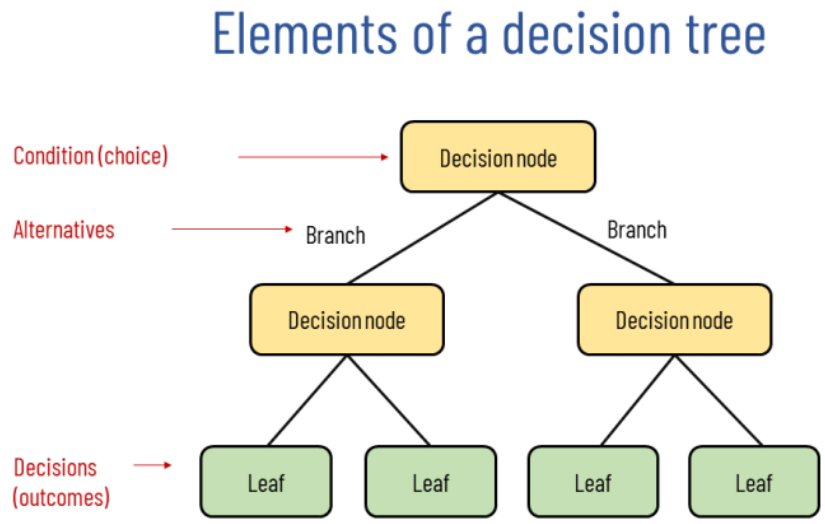
Understanding Decision Trees:

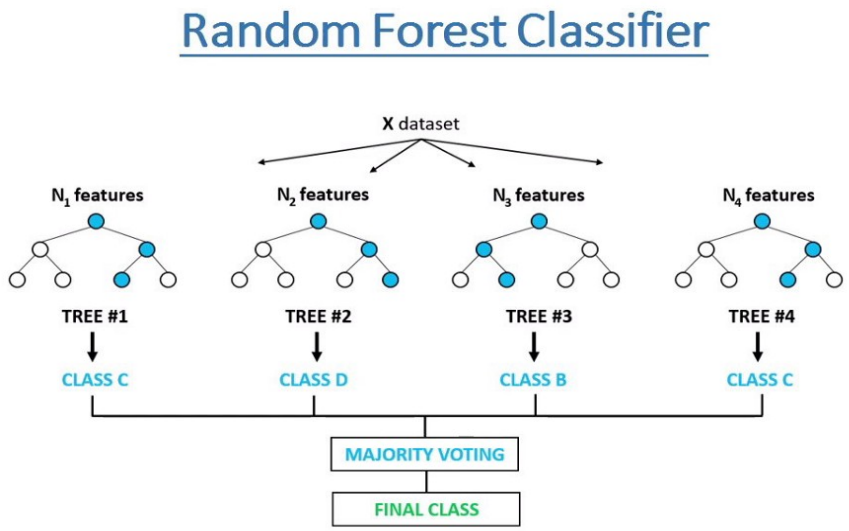


(Because a random forest classifier is simply a large number of decision trees, it's important that you first understand what a decision tree is)

Links on decision trees:

- Decision and Classification Trees, Clearly Explained!!!
- Regression Trees, Clearly Explained!!!
- https://scikit-learn.org/stable/modules/tree.html
- https://www.geeksforgeeks.org/decision-tree-implementation-python/?ref=lbp

Understanding the Random Forest Classifier:



Links on random forest classifiers:

- Random Forest Algorithm Clearly Explained!
- StatQuest: Random Forests Part 1 - Building, Using and Evaluating
- https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html
- https://www.geeksforgeeks.org/random-forest-classifier-using-scikit-learn/

Implementing the Random Forest Classifier on Diabetes data:

Implementation of Random Forest Classiffier

Import the necessary libraries

In [79]:
```
import pandas as pd
from sklearn.metrics import confusion_matrix
from sklearn.metrics import matthews_corrcoef
# metrics are used to find accuracy or error
from sklearn import metrics
```

Load in the diabetes dataset

In [ ]:
```
data = pd.read_csv('diabetes.csv')
```

Split target feature from the rest of the data and run traintestsplit

In [32]:
```
X = data.copy().drop(columns='Outcome')
y = data['Outcome'].copy()
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=0)
```

Set hyper paramaters, fit the model on the training data, and then run predictions on the test data

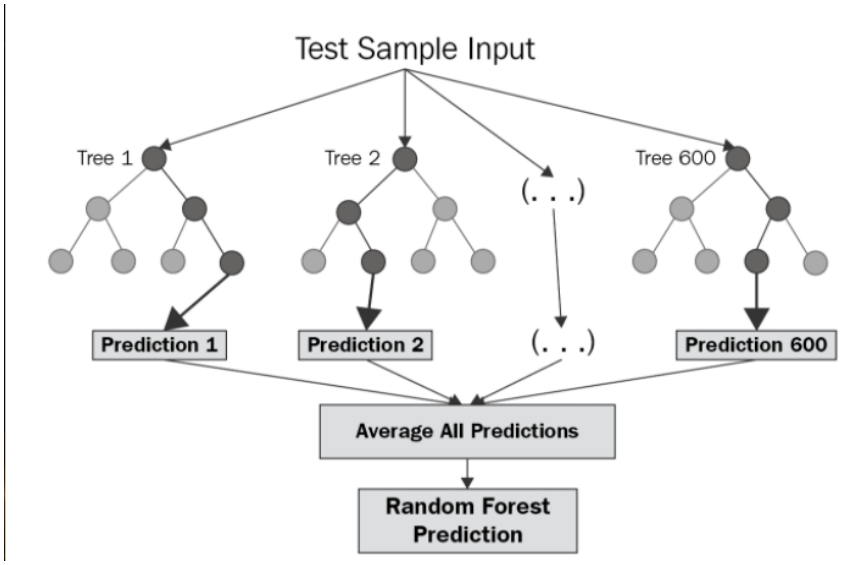
In [91]:
```
clf = RandomForestClassifier(n_estimators = 50, max_features= 'log2', bootstrap = True)

# Training the model on the training dataset
# fit function is used to train the model using the training sets as parameters
clf.fit(X_train, y_train)

# performing predictions on the test dataset
y_pred = clf.predict(X_test)
```

Check R-squared

In [92]:
```
# using metrics module for accuracy calculation
print("R-squared: ", clf.score(X_test,y_test))
```

```
R-squared:  0.7864583333333334
```

Understanding the Random Forest Regression:



Links on random forest Regression:

- https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html
- https://www.geeksforgeeks.org/random-forest-regression-in-python/?ref=lbp

Regression tree

Importing libraries

In [ ]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
```

Read the Cars93 csv

In [ ]:
```
cars = pd.read_csv('Cars93.csv')
```

Clean the cars data and get dummies

In [ ]:
```
rowcleancars = cars.dropna()
ccars = rowcleancars.drop(columns=['Unnamed: 0'])
ccars = pd.get_dummies(ccars,drop_first=True)
ccars.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 82 entries, 0 to 92
Columns: 225 entries, Min.Price to Make_Volvo 850
dtypes: float64(7), int64(11), uint8(207)
memory usage: 28.7 KB
```

Set up the testing and training data

In [ ]:
```
X = ccars.drop(columns=['MPG.highway'])
y = ccars['MPG.highway']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=0)
```

Set up and train the regressor

In [ ]:
```
regressor = RandomForestRegressor(n_estimators = 100, random_state = 0)
regressor.fit(X_train, y_train)
```

Out[ ]:
```
RandomForestRegressor(random_state=0)
```

Performing prediction and checking r squared

In [ ]:
```
pred = regressor.predict(X_test)
print("R-squared: ",  regressor.score(X_test,y_test))
```

```
R-squared:  0.8179437894454142
```