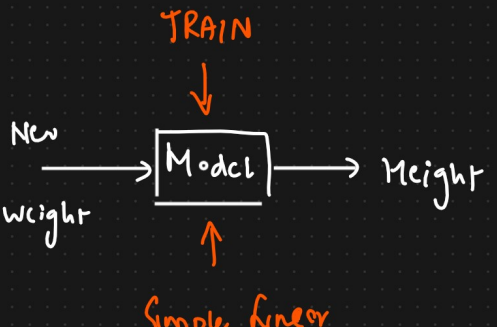
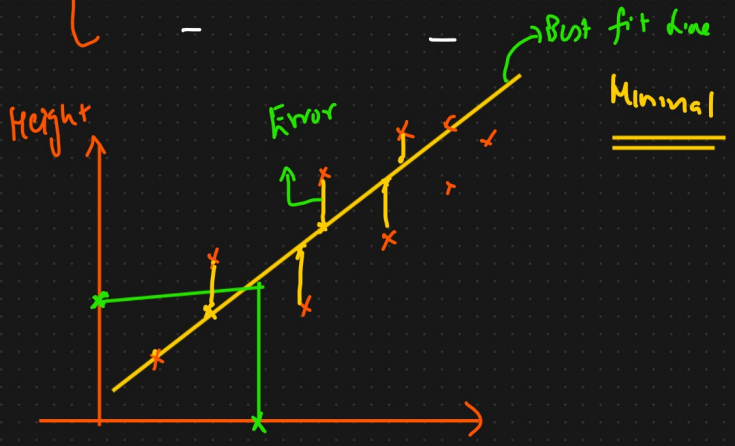
Simple Linear Regression

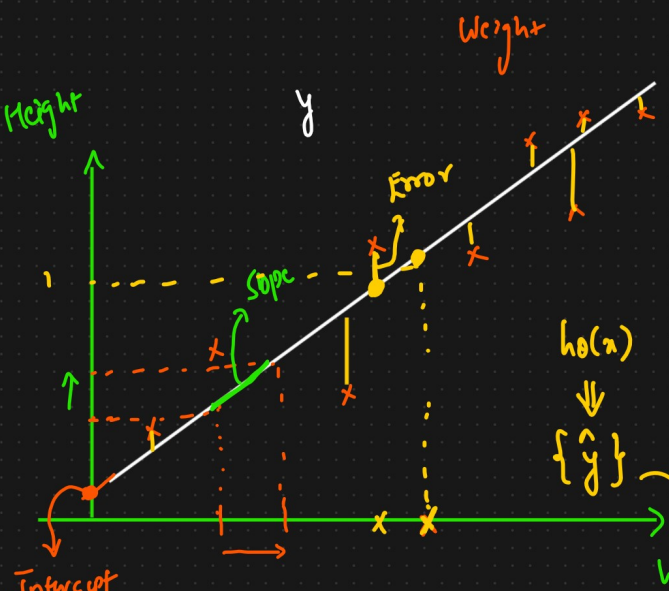
Supervised ML → Regression



Dataset — I/P features ↑ X — O/P or dependant feature ↑ y

Weight	Height
74	170cm
80	180cm
75	175.5cm
–	–

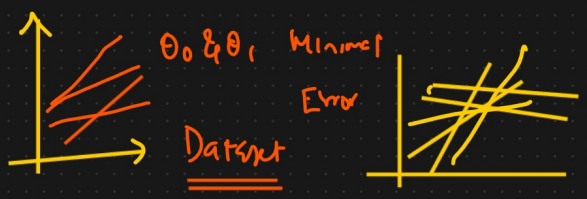




$h_\theta(x) = \theta_0 + \theta_1 x$

θ_0 = Intercept

θ_1 = slope or coefficient



$y = mx + c$

$y = \beta_0 + \beta_1 x$

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

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

$h_\theta(x) \Rightarrow \{\hat{y}\}$ → predicted point

Error $(y - \hat{y})$

if $x = 0$

$h_\theta(x) = \theta_0$

Cost function

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

$h_\theta(x)^{(i)}$: predicted; $y^{(i)}$: true O/P; $\left(h_\theta(x)^{(i)} - y^{(i)}\right)$: Error

$\Rightarrow$ Mean Squared Error

Final Aim: What we need to solve

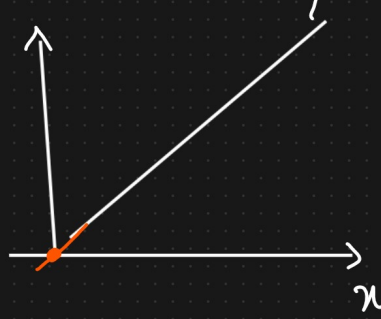
Minimize $J(\theta_0,\theta_1) = \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta(x)^{(i)} - y^{(i)}\right)^2$ ↓↓↓

θ_0, θ_1

① $h_\theta(x) = \theta_0 + \theta_1 x$

$\boxed{\theta_0 = 0}$

$\boxed{h_\theta(x) = \theta_1 x}$

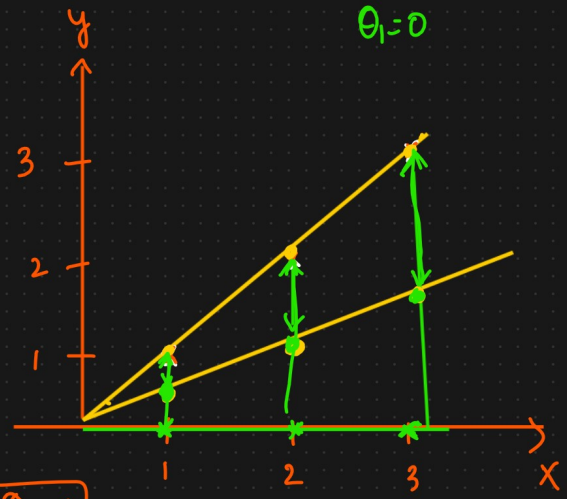


DATASET

x	y
1	1
2	2
3	3

$\theta_0 = 0$

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



$\theta_1 = 0$

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

Let $\theta_1 = 1$ {slope}

$h_\theta(x) = 1$ $x=1$

$h_\theta(x) = 2$ $x=2$

$h_\theta(x) = 3$ $x=3$

$h_\theta(x) = \theta_1 x$ ↓ Let $\theta_1 = 0.5$

$h_\theta(x) = 0.5$ if $x=1$

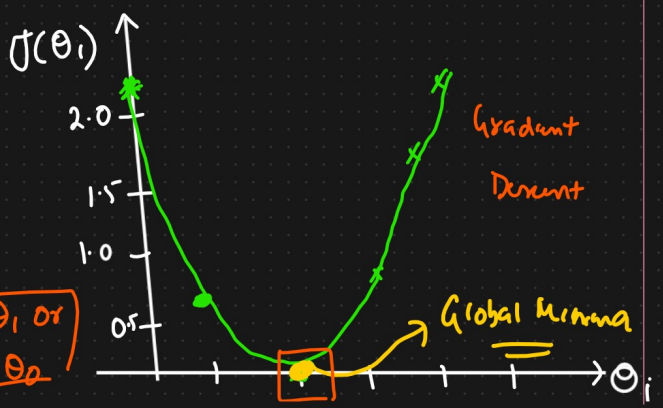
$h_\theta(x) = 1$ if $x=2$

$h_\theta(x) = 1.5$ if $x=3$

$\boxed{\theta_1 = 1}$

$$J(\theta_1) = \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta(x)^{(i)} - y^{(i)}\right)^2$$

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



$J(\theta_1) = 0$ ←

$\theta_1 = 0.5$

Error has been minimized

0.5 1 1.5 2.0 2.5

$$J(\theta_1) = \frac{1}{2\times3}\left[(0.5-1)^2 + (1-2)^2 + (1.5-3)^2\right]$$

$J(\theta_1) \approx 0.58$

if $\theta_1 = 0$

$$J(\theta_1) = \frac{1}{2\times3}\left[(0-1)^2 + (0-2)^2 + (0-3)^2\right]$$

$J(\theta_1) \approx 2.3$

Convergence Algorithm {Optimize the changes of θ_1 value}

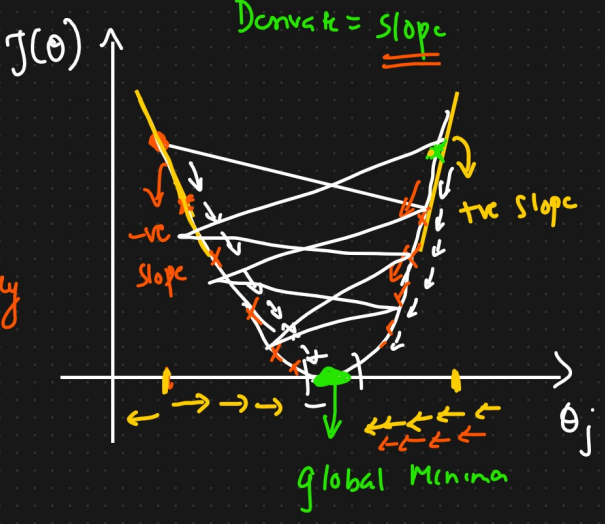
Repeat until convergence

{

$$\theta_j := \theta_j - \alpha\left[\frac{\partial}{\partial\theta_j} J(\theta_j)\right] \rightarrow -ve$$

}

θ_1 value reach more efficiently



$\theta_j = \theta_j - \alpha(+ve)$

$= \theta_j - (+ve)$

α = Learning Rate

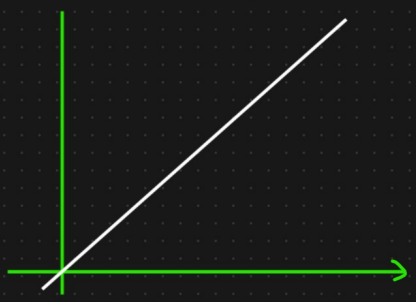
$\alpha = 0.001$ ←

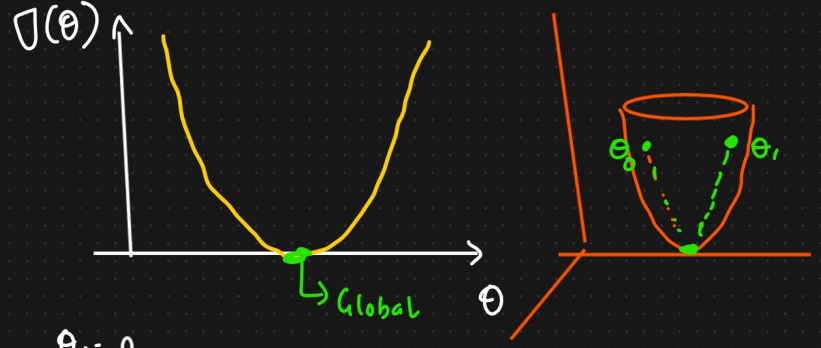
{ $\theta_j = \theta_j - \alpha(-ve)$

$\theta_j = \theta_j + (+ve)$ }

Final Conclusion

GRADIENT DESCENT





$\theta_0 = 0$

θ_1 = Changeable

Convergence Algorithm

repeat until convergence

{

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

}

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

$j = 0$ and 1

$$\frac{\partial}{\partial x} x^n = n x^{n-1} \qquad \frac{\partial}{\partial x} (x)^2 = 2x$$

$$\rightarrow \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) = \frac{\partial}{\partial \theta_j} \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x)^{(i)} - y^{(i)}\right)^2$$

if

$$h_\theta(x) = \theta_0 + \boxed{\theta_1 x} \rightarrow 0$$

$$j = 0 \Rightarrow \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1) = \frac{\partial}{\partial \theta_0} \frac{1}{2m} \left[\sum_{i=1}^{m} \left(h_\theta(x)^{(i)} - y^{(i)}\right)^2 \right]$$

$$= \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x)^{(i)} - y^{(i)}\right) \times 1$$

$$j=1 \Rightarrow \frac{\partial}{\partial \theta_1} J(\theta_0,\theta_1) = \frac{\partial}{\partial \theta_1} \frac{1}{2m} \left[\sum_{i=1}^{m} \left((\theta_0+\theta_1 x) - y^{(i)} \right)^2 \right]$$

$$= \frac{1}{m} \sum_{i=1}^{m} \left((\theta_0+\theta_1 x) - y^{(i)} \right) (x)$$

$$\frac{\partial}{\partial \theta_1} \left[\theta_0 + \theta_1 x \right] \Rightarrow x$$

Repeat until Convergence

{

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x)^{(i)} - y^{(i)} \right)$$

$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x)^{(i)} - y^{(i)} \right) x^{(i)}$$

}

Multiple Linear Regression

Dataset

I/p (Weight)	O/p feature (Height)
—	—
—	—

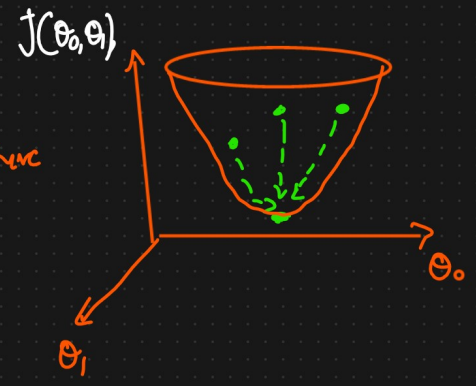
$h_\theta(x) = \theta_0 + \theta_1 x$ ← I/p or Independent

θ_0 = Intercept

θ_1 = Slope

House Pricing Dataset

Independent features			O/p feature
No. of Rooms	Size of House	Location	Price



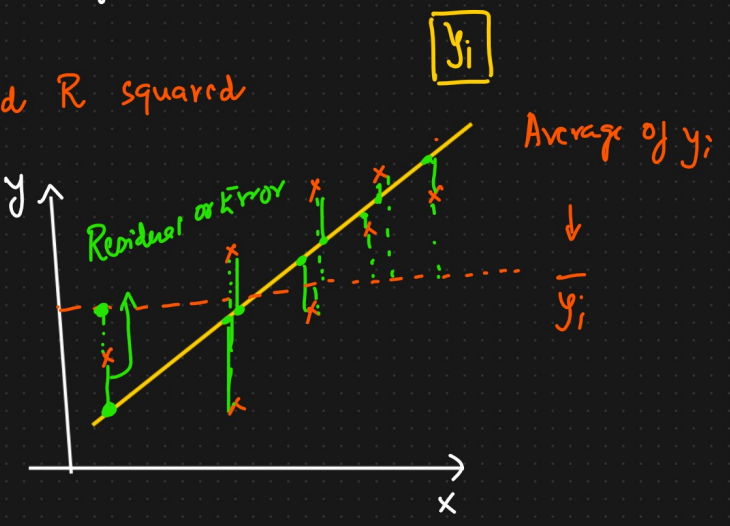
$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3$ {Multiple Linear Regression}

$\theta_1, \theta_2, \theta_3$ = coefficient

θ_0 = Intercept

Performance Metrics Used In Linear Regression

① R squared ② Adjusted R squared



$$R^2 = 1 - \frac{SS_{Res}}{SS_{TOTAL}}$$

$$= 1 - \frac{\sum (y_i - \hat{y}_i)^2}{\sum (y_i - \bar{y}_i)^2}$$

$$= 1 - \frac{\text{Small number}}{\text{Bigg number}}$$

$$= 1 - \text{Smaller number}$$

$$\approx 1$$

$0.70 \Rightarrow 70\%$

$0.85 \Rightarrow 85\%$

$0.90 \Rightarrow 90\%$

{Overfitting, Underfitting}

1 ⇓

Accurate My Model Is?

R squared ↑↑↑

Size of the house ↑ Price ↑

+ve correlation

② Adjusted R Squared

Dataset → (Price)

Gender | Size of the house | No. of bedrooms | Location | Price

No. of bedroom ↑ Price ↑

+ve correlation

R squared = 75% ⇒ 0.75

This is the problem of R Squared

R squared => 80% => 0.80

R squared => 85% => 0.85

R squared => 87% => 0.87

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

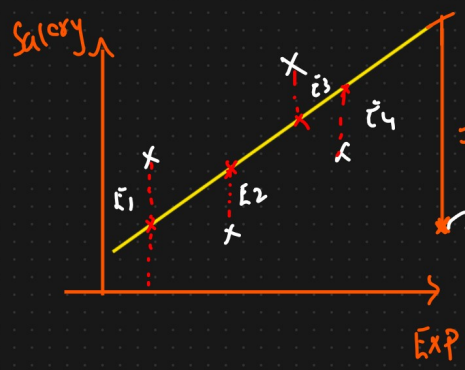
- $P=2$ $R^2 = 90\%$ $R^2_{adjusted} = 86\%$
- $P=3$ $R^2 = 92\%$ $R^2_{adjusted} = 82\%$

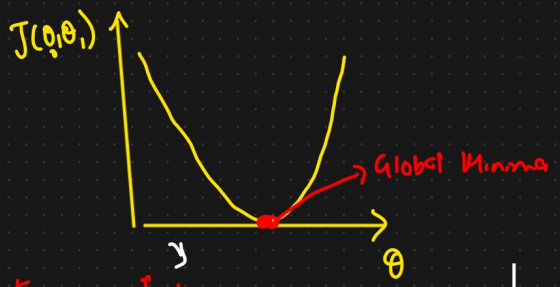
N = No. of data points

P = No. of Independent features

MSE, MAE, RMSE [Cost function] → Performance Metrics

R^2 and Adjusted R^2





EXP	Salary
—	—
—	—
—	—

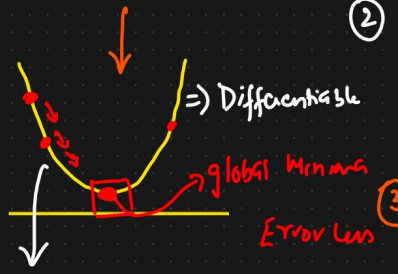
MSE↑↑

MSE

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

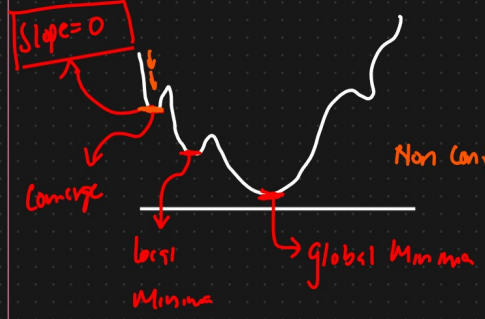
⇒ Cost function ↓↓

→ Quadratic Equation



$ax + by + c$

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



Advantage

1. Differentiable ✓
2. It has one local and one global Minima
3. Converges faster

Disadvantage

4. Not Robust to outliers
5. It is not in same unit.

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

Salary (lakhs)

x	y	$(y_i - \hat{y}_i)^2$ (MSE)
		$(1\text{akh})^2$

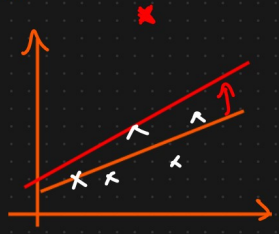
Error < 2.5 ⇒ $(1\text{akh})^2$ ⇐

② Mean Absolute Error (MAE)

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

lakhs

MAE↑↑

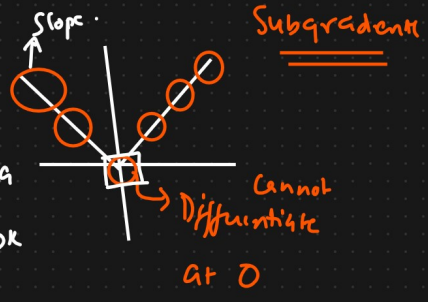


Advantage

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

Disadvantage

1. Convergence usually take more time. Optimization is a complex task
2. Time consuming



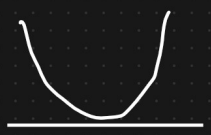
3. RMSE {Root Mean Squared Error}

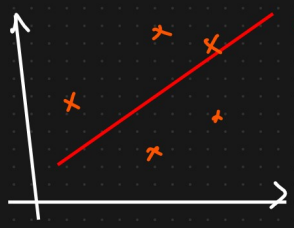
MAE, MSE, RMSE

Performance metric ↑

$$RMSE = \sqrt{MSE}$$

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





- → MSE ✓
- → MAE ✓
- → RMSE ✓
- → R^2 ✓
- → Adjusted R^2 ✓

Advantage

1. Same unit
2. Differentiable

Disadvantage

4. Not Robust to outliers

MSE Vs MAE Vs RMSE

R^2 Vs Adjusted R^2

Overfitting And Underfitting (Bias And Variance)

1. Training dataset
2. Test dataset
3. Validation dataset

70%-30%

Dataset

1000 datapoints

→ TRAINING DATASET (700) → Model To TRAIN

→ TEST DATASET (300) → Testing Model

Size of House	No. of bedrooms	Price
—	—	—
—	—	—
—	—	—
—	—	—

TRAINING DATASET

- TRAIN → Train the model
- VALIDATION → Hyperparameter Tunning your Model

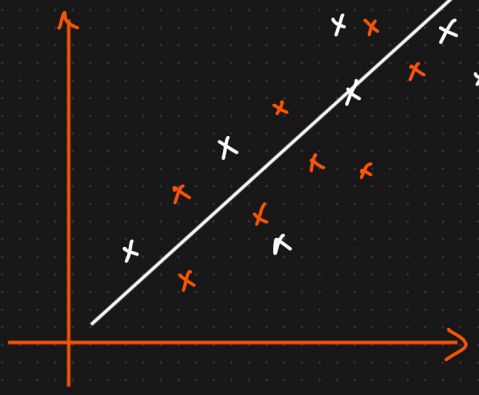
TRAIN	Very Good Accuracy (90%) [Low Bias]	Very Good Accuracy (90%) [Low Bias]
TEST	Very Good Accuracy (85%) [Low Variance]	Bad Accuracy (50%) [High Variance]
	⇑	⇓
	→ Generalized Model	Model is Overfitting

TRAIN — Model Accuracy is low [High Bias]

TEST — Model Accuracy is low [High Variance]

⇓

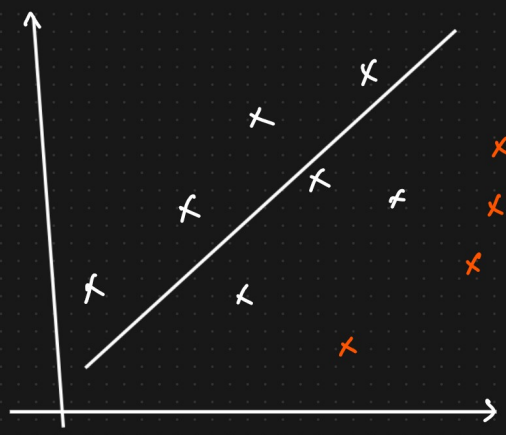
Model is Underfitting



Generalized Model

⇓

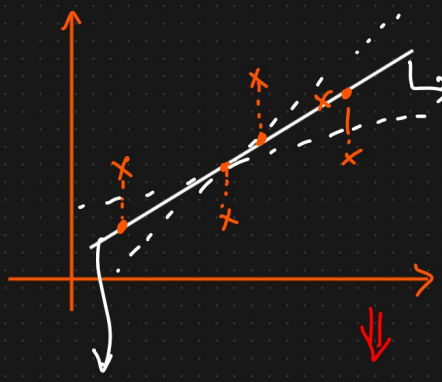
Low Bias, Low Variance



Overfitting

Low Bias, High Variance

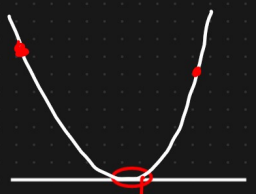
Linear Regression Using OLS {Ordinary Least Square}



↳ optimization

⇓

loss function



↳ Global Minima

⇓

→ OLS → Formula and Calculate

$h_\theta(x) = \beta_0 + \beta_1 x_1$

↳ $\hat{y}$

β_0 & β_1

= =

Global Minima

Ordinary Least Square

Error = MSE

$$S(\beta_0, \beta_1) = \frac{1}{n} \sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 x_i)^2$$

$y = x^2$

$\frac{dy}{dx} = 2(x)^{2-1} \frac{d}{dx}(x)$

$= 2x$

⇓

Him β_0, & β_1

= =

$$\frac{\partial S}{\partial \beta_0}(\beta_0, \beta_1) = \frac{2}{n} \sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 x_i)(0 - 1 - 0)$$

$$= -\frac{2}{n} \sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 x_i) = 0 \longrightarrow (1)$$

$$\frac{\partial S}{\partial \beta_1}(\beta_0, \beta_1) = \frac{2}{n} \sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 x_i)(-x_i) = 0 \longrightarrow (2)$$

$$= -\frac{2}{n} \sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 x_i)(x_i) = 0 \longrightarrow (2)$$

Eq → ①

$$\frac{-2}{n}\sum_{i=1}^{n}(y_i - \beta_0 - \beta_1 x_i) = 0$$

$$\Downarrow$$

$$-\sum_{i=1}^{n}(y_i - \beta_0 - \beta_1 x_i) = 0$$

$$-\sum_{i=1}^{n} y_i + n\beta_0 + \beta_1 \sum_{i=1}^{n} x_i = 0$$

$$n\beta_0 + \beta_1 \sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i$$

$$\beta_0 = \frac{\sum_{i=1}^{n} y_i}{n} - \beta_1 \frac{\sum_{i=1}^{n} x_i}{n}$$

Intercept →

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

Eq 2 ÷

$$\frac{-2}{n}\sum_{i=1}^{n}(y_i - \beta_0 - \beta_1 x_i)(x_i) = 0$$

$$\Downarrow$$

$$\sum_{i=1}^{n}(y_i - \beta_0 - \beta_1 x_i)(x_i) = 0$$

$$\Downarrow$$

$$\sum_{i=1}^{n} x_i y_i - \beta_0 \sum_{i=1}^{n}(x_i) - \beta_1 \sum_{i=1}^{n}(x_i)^2 = 0$$

$$\sum_{i=1}^{n} \left(x_i y_i - \beta_0 x_i - \beta_1 x_i^2 \right) = 0$$

$$\downarrow$$

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

$$\sum_{i=1}^{n} \left(x_i y_i - (\bar{y} - \beta_1 \bar{x}) x_i - \beta_1 x_i^2 \right) = 0$$

$$\sum_{i=1}^{n} \left(x_i y_i - x_i \bar{y} + \beta_1 \bar{x} x_i - \beta_1 x_i^2 \right) = 0$$

$$\sum_{i=1}^{n} \left(y_i - \bar{y} + \beta_1 \bar{x} - \beta_1 x_i \right) x_i = 0$$

$$\sum_{i=1}^{n} \left[(y_i - \bar{y}) + \beta_1 (\bar{x} - x_i) \right] = 0$$

$$\sum_{i=1}^{n} (y_i - \bar{y}) + \sum_{i=1}^{n} \beta_1 (\bar{x} - x_i) = 0$$

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

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

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

Coefficient =>

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

OLS

Intercept ⇓

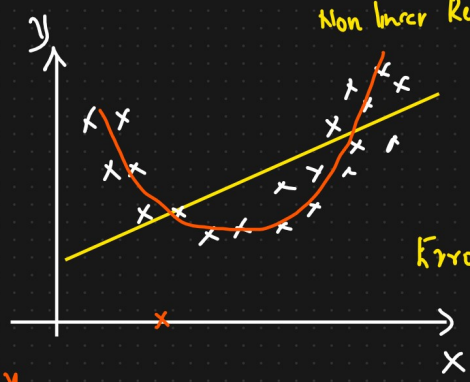
Coefficient ↑

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

x	y	$(y_i - \bar{y}) \div (x_i - \bar{x})$	β_1 (coefficient ↓)	$\beta_0 = (\bar{y} - \beta_1 \bar{x})$ (Intercept ↓)
–	–	–		
–	–	–		
–	–	–		
–	–	–		
$\bar{x}$	$\bar{y}$			

OLS ≈ Linear Regression (SKlearn)

Polynomial Regression



$h_\theta(x) = \beta_0 + \beta_1 x$ — Simple Linear Regression

$h_\theta(x) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3$ — Multiple Linear Regression

Error ↑↑ Error ↓↓ ⟶ Polynomial Regression

Polynomial Degrees

→ Hyperplane degree = 0

Simple Polynomial Regression {1 I/p and 1 o/p feature}

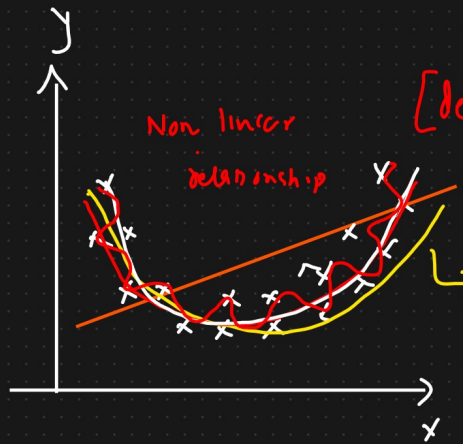
deg = 0

polynomial degree = 0 $h_\theta(x) = \beta_0 * x^0$ ⇒ Constant value

polynomial degree = 1 $\boxed{h_\theta(x) = \beta_0 * x^0 + \beta_1 * x^{(1)}}$ → Simple Linear Regression

polynomial degree = 2 $h_\theta(x) = \beta_0 * x^0 + \beta_1 * x^{(1)} + \beta_2 * x^{(2)}$

polynomial degree = n $\boxed{h_\theta(x) = \beta_0 * x^0 + \beta_1 * x^1 + \beta_2 * x^{(3)} + \beta_3 * x^{(3)} + \dots + \beta_n * x^{(n)}}$



[degree → values]

Degree = 15

↳ degree = 1

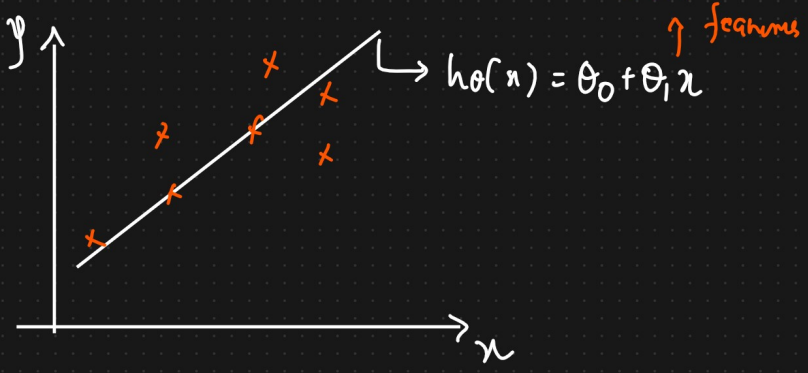
{2 independent feature}

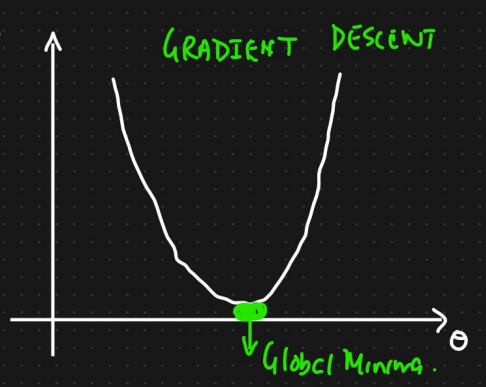
degree = 1 $h_\theta(x) = \beta_0 + \beta_1 x_1 + \beta_2 x_2$

degree = 2 $h_\theta(x) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1^2 + \beta_4 x_2^2$

Ridge Regression, Lasso Regression, Elasticnet Regression

Linear Regression





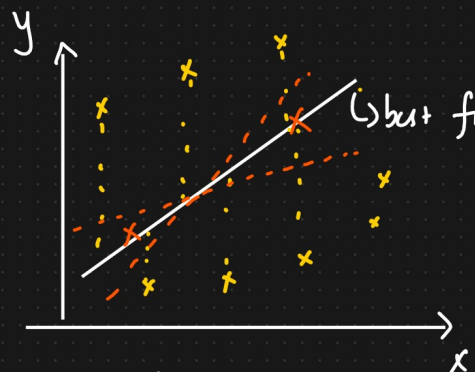
$$\text{Cost fn} = \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta(x)^{(i)} - y^{(i)}\right)^2$$

⇓

Mean Squared Error

① Ridge Regression (L2 Regularization) → Reduce Overfitting

Overfitting



- Traindata → Acc ↑ → low Bias
- Test data → Acc ↓ → High Variance

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

$\lambda = 0$

$\lambda = 1$

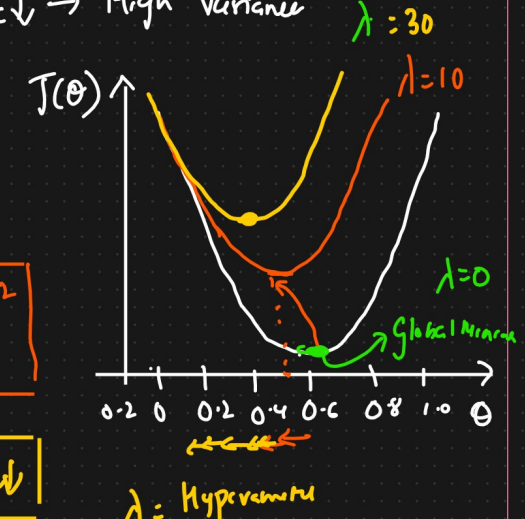
$\lambda = 10$

Hyperparameter

$$\text{Cost fn} = \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta(x)^{(i)} - y^{(i)}\right)^2 + \lambda\sum_{i=1}^{n}(\text{slope})^2$$

$$= 0 + (1)\left[(\theta_1)^2\right]$$

$\lambda \uparrow$ Slope ↓



$\lambda = 1$

> 0

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

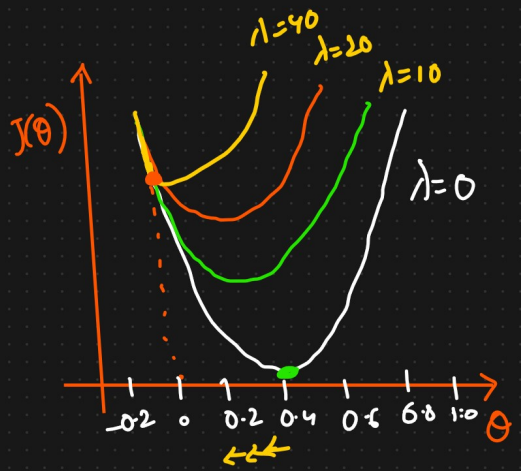
0.24

$$= 0.34 + 0.52x_1 + 0.48x_2 + 0.24x_3$$

$$= 0.34 + 0.40x_1 + 0.38x_2 + \boxed{0.14x_3}$$

② Lasso Regression (L_1 Regularization) → Feature Selection

$$\text{Cost fn} = \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta(x)^{(i)} - y^{(i)}\right) + \boxed{\lambda \sum_{i=1}^{n} |\text{slope}|}$$



$$h_\theta(x) = \theta_0 + \theta_1x_1 + \theta_2x_2 + \theta_3x_3 + \theta_4x_4$$

$$h_\theta(x) = 0.52 + 0.65x_1 + 0.72x_2 + 0.34x_3 + \boxed{0.12x_4}$$

⇓ Lasso Regression

⇓ → 0

$$= 0.52 + 0.51x_1 + 0.60x_2 + 0.14x_3 + \boxed{0x_4}$$

③ ElasticNet Regression

→ ① Reduce Overfitting

→ ② Feature Selection

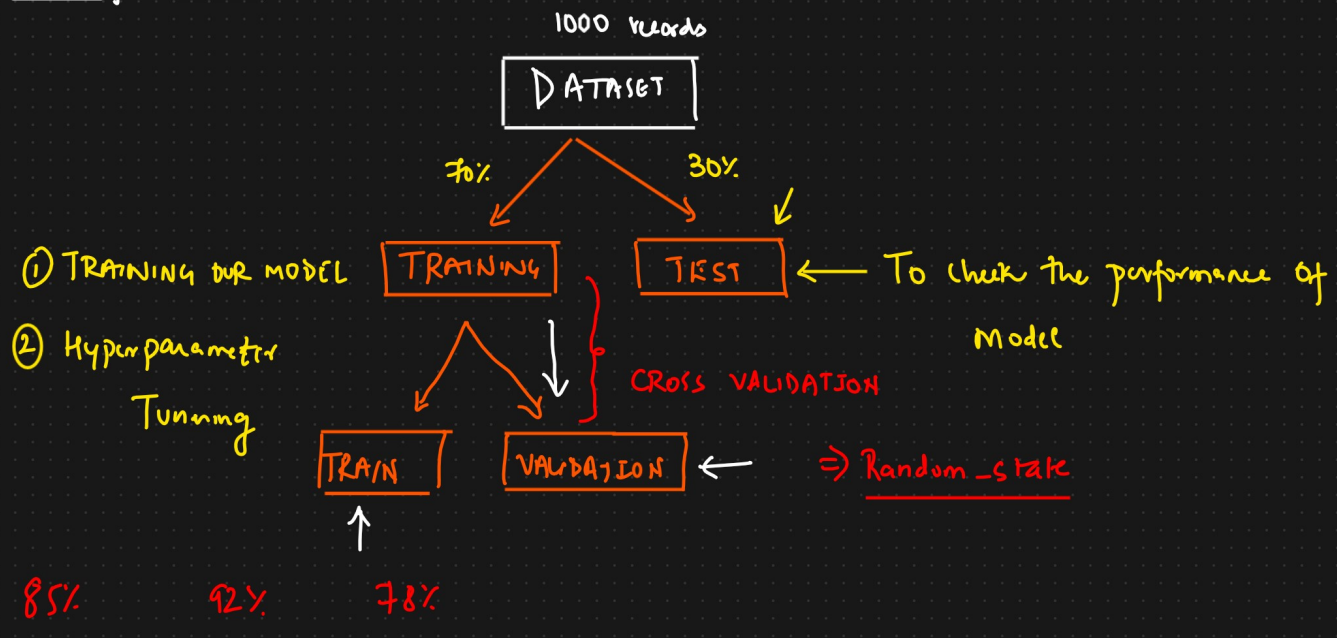
$$\text{Cost fn} = \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta(x)^{(i)} - y^{(i)}\right)^2 + \boxed{\lambda_1 \sum_{i=1}^{m} (\text{slope})^2} + \boxed{\lambda_2 \sum_{i=1}^{m} |\text{slope}|}$$

⇓ Reduce Overfitting

⇓ Feature Selection

{ Hyperparameter Tuning the Linear Regression }

Types of CROSS VALIDATION

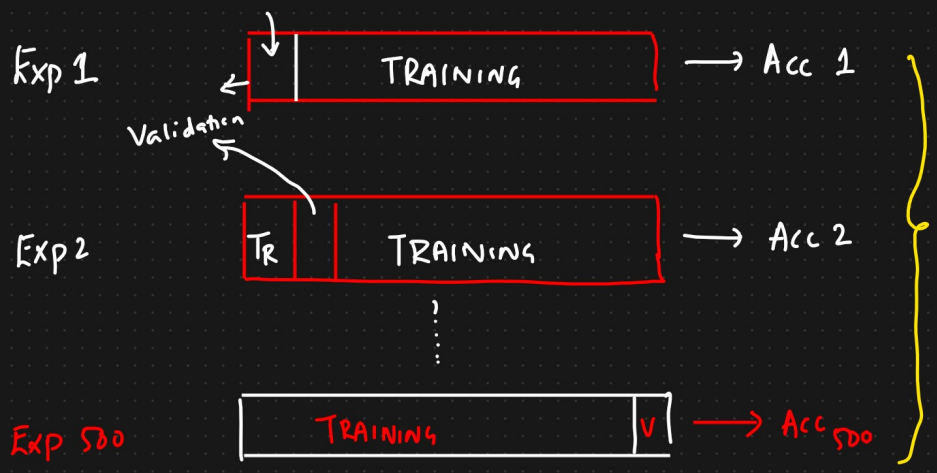


① leave One Out CV (LOOCV)

② leave p out CV

p=10 p=20 p=30

TRAINING → 500 Records ↑↑ Complexity of Training Model

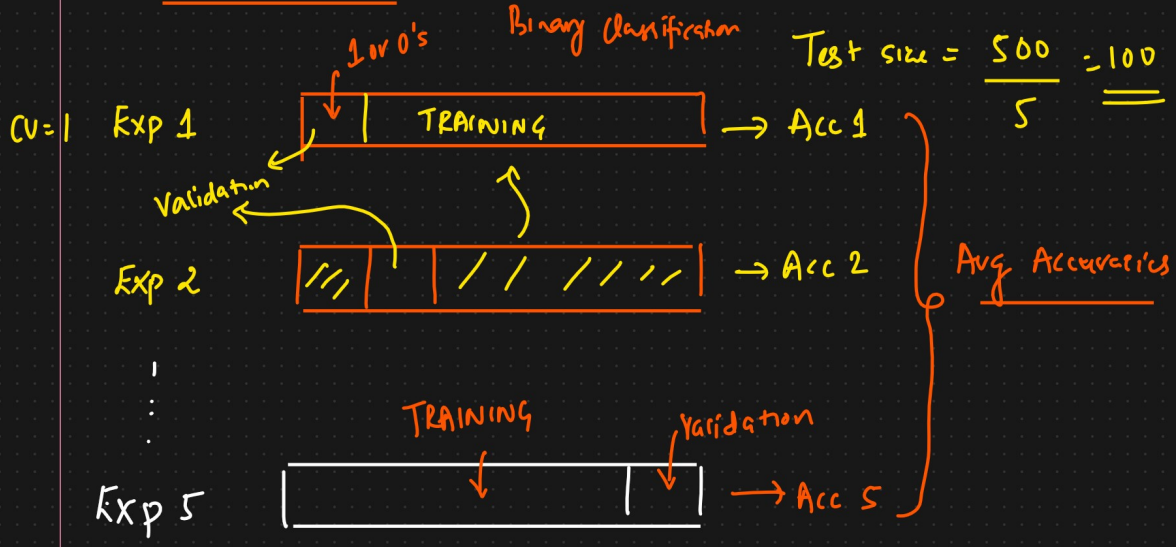


① Overfitting → TRAINING ↑↑ Acc → New Test Data → Acc ↓↓

Validation Acc ↓↓

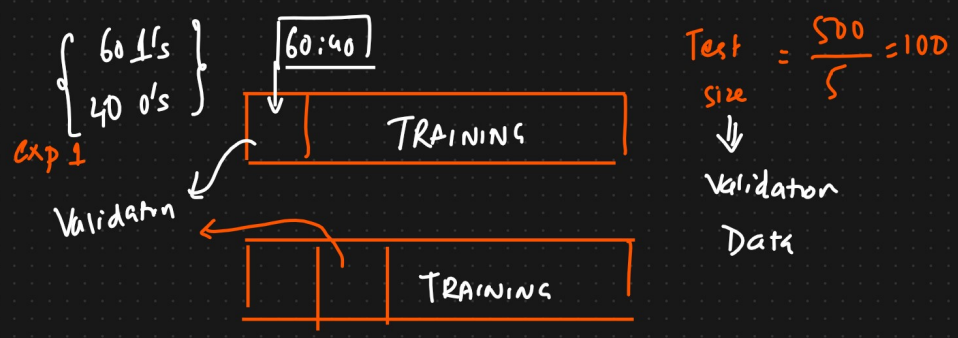
③ K Fold CV

K=5 n=500



④ Stratified K Fold CV

K=5



⑤ Time Series CV

Reviews

Product Sentiment Analysis

Time

JAN → DEC

TRAINING | Validation

DAY 1 DAY 2 DAY 3 DAY 4 . . . DAY N

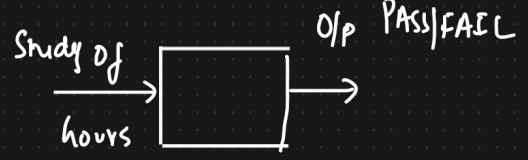
Time Series Application

Logistic Regression (Binary classification) ←

Dataset

Pass/Fail

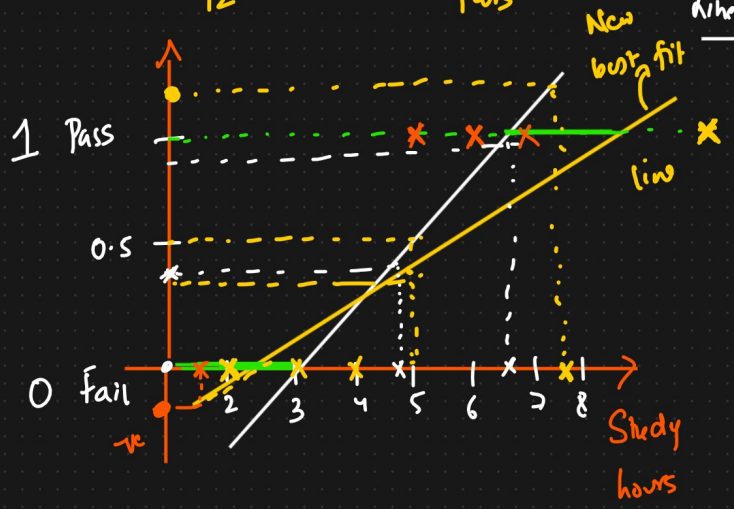
Study hours	O/p {Binary categories}
2	Fail
3	Fail
4	Fail
5	Pass
6	Pass
7	Pass
12	Pass



Logistic Regression

0 to 1

Linear Regression



$\leq 0.5 \Rightarrow 0$

$> 0.5 \Rightarrow 1$

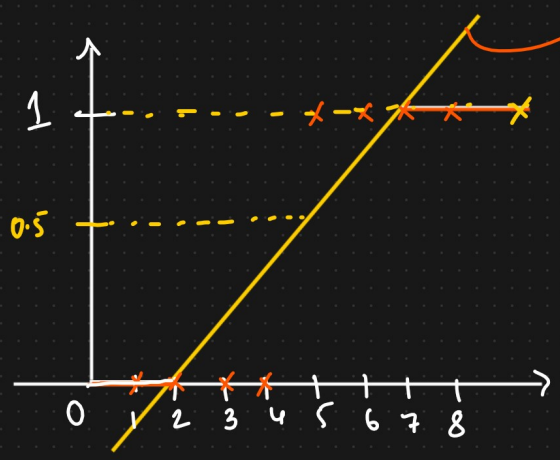
> 1 and < 0

Why we cannot use Linear Regression for Classification?

① Outlier {Best fit line Change}

② > 1 and < 0 {Squash line}

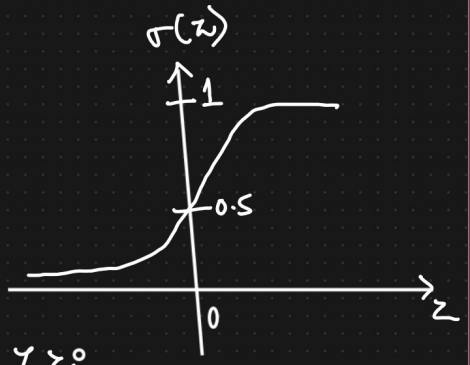
How Logistic Regression Solves Classification Problem



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

⇓

Best fit line



$z \geq 0$

$\sigma(z) \geq 0.5$

Sigmoid Activation

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

$z = \theta_0 + \theta_1 x_1$

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

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

$$= \sigma(z)$$

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

⇒ Logistic Regression hypothesis

$z = \theta_0 + \theta_1 x_1$

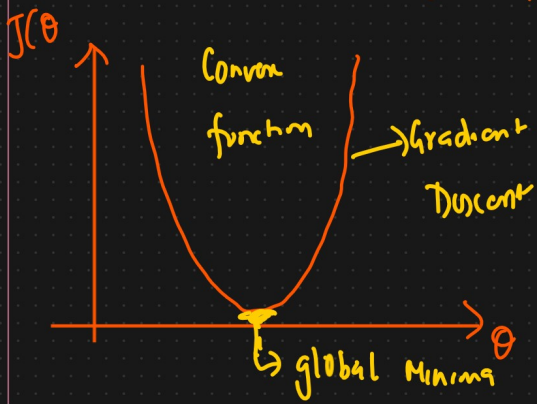
Linear Regression Cost function

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

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

⇓

Convex function



Logistic Regression Cost function

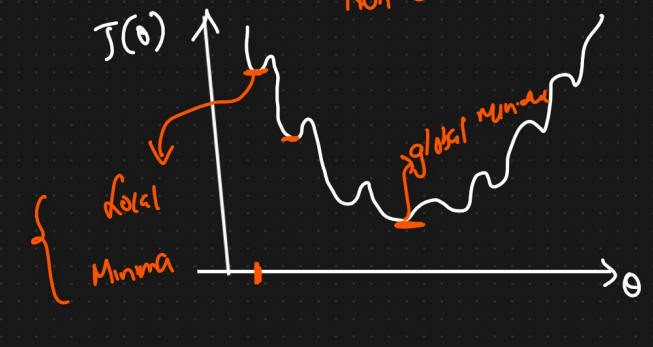
$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x)^{(i)} - y^{(i)}\right)^2$$

Non Convex function

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

Sigmoid

Non Convex



$$J(\theta_0,\theta_1) = \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta(x)^{(i)} - y^{(i)}\right)^2 \qquad h_\theta(x)^{(i)} = \frac{1}{1+e^{-z}} \quad z = \theta_0 + \theta_1 x_1$$

$\Downarrow$ Let's Denote $\quad \text{Cost}\left(h_\theta(x)^{(i)}, y^{(i)}\right)$

$\Downarrow$

{ log loss }

$$\text{Cost}\left(h_\theta(x)^{(i)}, y^{(i)}\right) = \begin{cases} -\log\left(h_\theta(x)\right) & \text{if } y = 1 \\ -\log\left(1-h_\theta(x)\right) & \text{if } y = 0 \end{cases}$$

$\Downarrow$ convex function

$$\text{Cost}\left(h_\theta(x)^{(i)}, y^{(i)}\right) = -y\log\left(h_\theta(x)\right) - (1-y)\log\left(1-h_\theta(x)\right)$$

$$J(\theta_0,\theta_1) = -\frac{1}{2m}\sum_{i=1}^{m}\left(y^{(i)}\log\left(h_\theta(x)^{(i)}\right) - (1-y^{(i)})\log\left(1-h_\theta(x)^{(i)}\right)\right)$$

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

θ_0 & θ_1

Convergence Algorithm

Repeat

{

$j = 0$ and 1

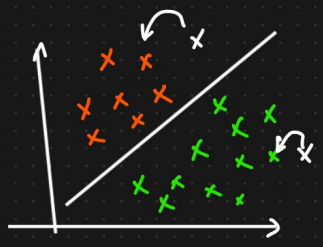
$$\theta_j := \theta_j - \alpha\frac{\partial}{\partial\theta_j}J(\theta_0,\theta_1)$$

}

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



Logistic Regression

R squared

Adjusted R squared

Dataset

x_1	x_2	o/p y	pred by model $\hat{y}$
-	-	0	1
-	-	1	1
-	-	0	0
-	-	1	1
-	-	1	1
-	-	0	1
-	-	1	0
-	-		

1. Confusion Matrix

Actual values \ Predicted values	1	0
1	3	2
0	1	1

Predicted values \ Actual	1	0
1	TP	FP
0	FN	TN

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

$$= \frac{3+1}{3+2+1+1}$$

$$= \frac{4}{7}$$

2. Dataset

Binary classification

- 1000 datapoints
  - 900 → 1
  - 100 → 0

Imbalanced Dataset

⇓

90% Accuracy

④ Precision = $\frac{TP}{TP+FP}$ ↓ } Out of all the actual value how many are correctly predicted

Predicted \ Actual	1	0
1	TP	FP
0	FN	TN

② Recall = $\frac{TP}{TP+FN}$ ↓ ⇒ Out of all the predicted value how many are correctly predicted

Usecase ①

Spam classification

Mail → Spam
Model → Spam
} Good

Spam \ Actual	1	0
1	TP	FP ✓ ↓↓
0	FN	TN

Mail → Not Spam
Model → Spam
} Blunder

↓↓

Precision = $\frac{TP}{TP+FP}$

Usecase 2

To predict whether person has diabetes or not

✓ Truth → diabetes
✓ Model → Doesn't diabetes
} Blunder

	Diabetes	No Diabetes
Diab	TP	FP
No Diab	FN ↓↓	TN

Recall = $\frac{TP}{TP+FN}$ ↓↓

Usecase of diseases

Truth → diabetes
Model → "
} good

Truth → Not diabetes
Model → Diabetes
} ⇒ 2nd opinion ⇓ check

Assignment

(*) Tommorow the stock market will crash or not

Reducing FP↓ or FN↓

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

① If FP & FN are both important

$\beta = 1$

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

⇒ Harmonic Mean

② If FP is more important FN

$\beta = 0.5$

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

③ If FN >> FP

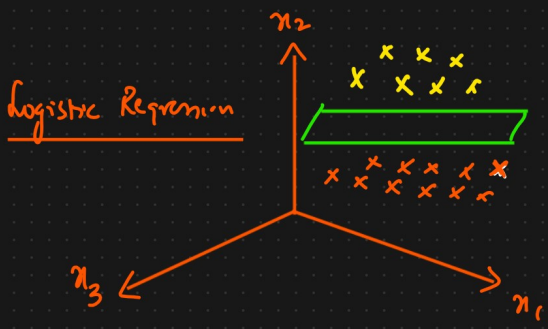
$\beta = 2$

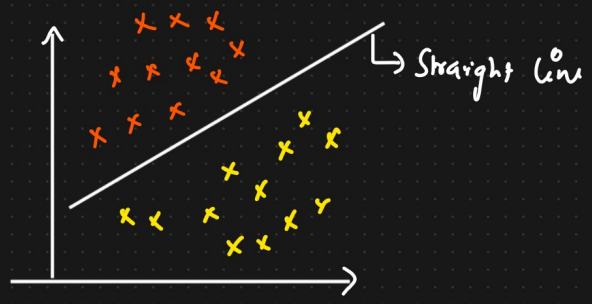
$$F\,2 \text{ Score} = (1+4)\frac{P \times R}{P+R}$$

Support Vector Machines ML Algorithms.

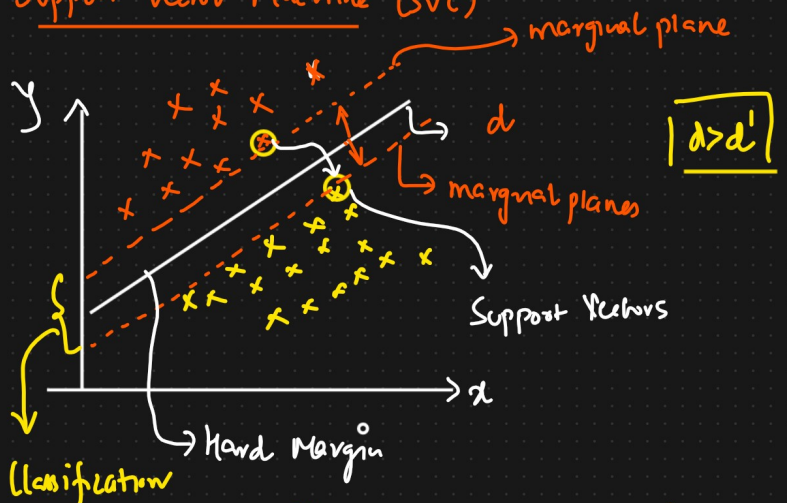
① SVC (Support Vector Classifier)

② SVR (Support Vector Regressor)

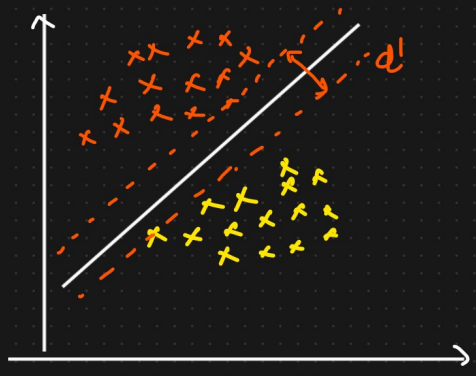


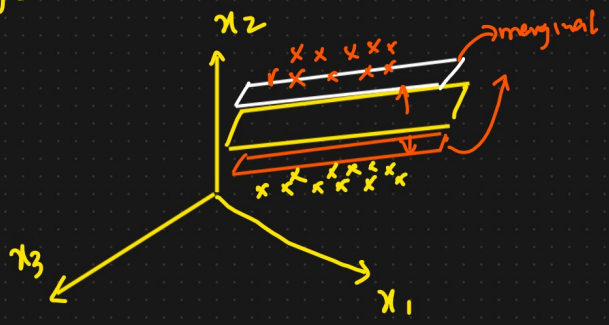


① Support Vector Machine (SVC)

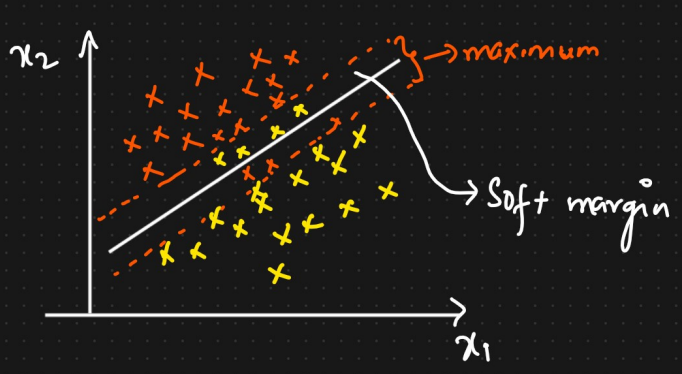


$d > d'$

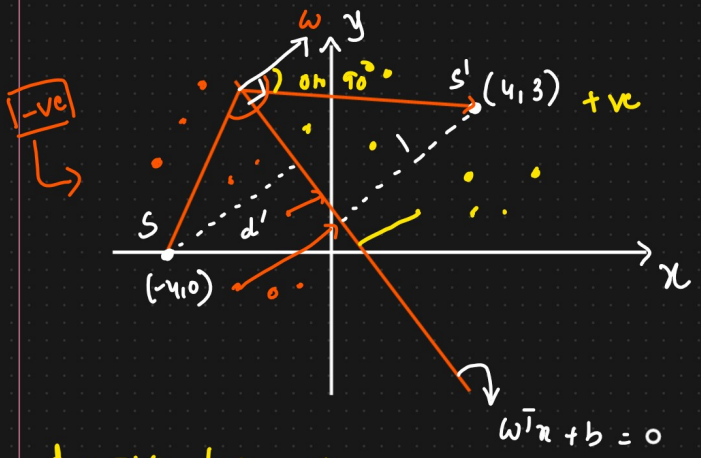




Soft Margin And Hard Margin In SVM



⑧ Support Vector Machines (SVC) Maths Intuition



$ax + by + c = 0$

$\Downarrow$

$w_1 x_1 + w_2 x_2 + b = 0$

$$\boxed{w^T x + b = 0}$$

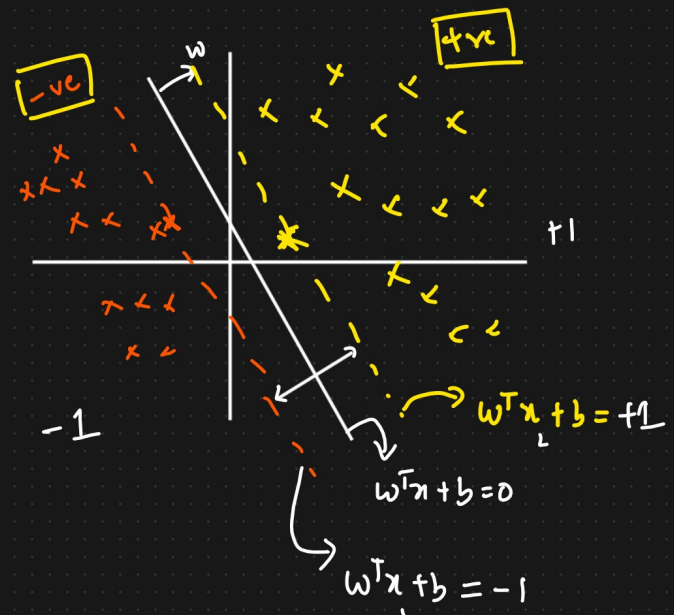
$\Downarrow$

$b = 0$

$$\boxed{w^T x = 0}$$

d = -ve below plane

d = +ve above plane



$w^T x_1 + b = 1$

$w^T x_2 + b = -1$

(-) (-) (+)

$$\frac{w^T (x_1 - x_2)}{\|w\|} = \frac{+2}{\|w\|}$$

Unit vector {Magnitude of the vector is 1}

Cost function

Maximize $\frac{2}{||w||}$ $\Rightarrow$ Distance between marginal plane

w, b

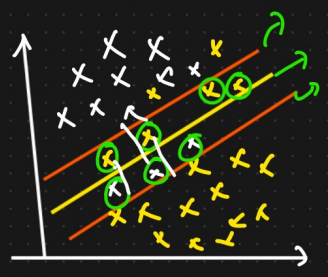
Constraint such that $y_i \begin{cases} +1 & w^T x + b \geq 1 \\ -1 & w^T x + b \leq -1 \end{cases}$ ⇗ correctly classified point

For all correct points

Constraint $\longrightarrow$ $\boxed{y_i * (w^T x + b) \geq 1}$

$C_i = 6$ ✓

Maximize $\frac{2}{||w||}$ $\Rightarrow$ $\boxed{\underset{(w,b)}{\text{Min}} \frac{||w||}{2}}$



Cost function of SVM (SVC)

hyperparameter

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

$\Downarrow$

Soft margin

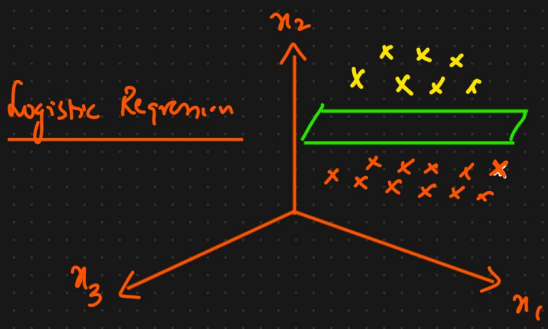
{How many points we want to avoid misclassification}

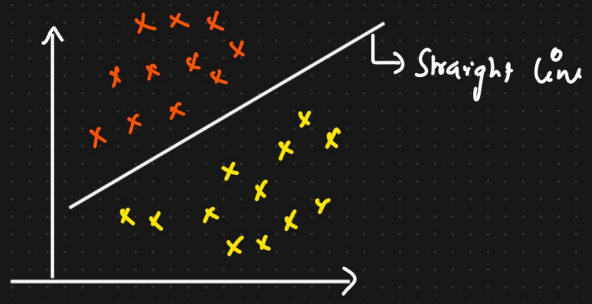
{Summation of the distance of the incorrect data points from the marginal plane}

Support Vector Machines ML Algorithms.

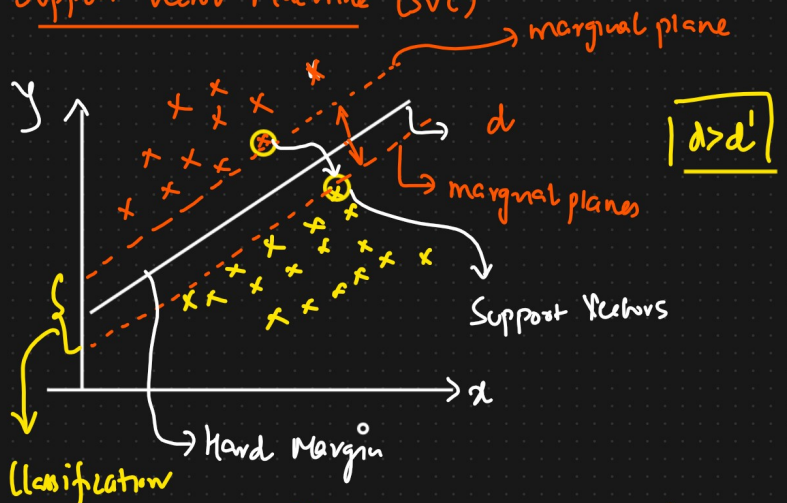
① SVC (Support Vector Classifier)

② SVR (Support Vector Regressor)

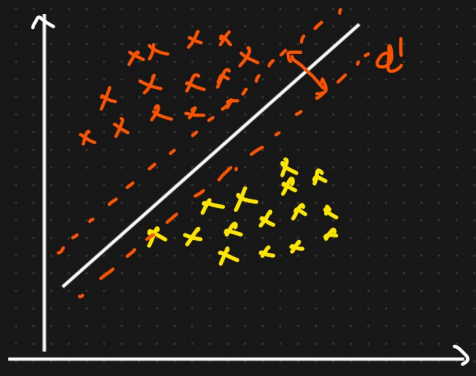


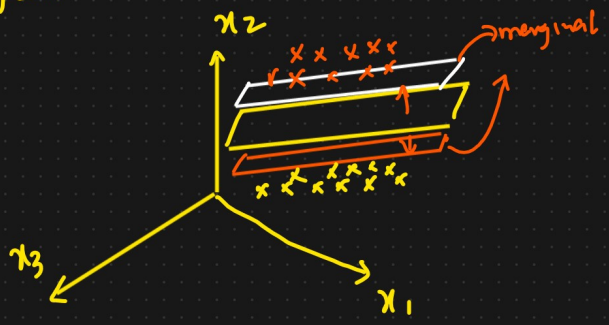


① Support Vector Machine (SVC)

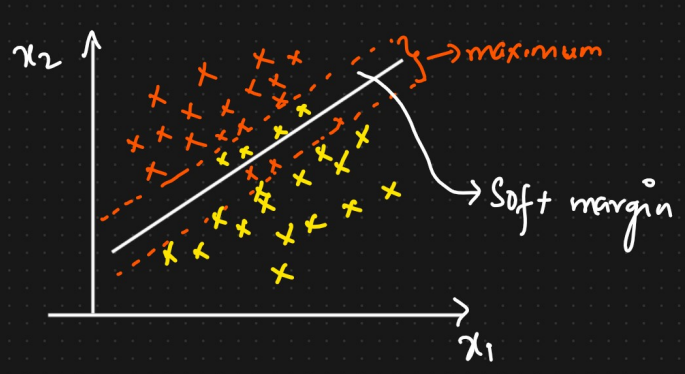


$d > d'$

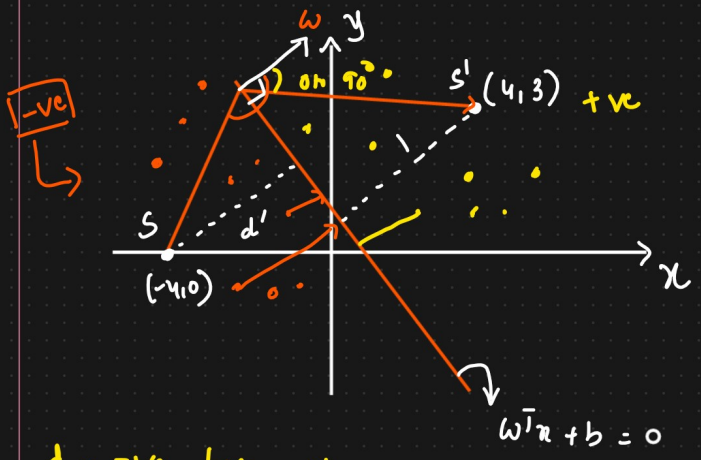




Soft Margin And Hard Margin In SVM



⑧ Support Vector Machines (SVC) Maths Intuition



$ax + by + c = 0$

$\Downarrow$

$w_1 x_1 + w_2 x_2 + b = 0$

$$\boxed{w^T x + b = 0}$$

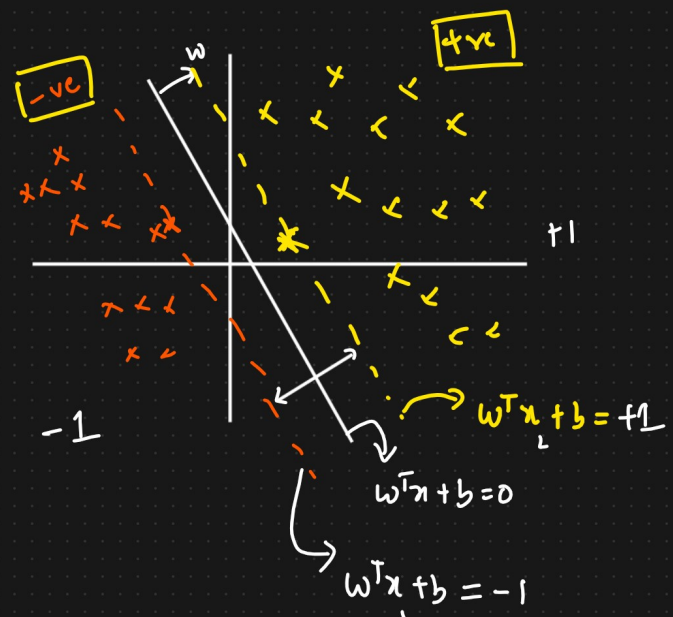
$\Downarrow$

$b = 0$

$$\boxed{w^T x = 0}$$

d = -ve below plane

d = +ve above plane



$w^T x_1 + b = 1$

$w^T x_2 + b = -1$

(-) (-) (+)

$$\frac{w^T (x_1 - x_2)}{\|w\|} = \frac{+2}{\|w\|}$$

Unit vector {Magnitude of the vector is 1}

Cost function

Maximize $\frac{2}{||w||}$ w,b $\Rightarrow$ Distance between marginal plane

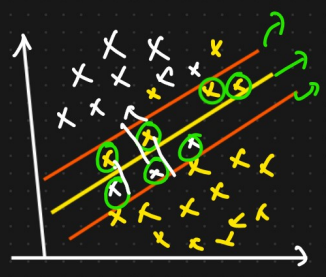
Constraint such that $y_i \begin{cases} +1 & w^T x + b \geq 1 \\ -1 & w^T x + b \leq -1 \end{cases}$ ← correctly classified point

For all correct points ← predicted points

Constraint $\longrightarrow$ $\boxed{y_i * (w^T x + b) \geq 1}$

$\boxed{C_i = 6}$ ✓

Maximize $\frac{2}{||w||}$ w,b $\Rightarrow$ $\boxed{\text{Min}_{(w,b)} \frac{||w||}{2}}$



Cost function of SVM (svc)

hyperparameter

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

C_i → {How many points we want to avoid misclassification}

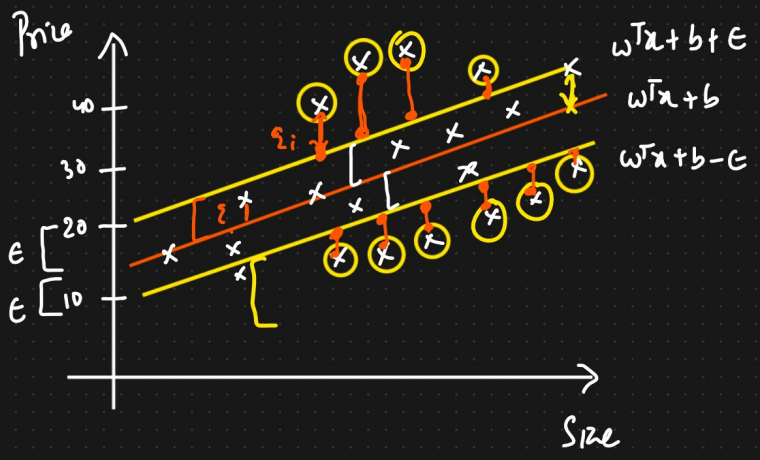
ξ_i → {summation of the distance of the incorrect data points from the marginal plane}

⇓

Soft margin

Support Vector Regression

ϵ : Marginal Error



Cost function

$$\underset{w,b}{\text{Min}} \quad \frac{\|w\|}{2} + \left[C \sum_{i=1}^{n} \xi_i \right]$$

Hyperparameter

→ Hinge loss

Relationship

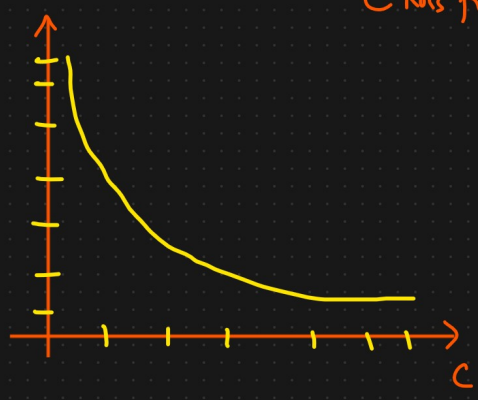
- C ↑↑
- Loss function ↓↓

Constraint :

$$|y_i - w_i x_i| \leq \epsilon + \xi$$

⇓

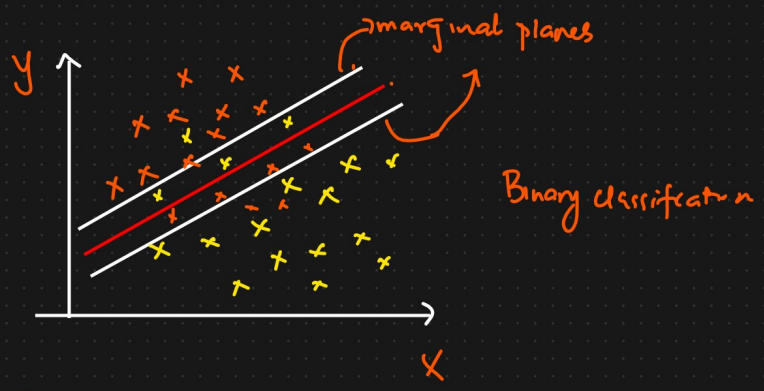
Loss function



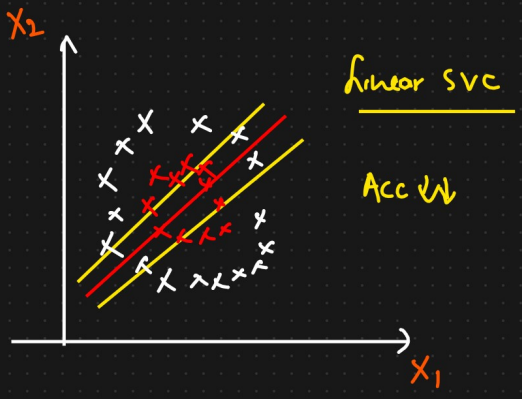
ϵ → margin Error

ξ_i → Error above the margin

SVM KERNELS



Binary classification



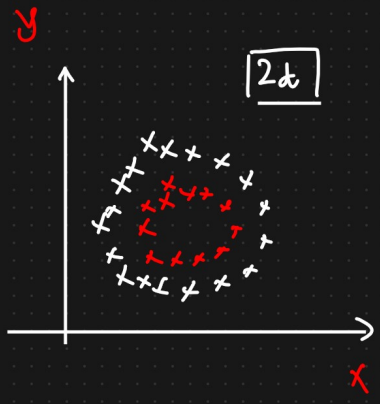
linear SVC

Acc ↓↓

linear SVC

⇓

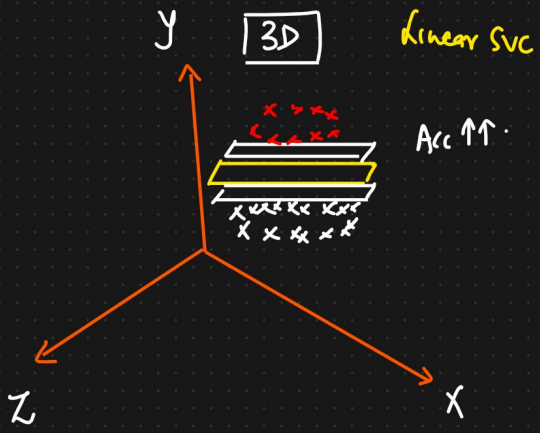
SVM Kernels



⇒ Transformations ⇒

⇓

Mathematical formula



linear SVC

Acc ↑↑.

Dataset : 1d

SVM Kernel

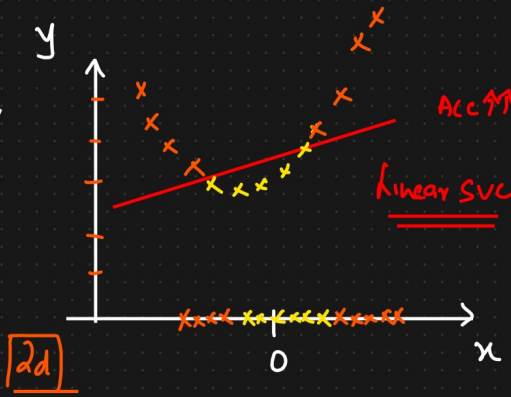
⇒ Transformation

$y = x^2$

SVC

linear SVC

1d



2d

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

Naive Baye's Algorithm (Classification)

1. Probability
2. Baye's Theorem

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 removing a orange marble and then a yellow marble

O O O
O O

↳ $P(O) = \frac{3}{5}$ → 1st Event → Orange marble has been Removed

O O
O O

→ $P(Y/O) = \frac{2}{4} = \frac{1}{2}$ ⇒ 2nd Event ⇒ Removed the Yellow Marble

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

(1st, 2nd)

$= 3/5 * \frac{1}{2} = \frac{3}{10}$

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

Bayes Theorem

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

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

$$P(A/B) = \frac{P(A) * P(B/A)}{P(B)} \Rightarrow \text{Bayes Theorem.}$$

$P(A/B)$: 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.

$$P(A/B) = \frac{P(A) * P(B/A)}{P(B)} \Rightarrow \text{Bayes Theorem.}$$

I/P features

x_1	x_2	x_3	↓ Dependent y
–	–	–	Yes
–	–	–	No
–	–	–	Yes
–	–	–	No

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

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

$$= \frac{P(y) * P(x_1/y) * P(x_2/y) * P(x_3/y)}{P(x_1) * P(x_2) * P(x_3)}$$

I/P features

Dependent

x_1	x_2	x_3	y
–	–	–	Yes
–	–	–	No
–	–	–	Yes
–	–	–	No

[0.60, 0.40]

⇓

Yes

New test data

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

$P(x_1) * Pr(x_2) * Pr(x_3)$ ⇒ Constant

⇓

Yes

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

$P(x_1) * Pr(x_2) * Pr(x_3)$ ⇒ Constant

Lets Solve this Problem

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

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

Temperature

→ Test (Sunny, Hot) → O/P

PLAY (Y/N)

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

		P(Yes)	P(No)
Yes	9	9/14	5/14
No	5		

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

$$= 9/14 * \frac{2}{9} * \frac{2}{9}$$

$$= \frac{2}{63} = 0.031$$

$$P(No/(Sunny, Hot)) = P(No) * Pr(Sunny/No) * Pr(Hot/No)$$

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

$$= \frac{3}{35} = 0.085$$

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

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

New Test =>

Outlook	Temperature	O/P
Sunny	Hot	

73% => They will not play Tennis

27% => They will play Tennis.

⇓

0 => Person is not going to playing Tennis

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 need to use Bernoulli Naive Baye Algorithm.

Bernoulli → 0, 1

Dataset

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

2. Multinomial Naive Bayes ⇒ I/p = Text

Dataset : Spam Classification

I/P feature → Email Body	O/P Spam/Not Spam
You have Million $$ lottery	Spam
KRISH You have done Good JOB	HAM

↓

Numerical Values ⇒ Natural Language Procesing

↓

Vectors

① BOW

② Tf-Idf

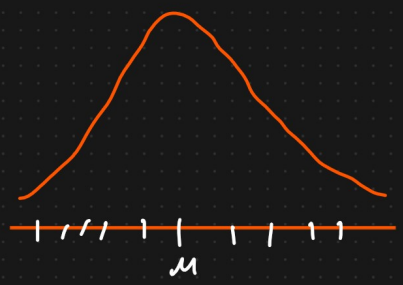
③ Word2vec

③ Gaussian Naive Bayes

If the features are following Gaussian Distbution, then we use Gaussian Naive Bayes

DATASET → CONTINUOUS

IRIS Dataset



Age	Height	Weight	Yes/No
25	170	78	
38	160	75	
22	150	60	
24	170	35	

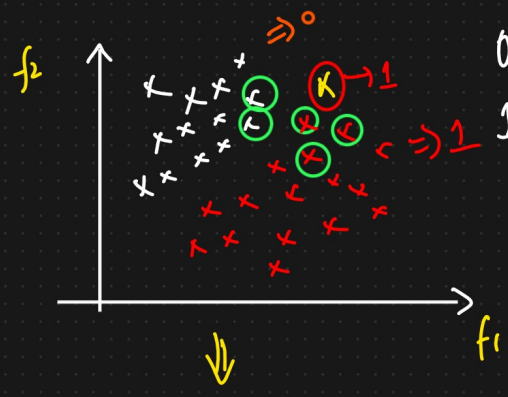
K Nearest Neighbour (KNN)

① Classification

② Regression

① Classification

K=5



TRAINING.

f1	f2	y [Binary Categories]
—	—	0
		1

① We have to initialize the K value K=5

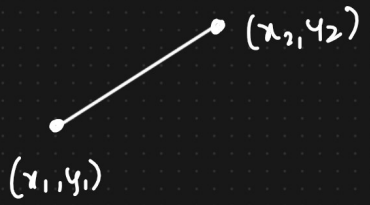
K>0 . . . ∞

K=1, 2, 3, 4, 5. . . . . ⇒ Hyperparameter

② Find the K Nearest Neighbour for The Test Data.

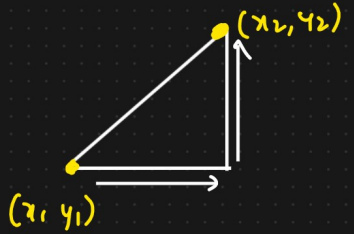
③ From those K=5 how many neighbour belong to 0 Category and 1 Category

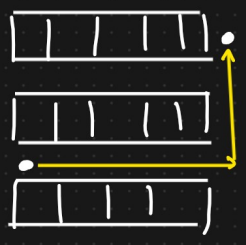
① Euclidian Distance



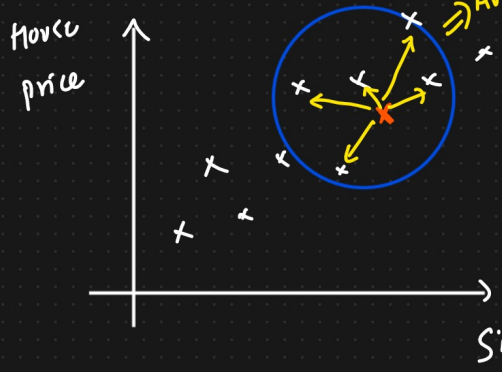
$$\text{Distance} = \sqrt{(x_2-x_1)^2+(y_2-y_1)^2}$$

② Manhattan Distance



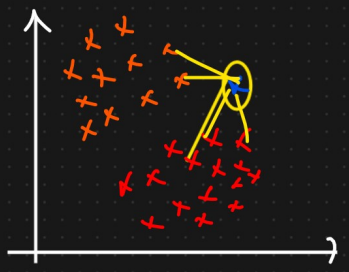


② Regression



Median

⇒ Average of all these points to find out the o/p



Time complexity

O(n) ⇒ Million of datapoint

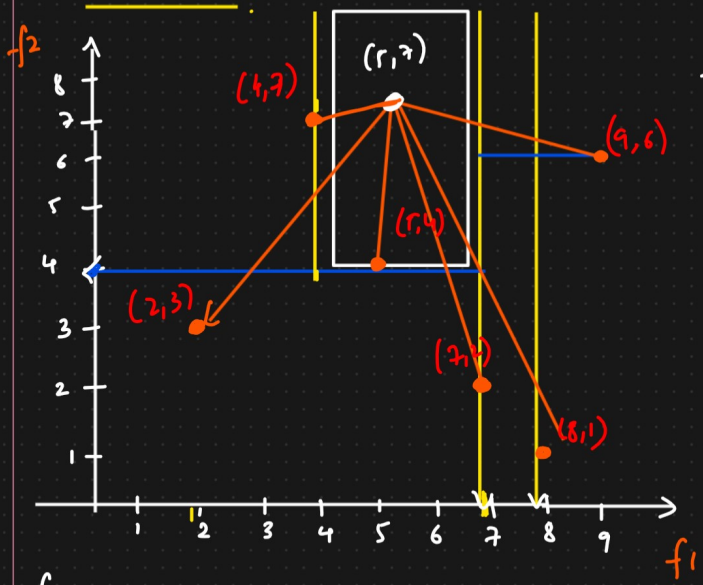
{ ① KD Tree
② Ball Tree } ⇒ Optimize

⇓

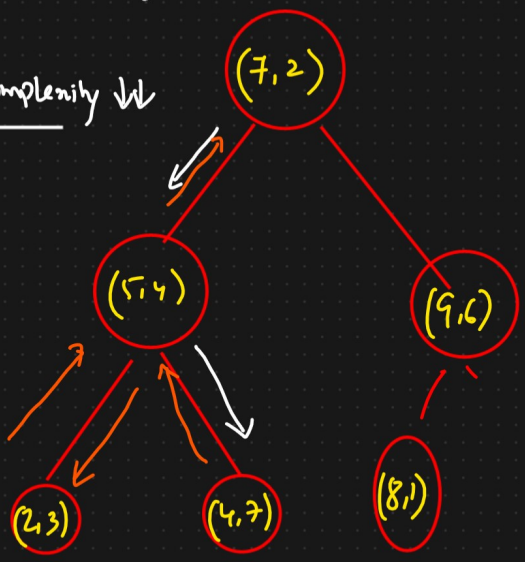
Binary Tree

Variants of KNN

KD Tree



Binary Tree [KD Tree]



Time Complexity ↓↓

Back Tracking

f1

2, 4, 5, 7, 8, 9

$\frac{5+7}{2} = \frac{12}{2} = 6.5$

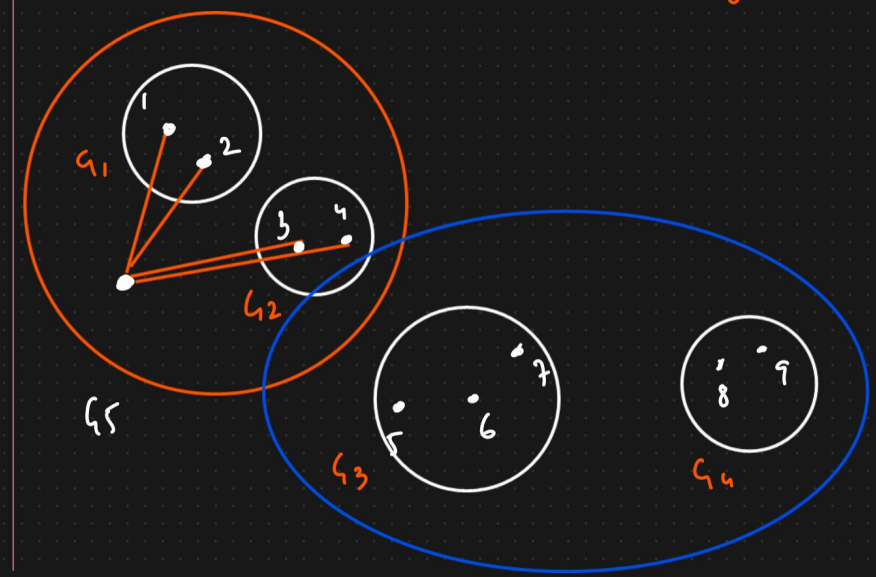
f2

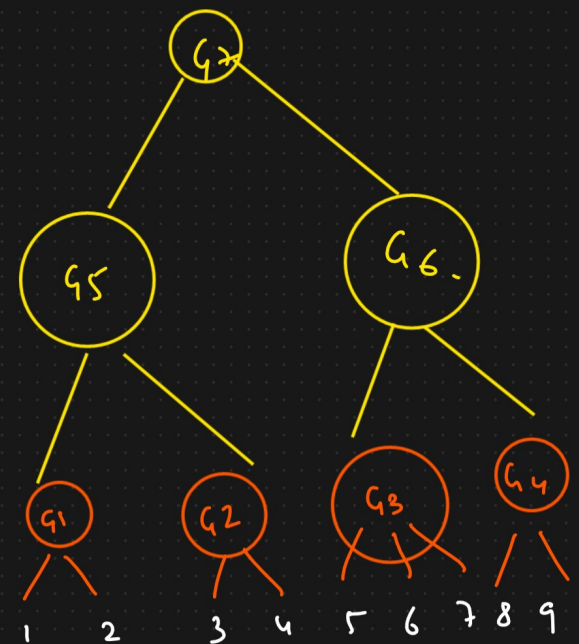
1, 2, 3, 4, 6, 7

f1	f2
7	2
5	4
9	6
2	3
4	7
8	1
Median	Median

② Ball Tree

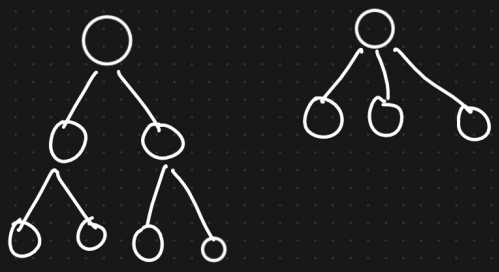
Time Complexity ↓↓





Decision Tree Classifier

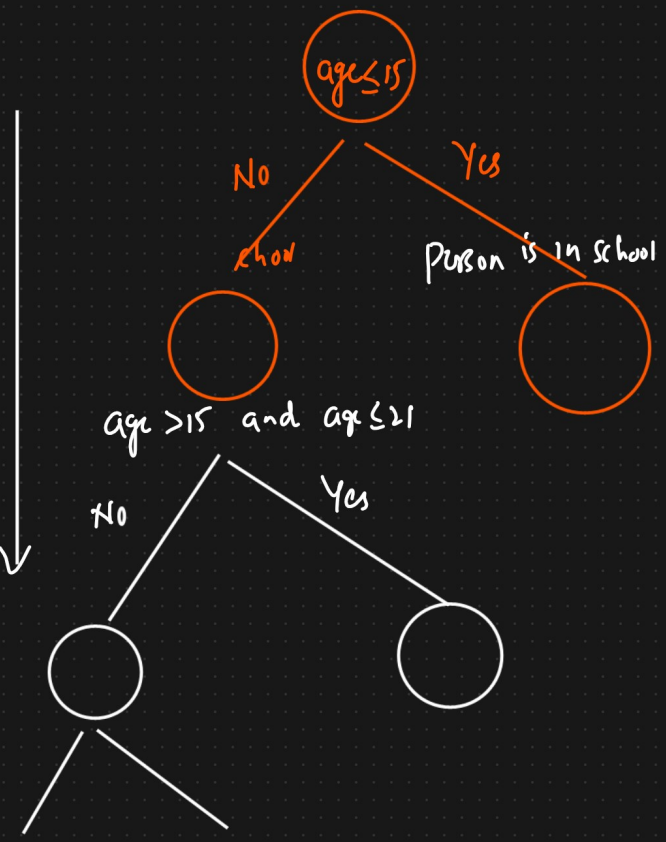
Decision Tree Classifier → ID3
→ CART ✓



a) Entropy and Gini Index → Purity Split

b) Information Gain → features to select for DT construction

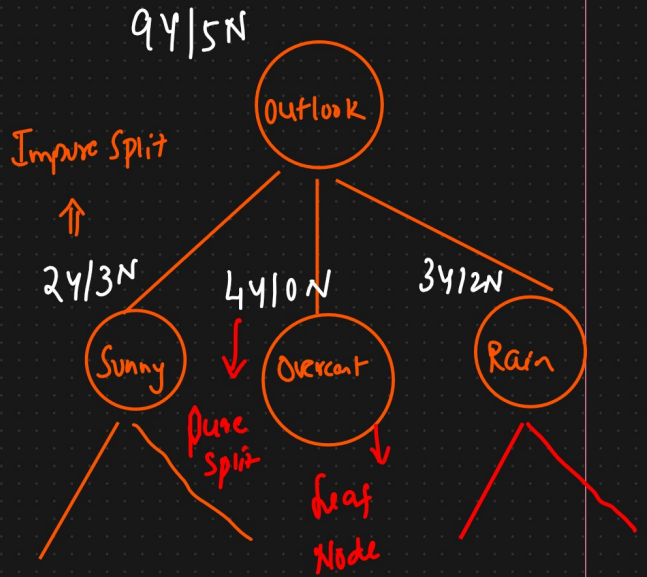
```
age = 14
if (age ≤ 15):
    Print ("The person is in School")
elif (age > 15 and age ≤ 21):
    Print ("The person may be college")
else:
    Print ("The person has passed")
```



Dataset

Binary classification

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 → Pure or Impure Split

↳ Entropy

↳ Gini Impurity

② What feature you need select for Splitting → Information Gain

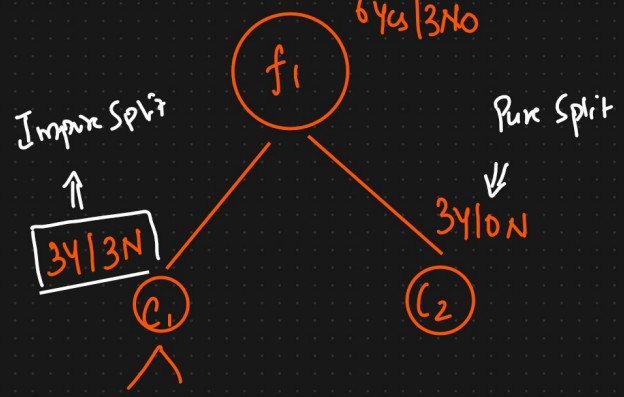
1

0

{Binary classification}

1) Entropy

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

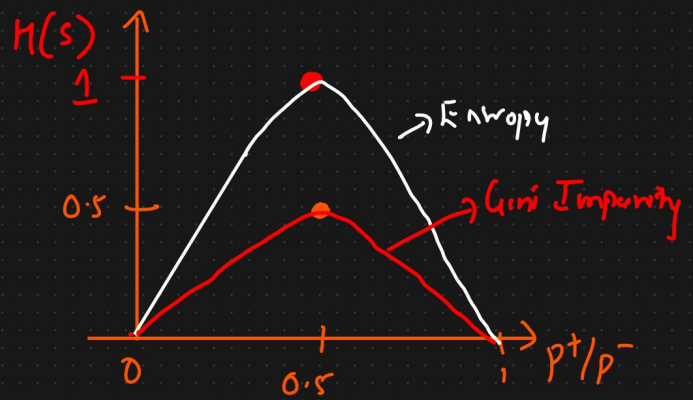


$$H(C_1) = -p_+ \log_2 p_+ - p_- \log_2 p_-$$

$$= -\frac{3}{6} \log_2 3/6 - \frac{3}{6} \log_2 3/6$$

2) Gini Impurity

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



$= 1 \Rightarrow$ Impure Split

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

$= -1\log_2 1 \Rightarrow 0 \Rightarrow$ Pure Split

② Gini Impurity

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

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

$= 1 - \left(\left(\frac{1}{2}\right)^2 + \left(\frac{1}{2}\right)^2\right)$

$= 0.5 \Rightarrow$ Impure Split

3Y/0N

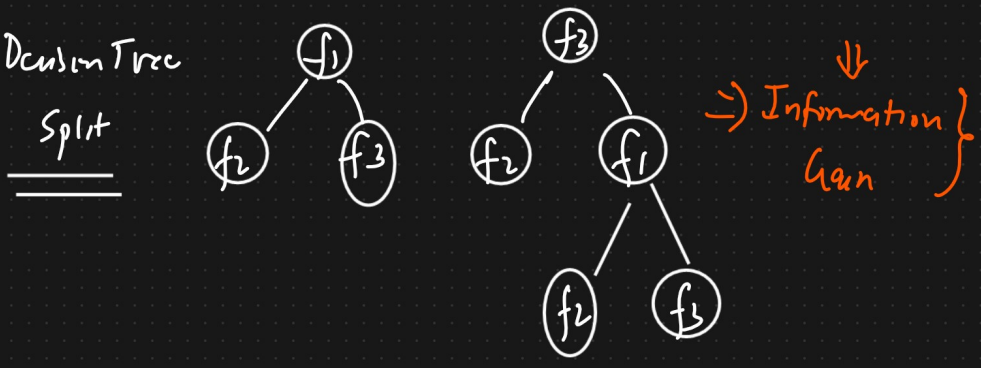
$= 1 - \left(\left(\frac{3}{3}\right)^2\right)$

$= 1 - 1$

$= 0 \Rightarrow$ Pure Split

f1 f2 f3

Decision Tree Split

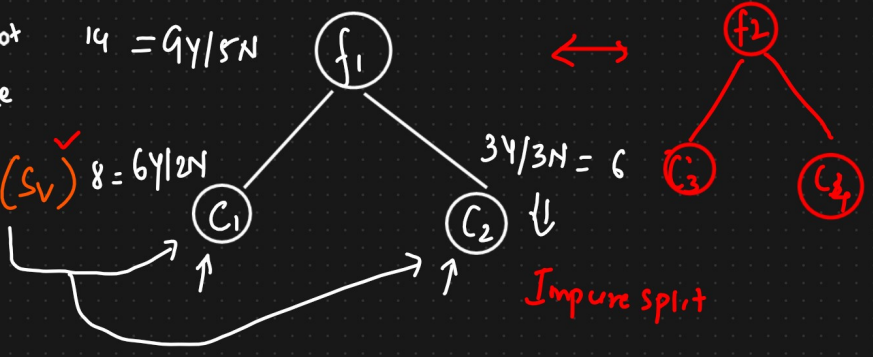


⇓ ⇒ Information Gain

Information Gain

f1 f2 f3 O/P

14 = 9Y/5N



$$Gain(S, f_1) = \boxed{H(S)} - \sum_{v \in val} \frac{|S_v|}{|S|} H(S_v)$$

↑ Entropy of the root node

↑ Root Node

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

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

≈ 0.94

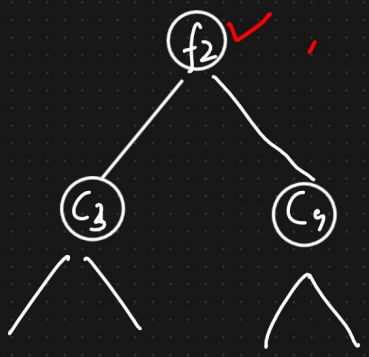
$H(C_1) = -\frac{6}{8}\log_2\frac{6}{8} - \frac{2}{8}\log_2\frac{2}{8}$

$H(C_1) = 0.81$

$H(C_2) = 1$

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

$Gain(S, f_1) = 0.049$



$Gain(S, f_2) = 0.051 > Gain(S, f_1) = 0.049$

Gain

Information is Basically calculated.

Entropy Vs Gini Impurity

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

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

O/P = 3 categories

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

Whenever dataset is small → Entropy

large → Gini Impurity

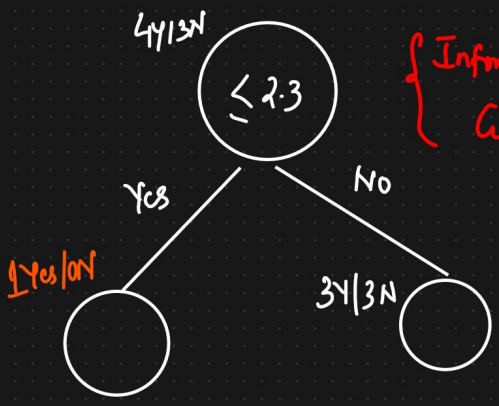
Decision Tree Split for Numerical Feature.

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

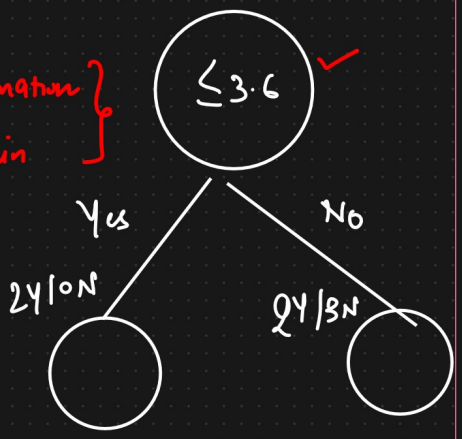
① Sort the feature value

f1	O/P
2.3	Yes
3.6	Yes
4	No
5.2	No
6.7	Yes
8.9	No
10.5	Yes

① Threshold = 2.3



② Threshold = 3.6



{ Information Gain }

Millions of records

Time Complexity ↑↑

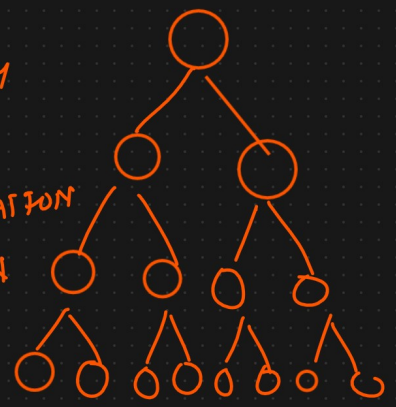
Decision Tree Regression

Dataset

Exp	Career Gap	Salary
2	Yes	40K
2.5	Yes	42K
3	No	52K
4	No	60K
4.5	Yes	56K

op

Entropy
GINI
INFORMATION GAIN



CART

[40K, 42K, 52K, 60K, 56K]

50K

60.8

≤ 2

Yes → 40K

No → 42, 52, 60, 50

V(C1) = 100

V(C2) = 51

[40K, 42K, 52K, 60K, 56K]

60.8

≤ 2.5

40, 42

52, 60, 56

Decision Tree Regressor

Var(C1) = $\frac{1}{2}\left[(40-50)^2 + (42-50)^2\right]$

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

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

Var(child2) = $\frac{1}{3}\left[4 + 100 + 36\right]$

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

Variance Reduction (Regression Problem)

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

Mean Square Error

→ Average

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

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

$= 60.8$

Variance Reduction

$= 60.8 - \left[\frac{2}{5} \times 82 + \frac{3}{5} \times 46.66\right]$

$= 60.8 - \left[32.8 + 27.996\right]$

$= +0.004$

Variance (C1) $= \frac{1}{n}\sum_{i=1}^{n}(y-\bar{y})^2$

$$= \frac{1}{1} (40-50)^2$$

$$= 100$$

$$\text{Variance (c2)} = \frac{1}{n} \sum_{i=1}^{n} (y-\bar{y})^2$$

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

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

$$= 51$$

Variance Reduction ↓

$$= \text{Var(Root)} - \sum w_i \text{Var(child)}$$

$$= 60.8 - \left[\frac{1}{5} \times \overset{20}{\cancel{100}} + \frac{4}{5} \times 51 \right]$$

$$= 60.8 - 20 - 40.8$$

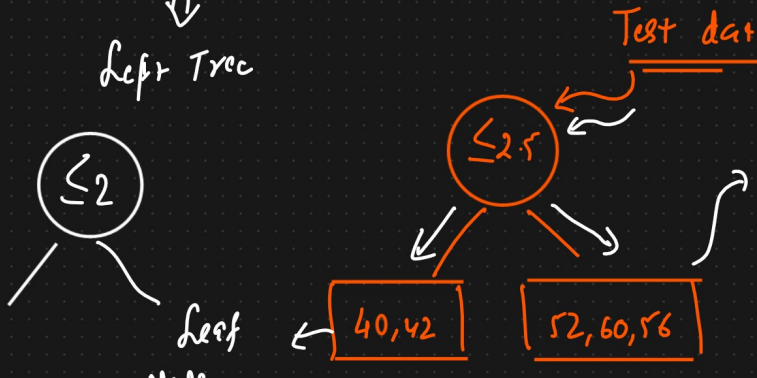
Variance Reduction = 0

0 0.004

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

⇓

Left Tree



Test data

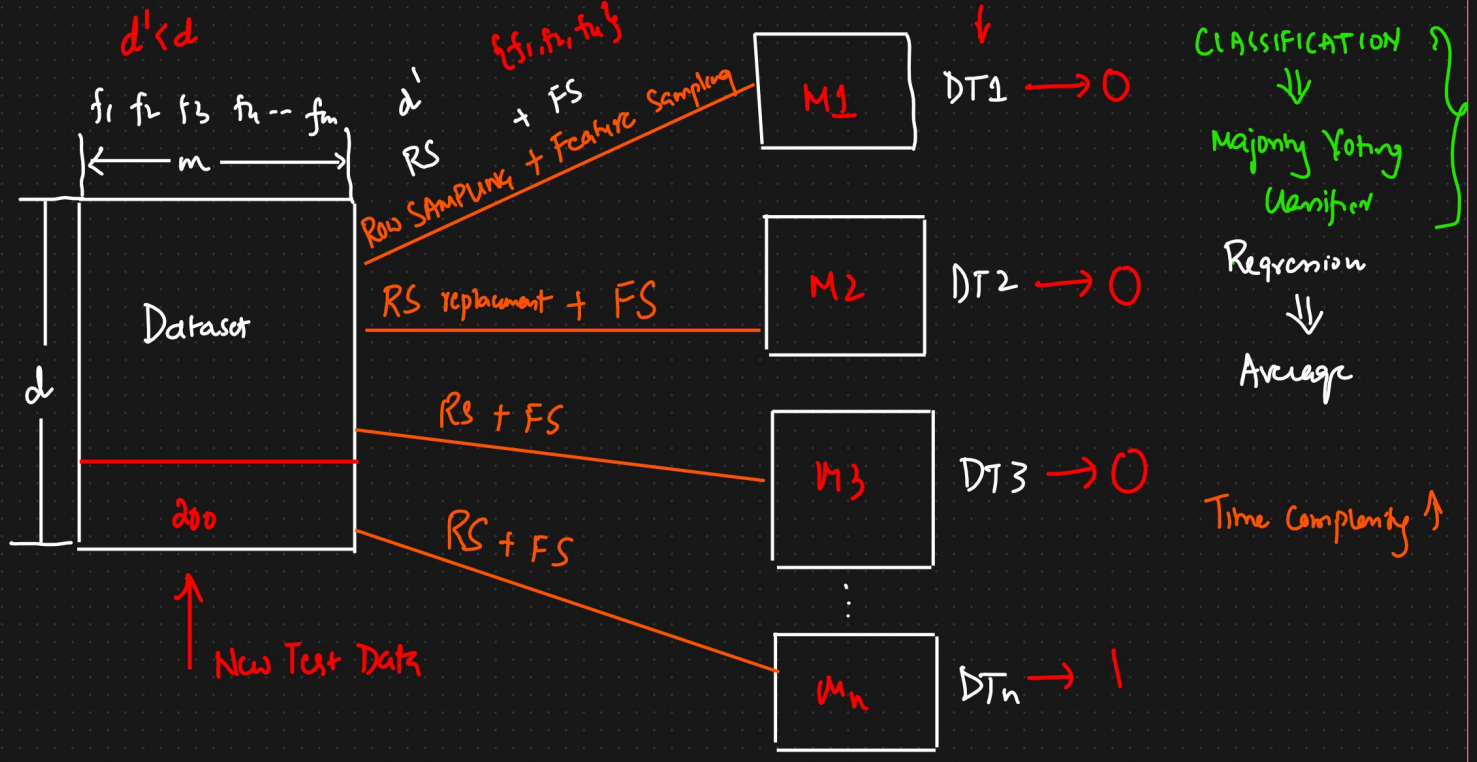
Decision Tree O/P Reg.

$\frac{52+60+56}{3} = 56$

O/P = $\left[\frac{40+42}{2} \right]$

$= 41$

Random Forest Classification And Regression



CLASSIFICATION - MAJORITY VOTING CLASSIFIER

REGRESSION - Average O/P of the Models.

Why should we use Random Forest instead of DT?

Decision Tree

Generalized Model → Low Bias
→ Low Bias

Overfitting

Train Acc ↑ → Low Bias

Test Acc ↓ → High Variance

Random Forest

→ Low Bias

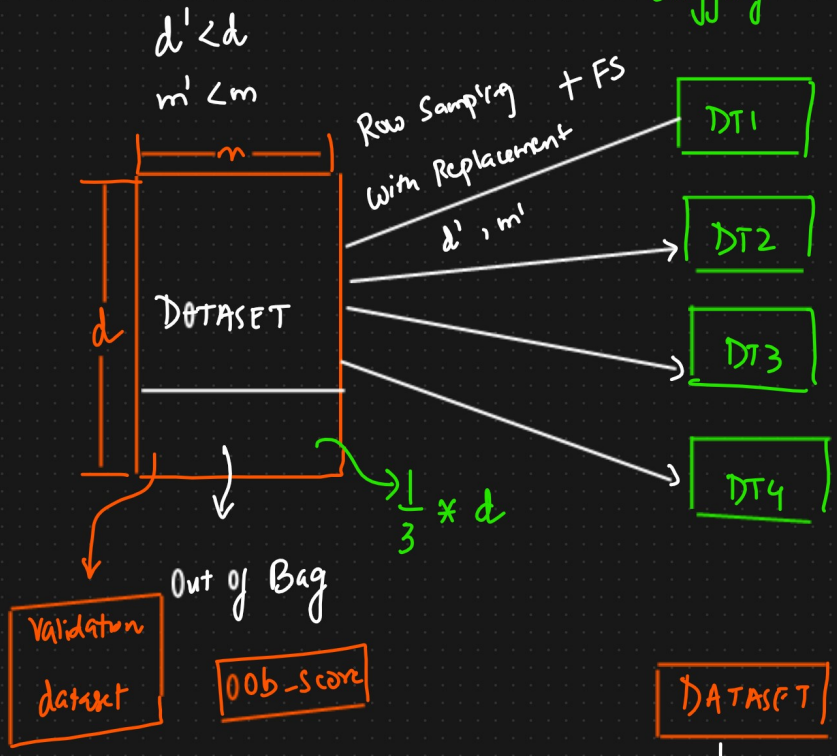
→ Low Variance

Out of Bag Evaluation (Random Forest)

Out of Bag Score?

⇓

Bagging



Sklearn

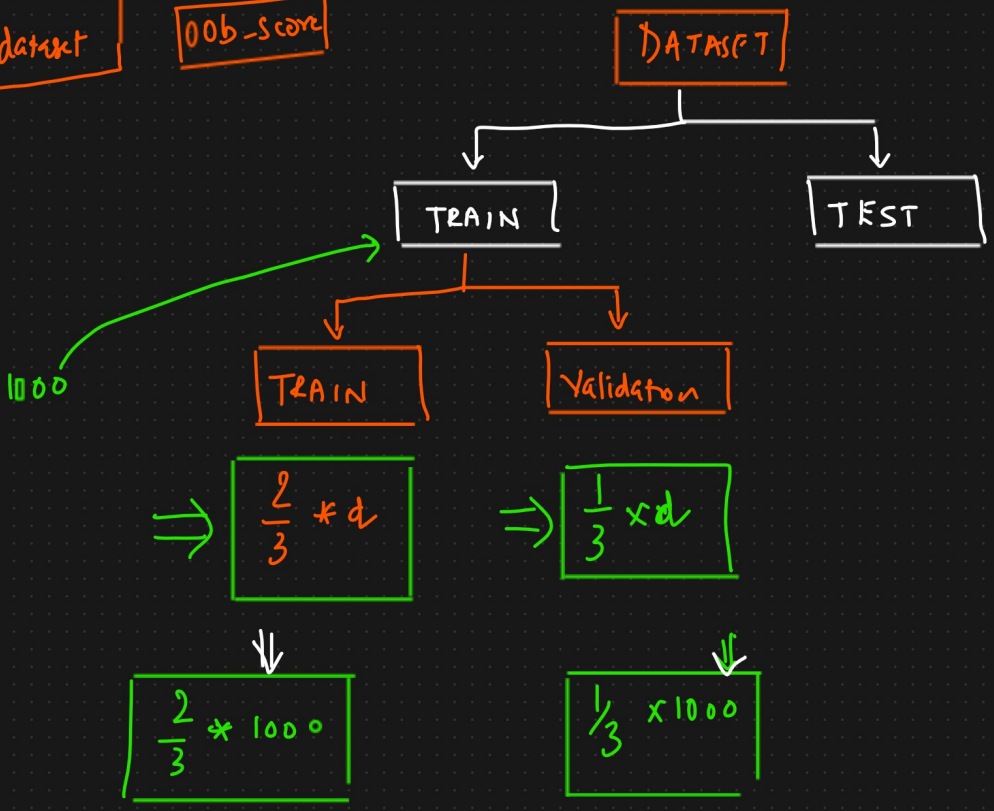
oob = TRUE

⇓⇓

oob score => percentage

⇓

Validation of Out of Bag DATA



AdaBoost Machine Learning Algorithms

Decision Trees

Ensemble Techniques

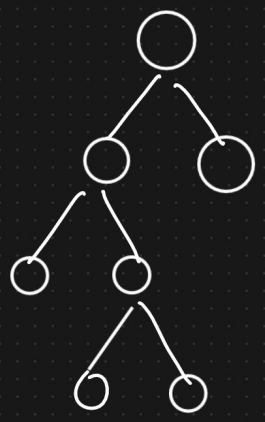
{Base Learners} → Bagging

{weak learners} → Boosting

{
① Random Forest Classifier
② Random Forest Regressor
}

① Adaboost
② Gradient Boosting
③ xgboost

Decision Tree :



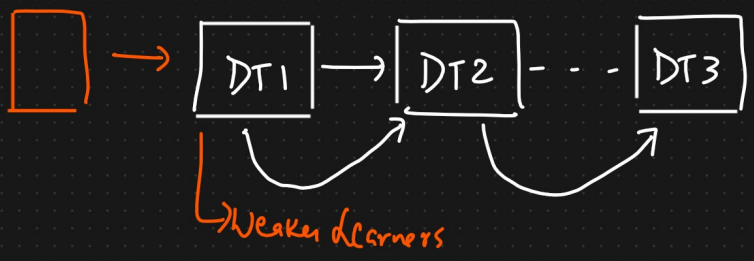
Overfitting : Train Acc ↑
Test Acc ↓

{
Low Bias
High Variance
}

Random Forest {Bagging}

{
Low Bias
Low Variance
}

Boosting {Sequentially connected}



⇒ Strong Learner

Weak Learner → Haven't learnt much from the Training Dataset

Random Forest → Majority Voting classifier
Average of (O/P)

AdaBoost ⟶ Assignment weights to the weak learner

$M_1 \dots M_n$ → Decision Tree Stumps

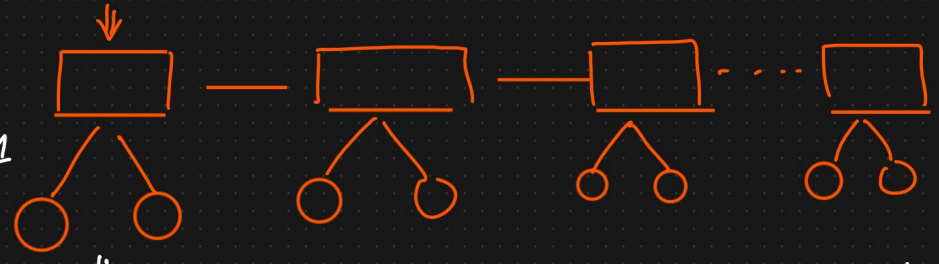
$$f = \alpha_1(M_1) + \alpha_2(M_2) + \alpha_3(M_3) + \dots \alpha_n(M_n)$$

→ CLASSIFICATION
→ REGRESSION

$\{\alpha_1, \alpha_2, \alpha_3 \dots \alpha_n\}$ ⇒ weights

Decision Stump

Depth = 1



Weak Learner

↳ Underfitting

↳ Train Acc ↓ 40%

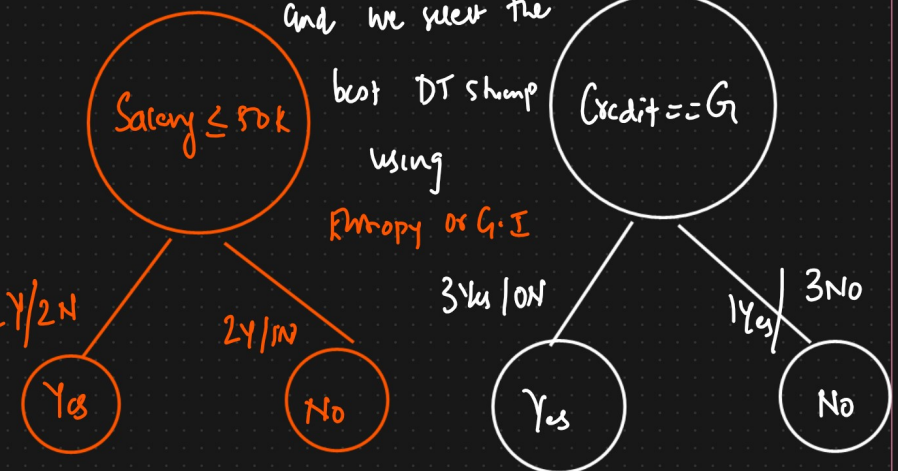
↳ Test Acc ↑ 45%

Decision Tree Stump ⇓ { High Bias, Low Variance } ⇒ { Low Bias, High Variance }

Adaboost Classifier Maths Indepth Intuition

Salary	Credit	Approval
<=50K	B	No
<=50K	G	Yes
<=50K	G	Yes
>50K	B	No
>50K	G	Yes
>50K	N	Yes
<=50K	N	No

① We create Decision Tree Stump and we select the best DT stump using Entropy or G.I



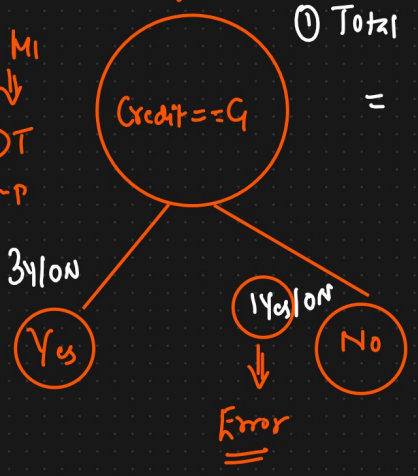
Entropy or Gini Impurity

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

② Sum of the Total Errors And Performance of Stump

Salary	Credit	Approval	Sample Weights
<=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

M1 or DT Stump



① Sum of all the Total Error = 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

③ Update the weights for correctly and Incorrectly classified points

Salary	Credit	Approval	Sample Weights	Update wts
<=50K	B	No	1/7 ↓ →	0.058
<=50K	G	Yes	1/7 ↓ →	0.058
<=50K	G	Yes	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 correct classified points

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

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

= 0.058

For Incorrect classified

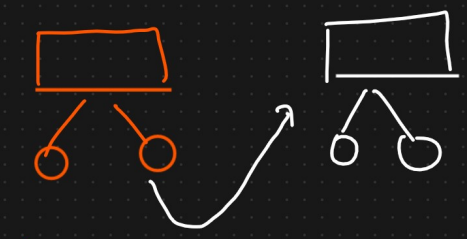
= Weight * $e^{\text{Performance of stump}}$

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

= 0.349

(4) Normalized Weights Computation And Assigning Bins

Salary	Credit	Approval	Update wts	Normalized weights	Bins Assignment
<=50K	B	No	0.058	0.08	0 - 0.08
<=50K	G	Yes	0.058	0.08	0.08 - 0.16
<=50K	G	Yes	0.058	0.08	0.16 - 0.24
>50K	B	No	0.058	0.08	0.24 - 0.32
>50K	G	Yes	0.058	0.08	0.32 - 0.40
>50K	N	Yes	0.349	0.50	0.40 - 0.70
<=50K	N	No	0.058	0.08	0.50 - 0.58
			0.697	≈ 1	



$\lambda_1 = 0.896$ Prepare datapoints

(5) Select data points to send to Next Stump

(1) Iteration process selecting random value between 0 and 1

Salary	Credit	Approval	Bins Assignment
<=50K	B	No	0 - 0.08
<=50K	G	Yes	0.08 - 0.16
<=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 - 0.70
<=50K	N	No	0.50 - 0.58

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

(6) These records will be sent to next DT Stump.

S	Credit	Approval	Sample weight
>50K	N	Yes	1/6
<=50K	G	Yes	1/6
>50K	N	Yes	1/6
>50K	N	Yes	1/6
<=50K	G	Yes	1/6
>50K	B	No	1/6
>50K	N	Yes	1/6

TE

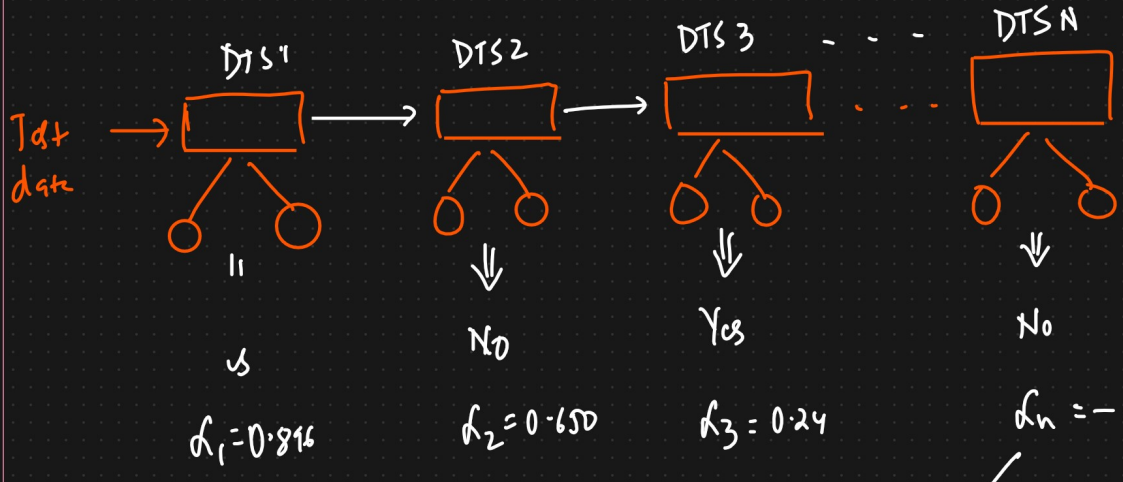
Performance Stump ⇒ 0.65

$f_1 = \alpha_1(M_1) + \alpha_2(M_2)$

$\alpha_1 = 0.896$ $\quad \alpha_2 = 0.65$

(7) Final Prediction

Test data $(\leq 50K, G)$



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

$f = \alpha_1(M_1) + \alpha_2(M_2) + \alpha_3(m_3) + \cdots \alpha_4(M_4)$

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

$$\begin{array}{r} 1\ 0.896 \\ 0.240 \\ \hline 1.136 \end{array}$$

$= \boxed{1.136}(Y_{es}) + 0.350\,(No)$ ⇒ O/p : Yes

Performance of say (Yes) = 1.136 >

Performance of say (No) = 0.350

Gradient Boosting Algorithm

① Regression

② Classification

Regression Dataset

Exp	Degree	Salary (y)	$\hat{y}$	$(y-\hat{y})$ R_1	R_2	$\hat{y}$	R_3	R_4
→ 2	B.E	50K	75K	-25K	-23	72.7	-22.7	-
3	Masters	70K	75K	-5K	-3	74.7	-4.7	-
5	Masters	80K	75K	5K	3	-	-	-
6	PHD	100K	75K	25K	20	-	-	-
		75K						

Steps

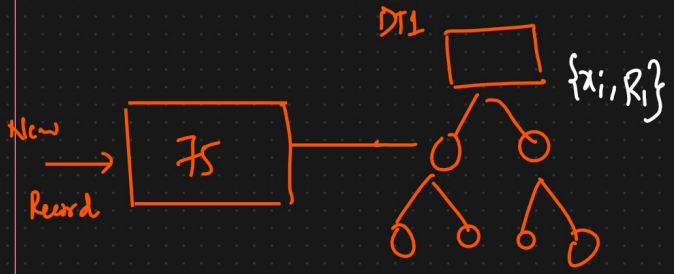
① Create a Base Model

Average = $\frac{50+70+80+100}{4} = 75$

② Compute Residuals, Error

③ Construct a Decision Tree

Consider inputs x_i and O/p R_i



Predicted O/p ⇒ 75 + (-23) = 75 - 23 = 52 {Overfitting}

Predicted O/P $\Rightarrow$ $75 + \alpha(DT_1) = 75 + (0.1)(-23)$

α = learning (0 to 1)

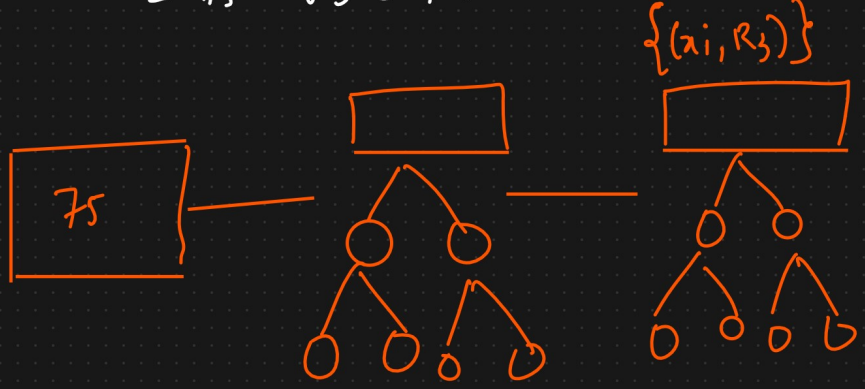
$\alpha = 0.1$

$= 75 - 2.3$

$= 72.7$

$\Rightarrow 75 + 0.1(-3)$

$= 75 - 0.3 = 74.7$



$$F(x) = \alpha_0 h_0(x) + \alpha_1(h_1(x)) + \alpha_2(h(x)) + \alpha_3(h(x)) - - + \alpha_n(h(x))$$

learning Rate $\alpha = 0.1$

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

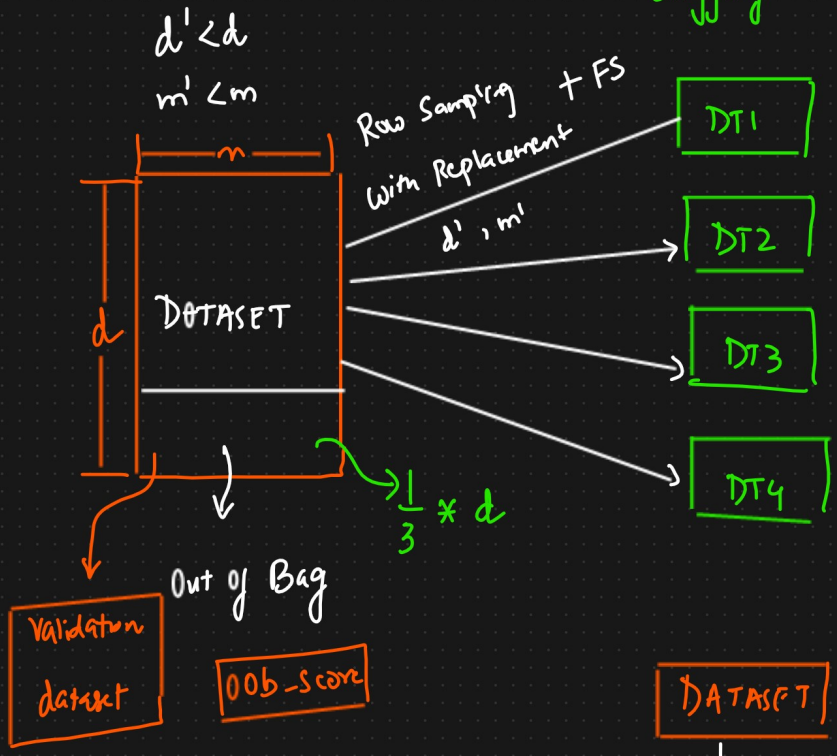
$\Rightarrow$ Final Function of Gradient Boosting

Out of Bag Evaluation (Random Forest)

Out of Bag Score?

⇓

Bagging



Sklearn

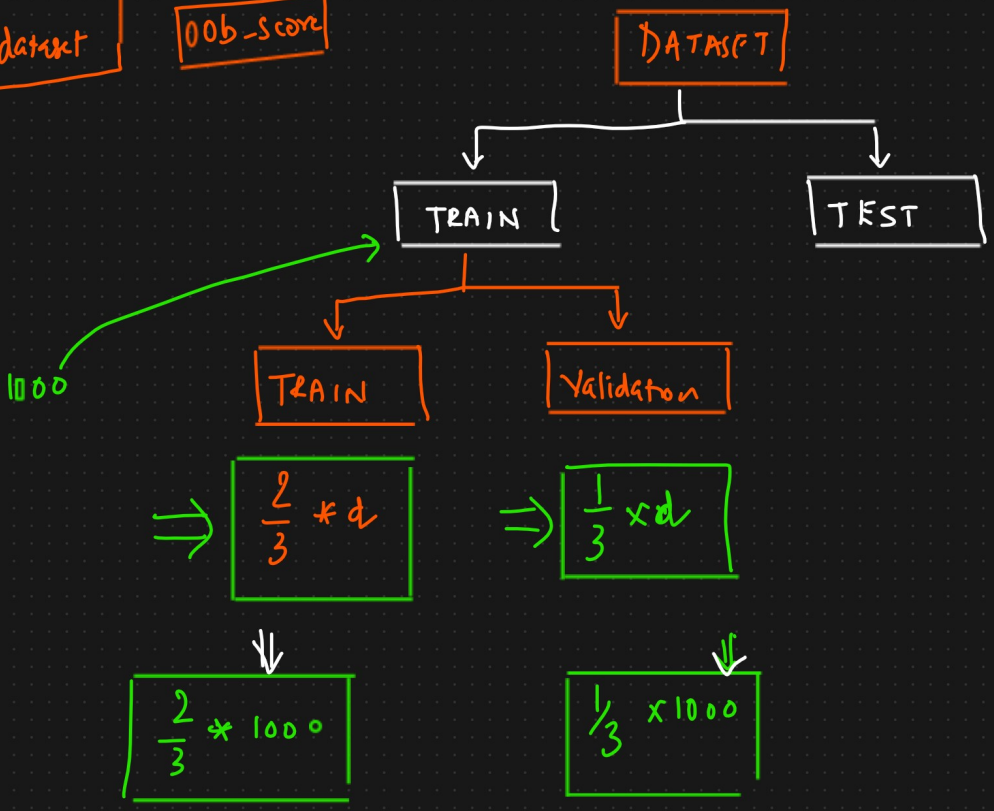
oob = TRUE

⇓⇓

oob score => percentage

⇓

Validation of Out of Bag DATA



Xgboost ML Algorithm (Classification)

$\hat{y} = 0.5$

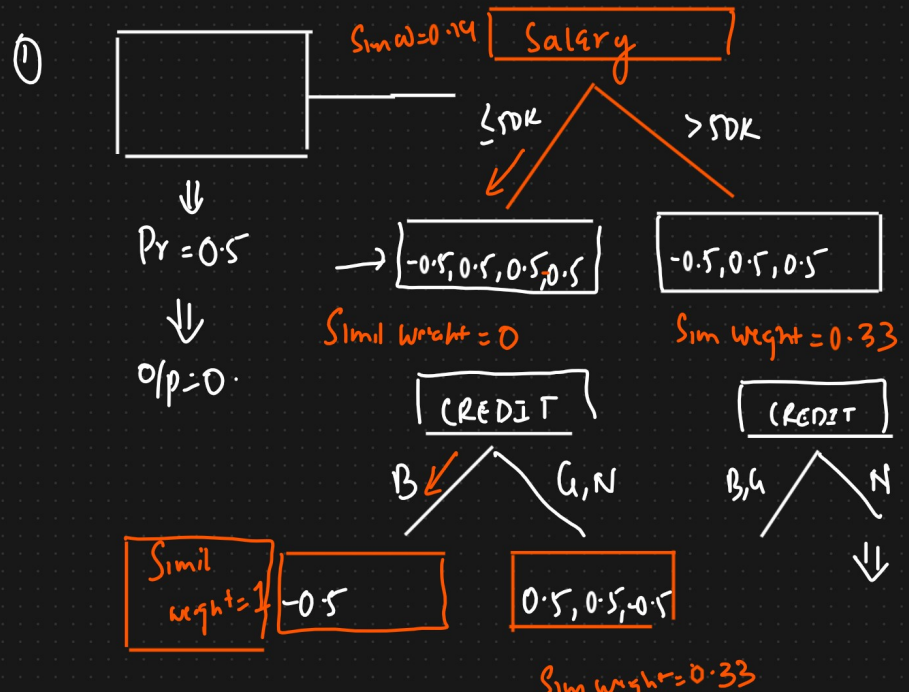
Dataset

Salary	Credit	y Approval	R1	$\hat{y}$	R2
<=50K	B	0	-0.5	0.52	-0.48
<=50K	G	1	0.5	0.58	0.42
<=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.
3. Calculate Similarity Weight
   $= \sum (\text{Residual})^2$
   Cover value $\leftarrow \sum Pr(1-Pr) + \lambda$ (hyperparameter)
4. Calculate Gain

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



Test data

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

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

$$= 0$$

$$\text{Similarity (Root)} = \frac{0.25}{0.25} = 1$$

$$\text{Similarity (Rest)} = \frac{0.25}{0.75} = 0.33$$

$$\text{Gain} = 1 + 0.33 - 0 = \boxed{1.33}$$

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

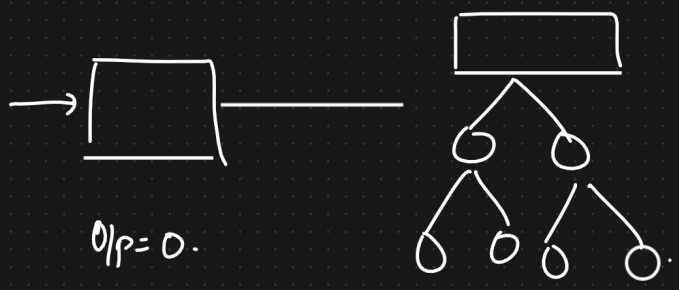
Similarity (RC) $= \frac{(-0.5+0.5+0.5)^2}{0.75} = \frac{0.25}{0.75} = \frac{1}{3}$

$= [(-0.5+0.5+0.5-0.5)^2] = 0$

$1 \Leftarrow [0.5(1-.5)+0.5(1-.5)+ .5(1-.5)+0.5(1-.5)]$

Gain $= 0 + 0.33 - .14 = .21$

Final O/p



New Data $= \sigma\left(0 + \lambda(1)\right)$ ← (Similarity weight), ↓

$\lambda = 0.1$

Logistic function ⇐ Sigmoid Activation $= \sigma\left(0 + (0.1)(1)\right)$

$= \frac{1}{1+e^{-0.1}} = 0.52$

Second Record O/p $= \sigma\left(0 + \lambda(0.33)\right)$

$= \sigma\left(0 + 0.1(0.33)\right)$

$$= \frac{1}{1+e^{-0.033}} = 0.508$$

Xgboost classifier

$$O/P = \sigma\left(\text{Base learner} + \alpha_1(DT1) + \alpha_2(DT2) + \alpha_3(DT3)\right).$$

Xgboost Regressor Mh Algorithm

Dataset {Regression}

Exp	Gap	Salary	R_1	$\hat{y}$	R_2
2	Yes	40K	-11	49.9	-9.9
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
4.5	Yes	62K	11	52.1	9.9
		≈ 51K			

$[51 + (0.1)(-10)] = 51 - 0.1 = 49.9$

$[51 + (0.1)(5)] = 51.5$

$[51 + 0.1(11)] = 51 + 1.1 = 52.1$

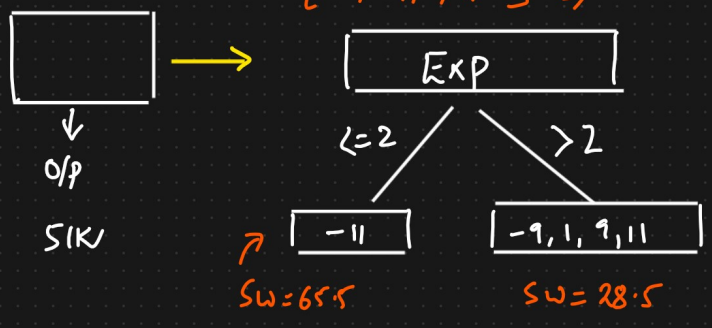
$$\text{Similarity weight} = \frac{\sum(\text{Residual})^2}{\sum p_r(1-p_r)}$$

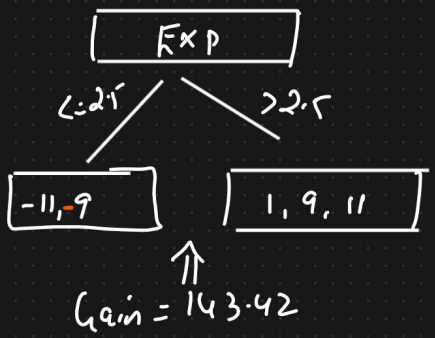
Gain

Steps

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

$[-11, -9, 1, 9, 11] \Rightarrow SW = 0.16$.





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

λ → Hyperparameter

$\lambda \uparrow$ SW ↓

$\lambda = 1$

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

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

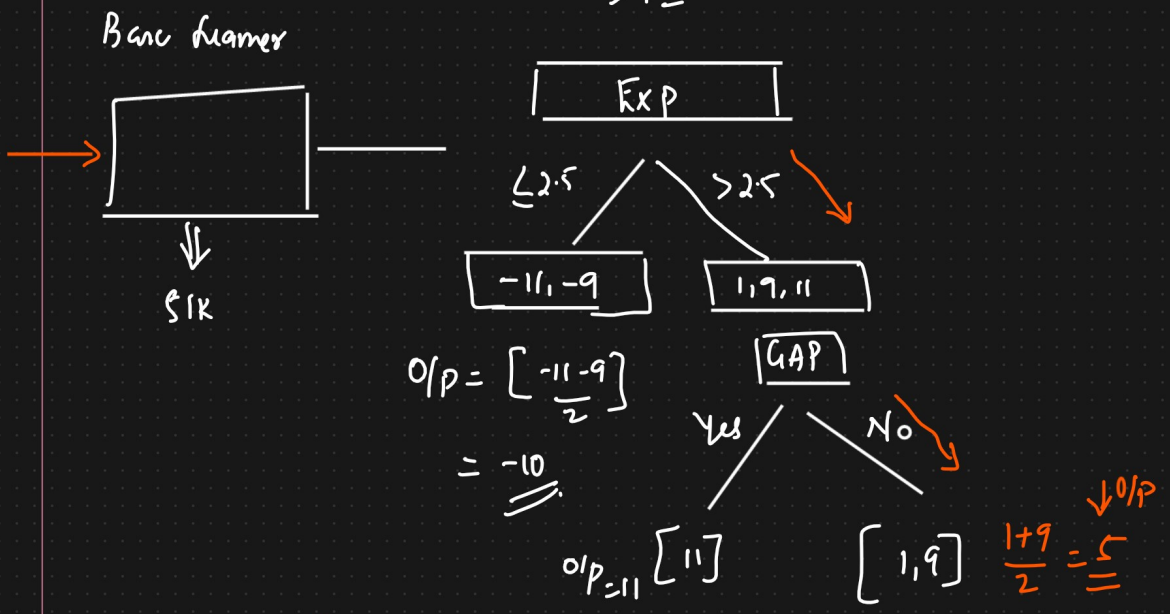
$$= \frac{144}{5} = 28.5$$

(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}

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

Unsupervised Machine Learning

Supervised ML → Regression
→ Classification

Dataset

I/p features | Dependent or O/p features

f1 f2 f3 f4 | O/p

Unsupervised ML → Clustering {group your data into similar clusters}

Dataset

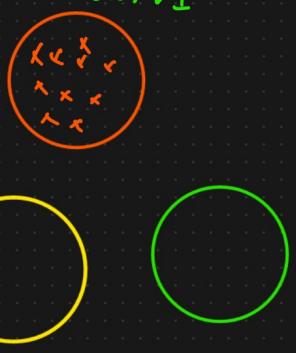
Age	Experience	Salary
—	—	—
—	—	—
—	—	—
—	—	—
—	—	—
—	—	—

No O/p

⇒ Unsupervised ML

⇓

Group or cluster this data

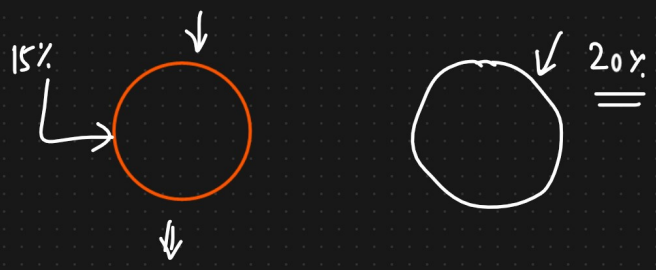


Eg: Customer Segmentation

Apple

Product : Data | Salary | Spending Score

↳ New Product ← Discount



Unsupervised ML

1. K Means Algorithm
2. Hierarchal Clustering
3. DBScan Clustering

Validate

4. Silhoutte Scoring

Principal Component Analysis (PCA) [Dimensionality Reduction]

① Curse of Dimensionality

→ Dimensions →

Dataset : 500 features

Price of the House

House size
No. of bedrooms
No. of bathrooms

3 features → M1 → Acc 1

6 features → M2 → Acc 2 ↑↑ > Acc 1

15 features → M3 → Acc 3 ↑↑↑ > Acc 2

50 features → M4 → Acc 4 ↓↓ < Acc 3

← Model is overfitted

100 features → M5 → < Acc 5 ↓↓↓ < Acc 4

500 features → M6 → Acc 6 ↓↓↓↓ < Acc 5

② Model performance Degrade

Domain

Person → ↓↓↓ → Confused

Accurate

f1 2BhK

Loca ← 450K - 550K

3BhK ← 500K - 600K

Beach ← ↑↑↑

Near to a celebrity house ← ↑↑↑

Grocery shop ← ↑↑

School ←

Two different ways to remove curse of Dimensionality

① Feature Selection ↑↑

② Dimensionality Reduction (PCA) ↓↓↓

original f1 f2 f3 ... fn

Reduce ↓ Feature Extraction

D1 D2 ... O/P

Imp features

↓

Feature Extraction

Feature Selection Vs Feature Extraction

↳ Dimensionality Reduction

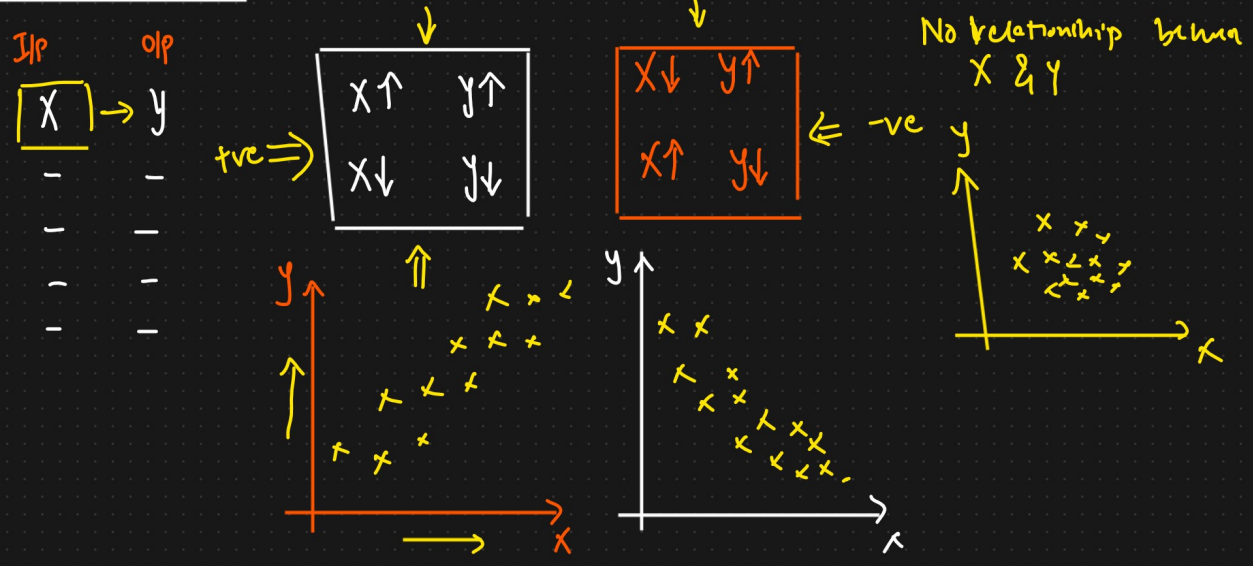
① Why Dimensionality Reduction?

- Prevent ⟶ Curse of Dimensionality
- Improve the performance of the model
- Visualize the data ⟶ understand the data

3d → 2d

100 d → 3d or 2d

Feature Selection



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

= +ve

≈0

= -ve

≈0 {No Relationship}

-ve correlated

$$\text{Pearson Correlation} = \frac{Cov(x,y)}{\sigma_x * \sigma_y} = \boxed{-1 \text{ to } 1}$$

The more toward the value of ±1 the

more +ve Correlated X & Y is

Dataset Housing

Independent feature

House Size | Fountain Size | Price → O/P feature

{Feature Selection}

DROP

Price

{Linear Relationship}

Covariance = +ve

-ve

House Size

Price of House

0 - 0.2

Fountain Size

Domain Expertise

Feature Extraction

+ve

-ve

2 feature → 1 feature

Dimensionality Reduction

{ Room Size | No. of Rooms | Price | Price

Transformation To extract New feature

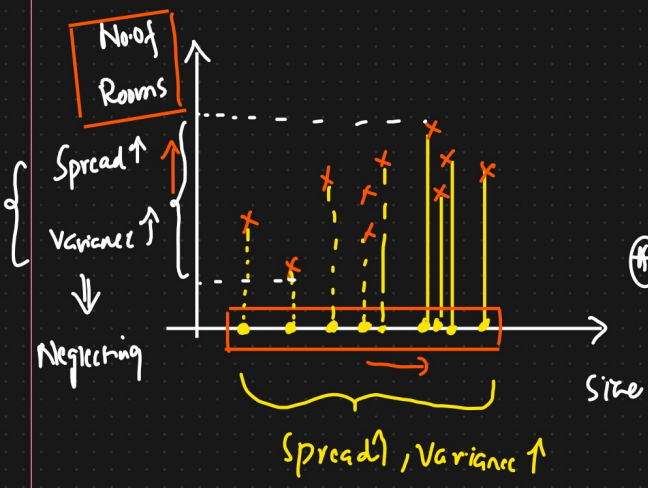
House Size | Price

PCA Geometric Intuition

{Dimensionality Reduction}

Housing Dataset

Size House	No. of Room	Price ↗ O/P



PCA

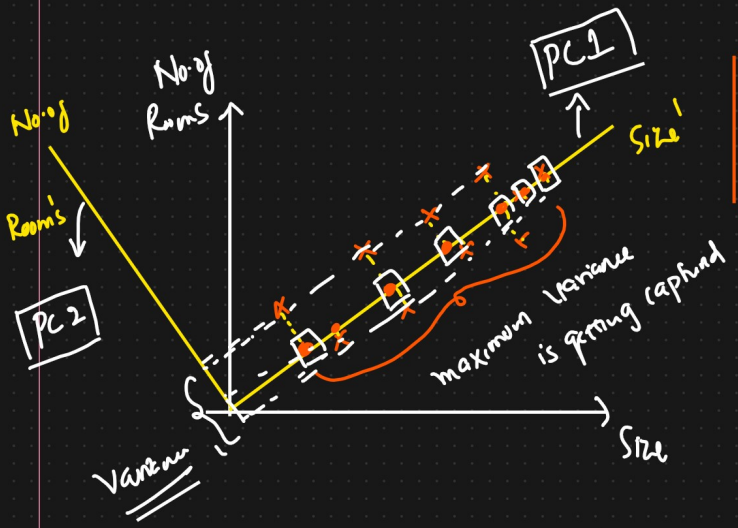
2 dimension ⟶ 1 dimension

2D → 1D

* Loss of information (No. of Rooms)

⇓

Feature Extraction



Eigen decomposition on Matrix

⇑

Transformation

2 Dimension

PC1, PC2

Var(PC1) > Var(PC2)

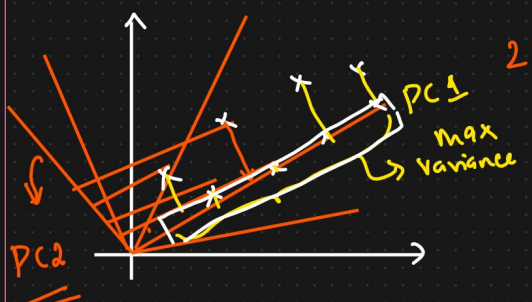
3 Dimension

PC1, PC2, PC3

Var(PC1) > Var(PC2) > Var(PC3)

2D ⟶ 1 Dimension

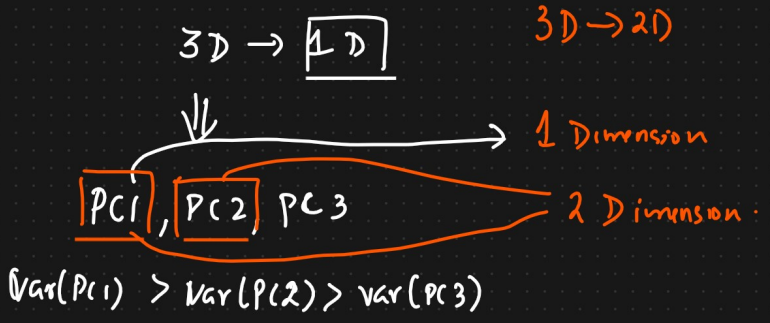
much information is not lost



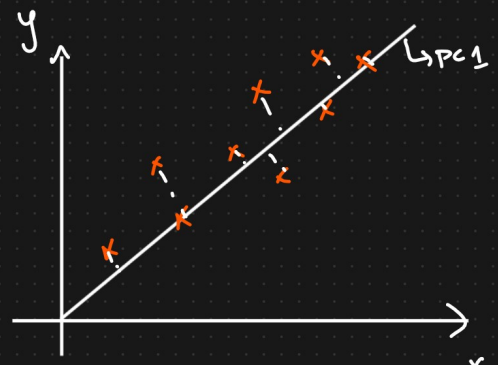
2D → 1D

2 Best Principal Component

To get the best Principal Component which captures maximum variance

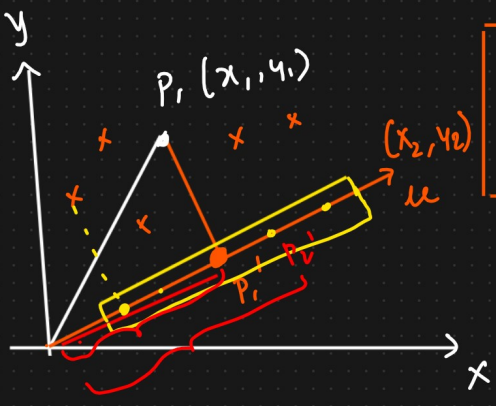


Maths Intuition behind PCA Algorithm



① Projection ✓

② Cost function → Variance ✓

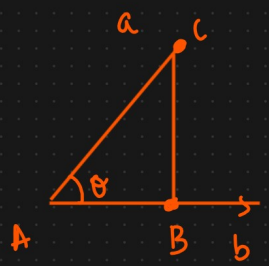


$$Proj_{P_1} u = \frac{P_1 \cdot u}{\|u\|}$$

$\|u\| = 1$ → unit vector

$$Proj_{P_1} u = P_1 \cdot u$$ ⇒ Scalar value

P_1'



$P_0', P_1', P_2', P_3', P_4', \ldots P_n'$

⇓

Scalar values

⇓

Variance

$P_0', P_1', P_2', P_3', P_4', \ldots P_n'$

⇓

$x_0', x_1', x_2', x_3', x_4' \ldots x_n'$

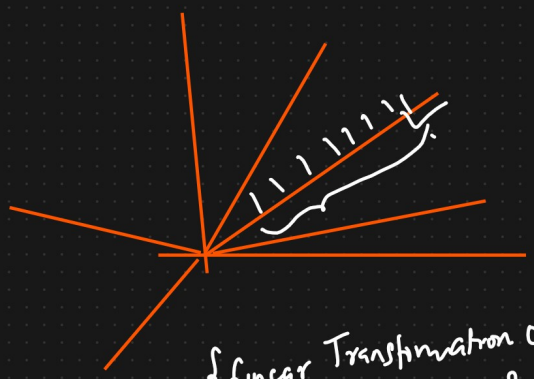
Max Variance = $\sum_{i=1}^{n} \frac{(x_i - \bar{x})^2}{n}$ ↑

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

⇓

Cost function

⇓ Eigen vectors And ⇓ Eigen values.



① Covariance Matrix between features

② Eigen vectors and Eigen values will found out from this covariance matrix

{Linear Transformation of matrix}

$Av = \lambda v$

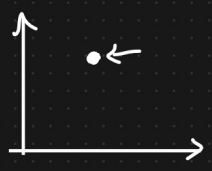
③ Eigen vector ⟶ Eigen value → magnitude of the Eigen vector ⟶ Capture the maximum variance

Eigen vectors And Eigen values [Linear Transformation]

[Eigen decomposition of covariance Matrix]

⇓

Eigen vector & Eigen values



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

λ ↓ Eigen value

$$A * v = \lambda * v$$

Eigen vector → Maximum magnitude

Eigen vector → Max Magnitude → Max Eigen vector → Best Principal Component → PC1

Principal Component → Max Var

Steps to calculate Eigen value and vectors

① Covariance of features

$[X, Y]$ → Z → X'

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

2×2

A =	X	Y
X	Var(X)	Cov(X,Y)
Y	Cov(Y,X)	Var(Y)

$Cov(X,X) = Var(X)$

$Cov(Y,Y) = Var(Y)$

	X	Y	Z
X			

$\lambda_1, \lambda_2, \lambda_3$

f_1 & f_2

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

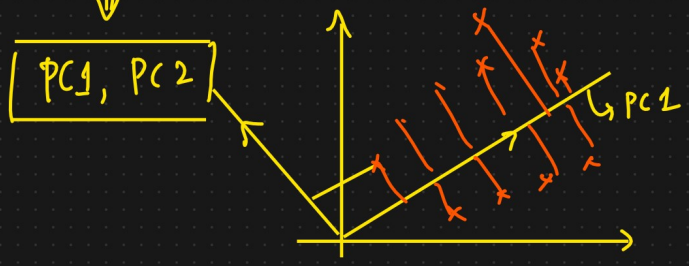
λ_1 λ_2

λ_1 ⇓ PC1 . λ_2 ⇓ PC2

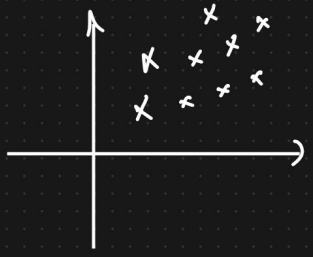
λ_1, λ_2 → Eigen values

⇓

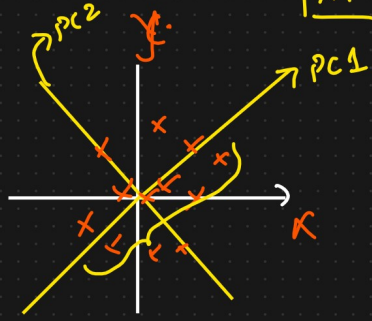
PC1, PC2



(4)



2D → 1D



λ_1 ⇒ magnitude of the Eigen vector

① Standardize the data

② Covariance Matrix of X & Y

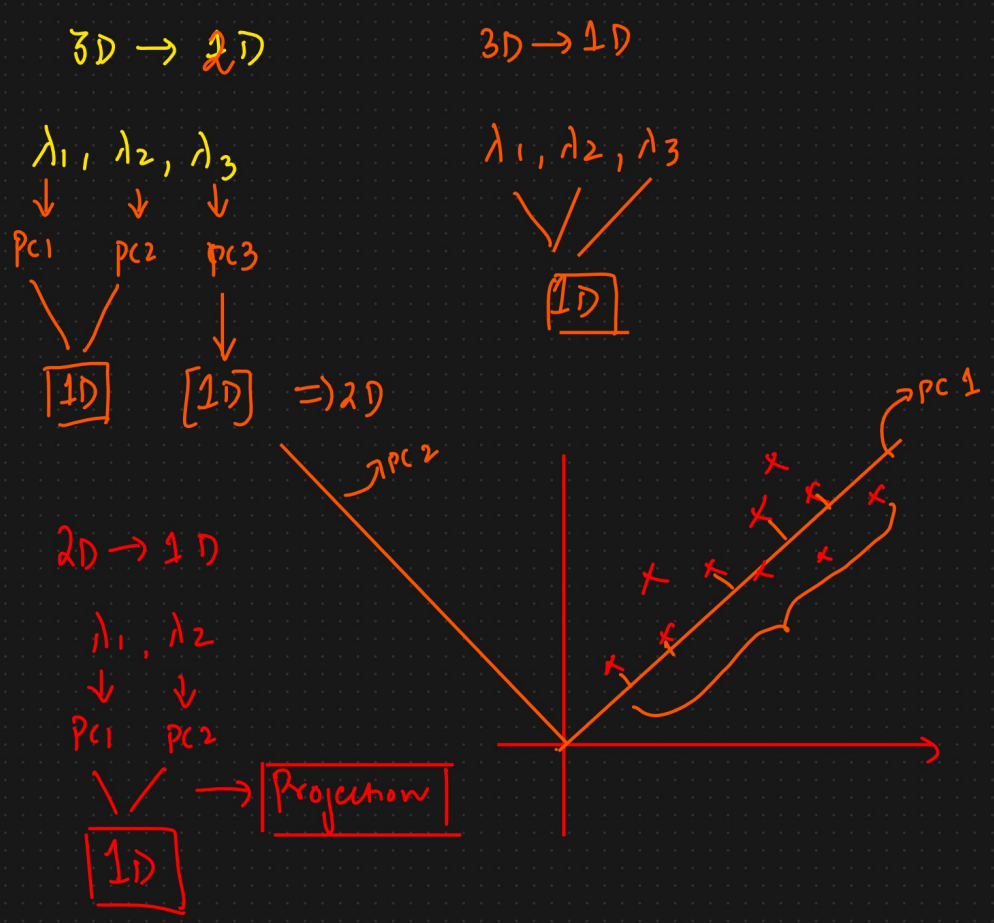
$$A = \begin{array}{c|cc} & X & Y \\ \hline X & Var(x) & Cov(x,y) \\ Y & Cov(y,x) & Var(y) \end{array} \quad 2 \times 2$$

③ Find out Eigen vectors And value

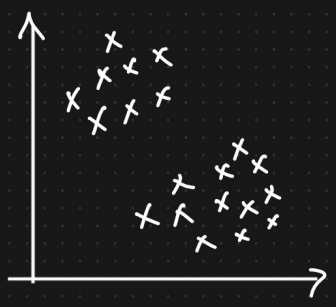
$A v = \lambda v$

λ_1, λ_2 ⇒ Eigen values.

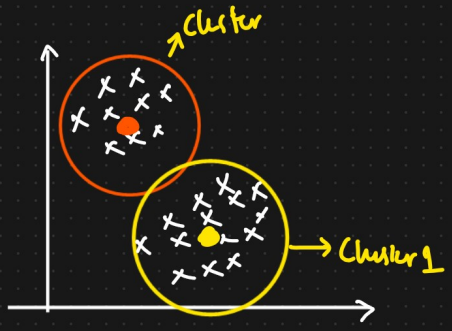
↓ PC1 ↓ PC2

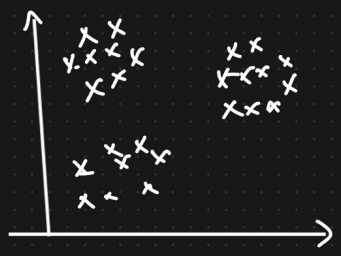

3D → 2D
λ1, λ2, λ3
PC1
PC2
PC3
1D
1D
=>2D
PC2
2D → 1D
λ1, λ2
PC1
PC2
Projection
1D
3D → 1D
λ1, λ2, λ3
1D
PC 1

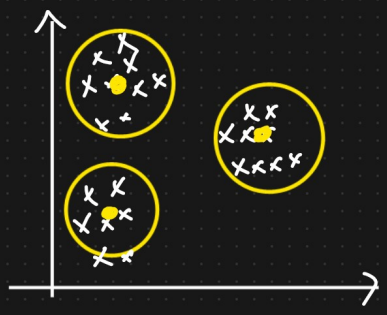
K Means Clustering Algorithm



K Means







Euclidian Distance
OR
MANHATTAN Distance

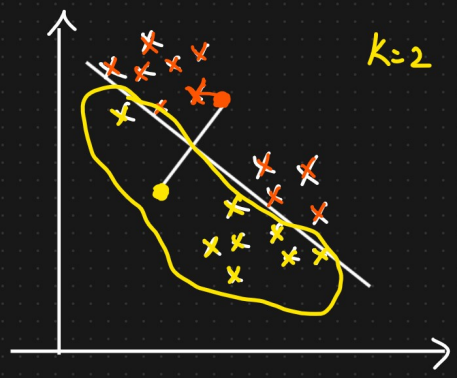
K=2

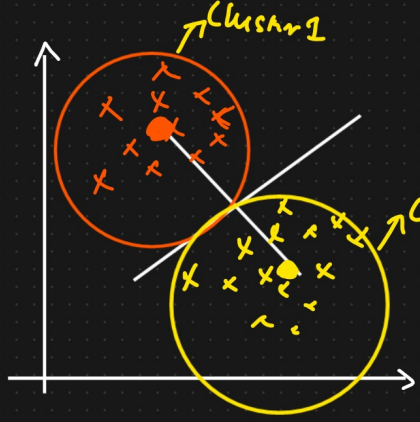
K Mean

① Initialize Some K → centroids

② Points that are nearest to the Centroids → Group

③ Move the centroids → Average

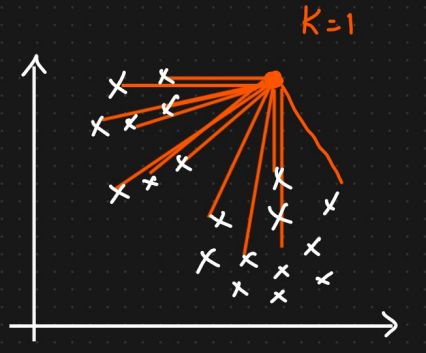




How do we select the K value?

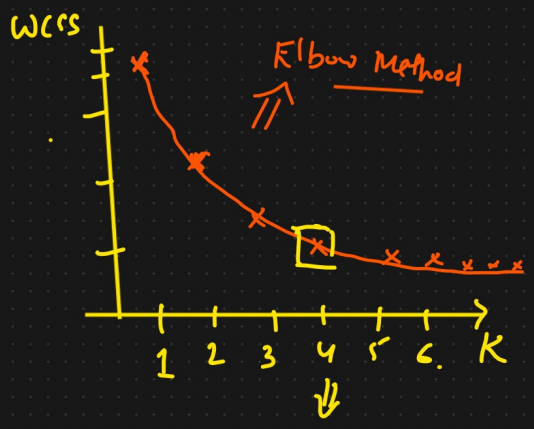
WCSS = Within Cluster Sum of Squares

Initialize K=1 to 20

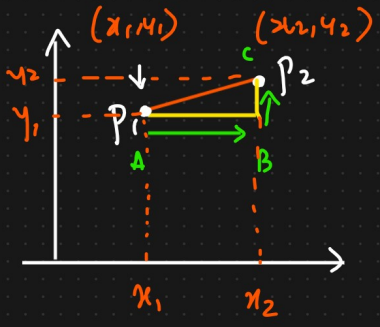


$$WCSS = \sum_{i=1}^{n} \left(\text{distance between points to nearest centroid}\right)^2$$

WCSS ↓



ⓕ Euclidean Distance



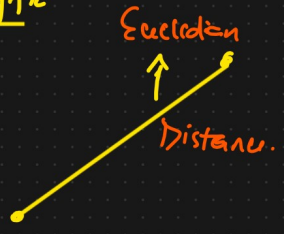
$$\text{Euclidean dist} = \sqrt{(x_2-x_1)^2+(y_2-y_1)^2}$$

$$\text{Manhattan dist} = |x_2-x_1| + |y_2-y_1|$$

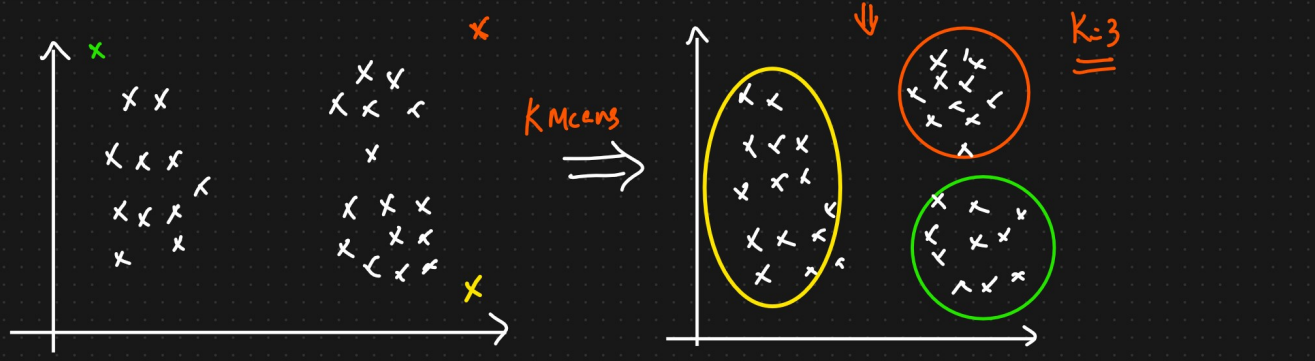
IRON MAN → U.S

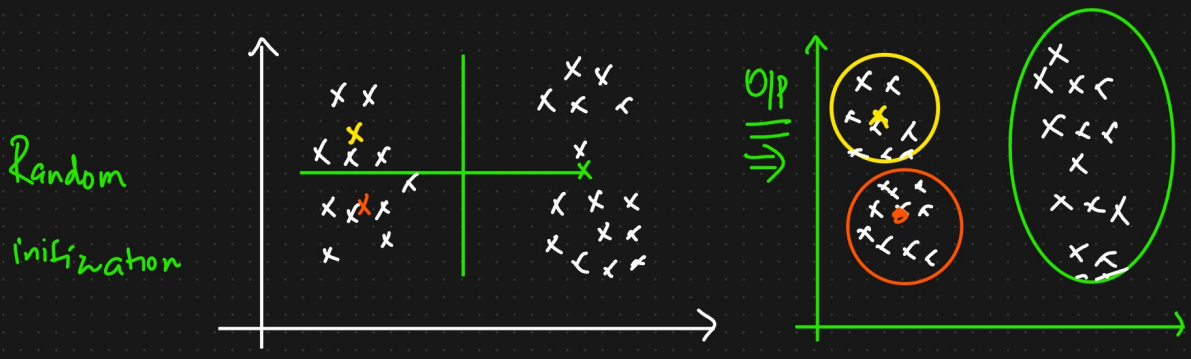
⇒ Manhattan distance

Air Traffic



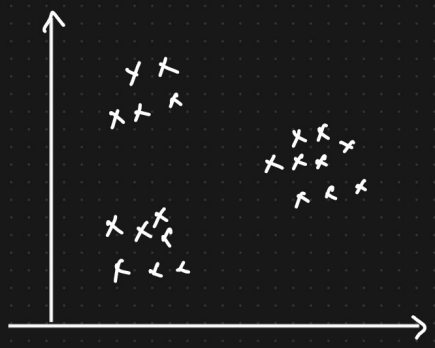
Random Initialization TRAP (Kmeans ++)



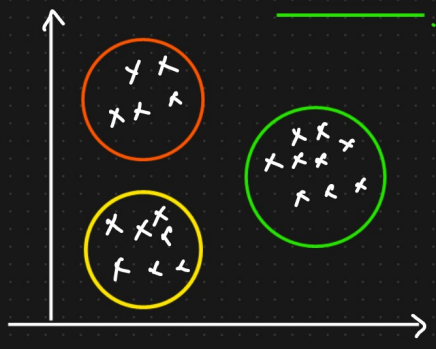


Kmeans ++ Initialization Technique

Hierarichal Clustering

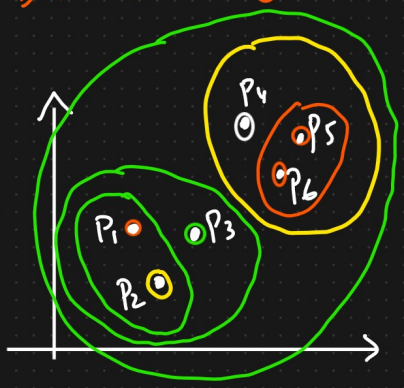


⇒ Hicrarichal Clustering



HC

① Agglomerative
② Divisive
⇒ Geometric Intuition.



Steps

① For each point initially will consider it as a Seperate Cluster

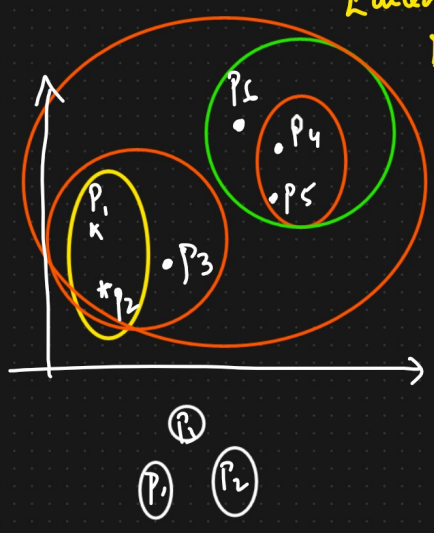
② Find the nearest point and create a new Cluster

③ Keep on doing the same process until we get a single Cluster

Dendogram

No. of clusters

Euclidian Distance



Threshold

K=2

K=4

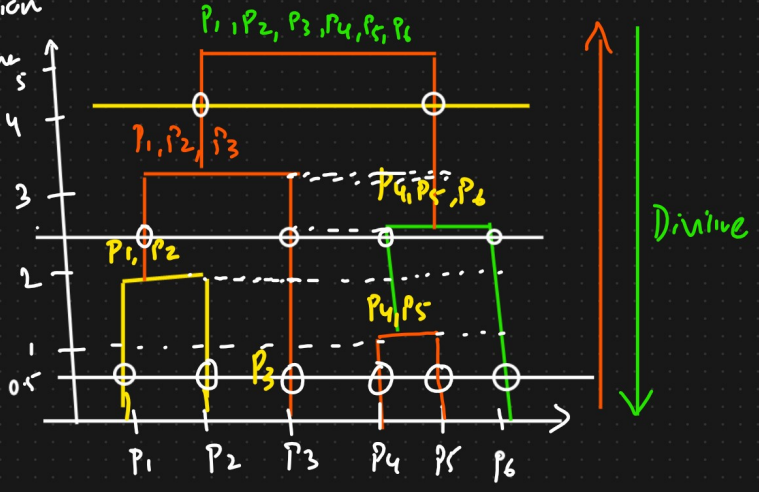
Dendogram

K=2

K=1

Agglomerative

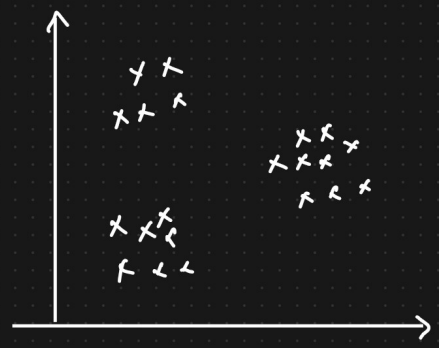
Euclidian Distance



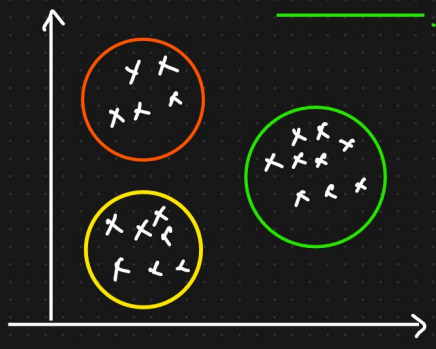
(K) Select the longest vertical line such that no horizontal line passes through it }

Threshold {Euclidean Distance}.

Hieraridal Clustering

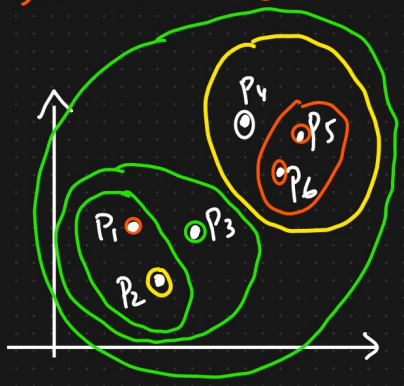


⇒ Hieraridal Clustering



HC

① Agglomerative
② Divisive
⇒ Geometric Intuition.



Steps

① For each point initially will consider it as a separate cluster

② Find the nearest point and create a new cluster

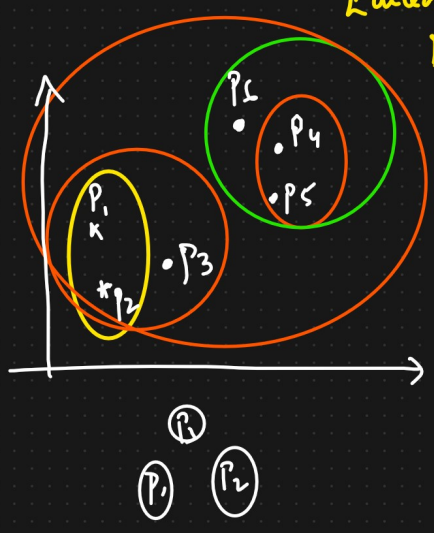
③ Keep on doing the same process until we get a single cluster

Cosine Similarity

No. of clusters

Dendogram

Euclidian Distance



Threshold

K=2

↑↑↑↑

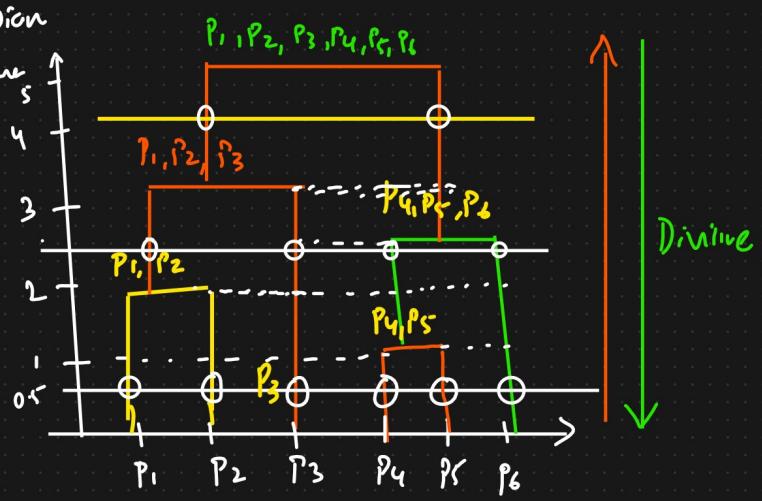
K=4

Dendogram

K=2

K=1

Agglomerative



④ Select the longest vertical line such that no horizontal line passes through it

Threshold {Euclidean Distance}.

K Means Vs Hierarchical Clustering

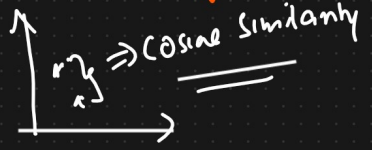
Scalability And Flexibility

① Dataset size → Huge → K Means

Small → Hierarchical Clustering

② Kmean → Numerical data

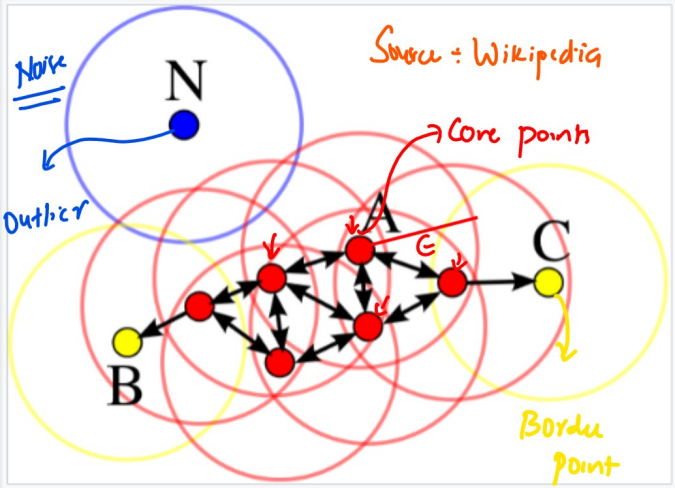
Hierarchical clustering → Variety of data.



③ Centroids → Elbow method → No. of centroids

→ No. of Clusters

DBSCAN Clustering.



In this diagram, minPts = 4. Point A and the other red points are core points, because the area surrounding these points in an ε radius contain at least 4 points (including the point itself). Because they are all reachable from one another, they form a single cluster. Points B and C are not core points, but are reachable from A (via other core points) and thus belong to the cluster as well. Point N is a noise point that is neither a core point nor directly-reachable.

- Core point
- border point
- Outlier

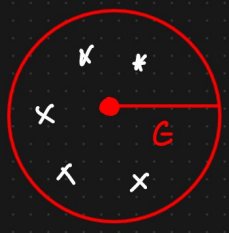
} Non linear Clustering

minpts = 4 ϵ = radius

Core point

minpts = 4

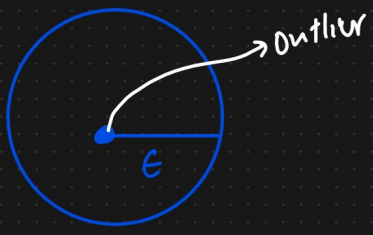
① No. of points within the ϵ should be greater ≥ 4



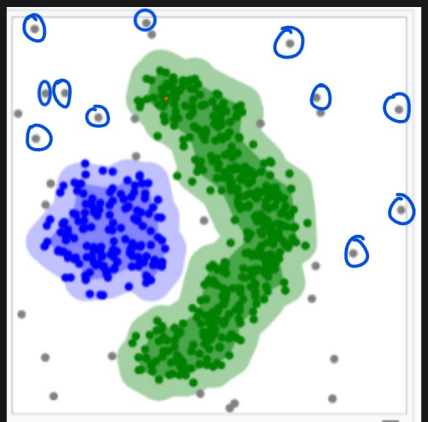
Border point

no. of data points within this radius will be less than minpts

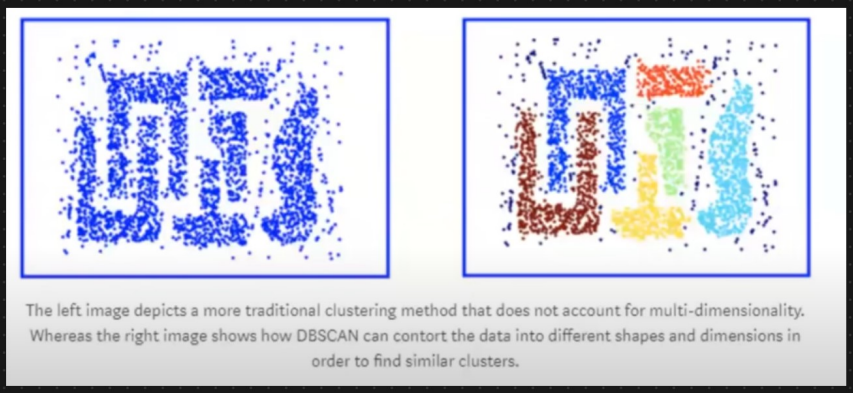
Outlier (Noise)



Some Examples after we apply DBScan Clustering



DBSCAN can find non-linearly separable clusters. This dataset cannot be adequately clustered with k-means or Gaussian Mixture EM clustering.

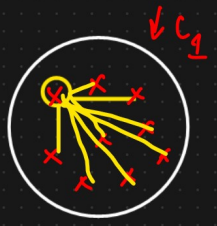


The left image depicts a more traditional clustering method that does not account for multi-dimensionality. Whereas the right image shows how DBSCAN can contort the data into different shapes and dimensions in order to find similar clusters.

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).

a(i)

②

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 $C_J \neq C_I$).

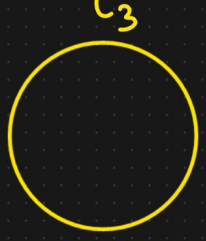
For each data point $i \in C_I$, we now define

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

⇒ b(i)

a(i) << b(i)

to be the *smallest* (hence the $\min$ operator in the formula) mean distance of i to all points in any other 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.



a(i)

② Silhouette Score

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

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

More near to 1 better

Clustering model we have created

Which can be also written as:

+ve. ≈ 1

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

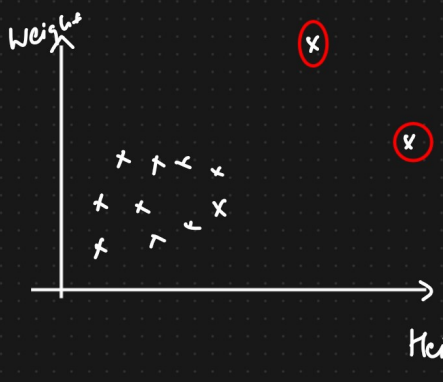
≈ −1

From the above definition it is clear that

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

Anomaly Detection [To detect Outliers]

↳ Play a important Role



IPL	
1	15
2	10
3	12
4	100

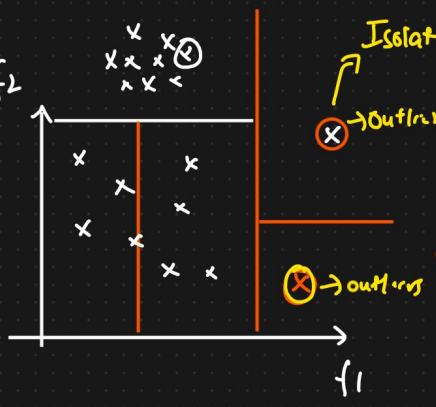
>36

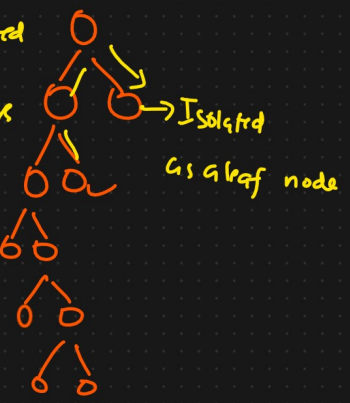
① Isolation Forest [Decision Trees].

Many Trees

Isolated Trees

f_1	f_2	f_3	f_n
-	-	-	-
-	-	-	-
-	-	-	-
-	-	-	-





Anomaly Score

Mathematical Formula : Compute anomaly score for a new point

$$S(x,m) = 2^{-\frac{E(h(x))}{c(m)}}$$

m = no. of data points

x = Data point.

$E(h(x))$ = Average search depth for x from the isolate tree.

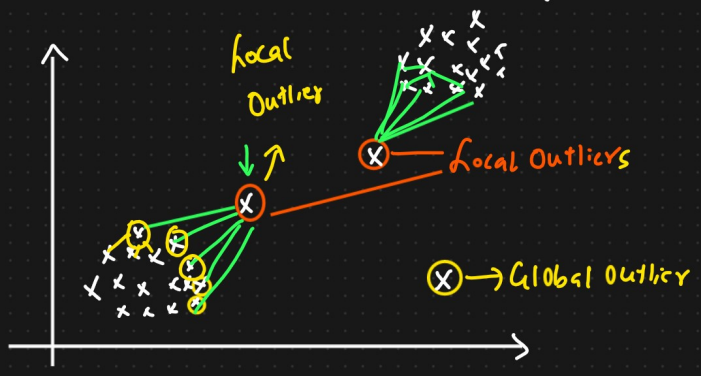
$c(m)$ = Average depth of $h(x)$

[Threshold >0.5]

$E(h(x)) << c(m) \Rightarrow S(x,m) \approx 1$. ⇒ Anomaly score ⇒ Outliers

$E(h(x)) >> c(m) \Rightarrow S(x,m) \approx 0.5$ ⇒ Normal data point.

Local Outlier factor Anamoly Detection.



① Local Outlier
② Global Outlier

LOF ⇒ Lof Score

K Nearest Neighbour

⇓

Local Density

K=5