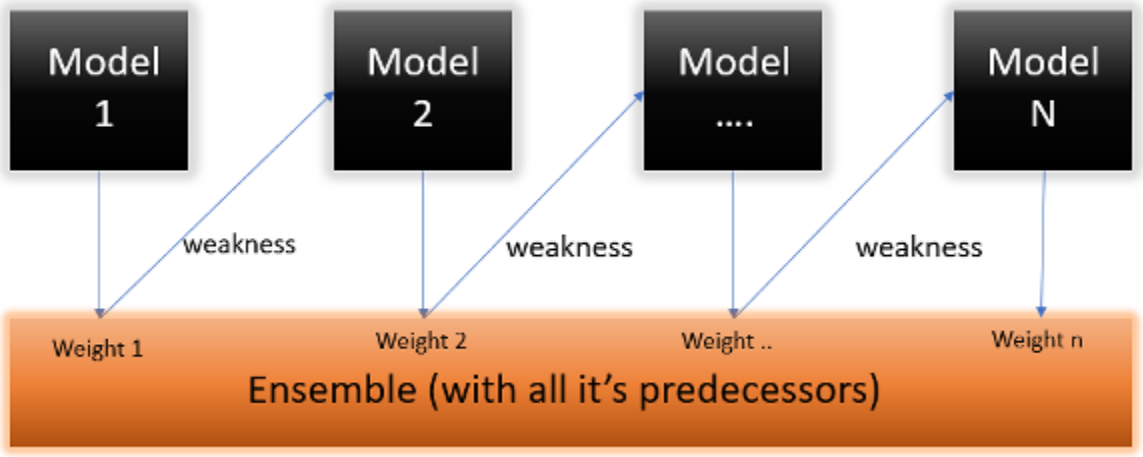
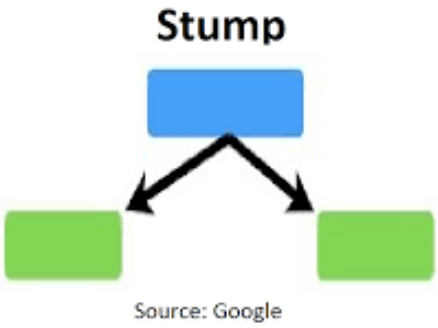
Adaboost

Boosting technique used as an Ensemble Method in Machine Learning. It is called Adaptive Boosting as the weights are re-assigned to each instance, with higher weights assigned to incorrectly classified instances.



Stumps:

Decision Stumps are like trees in a Random Forest, but not "fully grown." They have one node and two leaves.



Source: Google

Steps

Step 1: Assigning Equal Weights to all data points.

$$probabuliy\ of\ getting\ that\ particlar\ observation = Weight = \frac{1}{N}$$

Where N is the total number of data points

Here since we have 5 data points, the sample weights assigned will be 1/5.

Row no	gender	age	income	illness	Sample weights
1	Male	41	40000	Yes	1/5
2	Male	54	30000	No	1/5
3	Female	42	25000	No	1/5
4	Female	40	60000	Yes	1/5
5	Male	46	50000	Yes	1/5

Step 2: Classify the Samples

We'll create a decision stump for each of the features and then calculate the ***Gini Index or entropy*** of each tree. The tree with the lowest Gini Index or high entropy will be our first stump.

➔ Let's assume ***Gender*** has the lowest gini index

➔It will be our first stump.

Step 3: Calculate the Influence

Now calculate the **"performance"** or **"Importance"** or for this classifier in classifying the data points using this formula (performance of the stump (α))

$$\mathbf{performance}(\alpha) = \frac{1}{2} log\_e(\frac{1 - total\ error}{total\ error})$$

The total error:

The summation of all the sample weights of misclassified data points.

Let's assume there is 1 wrong output, so our total error will be 1/5.

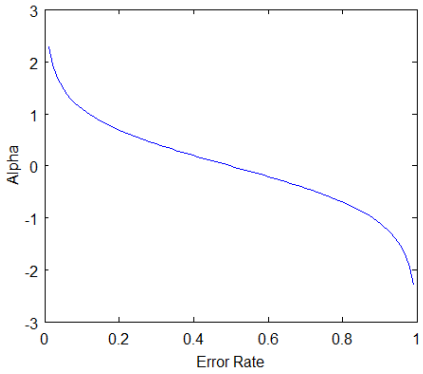
$$\textbf{performance of the stump}(\alpha) = \frac{1}{2} log_e(\frac{1 - tatal\ error}{total\ error})$$

$$\alpha = \frac{1}{2} log_e(\frac{1 - 1/5}{1/5})$$

$$\alpha = 0.69$$

Note: Total error will always be between 0 and 1.

0 Indicates perfect stump, and 1 indicates horrible stump.



- ➔ If there is no misclassification, then we have no error (Total Error = 0), so the "alpha" will be a large number.
- ➔ If the classifier predicts half right and half wrong, then the Total Error = 0.5, and the importance (alpha) of the classifier will be 0.
- ➔ If all the samples have been misclassification, then the error will be very high (approx. to 1), and hence our alpha value will be a negative integer.

Step 4: updating the weights

New sample weights=old weights *$e^{\pm\alpha}$

- ➔ Alpha will be ***negative*** when the sample is **correctly classified**.
- ➔ Alpha will be ***positive*** when the sample is **miss-classified.**

In our data four correctly classified samples and 1 wrong classified

Assume 4th observation are incorrectly classified

Sample weight of each point = $\frac{1}{5}$

Performance of the stump of ***Gender*** (α) = *0.69*

New sample weights=old weights * $e^{\pm\alpha}$

Row no	*gender*	*age*	*income*	*illness*	*Sample weights*	*New weights*
1	*Male*	*41*	*40000*	*Yes*	*1/5*	$\frac{1}{5}*e^{-0.69}$=0.1004
2	*Male*	*54*	*30000*	*No*	*1/5*	$\frac{1}{5}*e^{-0.69}$=0.1004
3	*Female*	*42*	*25000*	*No*	*1/5*	$\frac{1}{5}*e^{-0.69}$=0.1004
4	*Female*	*40*	*60000*	*Yes*	*1/5*	$\frac{1}{5}*e^{+0.69}$=0.3988
5	*Male*	*46*	*50000*	*Yes*	*1/5*	$\frac{1}{5}*e^{-0.69}$=0.1004

Normalize the weights

We know that the total sum of the sample weights must be equal to 1, but here if we sum up all the new sample weights, we will get 0.8004. To bring this sum equal to 1, we will normalize these weights by dividing all the weights by the total sum of updated weights

Row no	*gender*	*age*	*income*	*illness*	*Sample weights*	*New weights*	*Normalized weights*
1	*Male*	*41*	*40000*	*Yes*	*1/5*	*0.1004*	$\frac{0.1004}{0.8004}$=0.1254
2	*Male*	*54*	*30000*	*No*	*1/5*	*0.1004*	$\frac{0.1004}{0.8004}$=0.1254
3	*Female*	*42*	*25000*	*No*	*1/5*	*0.1004*	$\frac{0.1004}{0.8004}$=0.1254
4	*Female*	*40*	*60000*	*Yes*	*1/5*	*0.3988*	$\frac{0.3988}{0.8004}$=0.4982
5	*Male*	*46*	*50000*	*Yes*	*1/5*	*0.1004*	$\frac{0.1004}{0.8004}$=0.1254

Step-5: divide our data points into buckets.

Based on the "normalized weights," divide our data points into buckets.

no	*gender*	*age*	*income*	*illness*	*Normalized weights*	*buckets*
1	*Male*	*41*	*40000*	*Yes*	*0.1254*	*0.0000 to 0.1254*
2	*Male*	*54*	*30000*	*No*	*0.1254*	*0.1254 to 0.2508*
3	*Female*	*42*	*25000*	*No*	*0.1254*	*0.2508 to 0.3762*
4	*Female*	*40*	*60000*	*Yes*	*0.4982*	*0.3762 to 0.5016*
5	*Female*	*40*	*60000*	*Yes*	*0.4982*	*0.5016 to 0.6270*
6	*Female*	*40*	*60000*	*Yes*	*0.4982*	*0.6270 to 0.7524*
7	*Female*	*40*	*60000*	*Yes*	*0.4982*	*0.7524 to 0.8778*
8	*Male*	*46*	*50000*	*Yes*	*0.1254*	*0.8778 to 1.0000*

Modified buckets

no	*gender*	*age*	*income*	*illness*	*Normalized weights*	*buckets*
1	*Male*	*41*	*40000*	*Yes*	*0.1254*	*0.0000 to 0.1254*
2	*Male*	*54*	*30000*	*No*	*0.1254*	*0.1254 to 0.2508*
3	*Female*	*42*	*25000*	*No*	*0.1254*	*0.2508 to 0.3762*
4	*Female*	*40*	*60000*	*Yes*	*0.4982*	*0.3762 to 0.8778*
5	*Male*	*46*	*50000*	*Yes*	*0.1254*	*0.8778 to 1.0000*

Step-6: New Dataset

- Selects "N" random numbers from 0-1
- Since incorrectly classified records have higher sample weights, the probability of selecting those records is very high.
- Suppose the 5 random numbers are taken
  (0.38, 0.26, 0.98, 0.40, 0.55)
- Now we will see where these random numbers fall in the bucket, and according to it, we'll make our new dataset shown below.

no	gender	age	income	illness	Normalized weights
1	Female	40	60000	Yes	0.4982
2	Female	42	25000	No	0.1254
3	Male	46	50000	Yes	0.1254
4	Female	40	60000	Yes	0.4982
5	Female	40	60000	Yes	0.4982

Step-7: Repeat Previous Steps

Now this act as our new dataset, and we need to repeat all the above steps i.e.

1. Assign equal weights to all the data points.
2. Find the stump that does the best job classifying the new collection of samples by finding their Gini Index and selecting the one with the lowest Gini index.
3. Calculate the "performance of stump (alpha)" and "Total error" to update the previous sample weights.
4. Normalize the new sample weights.

Iterate through these steps until and unless a low training error is achieved.

Suppose, with respect to our dataset, we have constructed 3 decision trees (DT1, DT2, and DT3) in a *sequential manner.* If we send our **test data** now, it will pass through all the decision trees, and finally, we will see which class has the majority, and based on that, we will do predictions for our test dataset.

Advantages

- AdaBoost is a flexible algorithm that can be applied to a variety of machine-learning problems, including classification and regression.
- It can handle datasets with missing values and outliers well.

Disadvantages

- AdaBoost is highly sensitive to noisy data and outliers.
- It is computationally expensive and may take a long time to train the model.
- AdaBoost may overfit the data if the number of iterations is too high.