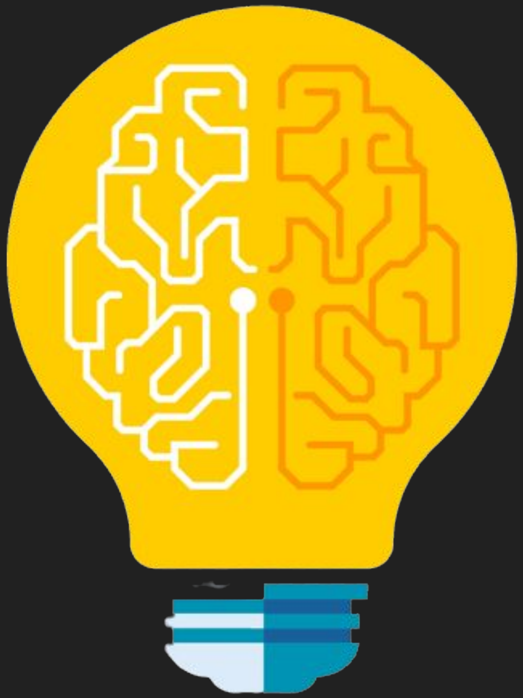
Bagging + RF



Solving the attrition problem for Jio

Recall in the previous lecture,

- We were predicting the attrition rate for Jio’s HR department using a Decision Tree with max_depth=4.
- Earlier, we got

Train accuracy : 84%

Test accuracy : 78%

How can we improve the previously obtained results?

We can improve the results by combining multiple models for the same task.

This is known as Ensembles.

What are Ensemble models?

Ensemble = { (models) many base learners + combine smartly }

many base learners ⇓ as different as possible

Types of Ensemble

- Bagging (RF)
- Boosting (GBDT)
- Stacking
- Cascading

What is Bagging?

Bagging simply means **Bootstrapped Aggregating**.

Suppose, you want to buy an iphone. You ask these people for their opinion:

Friend	Cousin	Mother
Yes	Yes	No

Yes

Majority Vote

Bagging refers to:

- Training different models for the same task.
- Then, smartly combining their predictions.

EXAMPLE

Task : Predicting credit score

(Regression)

Models	M_1	M_2	M_3
Prediction	82	90	78

Average

84

Can we make an ensemble for DT's ?

Yes, **Random Forest (RF)** is an ensemble method based on Bagging and Decision Trees.

C_1 C_2 C_3 C_4 C_5 C_6

Each tree is trained on a random subset of :

- Rows (d')
- Columns (m)

This is known as **Row & Column Sampling.**

Collection of DTs

$$RF = \underset{\text{Base Learner}}{\underset{\downarrow}{DT}} + \underset{d' \ll d}{\underset{\downarrow}{R.S.}} + \underset{m \ll n}{\underset{\downarrow}{C.S.}} + \text{Aggregation}$$

**NOTE: Column sampling is done without replacement.

How to use multiple DTs together?

- Assume we have dataset (D) with (n) records and (d) features.
- We sample:

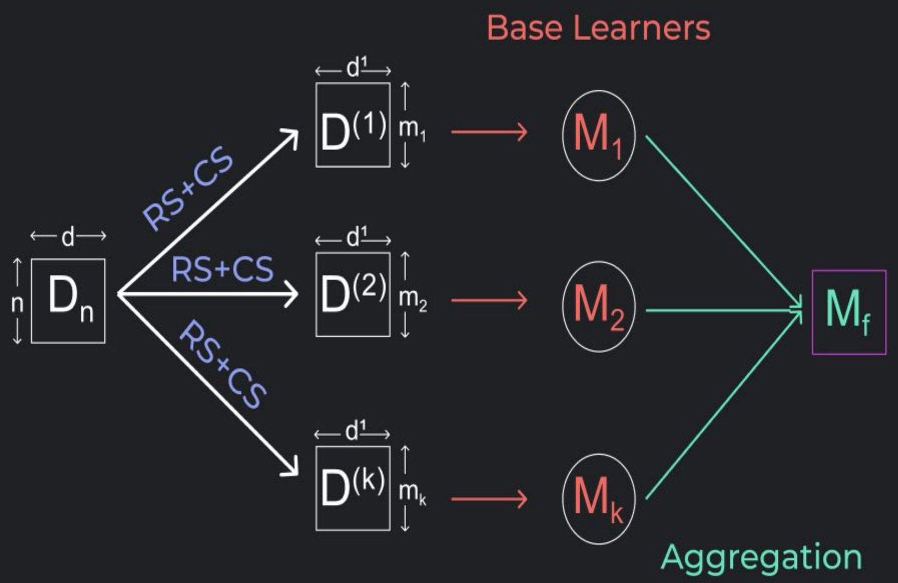
  (m_1) data points with replacement

  (d’) columns to get dataset (D_1)

- Repeat the same (k) times and we get (D_k)
- Train (k) different base learners (M_1 , M_2 , M_3 ……M_k) on these datasets.
- At the end perform Aggregation

Classification → Majority Vote

Regression → Mean / Median



But why do we need randomness?

Our goal is that base learners should be as different as possible.

Recall that iphone example:

- you ask 3 people who are all iphone fans.

If all of them have similar opinions,

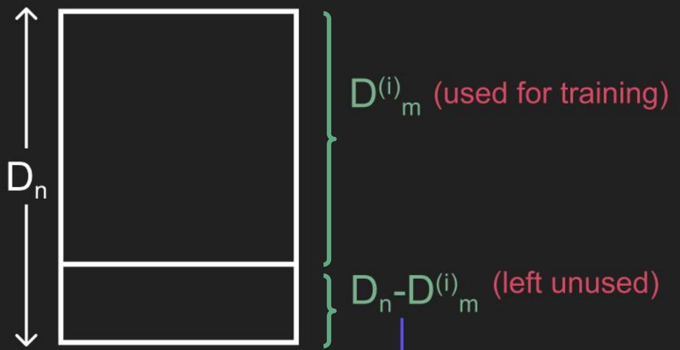
- the 2 extra people don't add much value.

How can we validate a RF?

We are training :

- k different models $(M_1, M_2 \ldots\ldots\ldots\ldots M_k)$
- K different datasets $(D^1_m, D^2_m \ldots\ldots\ldots\ldots D^3_m)$

For model M^i



$D^{(i)}_{n-m} = D_n - D^{(i)}_m$ = OOB Points (Out of Bag)

$c_1 \Leftrightarrow c_{10}$

100 rows

60 rows $\leftarrow$ DT_1 $(c_1 \Leftrightarrow c_4)$

40 rows $\leftarrow$ $(c_1 \Leftrightarrow c_7)$

Out of Bag samples

These remaining $(n-m)$ rows are used for validation of M^i

EXAMPLE

Original Set

Patient A
Patient B
Patient C
Patient D

Bag 1

Bootstrap Sample	Out-of-Bag-Set
Patient A	Patient B
Patient A	Patient D
Patient C	
Patient C	

Original Set

Patient A
Patient B
Patient C
Patient D

Bag 2

Bootstrap Sample	Out-of-Bag-Set
Patient A	
Patient B	
Patient C	
Patient D	

Original Set

Patient A
Patient B
Patient C
Patient D

Bag 3

Bootstrap Sample	Out-of-Bag-Set
Patient A	Patient B
Patient D	Patient C
Patient D	
Patient D	

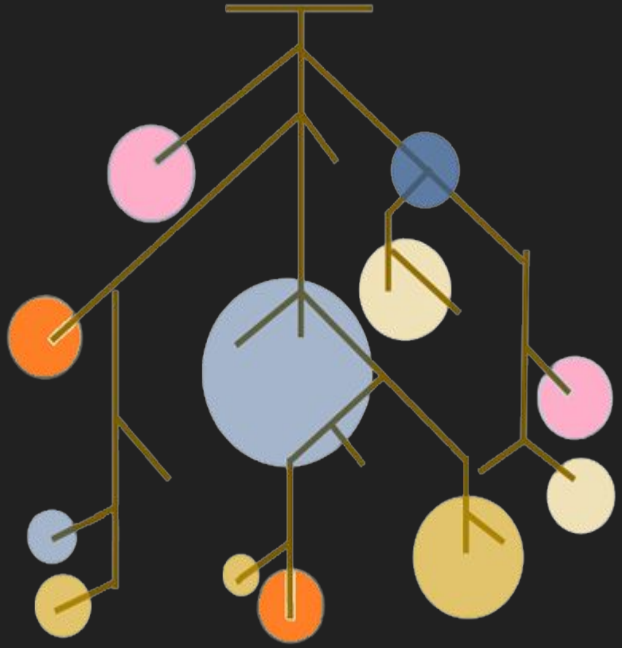
How to measure overall performance of RF?

In Random Forest,

Base Learners are validated using OOB data points.

But as a whole, RF still requires -

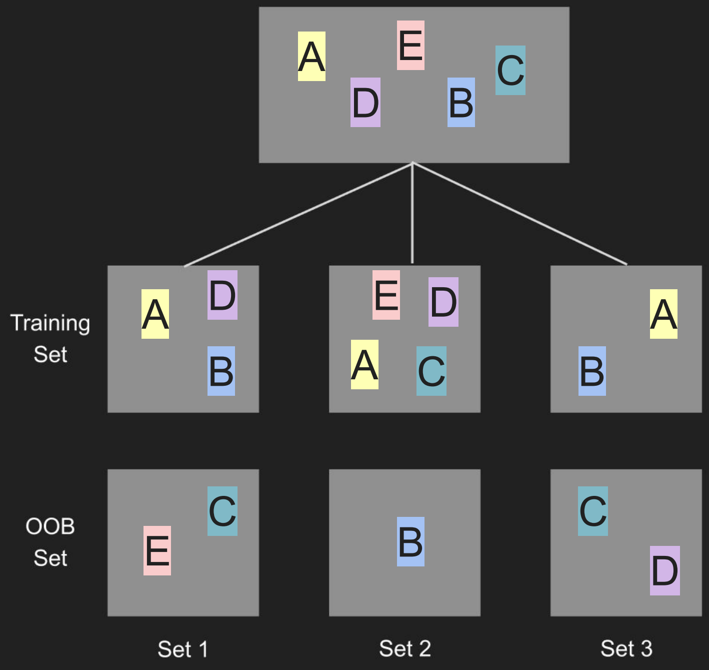
- A test data to tune the hyperparameters
- A cross validation data.



OOB Score

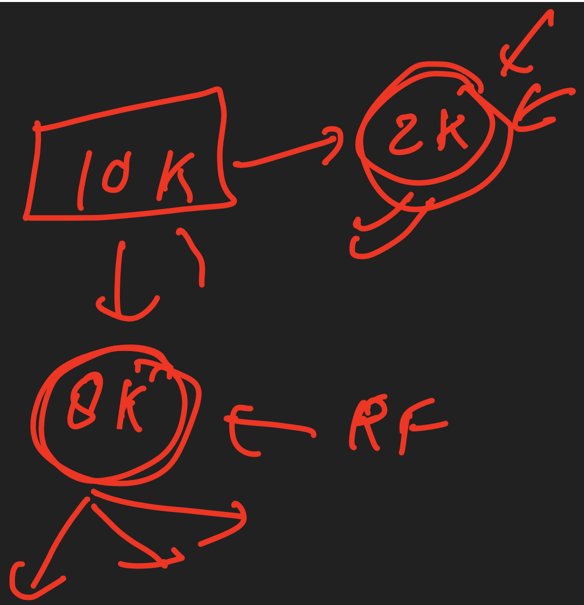
How to calculate the OOB score?

Example:-



- Point C will be OOB point for M_1 & M_2
- After training RF, pass point C through M_1 & M_2
- Take majority vote / mean of pred
- Compare pred with actual value of label for point C.
- Do this for all points to get OOB Score.

When should we use OOB Score?

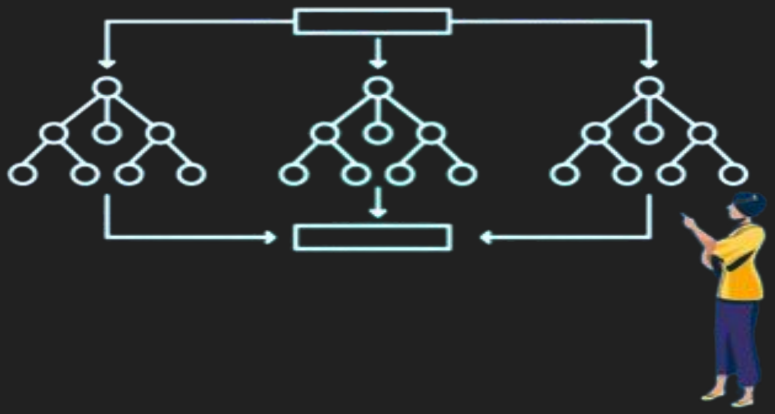


When our dataset is not large enough,

- We can’t afford to keep a subset for validation.

So in that case, we estimate overall model performance

- from individual model performances on the OOB points.

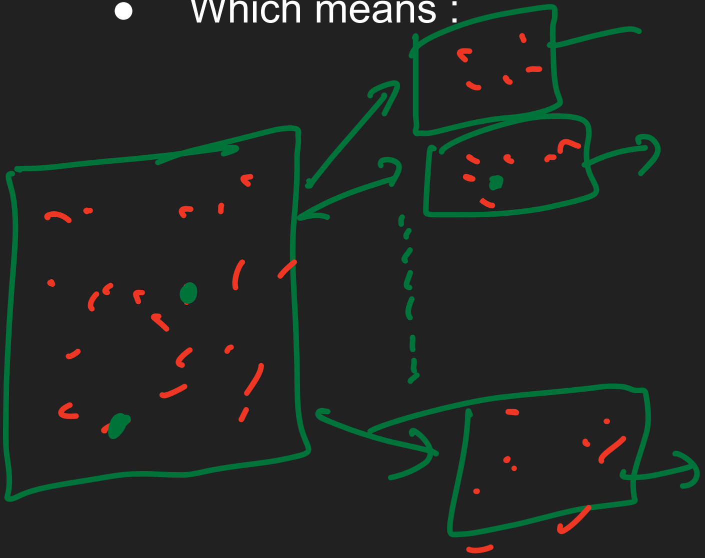


Bias - Variance Tradeoff

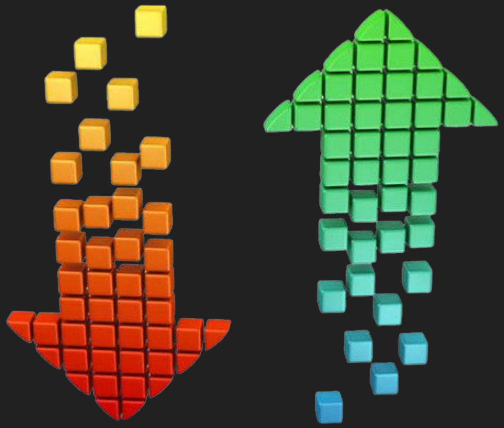
High var → overfit

The base learners in RF are Deep Decision Trees.

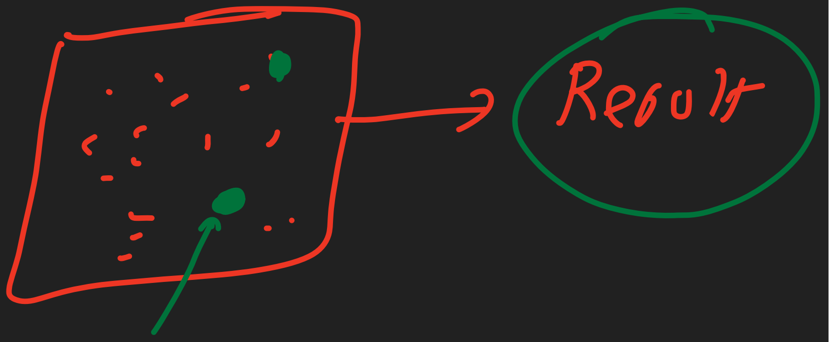
- So they slightly overfit on the sub sample of data
- Which means :



High Variance



Low Bias



DT → 1000

How to reduce variance?

$\boxed{\text{30 Trees}}$ ←

By using **Aggregation**.

- Suppose, we have multiple base learners with high variance.
- Predictions vary by ±20% of actual value.
- When we take the average of pred values +ve errors cancel out -ve errors.

Thus, we are left with smaller residual errors.

In Statistical ML,

Error = Bias^2 + Variance + Irreducible error

Base Learners

Bagging

Low bias, Low variance

Due to aggregation, variance decreases without trading of bias.

Thus, overall error of RF reduces.

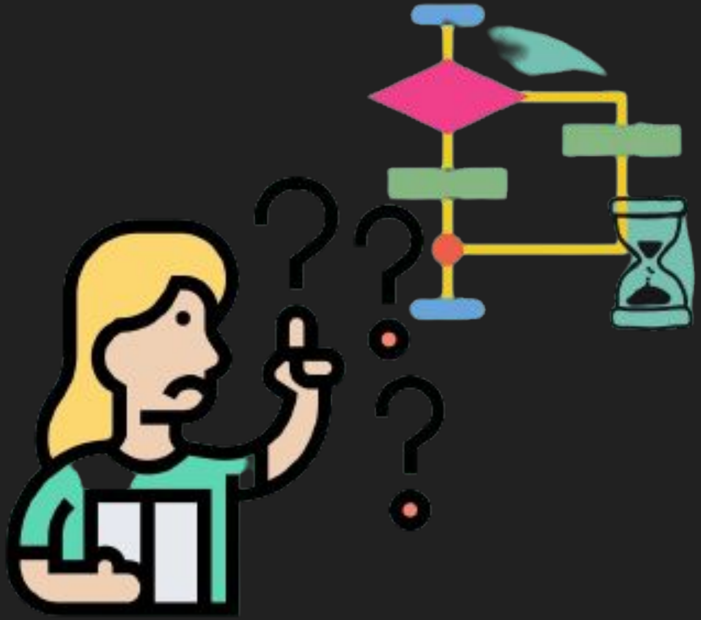
Training a RF Model

- The base learners in RF can be trivially parallelised.
- As each model is trained independently,
    - we can use distributed computing.

As a result, the training process become faster.

Time Complexity - O(k * max_depth)

Space Complexity - O(no. of nodes * k)



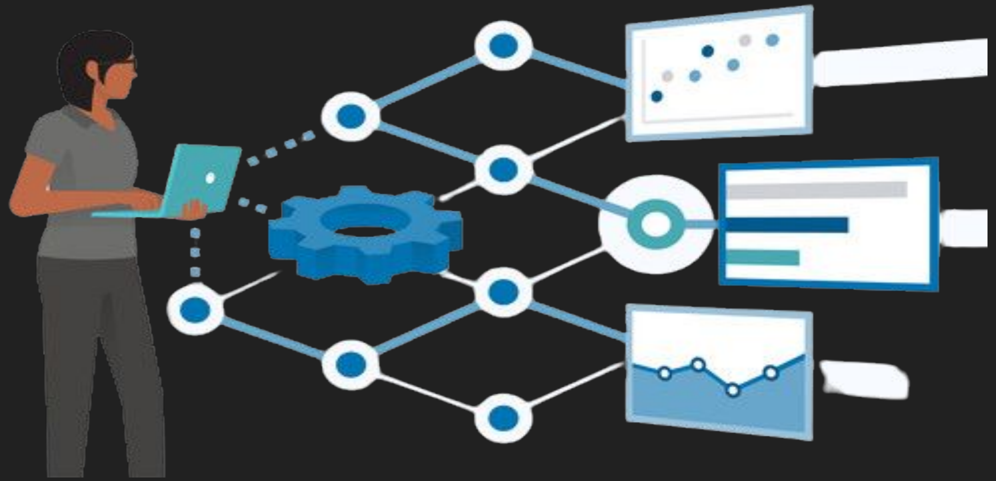
Optimizing our RF Model

The various hyperparameters of random forest are :

Break: 8: 17AM

1. **n_estimators** - Number of Trees (k)
   Default = 100

2. **max_samples** - Row sample size (m)
   Default = None, draw all the samples, otherwise m ∈ [0.0 , 1.0]



3. **max_features** - Number of columns samples (d')
   Can be {"sqrt" , "log2" , None}
   Default = "sqrt"

High vari

4. **max_depth** - Depth of base learners

$\alpha = [\,-,\ 0.1,\ -\ -\ -\,]$

2

5. **ccp_alpha** - Cost Complexity Pruning

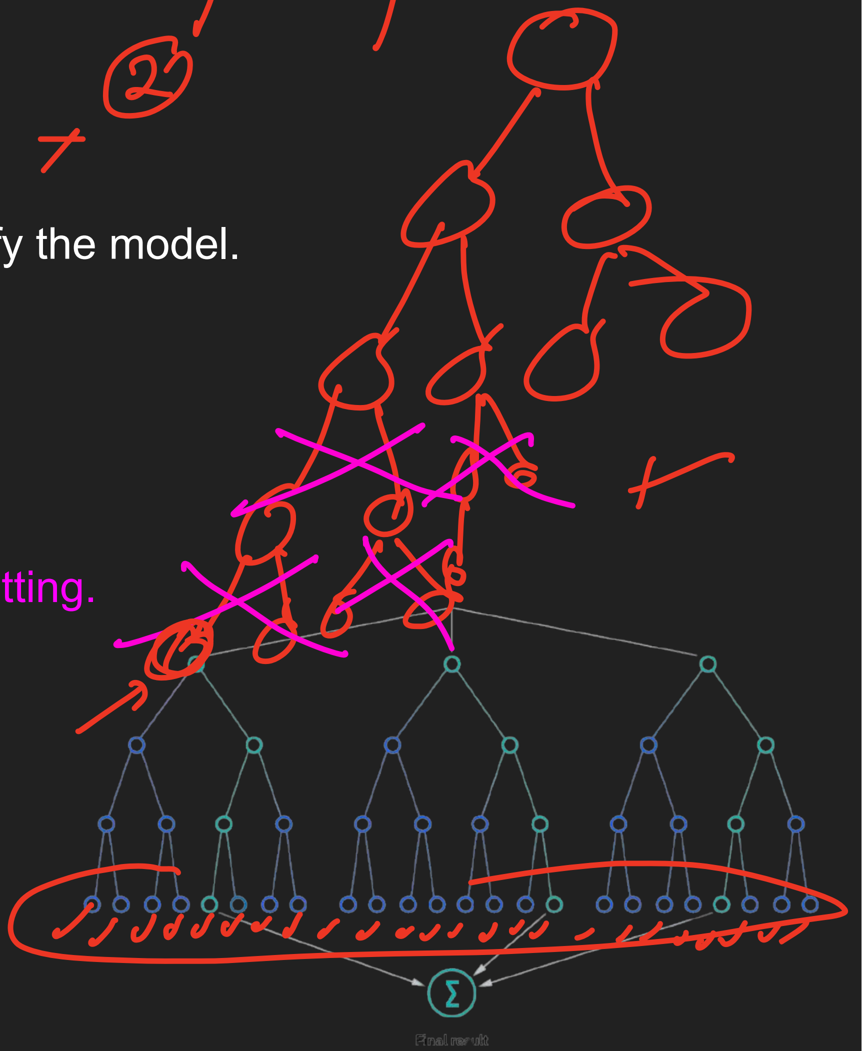
(α)

Gradually removing tree branches and leaves to simplify the model.

As α increases,

- more of the tree is pruned

Thus, creating a generalized model and prevents overfitting.



$+\alpha|w^2|$ $|w|$

0.2

$+\alpha$ leaf

100

20

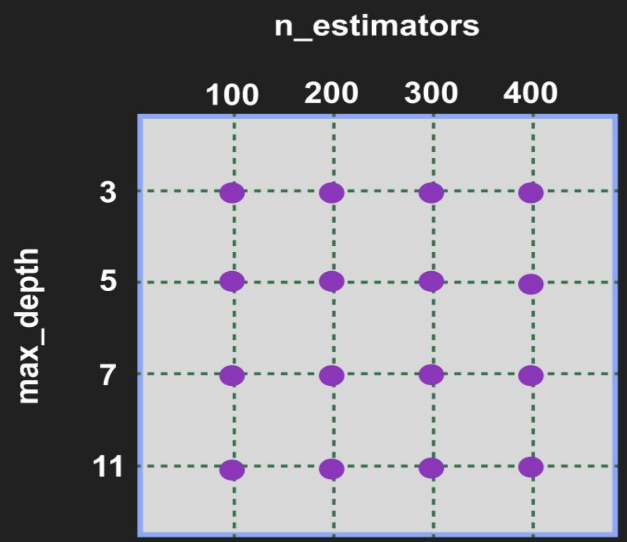
ccp-alpha

Fine tuning the hyperparameters

| leaf |

Grid Search

- Specify a range of hyperparameter values.
- Try every possible combination
- Choose the optimal set



Grid

Total Combinations = 4 * 4 = 16

"Brute Force Approach"

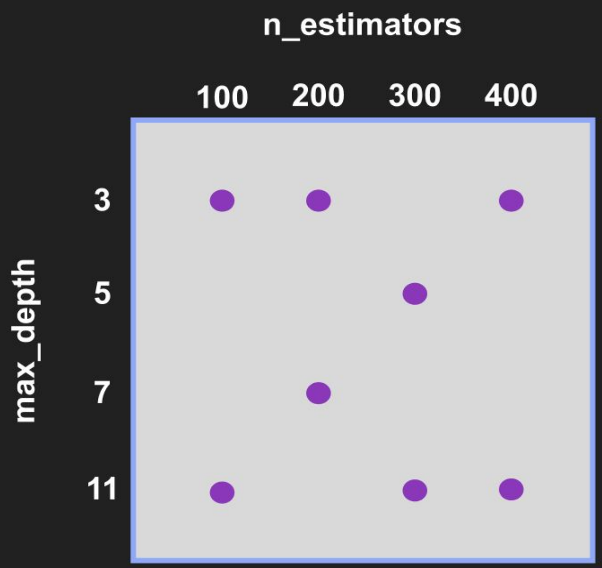
Disadvantages :

- Computationally expensive
- Time consuming

Fine tuning the hyperparameters (contd.)

Randomised Search

- Try random combinations of hyperparameters
- From a finite list of options or from a distribution



Use when dealing with :

1. Large hyperparameter space
2. Limited computational resources

Advantages :

- Reduced search space
- Allows faster exploration

How to compute feature importances?

100 columns
10% column sampling

- Compute importance of a feature in each DT.
- Take the average of these values.

$C_1 \quad C_2 \quad \ldots \quad C_{10}$

100 DT → 10 DT

fI

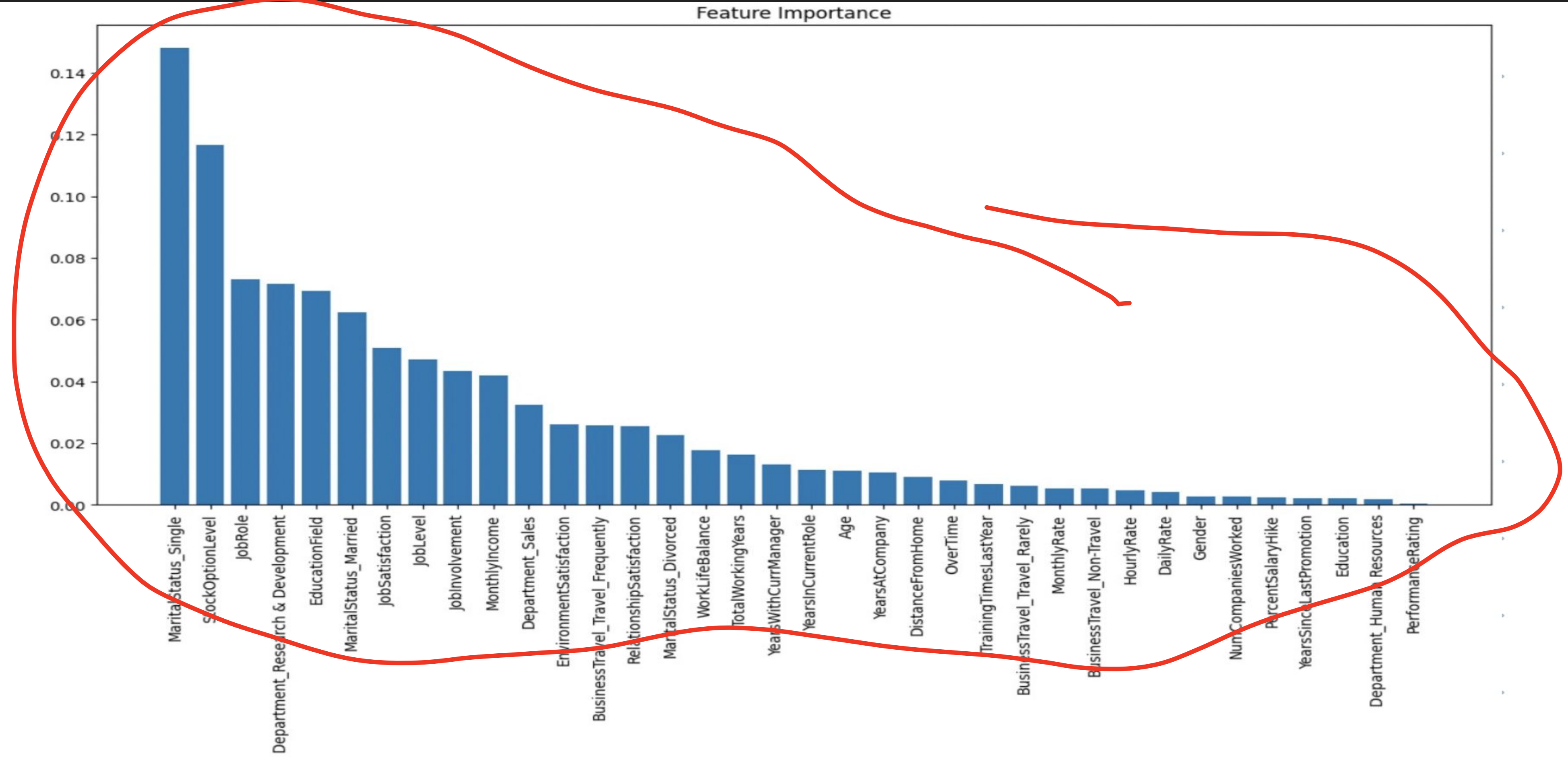
What if some base learners don't have those features?

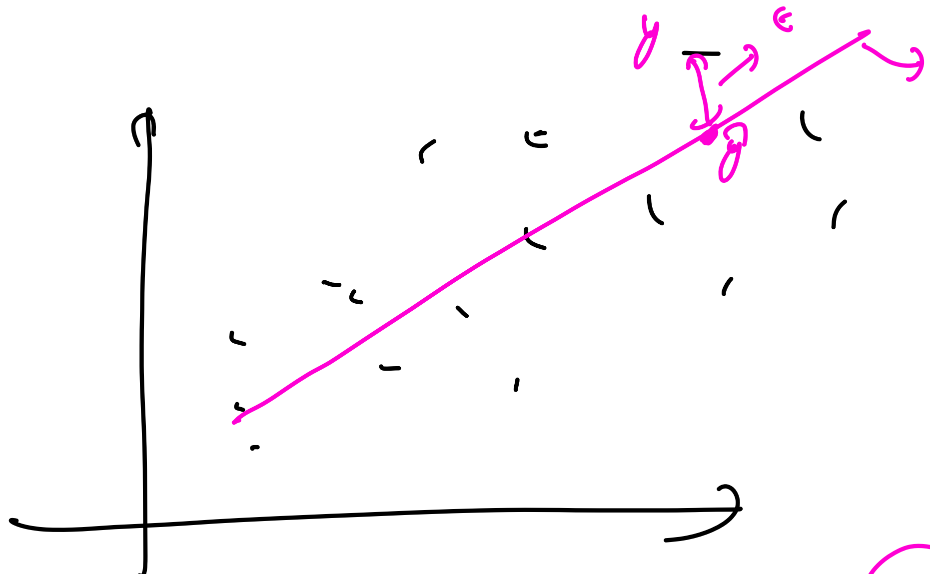
Remember column sampling?

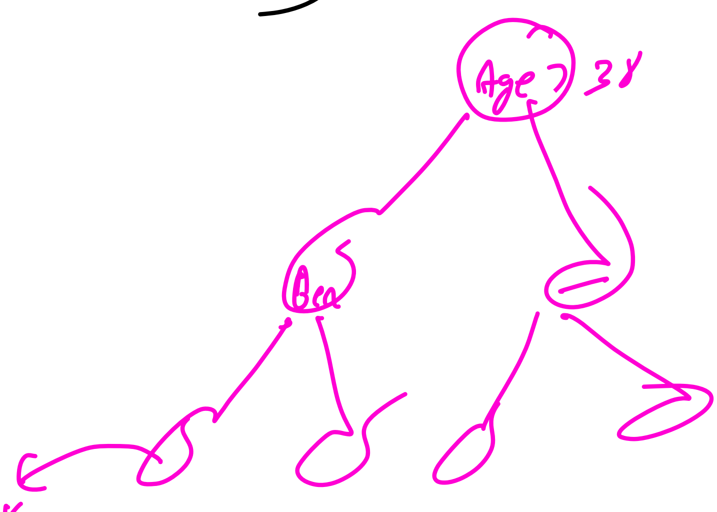
- In such cases, the importance of the missing feature for that base learner is 0.

The presence of multiple decision trees compensates for the absence of certain features in individual trees.

Feature Importance







— — — 20K

— — — 40K

— — — 30K