Mohamed Noordeen Alaudeen

Senior Data Scientist – Logitech
Random Forest

Ensemble

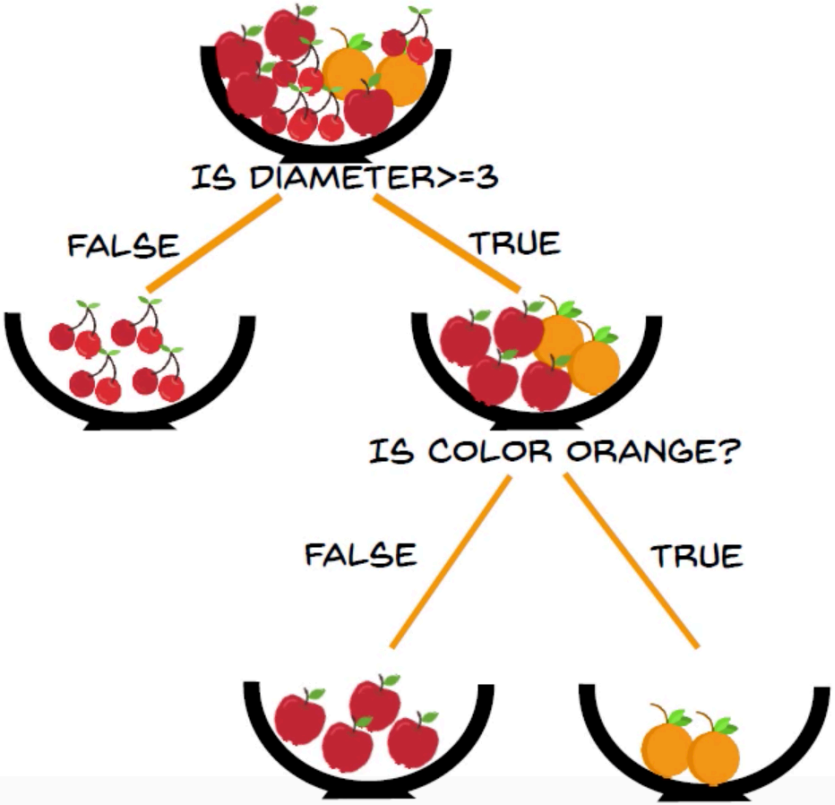
Ensemble

Machine learning paradigm which combine weak learners to become a strong learner

Decision Tree Summary

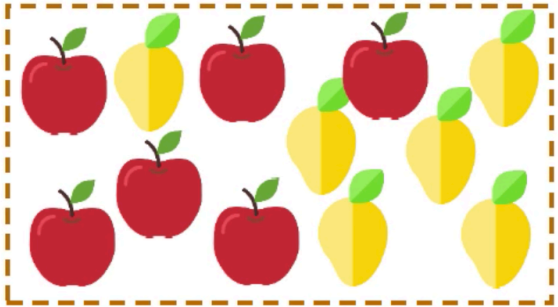
- Rule based system
- Set of rules
- Calculating the nodes
- Information gain and Gini index

Decision Tree is a tree shaped diagram used to determine a course of action. Each branch of the tree represents a possible decision, occurrence or reaction



Entropy

Entropy is the measure of randomness or unpredictability in the dataset

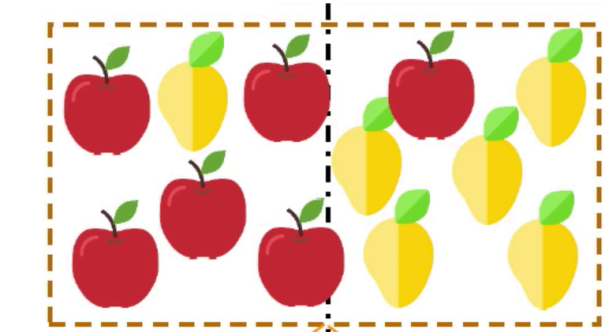


High entropy

E1

Entropy

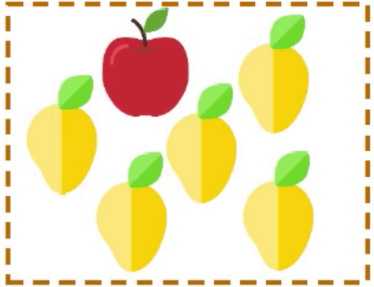
Entropy is the measure of randomness or unpredictability in the dataset



Initial Dataset

Decision Split

Set 1



Set 2

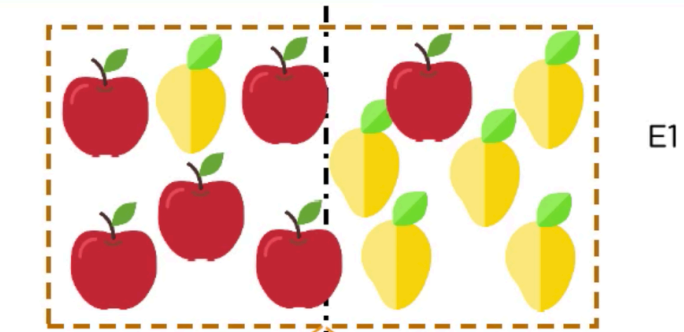
High entropy E1

After Splitting

Lower entropy E2

Information gain

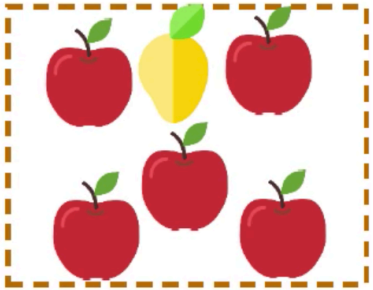
It is the measure of decrease in entropy after the dataset is split



E1

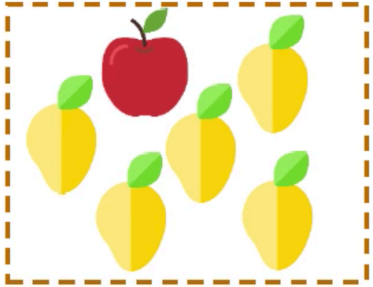
Initial Dataset

Decision Split



E2

Set 1



E2

Set 2

High entropy E1

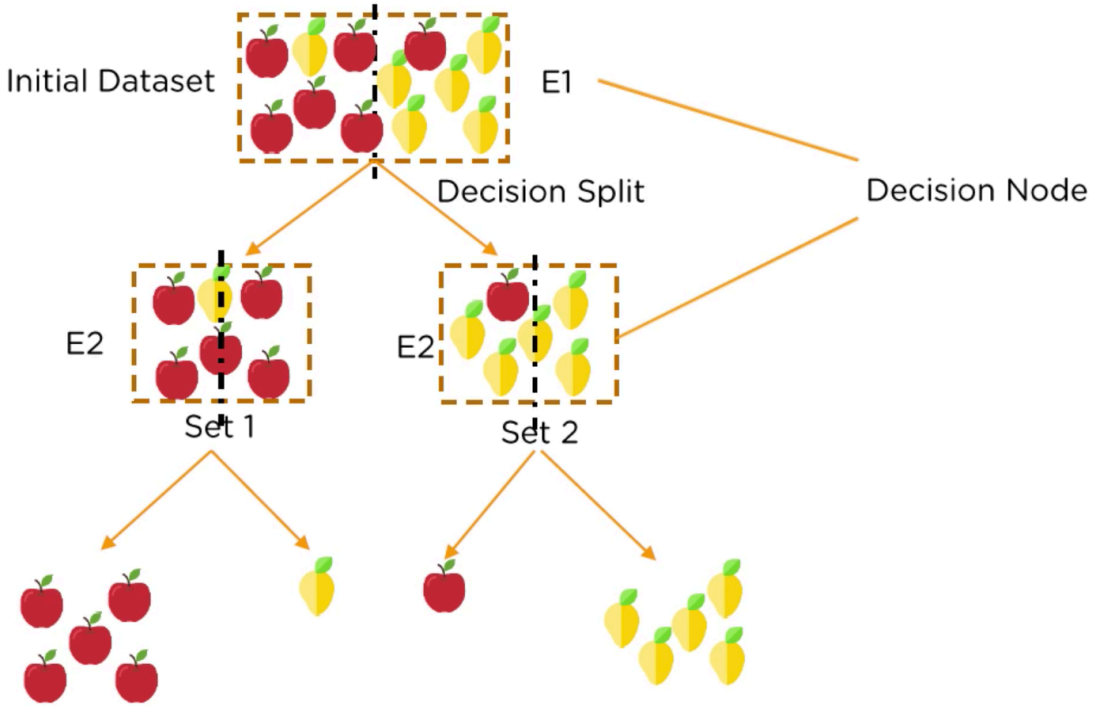
After splitting

Lower entropy E2

Information gain= E1-E2

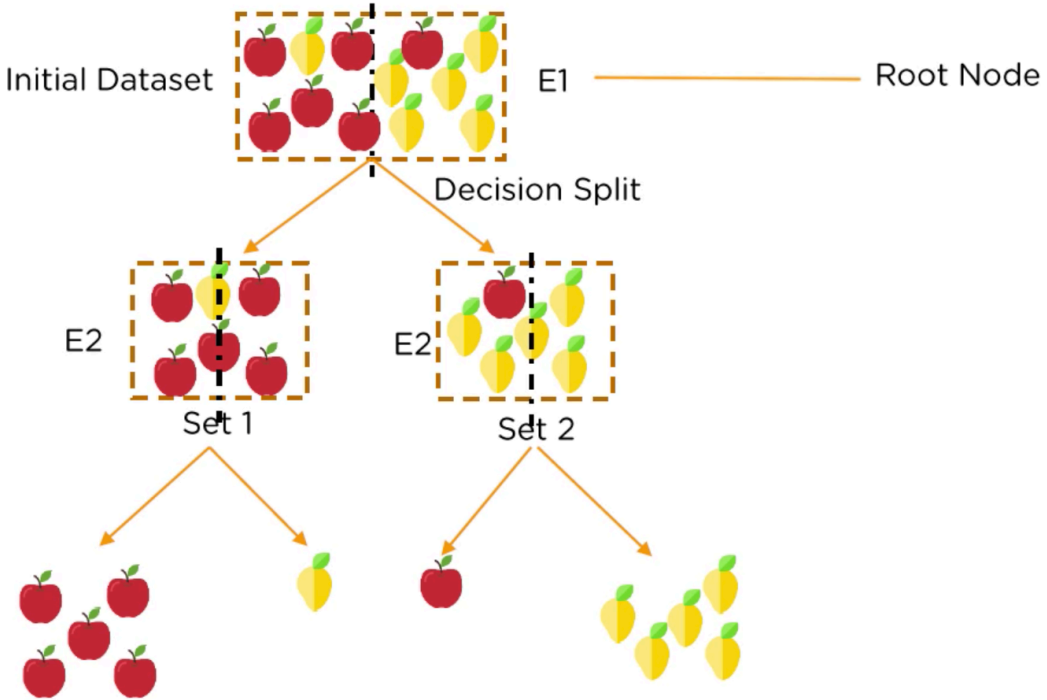
Decision Node

Decision node has two or more branches



Root Node

The top most Decision node is known as the Root node



PROBLEM STATEMENT

TO CLASSIFY THE DIFFERENT TYPES OF FRUITS IN THE BOWL BASED ON DIFFERENT FEATURES

THE DATASET(BOWL) IS LOOKING QUITE MESSY AND THE ENTROPY IS HIGH IN THIS CASE

TRAINING DATASET

COLOR	DIAMETER	LABEL
RED	3	APPLE
YELLOW	3	LEMON
PURPLE	1	GRAPES
RED	3	APPLE
YELLOW	3	LEMON
PURPLE	1	GRAPES

HOW TO SPLIT THE DATA

WE HAVE TO FRAME THE CONDITIONS THAT SPLIT THE DATA IN SUCH A WAY THAT THE INFORMATION GAIN IS THE HIGHEST

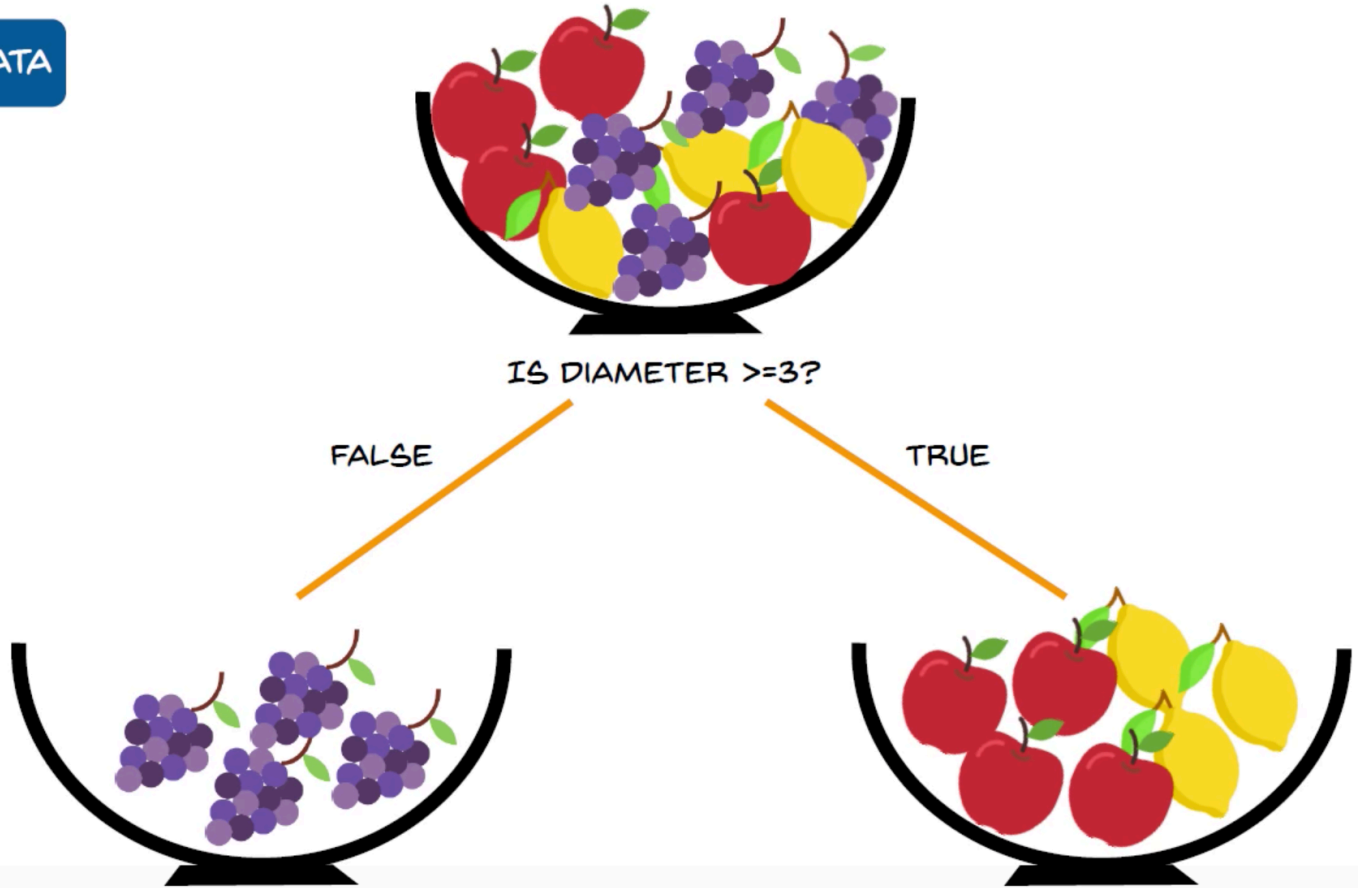
HOW TO SPLIT THE DATA

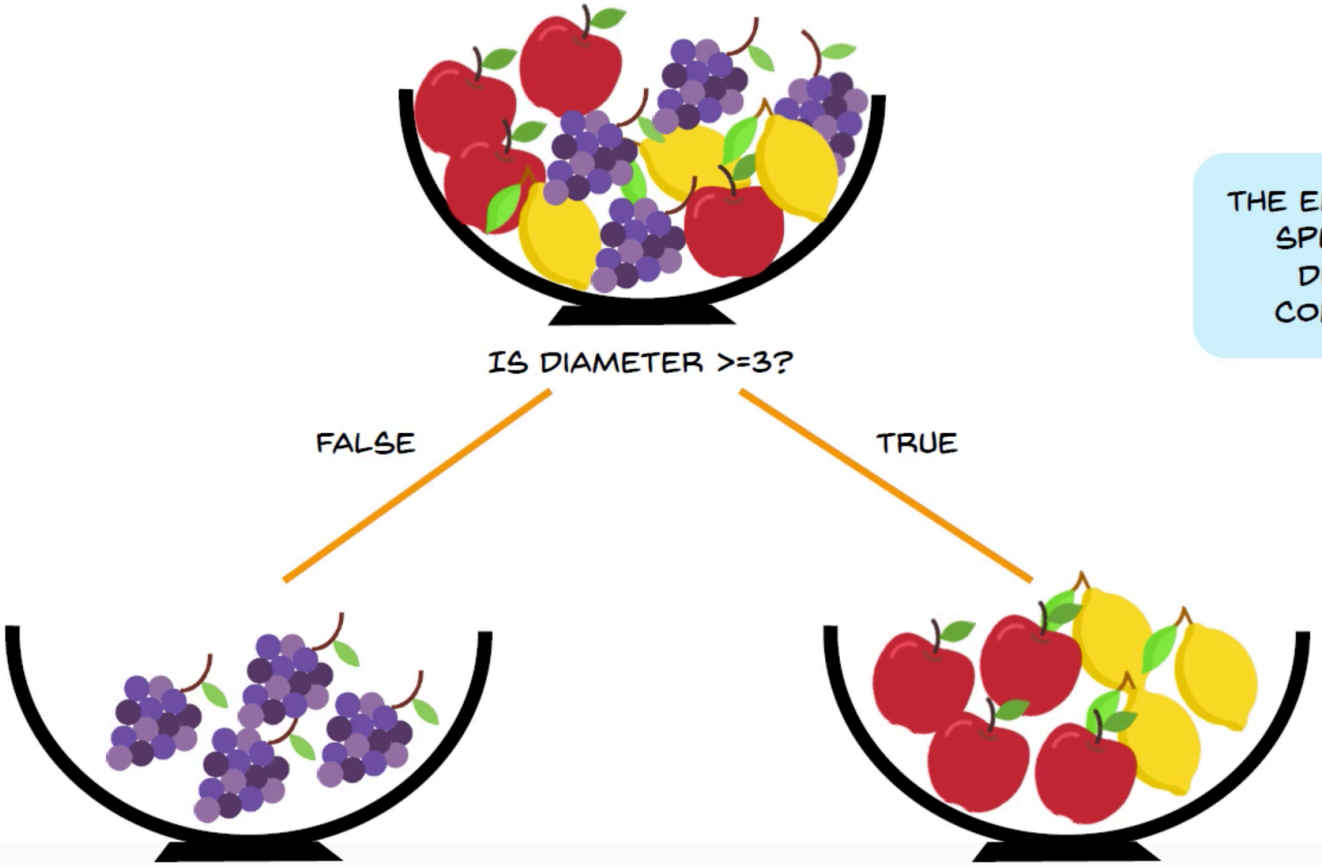
WE HAVE TO FRAME THE CONDITIONS THAT SPLIT THE DATA IN SUCH A WAY THAT THE INFORMATION GAIN IS THE HIGHEST

NOTE

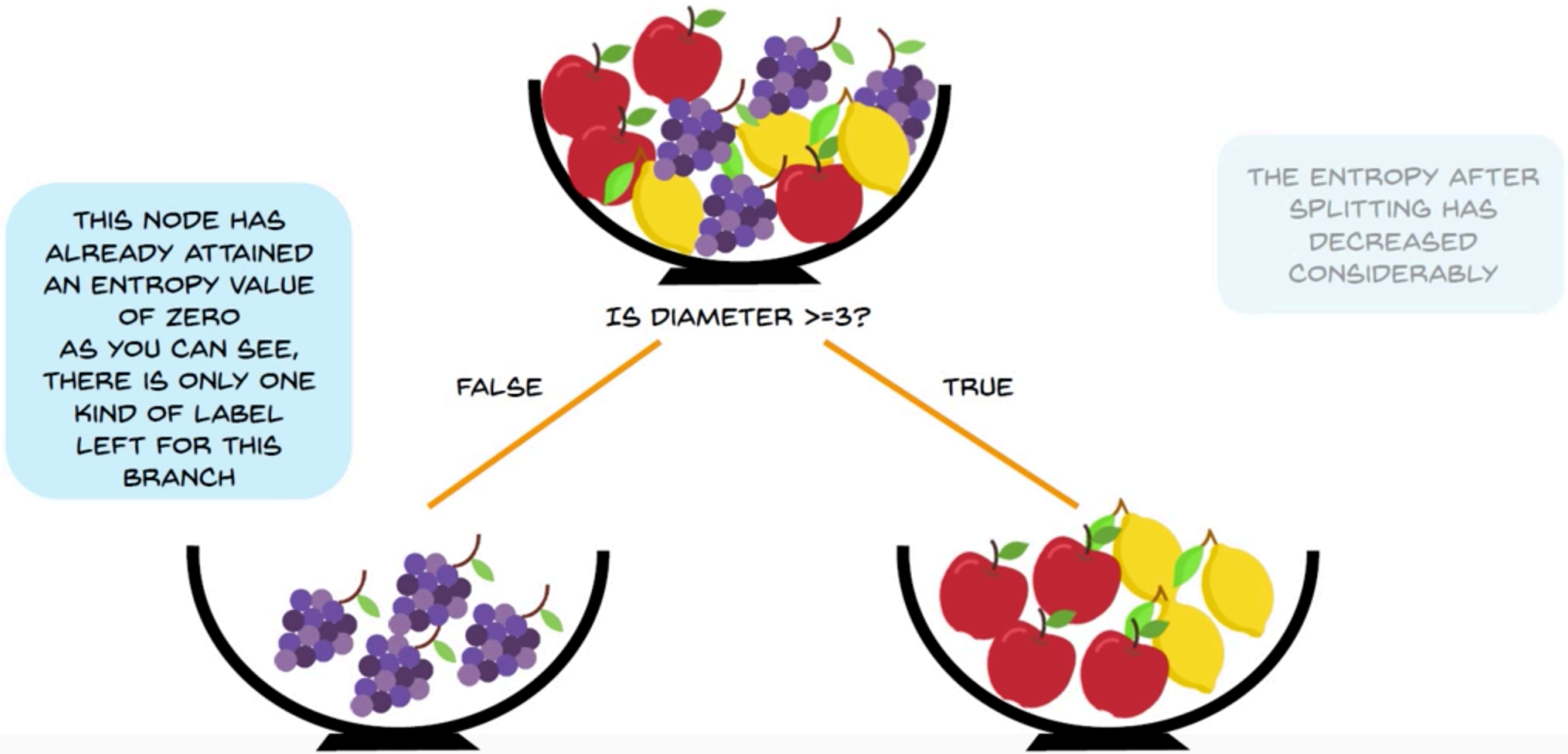
GAIN IS THE MEASURE OF DECREASE IN ENTROPY AFTER SPLITTING

WE SPLIT THE DATA





THE ENTROPY AFTER SPLITTING HAS DECREASED CONSIDERABLY

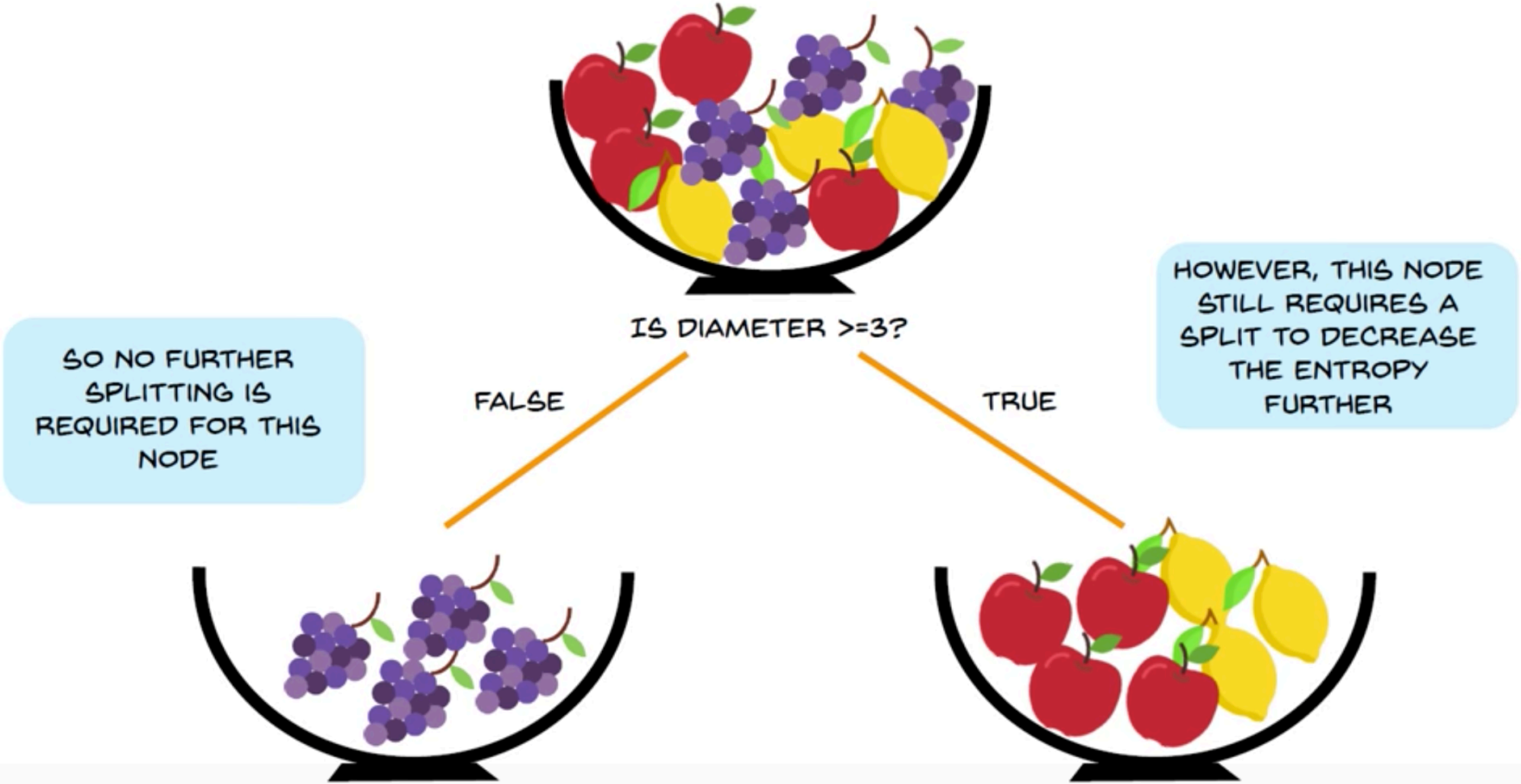

THIS NODE HAS ALREADY ATTAINED AN ENTROPY VALUE OF ZERO
AS YOU CAN SEE, THERE IS ONLY ONE KIND OF LABEL LEFT FOR THIS BRANCH
IS DIAMETER >=3?
FALSE
TRUE
THE ENTROPY AFTER SPLITTING HAS DECREASED CONSIDERABLY

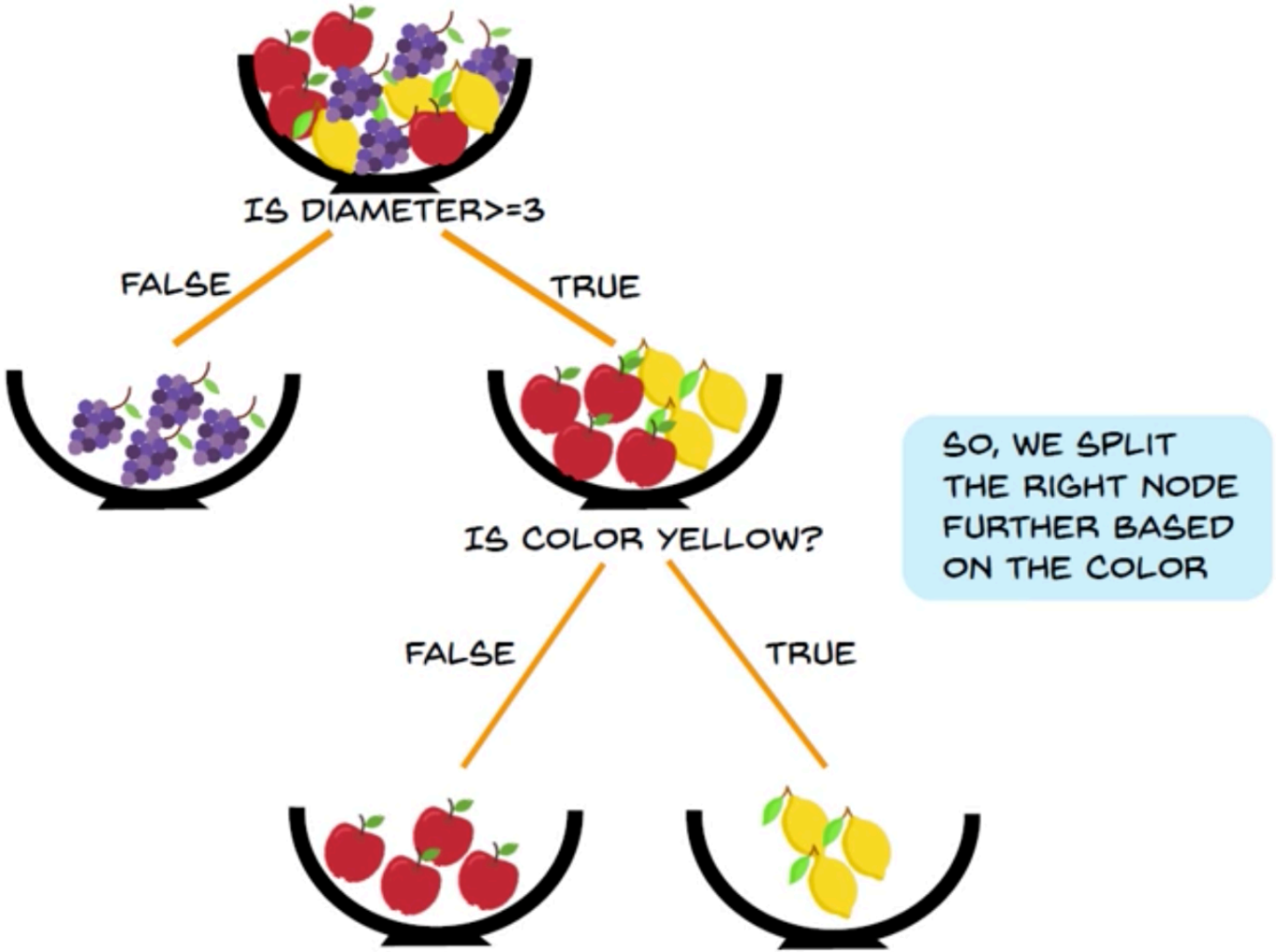
IS DIAMETER >=3?

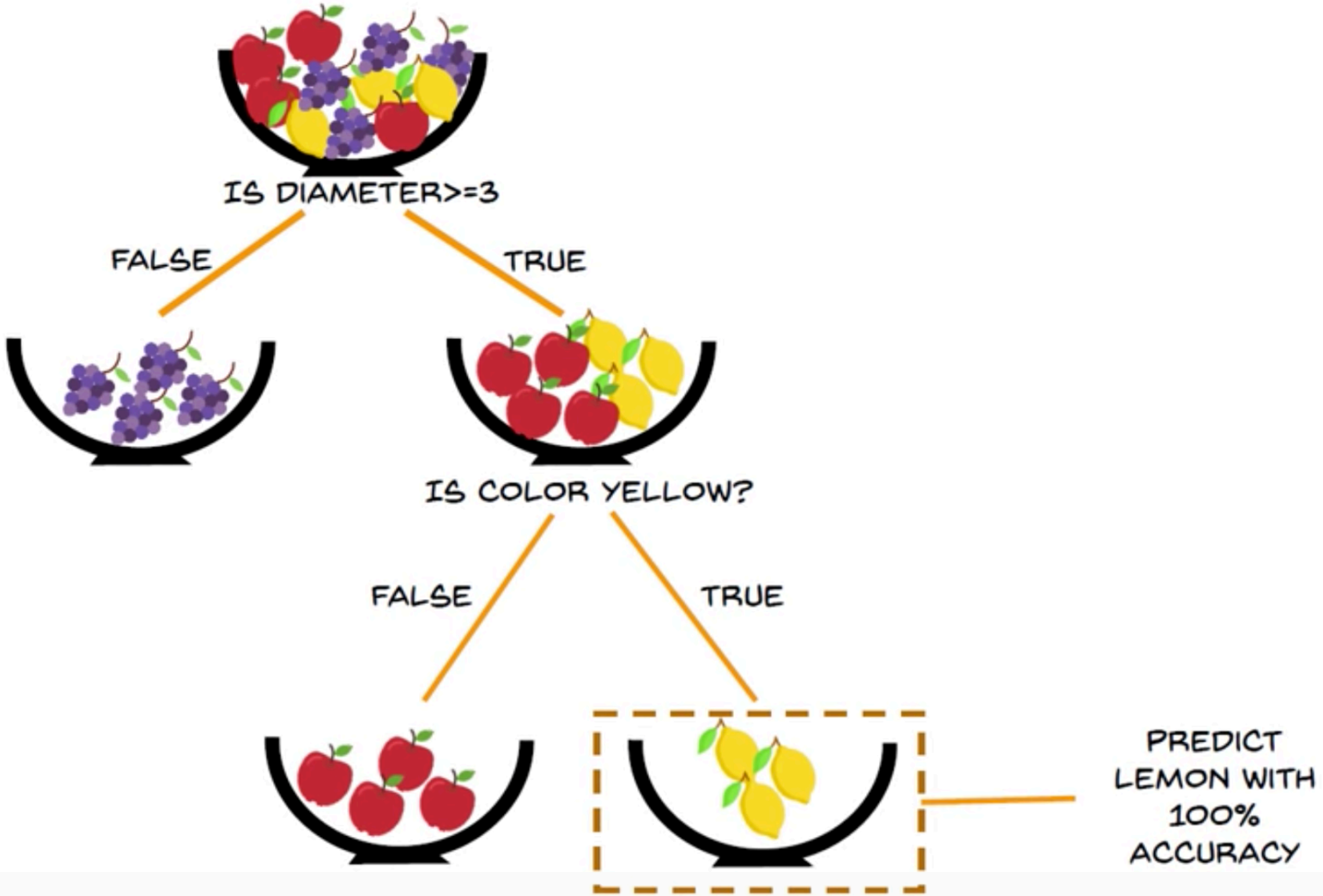
FALSE

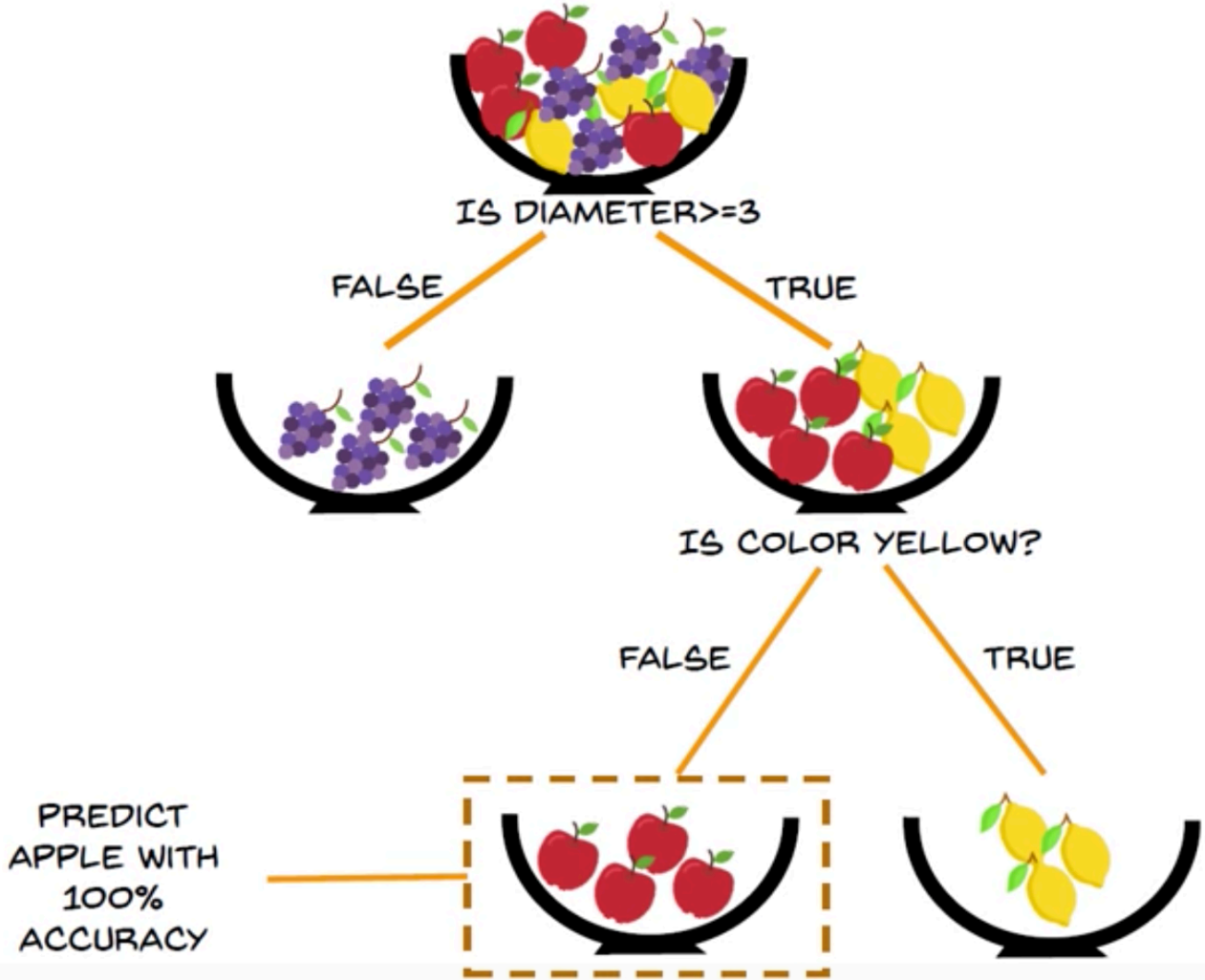
TRUE

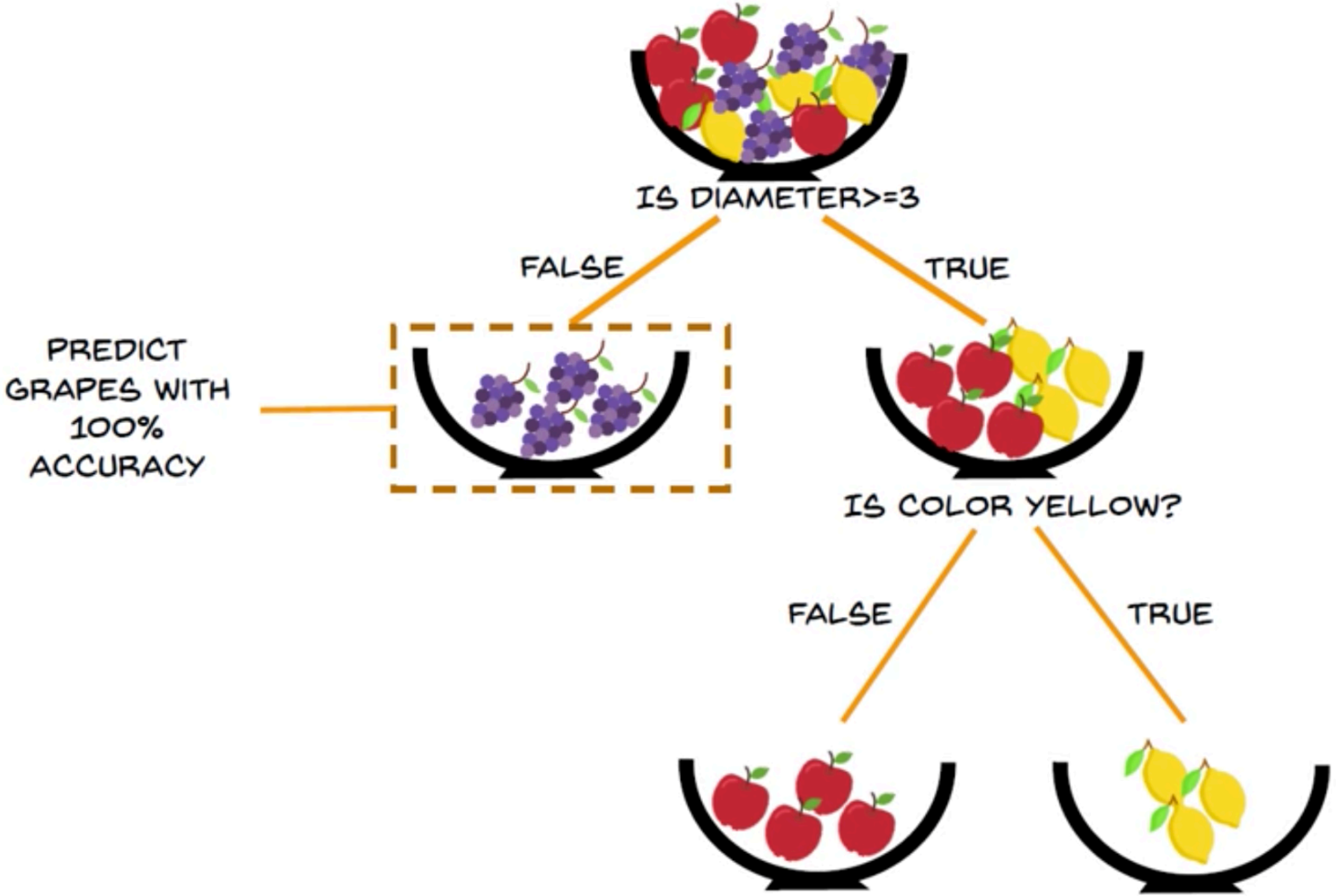
SO NO FURTHER SPLITTING IS REQUIRED FOR THIS NODE


IS DIAMETER >=3?
FALSE
TRUE
SO NO FURTHER SPLITTING IS REQUIRED FOR THIS NODE
HOWEVER, THIS NODE STILL REQUIRES A SPLIT TO DECREASE THE ENTROPY FURTHER


IS DIAMETER>=3
FALSE
TRUE
IS COLOR YELLOW?
FALSE
TRUE
SO, WE SPLIT THE RIGHT NODE FURTHER BASED ON THE COLOR


IS DIAMETER>=3
FALSE
TRUE
IS COLOR YELLOW?
FALSE
TRUE
PREDICT LEMON WITH 100% ACCURACY


IS DIAMETER>=3
FALSE
TRUE
IS COLOR YELLOW?
FALSE
TRUE
PREDICT APPLE WITH 100% ACCURACY


IS DIAMETER>=3
FALSE
TRUE
PREDICT GRAPES WITH 100% ACCURACY
IS COLOR YELLOW?
FALSE
TRUE

Problem with Decision Trees?

- Greedy Algorithm
- Overfitting
- Low prediction accuracy
- Calculations can become complex when there are many class labels.

Bagging

Bootstrap **agg**regat**ing**, also called **bagging**, is a machine learning ensemble meta-algorithm designed to improve the **stability** and **accuracy** of machine learning algorithms used in statistical classification and regression.

It also reduces variance and helps to avoid overfitting.

Why Random Forest?

No overfitting

Use of multiple trees reduce the risk of overfitting

Training time is less

High accuracy

Runs efficiently on large database

For large data, it produces highly accurate predictions

Estimates missing data

Random Forest can maintain accuracy when a large proportion of data is missing

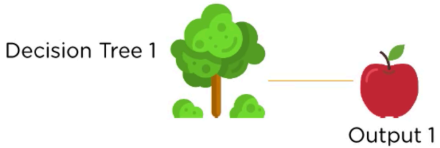
Real Life Example

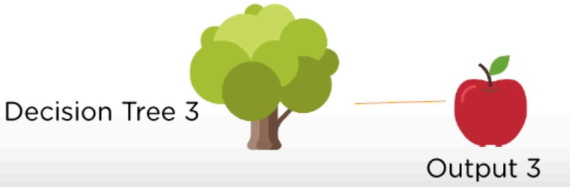
- You have decided to go for a tour
- But you need to choose which place to go

- You have set of attributes like
- Budget
- Type of Place (City, Village, Hill station)
- Domestic or International

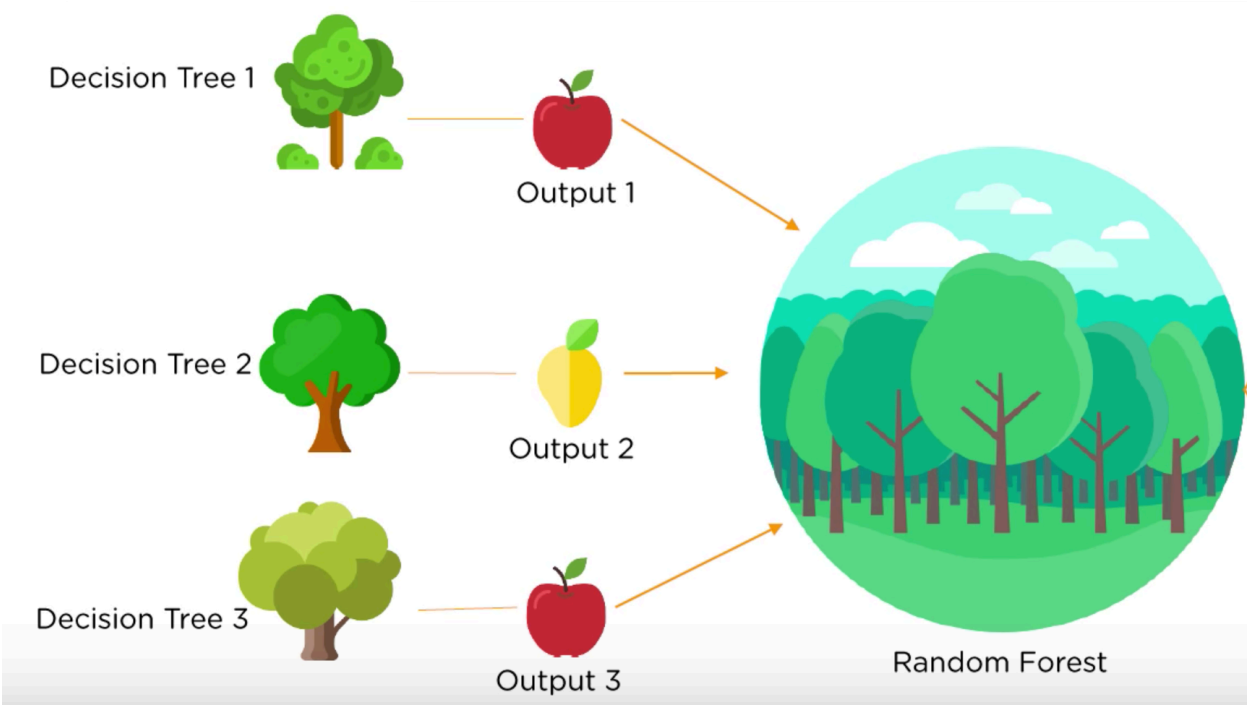
- Your option are (Ooty, Shimla, Switzerland, coorg, Newyork,etc)
- You go and ask many people who have been to the place with any two conditions
- Finally after getting the result you decide based on maximum votes

Random forest or Random Decision Forest is a method that operates by constructing multiple Decision Trees during training phase.
The Decision of the majority of the trees is chosen by the random forest as the final decision

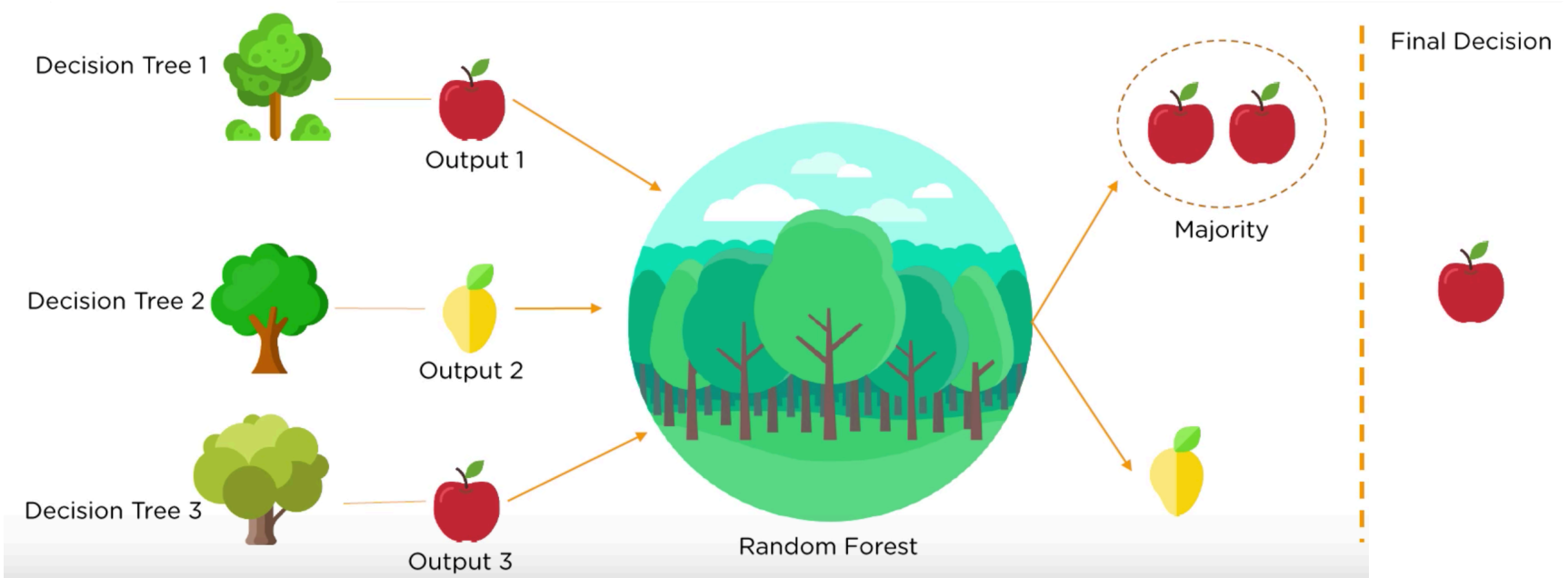




Random forest or Random Decision Forest is a method that operates by constructing multiple Decision Trees during training phase.
The Decision of the majority of the trees is chosen by the random forest as the final decision



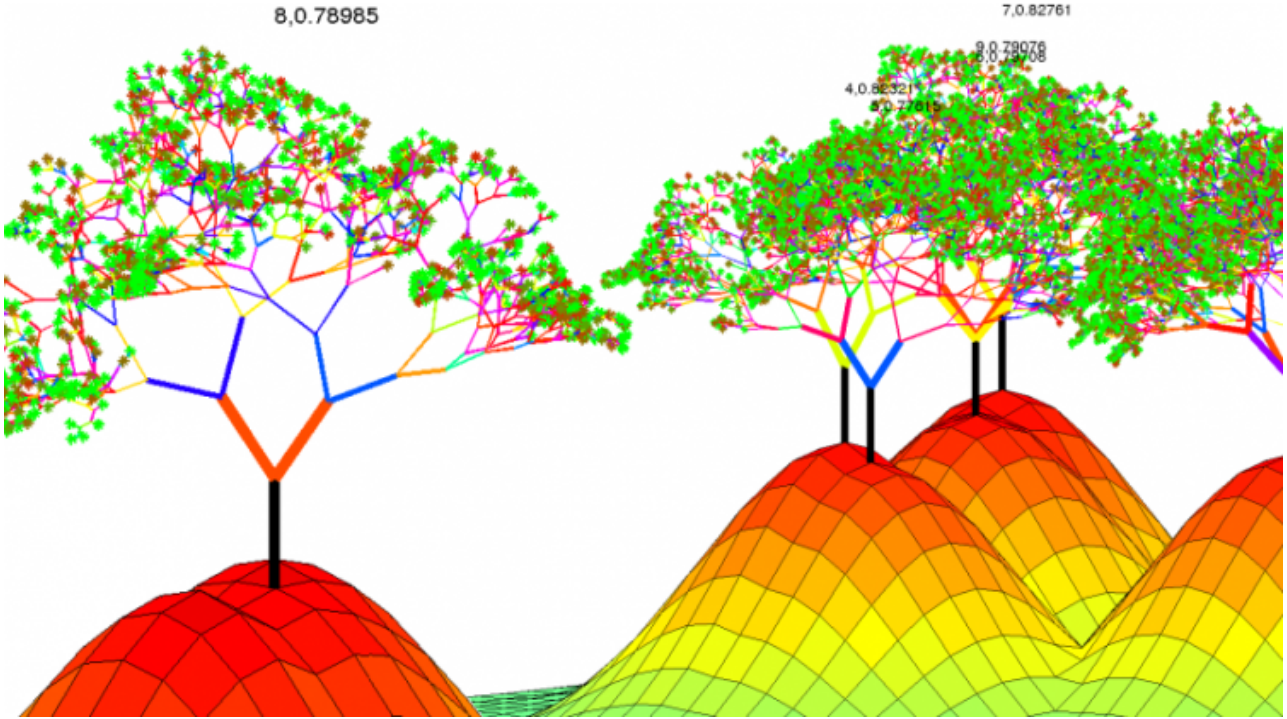
Random forest or Random Decision Forest is a method that operates by constructing multiple Decision Trees during training phase.
The Decision of the majority of the trees is chosen by the random forest as the final decision

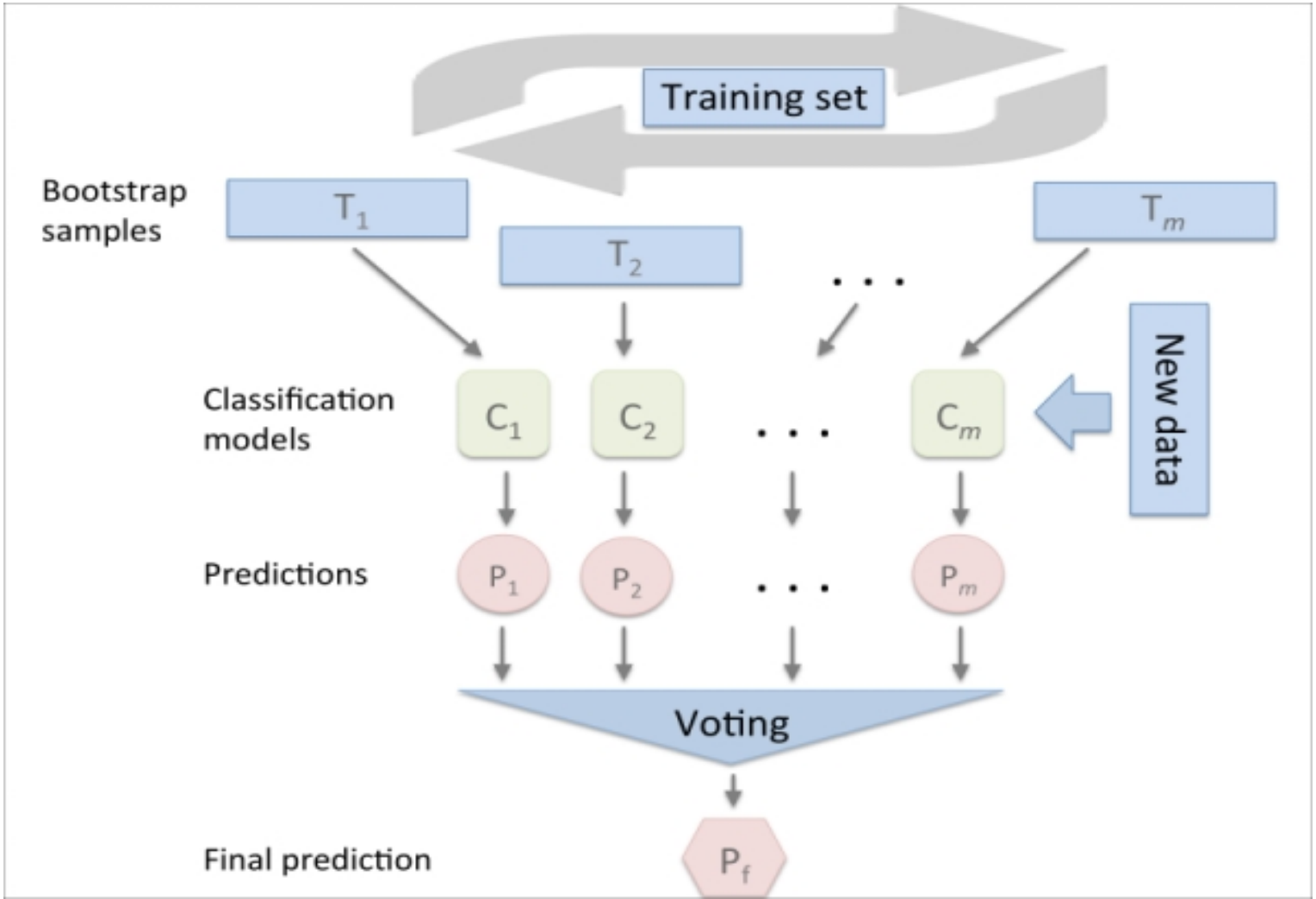


How Random Forest Work?

Most used algorithm- Bagging Technique

What insight can u get?




Training set
Bootstrap samples
T_1
T_2
T_m
Classification models
C_1
C_2
C_m
New data
Predictions
P_1
P_2
P_m
Voting
Final prediction
P_f

Random Forest Pseudocode

(a) Draw a bootstrap sample $\mathbf{Z}^*$ of size N from the training data.

(b) Grow a random-forest tree T_b to the bootstrapped data, by recursively repeating the following steps for each terminal node of the tree, until the minimum node size n_{min} is reached.

i. Select m variables at random from the p variables.

ii. Pick the best variable/split-point among the m.

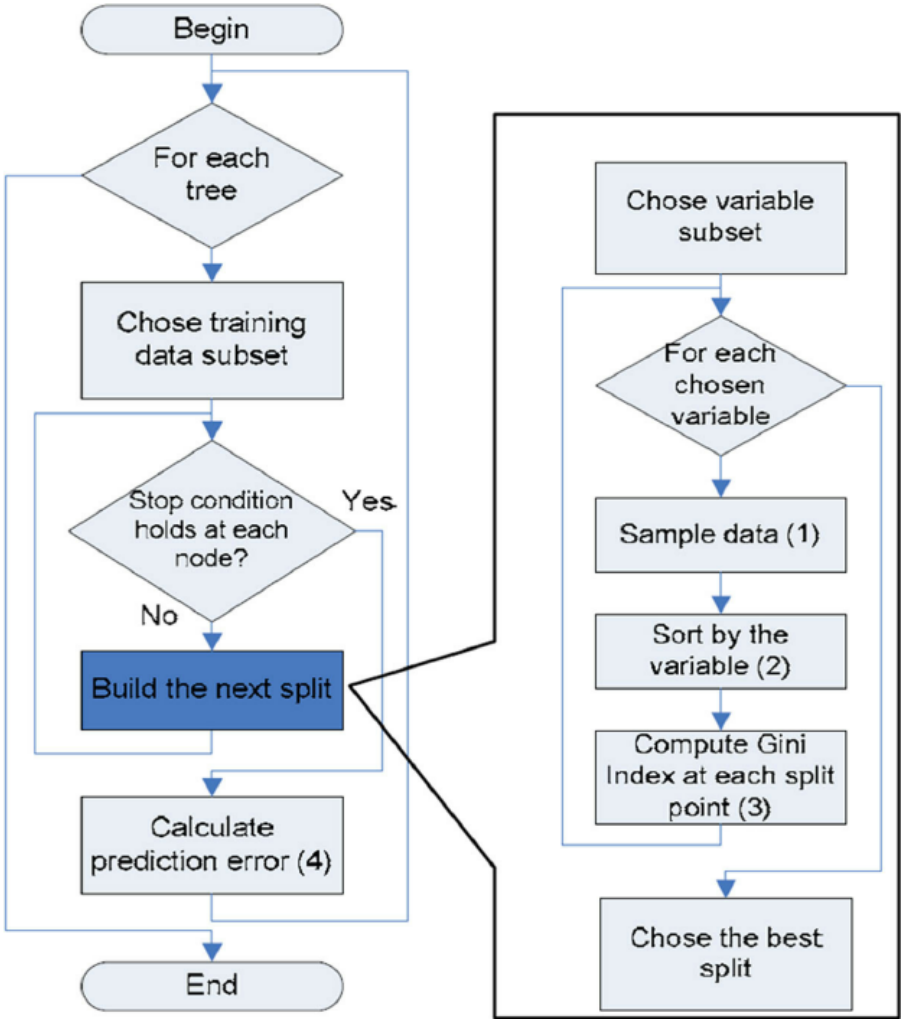
iii. Split the node into two daughter nodes.

2. Output the ensemble of trees $\{T_b\}_1^B$.

To make a prediction at a new point x:

Regression: $\hat{f}_{\text{rf}}^B(x) = \frac{1}{B}\sum_{b=1}^{B} T_b(x)$.

Classification: Let $\hat{C}_b(x)$ be the class prediction of the bth random-forest


Begin
For each tree
Chose training data subset
Stop condition holds at each node?
Yes
No
Build the next split
Calculate prediction error (4)
End
Chose variable subset
For each chosen variable
Sample data (1)
Sort by the variable (2)
Compute Gini Index at each split point (3)
Chose the best split

CONDITIONS

COLOR== PURPLE?

DIAMETER=3

COLOR== YELLOW?

COLOR== RED?

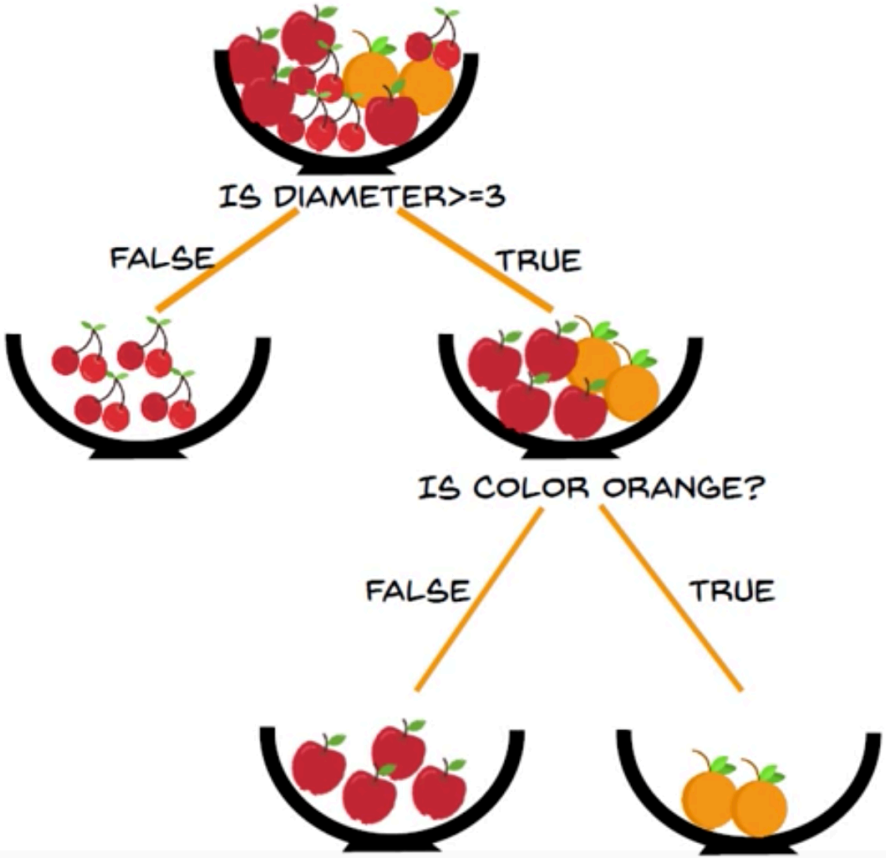
DIAMETER=1

LET'S SAY THIS CONDITION GIVES US THE MAXIMUM GAIN

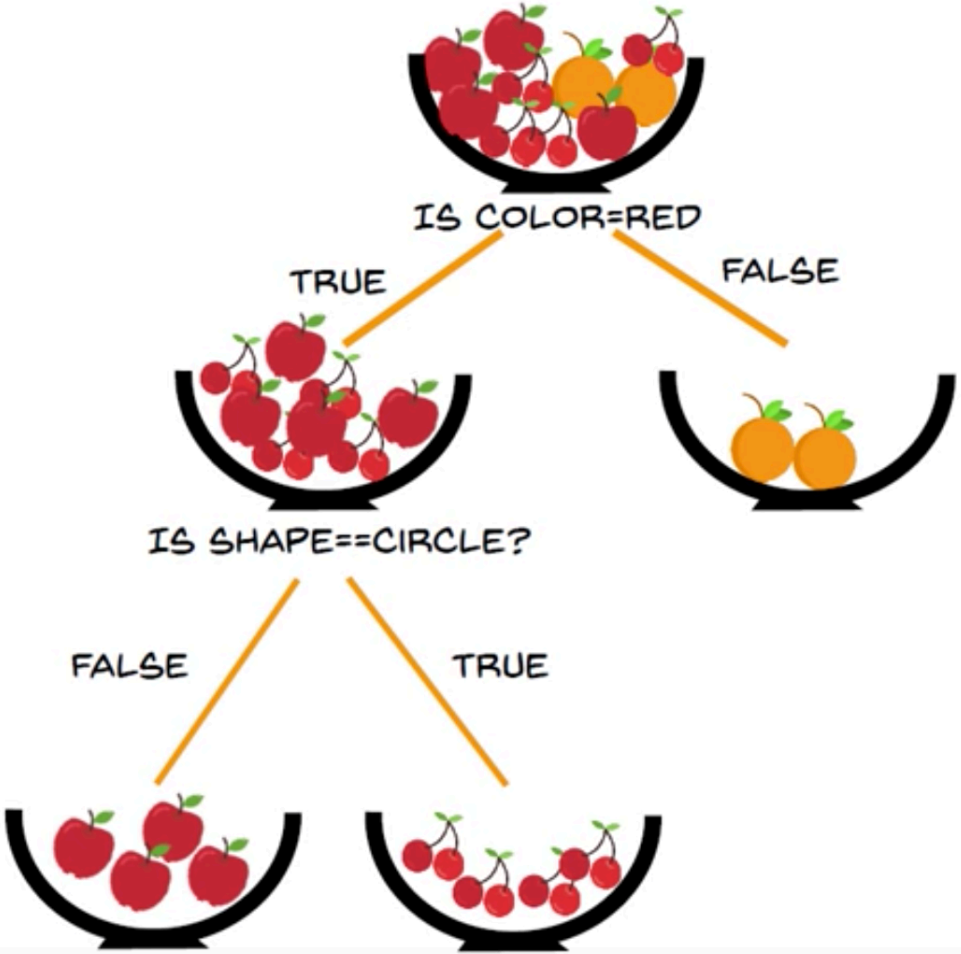
TRAINING DATASET

COLOR	DIAMETER	LABEL
RED	3	APPLE
YELLOW	3	LEMON
PURPLE	1	GRAPES
RED	3	APPLE
YELLOW	3	LEMON
PURPLE	1	GRAPES

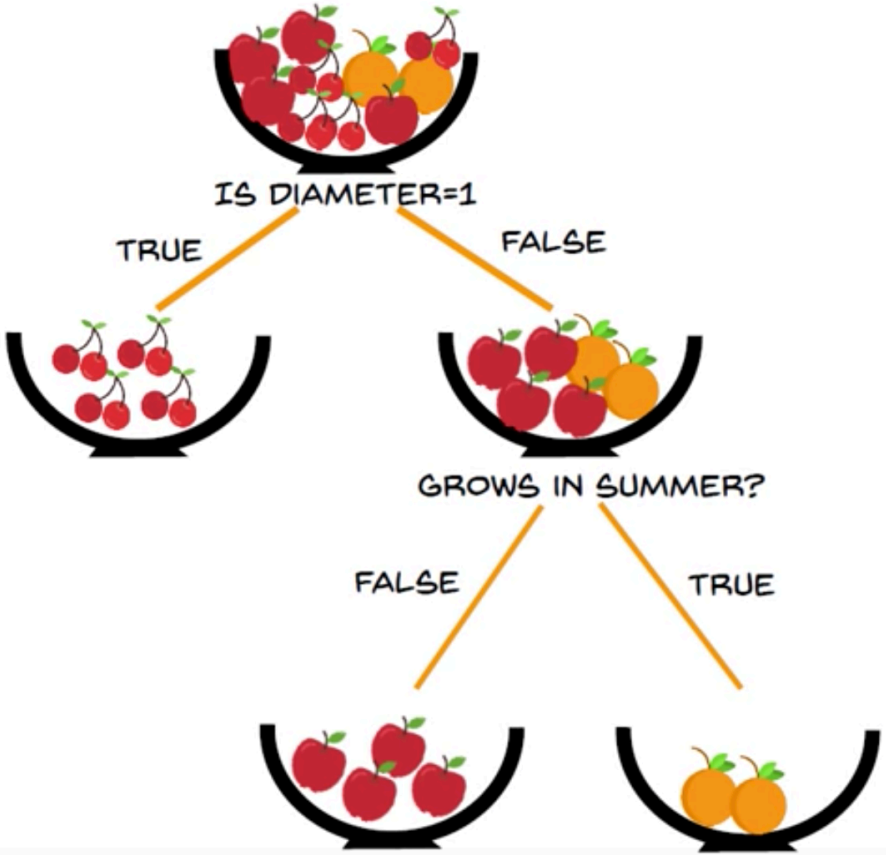
LET THIS BE TREE 1



LET THIS BE TREE 2



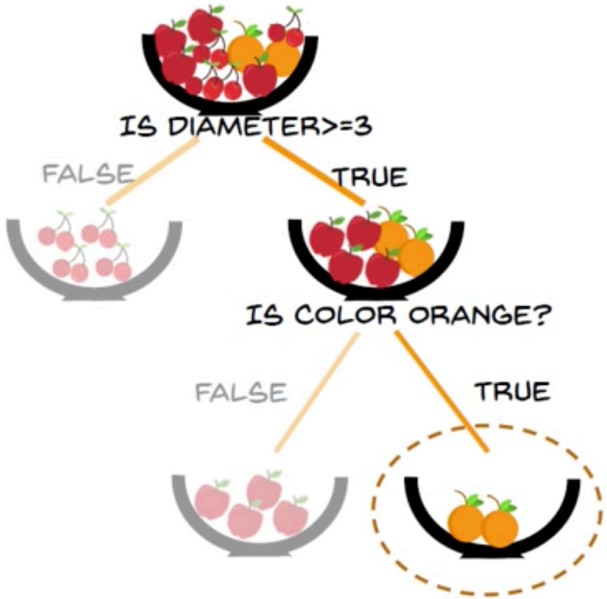
LET THIS BE TREE 3



NOW LETS TRY TO CLASSIFY THIS FRUIT

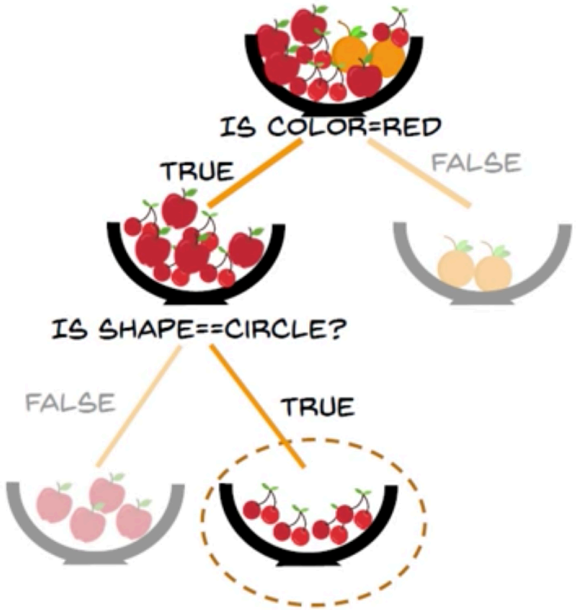
TREE 1 CLASSIFIES
IT AS AN ORANGE

DIAMETER = 3
COLOUR = ORANGE
GROWS IN SUMMER = YES
SHAPE = CIRCLE



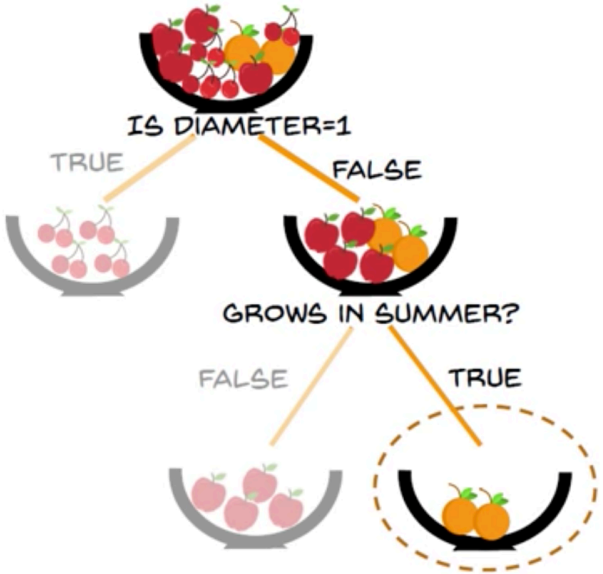
TREE 2 CLASSIFIES
IT AS CHERRIES

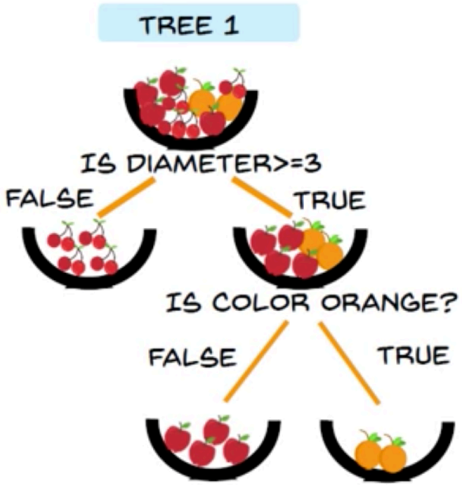
DIAMETER = 3
COLOUR = ORANGE
GROWS IN SUMMER = YES
SHAPE = CIRCLE

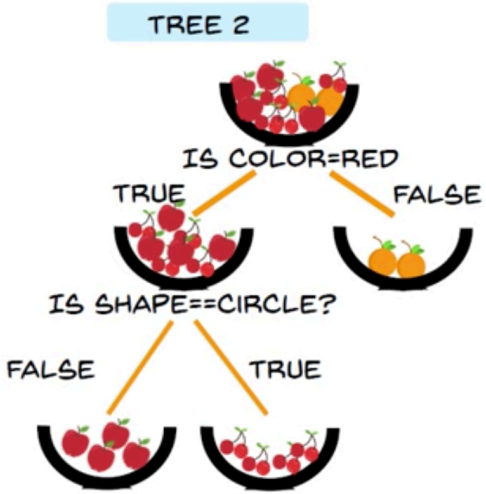


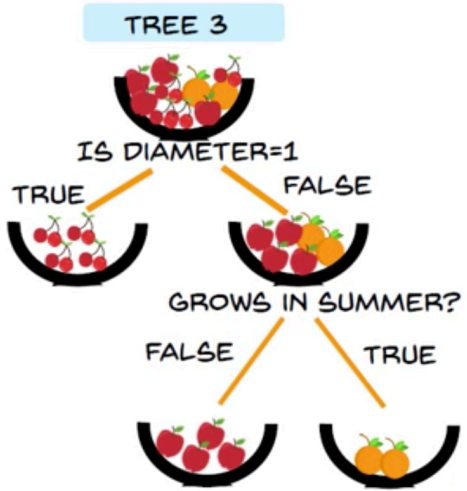
TREE 3 CLASSIFIES IT AS ORANGE

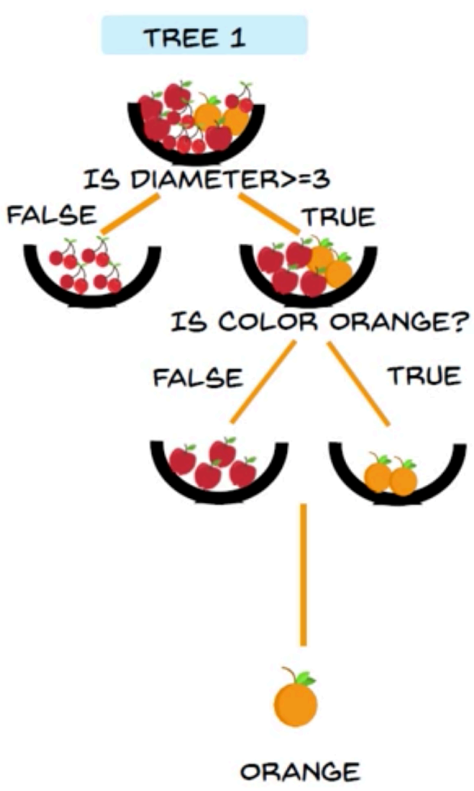
DIAMETER = 3
COLOUR = ORANGE
GROWS IN SUMMER = YES
SHAPE = CIRCLE

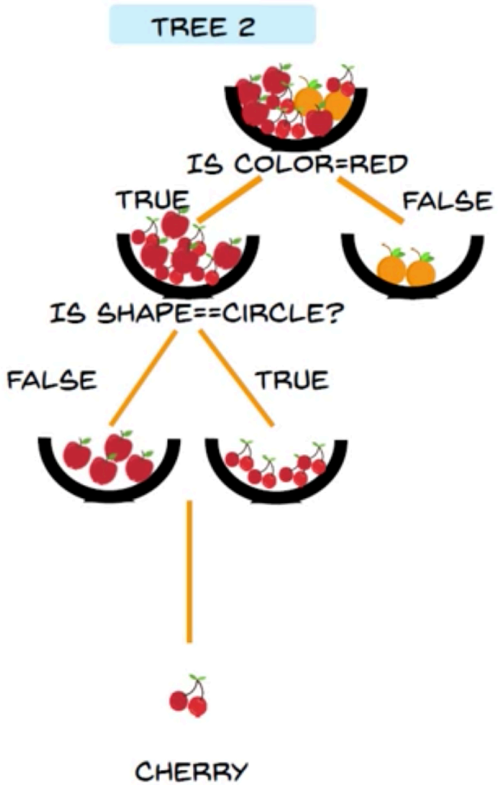


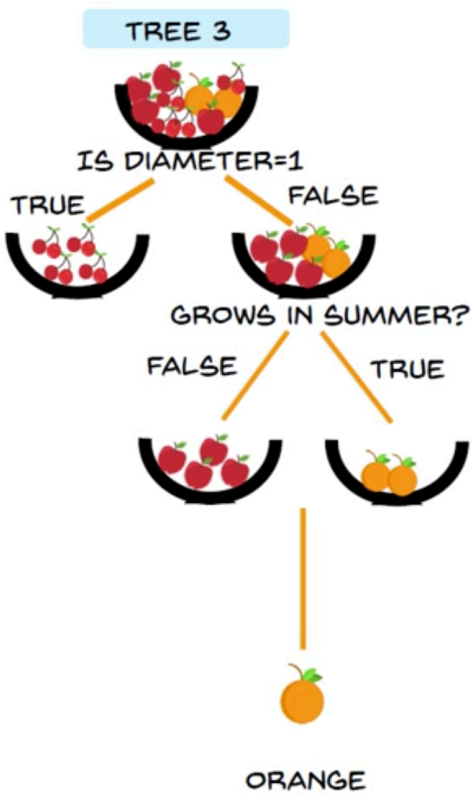

TREE 1
IS DIAMETER>=3
FALSE
TRUE
IS COLOR ORANGE?
FALSE
TRUE


TREE 2
IS COLOR=RED
TRUE
FALSE
IS SHAPE==CIRCLE?
FALSE
TRUE


TREE 3
IS DIAMETER=1
TRUE
FALSE
GROWS IN SUMMER?
FALSE
TRUE


TREE 1
IS DIAMETER>=3
FALSE
TRUE
IS COLOR ORANGE?
FALSE
TRUE
ORANGE


TREE 2
IS COLOR=RED
TRUE
FALSE
IS SHAPE==CIRCLE?
FALSE
TRUE
CHERRY


TREE 3
IS DIAMETER=1
TRUE
FALSE
GROWS IN SUMMER?
FALSE
TRUE
ORANGE

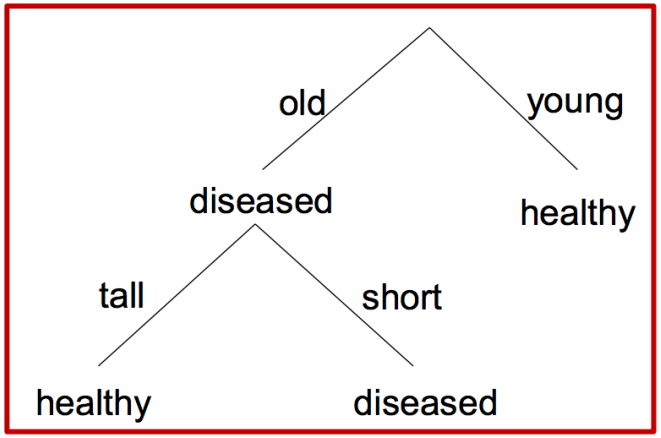
ORANGE

MAJORITY
VOTED

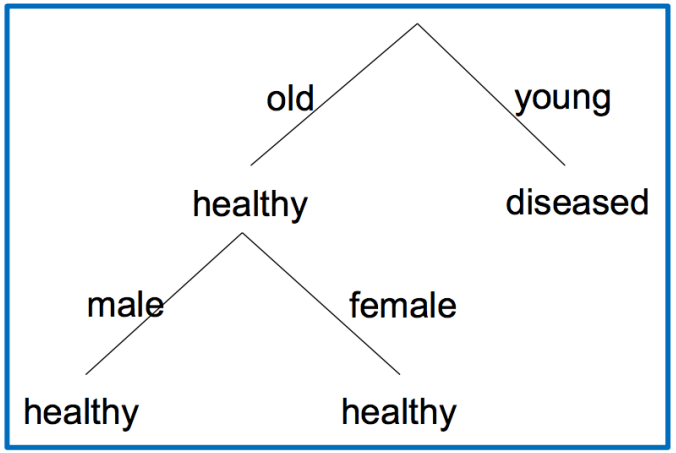
CHERRY

SO THE FRUIT IS CLASSIFIED AS AN ORANGE

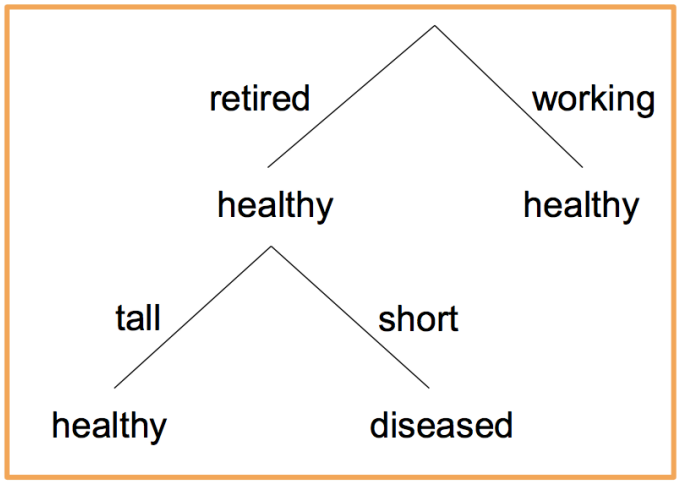
Tree 1



Tree 2



Tree 3



New sample:

old, retired, male, short

Tree predictions:

diseased, healthy, diseased

Majority rule:

diseased

Points to remember

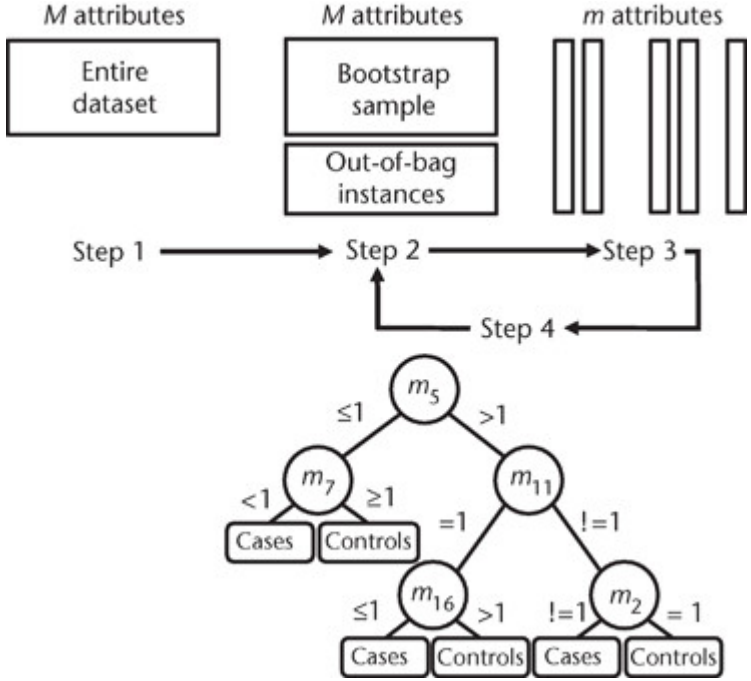
- For classification a good default is: **m = sqrt(p)**
- For regression a good default is: **m = p/3**

Estimated Performance

- For each bootstrap sample taken from the training data, there will be samples left behind that were not included. These samples are called **Out-Of-Bag samples** or OOB.
- The performance of each model on its left out samples when averaged can provide an estimated accuracy of the bagged models. This estimated performance is often called the OOB estimate of performance.
- These performance measures are reliable test error estimate and correlate well with cross validation estimates.

Variable importance

- It will find most important variable for feature selection based on Gini index


M attributes
Entire dataset
M attributes
Bootstrap sample
Out-of-bag instances
m attributes
Step 1
Step 2
Step 3
Step 4
m_5
≤1
>1
m_7
m_{11}
<1
≥1
=1
!=1
Cases
Controls
m_{16}
m_2
≤1
>1
!=1
= 1
Cases
Controls
Cases
Controls

Random forest cons

It is one of the most accurate learning algorithms available.

For many data sets, it produces a highly accurate classifier.

It runs efficiently on large databases.

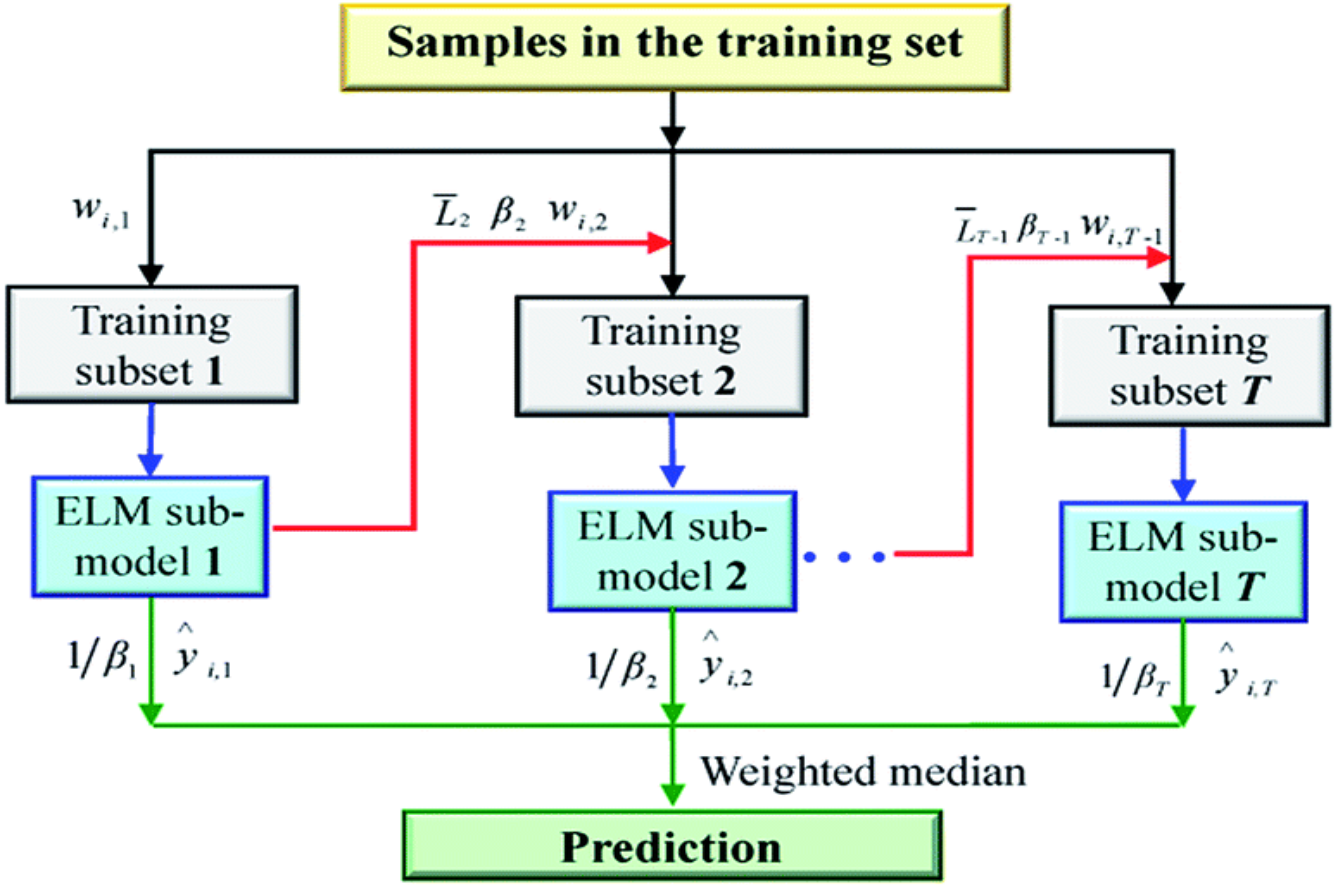
Conclusion

- Random forests are an effective tool in prediction.
- Forests give results competitive with boosting and adaptive bagging, yet do not progressively change the training set.
- Random inputs and random features produce good results in classification- less so in regression.
- For larger data sets, we can gain accuracy by combining random features with boosting.

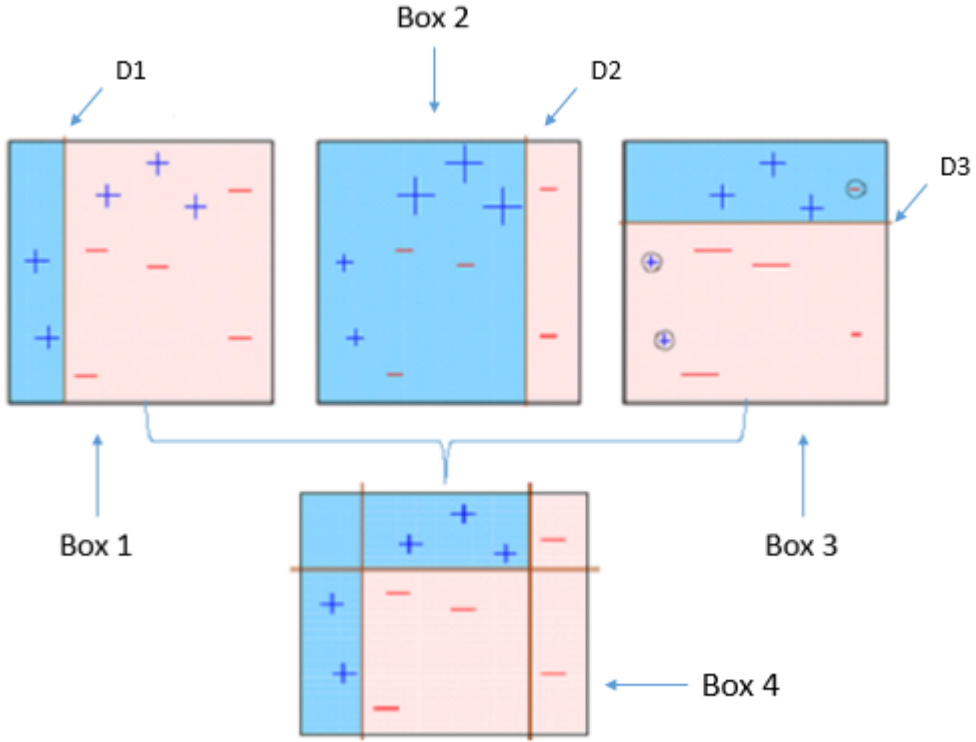
Boosting

Competition winning algorithm

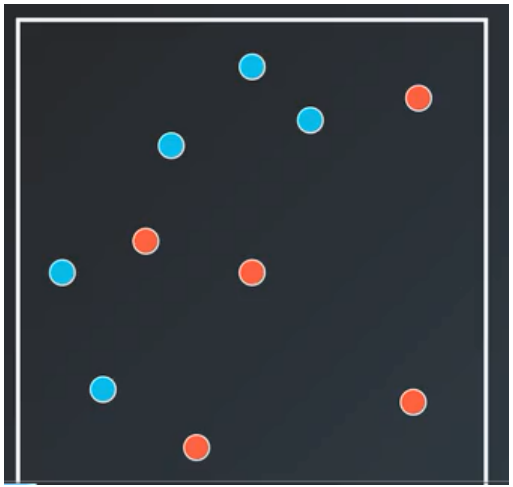
Boosting


Samples in the training set
$w_{i,1}$
$\overline{L}_2\ \beta_2\ w_{i,2}$
$\overline{L}_{T-1}\ \beta_{T-1}\ w_{i,T-1}$
Training subset 1
Training subset 2
Training subset T
ELM sub-model 1
ELM sub-model 2
ELM sub-model T
$1/\beta_1$ $\hat{y}_{i,1}$
$1/\beta_2$ $\hat{y}_{i,2}$
$1/\beta_T$ $\hat{y}_{i,T}$
Weighted median
Prediction

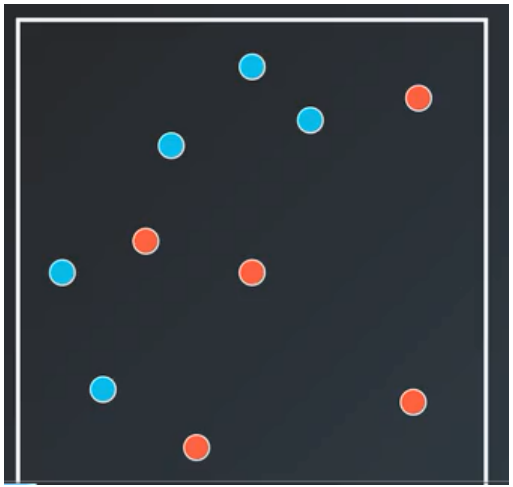
AdaBoos



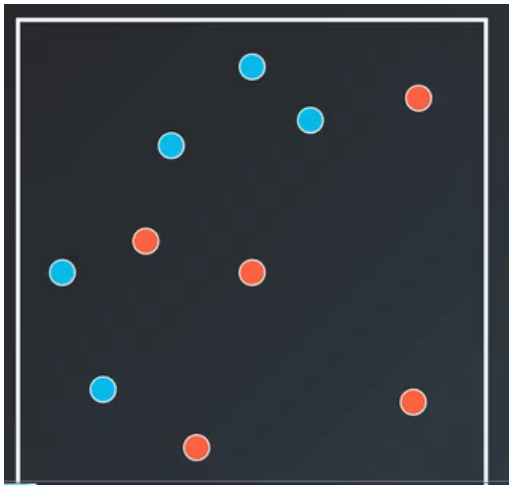
AdaBoost (Adaptive Boosting)

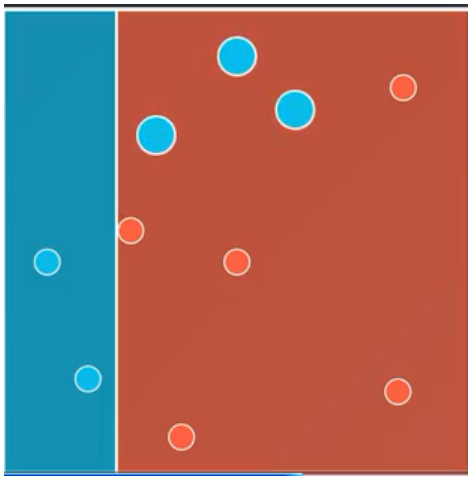


AdaBoost – Pattern 1

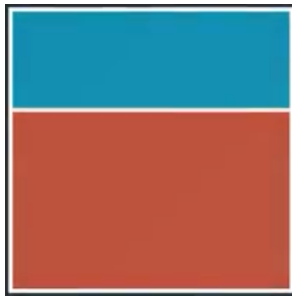


AdaBoost – Pattern 1



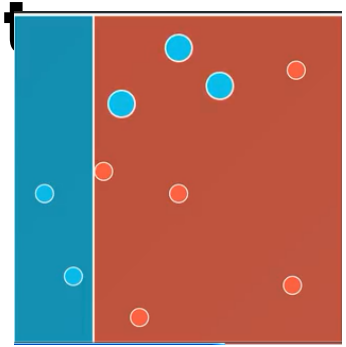


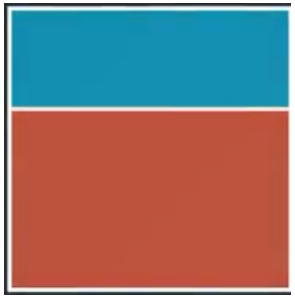
AdaBoost – Pattern 2

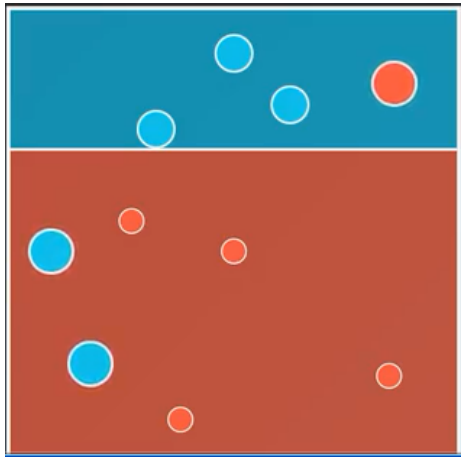


AdaBoost – Pattern 2

Input Dat





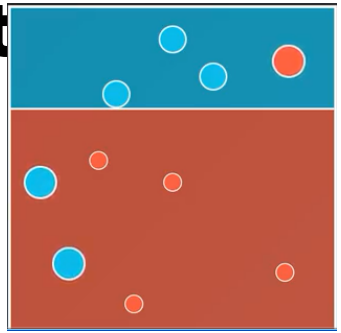


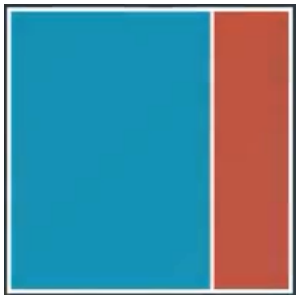
AdaBoost – Pattern 3

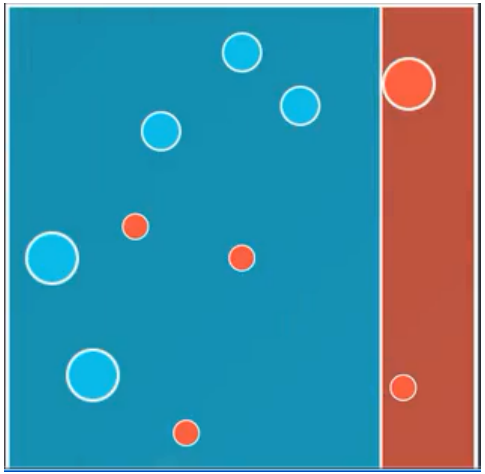
Apply pattern 3 on the Input Data from

AdaBoost – Pattern 3

Input Dat

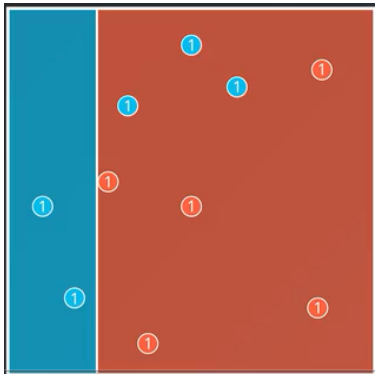




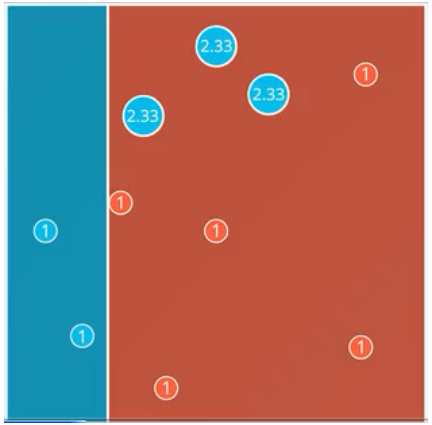


AdaBoost – Pattern 1

Weights after applying pattern 1



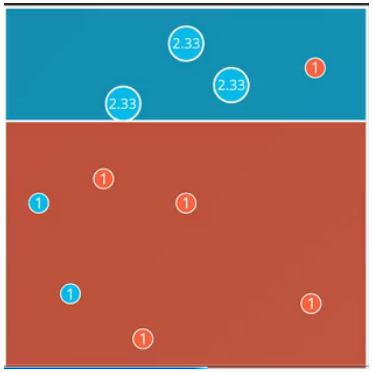
Correct: 7
Incorrect: 3



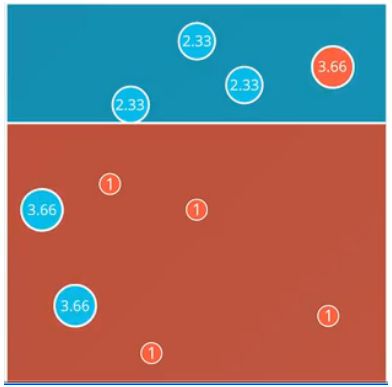
Correct: 7
Incorrect: 7

AdaBoost – Pattern 2

Weights after applying pattern 2



Correct: 11
Incorrect: 3



Correct: 11
Incorrect: 11

AdaBoost – Pattern 3

Weights after applying pattern 3

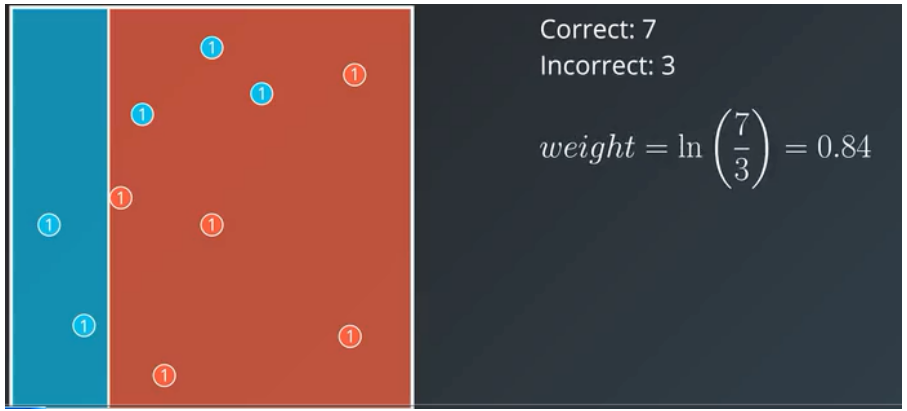
Correct: 19
Incorrect: 3

AdaBoost – 3 Models

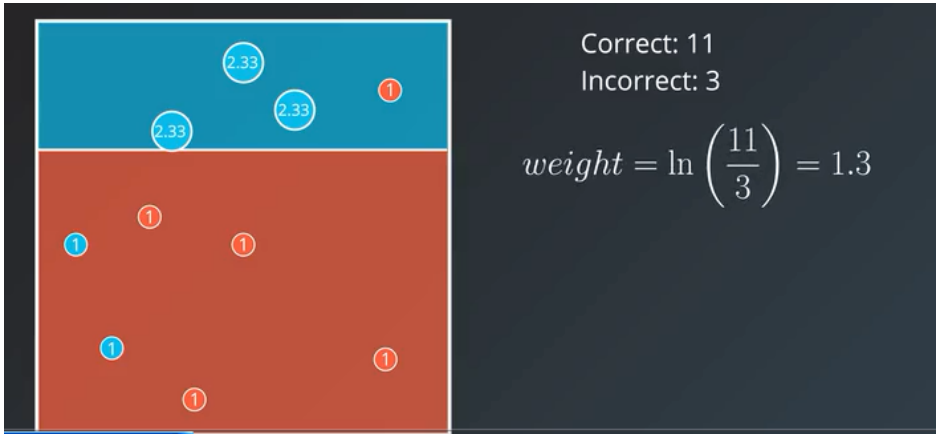
Weightage of a Model

$$weight = \ln\left(\frac{\#correct}{\#incorrect}\right)$$

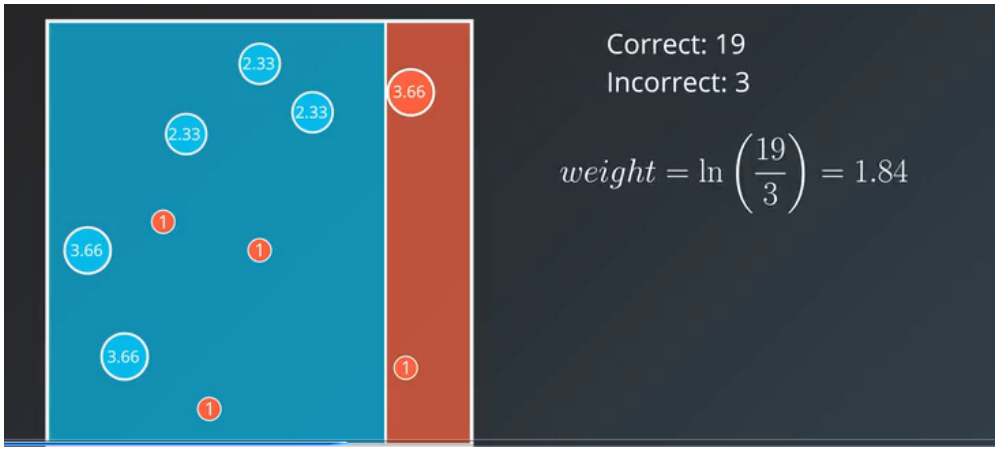
Weight of Model 1



Weight of Model 2

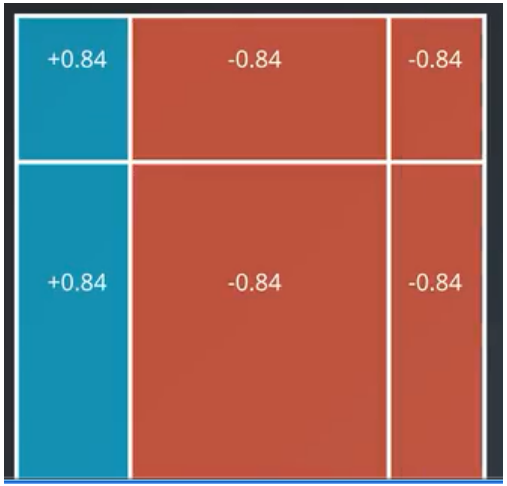


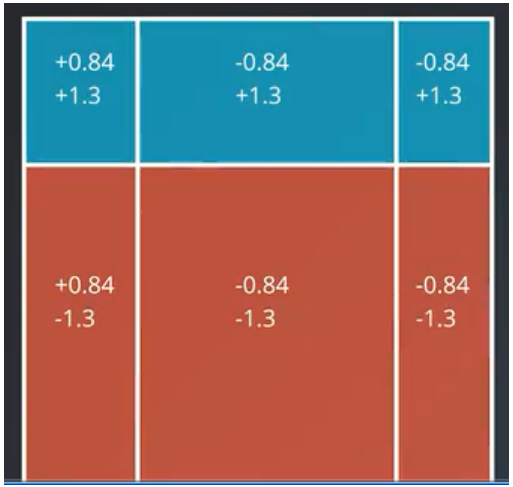
Weight of Model 3

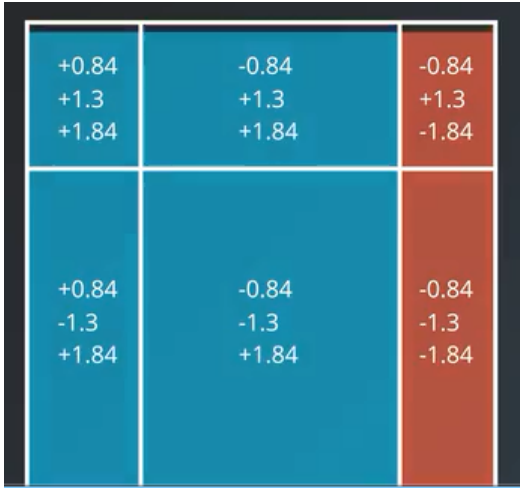


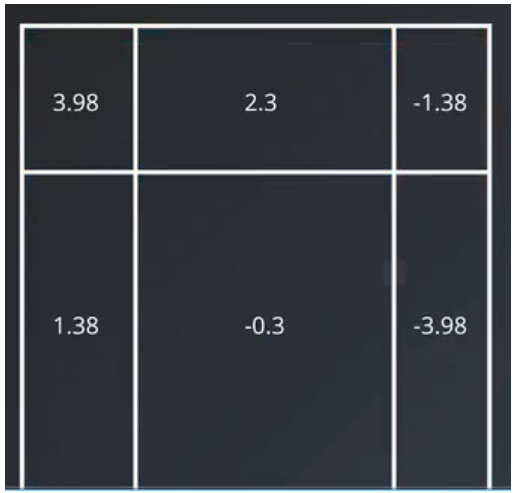
Weight of 3 Models

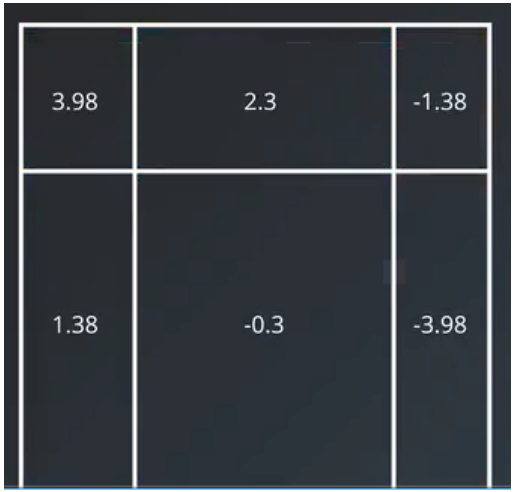
Assinging weights to 2 categories


+0.84
-0.84
-0.84
+0.84
-0.84
-0.84

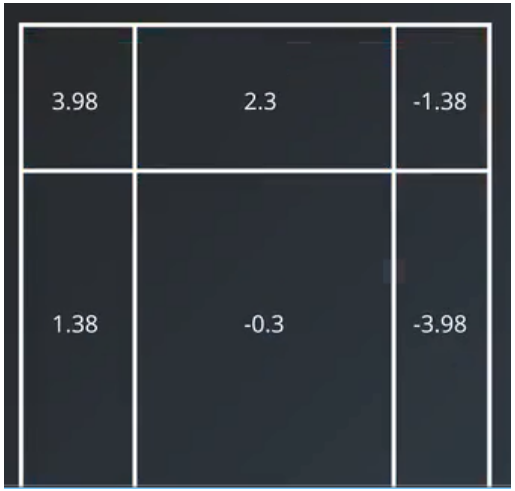

+0.84
+1.3
-0.84
+1.3
-0.84
+1.3
+0.84
-1.3
-0.84
-1.3
-0.84
-1.3


+0.84
+1.3
+1.84
-0.84
+1.3
+1.84
-0.84
+1.3
-1.84
+0.84
-1.3
+1.84
-0.84
-1.3
+1.84
-0.84
-1.3
-1.84


3.98
2.3
-1.38
1.38
-0.3
-3.98


3.98
2.3
-1.38
1.38
-0.3
-3.98


3.98
2.3
-1.38
1.38
-0.3
-3.98


3.98
2.3
-1.38
1.38
-0.3
-3.98


3.98
2.3
-1.38
1.38
-0.3
-3.98


3.98
2.3
-1.38
1.38
-0.3
-3.98

Final Model