In [168]:
```
import numpy as np
import pandas as pd
from sklearn import datasets
from matplotlib import pyplot as plt
from sklearn.preprocessing import LabelEncoder
from sklearn import tree
import six
import sys
sys.modules['sklearn.externals.six'] = six
from id3 import Id3Estimator, export_text
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.metrics import accuracy_score, classification_report
from sklearn.datasets import load_iris
```

In [149]:
```
iris = datasets.load_iris()
X = iris.data
y = iris.target
```

In [150]:
```
df = pd.read_csv('PredictingSignupsTrain.csv')
df
```

Out[150]:

	Referrer	Location	Read_FAQ	Pages_Viewed	Service_Chosen
0	Slashdot	USA	Yes	Mid	None
1	Google	France	Yes	High	Premium
2	Digg	USA	Yes	High	Basic
3	Kiwitobes	France	Yes	High	Basic
4	Google	UK	No	Mid	Premium
5	(direct)	'New Zealand'	No	Low	None
6	(direct)	UK	No	Mid	Basic
7	Google	USA	No	High	Premium
8	Slashdot	France	Yes	Mid	None
9	Digg	USA	No	Mid	None
10	Google	UK	No	Mid	None
11	Kiwitobes	UK	No	Mid	None
12	Digg	'New Zealand'	Yes	Low	Basic
13	Google	UK	Yes	Mid	Basic
14	Kiwitobes	France	Yes	Mid	Basic
15	Google	UK	No	Mid	Premium
16	Digg	USA	No	Low	Basic
17	Slashdot	'New Zealand'	Yes	High	None

In [151]:
```
lb_enc = LabelEncoder()
df['Referrer'] = lb_enc.fit_transform(df['Referrer'])
df['Location'] = lb_enc.fit_transform(df['Location'])
df['Read_FAQ'] = lb_enc.fit_transform(df['Read_FAQ'])
df['Pages_Viewed'] = lb_enc.fit_transform(df['Pages_Viewed'])
df['Service_Chosen'] = lb_enc.fit_transform(df['Service_Chosen'])
df.to_csv("PredictingSignupsTrain_label.csv")
df
```

Out[151]:

	Referrer	Location	Read_FAQ	Pages_Viewed	Service_Chosen
0	4	3	1	2	1
1	2	1	1	0	2
2	1	3	1	0	0
3	3	1	1	0	0
4	2	2	0	2	2
5	0	0	0	1	1
6	0	2	0	2	0
7	2	3	0	0	2
8	4	1	1	2	1
9	1	3	0	2	1
10	2	2	0	2	1
11	3	2	0	2	1
12	1	0	1	1	0
13	2	2	1	2	0
14	3	1	1	2	0
15	2	2	0	2	2
16	1	3	0	1	0
17	4	0	1	0	1

In [152]:
```
%time
clf = tree.DecisionTreeClassifier()
X = df[['Referrer', 'Location', 'Read_FAQ', 'Pages_Viewed']]
y = df['Service_Chosen']
clf = clf.fit(X, y)
y_pred = clf.predict(X)
y_pred
```

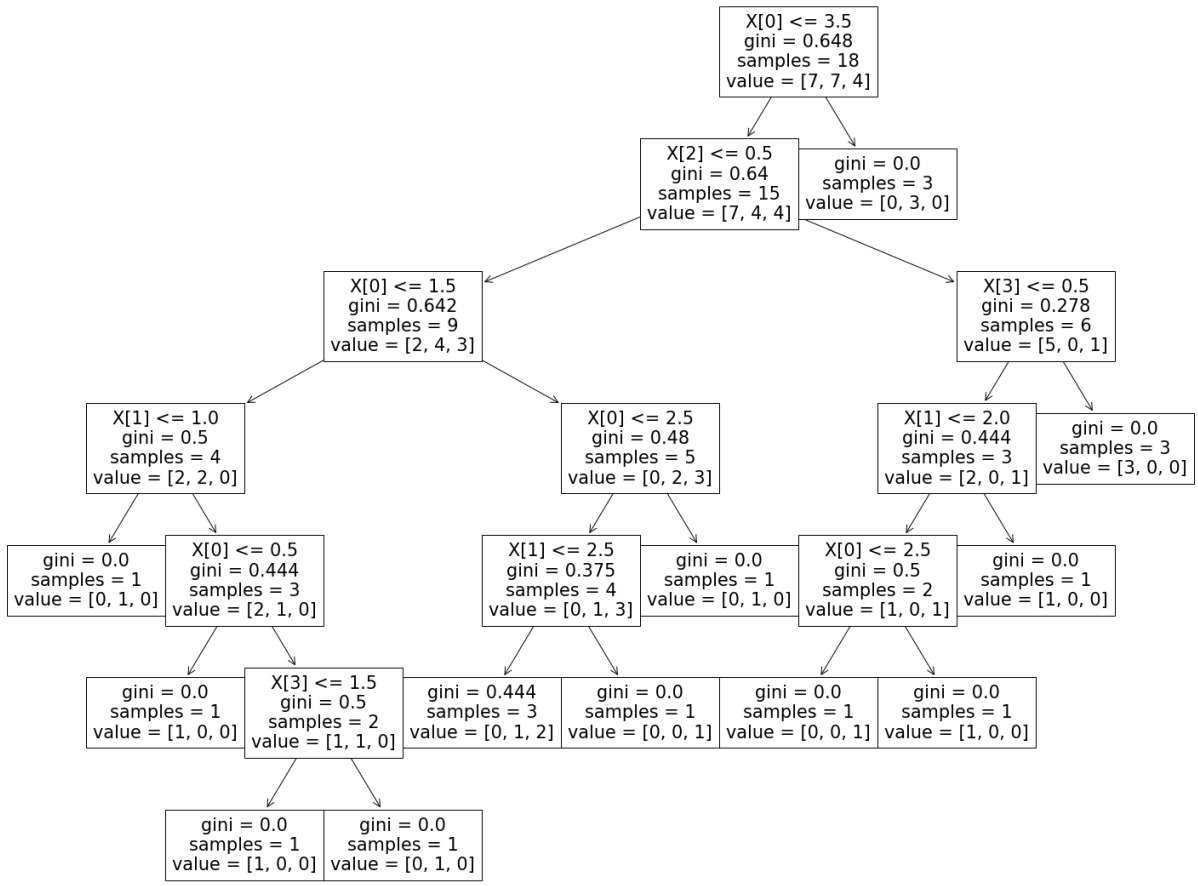
```
Wall time: 0 ns
```

Out[152]:
```
array([1, 2, 0, 0, 2, 1, 0, 2, 1, 1, 2, 1, 0, 0, 0, 2, 0, 1])
```

In [153]:
```
%time
fig = plt.figure(figsize=(25,20))
_ = tree.plot_tree(clf, filled=True)
fig.savefig("decistion_tree.png")
tree.plot_tree(clf)
```

```
Wall time: 0 ns
```

Out[153]:
```
[Text(930.0, 1009.5428571428572, 'X[0] <= 3.5\ngini = 0.648\nsamples = 18\nva
lue = [7, 7, 4]'),
 Text(837.0, 854.2285714285715, 'X[2] <= 0.5\ngini = 0.64\nsamples = 15\nvalu
e = [7, 4, 4]'),
 Text(465.0, 698.9142857142858, 'X[0] <= 1.5\ngini = 0.642\nsamples = 9\nvalu
e = [2, 4, 3]'),
 Text(186.0, 543.6, 'X[1] <= 1.0\ngini = 0.5\nsamples = 4\nvalue = [2, 2,
0]'),
 Text(93.0, 388.28571428571433, 'gini = 0.0\nsamples = 1\nvalue = [0, 1,
0]'),
 Text(279.0, 388.28571428571433, 'X[0] <= 0.5\ngini = 0.444\nsamples = 3\nval
ue = [2, 1, 0]'),
 Text(186.0, 232.97142857142865, 'gini = 0.0\nsamples = 1\nvalue = [1, 0,
0]'),
 Text(372.0, 232.97142857142865, 'X[3] <= 1.5\ngini = 0.5\nsamples = 2\nvalue
= [1, 1, 0]'),
 Text(279.0, 77.65714285714284, 'gini = 0.0\nsamples = 1\nvalue = [1, 0,
0]'),
 Text(465.0, 77.65714285714284, 'gini = 0.0\nsamples = 1\nvalue = [0, 1,
0]'),
 Text(744.0, 543.6, 'X[0] <= 2.5\ngini = 0.48\nsamples = 5\nvalue = [0, 2,
3]'),
 Text(651.0, 388.28571428571433, 'X[1] <= 2.5\ngini = 0.375\nsamples = 4\nval
ue = [0, 1, 3]'),
 Text(558.0, 232.97142857142865, 'gini = 0.444\nsamples = 3\nvalue = [0, 1,
2]'),
 Text(744.0, 232.97142857142865, 'gini = 0.0\nsamples = 1\nvalue = [0, 0,
1]'),
 Text(837.0, 388.28571428571433, 'gini = 0.0\nsamples = 1\nvalue = [0, 1,
0]'),
 Text(1209.0, 698.9142857142858, 'X[3] <= 0.5\ngini = 0.278\nsamples = 6\nval
ue = [5, 0, 1]'),
 Text(1116.0, 543.6, 'X[1] <= 2.0\ngini = 0.444\nsamples = 3\nvalue = [2, 0,
1]'),
 Text(1023.0, 388.28571428571433, 'X[0] <= 2.5\ngini = 0.5\nsamples = 2\nvalu
e = [1, 0, 1]'),
 Text(930.0, 232.97142857142865, 'gini = 0.0\nsamples = 1\nvalue = [0, 0,
1]'),
 Text(1116.0, 232.97142857142865, 'gini = 0.0\nsamples = 1\nvalue = [1, 0,
0]'),
 Text(1209.0, 388.28571428571433, 'gini = 0.0\nsamples = 1\nvalue = [1, 0,
0]'),
 Text(1302.0, 543.6, 'gini = 0.0\nsamples = 3\nvalue = [3, 0, 0]'),
 Text(1023.0, 854.2285714285715, 'gini = 0.0\nsamples = 3\nvalue = [0, 3,
0]')]
```



In [154]:
```
from sklearn.metrics import accuracy_score,confusion_matrix,classification_report
print("Accuracy: ", accuracy_score(y,y_pred))
print("Confusion: \n", confusion_matrix(y,y_pred))
print("Classification Report: \n", classification_report(y,y_pred,target_names = ['None', 'Basic', 'Premium']))
```

```
Accuracy:  0.9444444444444444
Confusion:
 [[7 0 0]
 [0 6 1]
 [0 0 4]]
Classification Report:
              precision    recall  f1-score   support

        None       1.00      1.00      1.00         7
       Basic       1.00      0.86      0.92         7
     Premium       0.80      1.00      0.89         4

    accuracy                           0.94        18
   macro avg       0.93      0.95      0.94        18
weighted avg       0.96      0.94      0.95        18
```

In [155]:
```
df_test = pd.read_csv('PredictingSignupsTest.csv')
df_test
```

Out[155]:

	Referrer	Location	Read_FAQ	Pages_Viewed	Service_Chosen
0	Google	UK	No	Mid	Premium
1	Digg	USA	No	Low	Basic
2	Slashdot	'New Zealand'	Yes	High	None

In [156]:
```
lb_enc = LabelEncoder()
df_test['Referrer'] = lb_enc.fit_transform(df_test['Referrer'])
df_test['Location'] = lb_enc.fit_transform(df_test['Location'])
df_test['Read_FAQ'] = lb_enc.fit_transform(df_test['Read_FAQ'])
df_test['Pages_Viewed'] = lb_enc.fit_transform(df_test['Pages_Viewed'])
df_test['Service_Chosen'] = lb_enc.fit_transform(df_test['Service_Chosen'])
df_test
```

Out[156]:

	Referrer	Location	Read_FAQ	Pages_Viewed	Service_Chosen
0	1	1	0	2	2
1	0	2	0	1	0
2	2	0	1	0	1

In [157]:
```
%time
clf = tree.DecisionTreeClassifier()
X = df_test[['Referrer', 'Location', 'Read_FAQ', 'Pages_Viewed']]
y = df_test['Service_Chosen']
clf = clf.fit(X, y)
y_pred = clf.predict(X)
y_pred
```

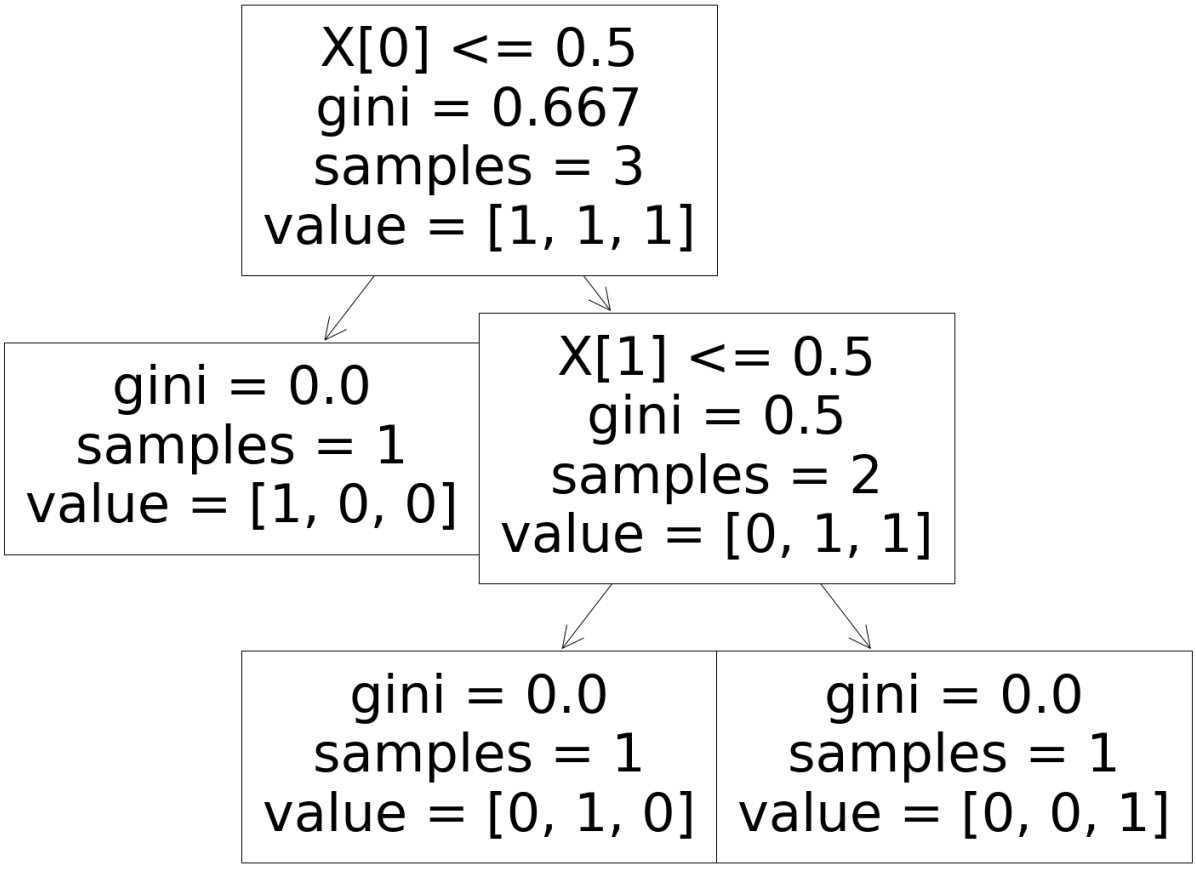
```
Wall time: 0 ns
```

Out[157]:
```
array([2, 0, 1])
```

In [158]:
```
%time
fig = plt.figure(figsize=(25,20))
_ = tree.plot_tree(clf, filled=True)
fig.savefig("decistion_tree.png")
tree.plot_tree(clf)
```

```
Wall time: 0 ns
```

Out[158]:
```
[Text(558.0, 906.0, 'X[0] <= 0.5\ngini = 0.667\nsamples = 3\nvalue = [1, 1,
1]'),
 Text(279.0, 543.6, 'gini = 0.0\nsamples = 1\nvalue = [1, 0, 0]'),
 Text(837.0, 543.6, 'X[1] <= 0.5\ngini = 0.5\nsamples = 2\nvalue = [0, 1,
1]'),
 Text(558.0, 181.19999999999993, 'gini = 0.0\nsamples = 1\nvalue = [0, 1,
0]'),
 Text(1116.0, 181.19999999999993, 'gini = 0.0\nsamples = 1\nvalue = [0, 0,
1]')]
```



In [159]:
```
from sklearn.metrics import accuracy_score,confusion_matrix,classification_report
print("Accuracy: ", accuracy_score(y,y_pred))
print("Confusion: \n", confusion_matrix(y,y_pred))
print("Classification Report: \n", classification_report(y,y_pred,target_names = ['None', 'Basic', 'Premium']))
```

```
Accuracy:  1.0
Confusion:
 [[1 0 0]
 [0 1 0]
 [0 0 1]]
Classification Report:
              precision    recall  f1-score   support

        None       1.00      1.00      1.00         1
       Basic       1.00      1.00      1.00         1
     Premium       1.00      1.00      1.00         1

    accuracy                           1.00         3
   macro avg       1.00      1.00      1.00         3
weighted avg       1.00      1.00      1.00         3
```

In [160]:
```
X.shape, y.shape
```

Out[160]:
```
((3, 4), (3,))
```