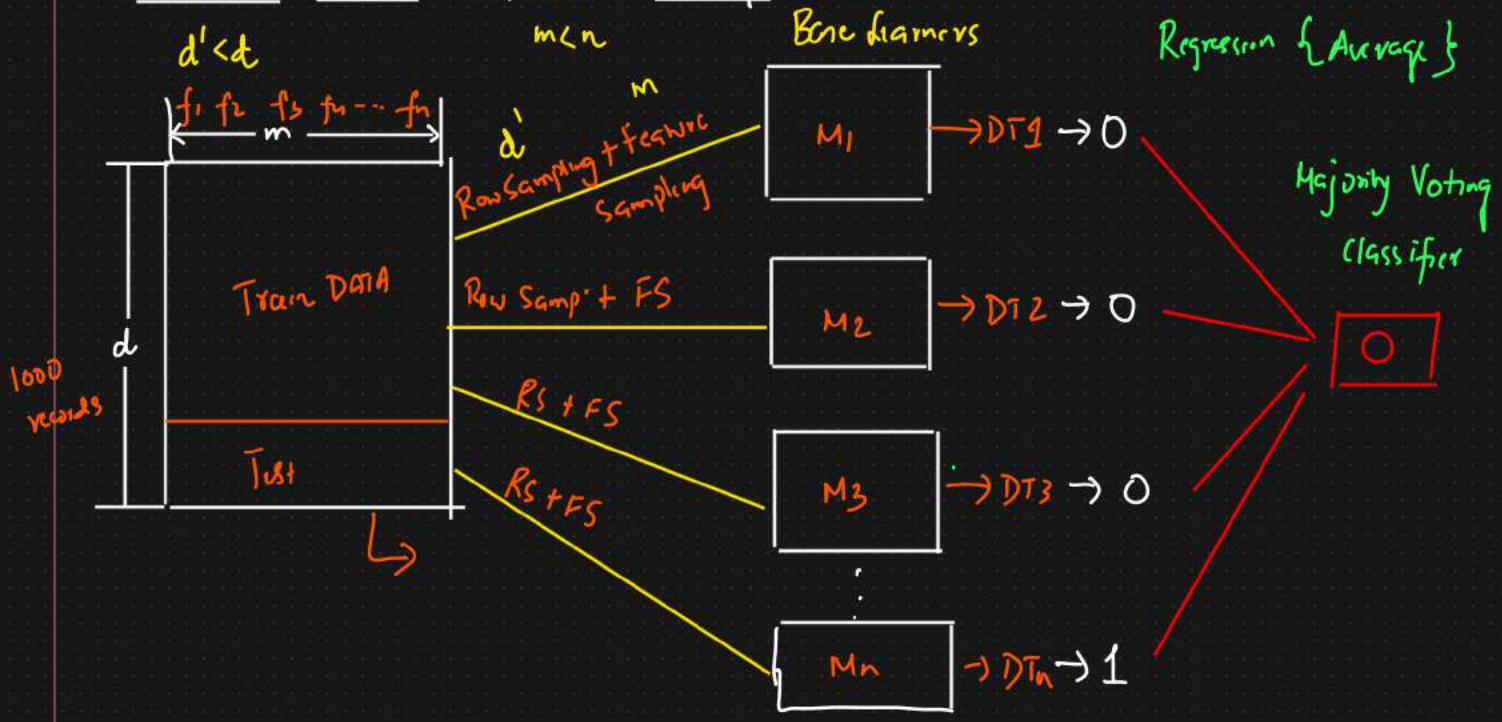
Bagging Technique



Custom Bagging Technique

Regression → o/p → Average

Random Forest Classifier And Regressor

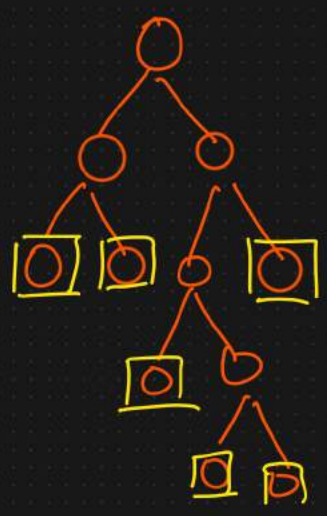


Note :

Classification → Majority Voting Classifier
Regression → Average O/p of the Model
} New Data

⑧ Why should we use Random Forest instead of DT?

Decision Tree

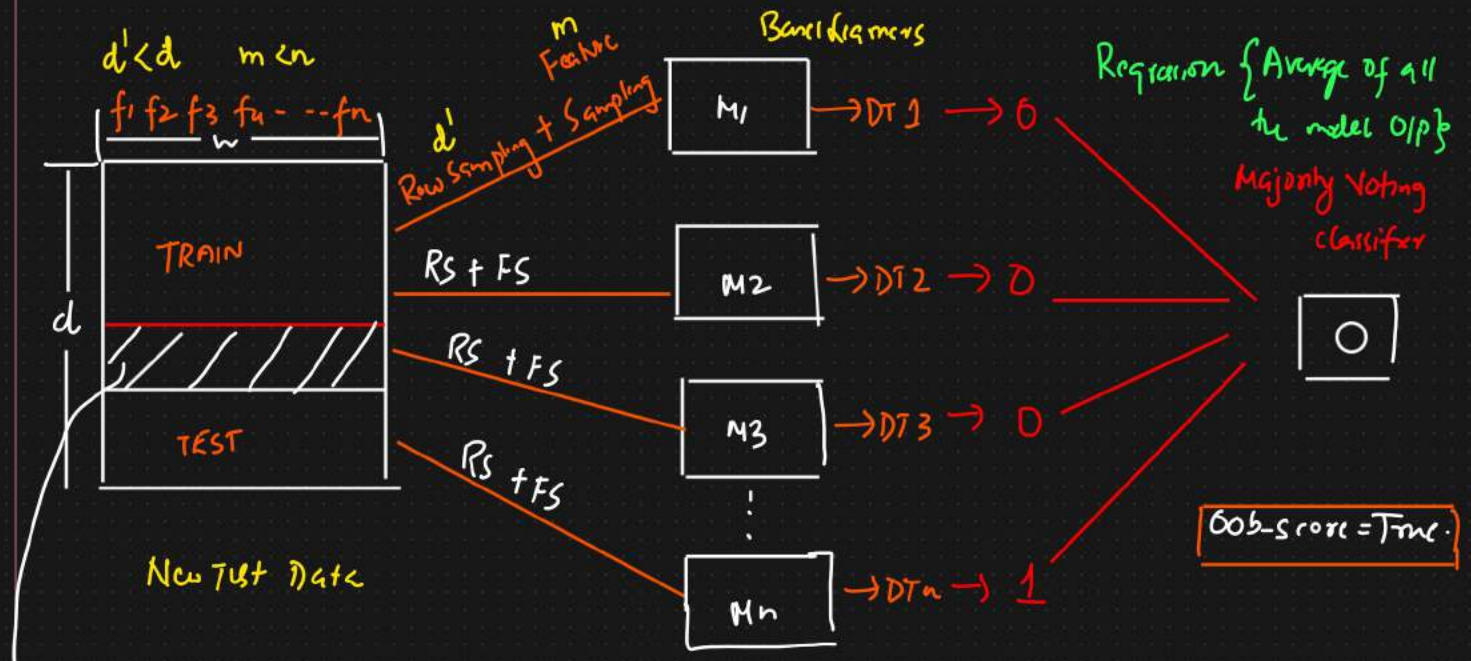


Overfitting

Generalized Model
{Random Forest}

TRAINING ACC ↑↑ → Low Bias → Low Bias
TEST DATA ↓↓ → High Variance → Low Variance

Out of Bag Score



→ Out of Bag DATA := Validation data :=) validation accuracy

⇓

OOB-Score

Boosting Algorithms

① Adaboost

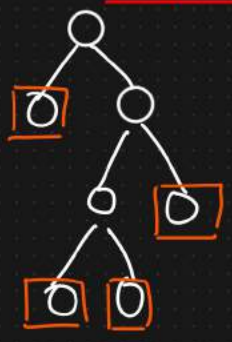
② Gradient Boost

③ Xgboost.

Boosting

Decision Tree

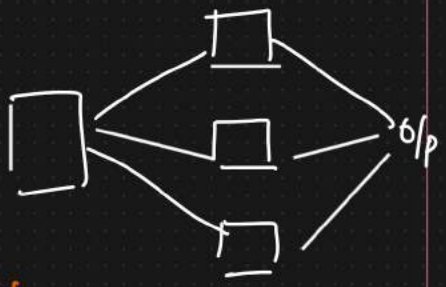
TRAINING DATA



Overfitting :

TRAIN ACC ↑↑

TEST ACC ↓↓

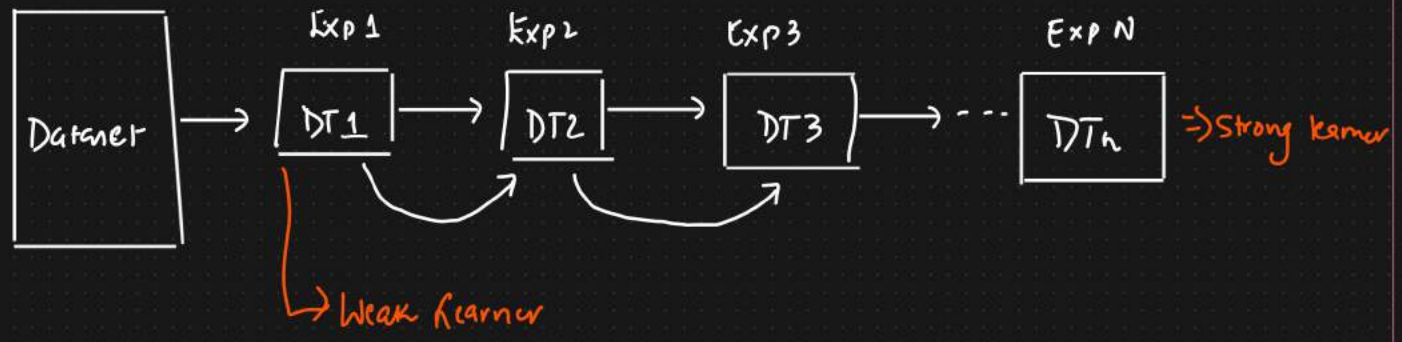


Generalized Model

Low Bias
High Variance
} ⇒ Random Forest ⇒ {
Low Bias
Low Variance

Boosting { Decision Trees Sequentially Connected }

KBC



Weak learners : Haven't learnt much from the Training Dataset

Bagging → Random Forest → Majority Voting [Average of all o/p]

Boosting → Assignments of weights to the weak learners

Boosting

$$f = \alpha_1(M_1) + \alpha_2(M_2) + \alpha_3(M_3) + \cdots + \alpha_n(M_n)$$

→ Classification

→ Regression

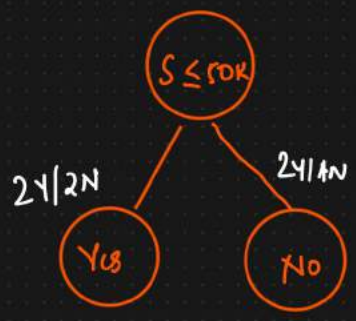
$\{\alpha_1, \alpha_2, \alpha_3 \cdots \alpha_n\} \Rightarrow$ weights

Adaboost Classifier → Boosting Technique

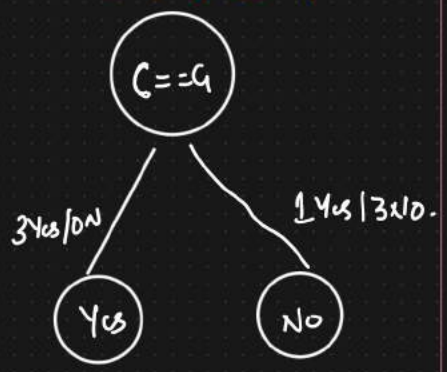
Dataset

Salary	Credit	Approval	SW
<=50K	B	No	1/7
<=50K	G	Yes	1/7
<=50K	G	Yes	1/7
>50K	B	No	1/7
>50K	G	Yes	1/7
>50K	N	Yes	1/7
<=50K	N	No	1/7

① We create Decision Tree Stump and we select the best stump



Best DT Stump.



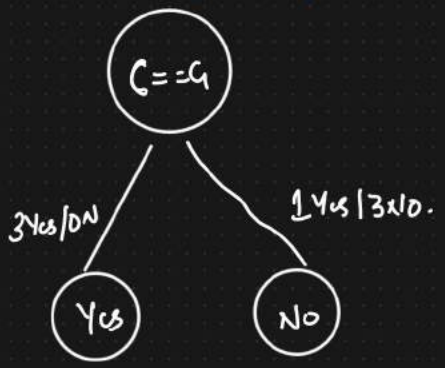
Entropy or Gini and Information gain

③ Sum of Total Errors And performance of stump

Dataset

Salary	Credit	Approval	SW
<=50K	B	No	1/7
<=50K	G	Yes	1/7
<=50K	G	Yes	1/7
>50K	B	No	1/7
>50K	G	Yes	1/7
>50K	N	Yes	1/7
<=50K	N	No	1/7

Best DT Stump.



Sum of TE = $\frac{1}{7}$.

② Performance of Stump = $\frac{1}{2} \ln\left[\frac{1-TE}{TE}\right]$

$= \frac{1}{2} \ln\left[\frac{1-1/7}{1/7}\right]$

$= \frac{1}{2} \ln[6] \approx 0.896$

Performance of Stump = 0.896

$f = \alpha_1(M_1) + \alpha_2(M_2) + \cdots + \alpha_n(M_n)$

$\alpha_1 = 0.896$ ⇒ Weight.

④ Update the weights for correctly and Incorrectly classified points.

Salary	Credit	Approval	SW	Updated weights
<=50K	B	No	$\frac{1}{7}$ ↓	0.058
<=50K	G	Yes	$1/7$ ↓	0.058
<=50K	G	Yes	$\frac{1}{7}$ ↓	0.058
>50K	B	No	$1/7$ ↓	0.058
>50K	G	Yes	$1/7$ ↓	0.058
>50K	N	Yes	$1/7$ ↑↑	0.349
<=50K	N	No	$1/7$ ↓	0.058

For correctly Classified points

$= \text{weight} * e^{-\text{Performance of stump}}$

$= \frac{1}{7} * e^{-(0.896)}$

$= 0.058$

For Incorrect Classified points

$= \text{weight} * e^{\text{Performance of stump}}$

$= \frac{1}{7} * e^{(0.896)}$

$= 0.349$

⑤ Normalized Weights Computation And Assigning Bins

0.45 0.17

Salary	Credit	Approval	SW	Updated weights	Normalized Wt	Bin Assignment
<=50K	B	No	$\frac{1}{7}$ ↓	0.058	0.083	0 – 0.08
<=50K	G	Yes	$1/7$ ↓	0.058	0.083	0.08 – 0.16 ← 0.095
<=50K	G	Yes	$\frac{1}{7}$ ↓	0.058	0.083	0.16 – 0.24
>50K	B	No	$1/7$ ↓	0.058	0.083	0.24 – 0.32
>50K	G	Yes	$1/7$ ↓	0.058	0.083	0.32 – 0.40
>50K	N	Yes	$1/7$ ↑↑	0.349	0.500	→ 0.40 – 0.90
<=50K	N	No	$1/7$ ↓	0.058	0.083	0.90 – 0.98
				0.697	1	

⑥ Select data points to the Next Stump

Salary	Credit	Approval	Bin Assignment
<=50K	B	No	0 - 0.08
<=50K	G	Yes	0.08 - 0.16 ← 0
<=50K	G	Yes	0.16 - 0.24
>50K	B	No	0.24 - 0.32
>50K	G	Yes	0.32 - 0.40
>50K	N	Yes	0.40 - 0.90
<=50K	N	No	0.90 - 0.98

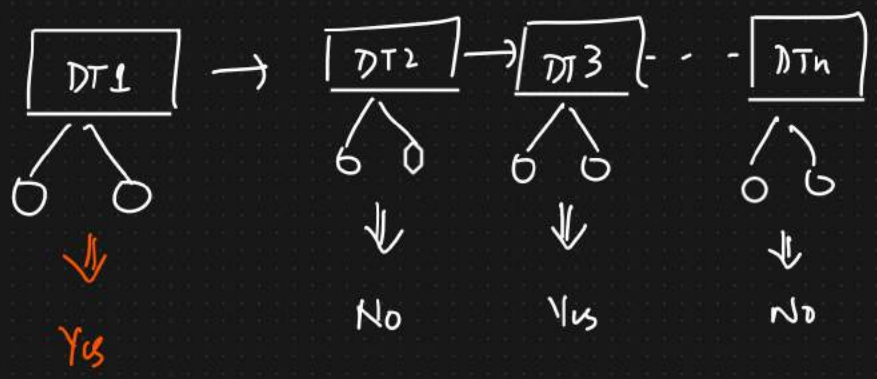
⑤ Iteration process (picking random values between 0 and 1)

S	Credit	Approval	Random
>50K	N	Yes	0.50
<=50K	G	Yes	0.10
>50K	N	Yes	0.60
>50K	N	Yes	0.75
<=50K	G	Yes	0.24
>50K	B	No	0.32
>50K	N	Yes	0.87

⑦ Final Prediction

Test data (≤50K, G) ←

f: Test data →



$\alpha_1 = 0.896$ $\alpha_2 = 0.650$ $\alpha_3 = 0.24$ $\alpha_n = -0.30$

$f = 0.896\,(\text{Yes}) + 0.650\,(\text{No}) + 0.24\,(\text{Yes}) - 0.30\,(\text{No})$

$= 1.136\,(\text{Yes}) + 0.350\,(\text{No})$ ⇒ O/P ⇒ Yes

Performance of say(Yes) = 1.136 >>

Performance of say(No) = 0.350

Gradient Boosting Algorithms

① Regression

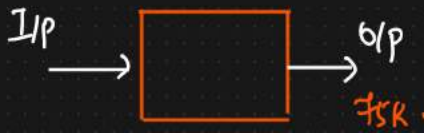
② classification

Dataset

Exp (x_1)	Degree (x_2)	Salary (y)	$(y-\hat{y})$ R_1	Predicted R_2	$\hat{y}$	R_3
→ 2	B.E	50K	-25K	-23K	47.77	-24.77
→ 3	Masters	70K	-5K	-3K	4.97	-4.97
5	Masters	80K	5K	3	-	-
6	PHD	100K	25K	20K	-	-
		75K				

Steps

① Create a Base Model

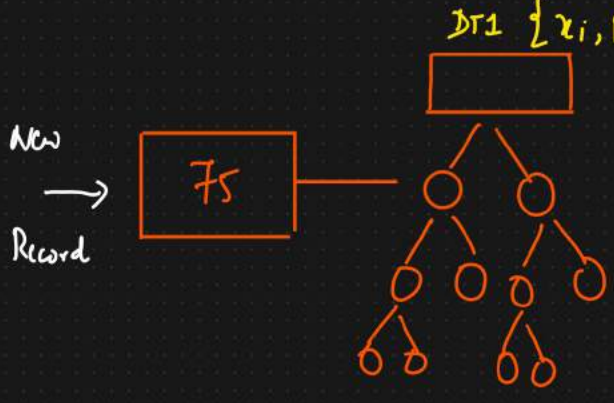


$$\text{Average} = \frac{[50K + 70K + 80K + 100K]}{4}$$

$$= 75K$$

② Compute Residuals, Error

③ Construct a Decision Tree consider inputs x_i and o/p R_1



Base Model

Predicted o/p : 75 + (-23) = 75 - 23 = 52 {overfitting}.

Predicted o/p : $75 + \alpha(-23)$

α = Learning Rate

$\alpha = 0.01 \rightarrow \{0 \text{ to } 1\}$

$= 75 + (0.01)(-23)$

$= 75 - 0.23$

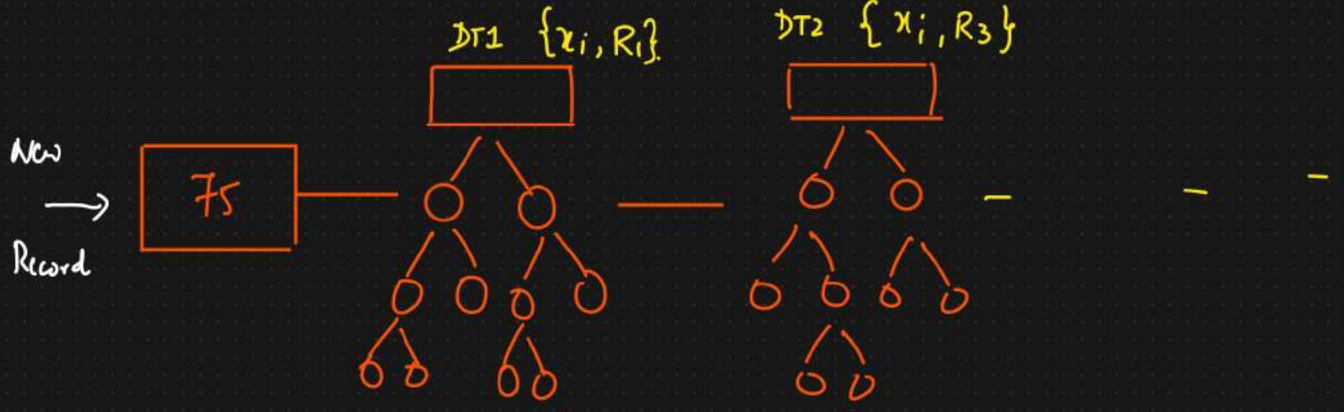
$= 74.77$

② Predicted o/p = $75 + \alpha(-3\alpha)$

$= 75 + (0.01)(-3)$

$= 75 - 0.03$

$= 74.97$ //



Base Model

o/p = $75 + \alpha_1(DT_1) + \alpha_2(DT_2) + \cdots \alpha_n(DT_n)$.

Mathematical

$\alpha_0 = 1$

$F(x) = \alpha_0 h_0(x) + \alpha_1 h_1(x) + \alpha_2 h_2(x) + \cdots + \alpha_n h_n(x)$

$\{\alpha_0, \alpha_1, \alpha_2, \cdots \alpha_n\} \rightarrow$ Learning Rate $\{0 \text{ to } 1\}$

$$F(x) = \sum_{i=0}^{n} \alpha_i h_i(x)$$

Xgboost ML Algorithm Classification

Dataset

x_1 Salary	x_2 Credit	y Approval	Error $(y-0.5)$ R_1
<=50K	B	0	-0.5
<=50K	G	1	0.5
<=50K	G	1	0.5
>50K	B	0	-0.5
>50K	G	1	0.5
>50K	N	1	0.5
<=50K	N	0	-0.5

Steps

1. Construct a base Model
2. Construct a Decision Tree with root node.
3. Calculate Similarity Weight

$$S.W = \frac{(\sum \text{Residuals})^2}{\sum P_r(1-P_r)}$$

4. Calculate Gain

$$S.W(\text{Root}) = \frac{0.25}{1.75} = 0.14$$

$$1 \quad 0.25 \times 7 \quad 1.75$$

DT 1

$\{x_i, R_i\}$

$[-0.5, 0.5, 0.5, -0.5, 0.5, 0.5, -0.5]$

① Base Model → $Pr = 0.5$

SW(Root) = 0.14

Salary

<50K: $-0.5, 0.5, 0.5, -0.5$ — SW = 0

>50K: $-0.5, 0.5, 0.5$ — SW = 0.33

Credit (<50K): B → -0.5; G, N → $0.5, 0.5, -0.5$ — SW = 1

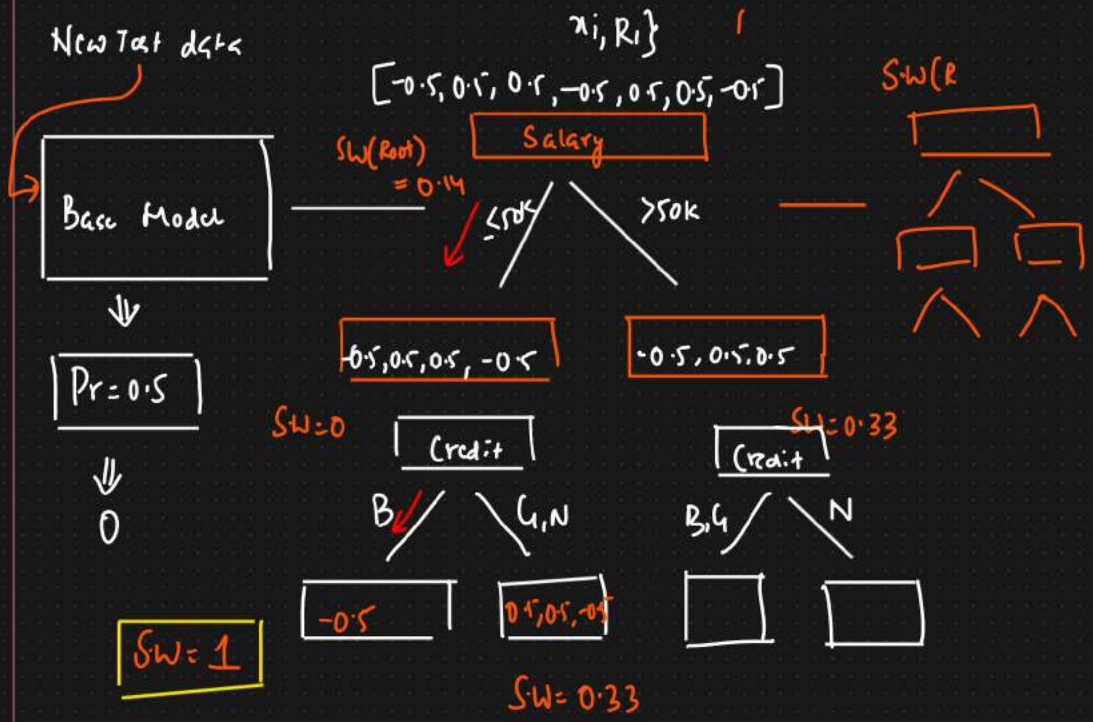
Credit (>50K): B, G → ; N → — SW = 0.33

$$S.W(LC) = \frac{(\sum \text{residual})^2}{\sum P_r(1-P_r)} = \frac{[-0.5+0.5+0.5-0.5]}{0.5(0.5)+0.5(0.5)+0.5(0.5)+0.5(0.5)} = 0$$

$$SW(RC) = \frac{[-0.5+0.5+0.5]^2}{0.5(0.5)+0.5(0.5)+0.5(0.5)} = \frac{0.25}{0.75} = 0.33$$

$$\text{Gain} = 0 + 0.33 - 0.14 = 0.19 //.$$

Final O/P Classification



Test data

[≤50k, B] ⇒ O/P ??

log odds formula

$$\log(\text{odds}) = \log\left(\frac{P}{1-P}\right)$$

$$= \log\left(\frac{0.5}{0.5}\right)$$

$$= 0 //$$

Test Data O/P = $\sigma(0 + \alpha(1))$

α = learning Rate

$\alpha = 0.1$

$= \sigma(0 + 0.1)$

$= \sigma(0.1)$

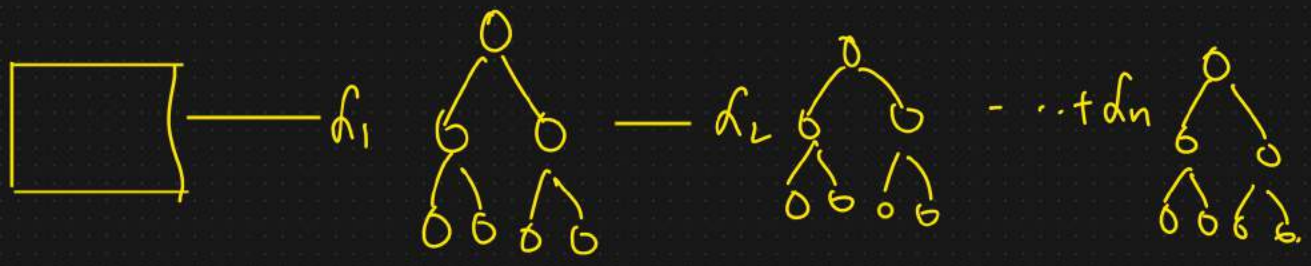
$\sigma = \frac{1}{1+e^{-z}}$

$= \frac{1}{1+e^{-0.1}}$

$= 0.52 //$ ⇒ Threshold ⇒ 0.6 ⇒ Domain Expertise

$0.52 < 0.6 \Rightarrow 0$

Xgboost Classifier



$$O/p = \sigma\left(\text{Base learner} + f_1(DT_1) + f_2(DT_2) + \cdots + f_n(DT_n)\right)$$

Xg Boost classifier

Xgboost Regressor ML Algorithm

Dataset {Regression}

Exp	Gap	Salary	R_1	$\hat{y}$	R_2	
2	Yes	40K	-11	49.9	-9.9	$[51 + (0.1)(-10)] = 51 - 0.1 = 49.9$
2.5	Yes	42K	-9	49.9	-7.9	
3	No	52K	1	51.5	0.5	
4	No	60K	9	51.5	8.5	$[51 + (0.1)(5)] = 51.5$
4.5	Yes	62K	11	52.1	9.9	$[51 + 0.1(11)] = 51 + 1.1 = 52.1$

≈ 51K

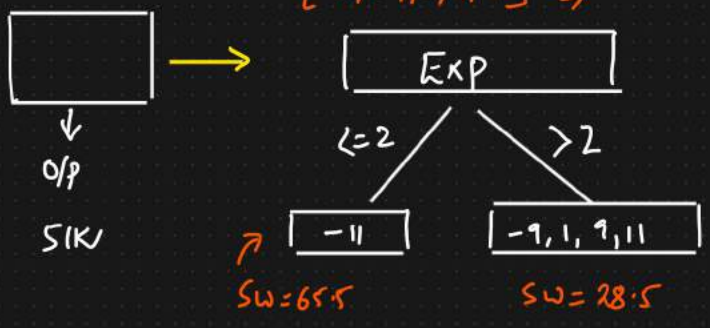
$$\text{Similarity weight} = \frac{\Sigma(\text{Residual})^2}{\Sigma p_r(1-p_r)}$$

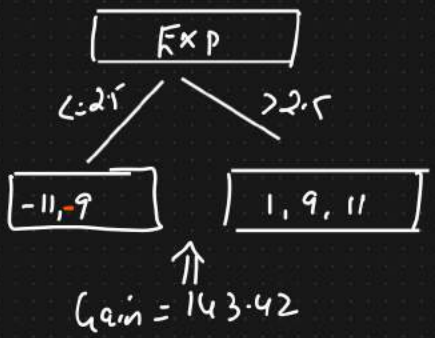
Gain

Steps

① Create a Base Model ② Residual Computation ③ Construct DT1 using $\{x_i, R_1\}$

[-11, -9, 1, 9, 11] ⇒ SW = 0.16.





$$\text{Similarity weight} = \frac{\Sigma(\text{Residual})^2}{\text{No. of Residuals} + \lambda}$$

λ → Hyperparameter

$\lambda = 1$

$\lambda \uparrow$ SW ↓

$$SW(\text{left child}) = \frac{121}{1+1} = 121/2 = 65.5$$

$$SW(\text{Right child}) = \frac{(-9+1+9+11)^2}{4+1}$$

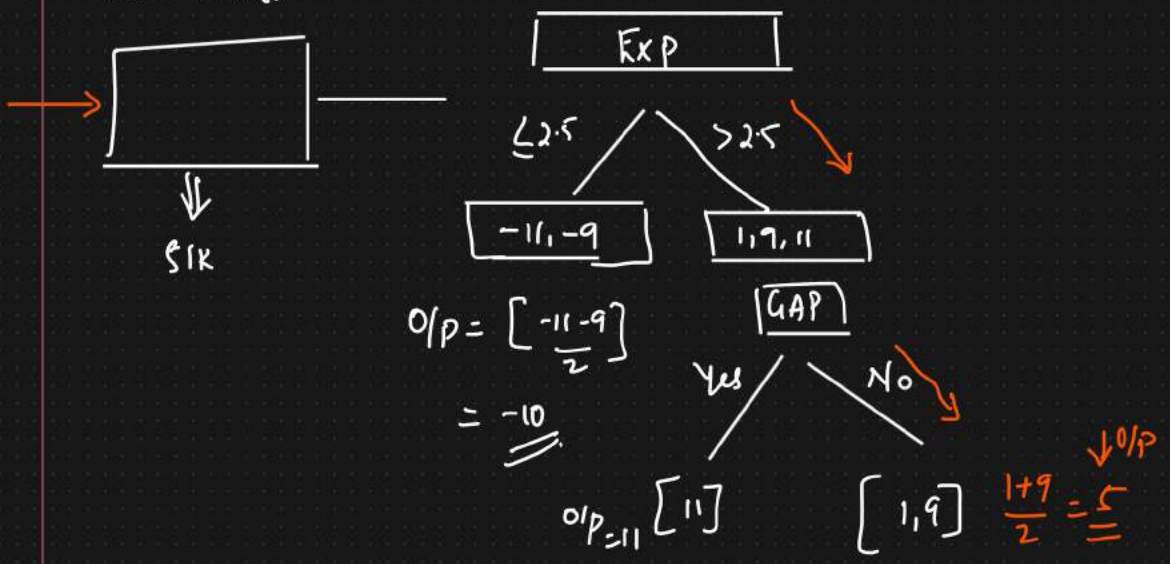
$$= \frac{144}{5} = 28.5$$

(5) Calculate Gain

$$\text{Gain} = 65.5 + 28.5 - 0.16$$

$$= 98.34$$

DT 1



α = Learning Rate $\alpha = 0.1 \Rightarrow$ Hyperparameter

$$\text{XGBoost Classifier} = \text{Base Learner} + \alpha_1(DT1) + \alpha_2(DT2) + \cdots + \alpha_n(DTn)$$

$$\text{XGBoost Regressor} = \overset{O/P}{51K} + 0.1(5)$$

$$= 51 + 0.5$$

$$= 51.5$$

$$\text{Similarity weight} = \frac{\Sigma(\text{Residual})^2}{\text{No. of Residuals} + \lambda}$$

{Regression}

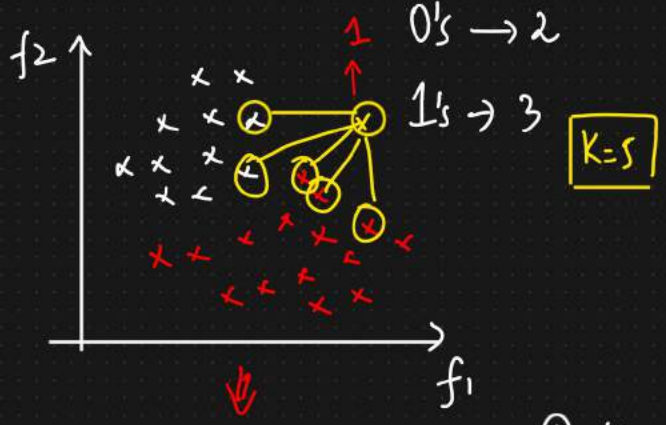
$$\text{Similarity Weight} = \frac{\sum (\text{Residual})^2}{\sum Pr(1-Pr) + \boxed{\lambda}}$$

K Nearest Neighbour (KNN)

① Classification

② Regression

① Classification



{Binary Category} {Multi class}.

f1	f2	y
–	–	0
		1
		1
		0

① We have to initialize the K value

K=5

K ≥ 0

K = 1, 2, 3, 4, 5, 6 ⇒ Hyperparameter

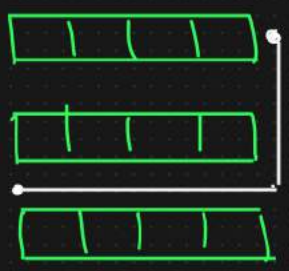
② Find the K Nearest Neighbour from the Test DATA

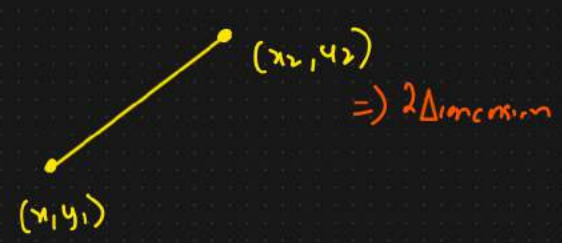
③ From those K=5 how many neighbours belongs to 0 category and 1 category.

Distance Metrics

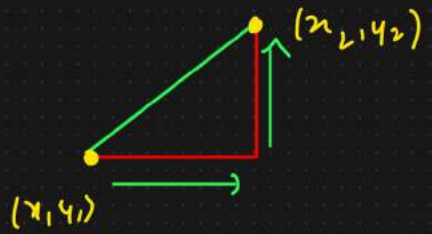
① Euclidian Distance

② Manhattan Distance





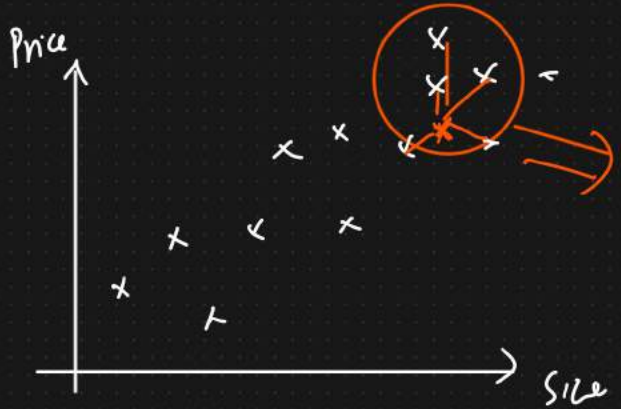
=) 2 Dimension



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

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

(2) Regression

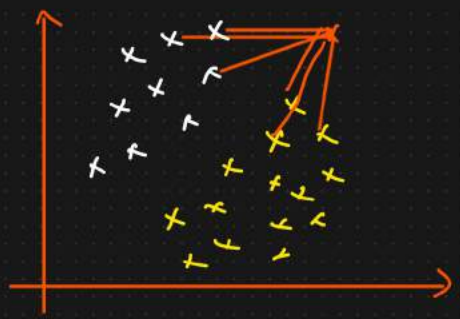


$k = 5$

Average or Median.

O/P Continuous O/P.

Variant of KNN



Time Complexity ↑↑↑

$O(n)$

① KD Tree
② Ball Tree
} Binary Tree

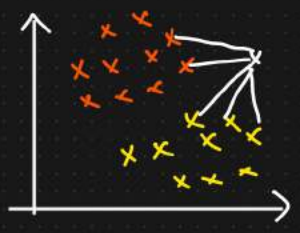
⇓

Time complexity ↓↓.

Variants of KNN

Two Variants

① K D Tree ✓
② Ball Tree ✓



Time Complexity

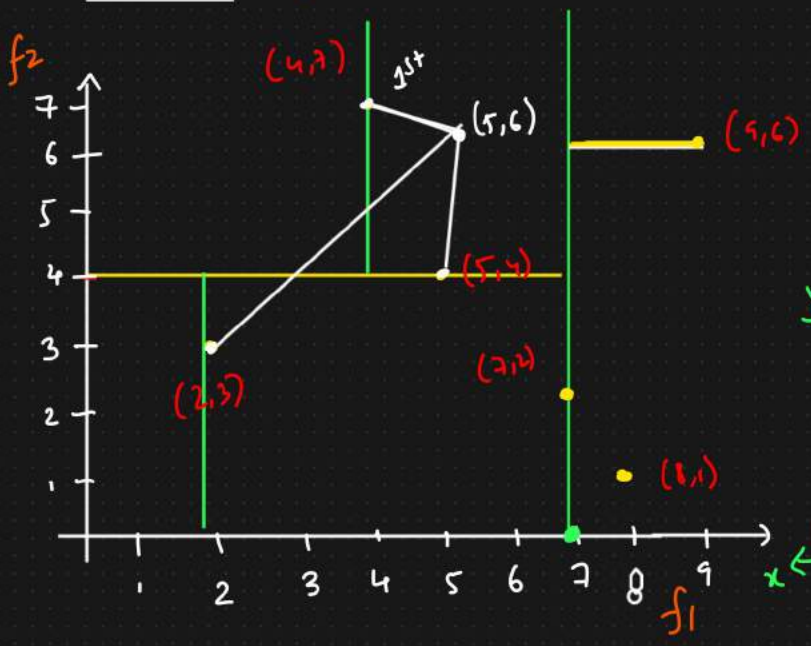
Brute Search

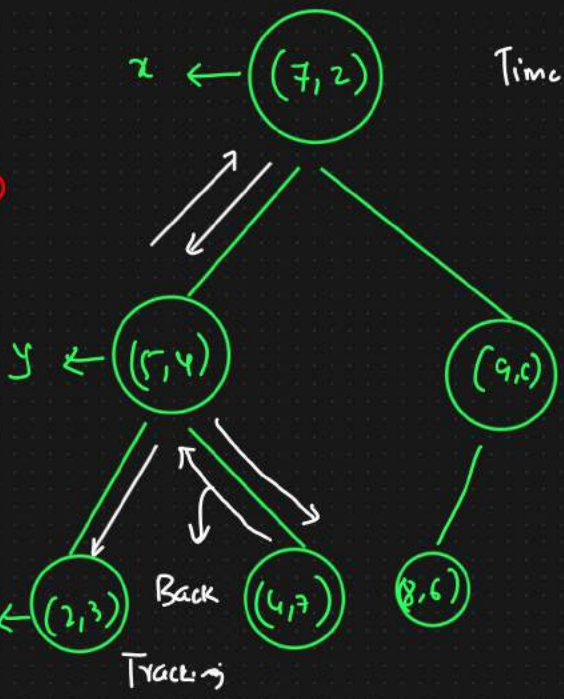
Auto

Binary Tree

K=3

① KD Tree => k - Dimensions Tree





Time Complexity ↓↓.

① Median of the x Cordinates

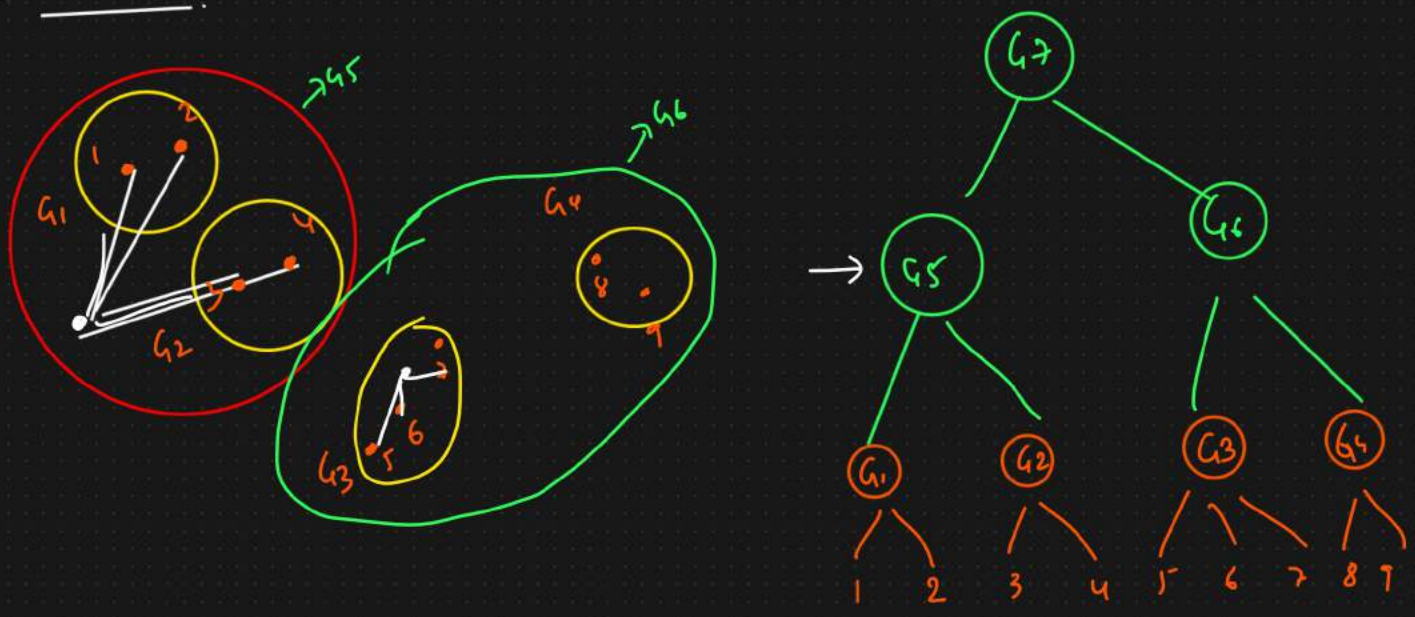
2, 4, [5, 7] 8, 9

$\frac{5+7}{2} = 6.5 \Rightarrow 7$

② Median of y axis

1, 2, [3.4,] 6.7

② Ball Tree



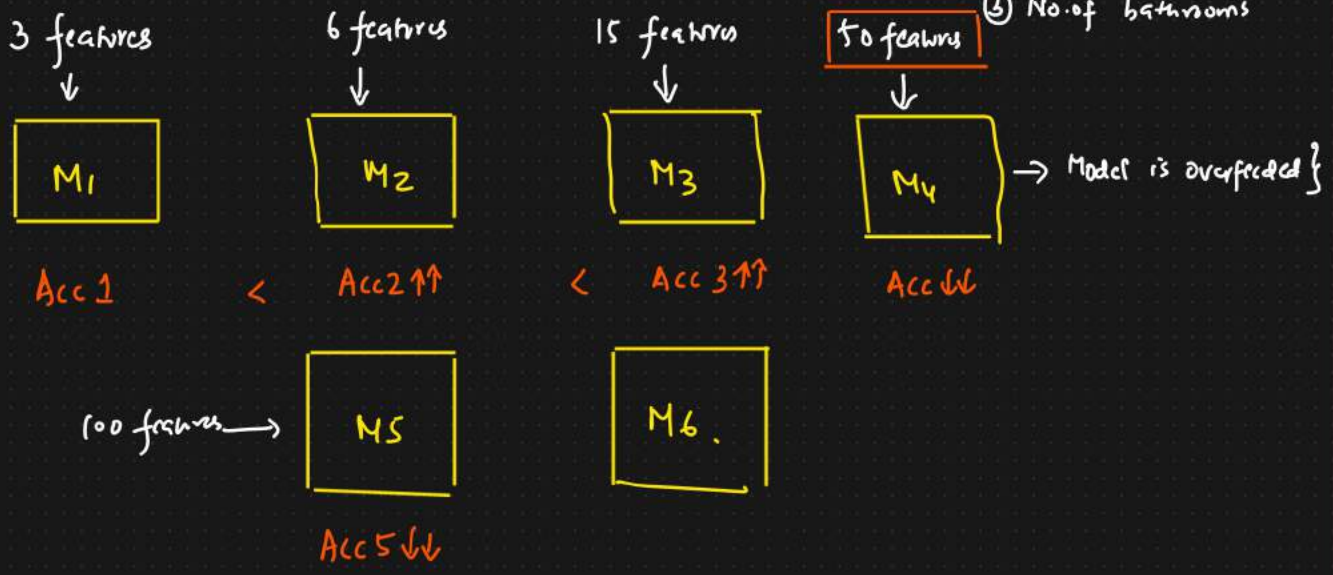
Principal Component Analysis (PCA) [Dimensionality Reduction]

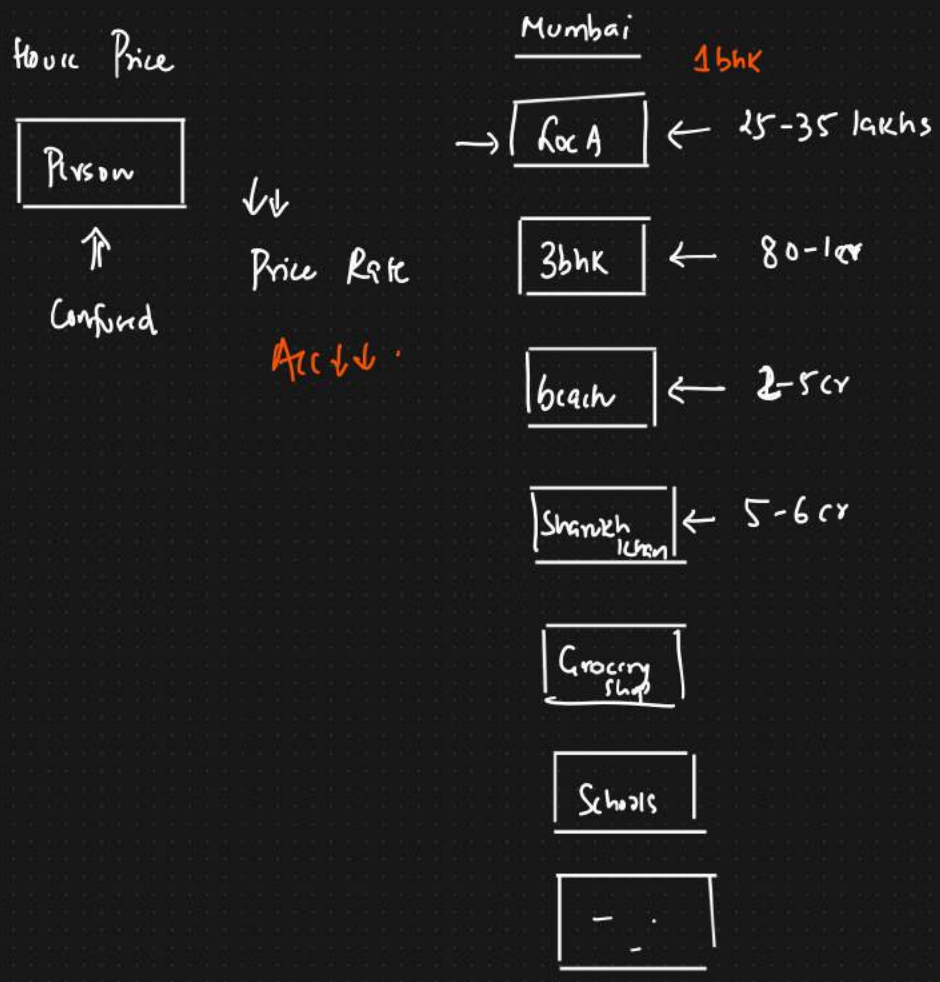
⇒ Price of the House ⇐

① Curse Of Dimensionality.

Dataset = 500 features ⇒

(i) House size
(2) No. of bedrooms
(3) No. of bathrooms





f1 f2 . . . f100
↓↓
PCA
f1 f2 . . . f20

Two different ways to remove Curse of Dimensionality

500 D ⇓ 20 D ⇒ Problem

① Feature Selection ⇓ Imp features

② Dimensionality Reduction (PCA) ⇓ Feature Extraction ⇓ Principal Component Analysis ⇓ Eigen Value & Vectors

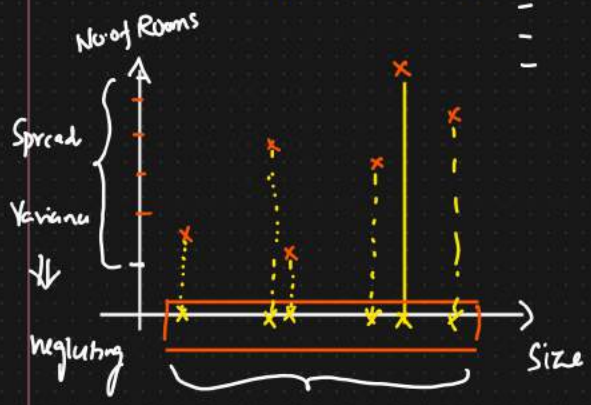
[500 features] ⇓ [20 features]

PCA Geometric Intuition [Dimensionality Reduction]

x_1	x_2	o/p
Size of Rooms	No. of Rooms	Price

Aim → 2d → 1 dimension

① Feature Selection ⇓ 2D → 1D



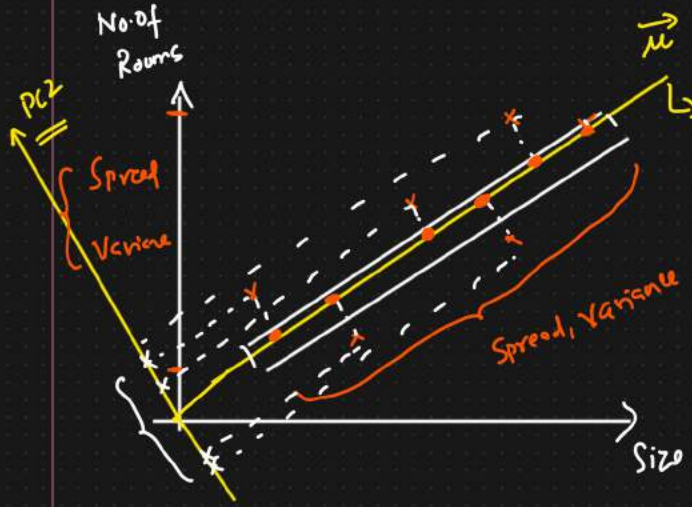
Spread, Variance

3D → 2D → PC1, PC2, PC3

Feature Extraction

② PCA → 2D → 1D

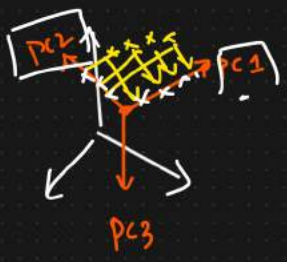
Variance, Spread ⇓ Information of the DATA



$\vec{\mu}$ → PC1 {Principal Component 1}

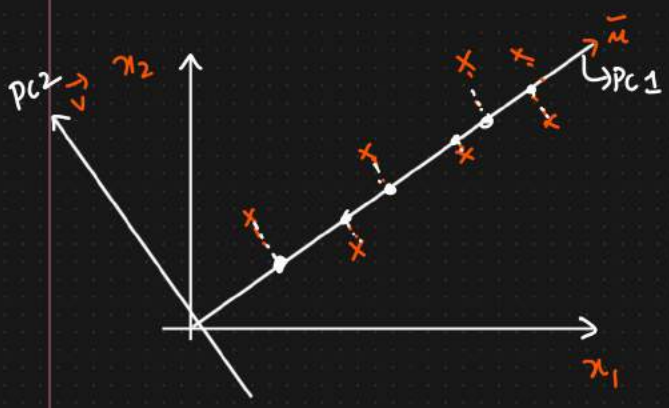
2 features → PC1, PC2

3 feature → PC1, PC2, PC3



3D - 2D

Maths Intuition behind PCA Algorithms

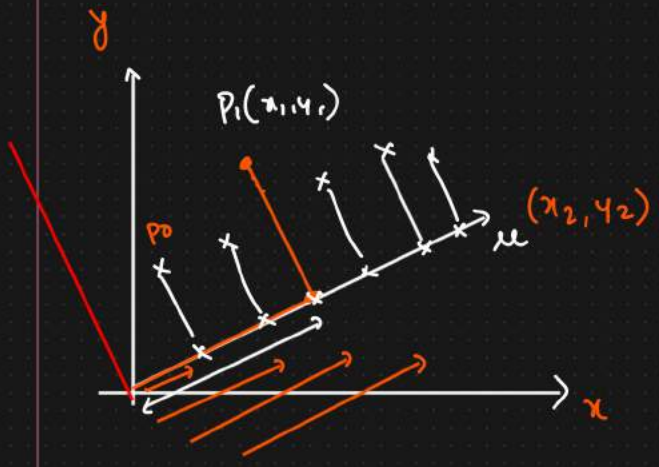


2D → 1D

PC1 > PC2

① Projections

② Optimization → Principal Component Max Variance



$$\text{Proj}_{P_1}\mu = \frac{P_1 \cdot \mu}{\|\mu\|} \qquad \|\mu\| = 1$$

$$\boxed{\text{Proj}_{P_1}\mu = P_1 \cdot \mu} \qquad \begin{bmatrix} x_1 \\ y_1 \end{bmatrix} \cdot \begin{bmatrix} x_2 & y_2 \end{bmatrix}$$

$$\therefore \boxed{x_1x_2 + y_1y_2} \Rightarrow \text{Scalar value}$$

$$\boxed{P_0', P_1', P_2', P_3' \cdots P_n'}$$

⇓

Scalar Value.

⇓

Variance

$P_0', P_1', P_2', P_3' \dots P_n'$

$x_0, x_1, x_2, x_3 \dots x_n$

② Max Variance $= \sum_{i=1}^{n} \frac{(x_i - \bar{x})^2}{n}$ ⇒ Cost function.

Goal : { Find the best unit vector which capture maximum variance }.

(Q) → How to find the vectors ?

Eigen value Decomposition → Eigen values and Eigen Vectors

f_1 f_2 O/p

2D → 1D

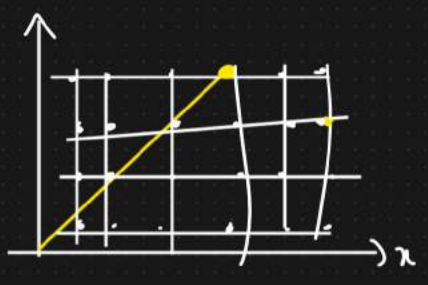
① Covariance Matrix between features $Cov[f_1, f_2]$.

② Eigen value and Eigen will be found out using this Covariance Matrix $\boxed{A v = \lambda v}$

λ → Eigen value

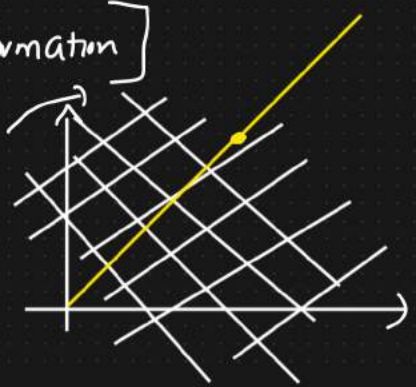
③ Eigen Vector → Eigen Value → Capturing the maximum variance.

Eigen vectors And Eigen Values [Linear Transformation]



$Cov[x, y]$

$\begin{bmatrix} \cdot & \cdot \\ \cdot & \cdot \end{bmatrix}$



$$\begin{bmatrix} & \\ & \end{bmatrix} * [v] = \lambda * v$$

λ ↓ Eigen value ⇓ Magnitude

Eigen vector ⟶ Max Magnitude ⇓ Max Eigen Value ⇓ PC1

Steps to Calculate Eigen Value and Eigen Vector [2d → 1d]

① Covariance of features.

x, y, z → °/p

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

$Cov(x,x) = Var(x)$

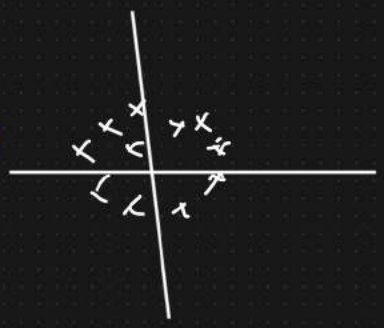
$$A = \begin{array}{c|cc} & x & y \\ \hline x & Var(x) & Cov(x,y) \\ y & Cov(y,x) & Var(y) \end{array}$$

$$A \cdot v = \lambda \cdot v$$

λ_1 λ_2 ⇒ Eigen Values

3d - 2d ⇓ λ_1 ⇓ PC1, λ_2 ⇓ PC2.

3d - 1d ⇓ λ_1 ⇓ PC1

①  ⇒ Standardize the dataset ⇒ 

② Cov (X, Y)

③ Find out the Eigen value & Eigen vector

$$Av = \lambda \cdot v$$

2 3 . . . n

Variance ↑↑ ⇒ PC1, PC2, PC3 . . .

Unsupervised Machine Learning

Clustering Algorithms

1. K Means Clustering
2. Hiranichal Clustering.
3. DBScan Clustering.

I/P — Dependent

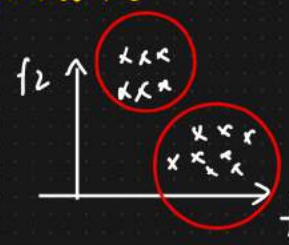
f_1 f_2 - - f_n | o/p

Supervised ML

f_1 f_2 f_3 - - f_n

_ _ _ _

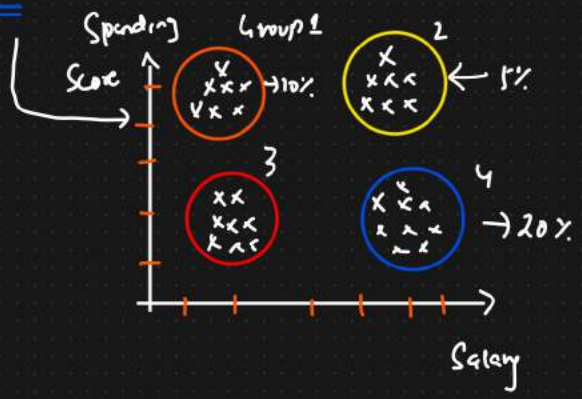
_ _ _ _



Ecommerce → Iphone 20 → 1.8 lakhs

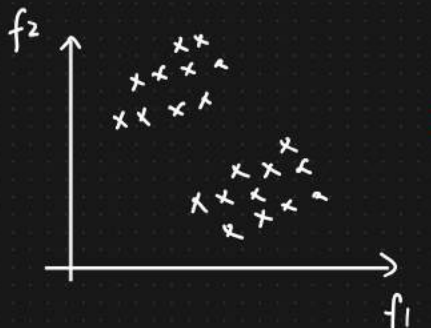
Dataset

Salary	Spending Score (1-10)
-	-
-	-
-	-

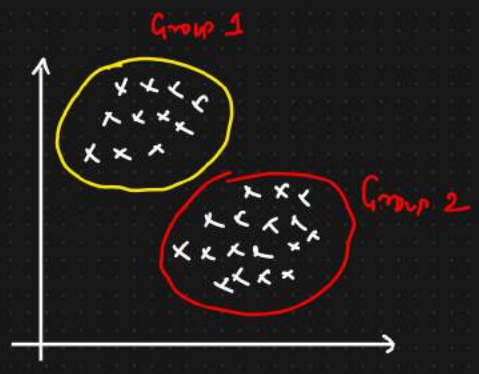


1. K Means Clustering

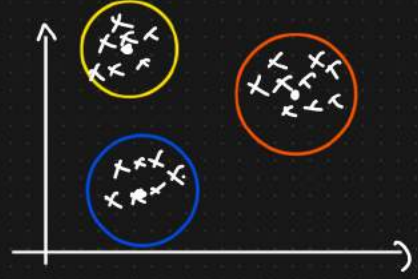
Geometric Intuition



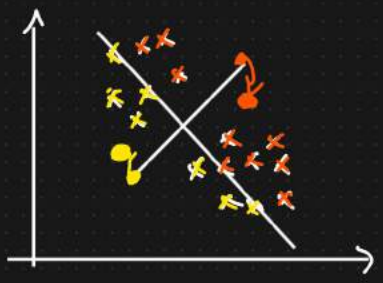
⇒ K Means

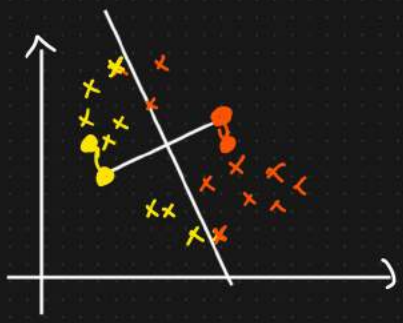


⇒ K Means



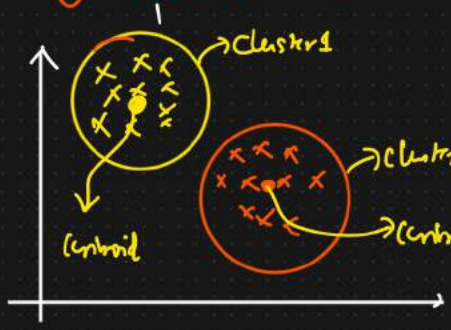
K Means Mathematical Intuition





Steps: $k=2$

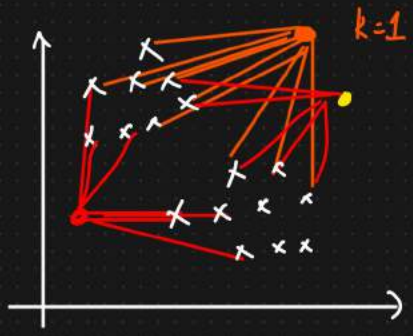
1. Initialize some K → centroids
2. Points that are nearest to the centroid → Group
3. Move the centroids → Mean



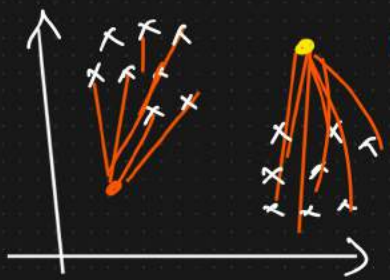
How do we select the K Value ?

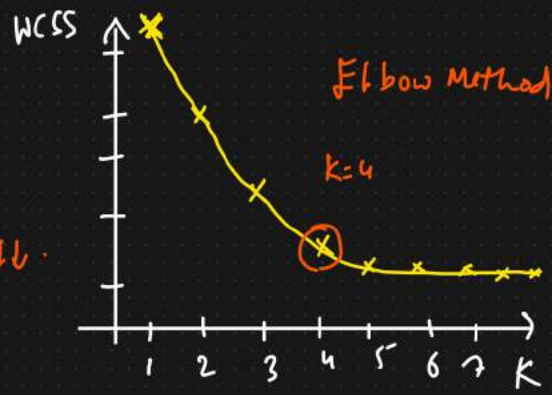
WCSS = Within Cluster Sum of Squares

Initialize K = 1 to 20

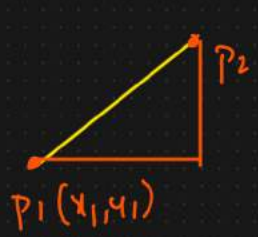


$$WCSS = \sum_{i=1}^{K} (\text{Distance between point to the nearest centroid})^2$$





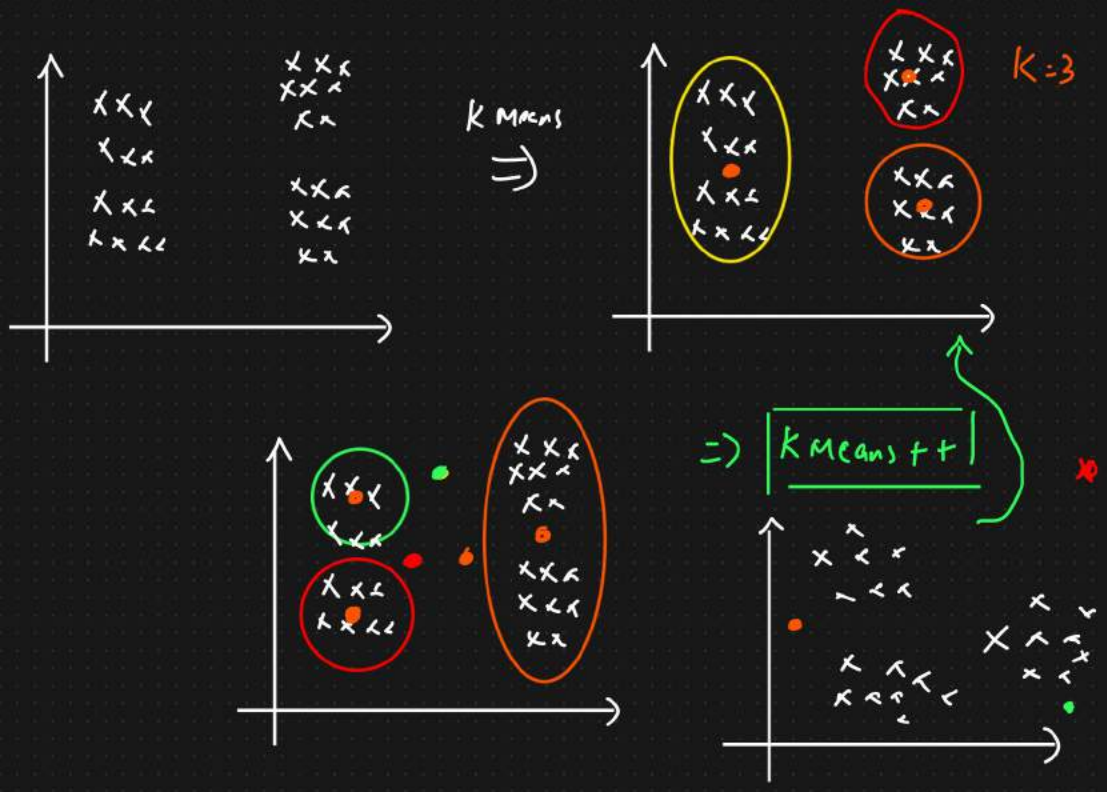
Euclidian Distance or Manhattan Distance



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

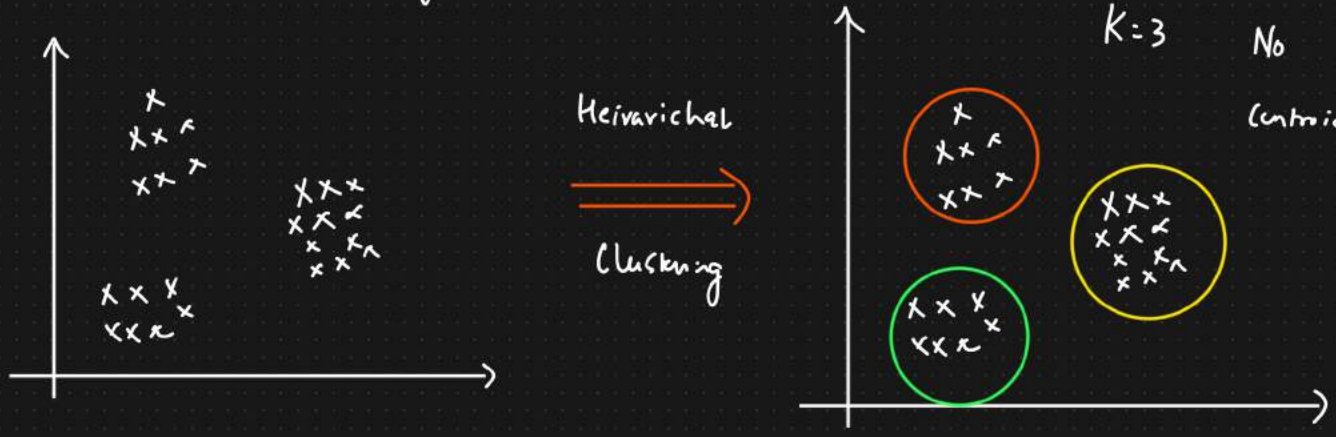
$$MD = |x_2 - x_1| + |y_2 - y_1|$$

Random Initialization Trap (K Means ++)



K Means ++ Initialization Technique

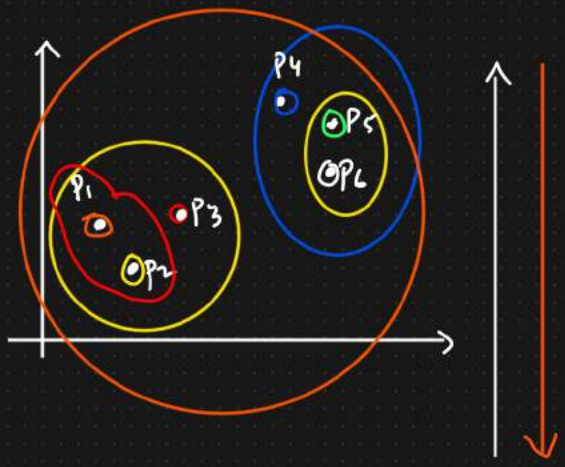
Hierarichal Clustering



HC

① Agglomerative
② Divisive
} => Geometric Intuition

Steps



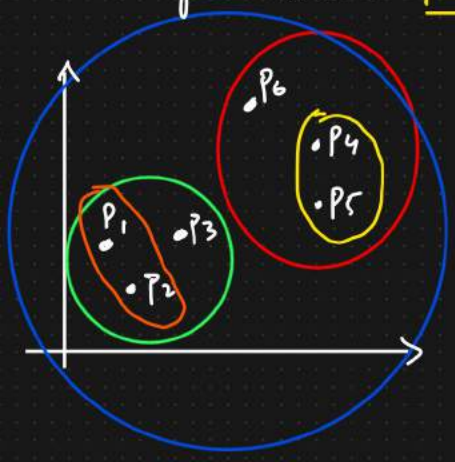
① For each point initially we will consider it as a seperate cluster

② Find the nearest point and Creat a new cluster

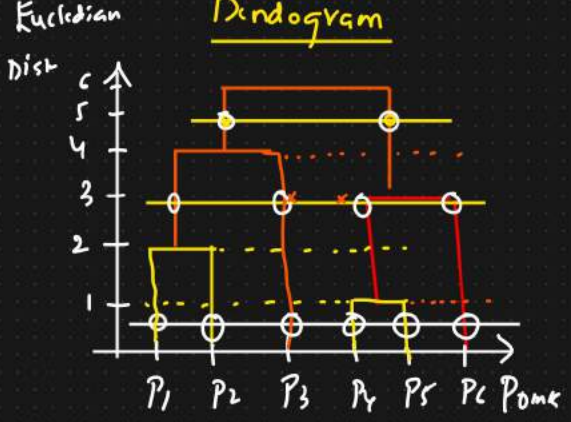
③ Keep on doing the same process [step 2] until we get a single cluster.

Threshold

How many clusters? K=2



Dendogram



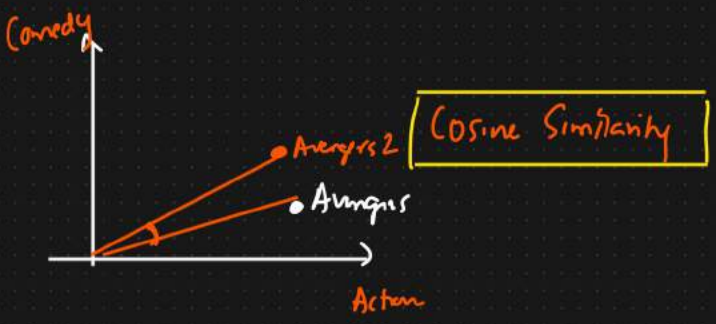
⑧ Select the longest vertical line such that no horizontal line passed through it.

K Means Vs Hierarchical Clustering

Scalability And Flexibility

① Dataset size
- Huge => K Means
- Small => Hierarchical Clustering

② Types of Data
- Numerical data → K Means or Hierarchical
- Variety of data → Hierarchical



Cosine Similarity

Silhouette Clustering

①

a(i)

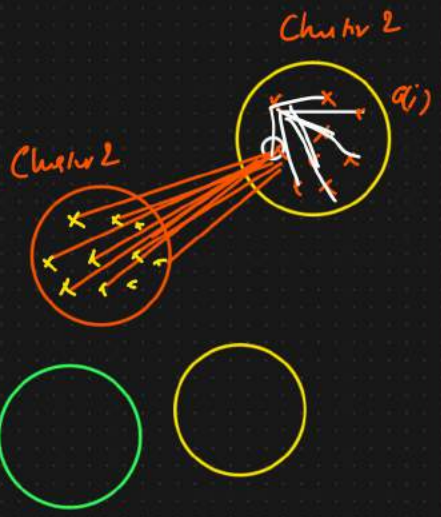
For data point $i \in C_I$ (data point i in the cluster C_I), let

$$a(i) = \frac{1}{|C_I| - 1} \sum_{j \in C_I, i \neq j} d(i, j)$$

be the mean distance between i and all other data points in the same cluster, where $|C_I|$ is the number of points belonging to cluster i, and d(i,j) is the distance between data points i and j in the cluster C_I (we divide by $|C_I| - 1$ because we do not include the distance d(i,i) in the sum). We can interpret a(i) as a measure of how well i is assigned to its cluster (the smaller the value, the better the assignment).

②

b(i)



We then define the mean dissimilarity of point i to some cluster C_J as the mean of the distance from i to all points in C_J (where $C_J \neq C_I$).

For each data point $i \in C_I$, we now define

$$b(i) = \min_{J \neq I} \frac{1}{|C_J|} \sum_{j \in C_J} d(i, j)$$

to be the *smallest* (hence the $\min$ operator in the formula) mean distance of i to all points in any other cluster (i.e., in any cluster of which i is not a member). The cluster with this smallest mean dissimilarity is said to be the "neighboring cluster" of i because it is the next best fit cluster for point i.

③ Silhouette Score

small ← a_i/b_i

$a_i > b_i$ => Good cluster

$a_i < b_i$ => bad cluster

We now define a *silhouette* (value) of one data point i

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}, \text{ if } |C_I| > 1$$ => -1 to 1

and

$$s(i) = 0, \text{ if } |C_I| = 1$$

Which can be also written as:

$$s(i) = \begin{cases} 1 - a(i)/b(i), & \text{if } a(i) < b(i) \\ 0, & \text{if } a(i) = b(i) \\ b(i)/a(i) - 1, & \text{if } a(i) > b(i) \end{cases}$$

From the above definition it is clear that

$$-1 \leq s(i) \leq 1$$