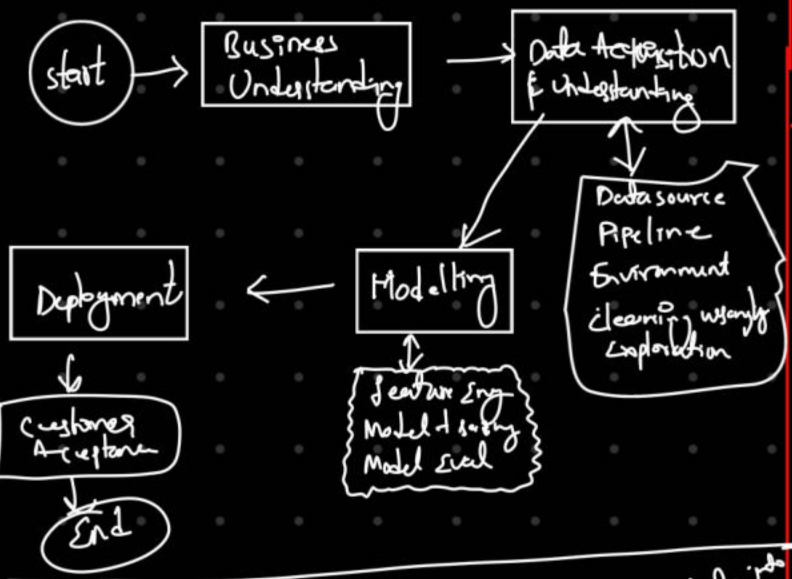
Life Cycle of a Data Science Project



EDA
↳ Data Analysis
↳ Data preprocessing
→ feature Eng → making the data into right format to train machine
→ feature selection → only selecting necessary feature to train

feature selection techniques

1. Univariate selection
2. feature importance
3. Correlation matrix with heatmap.

① Filter method

set of all features → selecting the Best Subset → ML algo → Perform

↳ chi square test
↳ Anova test
↳ corr coeff

② Wrapper method

i) forward selection :- first A train
next A & B if good keep
→ A, B, C → go like this

ii) Backward elimination :-
A, B, C, D, E → find which has low impact on our target variable.
→ remove that one.
by chi² test
↓ $p \le 0.05$ imp
$p > 0.05$ → not imp → remove that

③ Recursive feature elimination :-
↳ greedy optimization algo.
→ keeps best performing subset in each iteration.

* These 2 techniques works only if the dataset is small.

③ Embedded Methods

A, B, C, D, E → finds the permutation & comb of all the features

① Univariate Selection

Statistical test can be used to select those features that have the strongest rlnship with the output var.
→ The scikit-learn library provides the SelectKBest class that can be used with a suite of different statistical tests to select a specific no. of features
↳ internally it uses some tests eg:- chi² test.

Types of encoding

① Nominal Encoding
→ One hot encoding
→ one hot En with many categorical
→ Mean Encoding

② ORDINAL ENCODING
→ Label Encoding
→ Target guided ordinal Encoding.

Handling missing values in categorical variables

① Deleting the rows → may miss imp data
② Replace with the most frequent values ↳ imbal dataset
③ Apply classifier algo to predict → good
④ Apply unsupervised ML → good.

Ordinal Encoding / Label encoding

→ Categorical
- Nominal
  - pen
  - areo
  - scale
- Ordinal → order in imp
  - Excellent
  - good
  - Bad.

grades
A, B, C, D

Linear Regression

Cost function

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

↓

minimize

slope this eq is '0'

gradient descent

Global minima

→ Convergence Theorem :

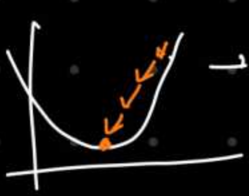
$$\boxed{m_{new} = m_{old} - \frac{\partial [J(m)]}{\partial m} \times \alpha}$$

learning rate 0.001 small.

if our pt is left. points downwards -ve slope.

→ so m = m − (−ve)

= m + (+ve) → moves toward global minima

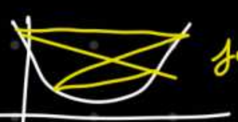


→ If towards right:

→ m = m − (+ve)

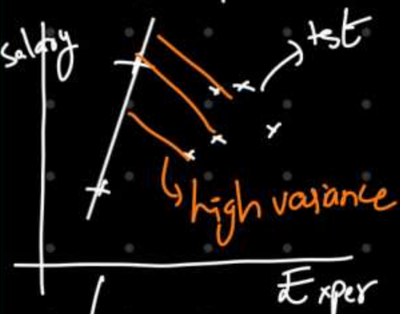
= m − (+ve) → moves left.

→ If α is big →



jumps

Ridge & Lasso Regression



↳ high variance

Cost fn

Sum of residuals

$$\sum_{i=1}^{n} (y - \hat{y})^2 = 0$$

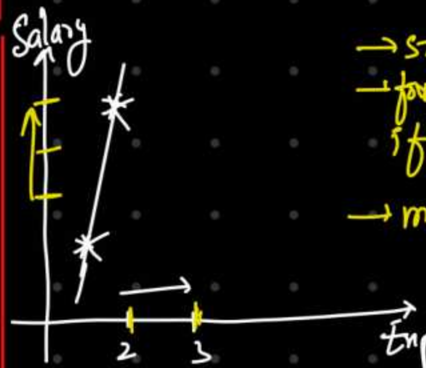
$$\hat{y} = mx + c$$

Over fitting → High Variance

↓ to

Low variance

⇒ Ridge



→ steep

→ for unit ↑ in x,
if high ↑ in 'y'.

→ more steep slope mean
→ overfit

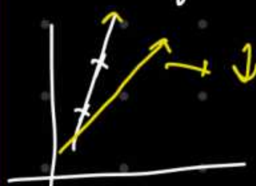
→ Cost function :

$$\boxed{\sum_{i=1}^{n} (y - \hat{y})^2 + \underbrace{\lambda \times (\text{slope})^2}_{} = 0}$$

↳ After the actual value reache '0', but still $\lambda(\text{slope})^2$ is +ve, it even reduces.

$$\underbrace{\sum (y - \hat{y})^2}_{0} + \lambda \times (2)^2$$

$$0 + 4 \rightarrow \text{I will even} \downarrow$$



→ ↓'s. → we are penalizing the slopes which are highly steep.

→ As we ↑ λ, slope will ↓ closer to '0'.

→ lambda vale is selected using CV.

→ $y = m_1 x_1 + m_2 x_2 + c$, ⇒ $\lambda \times (m_1^2 + m_2^2)$

⇒ Created Generalized model.

Lasso : L_2 regularization

→ It not only ↓'s overfitting, but also helps in feature selection.

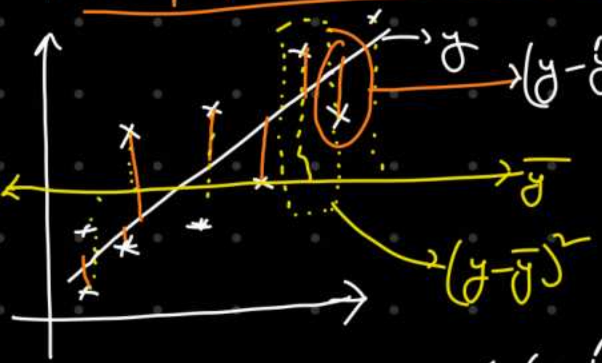
Cost fn : $\sum_{i=1}^{n}(y-\hat{y})^2 + \lambda\underbrace{|\text{slope}|}$

→ $y = m_1a_1 + m_2a_2 + \cancel{m_3a_3} + \cancel{m_4a_4} + c$

→ $\lambda\,|m_1 + m_2 + m_3 + m_4|$ ($m_3 \to 0$, $m_4 \to 0$)

If slopes are closer to '0' remove them.

R square and Adjusted R square



$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}} = \frac{\sum(y-\hat{y})^2}{\sum(y-\bar{y})^2}$$

→ SS_{res} → sum of residuals.

→ R^2 → goodness of the fit

→ 1 → good, 0 → bad

→ If the best fit is worser than avg line, then

$1 - \frac{big}{small} \Rightarrow R^2 = $ -ve value

Adjusted R^2

→ R^2 always ↑'s as we go on ↑'g features even they are not correlated.

→ Adj. R^2 penalizes the features that are not correlated.

$$\boxed{R^2_{adj} = 1 - \frac{(1-R^2)(N-1)}{N-P-1}}$$

R^2 = sample R²
P = no. of predictors
N = sample size

→ if we ↑ features, will be added in denomi. So, R^2_{adj} will ↓.

↳ But if correlated, Numeric multiplied by (N-1) also, R^2 ↑'s. so overall ↑'s.

→ Adj R^2 value always less than or equal to R^2 value.

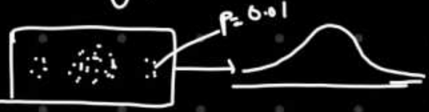
→ Hypothesis Testing
- Null H_0
- Alt H_1

① make assumption

② Collect evidence to prove H_0, if not found reject.

	H_0	H_1 / Truth
do not reject	OK	Type-2 Error
reject	Type-1 Error	OK

⇒ **P-Value** : It is the probability for the Null hypothesis to be true. {if P ≤ 0.05} reject.



If we touch 100 times we only touch that 1 time.

Null hypothesis : Treats everything same or equal.

METRICS For CLASSIFICATION

1) Confusion matrix
2) FPR [Type 1 Error]
3) FNR [T2 Error]
4) Recall [TPR, Sensitivity]
5) Precision (+ve pred val)
6) Accuracy → for balanced
7) F_β score
8) Cohen Kappa
9) ROC curve, AUC score
10) PR curve

Problem → classification

Problem → Class labels
Problem → Probability (0.3, 0.4)

AB = 0.5

1000 records
- 500 yes , 500 no
- 600 400
- 700 300
- 800 200

for imbalanced.

	1	0	Actual Value
Predicted Val 1	TP	FP	→ Type 1
0	FN	TN	
	Type 2		

$\Rightarrow FPR = \frac{FP}{FP+TN}$

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

A ← 900 100 → B

$Acc = \frac{900}{1000} \Rightarrow 90\%$

1) **Precision**: Out of all the predicted +ve how many of them are actual +ve.

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

→ spam detection : FN
→ if spam, predicted not spam
→ if not spam, predicted spam ××
↓
so FP ↓↓↓
⇒ we may miss imp mail

② **Recall**: Out of all +ve values, how many +ve values are predicted.

$$\boxed{Recall = \frac{TP}{TP+FN}}$$

→ cancer → FP
→ if no cancer, predicted cancer, no problem
→ if cancer, predicted no cancer
↓
FN ↓↓↓

⇒ So, +ve guy detected no cancer, big problem.
→ So in cancer detection we need to focus to reduce FN ↓↓.

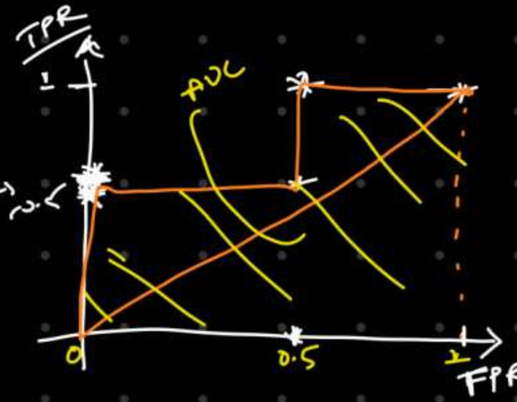
③ $F_\beta = (1+\beta^2)\frac{Precision \times Recall}{Precision + Recall}$

→ $\beta = 1$ → Both preci, Recall imp.
→ $\beta = 0.5$ → FP has higher impact.
→ $\beta = 2$ → FN has higher impact.

Roc & AUC

[0, 0.2, 0.4, 0.6, 0.8, 1]

y	$\hat{y}_{Pred}$	$\hat{y}(0)$	$\hat{y}(0.2)$	$\hat{y}(0.4)$	$\hat{y}(0.6)$	$\hat{y}(0.8)$	
1	0.8	1	1	1	1	0	0
0	0.96	1	1	1	1	1	0
1	0.4	1	1	0	0	0	0
1	0.3	1	1	0	0	0	0
0	0.2	1	0	0	0	0	0
1	0.7	1	1	1	1	0	0



$TPR = \frac{TP}{TP+FN}$

$FPR = \frac{FP}{FP+TN}$

① $\frac{2}{2+0} = 1$
$FPR = \frac{2}{2+0} = 1$

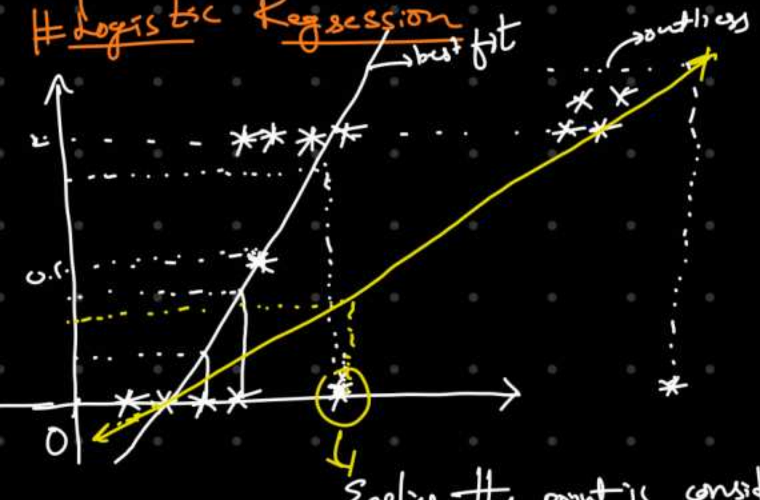
② $\frac{4}{4+0} = 1$
$FPR = \frac{1}{1+1} = 0.5$

③ $TPR = \frac{2}{2+2} = 0.5$
$FPR = \frac{1}{1+1} = 0.5$

④

⑤ $TPR = \frac{0}{0+2} = 0$
$FPR = \frac{1}{1+1} = 0.5$

Logistic Regression



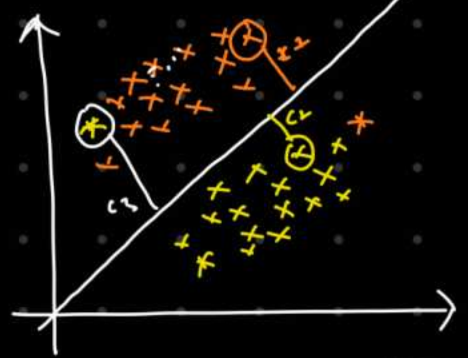
→ ① It will greatly affected by outliers
→ ② also some pts can go > 1
→ ③ So, we use logistic regression
→ we will squash the lines at 0 & 1.

Earlier the point is considered as 1, but after adding outliers, it is consider as 0 with lin regression

⇒ logistic regression → Binary classification.

Logistic Regression

① Geometric Intuition
② Mathematic Intuition



y_i +ve = +1
y_i −ve = −1

$y = mx + c$
$y = \beta_0 + \beta_1 x$
$y = w^T x + b$

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

w is distance from plane

$\dfrac{w^T x + b \to 0}{\|w\|_2}$

$\sum_{i=1}^{n} w_i^T x_i$

Case 1 [Red color]	C2 [yellow]	C3
$y_i = 1 \quad w^T x_i > 0$	$y_i = -1,\ w^T x_i < 0$	$y_i = -1,\ w^T x > 0$
$\boxed{y_i \times w^T x} > 0$	$\boxed{y_i \times w^T x_i} > 0$	$\boxed{y_i \times w^T x} < 0$
correct classif.	correct classifi	wrong classifi

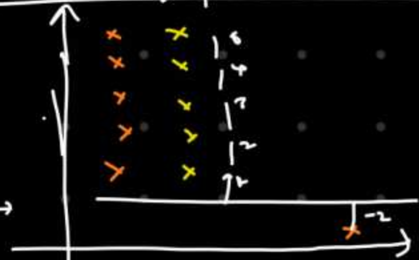
→ Optimization $\max \boxed{\sum_{i=1}^{n} y_i\, w_i^T x_i}$

→ Effect of Outliers :-

$y = -1$
$y = +1$

$= 2+2+2+2+2+2$
$+2+2+2+2+2+2$
-500
$= \underline{-480}$

→



$-1-2-3-4-5-6$
$+1+2+3+4 \ldots$
$+2$
$= \boxed{2}$

→ Seen that the first line is best, but 2nd one getting max for $\boxed{y_i \times w^T x}$ because of outlier.

→ So we introduce sigmoid to avoid this

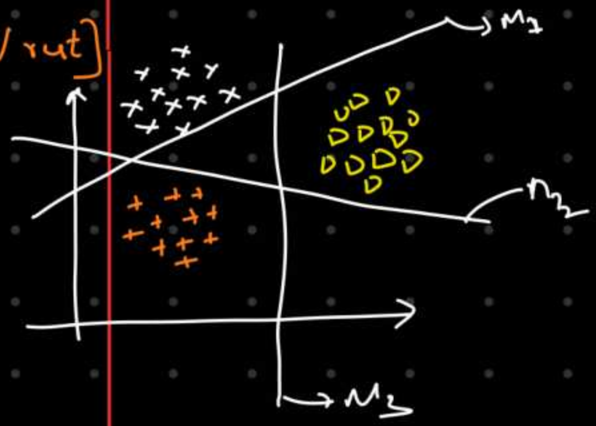
→ after finding $(y_i \times w^T x_i)$, we pass this into another function → Sigmoid function.

→ $\sum_{i=1}^{n} f(y_i \times w_i^T x_i)$ → sigmoid fn transform the values b/w (0,1). So removes the effect of outliers.

$\boxed{\text{Sigmoid fn} = \dfrac{1}{1+e^{-x}}}$

multiclass logistic classification [One V rest]

f_1	f_2	f_3	OP	O_1	O_2	O_3
I_1	I_2	I_3	O_1	+1	−1	−1
I_4	I_5	I_6	O_2	−1	+1	−1
I_7	I_8	I_9	O_3	−1	−1	+1
I_{10}	I_{11}	I_{12}	O_1	+1	−1	−1
I_{13}	I_{14}	I_{15}	O_2	−1	+1	−1
				M_1	M_2	M_3



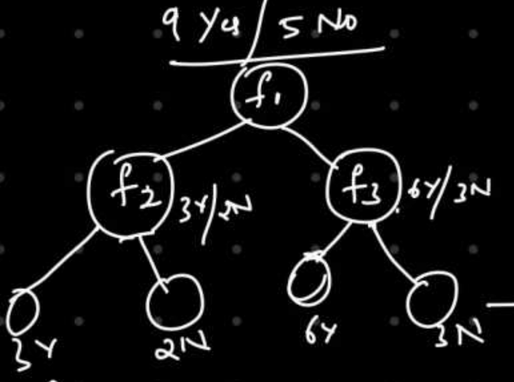
→ OVR
→ new test data
[0.20, 0.25, 0.85]
So, O_3

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

P_+/P_- : % of +ve class / % of −ve class.

Decision Tree Entropy

Entropy :- Measures the purity of split



→ stops splitting at pure split.

→ Entropy → 0 → 1

$\boxed{0 \to \text{best} \quad 1 \to \text{worst}}$

→ $f_1\ f_2\ f_3$ → which feature to select at 1st node.

→ at 1 node 3Y/3No → worst split.

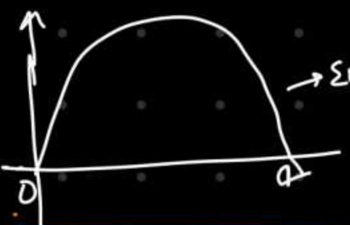
→ at 2 node 3Y/0N → best split.

→ 3Y/2N

S = subset of Training Ex.

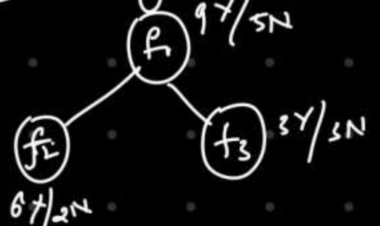
$= -\frac{3}{5}\log_2\left(\frac{3}{5}\right) - \left[\frac{2}{5}\log_2\left(\frac{2}{5}\right)\right]$

$= 0.78$ → entropy.



Information gain [Decision Tree]

① Entropy



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

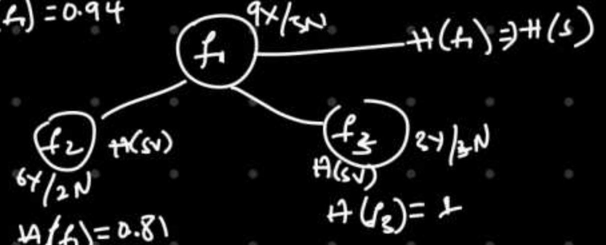
$= -\frac{6}{8}\log\left(\frac{6}{8}\right) - \frac{2}{8}\log\left(\frac{2}{8}\right)$

$= 0.81$

② Information gain

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

$H(f_1) = 0.94$



$Gain(S, f_1) = H(S) - \frac{8}{14}H(f_2) - \frac{6}{14}H(f_3)$

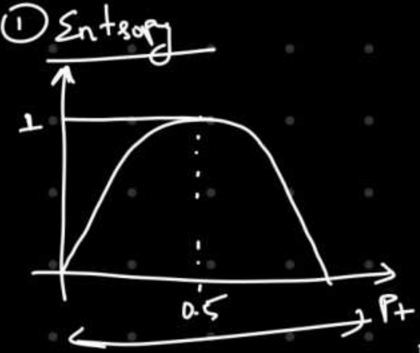
$= 0.94 - \frac{8}{14}(0.81) - \frac{6}{14}(1)$

$= \boxed{0.049}$

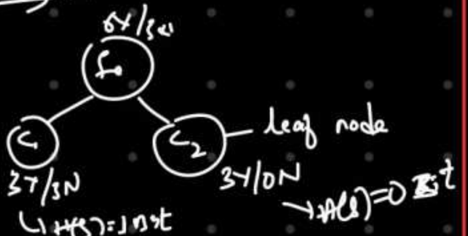
which split have the high information value that split will be selected.

Gini Impurity DT

① Entropy



f_1	f_2	f_3	OP
c_1	D_1		Y
c_2	D_2		Y
			N
			Y
			Y



→ ② GINI Impurity

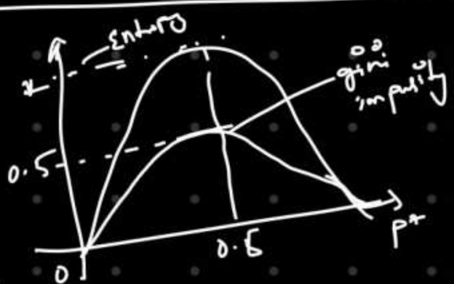
$$\boxed{GI = 1 - \sum_{i=1}^{n}(P)^2}$$

$GI = 1 - [P_+^2 + P_-^2]$

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

$= 1 - (0.25 + 0.25)$

$= 0.5$



(*) gini impurity ranges b/w 0—0.5

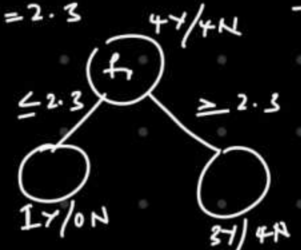
(*) Computationally efficient

(*) takes less time than entropy

Decision Tree Split for Numerical val

f_1	o/p
2.3	Y
3.6	Y
4	Y
5.2	N
6.7	N
8.4	N
9.5	Y
10.2	N

① Sorting all the values
② Th = 2.3



↳ Entropy
↳ Information gain

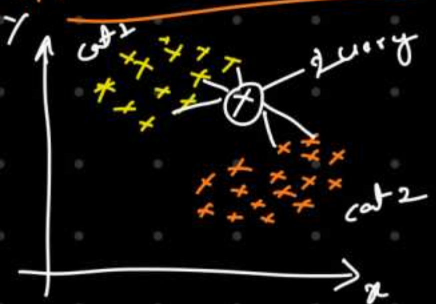
Next for Th = 3.6

↳ Entropy
↳ Info gain

→ Th = 4 calculates for all x_i values till 10.2

(*) After that finds the highest info gain
(*) Then selects that
(*) Disadvantage ÷ Time complexi

K Nearest Neighbour

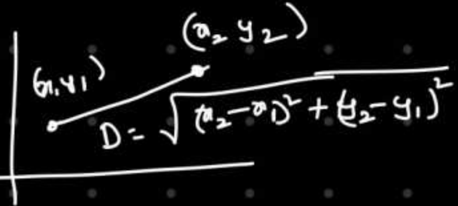


cat 1 → 3
cat 2 → ②

2 dimension

Algo ÷
① k = 5 Neighbour
② We calculate the distance of the neig
k = 5
③ How many nearest neighbours ∈ cat 1 & cat 2.

① Euclidean Distance

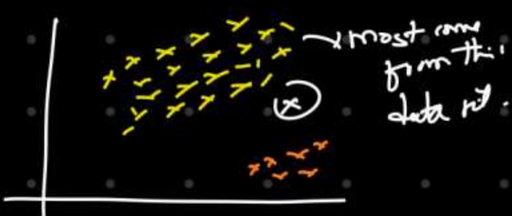


$$D = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

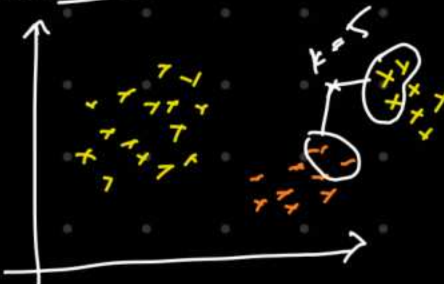
② Manhattan distance

Dataset → 900 Yes, → 100 NO

(*) K-Nearest neibour → biased.



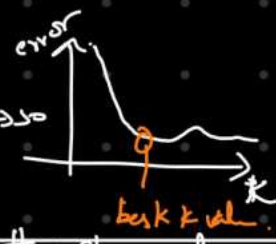
(*) Outliers ÷



→ impacted by outliers.

(*) KNN Regressor

→ finds best 'k' value by elbow method
→ classifier = KNNeighbCl (n_neighbors = ?) → loop in in range(1, 40) → 50



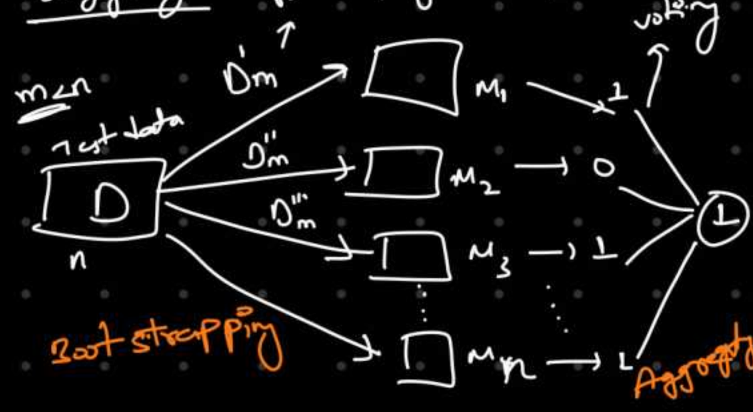
Ensemble Techniques ÷

Bootstrap aggrigation

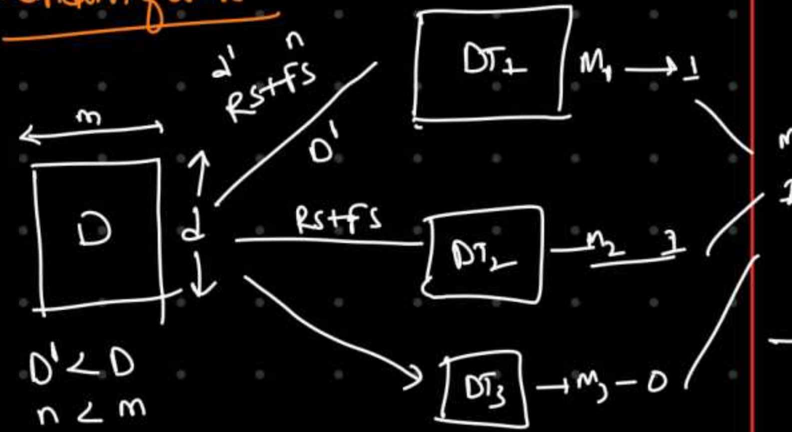
Bagging
① Random forest

Boosting
① Ada boost
② Gradient boost
③ Xg boost

Bagging — row sampling with replacement



Random forest



$D' < D$
$n < m$
$d' < d$

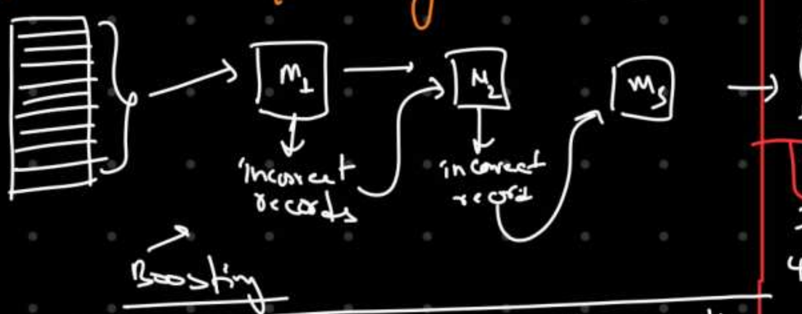
DT

① Low Bias high variance.

② RF → Row sampling → f sampling
↳ so RF becomes expert so we will get low variance.

→ for regression → mean of all O/P.

Ada boost { Boosting Technique }



1 features → 3 — stumps → DT with 1 depth

↳ Entropy / gini impurity
↳ information gain

if this has less entropy
more info gain
this is 1st weak learner.

Adaboost step ① $w = \frac{1}{n} = \frac{1}{5}$

①	f_1	f_2	f_3	O/P	sample wt.
1					1/5
2					1/5
3					1/5
4					1/5
5					1/5

incorrect record

step 2

calculate total error.

$\boxed{TE = \frac{1}{5}}$ sum of all error weights.

step 3 performance of stump

$$\boxed{\text{performance of stump} = \frac{1}{2}\log_e\left(\frac{1-TE}{TE}\right)}$$

$$= \frac{1}{2}\log_e\left(\frac{1-1/5}{1/5}\right)$$

$= 0.896$ (Eg).

	initial	updated wt	normalized	bucket
1	1/5	0.05	0.07	0–0.07
2	1/5	0.349	0.513	0.07–0.5[illegible]
3	1/5	0.05	0.07	0.58–[illegible]
4	1/5	0.05	0.07	0.65–[illegible]
5	1/5	0.05	0.07	
		sum		

step 4 update weights [↑ incorr, ↓ right]

Performance = 0.895

$$\boxed{\text{New sample wt} = wt \times e^{\text{performance}}} \text{ for incorr.} \rightarrow = \frac{1}{5} \times e^{+0.895} = 0.349$$

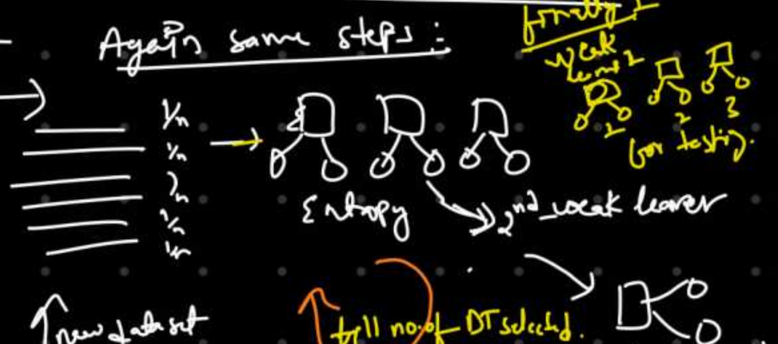
$$\boxed{\text{for correct} = wt \times e^{-\text{perfor}}} \rightarrow = \frac{1}{5} \times e^{-0.895} = 0.05$$

→ Next step is to create new dataset based on normalized wt & buckets.

f_1 f_2 f_3 OP

we will select mostly only the records with high error by algo. & create new data set.

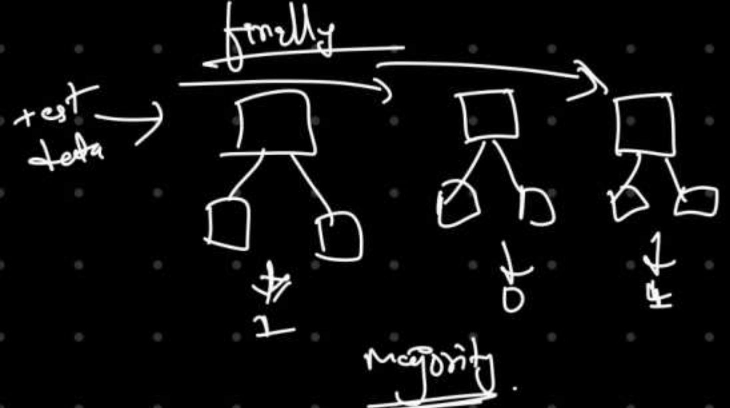
Again same steps :



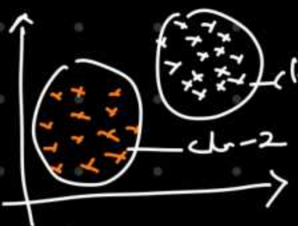
↑ new dataset

till no. of DT selected. finally error is less

normalized & bucket ⟸ update wts ⟸ performance of stump



Kmeans clustering:

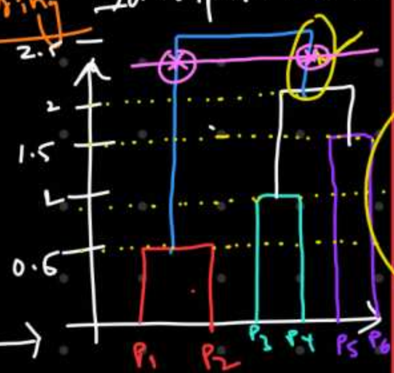


↳ Kmeans

① Algorithm
② metrics
③ → euclidean
→ manhattan
④ Elbow method
select the 'k' value.

① K-value → centroids k = 2
② Initialize the centroid randomly
③ Select the group & find the mean.

Hierarichal clustering — unsupervised.

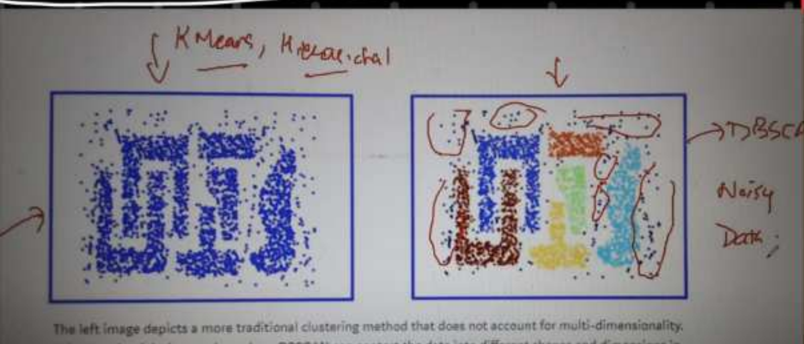


Dendogram.

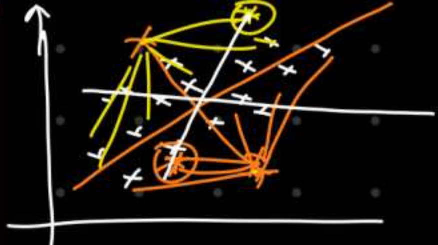
↳ we need to select the longest vertical line without any line cutting it.

↳ * draw a line, no of cuts = no. of clusters.

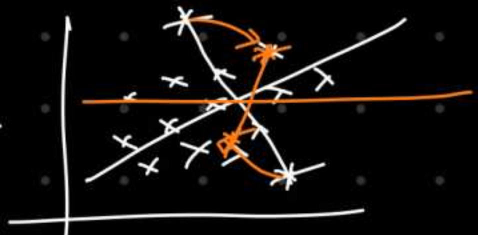
↳ so, 2 clusters.



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.



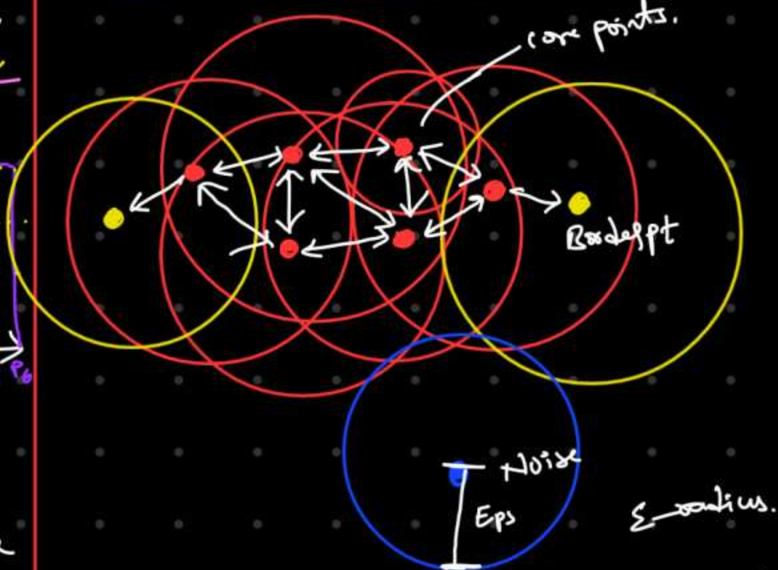
find the distance
→ then mean of both clusters
→ move centroid to them.



$$WCSS = \sum_{i=1}^{n} (c_i + x_i)^2$$

abrupt decrease.

Density Based Spacial clustering applications with Noise [DBSCAN] — unsupervised



→ Red core pts: min 4 pts inside a circle.

→ Yellow Borderpts: still part of clusters, no 4 pts, But atleast 1 Borderpt inside circle.

→ Blue Noise: no 4 pts, no Border pt

Advantages:
① great at creating clusters of high & low density.
② great at handling outliers.

⇒ Disadvantages:
① Doesn't work well when clusters of similar densities.
② struggles with very high dimensional data.

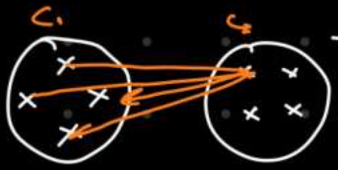
Silhouette [clustering]

① C_i

→ finds the distance within the cluster pts.

$\rightarrow a_i = \frac{\sum_{j \in C_i, i \neq j} d(i,j)}{|C_i| - 1}$

②



→ finds the dist. b/w one cluster pt to all pts of other cluster

$\rightarrow b_i = \min_{k \neq i} \frac{\sum_{j \in C_k} d(i,j)}{|C_k|}$

③ $a_i <<< b_i$ for correct clustering.

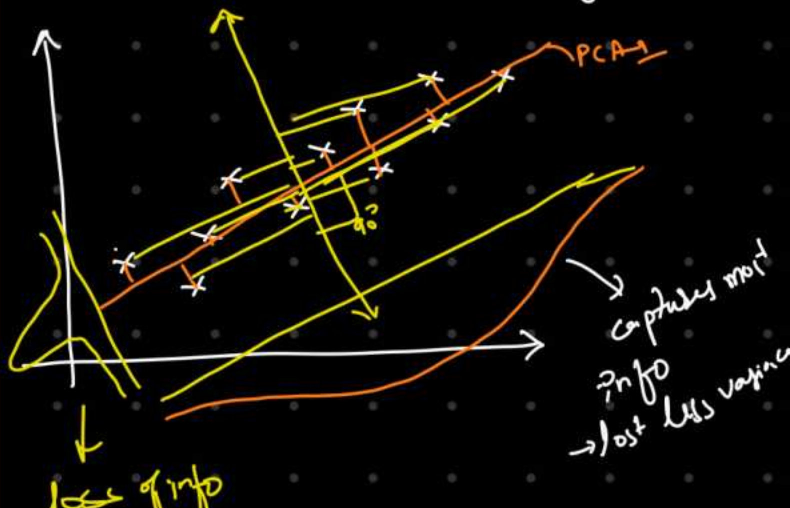
if $b_i > a_i$ → wrong clustering.

④ finally $S_i = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$, if $|C_i| > 1$

S_i is b/w −1 to 1 //

Principal Component Analysis (PCA)

→ PCA isn't exactly a fully ML algo, but instead an unsupervised learning algo.



→ so we select PCA1.

→ 2D → 1D.

Types of cross Validation

1000 records

70% train } train test split random state = 0

30% test

↓ randomly split data

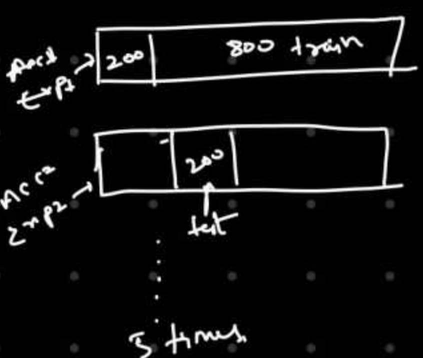
↓ 85% acc

if again change random state accuracy changes.

① K fold CV

$\frac{1000}{5} = 200$

K = 5

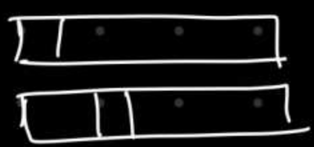


* Disadvantages :

① if in test dataset, may contain all data of 1 type of binary classification.

② ↓ so leads to biased.

② Stratified cross Validation K = 5



* → It will make sure that no. of instances of each class will contain in a good proportion in training and testing datasets

③ Time series CV : for stock market etc.

						O/P
Day 1	Day1	Day2	D3	D4	D5	D6
Day 2	D2	D3	D4	D5	D6	D7
Day 3	D3	D4	D5	D6	D7	D8
⋮						
Day 7						

Baye's Theorem :

① Conditional Probability

② Independent Events

③ Dependent Events

$P(A)=\frac{2}{5}$ $\quad P(B/A)=\frac{1}{4}$

$P(A\cap B)=\frac{2}{5}\times\frac{1}{4}=\frac{1}{10}$

$$\boxed{P(B/A)=\frac{P(A\cap B)}{P(A)}} \qquad P(B/A)=\frac{1}{4}$$

$$=\frac{P(A)\times P(B)}{P(A)}=\frac{\frac{2}{5}\times\frac{1}{4}}{2/5}=\frac{\frac{1}{10}}{2/5}=\frac{1}{4}$$

Baye's Theorem :

$$P(A/B)=\frac{P(A\cap B)}{P(B)} \quad ; \quad P(B/A)=\frac{P(B\cap A)}{P(A)}$$

$$P(A\cap B)=P(A/B)\times P(B) \quad ; \quad P(B\cap A)=P(B/A)*P(A)$$

But $P(A\cap B)=P(B\cap A)$

so $\Rightarrow$ $P(A/B)\times P(B)=P(B/A)\times P(A)$

$$\Rightarrow \boxed{P(A/B)=\frac{P(B/A)*P(A)}{P(B)}}$$

outlook

Outlook

	Yes	No	P(Y)	P(N)
Sunny	2	3	2/9	3/5
Overcast	4	0	4/9	0/5
Rainy	3	2	3/9	2/5
Total	9	5	100%	100%

temp

	Yes	No	P(Y)	P(N)
HOT	2	2	2/9	2/5
MILD	4	2	4/9	2/5
COOL	3	1	3/9	1/5
Total	9	5	100%	100%

		P(Y) & P(N)
Yes	9	9/14
No	5	5/14
Total	14	100%

$$P(Yes/Today)=\frac{P(Sunny/Yes)\times P(HOT/Yes)\times P(Yes)}{P(Today)}$$

$\rightarrow$ Today $(\overset{x_1}{Sunny}, \overset{x_2}{HOT})$

$P(Y/Today)\propto\frac{2}{9}\times\frac{2}{9}\times\frac{9}{14}=0.031$

$P(N/Today)\propto\frac{3}{5}\times\frac{2}{5}\times\frac{5}{14}=0.0857$

$P(Yes)=\frac{0.031}{0.031+0.0857}$

≈ 0.27 } Normalizing.

$P(N)=0.72$

for classification of text

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

$\rightarrow$ stopwords
$\rightarrow$ stemming
$\rightarrow$ BOW
$\rightarrow$ TFIDF
$\rightarrow$ NLP

Naive Baye's Classifier

Data set $\div$ $x=\{x_1, x_2, x_3, x_4 \ldots x_n\}$ $\{y\}$

f_1 $\quad f_2$ $\quad f_3$ $\quad y$

x_1 $\quad x_2$ $\quad x_3$ $\quad y_1$

$$P(y/\{x_1,x_2\ldots x_n\})=\frac{P(x_1/y)P(x_2/y)\ldots P(x_n/y)\times P(y)}{P(x_1)P(x_2)\ldots P(x_n)}$$

$$=\frac{P(y)\times\prod_{i=1}^{n}P(x_i/y)}{\underbrace{P(x_1)P(x_2)\ldots P(x_n)}_{const}}$$

$$P(y/x_1,x_2,x_3\ldots x_n)\propto P(y)\prod_{i=1}^{n}P(x_i/y)$$

Yes:-0.7
No= 0.3

$$\boxed{y=\arg\max_y P(y)\prod_{i=1}^{n}P(x_i/y_i)}$$

f_1	f_2	f_3	f_4	..
The	food	was	bad	O/P
1	1	1	0	1
1	1	0	1	0
0	1	0	1	0
0	1	1	0	1
0	0	0	1	0

sentence $\rightarrow$ Good
$\rightarrow$ Bad

$P(y=yes/sentence)$

sentence $[x_1, x_2, x_3, x_4$

$P(y/x_1,x_2,x_3,x_4\ldots x_n)$

$$\propto P(y)\prod_{i=1}^{n}P(x_1/y)P(x_2/y)\ldots P(x_n/y)$$

$$=P(y=yes)\times P(The/y)P(food/y)\ldots P(bad/y)$$

$$=\frac{2}{5}\times\frac{\cancel{2}}{\cancel{2}}\times\frac{\cancel{2}}{4}\times\frac{\cancel{2}}{\cancel{2}}$$

$$=1/5=0.01$$

$$P(N/x_1,x_2,x_3,x_4)=P(x_1/N)\times P(x_2/N)P(x_3/N)\,P(N)$$

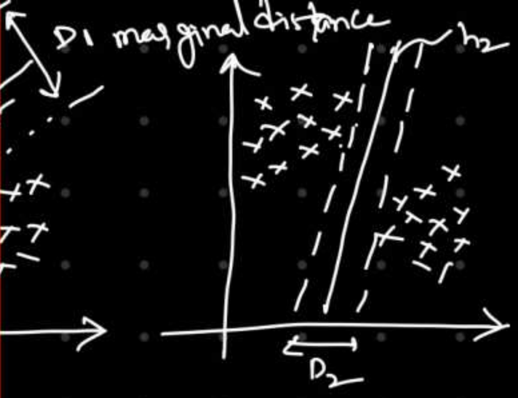
$$=\frac{3}{5}\times\frac{1}{2}\times\frac{2}{4}\times \approx 0.03$$

Normalize $\frac{0.01}{0.01+0.03}=\frac{0.01}{0.04}\approx\frac{1}{4}=\overset{Y}{0.25}$

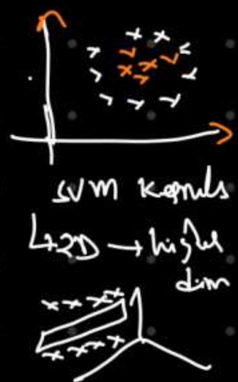
$N=0.75$

Support Vector Machine :

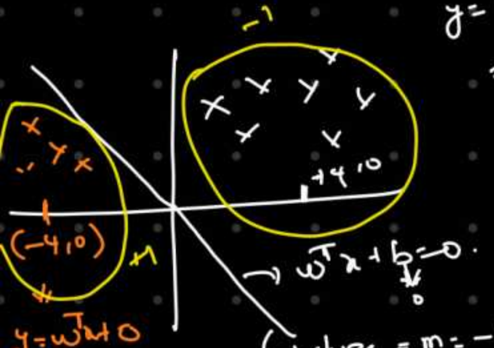
1. Support Vectors
2. Hyper planes
3. Marginal distance
4. Linear separable
5. Non-linear separable



$D_2 << D_1$



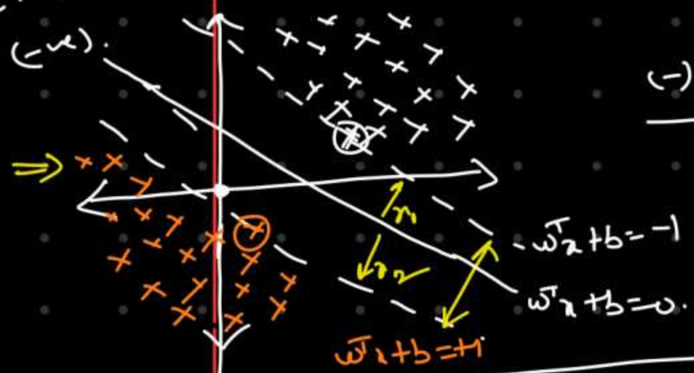
$y = \begin{bmatrix} -1 \\ 0 \end{bmatrix} [4\ 0]$
$= -4$ (-ve).



$y = w^T x + 0$
$= \begin{bmatrix} -1 \\ 0 \end{bmatrix} [-4\ 0]$
$= 4$ //

slope $= m = -1$

All pts below plane +ve //



$w^T x_1 + b = -1$
$(-)\ w^T x_2 + b = 1$

$w^T(x_2 - x_1) = 2$

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

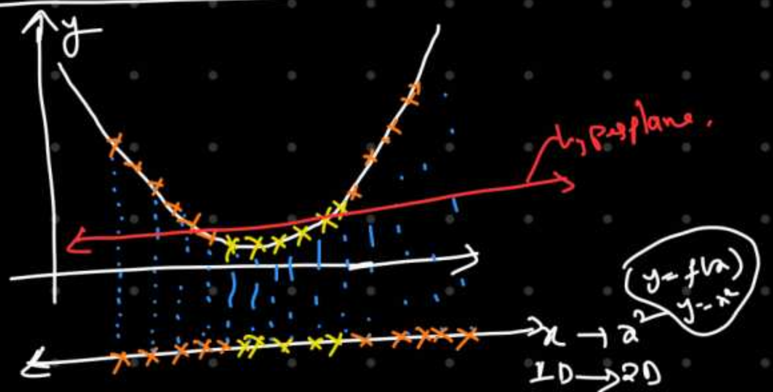
$w^*, b^*\ \max\left(\frac{2}{\|w\|}\right)$ st $y \begin{cases} +1 & w^T x + b \geq 1 \\ -1 & w^T x + b \leq -1 \end{cases}$

optimizers.

$\Rightarrow \boxed{y_i * w^T x_i + b_i \geq 1}$ //

$$\boxed{w^*, b^* = \min \frac{\|w\|}{2} + C_i \sum_{i=1}^{n} \xi_i}$$

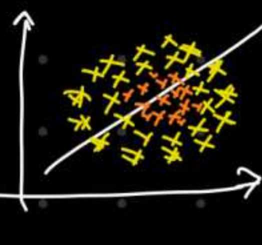
Regularization — how many errors — value of the error.



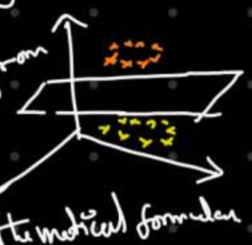
SVM Kernels :

1. Soft margin
2. Hard margin

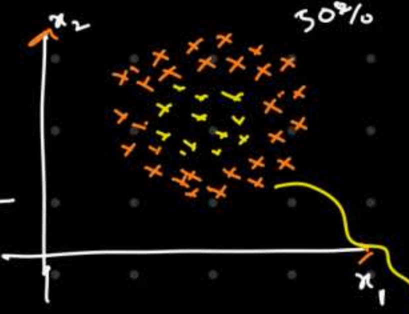
1. polynomial kernels
2. RBF kernels
3. Sigmoid kernels.



Transformation from lower dim to higher dim by mathematical formulas



1. Polynomial kernels :



$x_1 \quad x_2 \quad y_0 \quad y = f(x)$

$f(x_1, x_2) = (x_1^T \cdot x_2 + 1)^d$

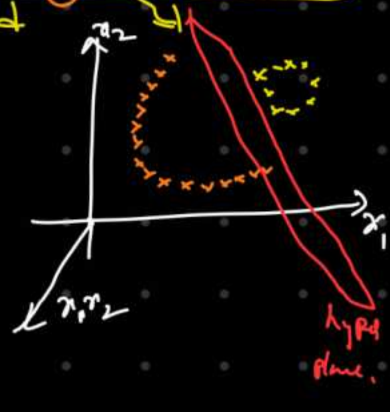
$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \cdot [x_1\ x_2]$

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

$\boxed{x_1 \quad x_2 \quad y_0} \Rightarrow \boxed{x_1 \quad x_2 \quad y_0 \quad x_1^2 \quad x_1 x_2 \quad x_2^2}$

2d → 5d

SVC kernel = "poly"
= "rbf"



Gradient Boosting (actu − pred)

Exp	Deg	Sal (act)	$\hat{y}$ (pred)	R_1	R_2	R_3
2	BE	50K	75	-25	-23	
3	master	70K	75	-5	-3	
5	master	80K	75	5	3	
6	Degree	100K	75	25	2	

① step① Base model 1 — for doing avg.

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

② compute residuals
(1) Error, pseudo Residuals.

③ Construct DT $\{x_i, R_1\}$

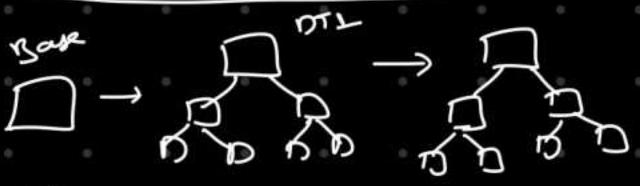
$= 75 + (-23)$
$= 52$ close to 50K [overfitting]

so add learning rate. α 0 to 1

$= 75 + \alpha(-23)$
$= 75 + 0.1(-23)$
$= 75 - 2.3 \Rightarrow 72.7$

$\Rightarrow f(x) = h_0(x) + \alpha_1 h_1(x) + \alpha_2 h_2(x) + \alpha_3 \ldots + \alpha_n h_n(x)$

$$\boxed{F(x) = \sum_{i=1}^{n} \alpha_i h_i(x)}$$



all the above is eg purpose
below is actual pseudo algo.

Exp	Deg	Sal
2	BE	50
3	PHD	70
4	Mtech	60

Pseudo algo

① Initialize model with const value.

$$F_0(x) = \arg\min_{\gamma} \left(\sum_{i=1}^{n} L(y, \gamma)\right)$$

$Loss = \sum_{i=1}^{n} \frac{1}{2}(y-\hat{y})^2 \Rightarrow \frac{1}{2}(50-\hat{y})^2 + \frac{1}{2}(70-\hat{y})^2 + \frac{1}{2}(60-\hat{y})^2$

$\Rightarrow \frac{\partial}{\partial Loss}$ for min $\Rightarrow \frac{2}{2}(50-\hat{y}) + (70-\hat{y}) - (60-\hat{y})$

$\Rightarrow \frac{180}{3} = 60 \Rightarrow \boxed{\hat{y} = 60}$ → initialize base model with this value.

② Iterate $M = 1$ to M (no. of trees)

Compute pseudo residuals

$$r_{im} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial F(x_i)}\right] \quad \text{for } i = 1 \text{ to } m$$

$F(x) = F_{m-1}(x)$

$Loss = \frac{1}{2}(y-\hat{y})^2$

$\frac{\partial L}{\partial \hat{y}} = (y-\hat{y})(-1)$

$\boxed{\frac{-\partial L}{\partial \hat{y}} = (y-\hat{y})} = r_{im} = -\left[\frac{\partial L(y, f(x_i))}{\partial F(x_i)}\right]$ (i, 60)

$\frac{-\partial L}{\partial \hat{y}} = \frac{\partial L(y, \hat{y})}{\partial \hat{y}} = \underbrace{(y-\hat{y})}$ we need to compute this

Sal (y)	$\hat{y}$
50	60
70	60
60	60

$(y-\hat{y})$

$r_{im} \Rightarrow$ i, model no.

$r_{11} = -10$
$r_{21} = 10$
$r_{31} = 0$

x_i → indep, r_{im} → dep

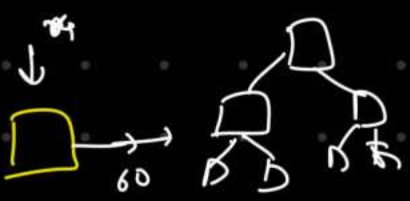
$\Rightarrow$ Fit a Base Learner $h_m(x)$ $\{(x_i, r_{im})\}$ ip op

③ $\boxed{\gamma_m = \arg\min_{\gamma} \sum_{i=1}^{n} L(y_i, F_{m-1}(x_i) + \gamma)}$

$= \sum_{i=1}^{n} \frac{1}{2}(y_i - (60 + \hat{y}))^2$

$\rightarrow \frac{\partial}{\partial L} \rightarrow$ find $\hat{y}$ → repeat step ① & ②

④ Update model $\boxed{F_m(x) = F_{m-1}(x) + \gamma_m h_m(x)}$



$60 + (0.1)(-10)$ Learning rate 0−1
$= 60 - 1$
$= 59$

2nd record $60 + (0.1)(10)$
$= 61$

XGBoost

Salary	Credit	Approval	Res
<=50k	B	0	-0.5
<=50k	G	1	0.5
<=50k	G	1	0.5
>50k	B	0	-0.5
>50k	G	1	0.5
>50k	N	1	0.5
<=50k	N	0	-0.5

→ output is 1 or 0, so $\frac{1+0}{2} = 0.5$ - Base model probability.

Approval - Base model prob.

① Construct tree with Root

② Calculate Similarity weight $= \frac{\sum (\text{Residuals})^2}{\sum[\text{Pr}(1-\text{prob})+\lambda]}$

③ Calculate Gain

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

Similarity wt:

$= \frac{(-0.5+0.5+0.5-0.5)^2 \to 0}{2\text{Pr}(1-\text{Pr})+\lambda}$

$= 0.$

Similarity wt:

$= \frac{(0.5)^2}{0.5(1-0.5)+0.5(1-0.5)+0.5(1-0.5)}$

$= \frac{0.25}{3 \times 0.5 \times 0.5} = 1/3$

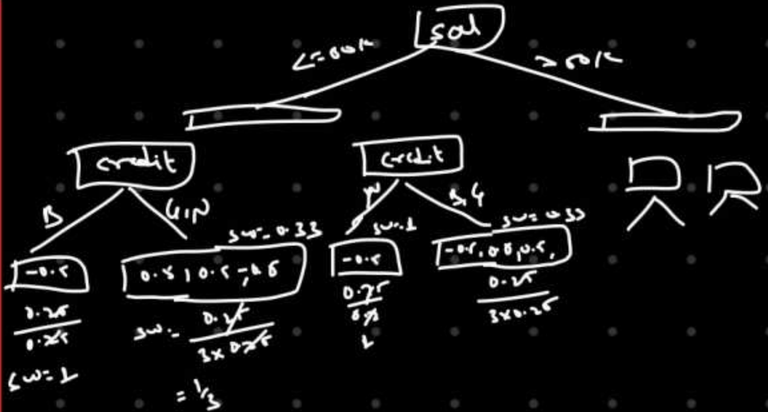
→ sw for root:

$sw = \frac{(0.5)^2}{7 \times 0.5 \times 0.5} = 1/7 \approx 0.143$

⇒ Gain = sw left + sw right - sw of root

$= 0 + 0.33 - 0.14$

$= 0.21$



gain: 1 + 0.33 - 0

= 1.33

* for post pruning → cover value → pr(1-prob) = 0.25

If gain < 0.25, cut that DT there..

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

$= \log\left(\frac{0.25}{0.25}\right) = \log 1 = 0$

↳ output is '0' for base model.

learning rate

$\sigma(0 + 0.1(1))$ sw.

$\sigma[0.1] \Rightarrow \frac{1}{1+e^{-0.1}} = 0.6.$ new probabilities computed for all rows

Sal	Crd	App	Res	New pr	New Res
				0.6	

Appr - new pr

Then for new Residual compute the same process of DT → splitting → sw → gain → new Pr.

XGBoost for Regressor:

Exp	Gap	Sal	Res1	O/P
2.5	Y	40k	-11	
2.5	Y	42k	-9	
3.0	N	52k	1	
3.5	N	60k	9	
4	Y	62k	11	

→ Res1 ⇒ Sal - Base model - Avg(Sal) $\frac{40+42\ldots}{5} = 51k$

→ Base model = 51k.

* Similarity wt = $\sum(\text{Residuals})^2 / \text{no of residuals} + \lambda$

51k $\boxed{\text{Base Model}}$ + ?

−11, −7, 1, 9, 11

$\boxed{\text{Exp}}$ → $SW = \frac{1}{6} = 0.16$

λ = 1

≤ 2.5 : $\boxed{-11}$

$SW = \frac{121}{2} = 60.1$

> 2.5 : $\boxed{-9, 11, 9, 11}$

$SW = \frac{144}{5} = 28.8$

Gain = 60.1 + 28.8 − 0.16
= 88.9 − 0.16 ≈ 82.84

↳ we calculate gain is only to confirm which split we should consider.

↳ after confirming → calculate output - avg

gap

Y | N

× not go this split

only 1 value

s = root node. no split.

OP = −11

x>7 | NO

−9, 11 | 7, 9

OP = $\frac{2}{2}$ = 1 | OP = 5

↳ take records and calculated residuals based on this tree.

now.

output = 51 + (0.5) [−11] for 1st record.

α

= 51 − 5.5

= 45.5 // → 1st ...

↳ records based on which split they belong.

Sal	gap	sal	Res 1	new OP	Res 2

Y sal − new OP.

↓

again repeate

↓ DT 2.

⇒ Base Model + $\alpha_1(T_1) + \alpha_2(T_2) + \cdots + \alpha_n(T_n)$