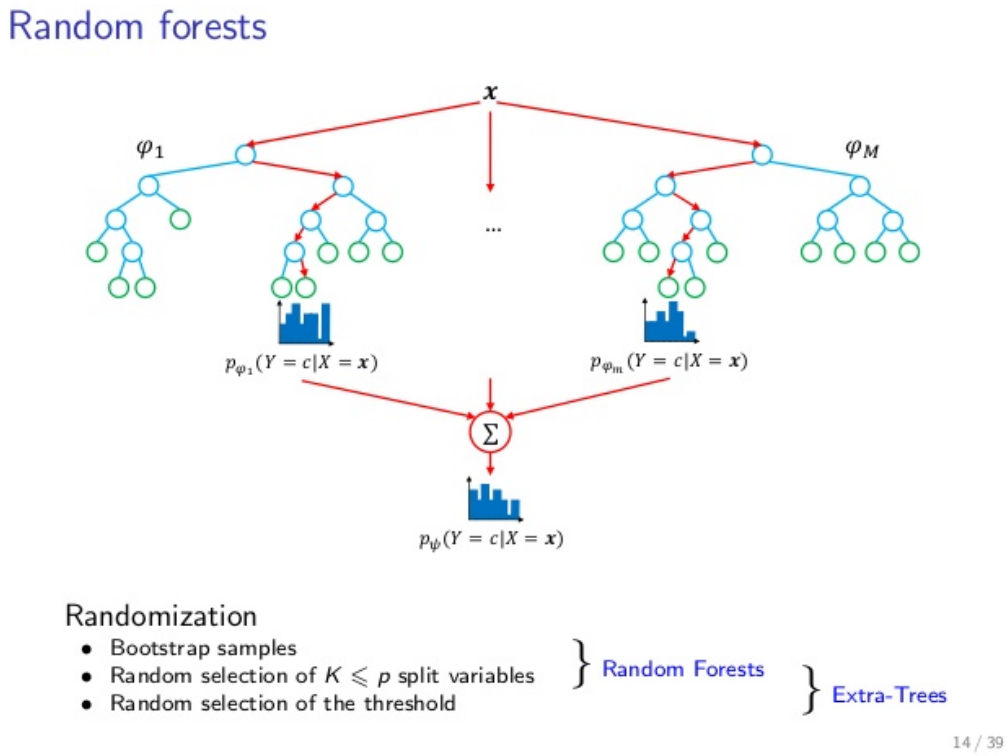
Random Forest

Random forest is popular **ensemble learning method**, which means group learning. Here model predicts based on the multiple models instead of single model. It is go to model for classification and regression. It is also called **bagging or bootstrap aggregating**. It is group of decision trees for prediction to achieve generalised model predictions.
Below image depicts a random forest:



Like the above figure, random forest creates multiple decision trees from subset of dataset. Every tree makes predictions and **average or majority voting** of predictions made is the ultimate prediction of the random forest model. It avoids the problem overfitting in normal decision tree.
Number of trees to be used can be decided by user and the efficiency of the predictions.
It chooses feature randomly instead of greedy approach of choosing best feature, this technique or approach is called as **feature bagging**. This randomness adds diversity to the model and makes it robust learning technique. Despite it's robustness, it is slow method. And random forest don't train well for small datasets (not suitable).

The dataset tells about the social network users and purchase of cars from ads displayed on the social network website.

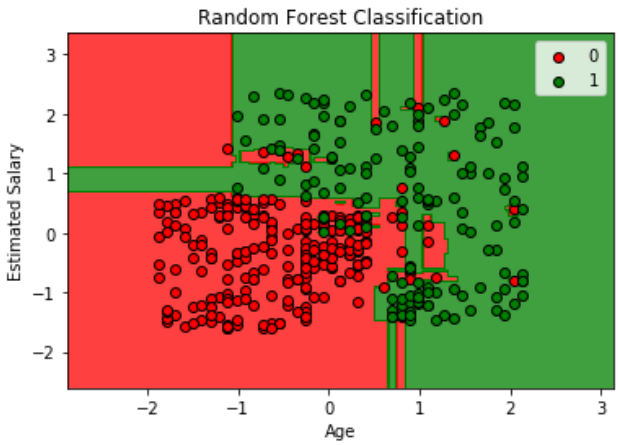
In [1]:
```
# Random Forest Classification

# Importing the libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

# Importing the dataset
dataset = pd.read_csv('Social_Network_Ads.csv')
X = dataset.iloc[:, [2, 3]].values
y = dataset.iloc[:, 4].values
```

In [2]:
```
# Feature Scaling
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X = sc.fit_transform(X)
```

In [3]:
```
# Fitting Random Forest Classification to the Training set
from sklearn.ensemble import RandomForestClassifier
classifier = RandomForestClassifier(n_estimators = 10, criterion = 'entr
opy', random_state = 0)
classifier.fit(X, y)
```

Out[3]:
```
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='entr
opy',
            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=1,
            oob_score=False, random_state=0, verbose=0, warm_start=False)
```

In [4]:
```
from matplotlib.colors import ListedColormap
X_set, y_set = X, y
X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 1, stop = X_s
et[:, 0].max() + 1, step = 0.01),
                     np.arange(start = X_set[:, 1].min() - 1, stop = X_s
et[:, 1].max() + 1, step = 0.01))
plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(), X2.ravel()
]).T).reshape(X1.shape),
             alpha = 0.75, cmap = ListedColormap(('red', 'green')))
plt.xlim(X1.min(), X1.max())
plt.ylim(X2.min(), X2.max())
for i, j in enumerate(np.unique(y_set)):
    plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1],
                c = ListedColormap(('red', 'green'))(i), label = j, edge
color='black')
plt.title('Random Forest Classification')
plt.xlabel('Age')
plt.ylabel('Estimated Salary')
plt.legend()
plt.show()
```



Decision trees constructed for example can observed here:https://drive.google.com/open?id=1tfdqGae61jCRZ1NkJgRv9nqK38f3cQ9 (https://drive.google.com/open?id=1tfdqGae61jCRZ1NkJgRv9nqK38f3cQ9)

Further reading

1. https://www.kdnuggets.com/2017/10/random-forests-explained.html (https://www.kdnuggets.com/2017/10/random-forests-explained.html)
2. https://www.datasciencecentral.com/profiles/blogs/random-forests-explained-intuitively (https://www.datasciencecentral.com/profiles/blogs/random-forests-explained-intuitively)