Tarea11_Arbol

December 14, 2017

1 Tarea 11. Árbol De Decisión

L.E. Rojón 138442

1.1 Boston Housing Data

This data set concerns housing values in suburbs of Boston and has the following sources: * (a) Origin: This dataset was taken from the StatLib library which is maintained at Carnegie Mellon University. * (b) Creator: Harrison, D. and Rubinfeld, D.L. 'Hedonic prices and the demand for clean air', J. Environ. Economics & Management, vol.5, 81-102, 1978. * (c) Date: July 7, 1993

Before it was used in: - Belsley, Kuh & Welsch, 'Regression diagnostics ...', Wiley, 1980. N.B. Various transformations are used in the table on pages 244-261. - Quinlan,R. (1993). Combining Instance-Based and Model-Based Learning. In Proceedings on the Tenth International Conference of Machine Learning, 236-243, University of Massachusetts, Amherst. Morgan Kaufmann.

Attribute Information:

```
1. CRIM      per capita crime rate by town
2. ZN        proportion of residential land zoned for lots over
             25,000 sq.ft.
3. INDUS     proportion of non-retail business acres per town
4. CHAS      Charles River dummy variable (= 1 if tract bounds
             river; 0 otherwise)
5. NOX       nitric oxides concentration (parts per 10 million)
6. RM        average number of rooms per dwelling
7. AGE       proportion of owner-occupied units built prior to 1940
8. DIS       weighted distances to five Boston employment centres
9. RAD       index of accessibility to radial highways
10. TAX      full-value property-tax rate per $10,000
11. PTRATIO  pupil-teacher ratio by town
12. B        1000(Bk - 0.63)^2 where Bk is the proportion of blacks
             by town
13. LSTAT    % lower status of the population
14. MEDV     Median value of owner-occupied homes in $1000's
```

The following document implements the decision trees algorithm to decide if the Crime Rates on the Boston city can be predicted by it's closeness to the Charles River.

First we load the libraries and packages needed.

In [2]:
```
from sklearn.externals.six import StringIO
from IPython.display import Image
import graphviz
from sklearn.tree import export_graphviz
import seaborn as sns
%matplotlib inline
import pydot
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from sklearn import datasets
from sklearn import tree
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
from sklearn import preprocessing
```

Now we load the data.

In [3]:
```
df=pd.read_csv('https://archive.ics.uci.edu/ml/machine-learning-databases/housing/housin
                names=['CRIM','ZN','INDUS','CHAS','NOX','RM','AGE','DIS','RAD',
                       'TAX','PTRATIO','B','LSTAT','MEDV'])
df.head()
#list(df)
```

Out[3]:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	\
0	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296.0	
1	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242.0	
2	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242.0	
3	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222.0	
4	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222.0	

	PTRATIO	B	LSTAT	MEDV
0	15.3	396.90	4.98	24.0
1	17.8	396.90	9.14	21.6
2	17.8	392.83	4.03	34.7
3	18.7	394.63	2.94	33.4
4	18.7	396.90	5.33	36.2

In [4]:
```
X = df['CRIM']
y = df['CHAS']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25, random_state
```

In [5]:
```
X_train=X_train.reshape(-1, 1)
y_train=y_train.reshape(-1, 1)

X_test=X_test.reshape(-1, 1)
y_test=y_test.reshape(-1, 1)
```

```
/home/luxorville/virtualenvs/ML/lib/python2.7/site-packages/ipykernel_launcher.py:1: FutureWarni
  """Entry point for launching an IPython kernel.
/home/luxorville/virtualenvs/ML/lib/python2.7/site-packages/ipykernel_launcher.py:2: FutureWarni

/home/luxorville/virtualenvs/ML/lib/python2.7/site-packages/ipykernel_launcher.py:4: FutureWarni
  after removing the cwd from sys.path.
/home/luxorville/virtualenvs/ML/lib/python2.7/site-packages/ipykernel_launcher.py:5: FutureWarni
  """
```

We adjust a decision tree to the data.

```
In [6]: classifier = DecisionTreeClassifier(random_state = 0)
        classifier.fit(X_train, y_train)
```

```
Out[6]: DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None,
                    max_features=None, 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, presort=False, random_state=0,
                    splitter='best')
```

Now we check with the test set and evaluate the accuracy and the confusion matrix.

```
In [7]: pred = classifier.predict(X_test)

        print(classification_report(y_test,pred))
        print(confusion_matrix(y_test,pred))
        print('Presicion:',accuracy_score(y_test,pred))
        cm=confusion_matrix(y_test,pred)
        sns.heatmap(cm,cmap='viridis')
```

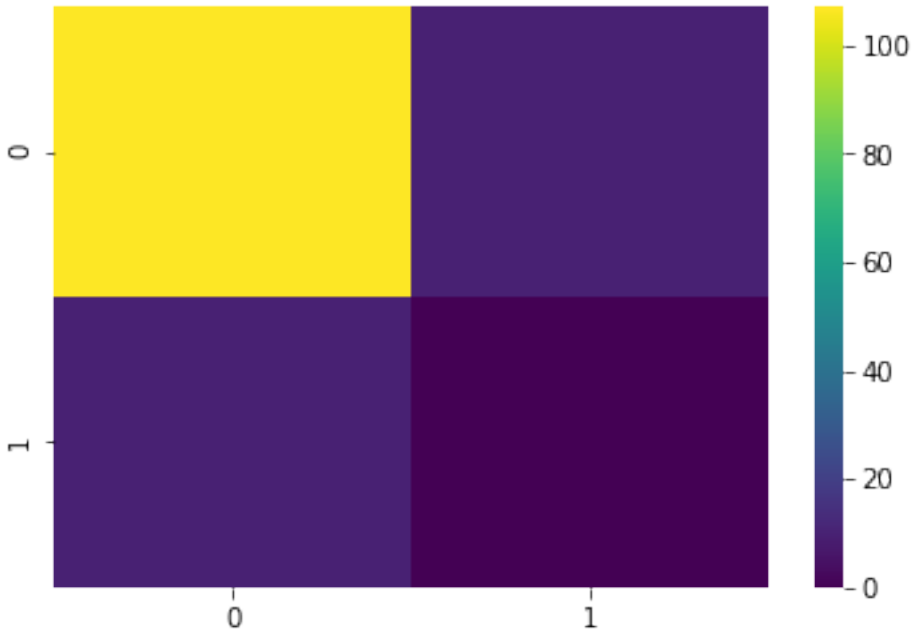
```
             precision    recall  f1-score   support

          0       0.91      0.91      0.91       117
          1       0.00      0.00      0.00        10

avg / total       0.84      0.84      0.84       127

[[107  10]
 [ 10   0]]
('Presicion:', 0.84251968503937003)
```

```
Out[7]: <matplotlib.axes._subplots.AxesSubplot at 0x7f33fce4e950>
```



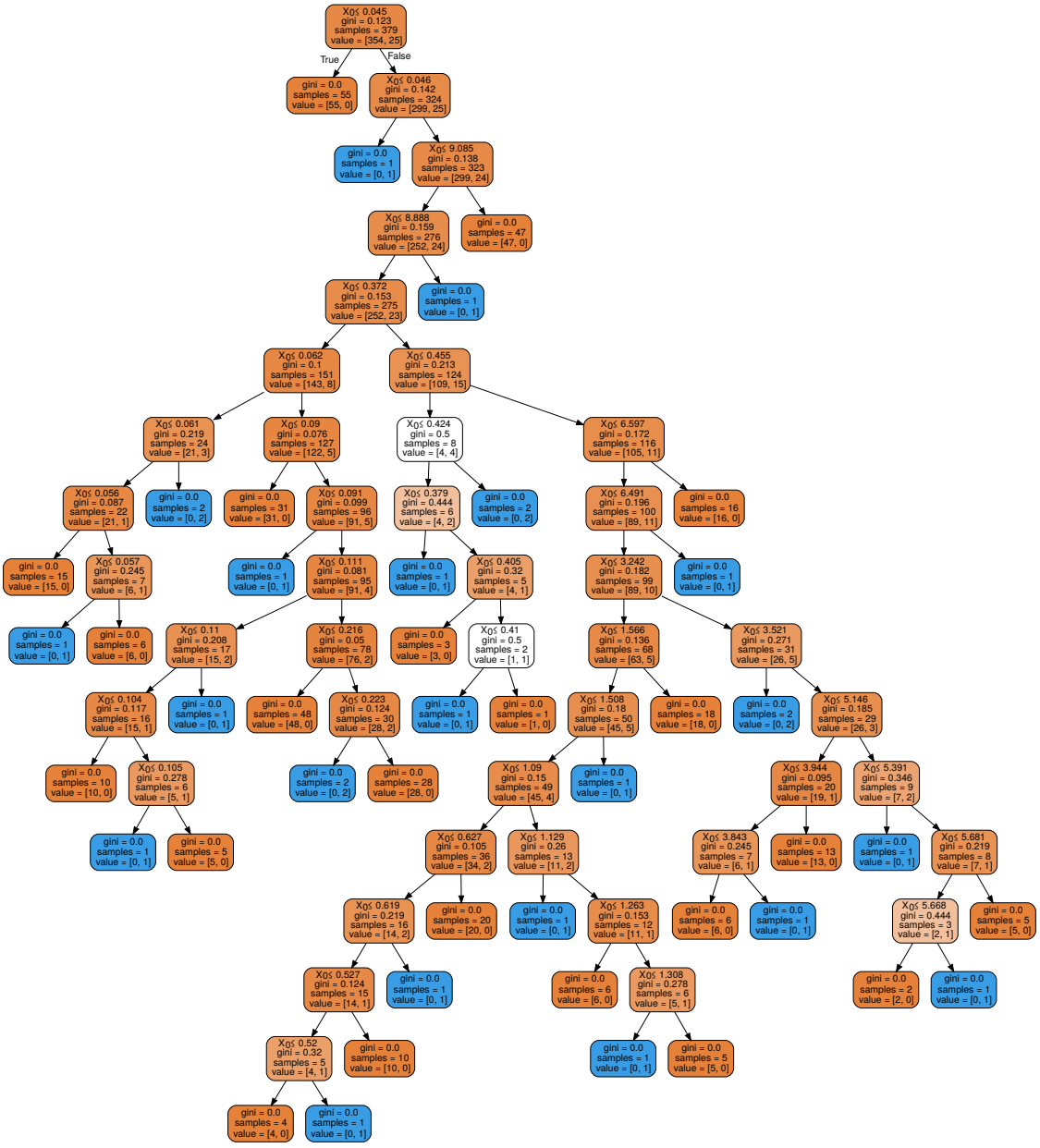
Finally, we graph the dendogram.

```
In [8]: dot_data = tree.export_graphviz(classifier, out_file=None,
                                       filled=True, rounded=True,
                                       special_characters=True)



        graph = graphviz.Source(dot_data)
        graph
```

Out[8]:


X0 ≤ 0.045
gini = 0.123
samples = 379
value = [354, 25]
True
False
gini = 0.0
samples = 55
value = [55, 0]
X0 ≤ 0.046
gini = 0.142
samples = 324
value = [299, 25]
gini = 0.0
samples = 1
value = [0, 1]
X0 ≤ 9.085
gini = 0.138
samples = 323
value = [299, 24]
X0 ≤ 8.888
gini = 0.159
samples = 276
value = [252, 24]
gini = 0.0
samples = 47
value = [47, 0]
X0 ≤ 0.372
gini = 0.153
samples = 275
value = [252, 23]
gini = 0.0
samples = 1
value = [0, 1]
X0 ≤ 0.062
gini = 0.1
samples = 151
value = [143, 8]
X0 ≤ 0.455
gini = 0.213
samples = 124
value = [109, 15]
X0 ≤ 0.061
gini = 0.219
samples = 24
value = [21, 3]
X0 ≤ 0.09
gini = 0.076
samples = 127
value = [122, 5]
X0 ≤ 0.424
gini = 0.5
samples = 8
value = [4, 4]
X0 ≤ 6.597
gini = 0.172
samples = 116
value = [105, 11]
X0 ≤ 0.056
gini = 0.087
samples = 22
value = [21, 1]
gini = 0.0
samples = 2
value = [0, 2]
gini = 0.0
samples = 31
value = [31, 0]
X0 ≤ 0.091
gini = 0.099
samples = 96
value = [91, 5]
X0 ≤ 0.379
gini = 0.444
samples = 6
value = [4, 2]
gini = 0.0
samples = 2
value = [0, 2]
X0 ≤ 6.491
gini = 0.196
samples = 100
value = [89, 11]
gini = 0.0
samples = 16
value = [16, 0]
gini = 0.0
samples = 15
value = [15, 0]
X0 ≤ 0.057
gini = 0.245
samples = 7
value = [6, 1]
gini = 0.0
samples = 1
value = [0, 1]
X0 ≤ 0.111
gini = 0.081
samples = 95
value = [91, 4]
gini = 0.0
samples = 1
value = [0, 1]
X0 ≤ 0.405
gini = 0.32
samples = 5
value = [4, 1]
X0 ≤ 3.242
gini = 0.182
samples = 99
value = [89, 10]
gini = 0.0
samples = 1
value = [0, 1]
gini = 0.0
samples = 1
value = [0, 1]
X0 ≤ 0.11
gini = 0.208
samples = 17
value = [15, 2]
X0 ≤ 0.216
gini = 0.05
samples = 78
value = [76, 2]
gini = 0.0
samples = 3
value = [3, 0]
X0 ≤ 0.41
gini = 0.5
samples = 2
value = [1, 1]
X0 ≤ 1.566
gini = 0.136
samples = 68
value = [63, 5]
X0 ≤ 3.521
gini = 0.271
samples = 31
value = [26, 5]
X0 ≤ 0.104
gini = 0.117
samples = 16
value = [15, 1]
gini = 0.0
samples = 1
value = [0, 1]
gini = 0.0
samples = 48
value = [48, 0]
X0 ≤ 0.223
gini = 0.124
samples = 30
value = [28, 2]
gini = 0.0
samples = 1
value = [0, 1]
gini = 0.0
samples = 1
value = [1, 0]
X0 ≤ 1.508
gini = 0.18
samples = 50
value = [45, 5]
gini = 0.0
samples = 18
value = [18, 0]
gini = 0.0
samples = 2
value = [0, 2]
X0 ≤ 5.146
gini = 0.185
samples = 29
value = [26, 3]
gini = 0.0
samples = 10
value = [10, 0]
X0 ≤ 0.105
gini = 0.278
samples = 6
value = [5, 1]
gini = 0.0
samples = 2
value = [0, 2]
gini = 0.0
samples = 28
value = [28, 0]
X0 ≤ 1.09
gini = 0.15
samples = 49
value = [45, 4]
gini = 0.0
samples = 1
value = [0, 1]
X0 ≤ 3.944
gini = 0.095
samples = 20
value = [19, 1]
X0 ≤ 5.391
gini = 0.346
samples = 9
value = [7, 2]
gini = 0.0
samples = 1
value = [0, 1]
gini = 0.0
samples = 1
value = [0, 1]
gini = 0.0
samples = 5
value = [5, 0]
X0 ≤ 0.627
gini = 0.105
samples = 36
value = [34, 2]
X0 ≤ 1.129
gini = 0.26
samples = 13
value = [11, 2]
X0 ≤ 3.843
gini = 0.245
samples = 7
value = [6, 1]
gini = 0.0
samples = 13
value = [13, 0]
X0 ≤ 5.681
gini = 0.219
samples = 8
value = [7, 1]
X0 ≤ 0.619
gini = 0.219
samples = 16
value = [14, 2]
gini = 0.0
samples = 20
value = [20, 0]
gini = 0.0
samples = 1
value = [0, 1]
X0 ≤ 1.263
gini = 0.153
samples = 12
value = [11, 1]
gini = 0.0
samples = 6
value = [6, 0]
gini = 0.0
samples = 1
value = [0, 1]
X0 ≤ 5.668
gini = 0.444
samples = 3
value = [2, 1]
gini = 0.0
samples = 5
value = [5, 0]
X0 ≤ 0.527
gini = 0.124
samples = 15
value = [14, 1]
gini = 0.0
samples = 1
value = [0, 1]
gini = 0.0
samples = 1
value = [0, 1]
X0 ≤ 1.308
gini = 0.278
samples = 6
value = [5, 1]
gini = 0.0
samples = 2
value = [2, 0]
gini = 0.0
samples = 1
value = [0, 1]
X0 ≤ 0.52
gini = 0.32
samples = 5
value = [4, 1]
gini = 0.0
samples = 10
value = [10, 0]
gini = 0.0
samples = 1
value = [0, 1]
gini = 0.0
samples = 5
value = [5, 0]
gini = 0.0
samples = 4
value = [4, 0]
gini = 0.0
samples = 1
value = [0, 1]