Module - 3

Ensemble learning →

3.1 Understand Ensembles

K-fold Cross Validation

⊖ Boosting

Stumping (AdaBoost)

XGBoost

3.2 ⊖ Bagging

Subagging

Random Forest

Comparison with Boosting

Different ways to Combine Classifiers

* Ensemble. ->

* In ML, ensemble is a model that combines the prediction from two or more models.

* The models that contributes to Ensemble are known as ensemble members.

* The members may or may not be trained on same training data and they may be of same type or different type.

* It is very powerful method to improve the performance of the model

* It is technique that uses group of weak learners in order to create a strong and aggregated learner.

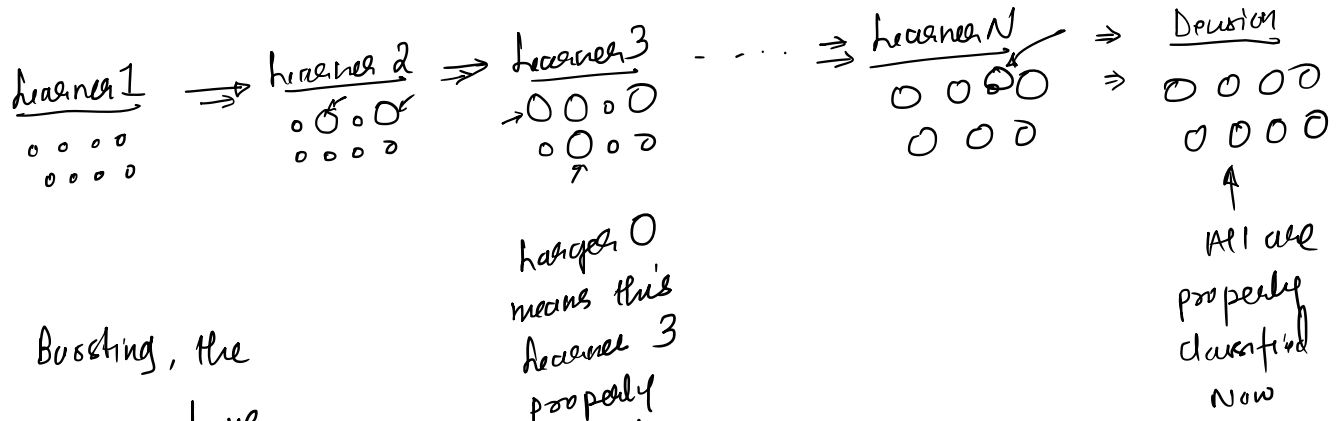
-> The Ensemble technique helps to reduce the Variance (By Bagging) and Bias (By Boosting) and thus helps in improving the predictions.

-> Boosting model: * It falls inside family of Ensemble method.

* It consist of filtering or weighting the data that is used to train team of weak learners, so that the new learner can give more weight on sample that is poorly classified by previous learner.

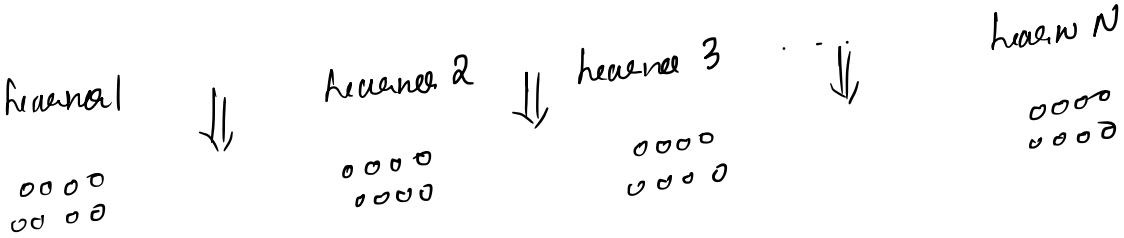
-> In Boosting the learners are trained

Sequentially.



* In Boosting, the learners can have diff importance or weight (represented by diff sizes of circles).

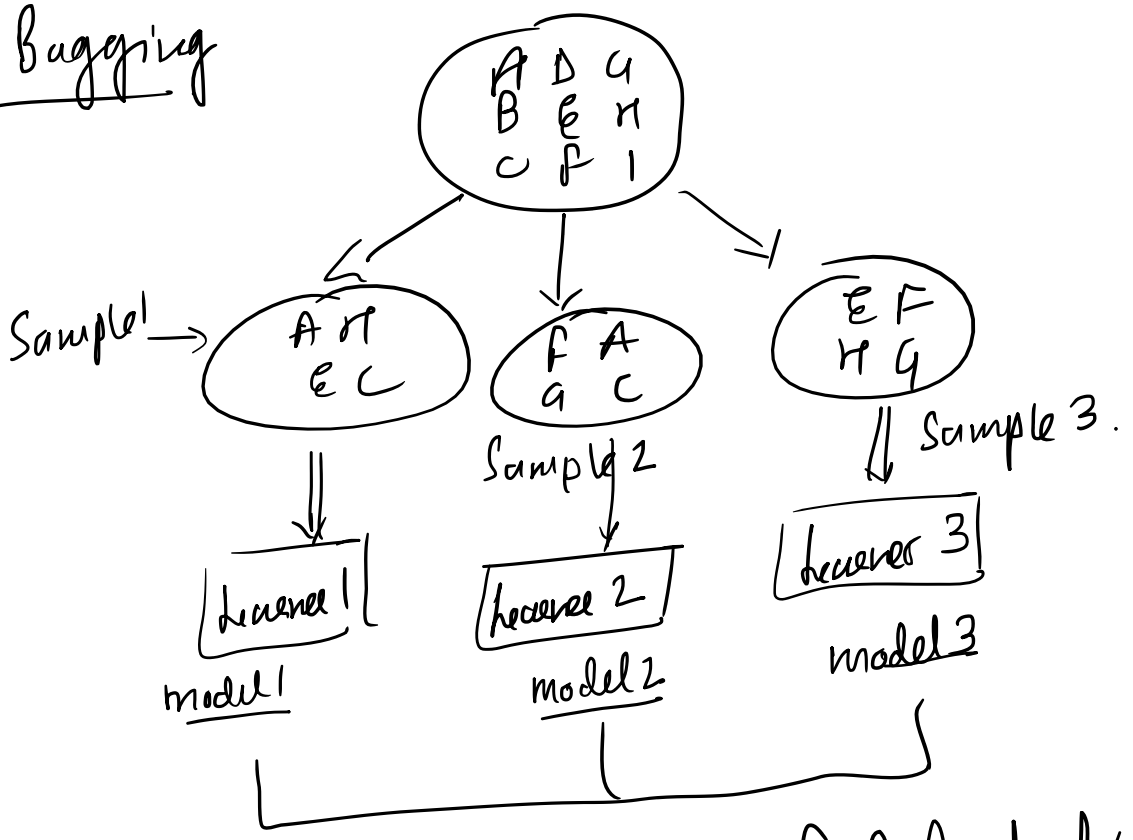
Bagging:



⇒ In Bagging the weak learners are trained in parallel using randomness

⇒ All learners have same weights.

Note Bagging



* Bagging helps in reducing variance
* Boosting helps in reducing Bias.

Cross Validation :→

→ In Supervised M.L

→ Train a model on a Dataset

→ Trained model is used to predict the target given new sample.

→ How to know if model we have trained will produce effective and accurate result on new input

Cross Validation → It is process that ensures the model will perform well on new Data.

Split the Dataset
- Training Set (60%)
- Validation Set (Hold-out set) (20%)
- Test Set (20%)

Training Set = part of data on which model is trained.
(This dataset will help to * build model)

***Validation Set** → * Evaluate the Model

→ Will help to chk if model overfits or Underfits.

→ Update the parameters and again train the model.

→ Repeat this until the model performs best on Validation set

→ typical ...

Validation set

Test Set ⇒* Prediction.

→ The fully trained model after being evaluated on validation set can be used on test set to generate Estimation.

Q) Types of Cross Validation ⇒

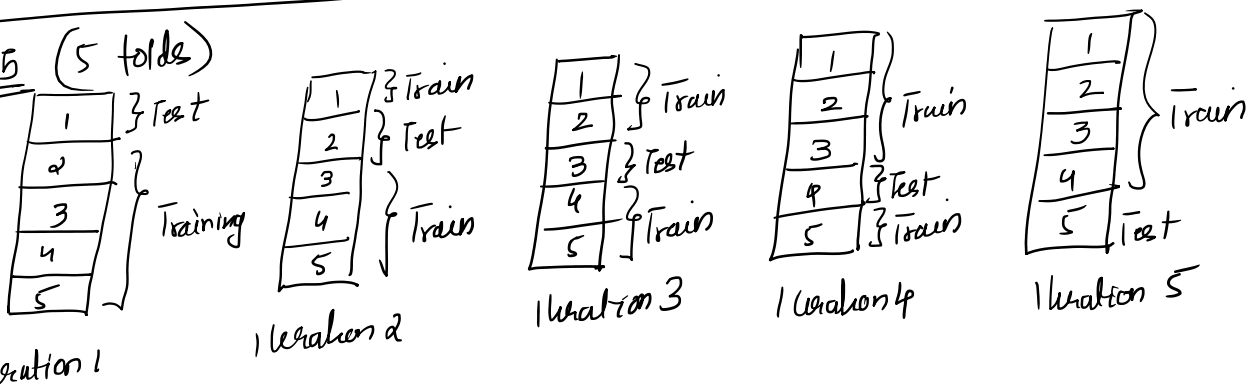
① The standard Validation Set Approach

② (LOOCV) Leave One Out Cross Validation.

③ K-fold Cross Validation.

K-fold Cross Validation ⇒

Ex K=5 (5 folds)

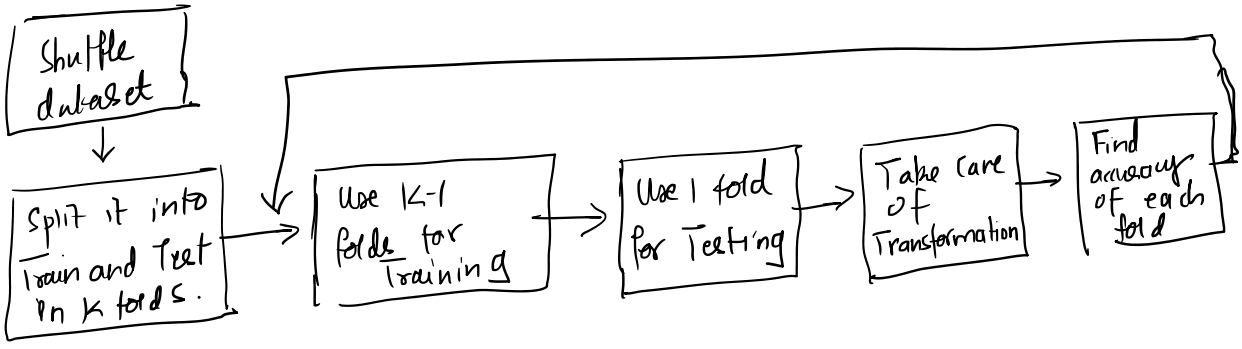


K fold ⇒ → K fold helps us to build the model in generalized form.

→ To achieve this K fold Cross Validation splits the dataset into Training Testing & Validation.

→ Here Test and Train data will support building the model.

→ Life cycle of K-fold Cross Validation.



* The No of iterations ideally is k time
* Finding mean of accuracy score of each iteration will give the consistency of the Trained model.

Rules

(1) $k \geq 2$

if $k == 2$ → just 2 iterations.

if $k = n >> 2$ ⇒ n-1 for Training
1 for Testing

(2) most commonly used value of $k = 10$

(3) If k is very large then the running time of process will increase.

(4) The value of k is inversely proportional to size of data i.e if dataset size is small then number of folds can increase.

Bagging ⇒ Random Forest

* Random Forest is Example of Bagging.

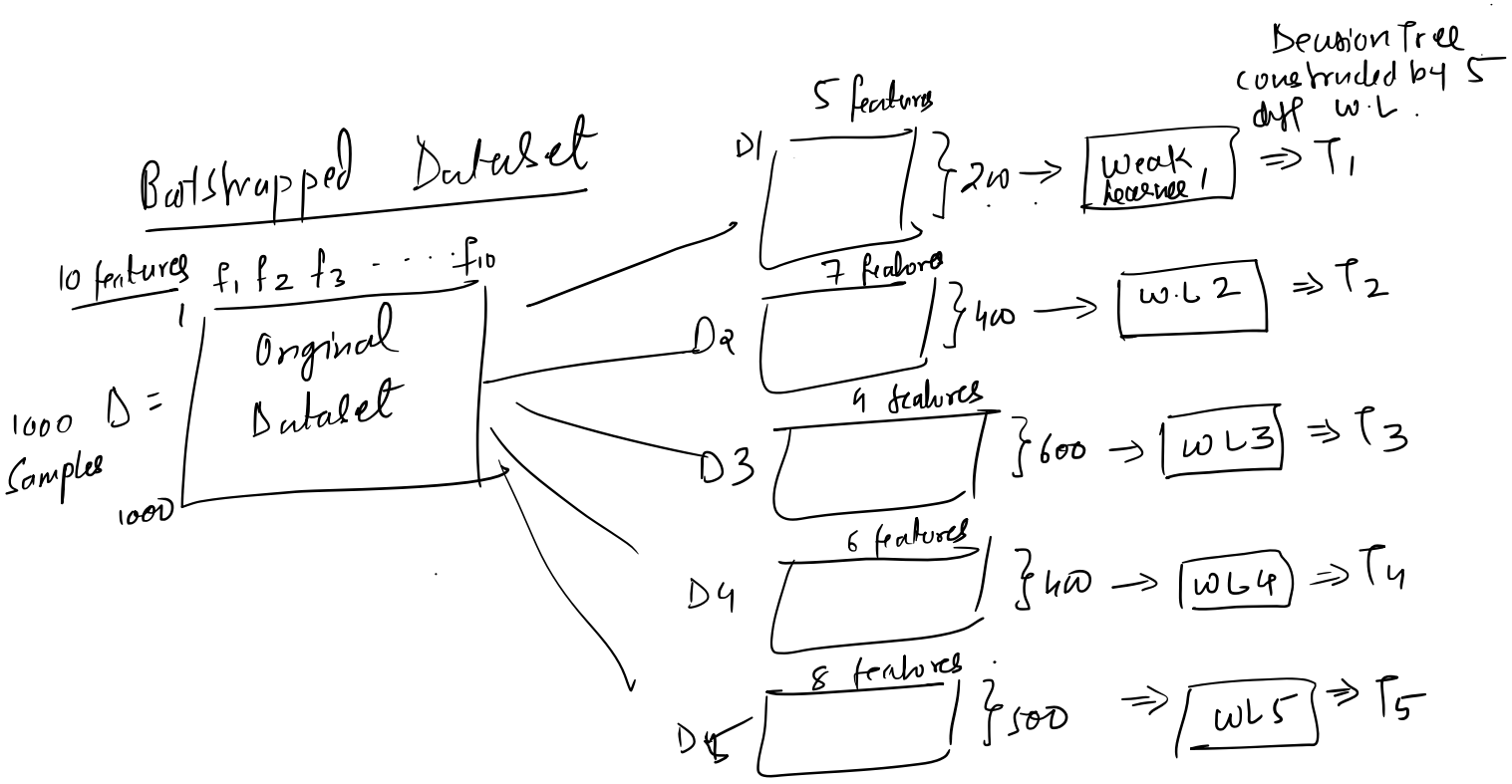
* You must know Decision Tree Construction [Gini Index]

* Decision Tree
  - Used for Regression ⇒ Regression Tree
  - Used for Classification / Categorization ⇒ Classification Tree.

* Decision Tree is easy to Build and Interpret in practical.

* The Decision Tree is not very accurate on new test sample

* Random Forest Combines Simplicity of Decision Tree with flexibility resulting into vast Improvement in accuracy.



We have 5 decision Tree Constructed one each by a weak learner.

Now Test Sample = S_{test}

Now S_{test} is Subjected to each of the 5 Decision Tree →

Suppose

$T_1(S_{test})$ = Yes
$T_2(S_{test})$ = NO
$T_3(S_{test})$ = Yes
$T_4(S_{test})$ = Yes
$T_5(S_{test})$ = NO

In 5 cases 3 cases are Yes
2 case are NO

Final Prediction Yes.

* Construct Random forest

Original Set

	Chest Pain	Good Blood Circulation	Blocked Arteries	Weight	Heart Disease
S1	NO	NO	NO	125	NO
S2	Yes	Yes	Yes	180	Yes
S3	Yes	Yes	NO	210	NO
S4	Yes	NO	Yes	167	Yes
	NO	Yes	Yes	150	?

Step1 => Create Bootstrapped DataSet

1. could be / not be of same size
2. Samples are randomly selected
3. Allowed to pick same sample more than once.

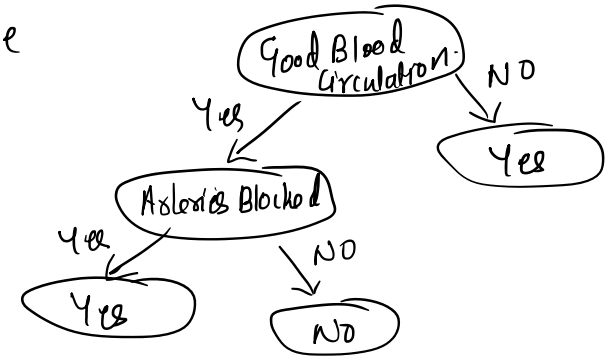
BootStrapped DataSet

Chest pain	Good Blood Circulation	Blocked Arteries	Weight	Heart Disease
Yes	Yes	Yes	150	Yes
NO	No	NO	125	NO
Yes	NO	Yes	167	Yes
Yes	NO	Yes	167	Yes

Step2 => Create a Decision Tree Using Bootstrapped dataset but Use random subset of Variables (features/columns)

Let us Consider we use only 2 features
(Good Blood Circulation and Blocked Arteries)

Let the Tree be (Gini Index)



Step 3 → Go to step1 and Repeat

* Ideally we repeat it for 100 times
* So we have $T_1, T_2 \ldots T_{100}$ (large no of Trees)
* Each time the Tree Constructed is a weak learner.
  [As all the features and samples are not considered while Tree Construction].
* Now we have Random Forest of 100 Trees and will be more effective than Individual Decision Tree.

6) How To Use the Random Forest (Here 100 Trees) →

Consider a Test Sample

Chest pain	Good Blood Circulation	Blocked Arteries	Weight	Heart Disease
Yes	NO	NO	168	?

Here we will Run it through each of 100 Trees and will note the Prediction

Let say of 100 Trees < 80 Trees Predicted Yes.
20 Trees Predicted NO

Answer = Yes

Answer = Yes

Q) To find Random Forest is Effective or Not ?

* There might be some Sample not considered by any of Bootstrapped Dataset.
* We will create a New Dataset with such samples this is known as "Out of Bag" Dataset
* Now we will chk how many samples from out of Bag Dataset are predicted correctly
* Numbers of Incorrectly Predicted Out of Bag Samples = Out of Bag Error.

Q) How to decide on Number of Columns to Use for Building Trees in Random Forest ?

(1) In our Case Dataset has 4 features

→ Create a Random Forest of Trees using 2 Features = F_1 (100 Trees)

→ Create a Random Forest of Trees using 3 features = F_2 (100 Trees)

(2) Now Chk the Accuracy of Out of Bag Data on each of the Random Forest

③ Use the one that gives More Accurate Answer.

Summarize

step 1. Create a Bootstrapped Data Set

step2 :- Create a Decision Tree using Subset of features

step 3 : Repeat step 1 and 2 approx 100 times.

step 4 : Use out of Bag Data Set to determine out of Bag Error

step 5 : If out of Bag Error is high then Repeat step 1 to step 4 until out of Bag Error is considerably low.

Note => To Compare the Performance for diff Random Forest Create Diff Random forest using different number of features

Use the one that gives Highest Accuracy.

ADABOOST (ADAPTIVE BOOST)

→ (Tree Stumping)

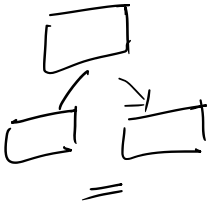
* Adaboost creates forest of Trees from scratch and then it uses it to make classification.

* [In Random Forest Everytime we make a Tree with different depth (depending on number of features considered).

* There is no predetermined maximum depth of Tree.]

* In ADABOOST, the forest of Trees are usually just node and Two leaves

* A Tree with one node and two leaves only is known as Tree Stump.



* In ADABOOST we have forest of stump instead of Trees.

* Stump alone is not great for classification as it takes only one parameter to make decision.

* Stumps are weak learners.

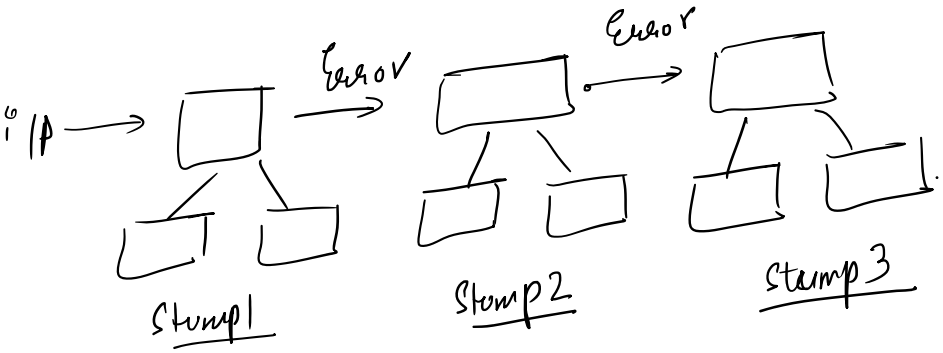
* ADABOOST combines many stumps.

* In Random forest each tree contribute equally in final decision.

more say in final

Final decision:

* In ADABOOST, some stump may get more say in final decision than others.
* In Random Forest the decision made by Trees are Independent to other, so the order of tree generation is not important
* In ADABOOST, we have forest of stump and hence order is Important
* Here the Errors of first stump is Corrected by next stump and so on.



* Here Error of one stump influences the other stump so the stumps are generated in Sequences.

Idea Behind ADABOOST. (Boosting Technique → reduce Bias)

1. Adaboost combines lot of weak learners to make classification
2. The weak learners are almost always stump.
3. Some stumps will have more say in classification.

a the ...

3. Some stump will have more say in classification than other stump.

4. Each stump is made by taking previous stump mistakes into account

6)

Working of ADABOOST with Example.

Consider

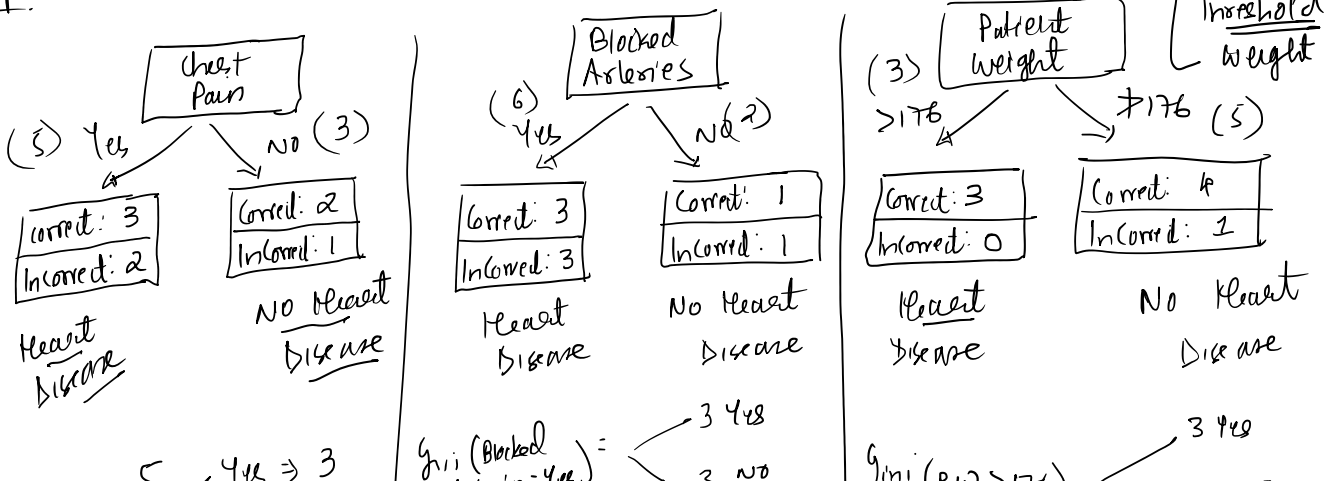
Chest Pain	Blocked Arteries	Patient weight	Heart Disease
Yes	Yes	205	Yes
NO	Yes	180	Yes
Yes	NO	210	Yes
Yes	Yes	167	Yes
NO	Yes	156	NO
NO	Yes	125	NO
Yes	NO	168	NO
Yes	Yes	172	NO

Step 1: Assign weight to every Sample (Initial) $= \frac{1}{\text{Total No of Sample}} = \frac{1}{8}$

*(Initially Equal Weights are assigned to each Sample)

Chest Pain	Blocked Arteries	Patient weight	Heart Disease	Sample weight
Yes	Yes	205	Yes	1/8
NO	Yes	180	Yes	1/8
Yes	NO	210	Yes	1/8
Yes	Yes	167	Yes	1/8
NO	Yes	156	NO	1/8
NO	Yes	125	NO	1/8
Yes	NO	168	NO	1/8
Yes	Yes	172	NO	1/8

Step 2: Create Stump on Each Feature



Div:

Gini (Chest Pain = Yes) = 5 → Yes ⇒ 3, No ⇒ 2 (Heart Disease)

$= 1 - \left(\left(\frac{3}{5}\right)^2 + \left(\frac{2}{5}\right)^2\right)$

$= 0.48$

Gini (Chest Pain = No) = 3 → Yes = 2, No = 1

$= 1 - \left(\left(\frac{2}{3}\right)^2 + \left(\frac{1}{3}\right)^2\right)$

$= 0.444$

Weighted Gini (Chest Pain)

$= 0.48 * \frac{5}{8} + 0.44 * \frac{3}{8}$

$= 0.4665 \approx 0.47$

Gini (Blocked Arteries = Yes) = 6 → 3 Yes, 3 No

$= 1 - \left(\left(\frac{3}{6}\right)^2 + \left(\frac{3}{6}\right)^2\right)$

$= 0.5$

Gini (Blocked arteries = No) = 2 → 1 Yes, 1 No

$= 1 - \left(\left(\frac{1}{2}\right)^2 + \left(\frac{1}{2}\right)^2\right)$

$= 0.5$

Gini (Blocked Arteries) = 0.5

Gini (P.W > 176) = 3 → 3 Yes, 0 No

$= 1 - \left(\left(\frac{3}{3}\right)^2\right)$

$= 0$

Gini (P.W ≯ 176) = 5 → 4 Yes, 1 No

$= 1 - \left(\left(\frac{4}{5}\right)^2 + \left(\frac{1}{5}\right)^2\right)$

$= 0.32$

Gini (weight) = 0.2

* The smallest Gini Index is 0.2 for weight > 176 feature

So first Stump will be weight > 176

Step 3 =>

To calculate Amount of Say of weight > 176 [feature].

$$\text{Amount of say} = \frac{1}{2} \ln\left(\frac{1 - \text{Total Error}}{\text{Total Error}}\right)$$

Total Error: for a stump the Sum of weights associated with Incorrectly classified sample.

Here only one sample is Incorrectly classified.

Yes	Yes	167	Yes	1/8

∴ Total Error = 1/8

* Amt of say = $\frac{1}{2} \ln\left(\frac{1-(1/8)}{(1/8)}\right) = \frac{0.97}{(\text{High})}$

[Note : If Amt of say is less then the stump is not doing good job]

* Now we have → First stump (weight > 176)
→ Amt of say.

Now Initially Every Sample had Equal weight →

Now Before Second stump we must modify weights of Samples

* Weights of Correctly classified Samples ⇒ must be decrease
" " Incorrectly classified Samples ⇒ must be increase
[So that next stump will focus more on Incorrectly identified Samples].

*

New Sample Weight (Correctly classified Sample) = Sample weight (old) * $e^{-\text{amount of say}}$

New Sample Weight (Incorrectly classified Sample) = Sample weight (old) * $e^{+\text{amount of say}}$

∴ for Incorrectly Classified Sample = $\frac{1}{8} * e^{+0.97} = 0.33$

For Correctly Classified Sample = $\frac{1}{8} * e^{-0.97} = 0.05$

	Chest Pain ✓	Blocked Arteries ✓	Patient weight ✓	Heart Disease ✓	Sample weight	New Sample weight ↓	Normalize Sample weight ↓
	Yes.	Yes	205	Yes.	1/8	0.05	0.05/0.68 = 0.07
✓	NO.	Yes	180	Yes.	1/8	0.05	0.07
	Yes.	No	210	Yes.	1/8	0.05	0.07
*	Yes.	Yes	(167)	(Yes)	1/8	0.33	0.33/0.68 = 0.49

Yes.	No	210	Yes.	1/8	0.05	
Yes.	Yes	167	Yes	1/8	0.33	0.33/0.68 = 0.49
No	Yes	156	No	1/8	0.05	0.07
No	Yes	125	No	1/8	0.05	0.07
Yes.	No	168	No	1/8	0.05	0.07
Yes	Yes	172	No	1/8	0.05	0.07

*Sum = 0.68

* Now we have New Normalized Sample weights.

* The Incorrectly classified Sample has Higher New Sample weight
  → which means Stump 2 will focus more on it.

* The Correctly classified Sample has Comparatively lower Sample weight
  → which means stump2 will focus less on it.

* Similarly we can have Many such Stumps

* This is how Adaboost Creates and Uses Stumps.

* Let us chk how forest of stumps made by Adaboost does Classification →

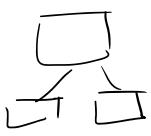
(1) Let us consider a test sample

(2) we will pass the test Sample through each stump

(3) Let there be few stumps that Classifies Sample as Yes (Has Heart Disease) and few stumps Classifies Sample as No (Do not have heart disease).

(4)

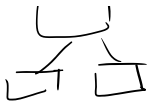
Has Heart Disease



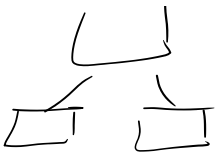
⇒ Amt of Say = 0.47

Has Not Heart Disease

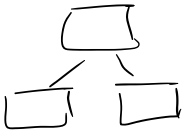
= Amt of Say = 0.41



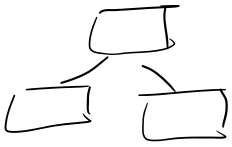
Say = 0.97



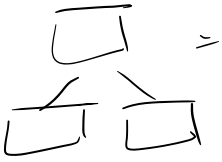
= Amt of Say = 0.82



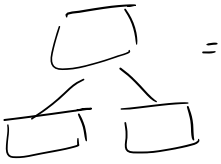
= Amt of Say
= 0.32



Total Amt of Say (NO) = 1.23.



= Amt of Say
= 0.78



= Amt of Say
= 0.63.

Total amt of Say (Yes) = 2.7

Total Amt of Say for Stumps that classifies as Heart Disease (Yes) is greater

∴ The Test Sample is classified as Yes (Has Heart Disease).

Summary of Adaboost

1. Assign Sample Weight (Initial Equal)
2. Create Stump for each feature.
3. Use Gini Index to Identify first stump.
4. For first stump.
   - (1) Calculate Total Error
   - (2) Amt of Say.

④ For first stump.
   ① Calculate Total Error
   ② Amt of say.

⑤ Update the sample weights for each sample

For Incorrectly classified = updated weight = old weight $* e^{+\text{Amt of say}}$.

For Correctly classified = updated weight = old weight $* e^{-\text{Amt of say}}$.

⑥ Now Specify updated weight for each sample & Normalize the weight to generate New Sample weight.

⑦ Identify New stump based on New Sample weight & Repeat step 3 to 7

⑧ For Classification (Testing)

8.1 ⇒ Run the test sample through all the stumps.

8.2 ⇒ Calculate Amt of say for sample classifying Yes & No

8.3 ⇒ Classify the test sample based on largest sum of Amount of say