An Introduction to Machine Learning with Python Programming
11 Sep 2023 - 20 Oct 2023

Conducted by:

Random Forests

iHUB Divya Sampark, IIT Roorkee
and
Ritvij Bharat Private Limited (RBPL)

Presented by:
Shreyas Shukla

fig ref: pierian data

Random Forests can greatly increase the performance based on ideas from the Decision Tree.

Also known as **ensemble** learners, since they rely on an ensemble of models (multiple decision trees).

fig ref: pierian data

What is the motivation behind Random Forests

How are they better than Decision Trees?

fig ref: pierian data

Imagine this data set:

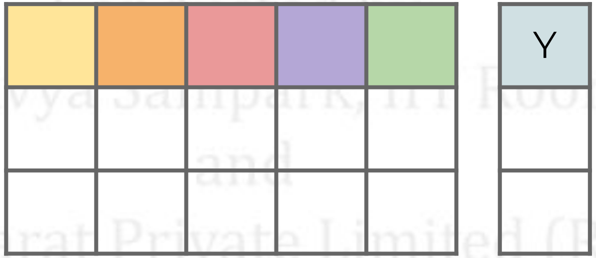


fig ref: pierian data

- Decision Tree are restricted by gini impurity.
- No guarantee of using all features
- Root node will always be the same

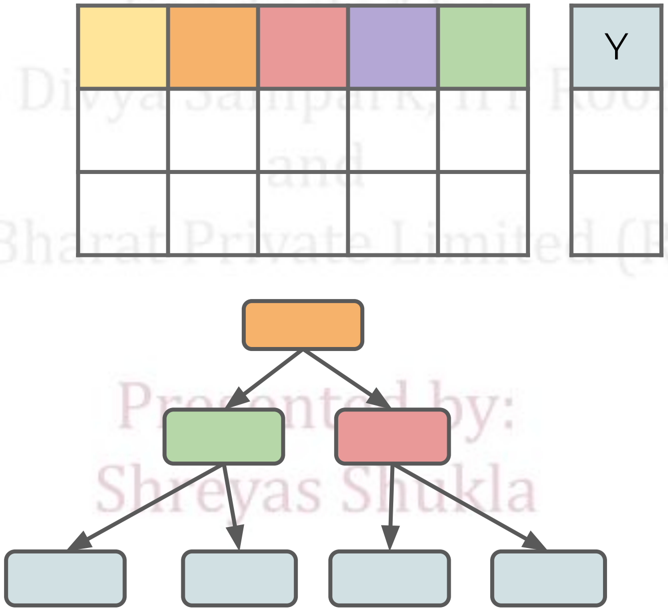


fig ref: pierian data

Root feature has huge influence over decision tree.

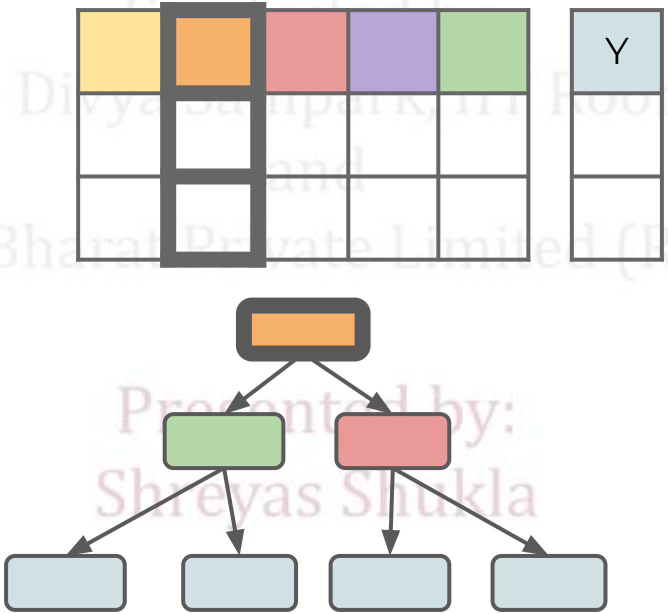


fig ref: pierian data

We could try adjusting rules, such as:

1. Splitting Criterion (Information Gain)
2. Minimum Gini Impurity Decrease
3. Setting Depth Limits
4. Limits on number of terminal leaf nodes

fig ref: pierian data

However even with all these added hyperparameter adjustments, the single decision tree is still limited:

- Single feature for root node.
- Splitting criteria can lead to some features not being used.
- Potential for overfitting to data.

fig ref: pierian data

In 1995, Tin Kam Ho presents *Random Decision Forests*

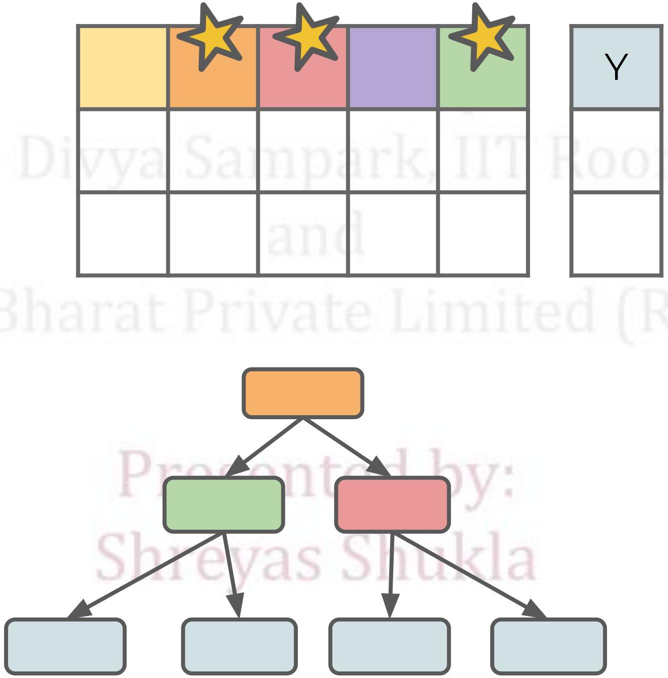


fig ref: pierian data

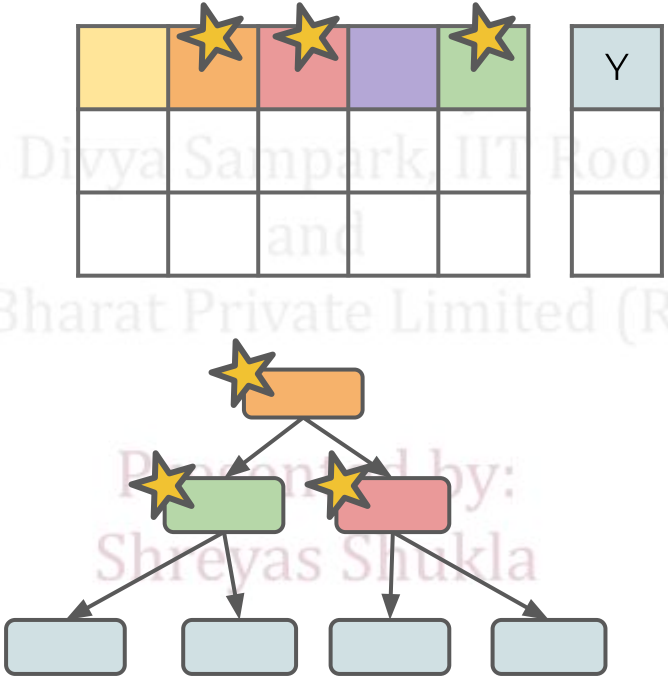


fig ref: pierian data

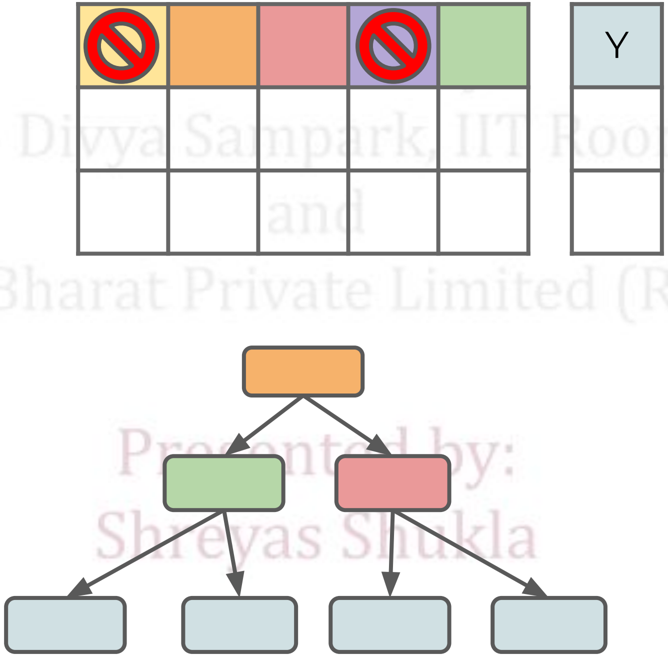
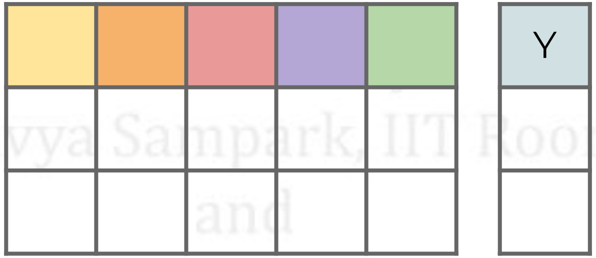


fig ref: pierian data



Create subsets of randomly picked features at each potential split

fig ref: pierian data

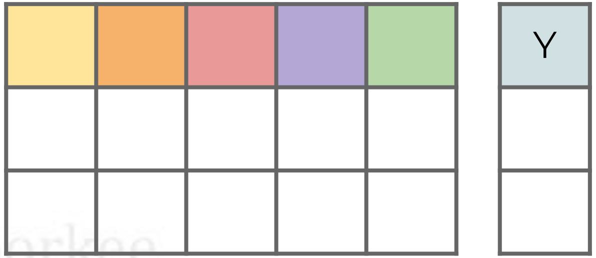


fig ref: pierian data

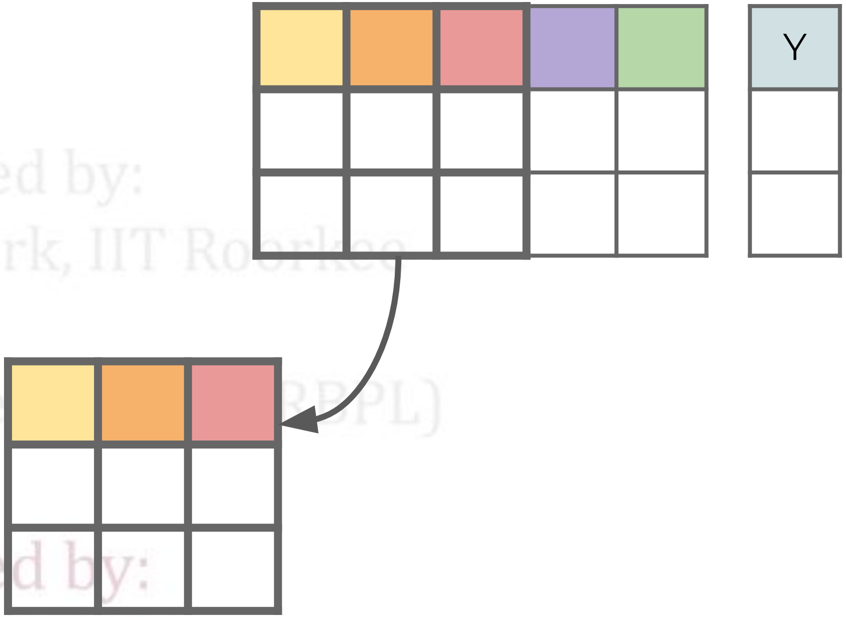
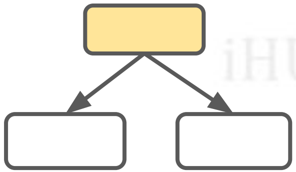


fig ref: pierian data



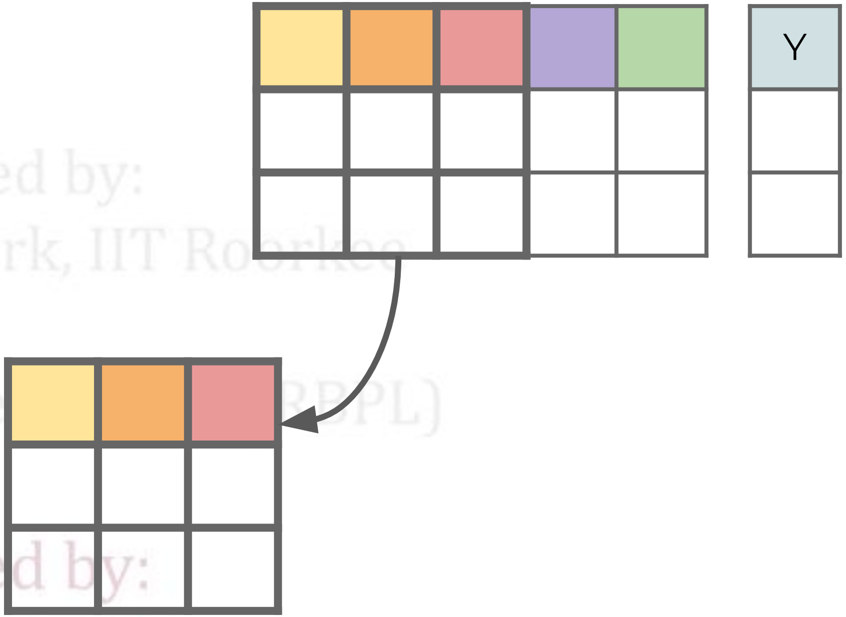
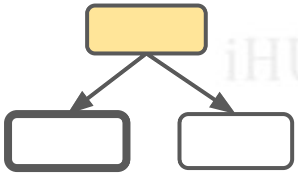


fig ref: pierian data



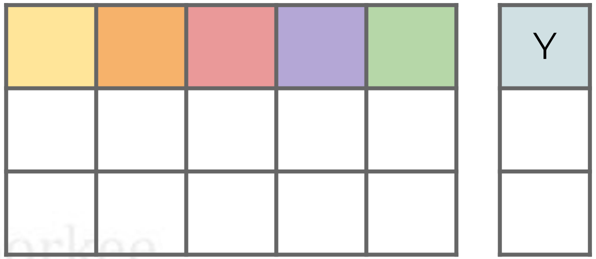
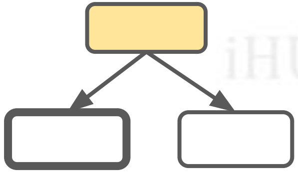


fig ref: pierian data



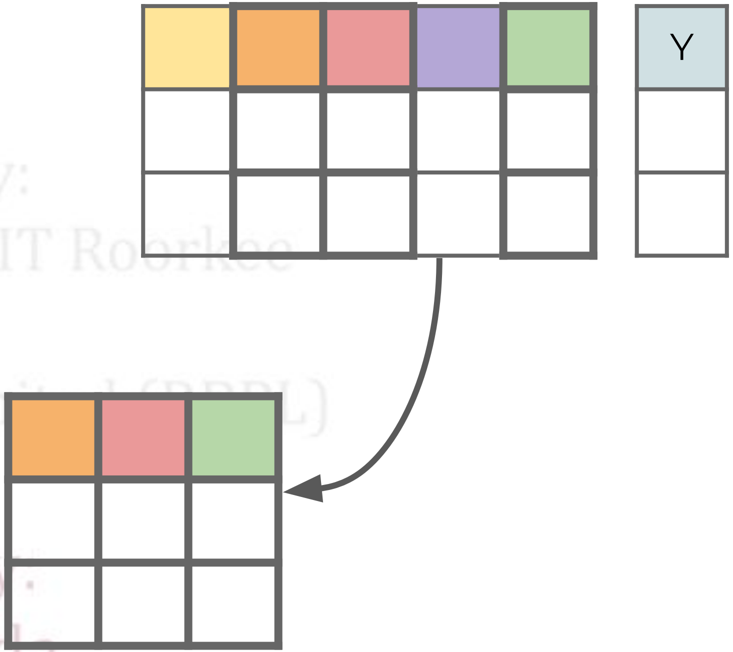
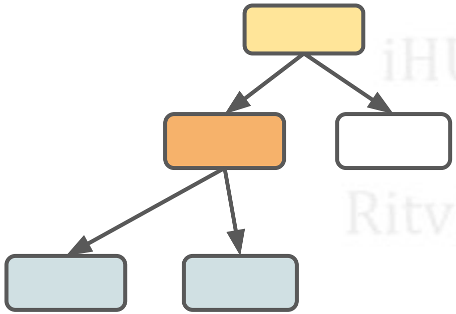


fig ref: pierian data



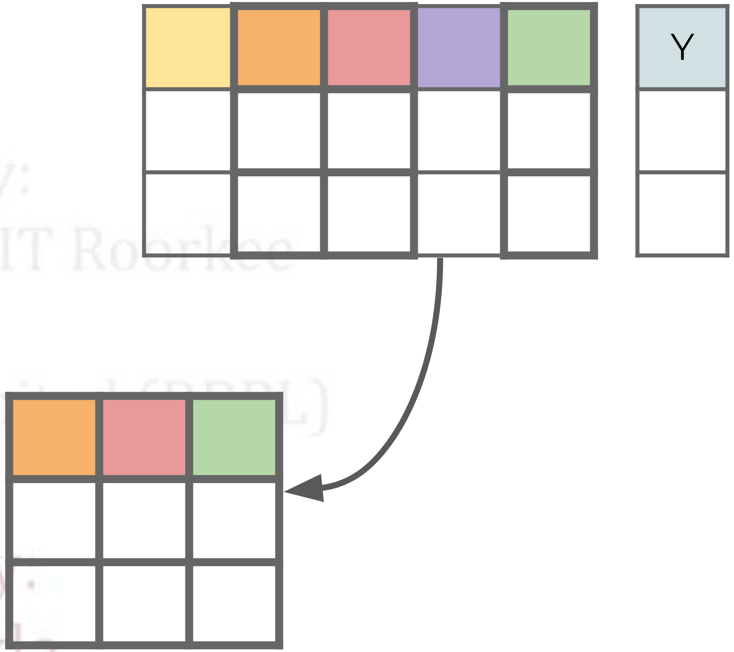
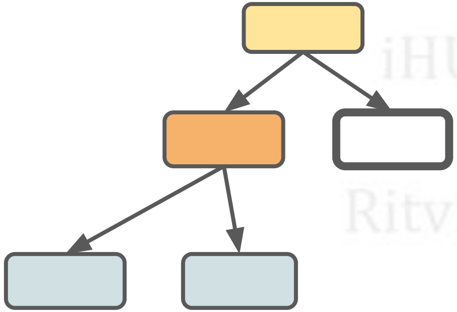


fig ref: pierian data



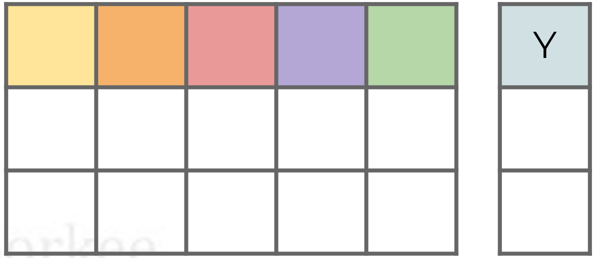
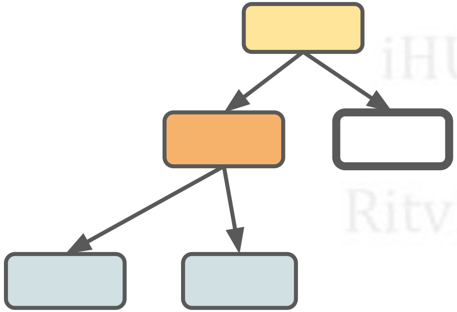


fig ref: pierian data



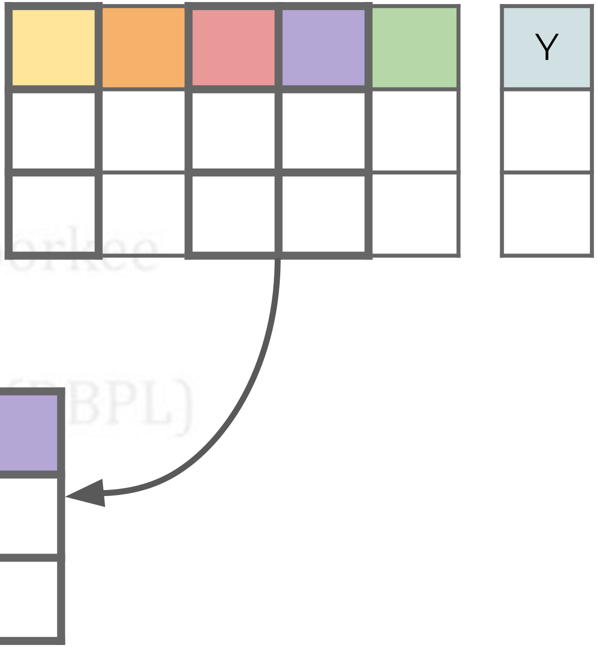
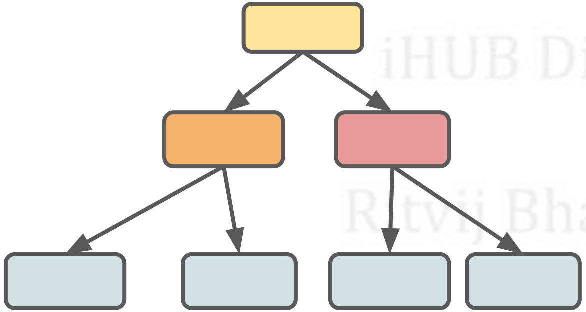


fig ref: pierian data



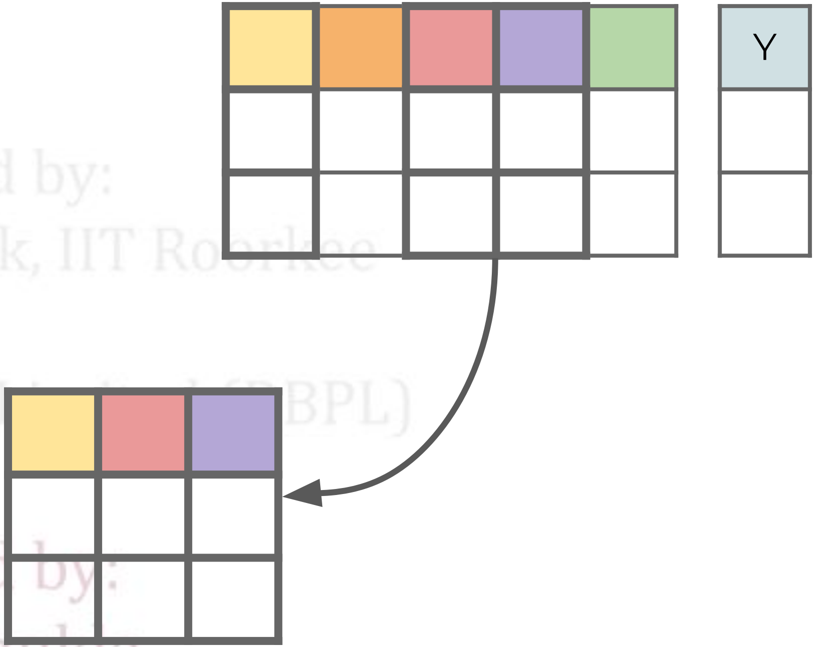
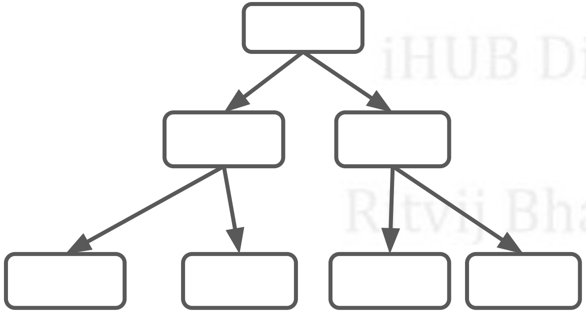
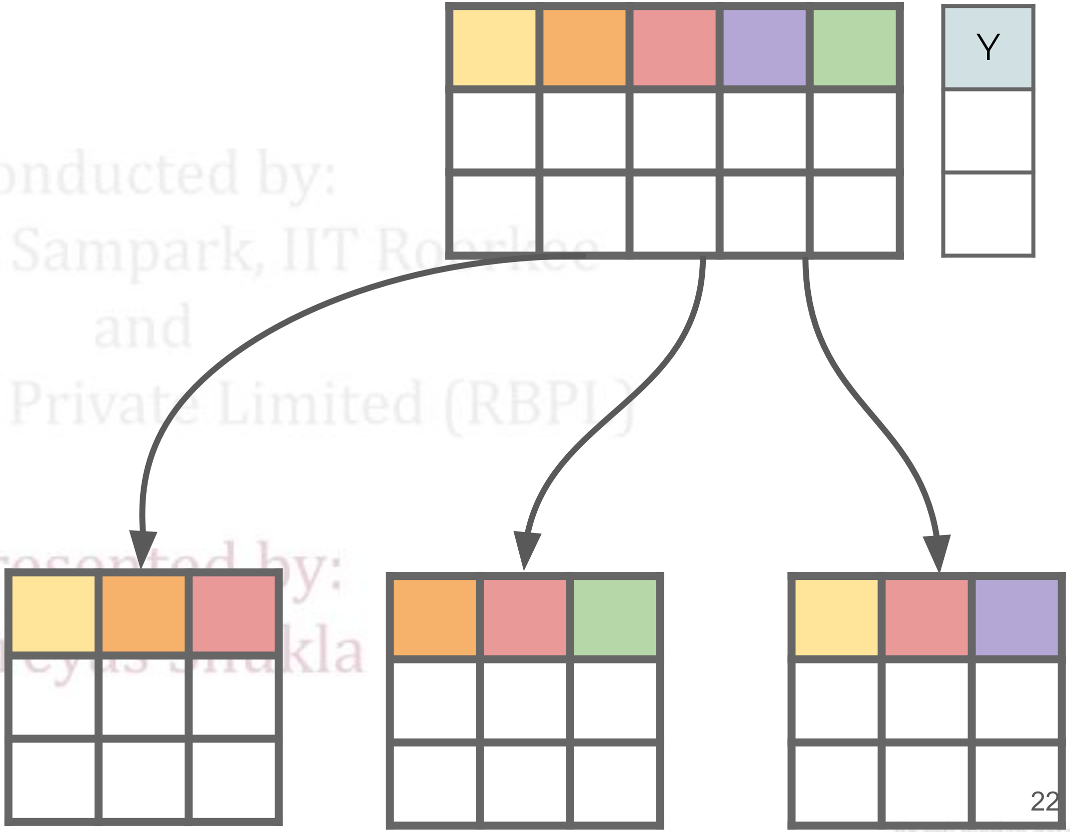


fig ref: pierian data





img ref: pierian data

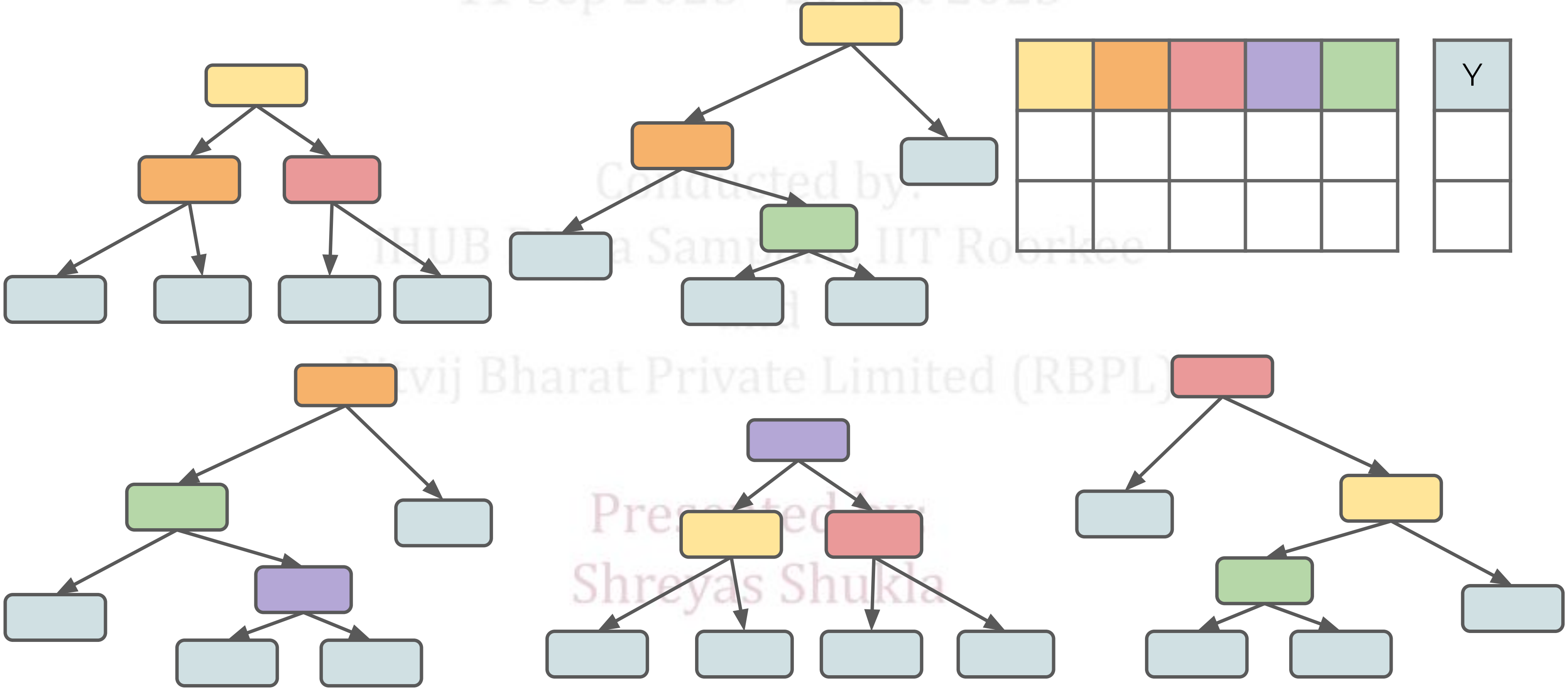


fig ref: pierian data

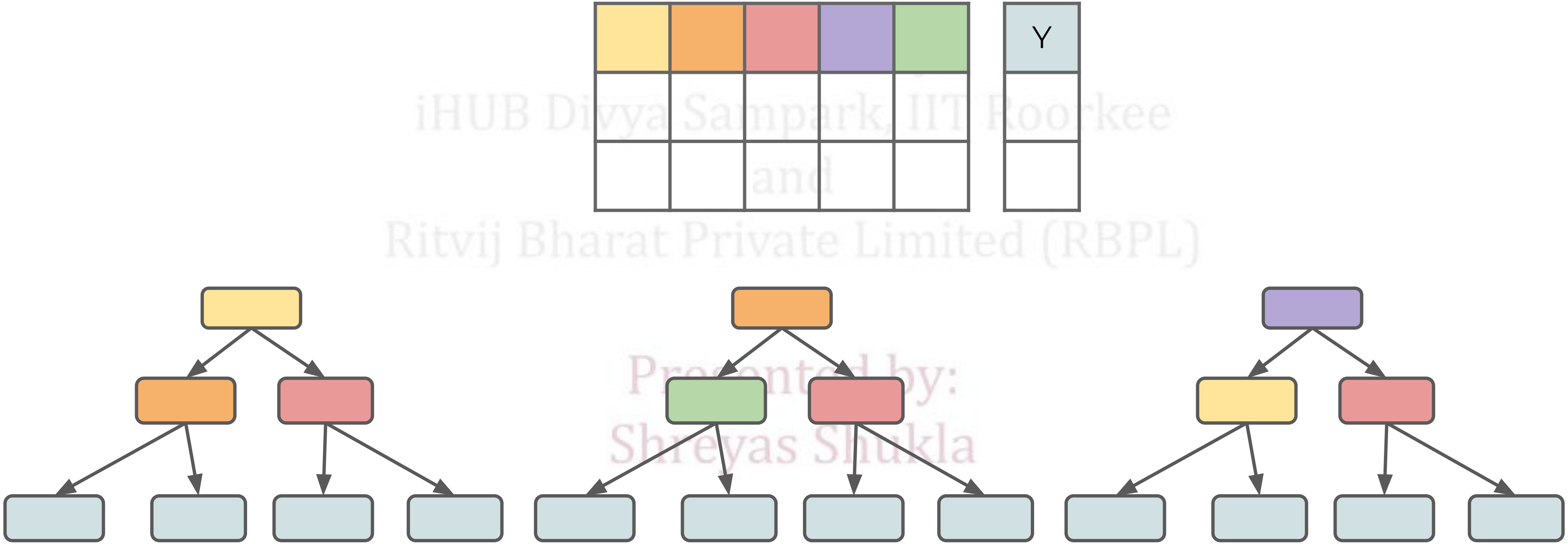


fig ref: pierian data

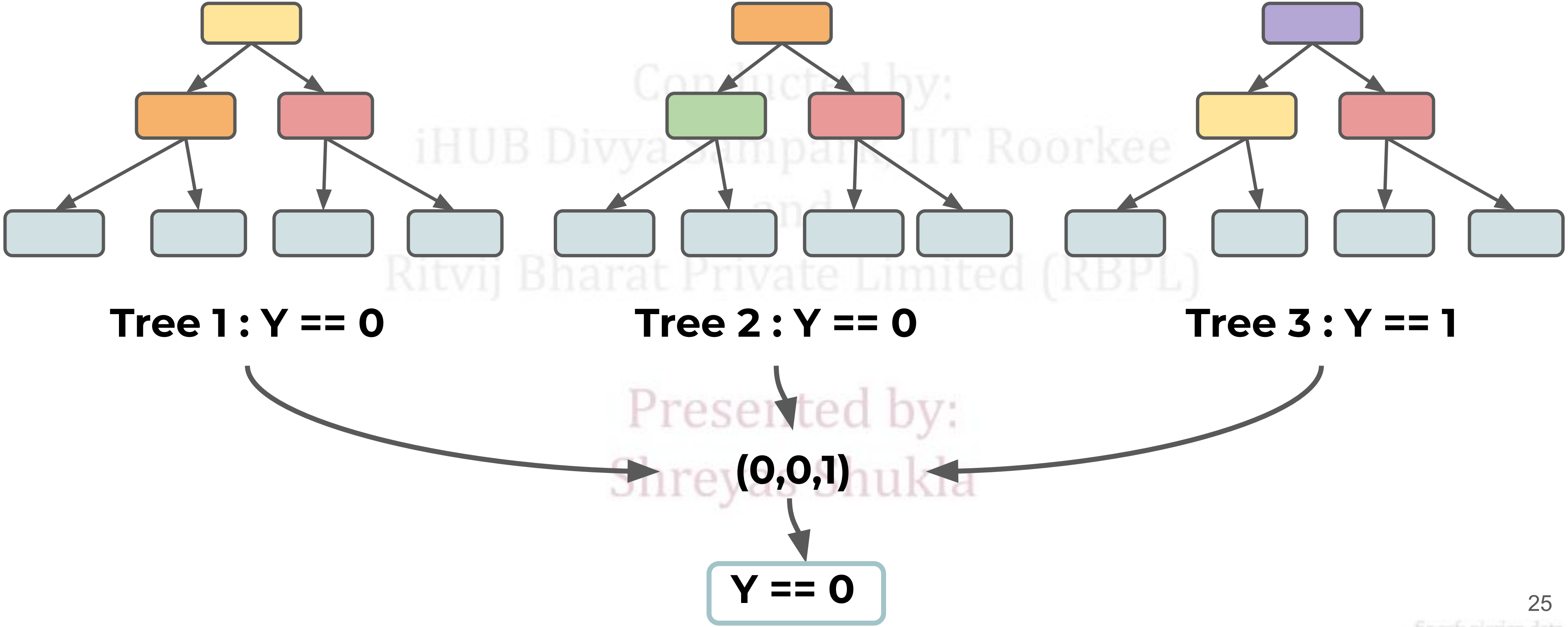


fig ref: pierian data

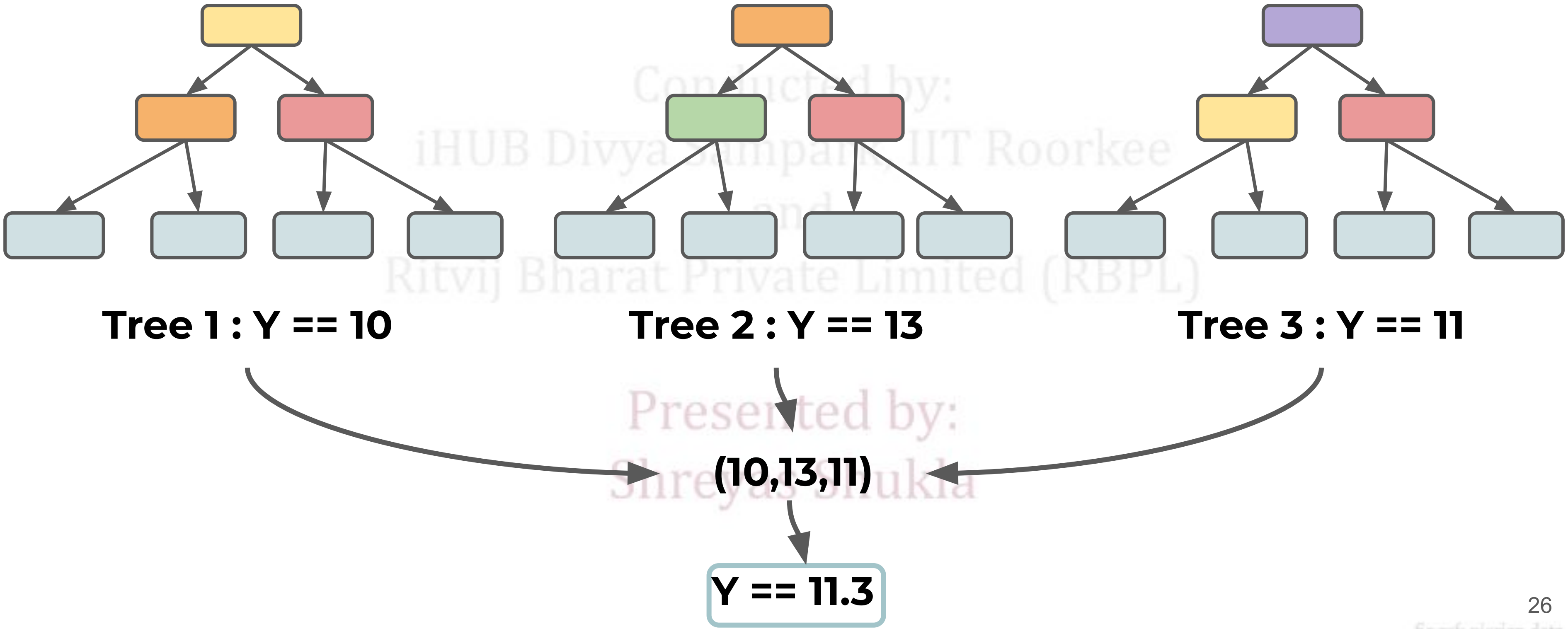
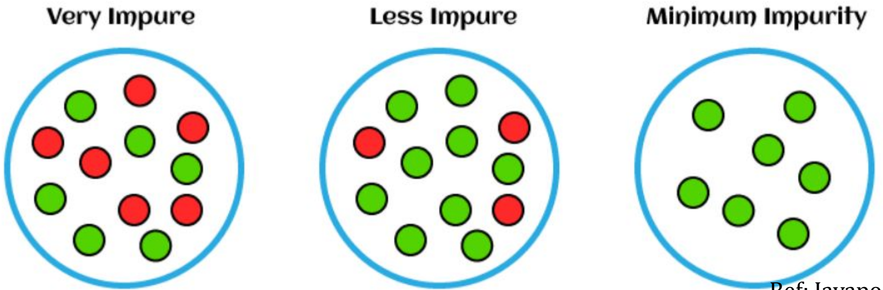


fig ref: pierian data

Random Forests

Hyperparameters

Entropy



Ref: Javapoint

```
class sklearn.tree.DecisionTreeClassifier(*, criterion='gini', splitter='best', max_depth=None, min_samples_split=2,
min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features=None, random_state=None, max_leaf_nodes=None,
min_impurity_decrease=0.0, min_impurity_split=None, class_weight=None, ccp_alpha=0.0) ¶                      [source]
```



```
class sklearn.ensemble.RandomForestClassifier(n_estimators=100, *, criterion='gini', max_depth=None, min_samples_split=2,
min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features='auto', max_leaf_nodes=None, min_impurity_decrease=0.0,
min_impurity_split=None, bootstrap=True, oob_score=False, n_jobs=None, random_state=None, verbose=0, warm_start=False,
class_weight=None, ccp_alpha=0.0, max_samples=None) ¶                                                        [source]
```


fig ref: pierian data

Therefore, Random Forest Hyperparameters:

- Number of Estimators: How many decision trees to use total in forest?
- Number of Features: How many features to include in each subset?
- Bootstrap Samples: Allow for bootstrap sampling of each training subset of features?
- Out-of-Bag Error: Calculate OOB error during training?


```
class sklearn.ensemble.RandomForestClassifier(n_estimators=100, *, criterion='gini', max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features='auto', max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, bootstrap=True, oob_score=False, n_jobs=None, random_state=None, verbose=0, warm_start=False, class_weight=None, ccp_alpha=0.0, max_samples=None)
```
[source]

Random Forests

Hyperparameters
Number of Estimators and Features

fig ref: pierian data

Random Forest Hyperparameters:

- Number of Estimators :
  - *How many decision trees to use total in forest?*

- Number of Features :
  - *How many features to include in each subset?*

fig ref: pierian data

Number of Estimators

- More the decision trees, more the opportunities to learn from a variety of feature subset combinations.
- Is there a limit to adding more trees?
- Is there a danger of overfitting?

fig ref: pierian data

“Random forests does not overfit. You can run as many trees as you want. It is fast.”

-Leo Breiman’s
(creator of Random Forests)

fig ref: pierian data

How to choose number of trees?

1. Reasonable Default Value: 100
2. Publications suggest 64-128 trees.
3. Cross Validate a grid search of trees.
4. Plot Error versus number of trees (similar to elbow method of KNN).
   –Should notice diminishing error reduction after some N trees.

fig ref: pierian data

Error vs. Trees

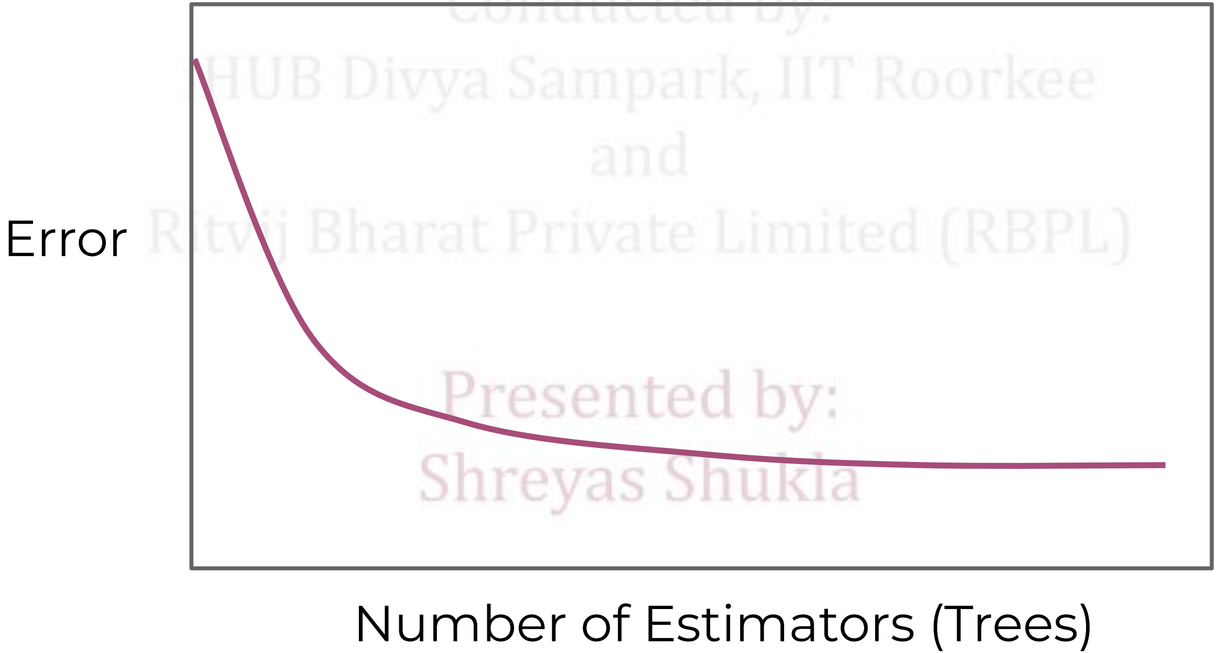


fig ref: pierian data

After a certain number of trees, two things that can occur:

- Different random selections don't reveal any more information. That is, Trees become highly correlated.
- Different random selections are simply duplicating trees that have already been created.

This allows us to be quite lenient in setting number of estimators hyperparameters, as overfitting is of minimal concern.

Now let’s discuss how to choose the number of features to randomly select at each split.

fig ref: pierian data

Random Forest Hyperparameters: Number of Features

How many features to include in each subset when splitting at a node?

fig ref: pierian data

What about Number of Features in Subset?

Original Publication suggested subset of $\log_2(N+1)$ random features in subset given a set of N total features.

fig ref: pierian data

"An interesting difference between regression and classification is that the correlation increases quite slowly as the number of features used increases."

- Leo Breiman's official page

Number of Features in Subset?

- Current suggested convention is sqrt(N) in the subset given **N** features.
- Later suggestions by Breiman indicated **N/3** may be more suitable for regression tasks, typically larger than sqrt(N).

Number of Features in Subset?

- As per ISLR, this can be treated as a tuning parameter, starting with **sqrt(N)**.
- It is likely you will need to adjust based on your specific dataset.

fig ref: pierian data

Hyperparameter Review:

- Number of Estimators:
  - Start with 100 as default, feel free to grid search for higher values.
- Number of Features for Selection:
  - Start with sqrt(N), grid search for other possible values (N/3).

Now let’s talk about Bootstrapping and Out-of-Bag Error!

fig ref: pierian data

Random Forests

Hyperparameters

Bootstrap Samples and OOB Error

fig ref: pierian data

- Random Forest Hyperparameters:
    - Bootstrap Samples: It *allow for bootstrap sampling of each training subset of features?*
    - Out-of-Bag Error: *Calculate OOB error during training?*

- Let’s understand “bootstrapping” in general terms...

fig ref: pierian data

What is Bootstrapping?

- Basically "random sampling with replacement".
- Let's see an example…

fig ref: pierian data

Bootstrapping

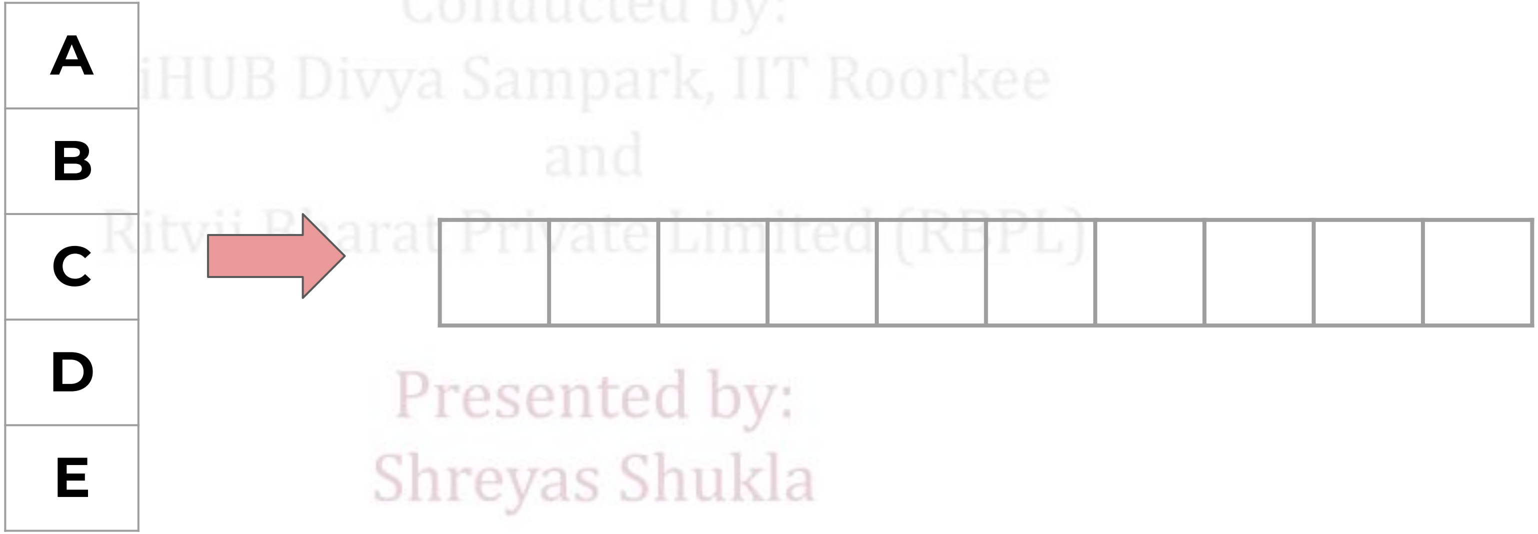


fig ref: pierian data

Bootstrapping

fig ref: pierian data

Bootstrapping

fig ref: pierian data

Bootstrapping

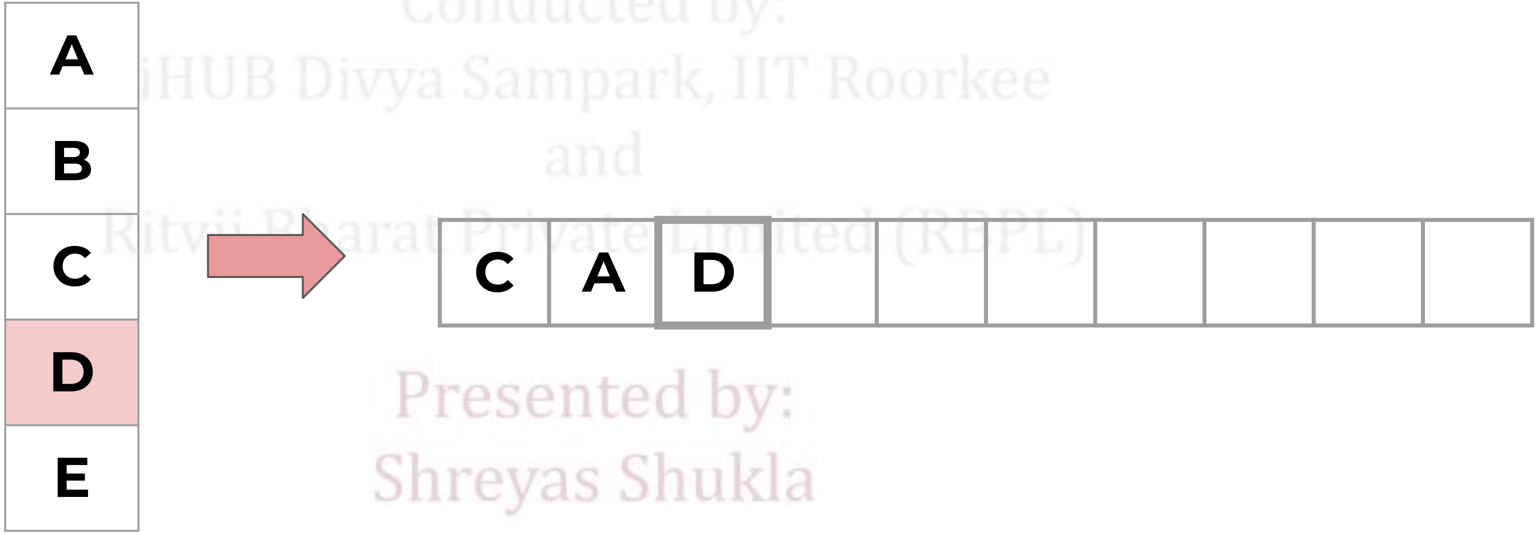


fig ref: pierian data

Bootstrapping

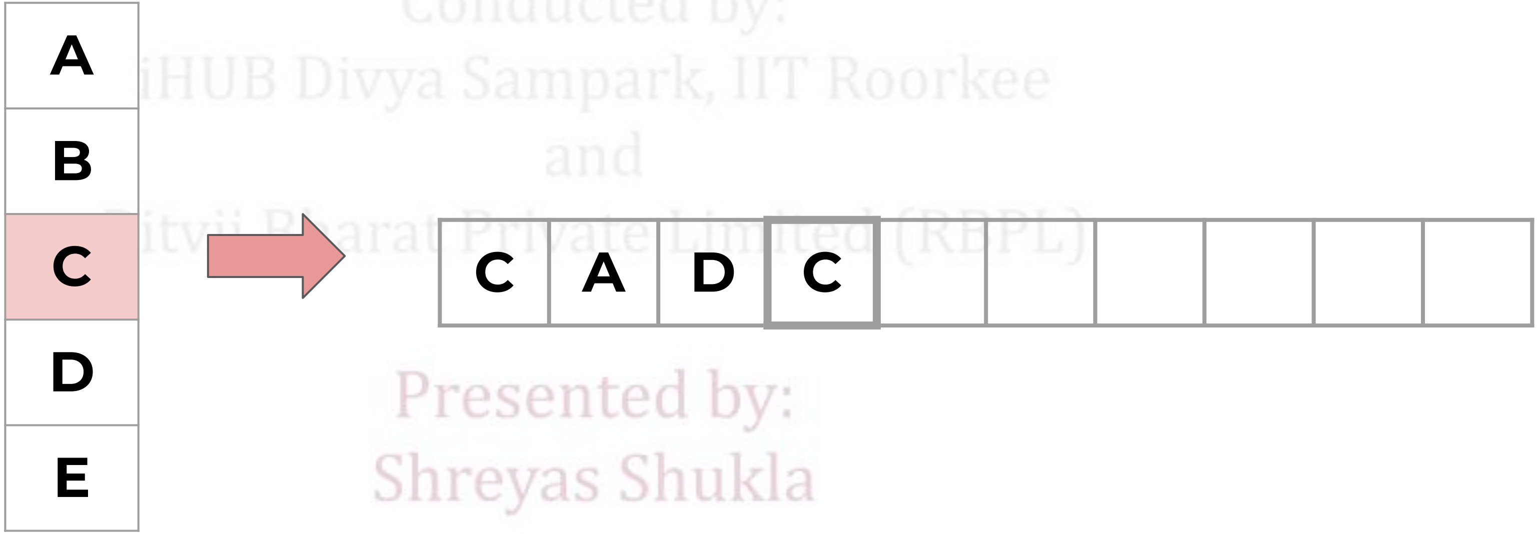


fig ref: pierian data

Bootstrapping

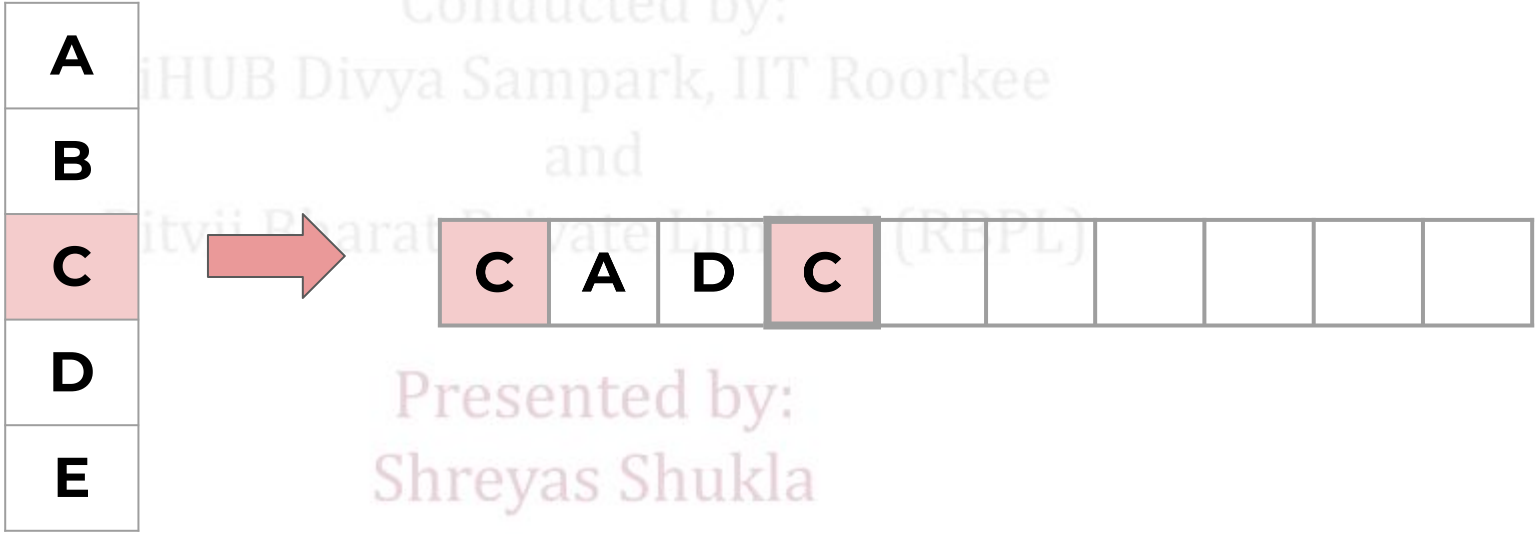


fig ref: pierian data

Bootstrapping

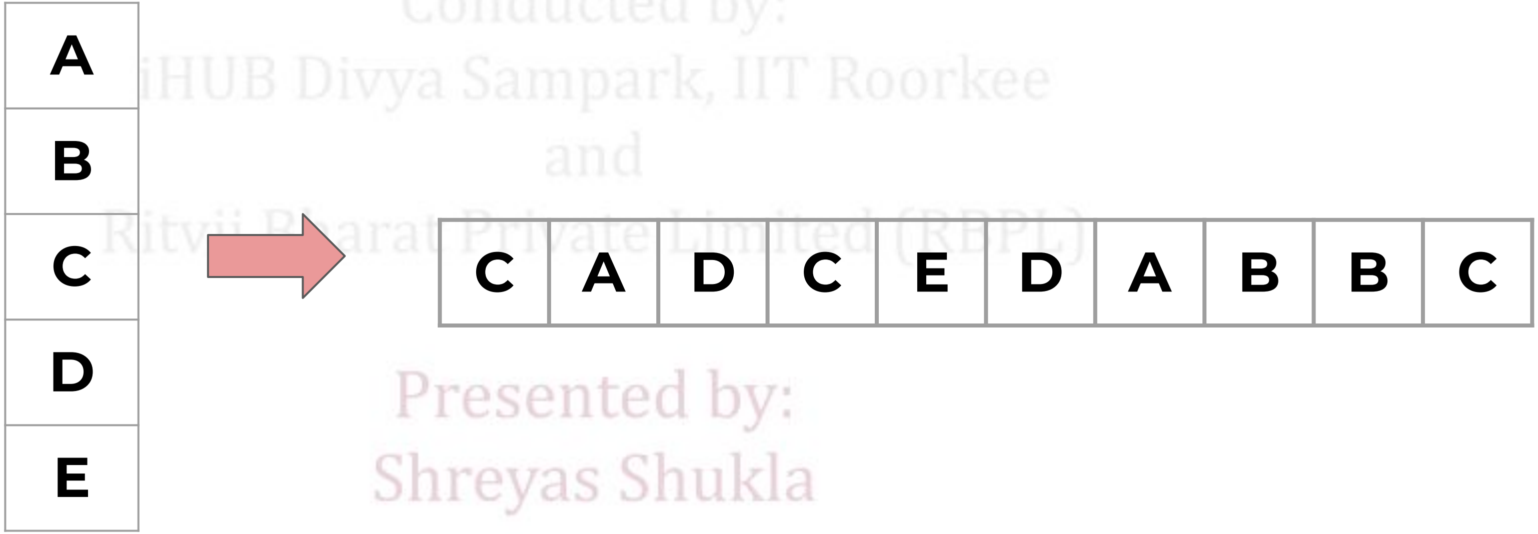
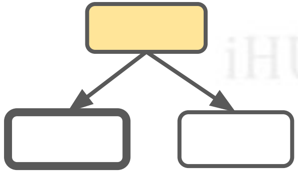


fig ref: pierian data

Bootstrapping

- Recall: for each split we are randomly selecting a subset of features.
- This random subset of features helps create more diverse trees that are not correlated to each other.

fig ref: pierian data



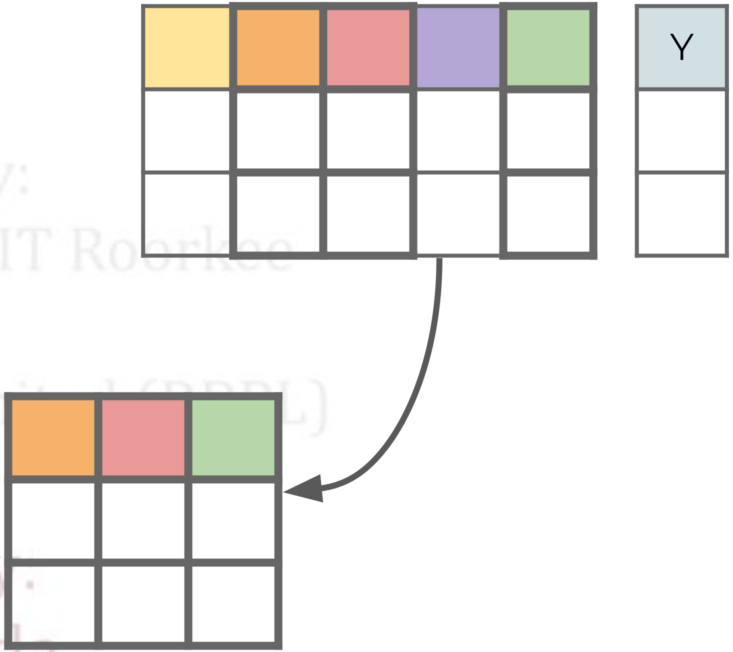
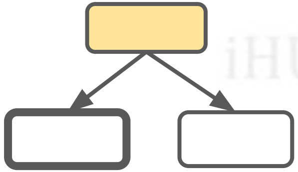


fig ref: pierian data

- Bootstrapping
    - To further differentiate trees, we could bootstrap a selection of rows for each split.
    - This results in two randomized training components:
        - Subset of Features Used
        - Bootstrapped rows of data

fig ref: pierian data



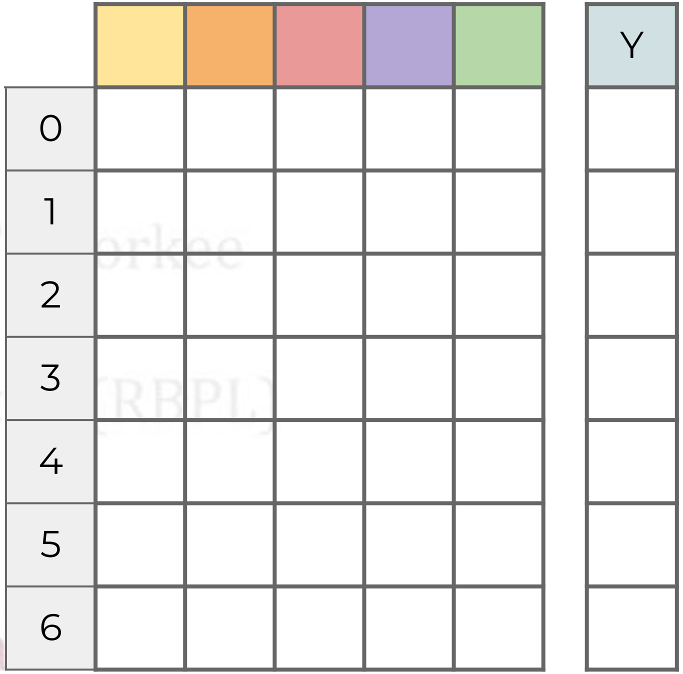


fig ref: pierian data

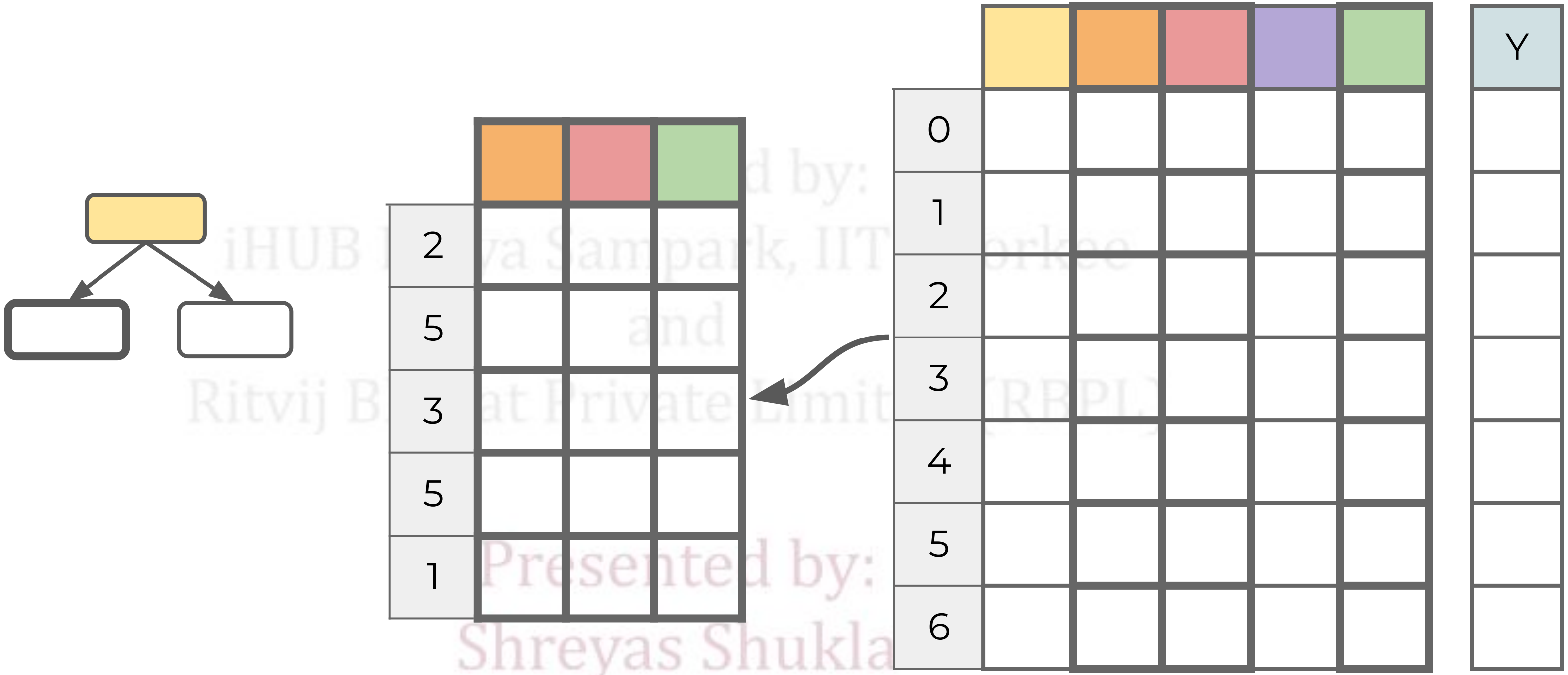


fig ref: pierian data

Bootstrapping can be set to False during training (it is True by default).

Bootstrapping is yet another hyperparameter meant to reduce correlation between trees, because trees are then trained on different subsets of feature columns and data rows.

Random Forest Hyperparameters:

Out-of-Bag Error

- *Calculate OOB error during training?*

fig ref: pierian data

Bagging

Recall that to actually use a Random Forest, we use **b**ootstrapped data and then calculate a prediction based on the **ag**gregated prediction of the trees:

- Classification: Most Voted Y Class
- Regression: Average Predicted Ys

fig ref: pierian data

Bagging

If we performed bootstrapping when building out trees, for certain trees, certain rows of data were not used for training.

fig ref: pierian data

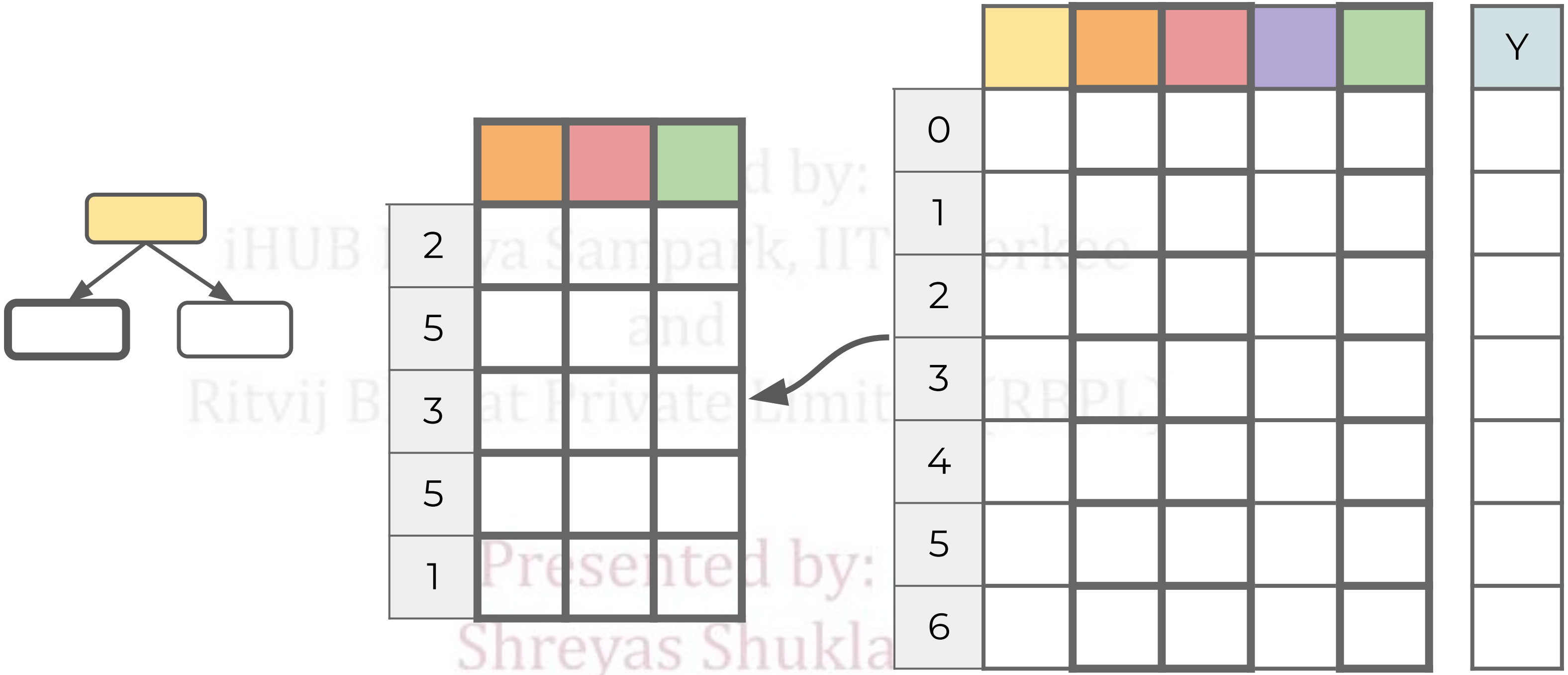
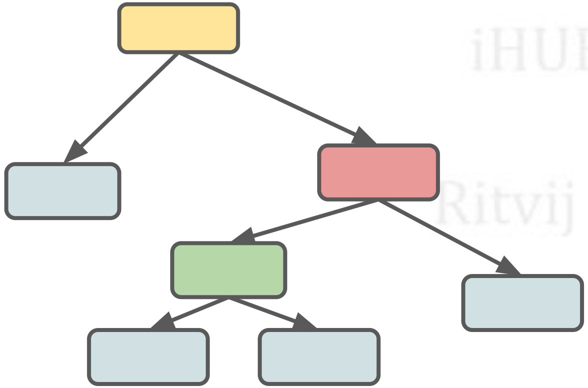


fig ref: pierian data



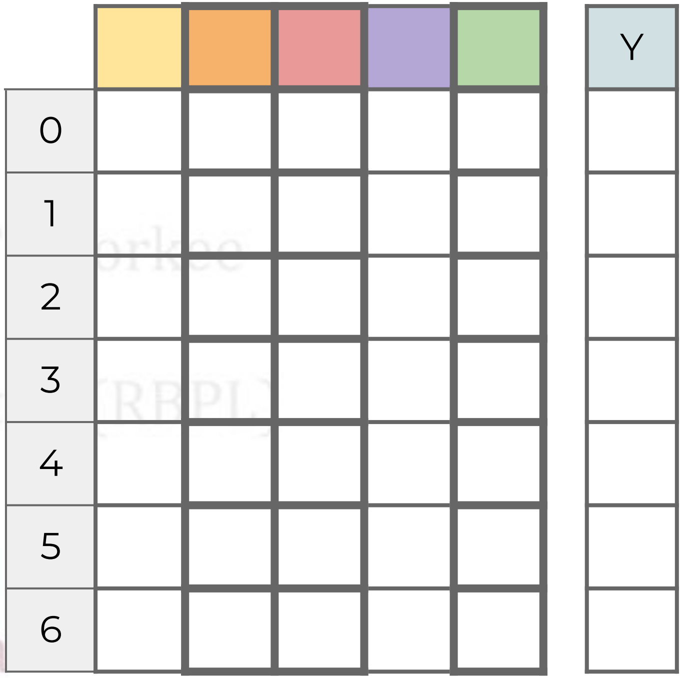


fig ref: pierian data

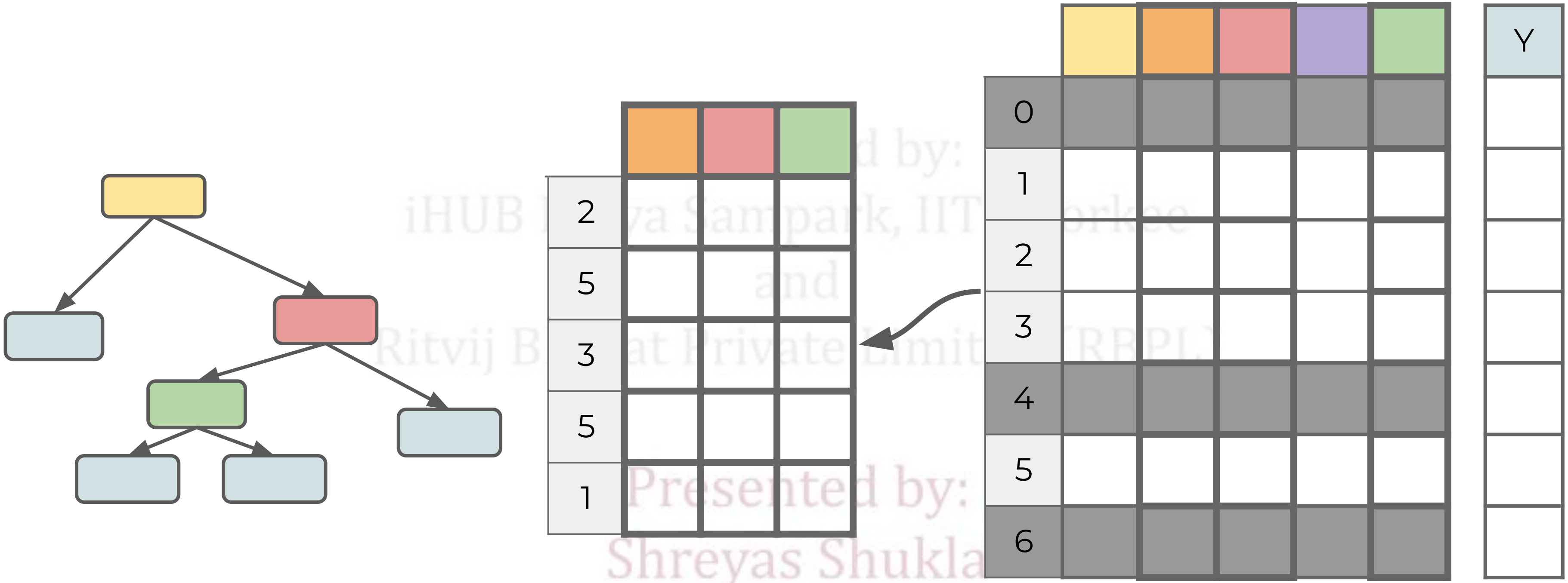
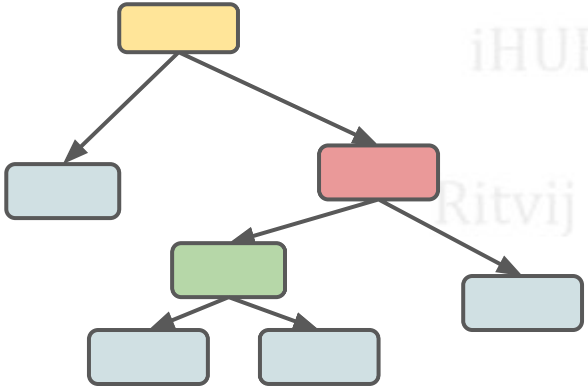


fig ref: pierian data



Out-of-Bag Samples
Not used for constructing some trees.

We could use these to get performance test metrics on trees that did not use these rows!

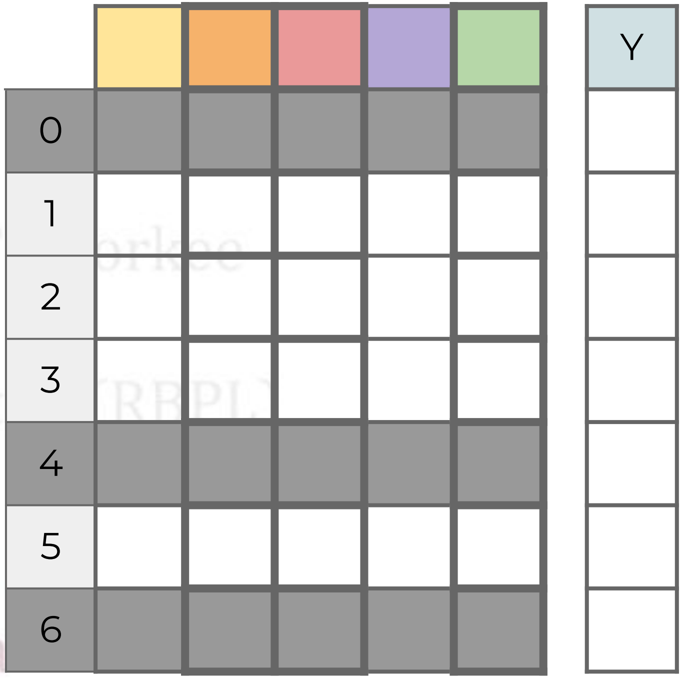


fig ref: pierian data

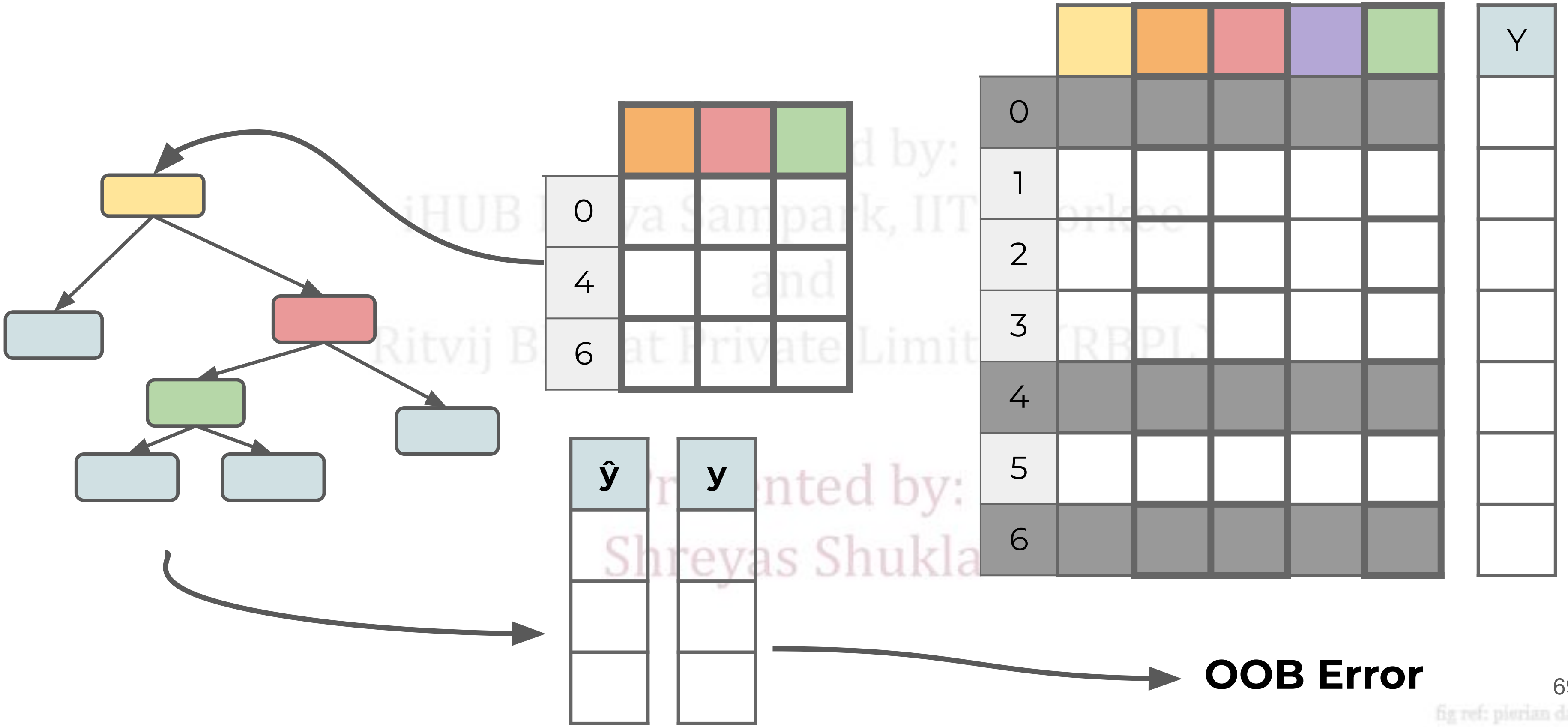


fig ref: pierian data

OOB Score is a hyperparameter that doesn’t really affect training process.

OOB Score is an optional way of measuring performance, an alternative to using a standard train/test split, since bootstrapping naturally results in unused data during training.

Note that OOB Score is also limited to not using all the trees in the random forest. It can only be calculated on trees that did not use the OOB data

Due to not using the entire random forest, the default value of OOB Score hyperparameter is set to False.

fig ref: pierian data

Let's Code !!

fig ref: pierian data