→ lm.score(X_test, y_test) – r^2 score
→ mse(y_test, y_pred)
[LR]
↳ sk.learn.metrics import mean_squared_error as mse

$y = \beta_0 + \beta_1 x + \varepsilon$ — ε: error term; β_0: intercept

residue = $(y_i - \hat{y}_i)$

sklearn.model_selection import train_test_split

sklearn.linear_model import LinearRegression

Given eq → $y = \beta_0 + \beta_1 x$

method of least squares → min of error
↓
$\Sigma (y_i - \hat{y}_i)^2$

$$b_1 = \frac{\Sigma (x_i - \bar{x})(y_i - \bar{y})}{\Sigma (x_i - \bar{x})^2}$$

$$b_0 = \bar{y} - \beta_1 \bar{x}$$

sample corr coeff
$r_{xy} = (\text{sign of } b_1)\sqrt{r^2}$.

* SST → Total sum of squares
  SSR → Sum of squares due to regression
  SSE → Sum of squares due to error.

↳ SST = SSR + SSE

$$\Sigma (y_i - \bar{y})^2 = \Sigma (\hat{y}_i - \bar{y})^2 + \Sigma (y_i - \hat{y}_i)^2$$

r^2 → coeff of Determination | $r^2 = \frac{SSR}{SST}$

$$1 = \frac{SSR}{SST} + \frac{SSE}{SST} = r^2 + x = 1$$

$$r^2 = 1 - x$$

$$r^2 = 1 - \frac{SSE}{SST} \rightarrow 1 - \frac{\Sigma (y_i - \hat{y}_i)^2}{\Sigma (y_i - \bar{y})^2}$$

Customized acc + optimizer

Spectogram

Signal Spectrogram Plot Python

~~Spiral~~

$MSE \rightarrow \frac{1}{n} \Sigma (y_i - \hat{y}_i)^2$

Date.....

Adjusted R^2 → used of MLR

why? → As we keep on adding new features, the R^2 value keeps on ↑. even if the added feature are not correlated to the Target variable.

Adj R^2 → It penalizes attributes that are not contributing towards the Target var.

$$1 - \left[\frac{(1-R^2)(N-1)}{N-p-1}\right]$$

- N → sample size. [No. of samples]
- p → No. of predictors

→ only ↑ when the added Ind var is correlated/ significant and affects the target var.

Adj $R^2 \leq R^2$.

Four Assumptions of LR.

① **Linear Relationship**

→ There should be a linear relⁿ B/w x and y.

→ ◆ How to determine if the Assumption is met?
→ Create a scatter plot (x vs y) → points in plot could fall along a straight line.

→ ◆ if Assumption is violated?
→ Apply a non-linear transformation to the variables [log, sq root, reciprocal]

Spiral

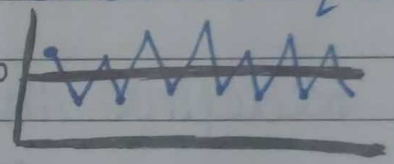
Date.....

→ Add Another Independent var to the model.
[if parabolic shape → might add x^2 as additional Independent variable]

2] Independence
→ residuals are Independent → No correlation/pattern among consecutive residuals.

→ ◆ How to determine if the assumption is met?
→ look at residual time series plot [residue vs time]
→ most of residual autocorrelations should fall within the 95% confidence Bands around zero, which are located at about +/- 2 - over the sq root of n (n = sample size)

±2 over √n



→ ◆ if assumption is Violated?
→ [1] For +ve serial correlation → Consider adding lags of the dependent/Independent variable, to the model

[2] for -ve serial correlation → check if none of your variables are overdifferenced.

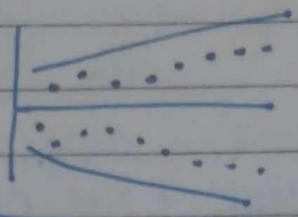
[3] for seasonal correlation → adding seasonal dummy variables to the Model.

3] Homo scedasticity
→ residuals have constant variance at every level of x.
↳ If violated → Hetro scedasticity.
↳ ↑ variance of the coeff → declares the term significant Spiral when it is not.

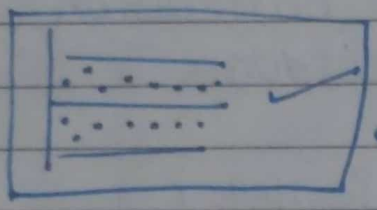
Date.....

→ How to check if assumption is met.

→ fitted value Vs Residual plot [scatter plot]



× every level of x the, Residual gets more apart (↑ in variance)



→ How to fix, if violated.

① Transform the dependent variable. → log of dependent var.

② Redefine the dependent variable → use rate, rather than raw value.

③ use weighted regression → assigns a weight to each data point Based on the variance of its fitted value.

→ gives small weights to DP that have higher variance.

4] NORMALITY

→ residuals are normally Distributed.

1) How to check.

① Using Q-Q plots

Date.....

if the points on the plot follow/form a straight diagonal line, then assumption is met.

- if violated

1) verify that outliers aren't having a huge impact on the Distribution.

2) non-linear tranformation on variables.

x x x

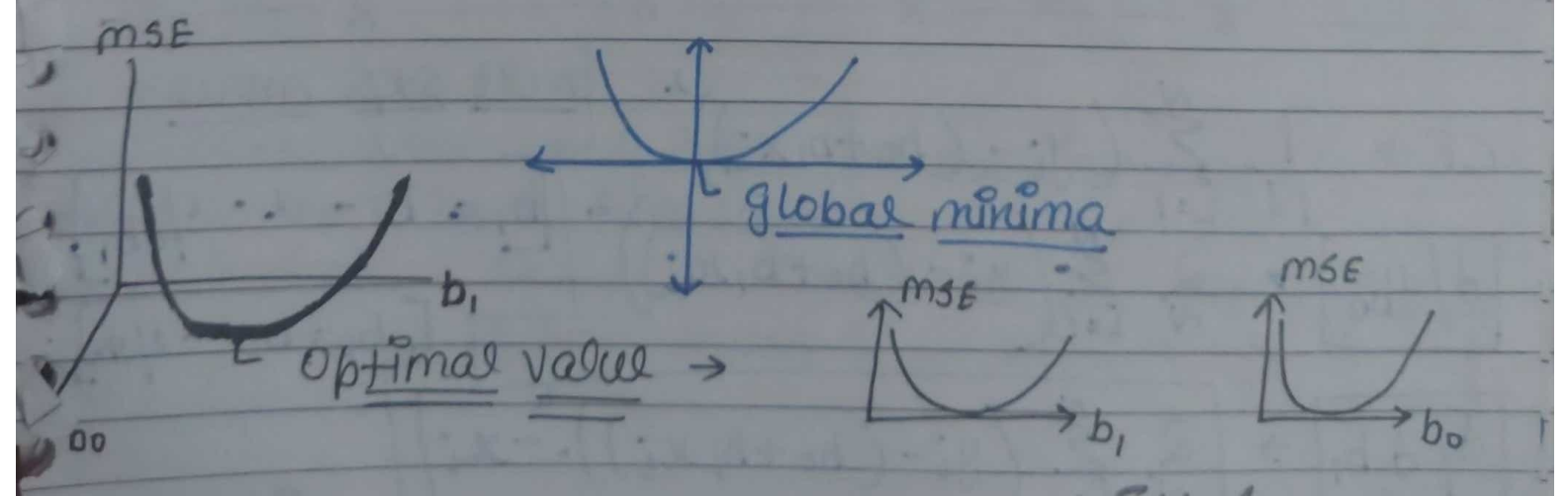
Cost func^n and gradient descent.

↳ find optimal values of β_0 and β_1.

minimum error ⟷ global minima [values of β_0 and β_1 where error is minimum → global minima].

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

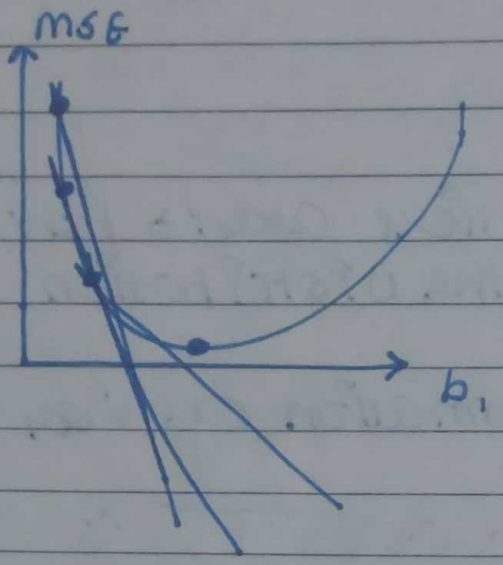
↳ cost func^n → $\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y_i})^2 \rightarrow \frac{1}{N}\sum_{i=1}^{N}(y_i - (b_0 + b_1 x_i))^2$



Spiral

Date.....

• **Gradient Descent** → Optimization Algo that helps our cost functh to reach optimal soln (values with min error).



① — **Slope at every point** → To get the direction where we want to move

→ Changing value of b_0 and b_1, so that we can get values with min error → optimal value.

✱ No. of steps / the method → Learning Rate.

→ High LR → Overshoots → miss the optimal value.
→ Small LR → more steps but will reach the optimal value.

② To take the next step → **Learning Rate** (α).

x ——— x ——— x ——— x ———

$$CF \rightarrow \frac{1}{N}\sum_{i=1}^{N}\left(y_i - (b_0 + b_1 x_i)\right)^2$$

$$\boxed{d/db_0} \rightarrow \frac{2}{N}\sum_{i=1}^{N} -\left(y_i - (b_0 + b_1 x_i)\right)^2$$

$$\boxed{d/db_1} \rightarrow \frac{2}{N}\sum_{i=1}^{N}\left(y_i - (b_0 + b_1 x_i)\right)^2 \cdot -x_i$$

Next step move →

$$\boxed{b_1 = b_1 - \alpha \cdot d/db_1}$$

$$\boxed{b_0 = b_0 - \alpha\, d/db_0}$$

Spiral

Date.....

- High Training accuracy → low Bias (Train data error)
- low Test accuracy → High Variance (Test Data error)

Overfitting → model captures noise too - makes that significant → every pattern and even noise/errors.

↳ low Bias and High variance

Underfitting → model fails to learn from Training data set → fails to capture every pattern or significant patterns.

↳ High Bias and low variance

Best fit → low Bias and low variance.

KNN

- measure Distance of Points from the test Data
- Sorts them in ↑ measure
- depending on No. of K, Considers K Data points
- majority class from those of is assigned to Test Data.

→ Distance Based learning Algo → Feature Scaling Date.....

① Choose value of K.

① Should be odd → Not multiple of no. of classes.

② Error Curves → (curves of error of Training and Testing Data)
└ Test error should be low.

```
Knn = KNeighbors Classifier (n_neighbors = K)
Knn.fit(X_train, Y_train).
y_pred1 = Knn.predict(X_train)
error1.append(np mean(y_train != y_pred1))
y_pred2 = Knn.predict(X_test)
error2.append(np mean(y_test != y_pred2))

plt.plot(error1).
plt.plot(error2).
```

[Distance → $\sqrt{(X_2-X_1)^2 \dots}$].

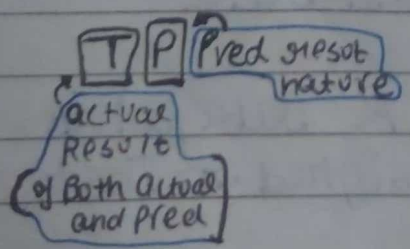
• Performance Metrics

① FPR ② FNR ③ Accuracy ④ Recall
⑤ Precesion ⑥ F-Beta

⑦ Cohen's Kappa Stats

X X X

• Confusion Matrix :



pred values ↓ \ Actual values →	1	0
1	TP	FP → Type1 error (FPR)
0	FN ↑ Type 2 error [FNR]	TN

Spiral

Date.....

$$FPR = \frac{FP}{TN+FP} \text{ (negative)}$$

$$FNR \rightarrow \frac{FN}{TP+FN} \text{ (positive)}$$

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

$$TNR \rightarrow \frac{TN}{TN+FP}$$

★ → TPR and TNR should be high.
FPR and FNR should be Low.

- Accuracy $= \frac{TP+TN}{N}$
- Precesion $\rightarrow \frac{\text{True +ve}}{\text{pred +ve}} \rightarrow \frac{TP}{TP+FP}$

→ Out of Total predicted +ve results how many results were actual +ves.

- Recall $\rightarrow \frac{\text{True +ve}}{\text{Actual +ve}} \rightarrow \frac{TP}{TP+FN}$

→ Out of Total actual +ve, how many +ve did we predict correctly.

★ When Prec and Recall Both are Imp, for that we use F-Beta

↳ $\frac{(1+\beta^2)(P \times R)}{\beta^2(P+R)}$ Spiral $\left[\frac{1+\beta^2}{\beta^2}\right]\left[\frac{P \times R}{P+R}\right]$

Date.....

when $\beta = 1 \rightarrow$ F_1 score

$$= 2\left(\frac{P \times R}{P+R}\right) \rightarrow \boxed{\frac{2PR}{P+R}}.$$

* Cohen's Kappa Stats → can handle Both multiclass and imbalanced class problems.

$$\boxed{K} \rightarrow (p_o - p_e)/(1 - p_e)$$

$p_o \rightarrow$ Observed agreement
$p_e \rightarrow$ Expected agreement.

$K < 0 \rightarrow$ No agg
$0 < K < .2 \rightarrow$ Slight
$.21 < K < .40 \rightarrow$ fair
$.41 < K < .60 \rightarrow$ moderate
$.61 < K < .80 \rightarrow$ substantial
$.81 < K < .99 \rightarrow$ near-perfect
$K = 1 \rightarrow$ perfect

$$\boxed{p_o} \rightarrow [(\text{Both said Yes}) + (\text{Both said No})] / \text{Total Ratings}.$$

$p_e \rightarrow P(\text{Yes}) \rightarrow [\text{Rater 1 (Yes)}/\text{Total}] \times [\text{Rater 2 (Yes)}/\text{Total}]$

$P(\text{No}) \rightarrow [\text{Rater 1 (No)}/\text{Total}] \times [\text{Rater 2 (No)}/\text{Total}]$

$$\boxed{p_e \rightarrow P(\text{Yes}) + P(\text{No}).}$$

Spiral

Date.....

- **Eager Learners** → given set of Train Data → create a model Before recieving test/new Data.

- **Lazy learners** → simply stores the train data → waits until it is given a Test Data → No model creation

  → less time in Training
  But more time in predicting / Testing

  → No training stage → all the work is done during test stage.

— X — X — X — X — X —

K-MEANS

$\Sigma(x_i - c)^2$

- Clustering → Dividing dataset into groups
  → Here we do not have a Target value - we look at Data and try to group similar observation and form diff grps / clusters.

Property 1 → All the DP in a cluster should be similar to each other.

Property 2 → DP from different clusters should be as different as possible.

Intra-cluster Dist → Distance of a point from centroid in a same cluster

Date.....

Inter-Cluster Dist → Distance B/w centroids of 2 Diff Clusters.

Evaluation Metrics

① Inertia → evaluates 1st property of clusters.

① → Sum of Intra-Cluster Distance

② we want less value of Inertia → so that DP in a cluster are close/similar to each other.

→ minimize the Intra Cluster Distance → Inertia should always be low.

② Dunn Index → evaluates 2nd property of clusters.

→ Dunn Index = $\frac{\min(\text{Inter cluster Distance}) \rightarrow \max}{\max(\text{Intra cluster Distance}) \rightarrow \min}$

→ helps in seperating diff clusters so that DP from Diff clusters are very far from each other.

→ we want max value of Dunn Index (maximise Dunn Index)

- num should be max → (min of Inter-cluster Dist)
  ↳ so that Dist B/w even the closest clusters are more (min Dist B/w clusters should be max).
- Denom should be min → max of intra-clus-Dist
  ↳ so that max Dist B/w Points and Centroid should be min.

Spiral

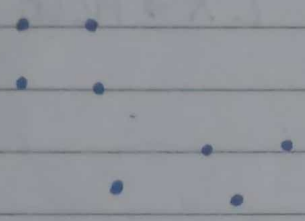
Date.....

☆ → RL → ∞

Working → min the Distance of the points within a cluster with their centroid.

→ Calc Distance → Assign a point to a cluster.

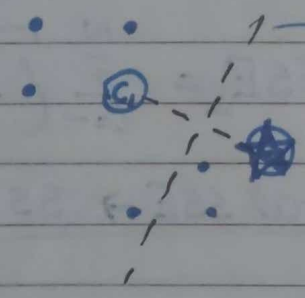
eg →



Step 1 → Choose the no. of clusters K

Step 2 → Select 'K' random points from the data as centroids.

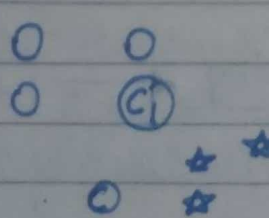
→



1— way of assigning points graphically

Step 3 → Assign all the points to the closest cluster centroid

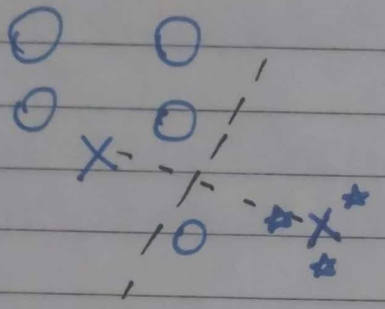
→ → → Calc Distance of DP from C_1 and C_2 and assign DP to that centroid where the Distance is minimum.



↳ Calcuate mean of each cluster = new centroid
↳ $(x_1+x_2+x_n)/n, (y_1+y_2+y_n)/n$ .... Date.....

Step 4 → Recompute the centroids of a newly form clusters.

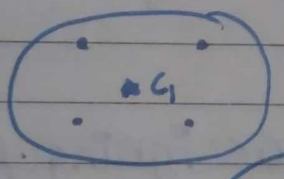
↳ Adjust centroids of the new clusters so that they become center of gravity of given cluster.



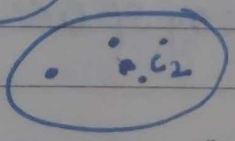
[X → new centroid]

Step 5 → Repeat **Step 3** and **Step 4**

Select model (K) with least SSE



$$SSE_1 \rightarrow \sum_{i=0}^{N} (x_i - C_1)^2$$



$$SSE_2 = \sum_{i=0}^{N} (x_i - C_2)^2$$

c intra

Total SSE → $SSE_1 + SSE_2 \ldots + SSE_K$

★ **When to Stop**

① Centroids of newly formed cluster do not change

② Points remain in the same clusters
→ No new points are being introduced in newly formed clusters.

Imp → Classification via clustering ← Imp

∴ Test data placed in Cluster its class = majority of that cluster
↳ can also give Probab (80% → A | 20% → B).

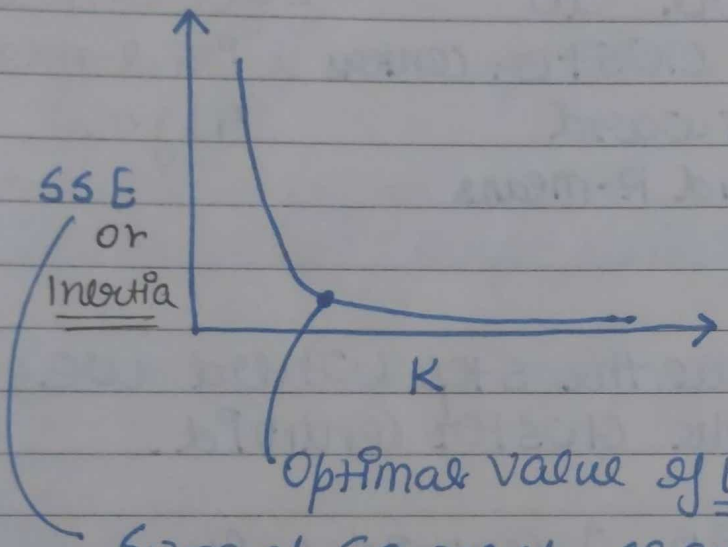
Spiral class

Date.....

How to find Best value of K?

① Draw a elbow curve

② Choose that 'K' where the graph starts to converge into constant or reach to 0.



[like an elbow → choose elbow part of the graph].

Sum of Squared errors.

SSE = Inertia

Inertia bhi laga sakte hai

from SKlearn.cluster import KMeans

Kmeans_obj = KMeans(n_clusters = 2, init = 'K-means++')

```
kmeans_obj.fit(data_scaled)
pred = Kmeans.predict(data_scaled)
```

Kmeans_obj.inertia_

elbow-curve (SSE = [])

```
for cluster in range(1,20):
    k = KMeans(n_clusters = cluster, init = 'k-means++')
    k.fit(data_scaled)
    SSE.append(k.inertia_)
```

Date.....................

```
Df = pd.DataFrame({'K': range(1,20), 'SSE': SSE}).
plt.plot(Df['K'], Df['SSE'], marker='o').
```

——— X ——— X ——— X ———

① K-Means ++ / K-means ++ → initializer of centroids Before Proceeding with K-means Algo.

→ Specifies a procedure to initialize the cluster centers before moving forward with the standard K-means clustering. algo.

① we Optimize / improve the step where we randomly pick the cluster centroid.

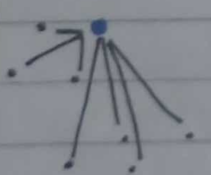
STEP 1 → Randomly pick 1 centroid only.

STEP 2 → Compute the Distance $D(x)$ of each DP (x) from the cluster center that was chosen.

STEP-3 → Choose new cluster center from the DP with the probab of $x \propto D(x)^2$.

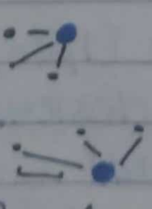
STEP-4 → Repeat step 2 and 3 until K-cluster centers are choosen.

[when 2, 5 or 3 calc DP dist to Both K. Choose the min one and then square it]

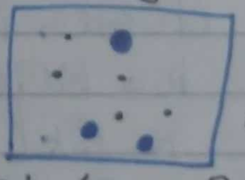


→ new centeroid → $(D(x))^2$ is farthest

← IMP



2nd



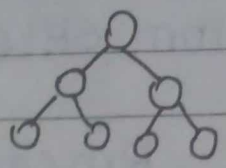
2nd → Distance of each point from closest centroid and the point with largest $(D(x))^2$ → centeroid Spiral

overfitting ← ↱ low Bias and High Variance

DT

Date.....

Binary split — f_1 f_2 f_3, ... output



└ Binary classification

1) Entropy → Measure of the purity of split
→ Helps in deciding which feature should be selected for splitting.

value ranges from 0 ⟶ 1 | Low value - Better |

- Pure split → ✕ Yes / ○ No OR ○ Yes / ✕ No
  └ Entropy = 0 (usually leaf Node).
- Impure split → ✕ Yes / ✕ No → Entropy = 1

Imp
- It only counts for single node But in splitting we need to calculate for other nodes also as they are part of tree so we use Information Gain which includes entropy.

$E \rightarrow H(s) = -P_{(+)} \log_2(P_+) - P_{(-)} \log_2(P_{(-)})$.

→ $P_{(+)}$ = % of +ve class (Probab)

→ $P_{(-)}$ = Probab of -ve class.

[3-Yes | 2-No]

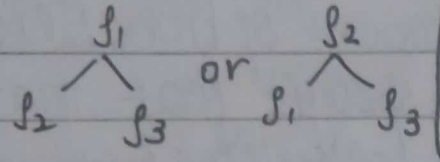
$= -\frac{3}{5} \log_2\left(\frac{3}{5}\right) - \frac{2}{5} \log_2\left(\frac{2}{5}\right)$

$= 0.97$ bits → Not good.

Spiral

Date.....

- **Information Gain** → avg of all Entropies of a tree to see which split is better. [order].



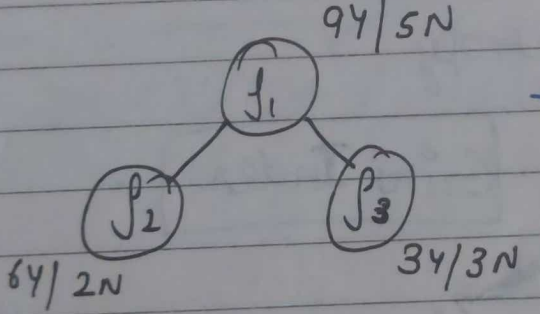
on which feature? using purity of split of each node and using their average ↑ Imp

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

(S ← Subset of Root; A ← Root)

→ $E_S - \sum_{v \in VAL} \frac{|S_v|}{|S|} E_{S_v}$ (subset after spliting.)

eg →



→ $S \rightarrow 14$
$S_{f_2} \rightarrow S_{v_1} \rightarrow 8$
$S_{f_3} \rightarrow S_{v_2} \rightarrow 6$

→ $E_S - \sum_{v \in VAL} \frac{|S_v|}{|S|} E_{S_v}$ | $E_{f_1} = 0.94$ | $E_{f_2} = 0.81$ | $E_{f_3} = 1$ |

→ $0.94 - \frac{8}{14} 0.81 - \frac{6}{14} \cdot 1 = \underline{0.049}$

★ → **Higher the value** – better it is. – That way is better for spliting.

Spiral

Date.....

* Gini Impurity (similar to Entropy But Better).

→ $1 - \sum_{i=1}^{n} (P)^2$

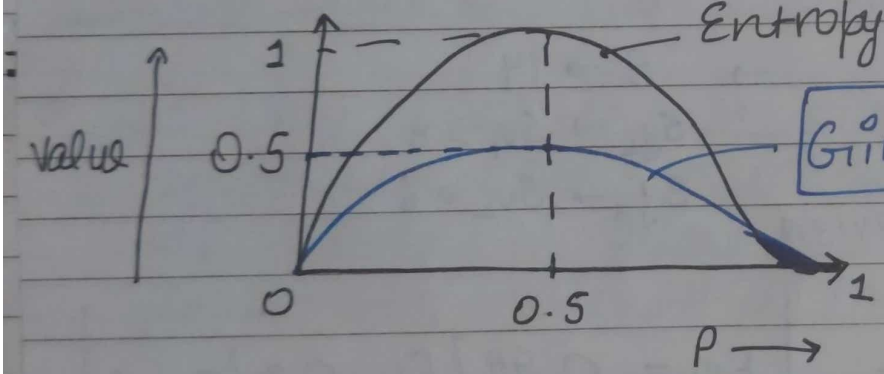
$= 1 - [P(+)^2 + (P(-))^2]$.

[Amount of Probability of a specific feat that is classified incorrectly when selected at Random]

→ Computationally Efficient

if I have 3+ and 3- → E = 1

But Gini Index would be 0.5



[P = P+ and P-]

→ Shorter period of Time for execution.

(log takes time).

Imp

- Then we can use Information gain by using entropy or Gini Impurity.

- Gini - CART Algo
- IG → ID3 algo.

DT for Numerical feat -

Date.....

→ R→ DT will ↴

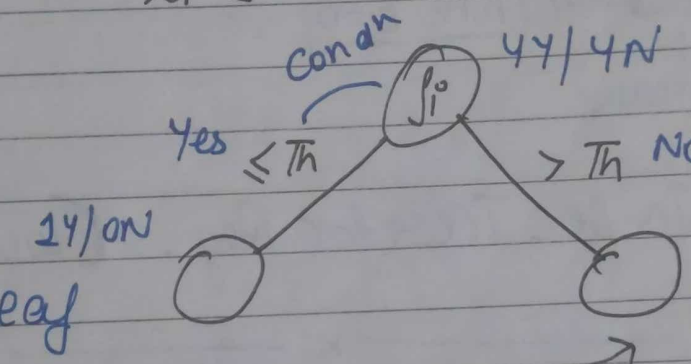
→ ① Sort all the values of that feat

② Take Threshdd value. (Th).

2.1 select a value (every value go-through)

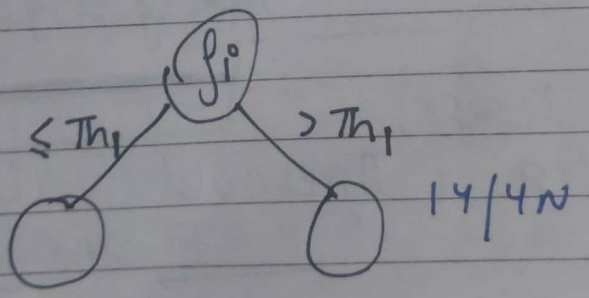
└ $x_i \leq Th$ → makes a Branch.

C	Y
V_1	- Th
V_2	- Th_1
V_3	- Th_2
V_4	- Th_3



└ 1st Split

• for 2nd Split (2nd way of split).



[Check / Split for every Threshold happens]

• Which is Better (with Th or with Th_1)?

└ Entropy and If

└ Better If will be taken.

[★ Takes Time]

• If of every feature is calculated wrt to the output variable in normal data.

Spiral

Ensemble learning

Date.....

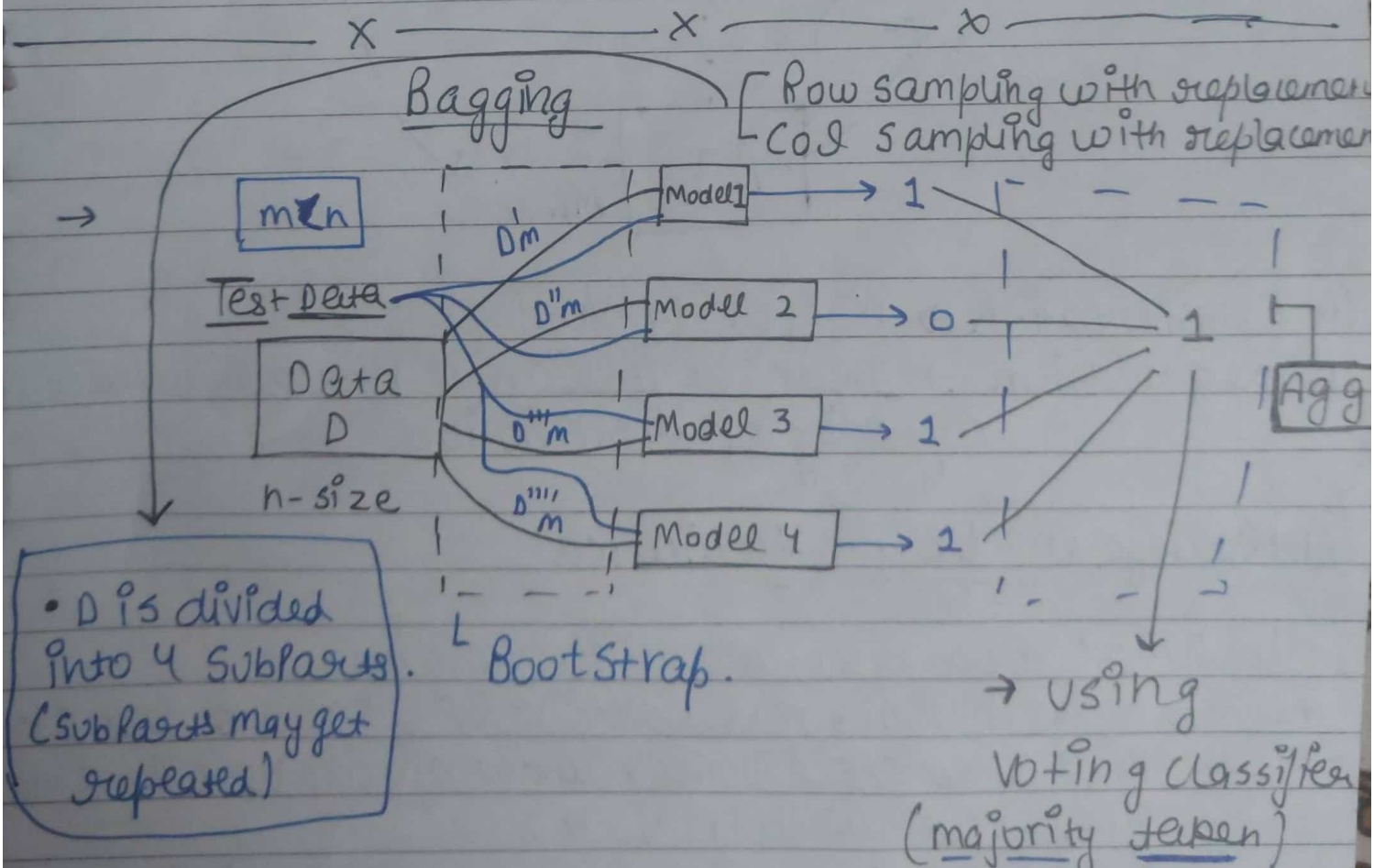
→ Combining multiple Models.

① Bagging (Bootstrap - Divide)
(Aggeration - Combine)
(Algorithm).

→ Random Forest

② Boosting

- AdaBoost
- Gradient Boosting] Aavance.
- Xg Boost



Spiral

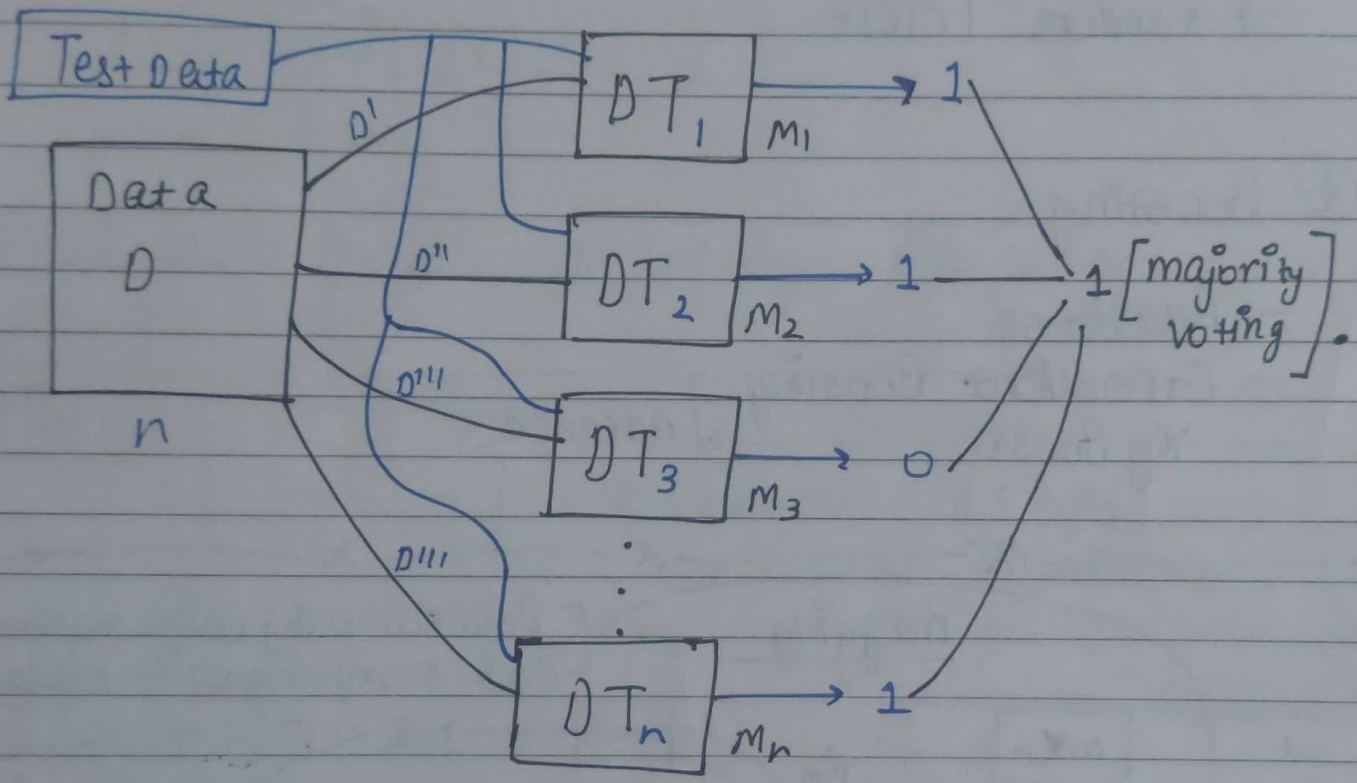
Random Forest

Date.....

→ Bagging Tech

- → RS → Row sampling with replacement
- → CS → Col sampling with replacement

RS + CS → D'



In Classification → majority voting
In Regression → mean or median of 'n' output values depending on distribution.

Advantage ① Low Variance

- In DT → low Bias and High variance
  + when multiple DTs are used in RF with RS and CS
    ↳ get TRAINED SPECEFICALLY and majority vote converts High Var to low Var.

② Change of m records (m<n) will not affect the model that much.