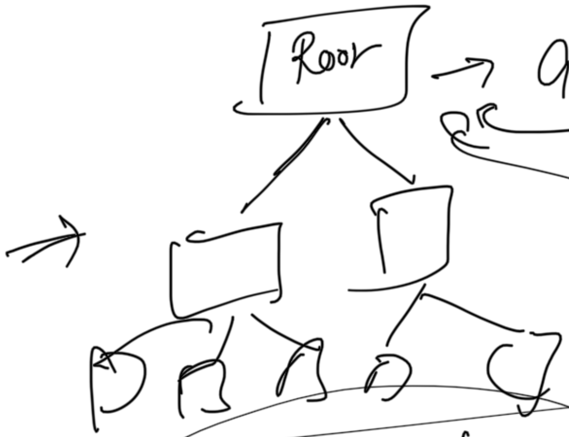
Random forest

Decision Trees

↳ hierarchical Tree Structure

↳ Classify ↳ Regn.



→ GINI INDEX / ENTROPY.

I.G ↑

Ensemble Learning

Combining different / multiple a weak learners to form a very strong model.

↓

- Give us — One Model
- combination of multiple models

(1) Bagging (2) Boosting

(1) Bagging Technique → Random Forest algorithm

Internally → Strong ensemble model with ensemble with samples of data

(1) Random forest is not prone to overfitting
(2) Less to very little / no feature Engineering
(3) No assumption of linearity / Distribution
(4) Multi-core processing / n-jobs
(5) Random forest can be used to fill missing data

6) Feature Importance → Random forest to say which columns are important to make the prediction

→ Bagging → Bootstrapping + Aggregation

1) Trees are inflexible / very rigid when we have new data

Random forest → aims to combine SIMPLICITY OF TREES + FLEXIBILITY → Better accuracy

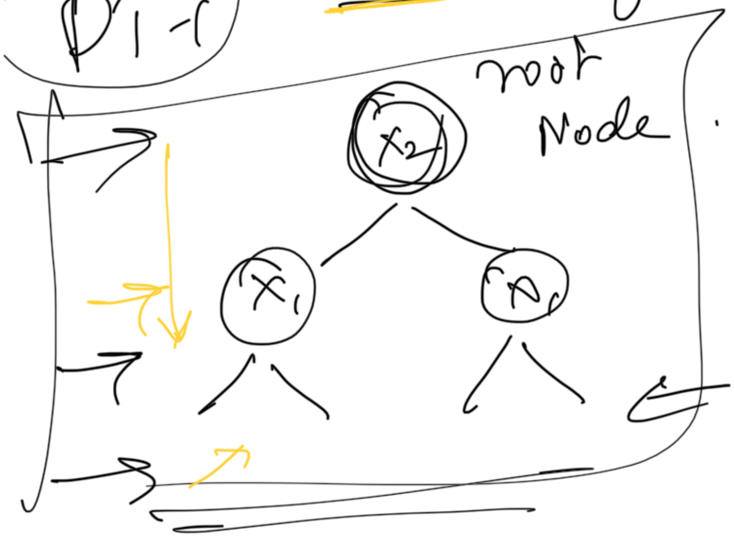
1) Creating a Bootstrapped dataset

1) Rule of Bootstrapping ↳ create a dataset same size as the original dataset

Decision Tree from

② Create a decision tree using this bootstrapped dataset; randomly select "subset" of columns at "each" step.

DT-1



We build a tree

① Use the bootstrapped dataset.

② Considering random subset of columns at "each" step.

Go back to step 1 → Make new bootstrapped data.

↓

build tree → new rand subset of cols.

What makes the random forest effective??

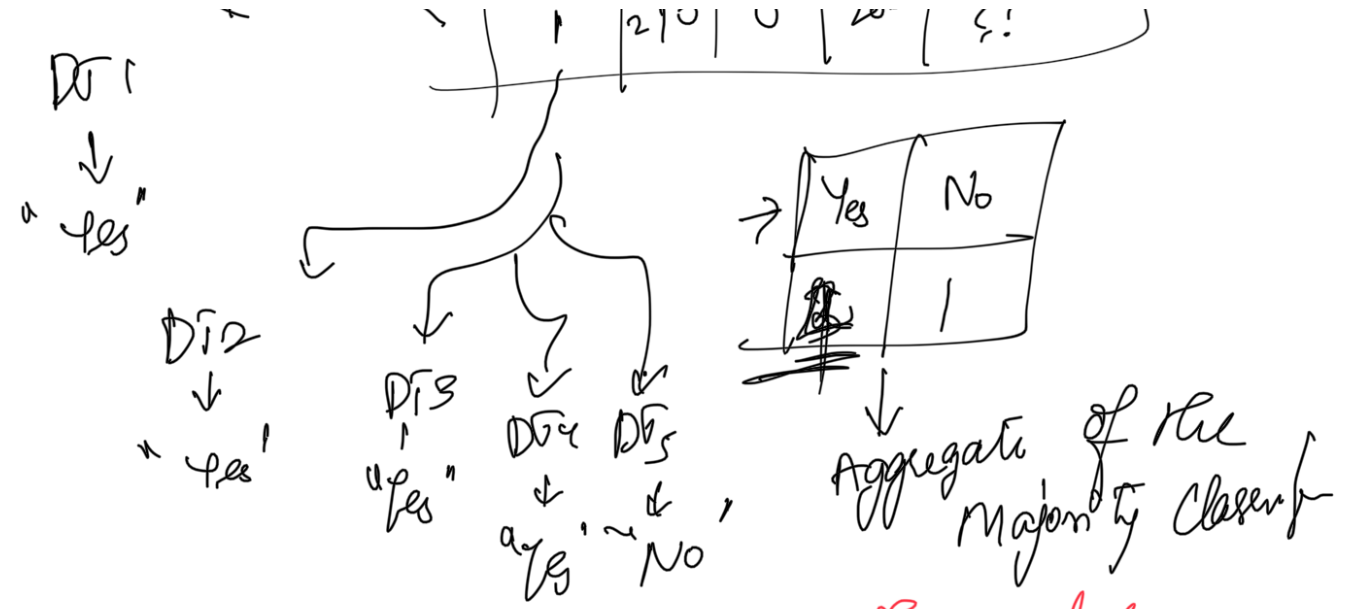
① using bootstrapped sample.
② Consider only a subset of cols } ⇒ Wide variety of flavours

~~more~~ effective in terms of capturing variation than individual decision trees.

Building done!!

Prediction | Inference

X_1	X_2	X_3	X_4	Y



Bootstrapping data to create the model
Aggregation to make a decision

Defined → "BAGGING!!"

Evaluation

Rows that were never a part of Bootstrap are called ⇒ "Out of the Bag" Dataset

OOB dataset

We perform evaluation on this

↳ OOB Score

Performance!!

OOB Error

Proportion of Out-of-Bag rows wrongly predicted.

... with "Hyperparameters"

You can play with

① Model Parameter

① The parameters that are learned as a part of model training.

↳ eg) Slope / Intercept ??

Hyper parameter

① These are parameter that are specified by the user / defined before hand /

↳ eg) 'k' value in KNN / depth of decision tree

Do Hyper parameters Tuning → find the optimal configuration for our model.

Break till 9:40 am IST

10 DTs

	X_1	X_2	X_3	X_4	Y
0	0	200	No	200	No
1	1	50	Yes	125	Yes
2	0	10	No	160	No
3	0	100	NaN	200	Yes

No

? No ??

DT 2

Step 1 :- Building the nodes for

Step 2 :- Run all the data into the tree

Proximity Matrix

	0	1	2	3
0			1+1+1+1	
1				1+1+1
2	1+1+1+1			
3		1	1+1+1	

Normalize / No. of trees

$\Longrightarrow$ No of trees

Based on weighted proximity

0 $\Leftrightarrow$ 1

higher weights e.e. frequen. will take precedence.

~~Co-oc~~

correlation matrix

	0	1	2	3
0		2	1	1
1	2		1	1
1				8
2	1	1		
3	1	1	8	

Each row

DT_1

Each Row

DT_2

Each

DT_3

DT_4 ~~DT_5~~

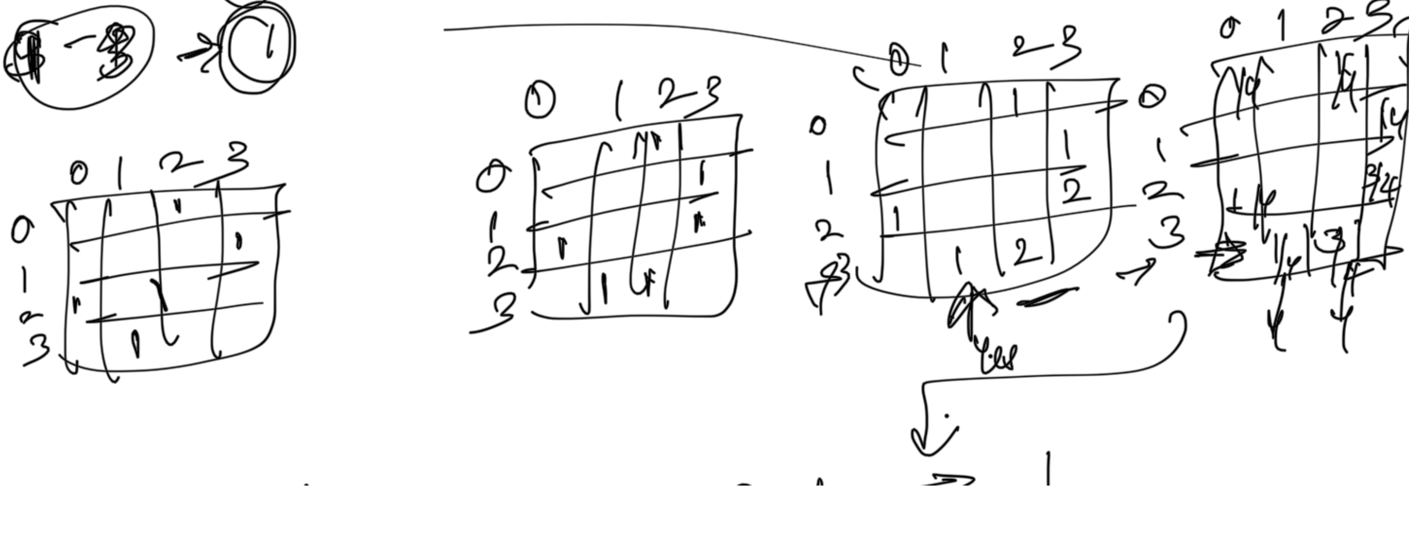
0 - 2 $\Rightarrow$ 1

~~1 - 3~~ $\Rightarrow$ 1

2-3 Sim

2-3 Sim

2-3



0 ←

KNN

Sampling ← Normalized Proximity matrix

→ Weighted Frequency → Fill missing val

2 No

1 Yes

$\cancel{2/3}$ X

→ 1/3

highest weight

→ Feature Reporting