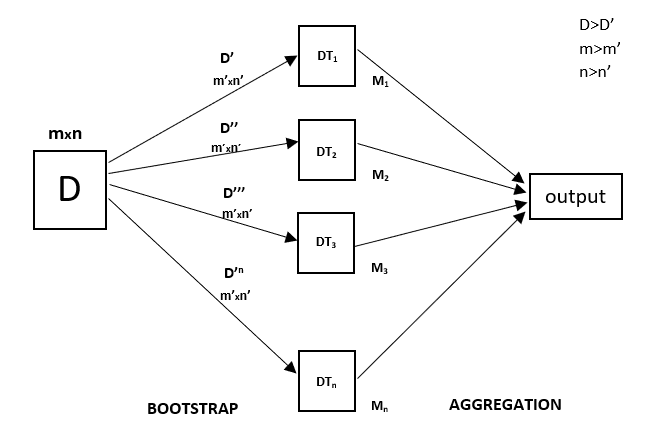
**Random Forest Regression Intuition**

Every decision tree has high variance, but when we combine all of them together in parallel then the resultant variance is low as each decision tree gets perfectly trained on that particular sample data, and hence the output doesn’t depend on one decision tree but on multiple decision trees. In the case of a classification problem, the final output is taken by using the majority voting classifier. In the case of a regression problem, the final output is the mean of all the outputs. This part is called **Aggregation**.



Random Forest is an ensemble technique capable of performing both regression and classification tasks with the use of multiple decision trees and a technique called Bootstrap and Aggregation, commonly known as **bagging**. The basic idea behind this is to combine multiple decision trees in determining the final output rather than relying on individual decision trees.   
Random Forest has multiple decision trees as base learning models. We randomly perform row sampling and feature sampling from the dataset forming sample datasets for every model. This part is called Bootstrap.

We need to approach the Random Forest regression technique like any other machine learning technique

* Design a specific question or data and get the source to determine the required data.
* Make sure the data is in an accessible format else convert it to the required format.
* Specify all noticeable anomalies and missing data points that may be required to achieve the required data.
* Create a machine learning model
* Set the baseline model that you want to achieve
* Train the data machine learning model.
* Provide an insight into the model with test data
* Now compare the performance metrics of both the test data and the predicted data from the model.
* If it doesn’t satisfy your expectations, you can try improving your model accordingly or dating your data, or using another data modeling technique.
* At this stage, you interpret the data you have gained and report accordingly.

You will be using a similar sample technique in the below example.

Below is a step-by-step sample implementation of Random Forest Regression**.**

**Implementation:**

Python3

*# Importing the libraries*

**import** **numpy** **as** **np**

**import** **matplotlib.pyplot** **as** **plt**

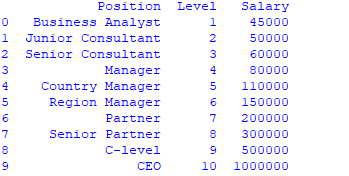
**import** **pandas** **as** **pd**

**Step 2:** Import and print the dataset

Python3

data = pd.read\_csv('Salaries.csv')

print(data)



**Step 3:** Select all rows and column 1 from dataset to x and all rows and column 2 as y

# the coding was not shown which is like that

x= df.iloc [:, : -1] # ” : ” means it will select all rows,    “: -1 ” means that it will ignore last column  
y= df.iloc [:, -1 :] # ” : ” means it will select all rows,    “-1 : ” means that it will ignore all columns except the last one

# the “iloc()” function enables us to select a particular cell of the dataset, that is, it helps us select a value that belongs to a particular row or column from a set of values of a data frame or dataset.

**Step 4:** Fit Random forest regressor to the dataset

Python3

*# Fitting Random Forest Regression to the dataset*

*# import the regressor*

**from** **sklearn.ensemble** **import** RandomForestRegressor

*# create regressor object*

regressor = RandomForestRegressor(n\_estimators = 100, random\_state = 0)

*# fit the regressor with x and y data*

regressor.fit(x, y)

