# RANDOM FOREST

simplilearn

# Application of Random Forest

**Remote Sensing**

Used in ETM devices to acquire images of the earth's surface.

Accuracy is higher and training time is less


**Object Detection**

Multiclass object detection is done using Random Forest algorithms

Provides better detection in complicated environments

Kinect

Random Forest is used in a game console called Kinect

Tracks body movements and recreates it in the game

# Application of Random Forest

**Remote Sensing**

Used in ETM devices to acquire images of the earth's surface.

Accuracy is higher and training time is less


**Object Detection**

Multiclass object detection is done using Random Forest algorithms

Provides better detection in complicated environments

**Kinect**

Random Forest is used in a game console called Kinect

Tracks body movements and recreates it in the game

# Application of Random Forest

Kinect

Random Forest is used in a game console called Kinect

Tracks body movements and recreates it in the game

# Application of Random Forest

User performs a step

Kinect registers the movement

Marks the user based on accuracy

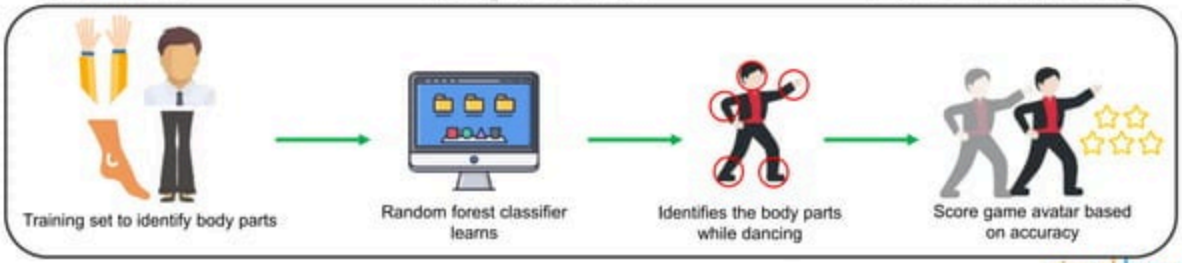
# Application of Random Forest

User performs a step

Kinect registers the movement



Marks the user based on accuracy



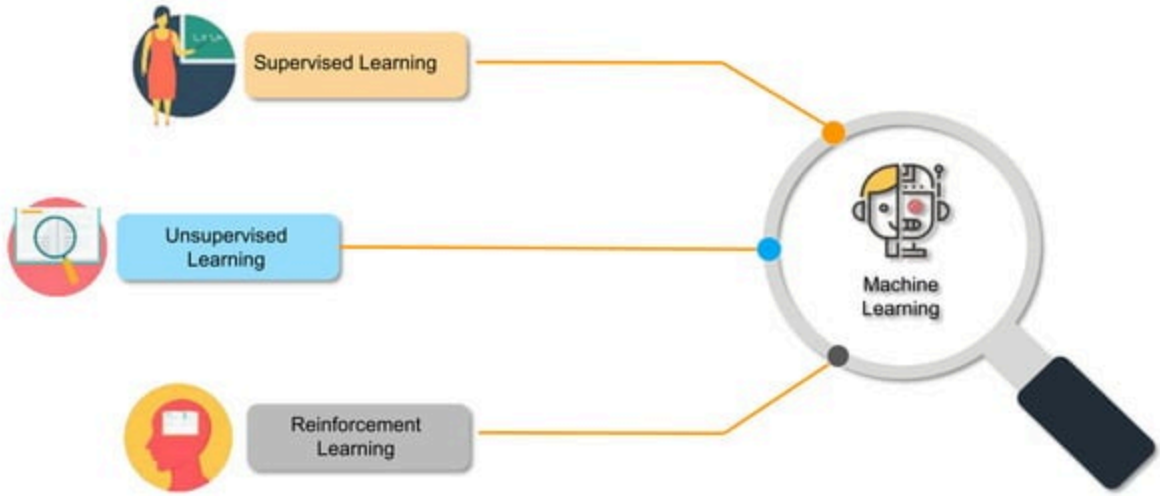
# What's in it for you?

- **What is Machine Learning?**
- Applications of Random Forest
- What is Classification?
- Why Random Forest?
- What is Random Forest?
- Random Forest and Decision Tree
- Comparing Random Forest and Regression
- Use Case – Iris Flower Analysis

simplilearn

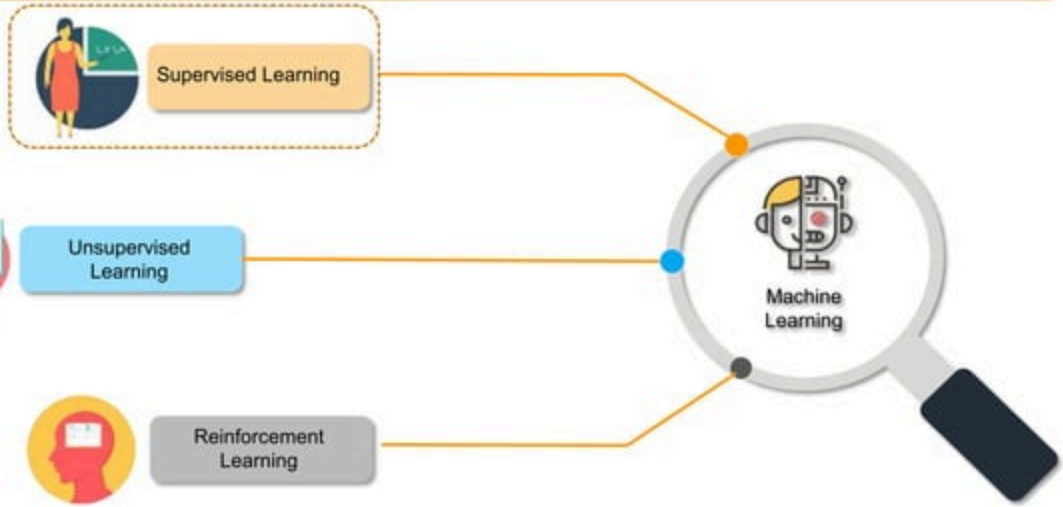

Types of Machine Learning
simplilearn

# Types of Machine Learning




simplilearn

# Types of Machine Learning




simplilearn

# Types of Supervised Learning

Supervised Learning



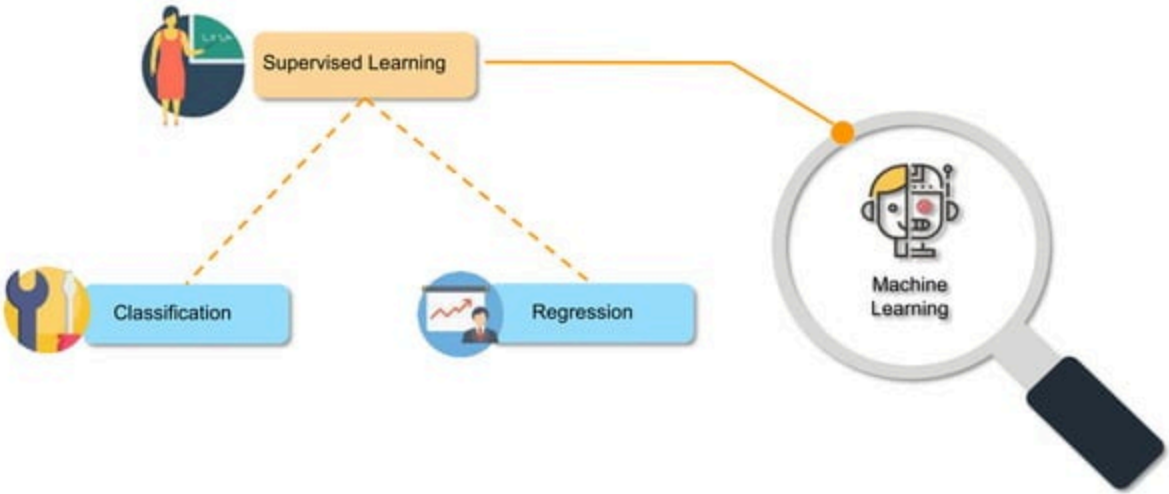
Unsupervised Learning



Reinforcement Learning


simplilearn

# Types of Supervised Learning




simplilearn

# What is Classification?

Supervised Learning

Classification

Regression


simplilearn

# What is Classification?

Supervised Learning

Classification

Classification is a kind of problem wherein the outputs are categorical in nature like 'Yes' or 'No', 'True' or 'False', '0' or '1'


simplilearn

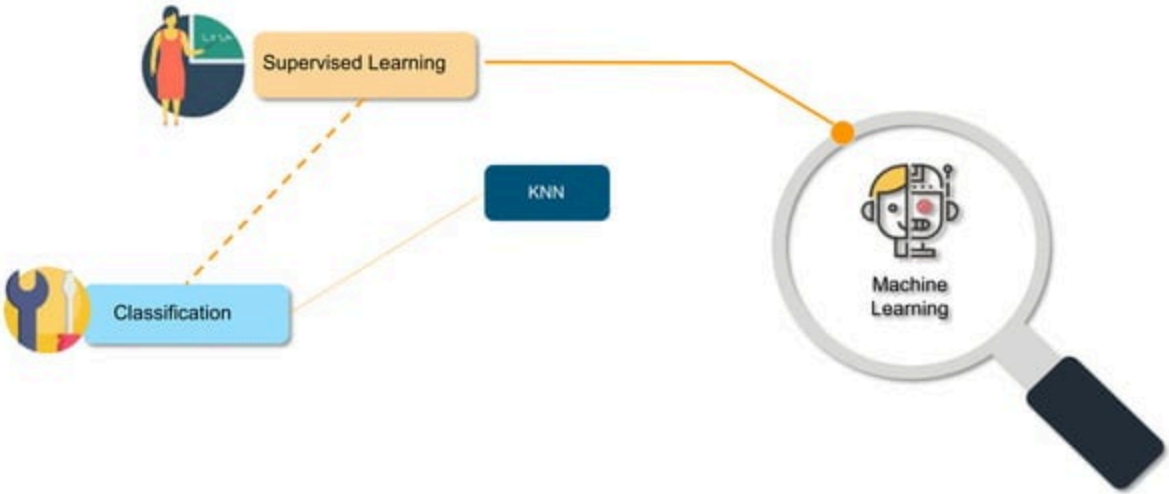
# Solutions under Classification

Supervised Learning

Classification

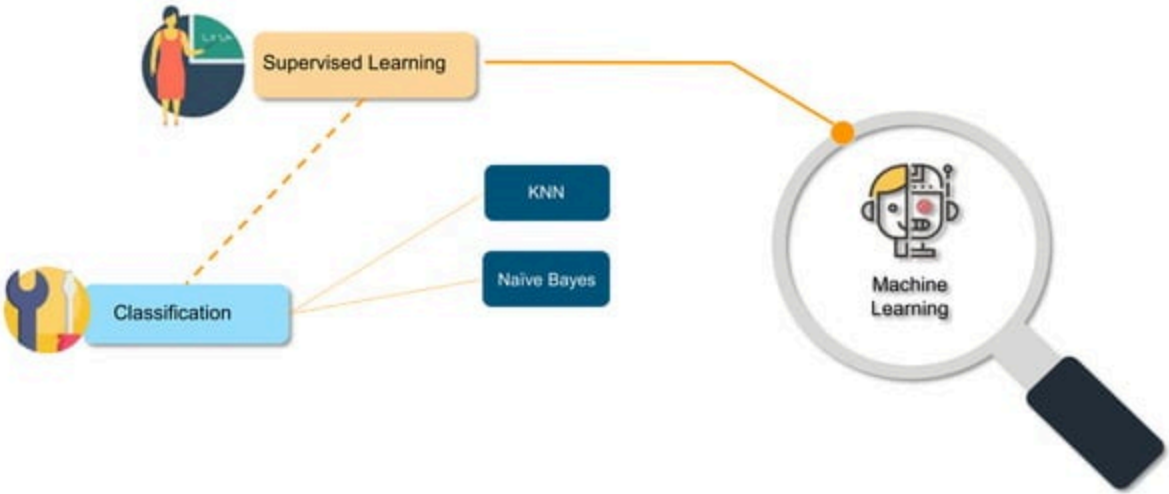

simplilearn

# Solutions under Classification



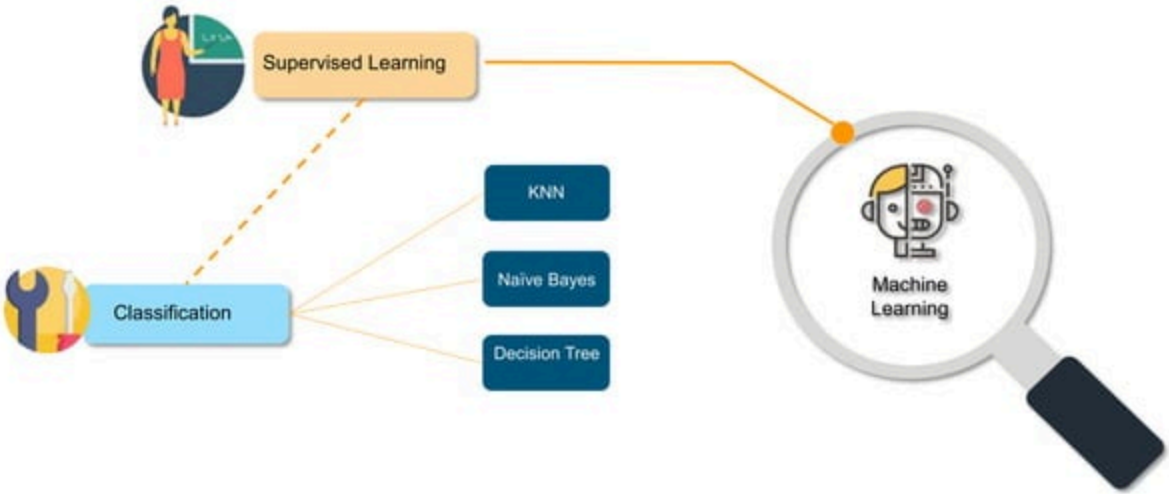

simplilearn

# Solutions under Classification




simplilearn

# Solutions under Classification




simplilearn

# Solutions under Classification

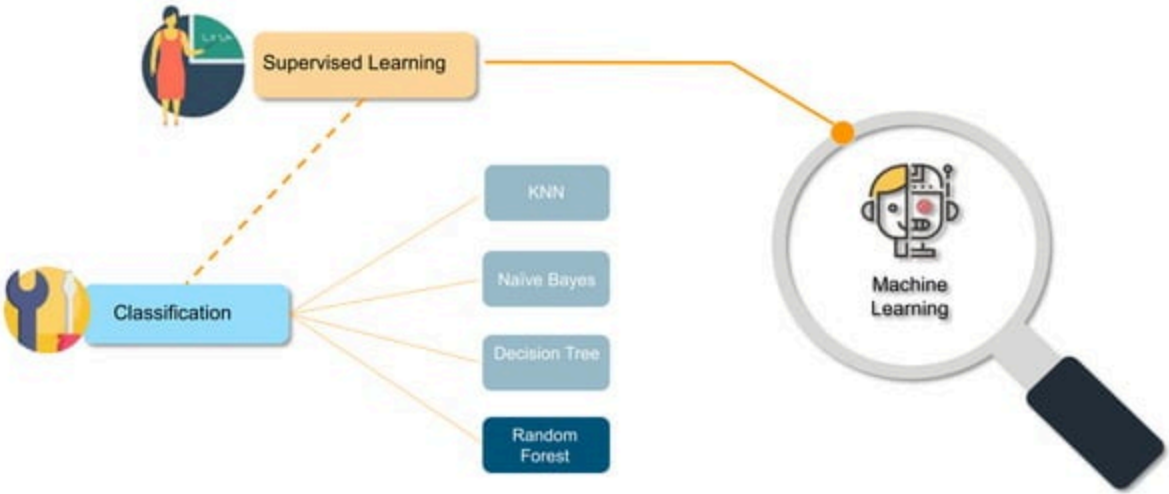
Supervised Learning

Classification

- KNN
- Naive Bayes
- Decision Tree
- Random Forest


simplilearn

# Solutions under Classification




simplilearn


Why Random Forest?
simplilearn

# 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


What is Random Forest?
simplilearn

# What is Random Forest?

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

# What is Random Forest?

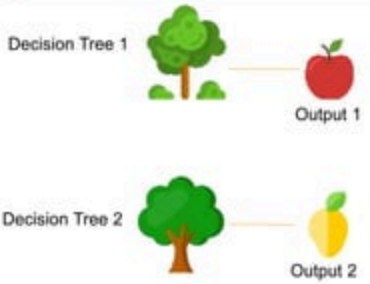
Random forest or Random Decision Forest is a method that operates by constructing multiple Decision Trees during training phase.
The Decision of the majority of the trees is chosen by the random forest as the final decision

Decision Tree 1

Output 1

# What is Random Forest?

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

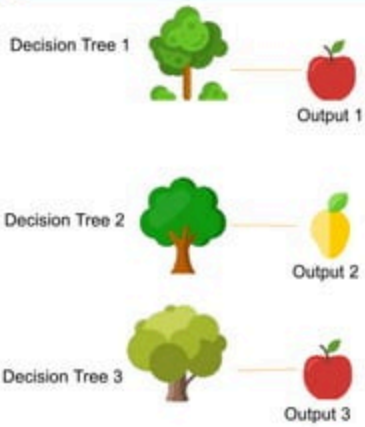





simplilearn

# What is Random Forest?

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

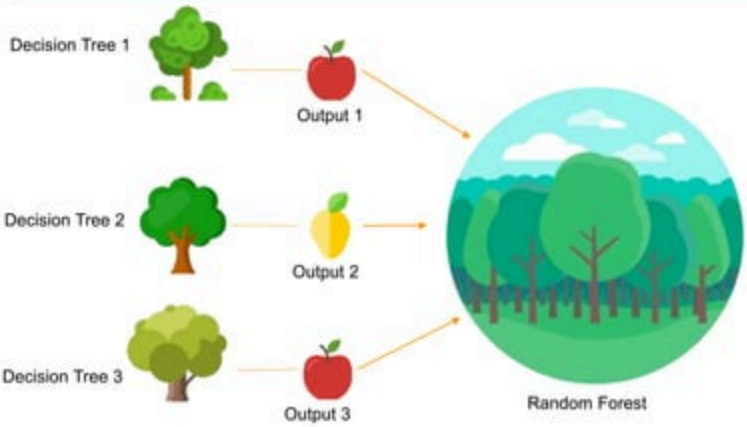





simplilearn

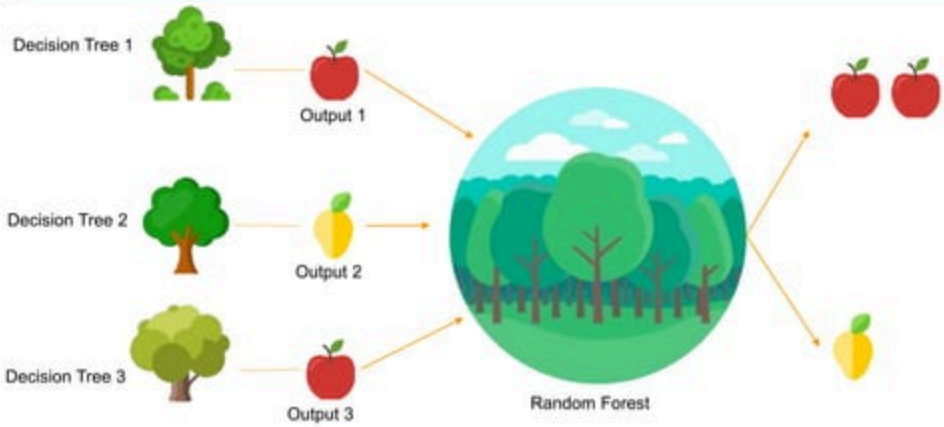
# What is Random Forest?

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



# What is Random Forest?

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

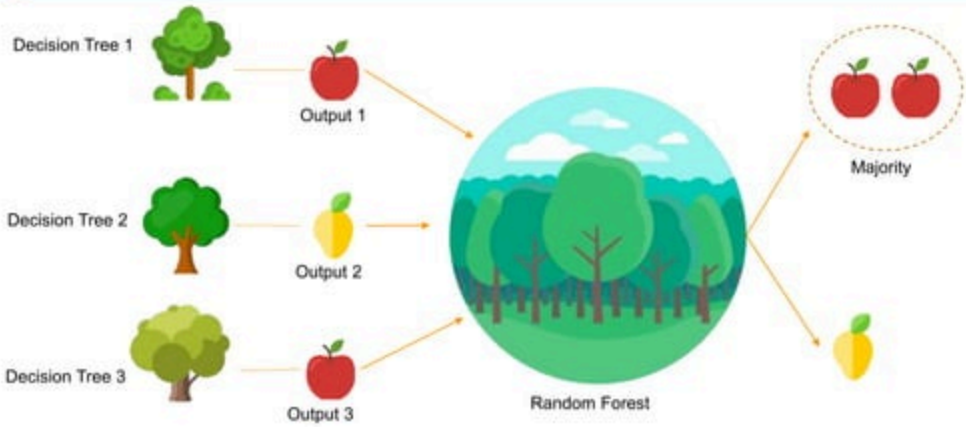





simplilearn

# What is Random Forest?

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

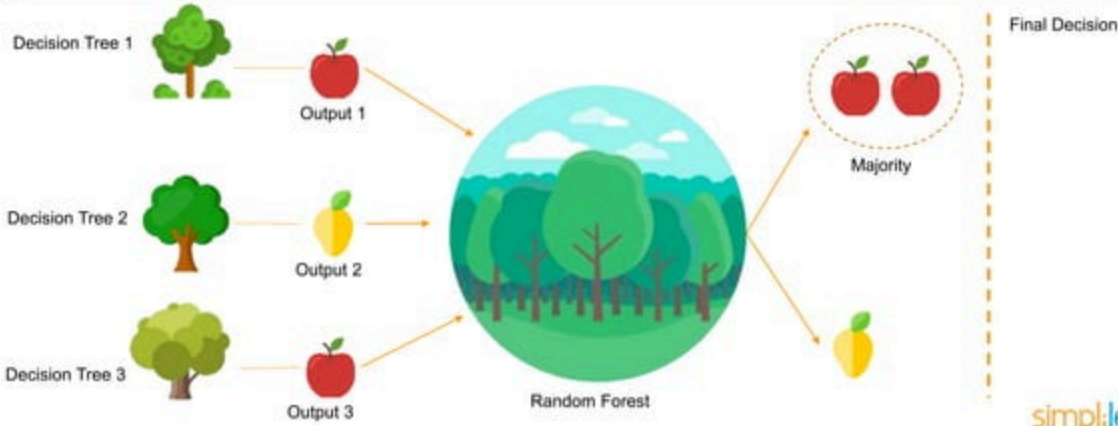





simplilearn

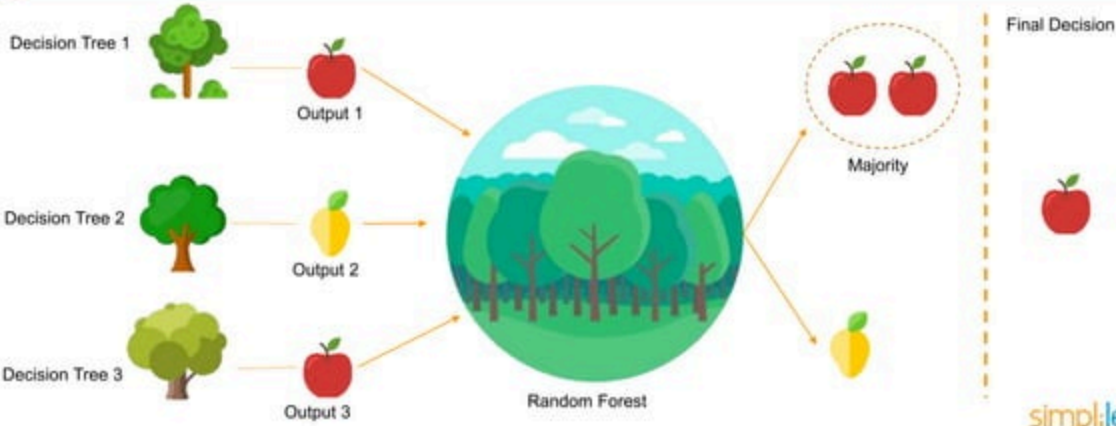
# What is Random Forest?

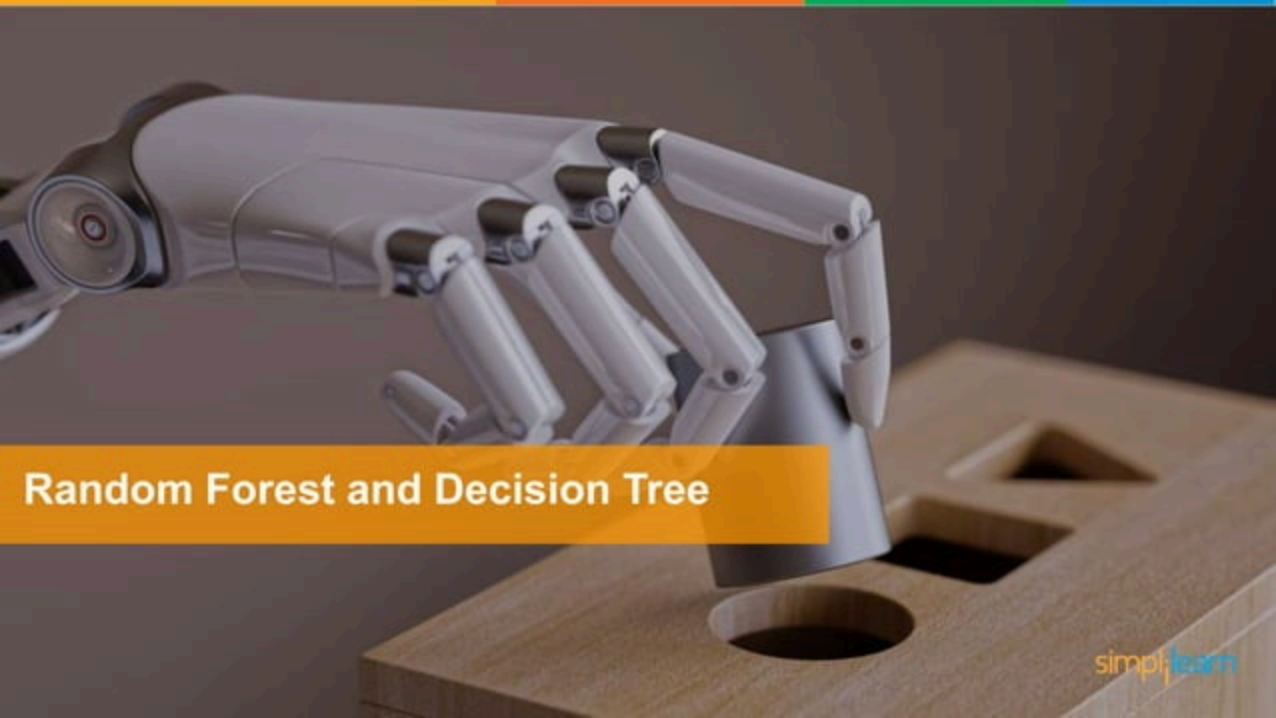
Random forest or Random Decision Forest is a method that operates by constructing multiple Decision Trees during training phase.
The Decision of the majority of the trees is chosen by the random forest as the final decision



# What is Random Forest?

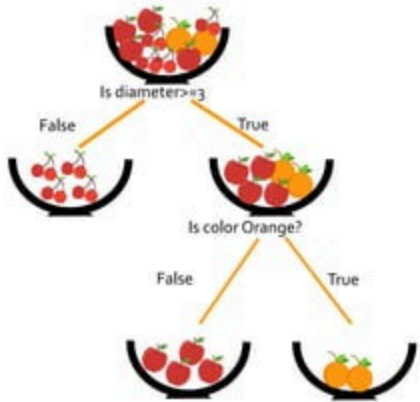
Random forest or Random Decision Forest is a method that operates by constructing multiple Decision Trees during training phase.
The Decision of the majority of the trees is chosen by the random forest as the final decision




Random Forest and Decision Tree
simplilearn

# Decision Tree

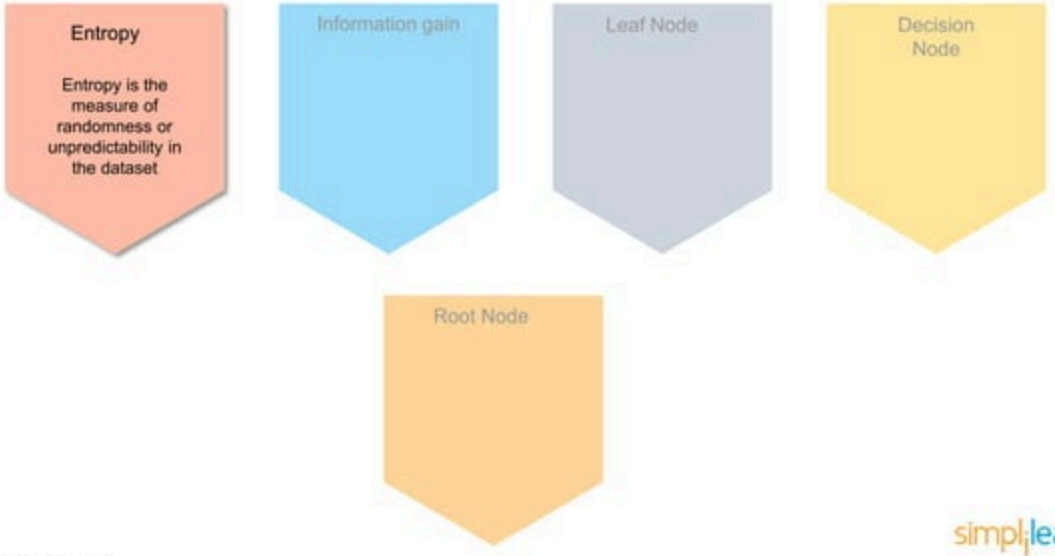
Decision Tree is a tree shaped diagram used to determine a course of action. Each branch of the tree represents a possible decision, occurrence or reaction






simplilearn

# Decision Tree- Important Terms






simplilearn

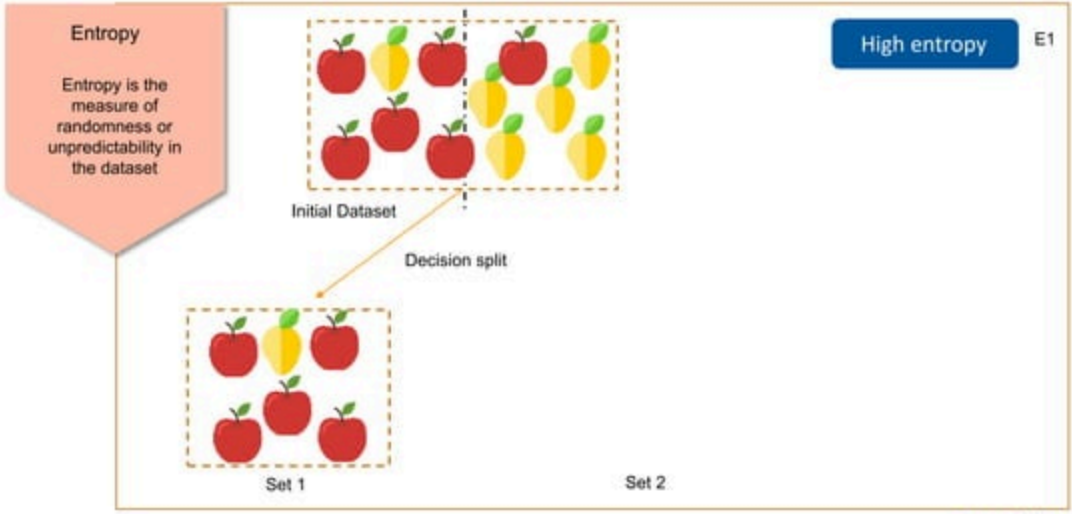
# Decision Tree - Important Terms

**Entropy**

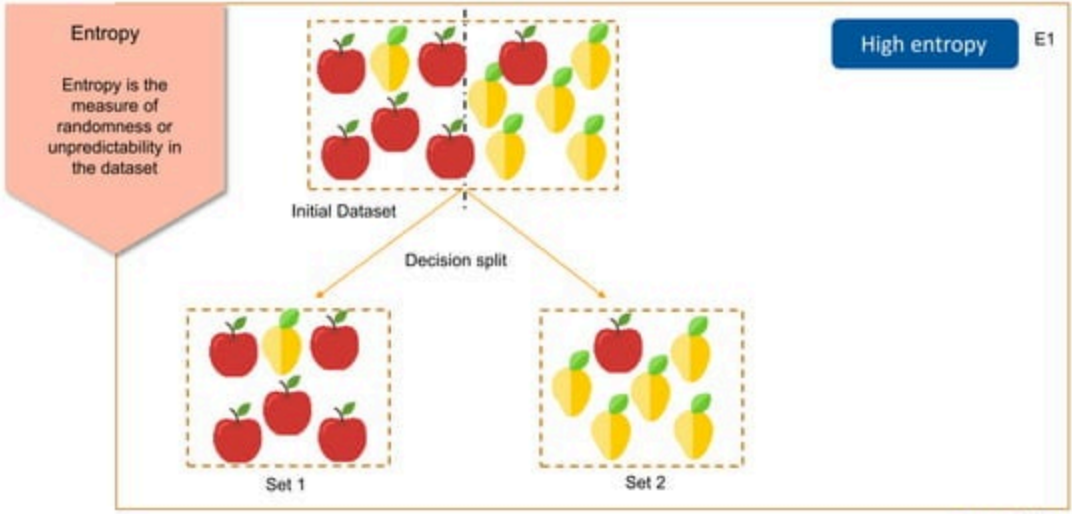
Entropy is the measure of randomness or unpredictability in the dataset

High entropy E1

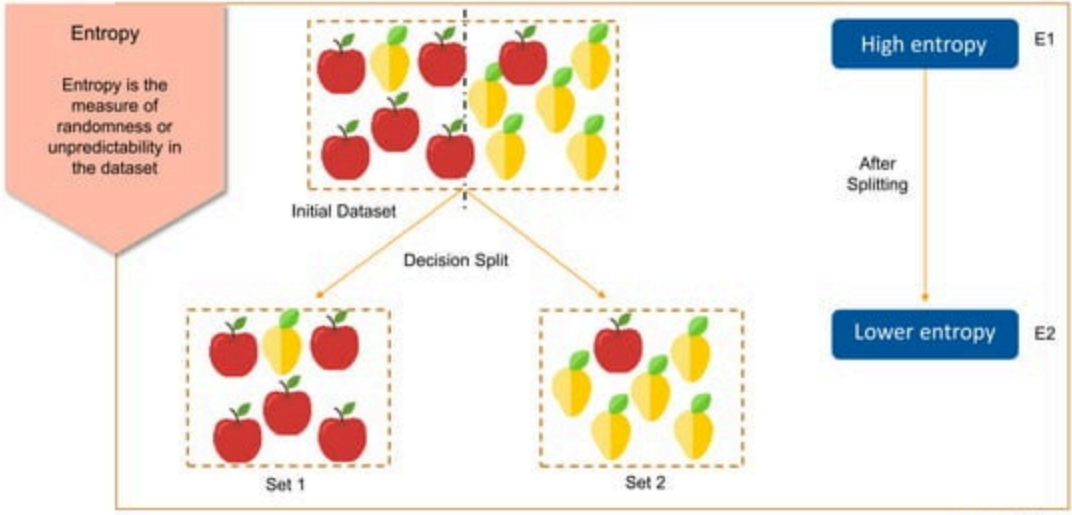
# Decision Tree - Important Terms



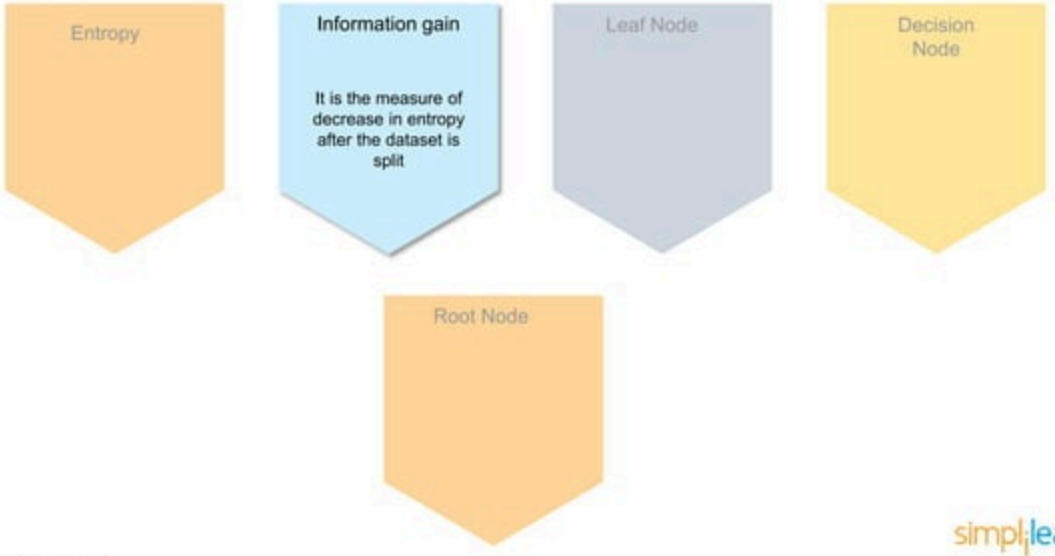
# Decision Tree - Important Terms



# Decision Tree - Important Terms



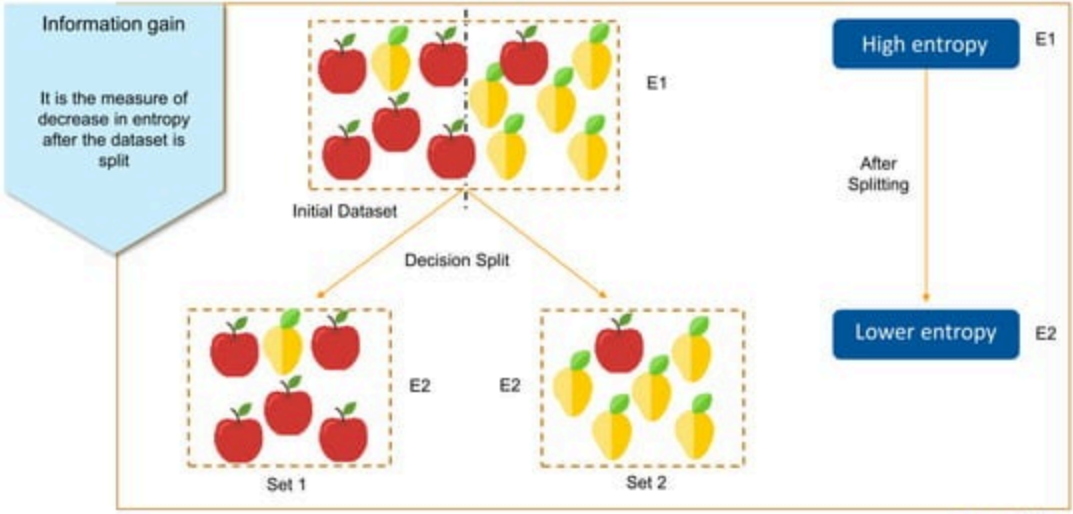
# Decision Tree - Important Terms





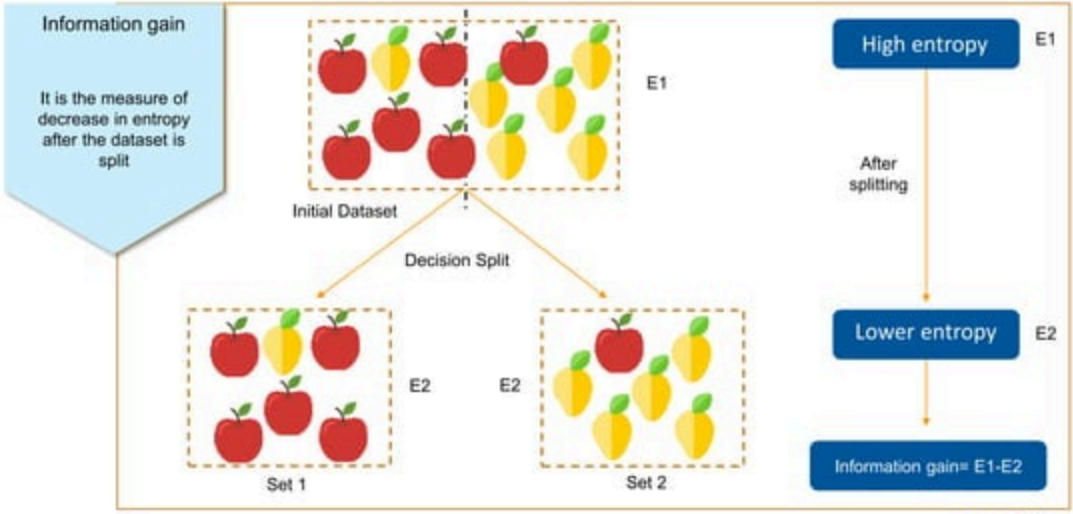

simplilearn

# Decision Tree - Important Terms



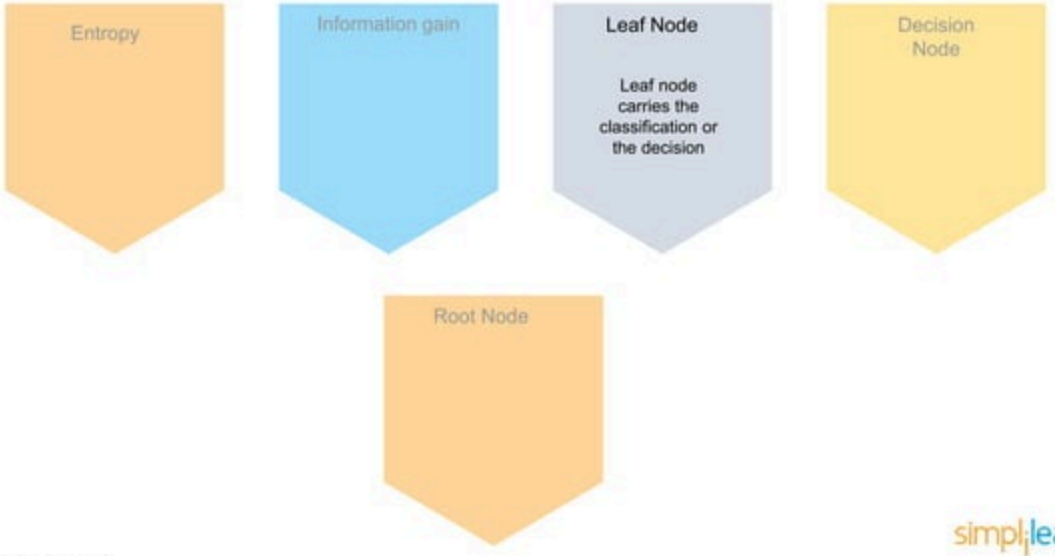

simplilearn

# Decision Tree - Important Terms




simplilearn

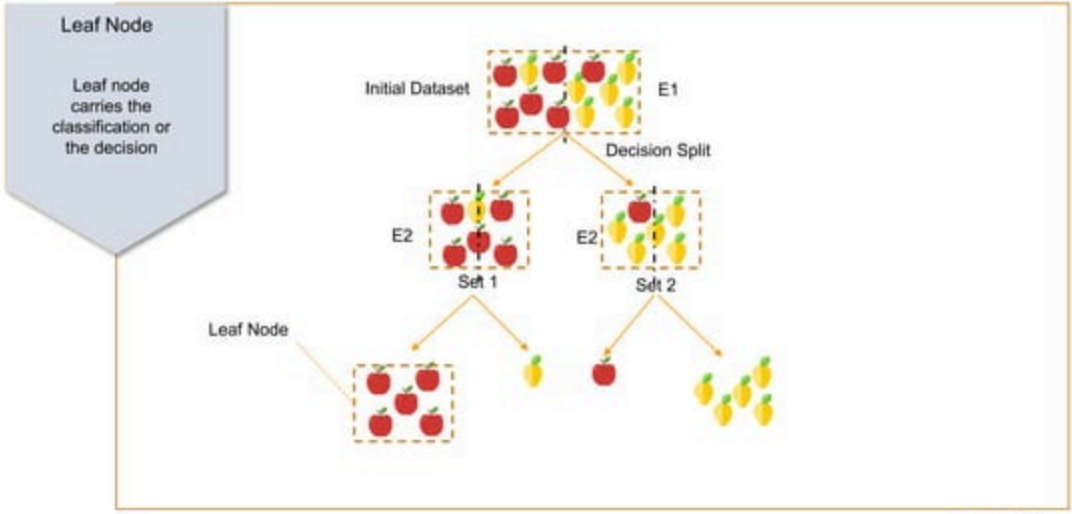
# Decision Tree - Important Terms





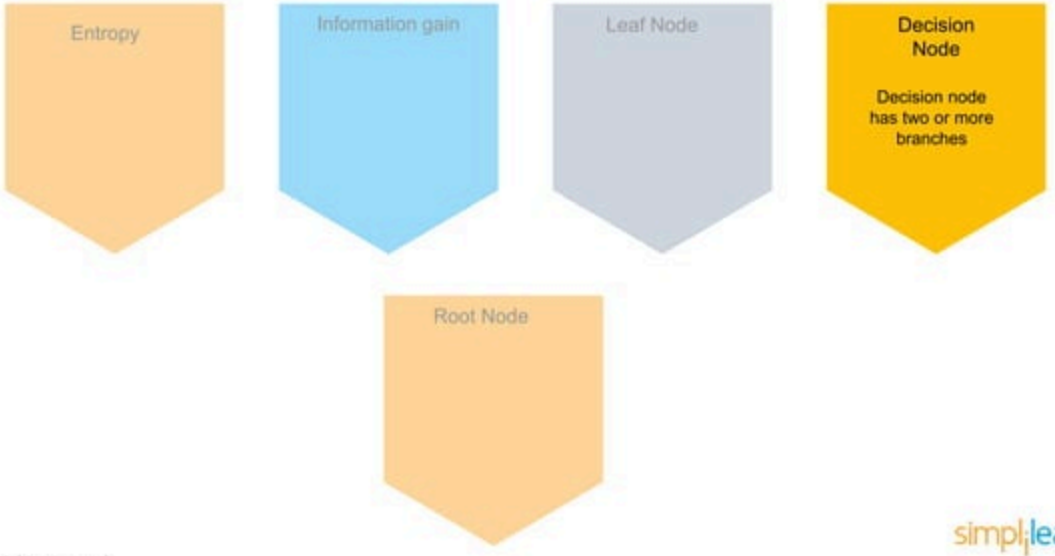

simplilearn

# Decision Tree - Important Terms




simplilearn

# Decision Tree - Important Terms



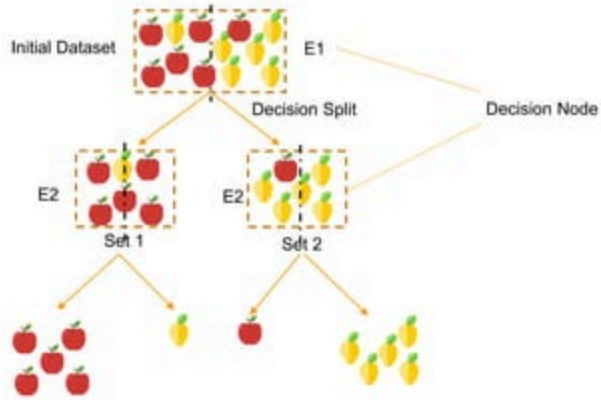



simplilearn

# Decision Tree - Important Terms

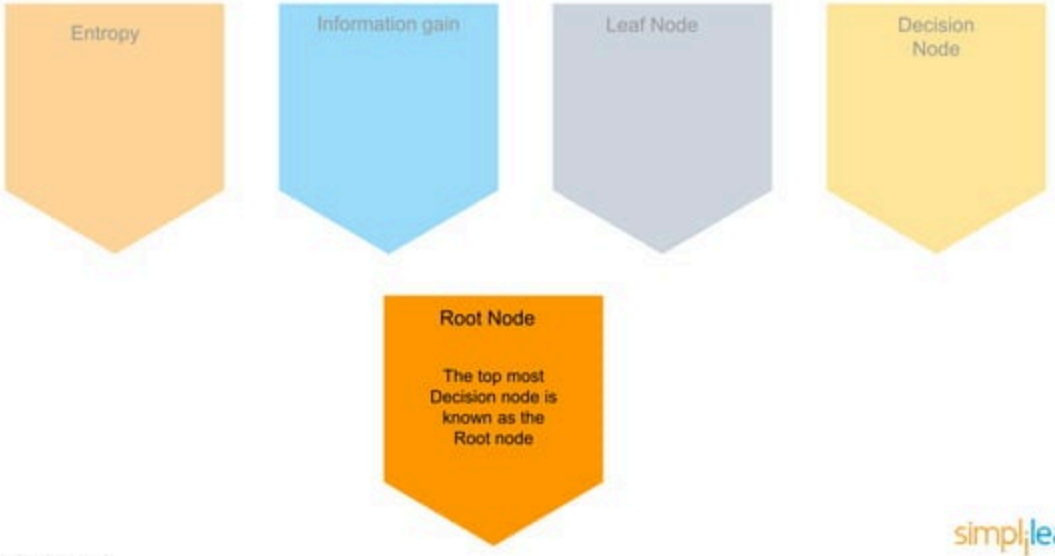
**Decision Node**

Decision node has two or more branches




simplilearn

# Decision Tree - Important Terms



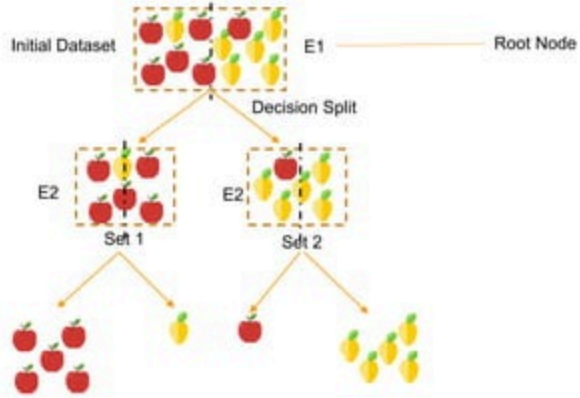



simplilearn

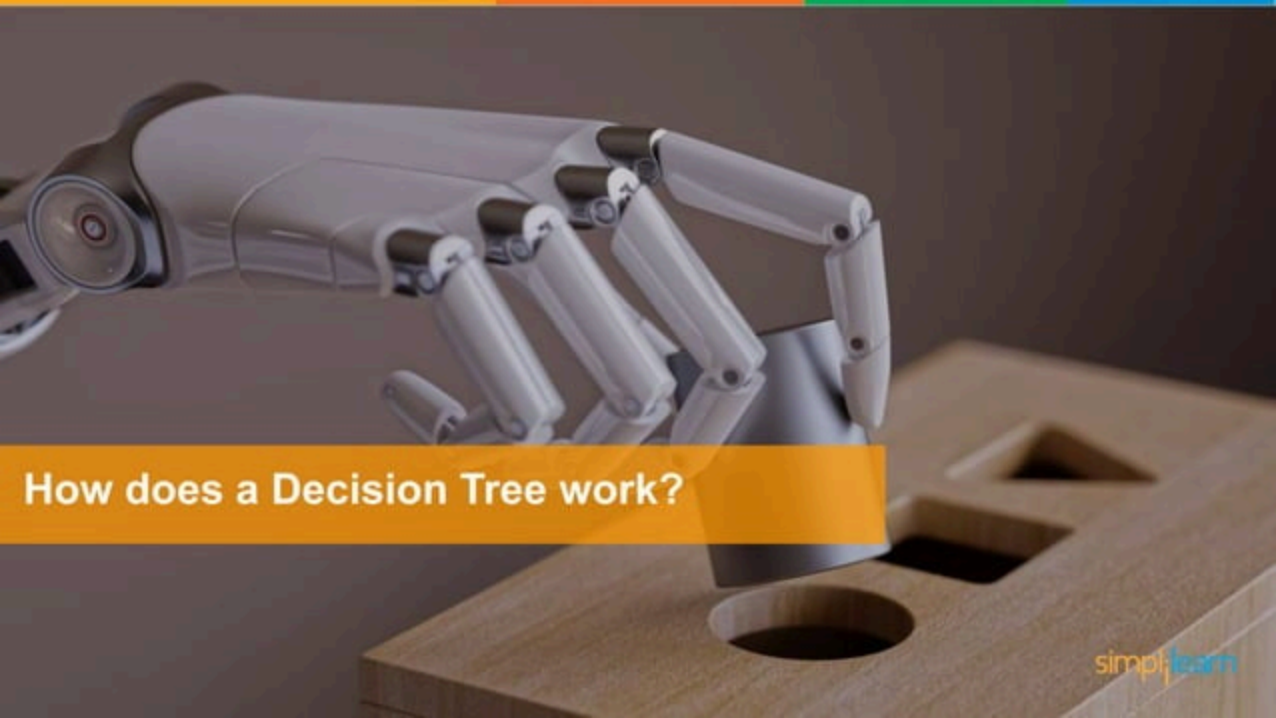
# Decision Tree - Important Terms

**Root Node**

The top most Decision node is known as the Root node




simplilearn


How does a Decision Tree work?
simplilearn

# How does a Decision Tree work?

# How does a Decision Tree work?

# How does a Decision Tree work?

**Problem statement**

To classify the different types of fruits in the bowl based on different features

# How does a Decision Tree work?

**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

# How does a Decision Tree work?

**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 |

simplilearn

# How does a Decision Tree work?

**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 does a Decision Tree work?

**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

# How does a Decision Tree work?

Now we will try to choose a condition that gives us the highest gain

# How does a Decision Tree work?

Now we will try to choose a condition that gives us the highest gain

We will do that by splitting the data using each condition and checking the gain that we get out them.

# How does a Decision Tree work?

We will do that by splitting the data using each condition and checking the gain that we get out them.

The condition that gives us the highest gain will be used to make the first split

# How does a Decision Tree work?

**Conditions**

Color== purple?

Diameter=3

Color== Yellow?

Color== Red?

Diameter=1

**Training Dataset**

| Color | Diameter | Label |
|---|---|---|
| Red | 3 | Apple |
| Yellow | 3 | Lemon |
| purple | 1 | Grapes |
| Red | 3 | Apple |
| Yellow | 3 | Lemon |
| purple | 1 | Grapes |

# How does a Decision Tree work?

**Conditions**

Color== purple?

Diameter=3

Color== Yellow?

Color== Red?

Diameter=1

Let's say this condition gives us the maximum gain

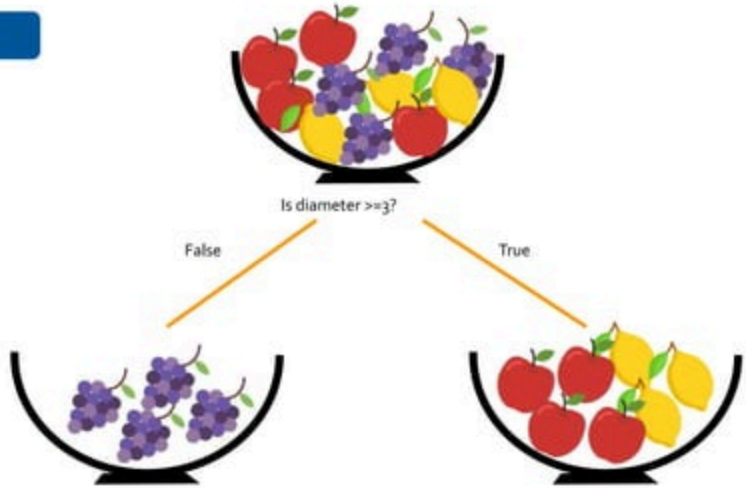
**Training Dataset**

| Color | Diameter | Label |
|---|---|---|
| Red | 3 | Apple |
| Yellow | 3 | Lemon |
| purple | 1 | Grapes |
| Red | 3 | Apple |
| Yellow | 3 | Lemon |
| purple | 1 | Grapes |

simplilearn

# How does a Decision Tree work?

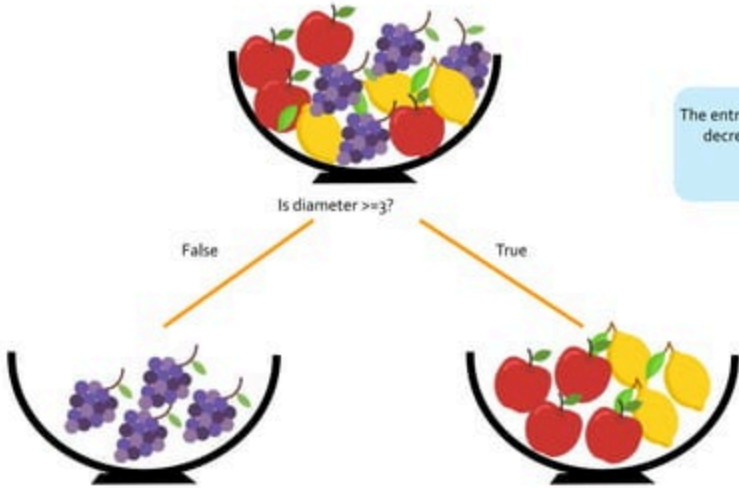
We split the data






simplilearn

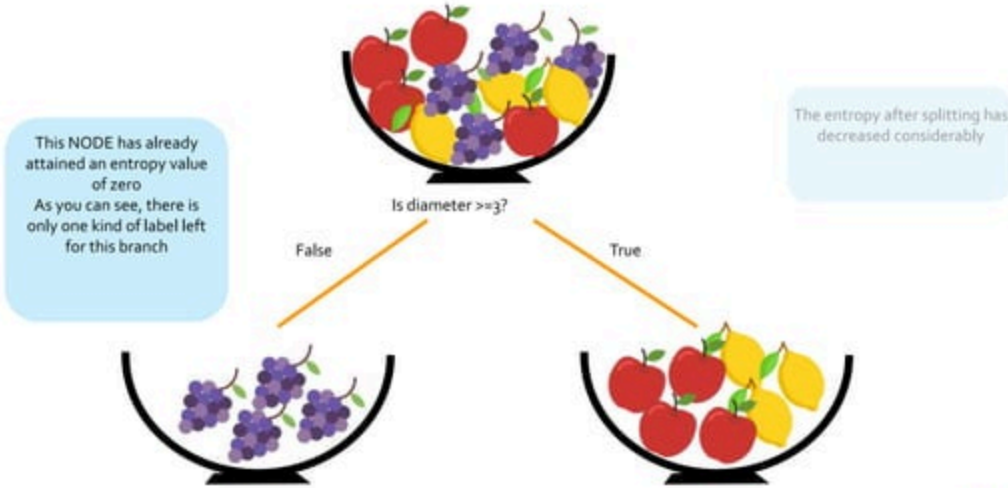
# How does a Decision Tree work?




The entropy after splitting has decreased considerably

simplilearn

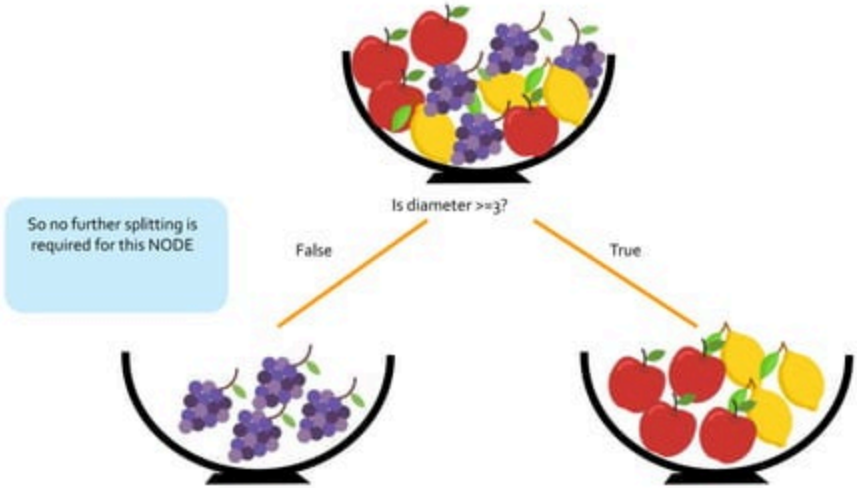
# How does a Decision Tree work?






simplilearn

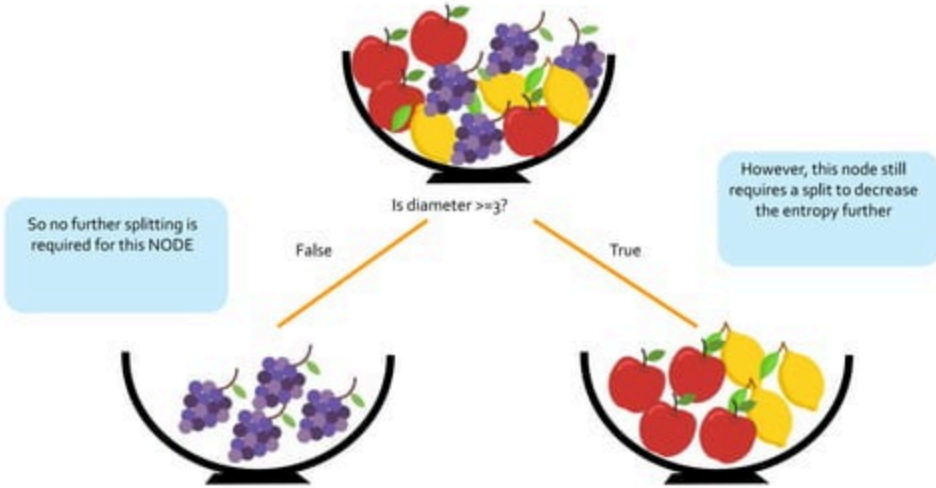
# How does a Decision Tree work?



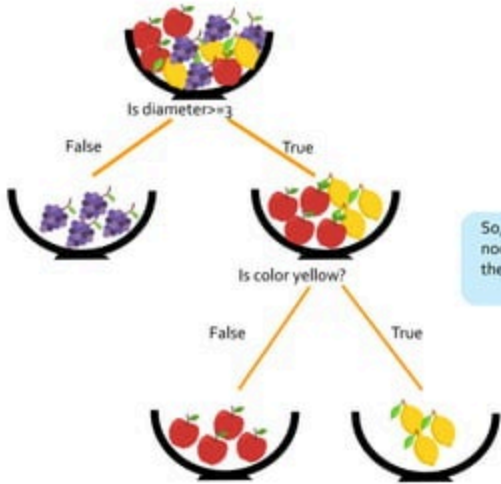



simplilearn

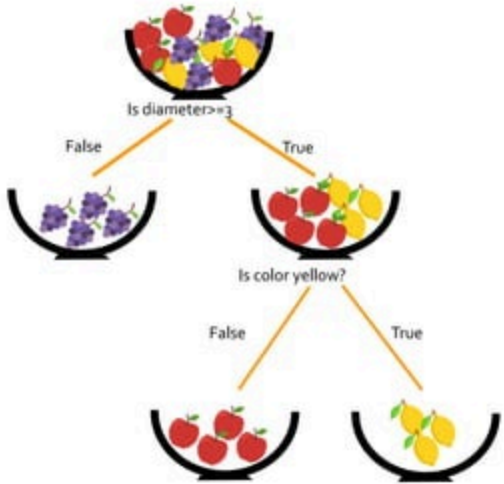
# How does a Decision Tree work?



# How does a Decision Tree work?



# How does a Decision Tree work?

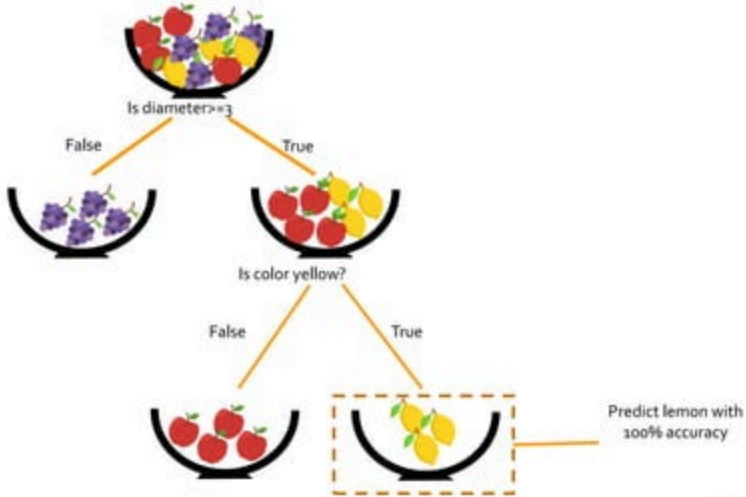



So the entropy in this case is now zero

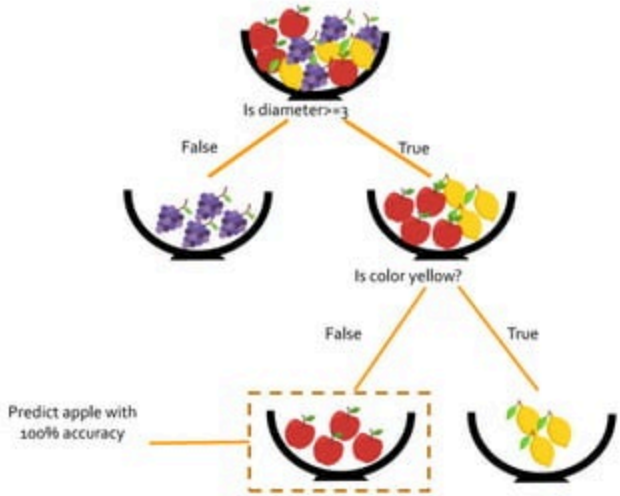


simplilearn

# How does a Decision Tree work?



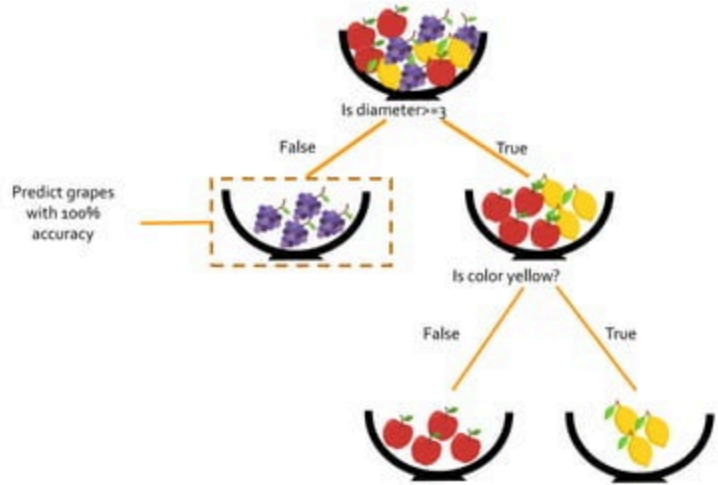
# How does a Decision Tree work?





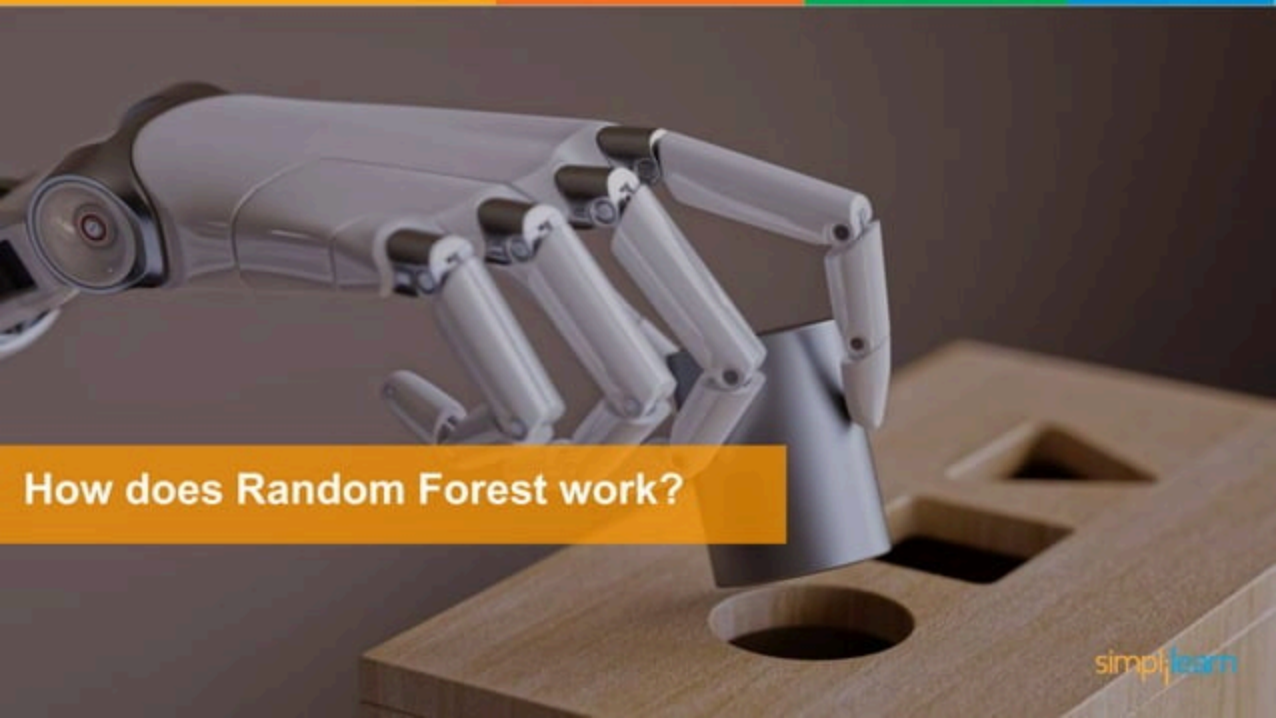

simplilearn

# How does a Decision Tree work?



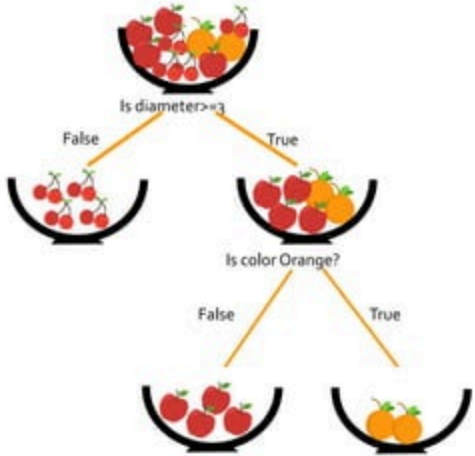



simplilearn


How does Random Forest work?
simplilearn

# How does a Random Forest work?

Let this be Tree 1

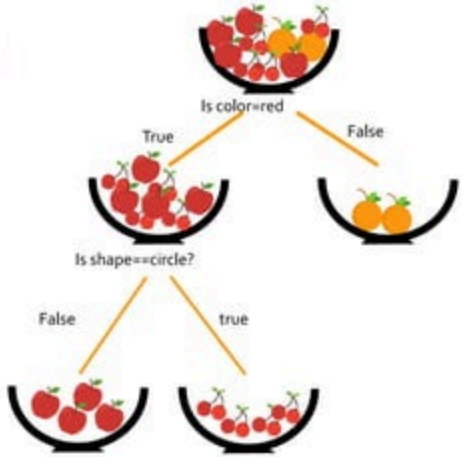





simplilearn

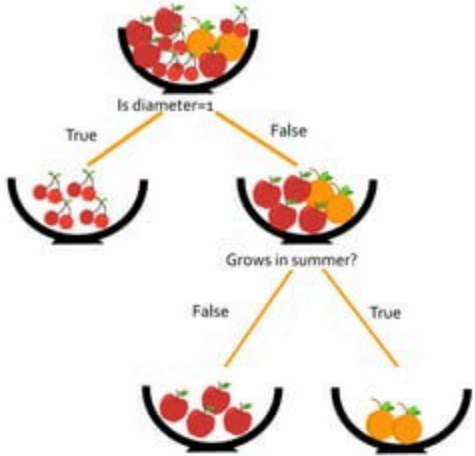
# How does a Random Forest work?

Let this be Tree 2



# How does a Random Forest work?

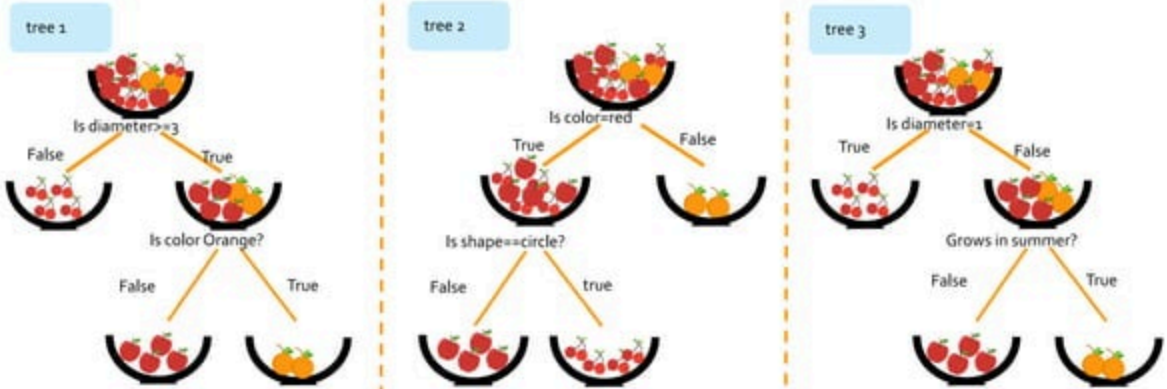
Let this be Tree 3






simplilearn

# How does a Random Forest work?



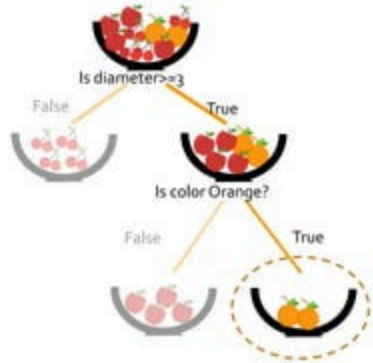
# How does a Random Forest work?

Now Lets try to classify this fruit

# How does a Random Forest work?

Tree 1 classifies it as an orange

Diameter = 3
Colour = orange
Grows in summer = yes
SHAPE = CIRCLE



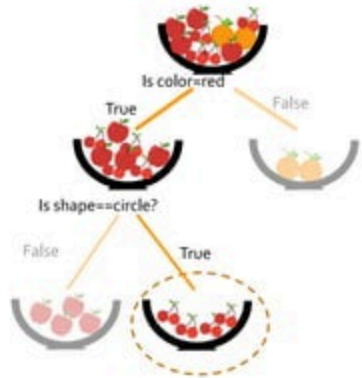



simplilearn

# How does a Random Forest work?

Tree 2 classifies it as cherries

Diameter = 3
Colour = orange
Grows in summer = yes
SHAPE = CIRCLE



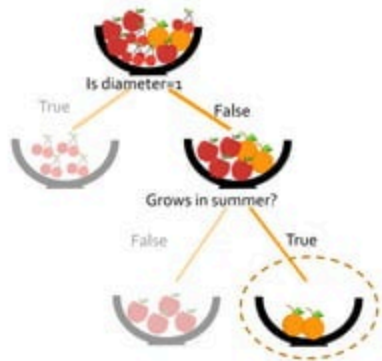



simplilearn

# How does a Random Forest work?

Tree 3 classifies it as orange

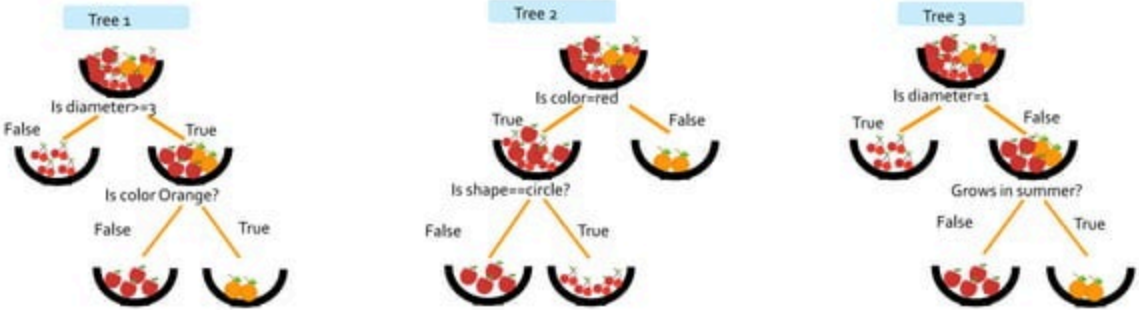
Diameter = 3
Colour = orange
Grows in summer = yes
SHAPE = CIRCLE






simplilearn

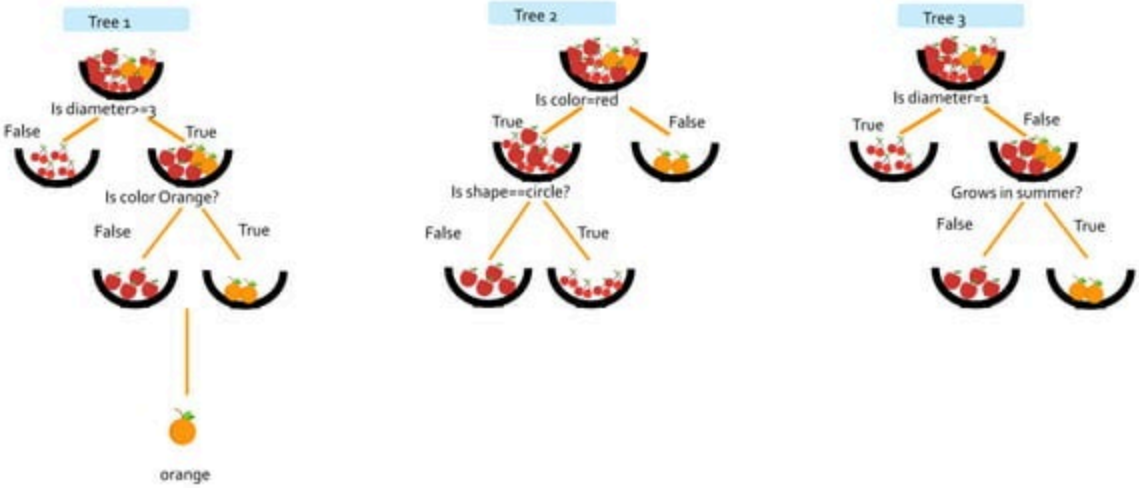
# How does a Random Forest work?






simplilearn

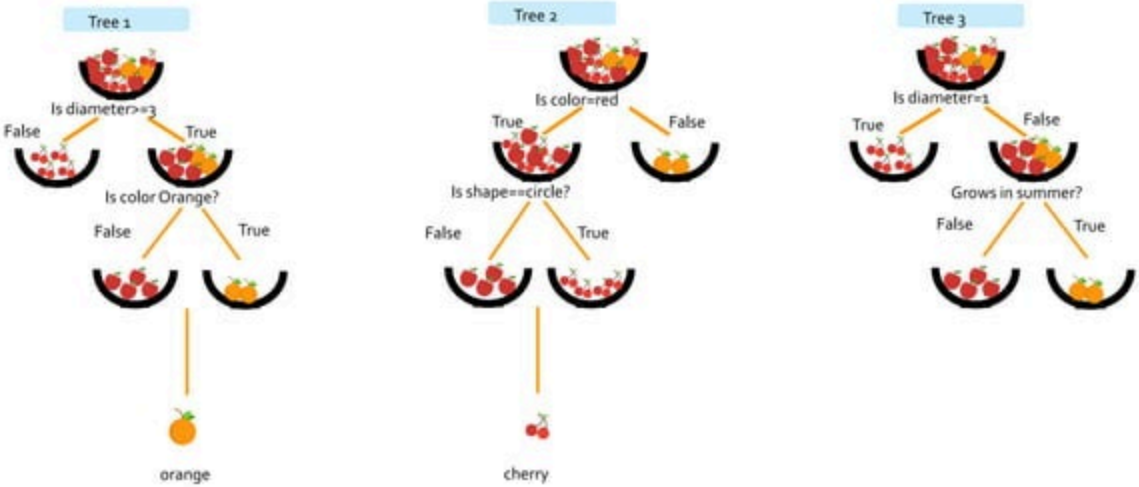
# How does a Random Forest work?






simplilearn

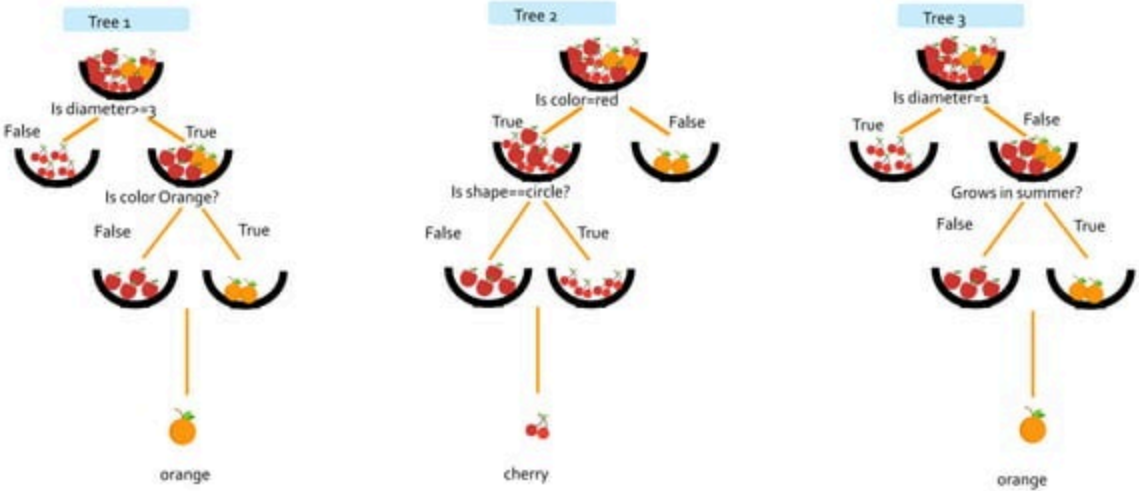
# How does a Random Forest work?






simplilearn

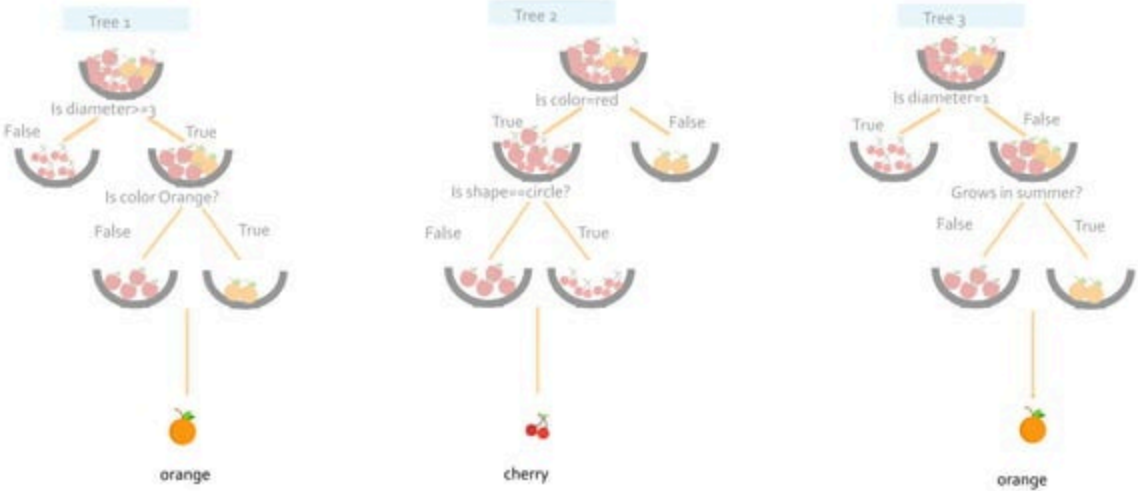
# How does a Random Forest work?






simplilearn

# How does a Random Forest work?






simplilearn

# How does a Random Forest work?

# How does a Random Forest work?

So the fruit is classified as an orange

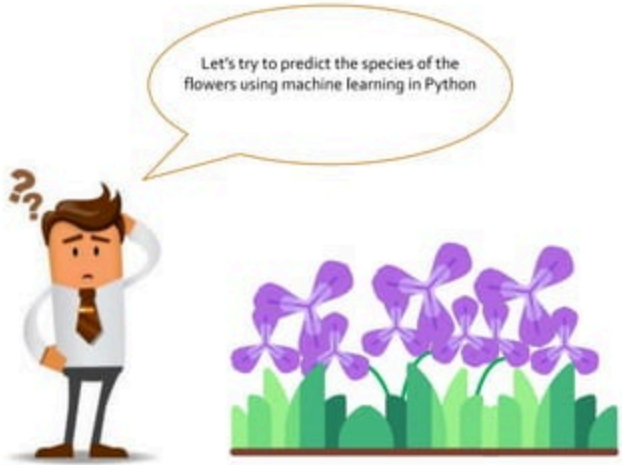
# How does a Random Forest work?

So the fruit is classified as an orange

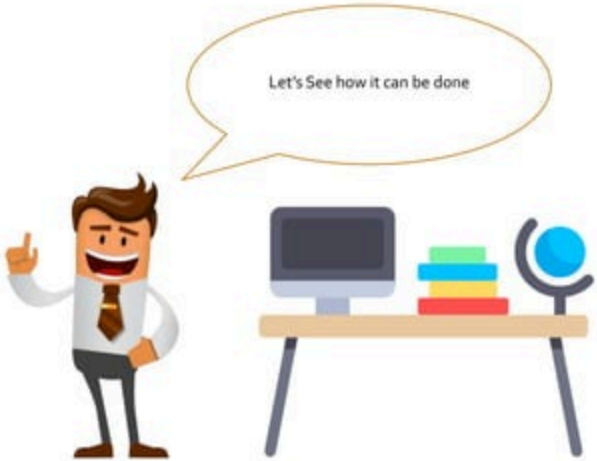

Use Case – Iris Flower Analysis
simplilearn

# Use Case - Problem Statement

# Use Case - Problem Statement



# Use Case - Implementation



# Use Case - Implementation

```
# Loading the library with the iris dataset
from sklearn.datasets import load_iris

# Loading scikit's random forest classifier library
from sklearn.ensemble import RandomForestClassifier

# Loading pandas
import pandas as pd

# Loading numpy
import numpy as np

# Setting random seed
np.random.seed(0)
```

# Use Case - Implementation

```
# Creating an object called iris with the iris data
iris = load_iris()

# Creating a dataframe with the four feature variables
df = pd.DataFrame(iris.data, columns=iris.feature_names)

# Viewing the top 5 rows
df.head()
```

Out[2]:

| | sepal length (cm) | sepal width (cm) | petal length (cm) | petal width (cm) |
|---|---|---|---|---|
| 0 | 5.1 | 3.5 | 1.4 | 0.2 |
| 1 | 4.9 | 3.0 | 1.4 | 0.2 |
| 2 | 4.7 | 3.2 | 1.3 | 0.2 |
| 3 | 4.6 | 3.1 | 1.5 | 0.2 |
| 4 | 5.0 | 3.6 | 1.4 | 0.2 |

# Use Case - Implementation

```
# Adding a new column for the species name
df['species'] = pd.Categorical.from_codes(iris.target,
iris.target_names)

# Viewing the top 5 rows
df.head()
```

Out[3]:

| | sepal length (cm) | sepal width (cm) | petal length (cm) | petal width (cm) | species |
|---|---|---|---|---|---|
| 0 | 5.1 | 3.5 | 1.4 | 0.2 | setosa |
| 1 | 4.9 | 3.0 | 1.4 | 0.2 | setosa |
| 2 | 4.7 | 3.2 | 1.3 | 0.2 | setosa |
| 3 | 4.6 | 3.1 | 1.5 | 0.2 | setosa |
| 4 | 5.0 | 3.6 | 1.4 | 0.2 | setosa |

# Use Case - Implementation

```
# Creating Test and Train Data
df['is_train'] = np.random.uniform(0, 1, len(df)) <= .75

# View the top 5 rows
df.head()
```

Out[4]:

| | sepal length (cm) | sepal width (cm) | petal length (cm) | petal width (cm) | species | is_train |
|---|---|---|---|---|---|---|
| 0 | 5.1 | 3.5 | 1.4 | 0.2 | setosa | True |
| 1 | 4.9 | 3.0 | 1.4 | 0.2 | setosa | True |
| 2 | 4.7 | 3.2 | 1.3 | 0.2 | setosa | True |
| 3 | 4.6 | 3.1 | 1.5 | 0.2 | setosa | True |
| 4 | 5.0 | 3.6 | 1.4 | 0.2 | setosa | True |

simplilearn

# Use Case - Implementation

```
# Creating dataframes with test rows and training rows
train, test = df[df['is_train']==True], df[df['is_train']==False]

# Show the number of observations for the test and training dataframes
print('Number of observations in the training data:', len(train))
print('Number of observations in the test data:',len(test))
```

```
Number of observations in the training data: 118
Number of observations in the test data: 32
```

# Use Case - Implementation

```
# Create a list of the feature column's names
features = df.columns[:4]

# View features
features
```

```
Out[7]: Index(['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)',
               'petal width (cm)'],
              dtype='object')
```

# Use Case - Implementation

```
# Converting each species name into digits
y = pd.factorize(train['species'])[0]

# Viewing target
y
```

```
Out[8]: array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1,
               1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
               1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
               2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
               2, 2, 2], dtype=int64)
```



simplilearn

# Use Case - Implementation

```
# Creating a random forest Classifier.
clf = RandomForestClassifier(n_jobs=2, random_state=0)

# Training the classifier
clf.fit(train[features], y)
```

```
Out[9]: RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
                               max_depth=None, max_features='auto', max_leaf_nodes=None,
                               min_impurity_decrease=0.0, min_impurity_split=None,
                               min_samples_leaf=1, min_samples_split=2,
                               min_weight_fraction_leaf=0.0, n_estimators=10, n_jobs=2,
                               oob_score=False, random_state=0, verbose=0, warm_start=False)
```

# Use Case - Implementation

```
# Applying the trained Classifier to the test
clf.predict(test[features])
```

```
Out[10]: array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 2, 2, 1, 1, 2, 2, 2,
                2, 2, 2, 2, 2, 2, 2, 2, 2], dtype=int64)
```

# Use Case - Implementation

```
# Viewing the predicted probabilities of the first 10
observations
clf.predict_proba(test[features])[0:10]
```

```
Out[11]: array([[ 1. ,  0. ,  0. ],
                [ 1. ,  0. ,  0. ],
                [ 1. ,  0. ,  0. ],
                [ 1. ,  0. ,  0. ],
                [ 1. ,  0. ,  0. ],
                [ 1. ,  0. ,  0. ],
                [ 1. ,  0. ,  0. ],
                [ 0.9,  0.1,  0. ],
                [ 1. ,  0. ,  0. ],
                [ 1. ,  0. ,  0. ]])
```

# Use Case - Implementation

```
# mapping names for the plants for each predicted plant class
preds = iris.target_names[clf.predict(test[features])]



# View the PREDICTED species for the first five
observations
preds[0:5]
```

```
Out[13]: array(['setosa', 'setosa', 'setosa', 'setosa', 'setosa'],
               dtype='<U10')
```

# Use Case - Implementation

```
# Viewing the ACTUAL species for the first five observations
test['species'].head()
```

```
Out[14]: 7     setosa
         8     setosa
         10    setosa
         13    setosa
         17    setosa
         Name: species, dtype: category
         Categories (3, object): [setosa, versicolor, virginica]
```

# Use Case - Implementation

```
# Creating confusion matrix
pd.crosstab(test['species'], preds, rownames=['Actual Species'],
colnames=['Predicted Species'])
```

Out[15]:

| Actual Species \ Predicted Species | setosa | versicolor | virginica |
|---|---|---|---|
| setosa | 13 | 0 | 0 |
| versicolor | 0 | 5 | 2 |
| virginica | 0 | 0 | 12 |

# Use Case - Implementation

Out[15]:

| Predicted Species / Actual Species | setosa | versicolor | virginica |
|---|---|---|---|
| setosa | 13 | 0 | 0 |
| versicolor | 0 | 5 | 2 |
| virginica | 0 | 0 | 12 |

Total number of predictions = 32



simplilearn

# Use Case - Implementation

Out[15]:

| Predicted Species | setosa | versicolor | virginica |
|---|---|---|---|
| Actual Species | | | |
| setosa | 13 | 0 | 0 |
| versicolor | 0 | 5 | 2 |
| virginica | 0 | 0 | 12 |

Number of accurate predictions = 30



simplilearn

# Use Case - Implementation

Out[15]:

| Predicted Species / Actual Species | setosa | versicolor | virginica |
|---|---|---|---|
| setosa | 13 | 0 | 0 |
| versicolor | 0 | 5 | 2 |
| virginica | 0 | 0 | 12 |

Number of inaccurate predictions = 2

# Use Case - Implementation

Out[15]:

| Predicted Species / Actual Species | setosa | versicolor | virginica |
|---|---|---|---|
| setosa | 13 | 0 | 0 |
| versicolor | 0 | 5 | 2 |
| virginica | 0 | 0 | 12 |

Model Accuracy

$$\frac{30}{32} \times 100 = 93$$

# Use Case - Implementation

Out[15]:

| Predicted Species / Actual Species | setosa | versicolor | virginica |
|---|---|---|---|
| setosa | 13 | 0 | 0 |
| versicolor | 0 | 5 | 2 |
| virginica | 0 | 0 | 12 |

Model Accuracy

$$\frac{30}{32} \times 100 = 93$$

So the model accuracy is 93%

# Key takeaways







simplilearn


THANK YOU
For more information, visit
www.simplilearn.com
simplilearn