In [1]:

```
import pandas as pd
```

In [3]:

```
import matplotlib.pyplot as plt
%matplotlib inline
```

In [10]:

```
from sklearn.datasets import load_iris
from sklearn import tree   # used for doing the prediction
from sklearn.ensemble import RandomForestClassifier

clf = RandomForestClassifier(n_estimators=20, random_state=0) # by default it take 100 tree
iris=load_iris()
x=iris.data
y=iris.target

clf=clf.fit(x,y)
```

In [11]:

```
iris
```

Out[11]:

```
{'data': array([[5.1, 3.5, 1.4, 0.2],
        [4.9, 3. , 1.4, 0.2],
        [4.7, 3.2, 1.3, 0.2],
        [4.6, 3.1, 1.5, 0.2],
        [5. , 3.6, 1.4, 0.2],
        [5.4, 3.9, 1.7, 0.4],
        [4.6, 3.4, 1.4, 0.3],
        [5. , 3.4, 1.5, 0.2],
        [4.4, 2.9, 1.4, 0.2],
        [4.9, 3.1, 1.5, 0.1],
        [5.4, 3.7, 1.5, 0.2],
        [4.8, 3.4, 1.6, 0.2],
        [4.8, 3. , 1.4, 0.1],
        [4.3, 3. , 1.1, 0.1],
        [5.8, 4. , 1.2, 0.2],
        [5.7, 4.4, 1.5, 0.4],
        [5.4, 3.9, 1.3, 0.4],
```

In [12]:

```
iris.target
```

Out[12]:

```
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, 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, 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, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2])
```

In [13]:

```
clf.estimators_
```

Out[13]:

```
[DecisionTreeClassifier(max_features='auto', random_state=209652396),
 DecisionTreeClassifier(max_features='auto', random_state=398764591),
 DecisionTreeClassifier(max_features='auto', random_state=924231285),
 DecisionTreeClassifier(max_features='auto', random_state=1478610112),
 DecisionTreeClassifier(max_features='auto', random_state=441365315),
 DecisionTreeClassifier(max_features='auto', random_state=1537364731),
 DecisionTreeClassifier(max_features='auto', random_state=192771779),
 DecisionTreeClassifier(max_features='auto', random_state=1491434855),
 DecisionTreeClassifier(max_features='auto', random_state=1819583497),
 DecisionTreeClassifier(max_features='auto', random_state=530702035),
 DecisionTreeClassifier(max_features='auto', random_state=626610453),
 DecisionTreeClassifier(max_features='auto', random_state=1650906866),
 DecisionTreeClassifier(max_features='auto', random_state=1879422756),
 DecisionTreeClassifier(max_features='auto', random_state=1277901399),
 DecisionTreeClassifier(max_features='auto', random_state=1682652230),
 DecisionTreeClassifier(max_features='auto', random_state=243580376),
 DecisionTreeClassifier(max_features='auto', random_state=1991416408),
 DecisionTreeClassifier(max_features='auto', random_state=1171049868),
 DecisionTreeClassifier(max_features='auto', random_state=1646868794),
 DecisionTreeClassifier(max_features='auto', random_state=2051556033)]
```

In [14]:

```
len(clf.estimators_)
```

Out[14]:

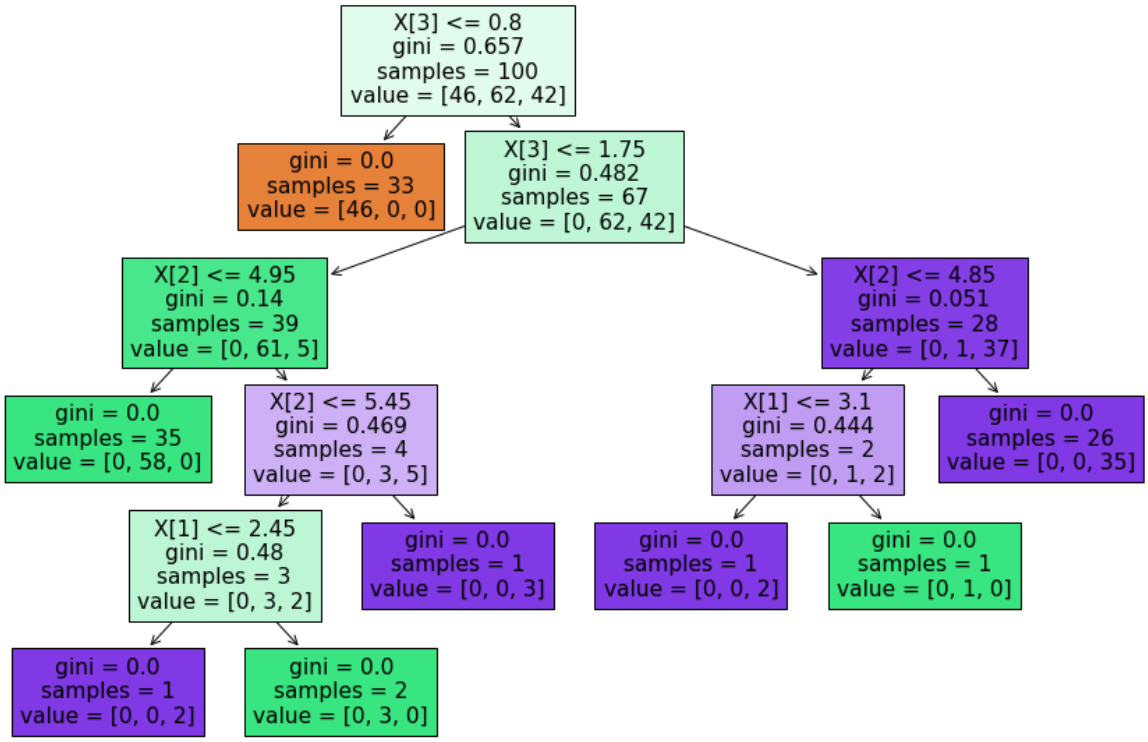
```
20
```

In [15]:

```
plt.figure(figsize=(15,10))
tree.plot_tree(clf.estimators_[1], filled = True)
```

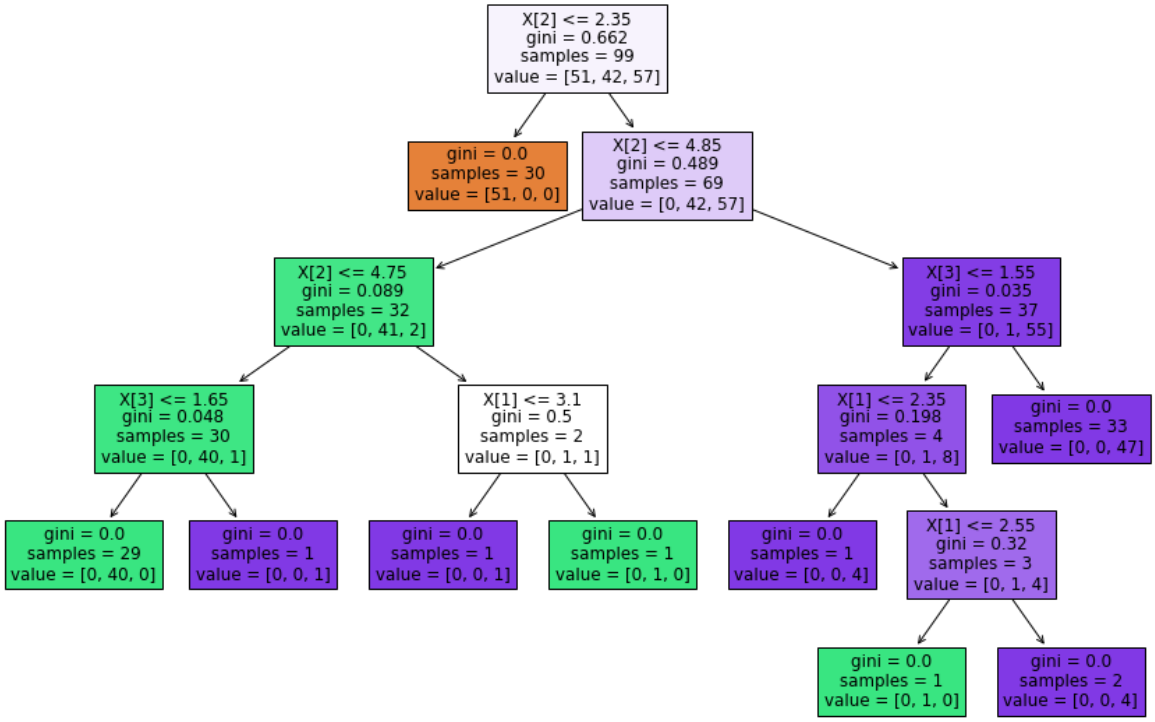
Out[15]:

```
[Text(334.8, 498.3, 'X[3] <= 0.8\ngini = 0.657\nsamples = 100\nvalue = [46,
62, 42]'),
 Text(251.10000000000002, 407.70000000000005, 'gini = 0.0\nsamples = 33\nval
ue = [46, 0, 0]'),
 Text(418.5, 407.70000000000005, 'X[3] <= 1.75\ngini = 0.482\nsamples = 67\n
value = [0, 62, 42]'),
 Text(167.4, 317.1, 'X[2] <= 4.95\ngini = 0.14\nsamples = 39\nvalue = [0, 6
1, 5]'),
 Text(83.7, 226.5, 'gini = 0.0\nsamples = 35\nvalue = [0, 58, 0]'),
 Text(251.10000000000002, 226.5, 'X[2] <= 5.45\ngini = 0.469\nsamples = 4\nv
alue = [0, 3, 5]'),
 Text(167.4, 135.89999999999998, 'X[1] <= 2.45\ngini = 0.48\nsamples = 3\nva
lue = [0, 3, 2]'),
 Text(83.7, 45.299999999999955, 'gini = 0.0\nsamples = 1\nvalue = [0, 0,
2]'),
 Text(251.10000000000002, 45.299999999999955, 'gini = 0.0\nsamples = 2\nvalu
e = [0, 3, 0]'),
 Text(334.8, 135.89999999999998, 'gini = 0.0\nsamples = 1\nvalue = [0, 0,
3]'),
 Text(669.6, 317.1, 'X[2] <= 4.85\ngini = 0.051\nsamples = 28\nvalue = [0,
1, 37]'),
 Text(585.9, 226.5, 'X[1] <= 3.1\ngini = 0.444\nsamples = 2\nvalue = [0, 1,
2]'),
 Text(502.20000000000005, 135.89999999999998, 'gini = 0.0\nsamples = 1\nvalu
e = [0, 0, 2]'),
 Text(669.6, 135.89999999999998, 'gini = 0.0\nsamples = 1\nvalue = [0, 1,
0]'),
 Text(753.3000000000001, 226.5, 'gini = 0.0\nsamples = 26\nvalue = [0, 0, 3
5]')]
```



In [16]:

```
plt.figure(figsize=(15,10))
for i in range(len(clf.estimators_)):
    tree.plot_tree(clf.estimators_[i], filled=True)
```



In [18]:

```
for i in range(len(clf.estimators_)):
    print(tree.export_text(clf.estimators_[i]))
```

```
|--- feature_3 <= 0.75
|   |--- class: 0.0
|--- feature_3 >  0.75
|   |--- feature_2 <= 4.85
|   |   |--- feature_3 <= 1.65
|   |   |   |--- class: 1.0
|   |   |--- feature_3 >  1.65
|   |   |   |--- feature_1 <= 3.00
|   |   |   |   |--- class: 2.0
|   |   |   |--- feature_1 >  3.00
|   |   |   |   |--- class: 1.0
|   |--- feature_2 >  4.85
|   |   |--- feature_0 <= 6.60
|   |   |   |--- class: 2.0
|   |   |--- feature_0 >  6.60
|   |   |   |--- feature_2 <= 5.20
|   |   |   |   |--- class: 1.0
|   |   |   |--- feature_2 >  5.20
|   |   |   |   |--- class: 2.0
```

In [ ]: