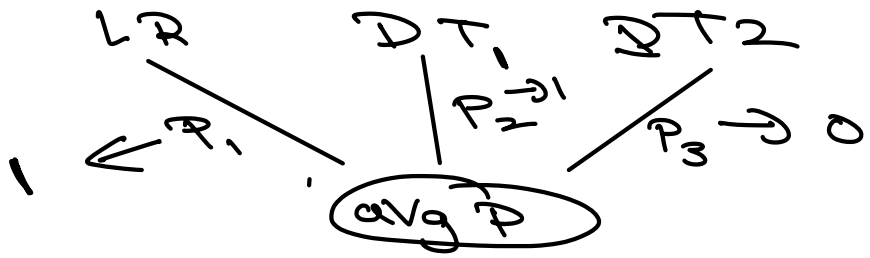
* Using DT classifier (max-depth=4)

Train-score ≈ 85 %

Test-Score ≈ 78 %

How to improve this ?



$\boxed{\text{Ensemble}}$

Base-Learners (as Unique as Possible) + Combine/Aggregate Result

Variants of Ensemble Technique

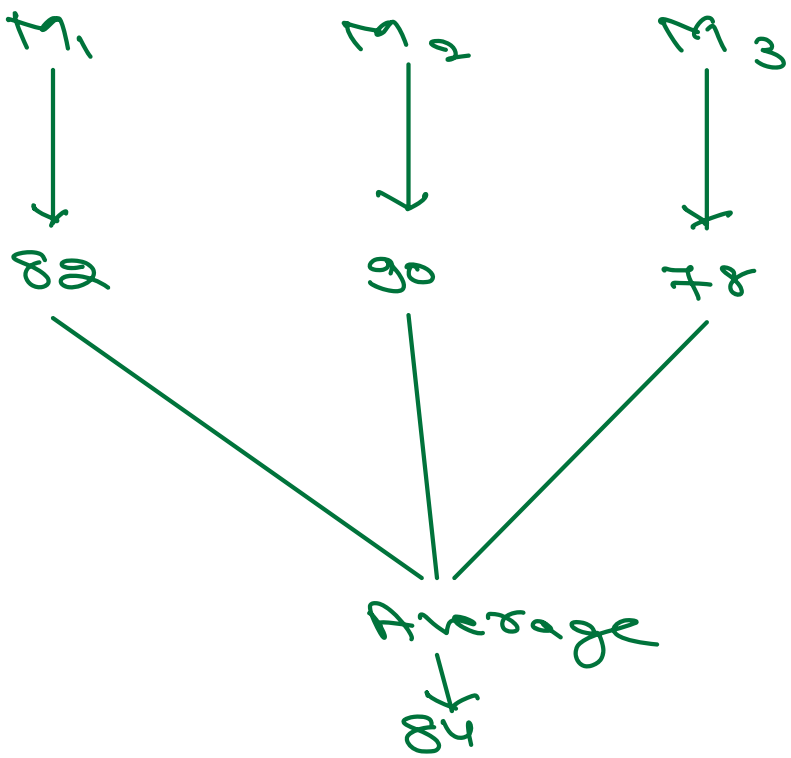
1) Bagging (Ex:- RF) *
2) Boosting* (Ex:- GBDT, XGBOOST, LightGBM
3) Stacking
4) Cascading

} not so important from industry perspective (Highly used ML Competition)

Bagging

(Bootstrap Aggregation)

Bootstrap Sampling + Aggregation



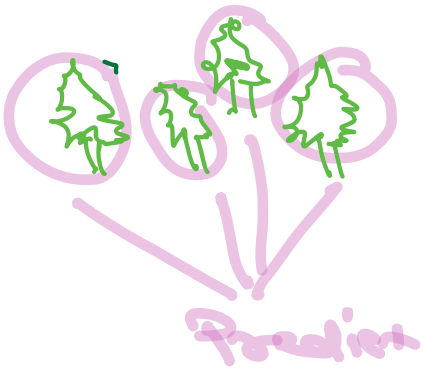
Random Forest

(Bagging Model)

Each tree shall be trained on Randomly Sampled subset of Data

1) Sample Rows
2) Sample Columns

(Row and Col Sampling)



RF ⇒ DT's + RS + C.S + Aggregation

Unique and different Trees

Prediction

$d = 10$

f_1	f_2	--	--	f_{10}

$n = 1000$

DT₁ ⇒ CS ⇒ 0.2
RS ⇒ 0.4

d

f_1	f_{10}

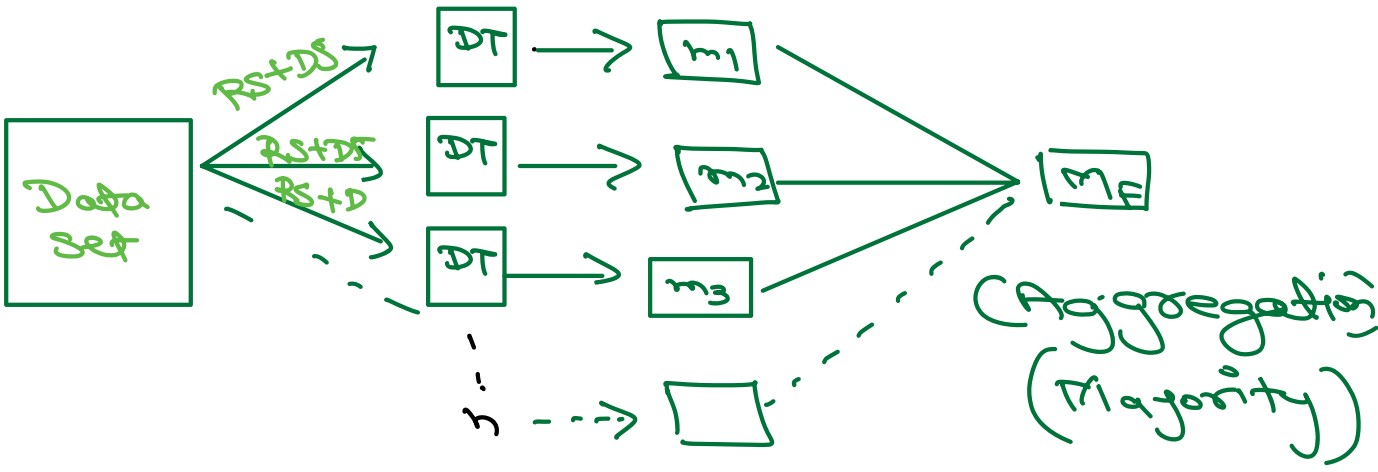
400 rows

Repeat this

n_estimators = 100

* RSR ⇒ rows
* CSR ⇒ Columns

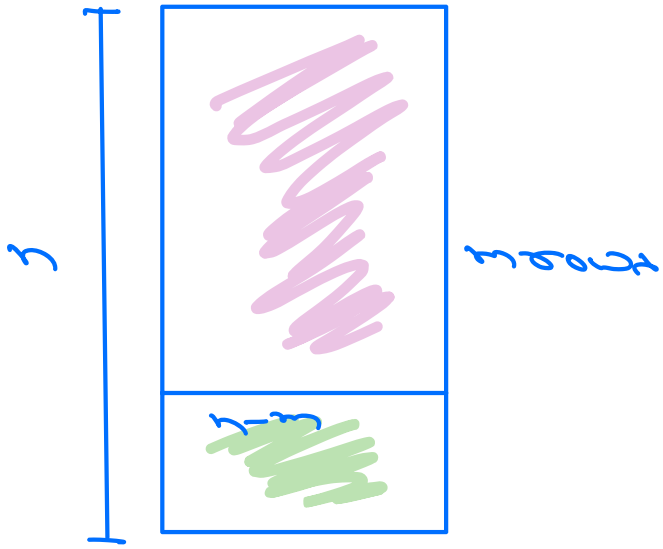
Bootstrap Sampling ⇒ Sampling with Replacement



CS + RS Leads to
⇒ Set of underfitted DT's
⇒ Different opinions

OOB

D ∈ 1000 row
R.S ⇒ 0.3
↓
300 rows



⇒ 

⇒ oob Evaluation

Trained

Evaluation

1000 − 300 ⇒ 700

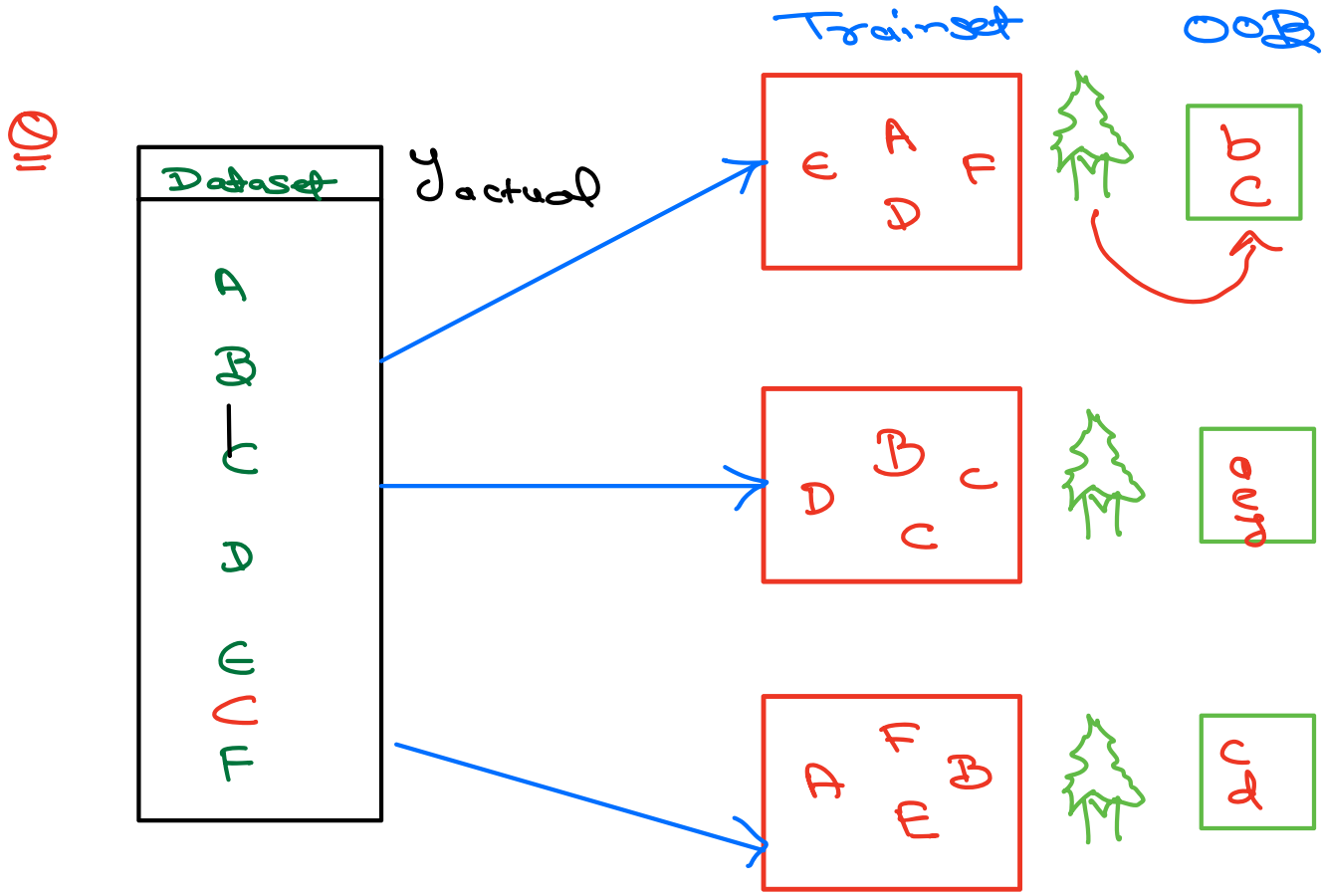
Row Sampling

DT2 ⇒ 700

OOB ⇒

Left out Rows after Row sampling from each tree can be used for Evaluating the performance of Model

OOB Score ⇒ Variation of Cross-Val Score



Bootstrap Sampling

OOb-score

$$P_c \Rightarrow \frac{dt_1(c) + dt_3(c)}{2}$$

$$y_c - P_c \Rightarrow Error_c$$

$$\sum_{i=1}^{n} Error_i$$

OOB-error

Pending

- Implement RF with Data
- Bias Variance in RF
- HyperParam Tuning with GridSearch an RandomSearch ?

Bias Variance Tradeoff

DT ⇒ High Variance (overfit)
+
Low Bias

Ideal Model ⇒ Low Bias and Low Var

Base Learner (DT) ⇒ Random Forest
(Low Bias
Low Variance)

+

Bootstrap aggregation
Bagging ⇒

Error ⇒ Bias + Variance + ∈
(Simple) Models (Underfitted)
(Complex) Models (over fit)

Training Parallelization

* Since Every tree is indepent, we can train using Multi-processing Jobs.

Hyperparameters

* n_estimators ⇒ No of trees
* Max_samples ⇒ R.S.R (0, 1)
* Max_features ⇒ Column Sampling (Sqrt, log, Num
* ccp_alpha ⇒ Cost Complexity pruning

Regression ⇒ Error + $\lambda \|w\|$

Tree ⇒ Error + λ (No of Terminal Leaf Node)

* GridSearch

⇒ Used for finding Best Combination of Hyperparameter

① No of trees ⇒ [10, 100, 1000, 5000]

② Max_depth ⇒ [5, 10, 20,]

No of tree \ ccp			
⇒	10, 5	10, 10	10, 20
⇒	100, 5	100, 10	100, 20
⇒	1000, 5	1000, 10	1000, 20
	5000, 5	5000, 10	5000, 20

4 x 3 ⇒ 12

Find Best model among 12

4 Notree

3 max_dep → 60

5 ccp_alpha

C.V → Cross-validation → 2 → 120

* Randomized Search →

Randomly Select a subset of Combination

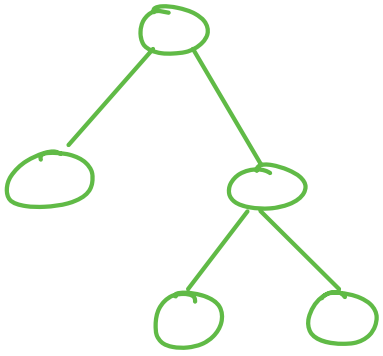
Feature Importance

$D \in C_1, C_2, C_3$

Random forest

Tree 1

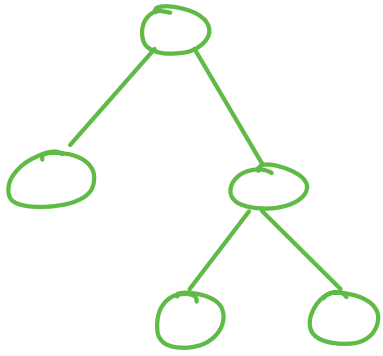
$C_1 C_2$



$FI(C_1)$

Tree 2

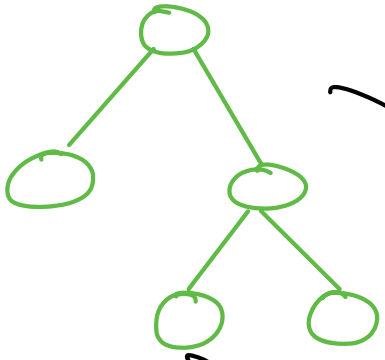
$C_1 C_3$



$FI(C_1)$

Tree 3

$C_2 C_3$



$FI(C_1) \ni 0$

66% C.S.R

$$FI_{Total} \Rightarrow \frac{FI(C_1) + FI(C_2) + FI(C_3) \ni 0}{3}$$