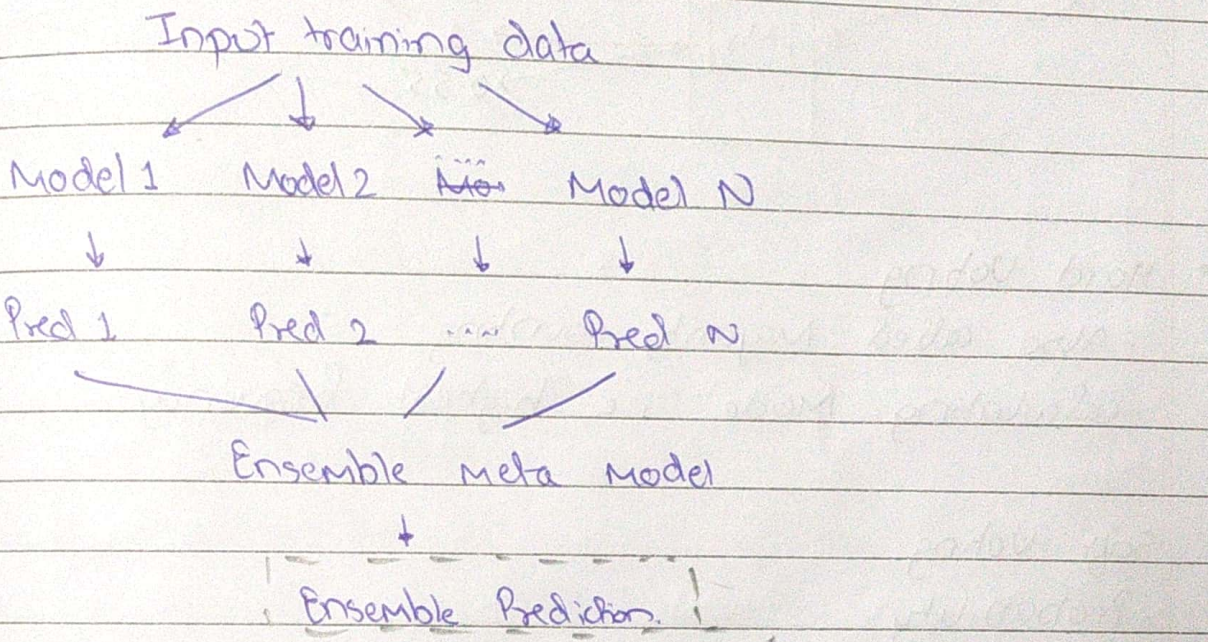
Ensemble Learning → ML

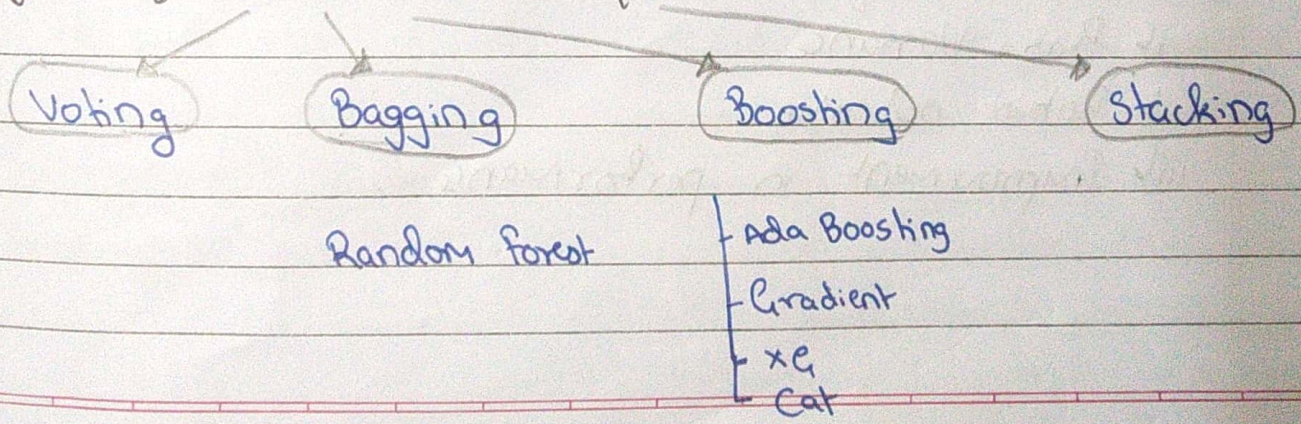
↳ Collection of many things

- Ensemble learning is group of models / Algorithms
- **Wisdom of crowd**

- ML technique that combines multiple models into a single model



→ Types of Ensemble Technique



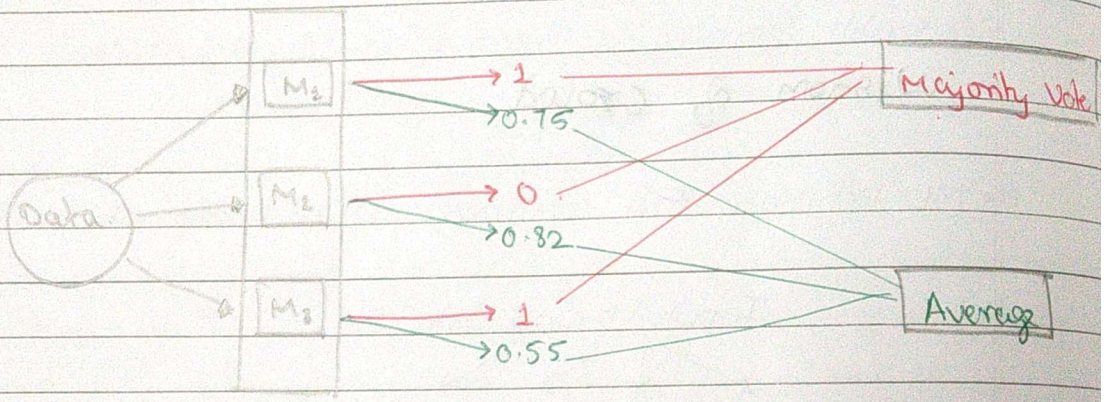
Voting Classifier

- Voting
  - Use different algorithms i.e Heterogeneous

Heterogeneous → Classification → Majority voting
CD, FF technique
→ Regression → Average voting

* Model Number
$M_1 \dots M_n$ ⇒ add number

Red : Classification
Green : Regression



- Hard Voting
  - Also called Majority counting
  - calculating Mode i.e highest frequency

- Soft Voting
  - Probability

→ Ensemble Benefits
  i) Bias Variance
  ii) Robust outcome
  iii) Improvement in performance

Bias	Variance
- error at the time of training	- error at the time of prediction
- At the time of pattern reading of data.	- test data how much varying from my predicted data.

• Generalised Model

To achieve this scenario, → - Bias ↓↓ Variance ↓↓

Ensemble Learning is used

→ When to use Ensemble learning
- Good Accuracy
- Competition

• Bagging Classifier
- Homogenous, parallel Learning

Bagging

Bootstrap (Bagg) Aggregation.

- way of sampling data
- 3 way of sampling
  1. Row sampling
  2. Column Sampling
  3. Combined
- Sampling with Replacement
- Sampling without replacement

- collecting final output
- either by majority voting or average value.

• Random Forest is special case of bagging
↳ by default we are using / have decision tree.

• Bagging classifier takes random subsets of data & aggregate their predictions to form final prediction.

• Row sampling : Taking sample from ~~every~~ every column & subset from rows.

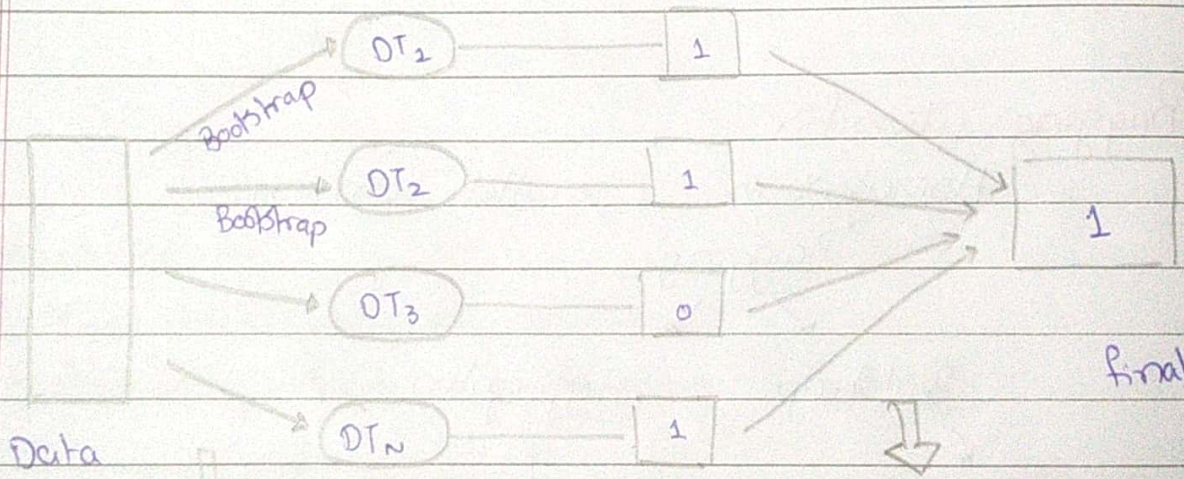
Ex: Sample-1 : $\{R_1, R_2\}$ [weight, height, Gender
S-2 : $\{R_3, R_4\}$ [— " —]

• Column Sampling : All rows and subset of colums.

Eg: S1 → 5 rows [subset of column]
S2 → 5 [— " —]

* Random Forest

– Homogenous, Bootstrap + Aggregation, Bagging technique.



• More Generalised Model
• Low Bias, low Variance

• classification
↳ based on majority

• Regression
↳ based on average

* oob - score [Out of bag score]

Ex: we have rows numbered 1 to 10 having columns as weight & height.

→

- So performing bootstrap with replacement.

$DT_1 = [1, 2, 3, 1, 4]$, $DT_2 = [2, 3, 3, 4, 1]$, $DT_3 = [1, 3, 3, 2, 2]$

Training

- Never selected record = {5, 6, 9, 10} we perform validation/testing Evaluation of the Model

whatever validation score we will get is known as "oob - score".

→ Bagging & Random Forest

Bagging	Random Forest
- Bootstrap Aggregation	- Bootstrap Aggregation
- Any sort of estimator (Model) [KNN, SVM, LRC...]	- Special case of bagging, selection DT.
DT	- Node level sampling

Bagging → RF

differ in column Sampling [Feature Sampling]

- If we select 2 feature, they remain same till the end.

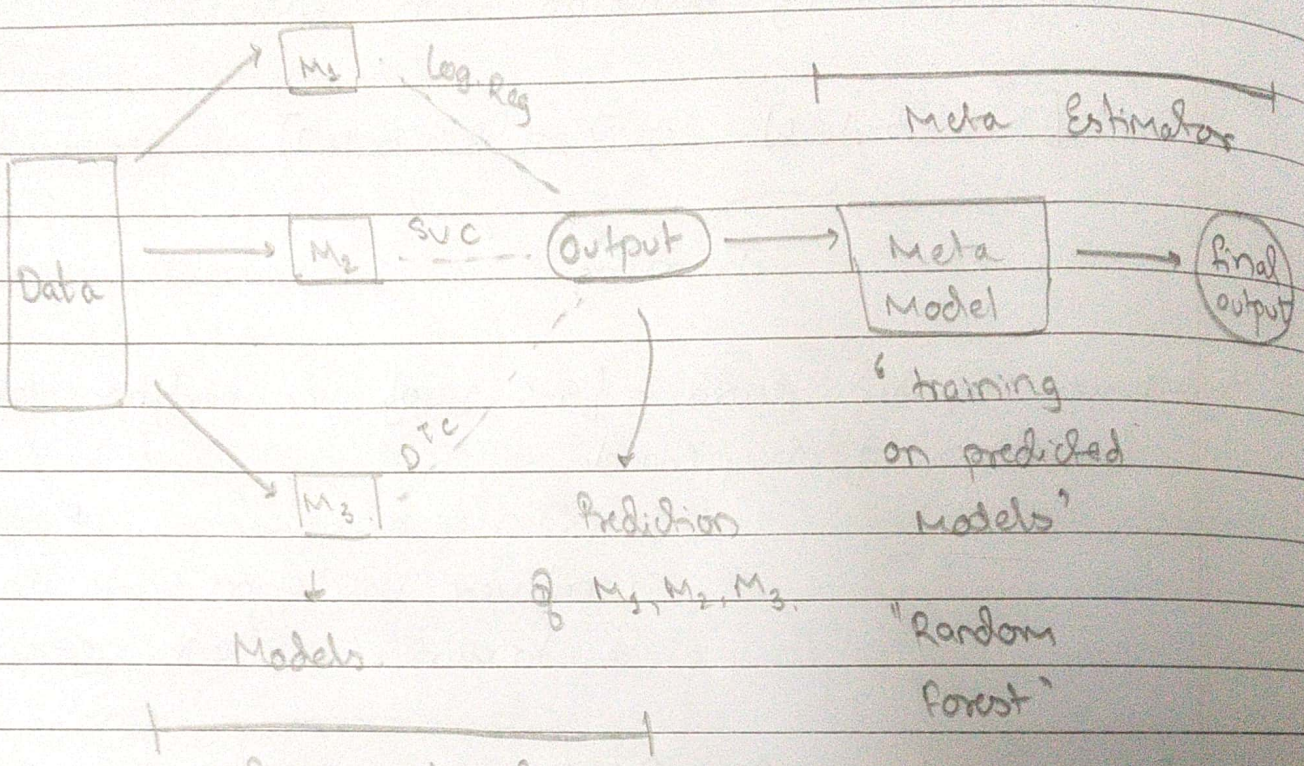
- **Stacking**

  Heterogeneous , Extension of voting.

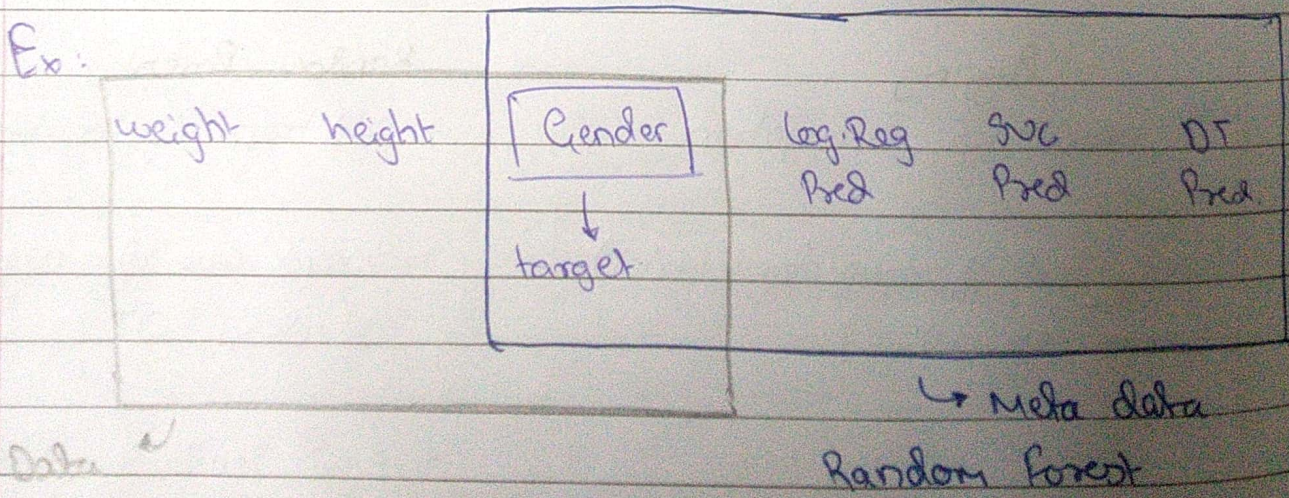
  2 stages → 1) Base Model

  2) Meta Model

  ↳ Data about data



Ex:



- **Blending** : Data Division

  To generalize Model

  To reduce overfitting

→ Information Gain

$$IG = \text{Impurity of Parent Node} - \left[\frac{\text{Sample}}{TNS} * \text{right impurity} + \frac{\text{Sample}}{TNS} * \text{left impurity} \right]$$

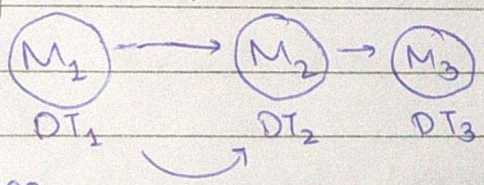
where,

TNS = Total number of samples

Sample = Sample inside that particular node.

- **Boosting**
  - Sequential technique, Homogeneous
  - Ex: Ada Boosting, Gradient, XgBoost, Cat Boosting
  - Boosting

* Ada Boosting



- DT_1 is created first than DT_2 and so on.

- Additive learning

$$f = \alpha_1 M_1 + \alpha_2 M_2 + \alpha_3 M_3 + \cdots + \alpha_N M_N$$

f: final prediction, α: weight, M = Model

↳ different for every model

→ Steps

1] Stump: Base estimator

↓

DT

↓

Max_depth = 1

⇓

Stump

Entropy & Gini-impurity

⇓

Information Gain

2] Performance of the stump:

$$POF = \frac{1}{2} \ln\left[\frac{1-TE}{TE}\right]$$

, TE : Total Error
ln = $\log_e$

3] Update the weight of incorrect & correct classified point

→ For correct classified Point

$$= \text{weight} \times e^{-POF}$$

∵ POF : Performance of stump

→ For incorrect classified point

$$= \text{weight} \times e^{POF}$$

4] Normalize the weight.

- addition for all the weight will be = 1

5] Creating a Bucket.

→ large bucket size will have more probability to select the datapoints.

↳ To achieve final goal → (Select a data for 2nd Stump or DT_2)

- More focus on those data points which are miss classified

↓

More weight

↓

Performing random sampling using bucket.

Gradient Boosting

- works on additive modeling → combination of multiple model with that we are finding out complex pattern

Gradient Boosting = $f_0(x) + f_1(x) + f_2(x) + \ldots + f_n(x)$

Base Model + $\alpha M_1 + \alpha M_2 + \ldots + \alpha M_n$

↓ avg value of output column

α = Learning Rate, hyperparameter
↳ same for all models

- Min_leaf_node = 8 & Max_leaf_node = 32
- Sequential

XGBoost [eXtreme Gradient Boosting]

- It is a library & uses Gradient Boosting Algorithm.

→ Similarity Score

$$S.S = \frac{[\text{Sum of Residual}]^2}{\text{No. of Residual} + \lambda}$$ ← hyper parameter.

λ: Regularization Parameter.

Initially $\lambda = 0$

Example : Data :

cgpa	Package	B.E	Res 1
6.7	4.5	7.37	$-\frac{2.8}{7}$
9.0	11.0	7.37	3.7
7.5	6.0	7.37	-1.3
5.0	8.0	7.37	0.7

1) Base Estimator [BE] = Avg of all values

$$\therefore \frac{4.5+11.0+6.0+8.0}{4} = 7.37$$

2) Residual [Res 1]

- Subtracting Package - B.E

- Training DecisionTree 1 based on res1.

Steps: i) Sort the features [cgpa]

ii) Calculating Adjacent Average.

- Sorting.

5.0 > 5.85
6.7
7.5 > 7.1
9.0 > 8.25

- $S.S = \frac{(-2.8+3.7+(-1.3)+0.7)^2}{4} = \frac{(0.3)^2}{4}$

→ similarity score for cgpa = 0.02
↳ original form

→ Splitting Criteria.

1) 5.85

cgpa < 5.85

0.7	-2.8, -1.3, 3.7

$SS_1 = \frac{(0.7)^2}{1+0} = 0.49$

$SS_2 = \frac{(-2.8-1.3+3.7)^2}{3+0} = 0.05$

→ Gain = $SS_{left} + SS_{Right} - SS_{root}$
= 0.49 + 0.05 - 0.02
= 0.52

2) cgpa < 7.1

0.7, -2.8 | -1.3, 3.7

$SS_L = \frac{(0.7-2.0)^2}{2}$ $SS_R = \frac{(-1.3+3.7)^2}{2}$

$= 2.20$ $= 2.88$

Gain = $SS_L + SS_R - SS_{root}$

$= 2.20 + 2.88 - 0.02$

$= 5.06$

3) cgpa < 8.25

0.7, -2.8, -1.3 | 3.7

$SS_L = 3.85$ $SS_R = 13.69$

Gain = $SS_L + SS_R - SS_{root}$

$= 3.85 + 13.69 - 0.02$

$= 17.52$

Highest Gain = 17.52

XGBoost

Flexibility | Speed | Performance

1) Flexibility :- Cross function
- Multiple language Support
- Integration with other library & tools
- Support all kind of ML problems

2) Speed : Parallel Processing → Multiple core's
- Optimized data Structure
- Cache Awareness
- Out of Core Computing
- Distributed Computing Master Slave
- GPU support

3) Performance :- Regularized Learning Objective
- Efficient split finding [weighted Quantile Sketch + Approximate tree learning]
- Tree Pruning [Post, Pre]
- Handling Missing values.