Data Visualization Example

CSI 5810

Ishwar K Sethi

Boston Housing Data

The Boston housing data was collected in 1978 and each of the 506 entries represent 13 independent features and one dependent feature for homes from various suburbs in Boston, Massachusetts. It is available at UCI Machine Learning Repository. It is also included in sklearn library for machine learning. The features and their meanings are:

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

```
#import necessary libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib import style
```

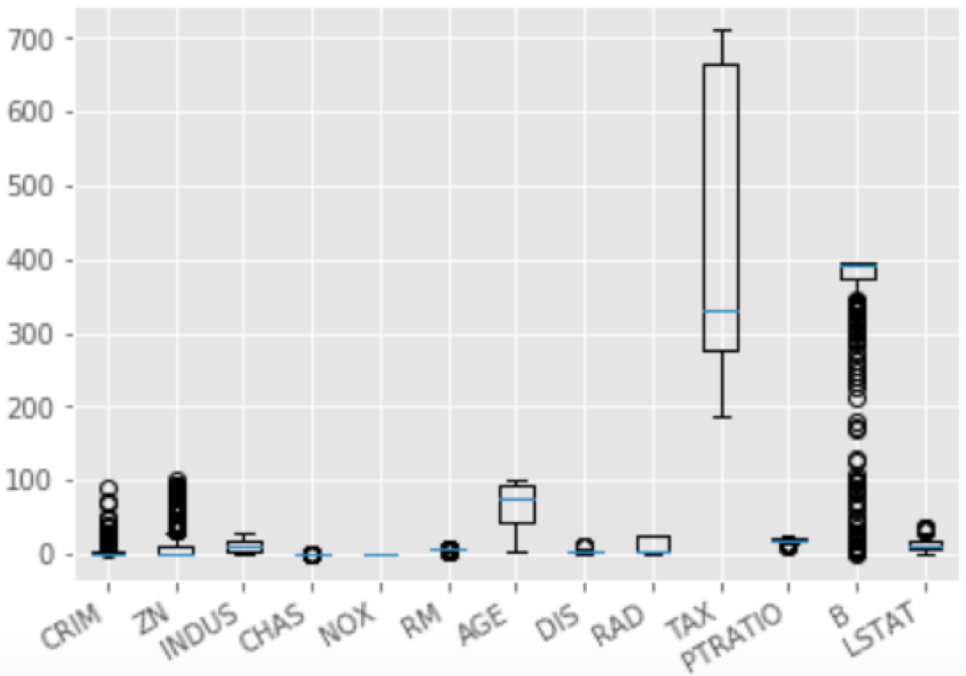
```
#Load data
from sklearn import datasets
boston = datasets.load_boston()
# Lets separate features and the dependent values
X = boston.data
Y = boston.target
print(X.shape,Y.shape)
print(boston.feature_names)
```

```
(506, 13) (506,)
['CRIM' 'ZN' 'INDUS' 'CHAS' 'NOX' 'RM' 'AGE' 'DIS' 'RAD' 'TAX' 'PTRATIO'
 'B' 'LSTAT']
```

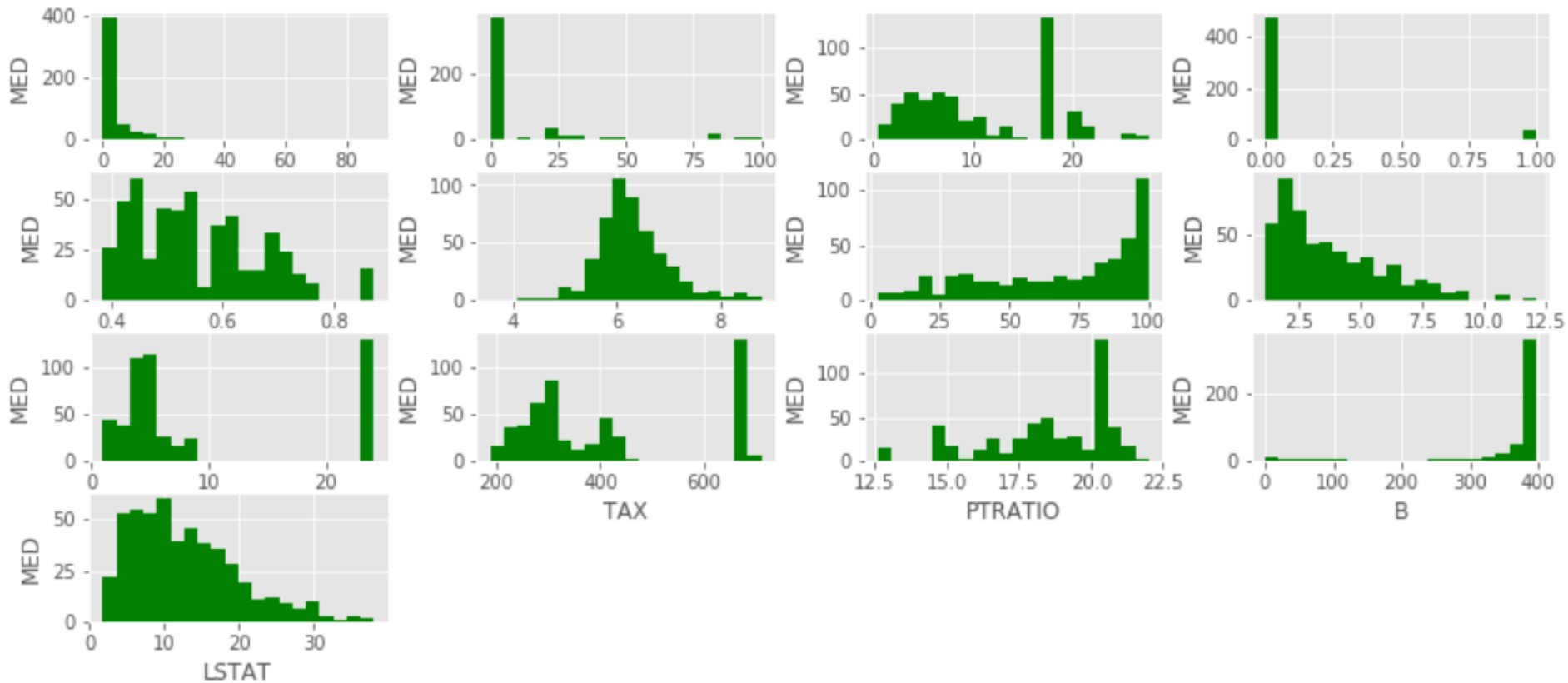
```python
#Lets first look at descriptive statistics of the data.
#Will do it by using pandas because it produces a nicer output
df = pd.DataFrame(X, columns=boston.feature_names)
print(df.describe())
print((pd.DataFrame(Y,columns=['MEDV'])).describe())
```

```
              CRIM          ZN       INDUS        CHAS         NOX          RM  \
count  506.000000  506.000000  506.000000  506.000000  506.000000  506.000000
mean     3.593761   11.363636   11.136779    0.069170    0.554695    6.284634
std      8.596783   23.322453    6.860353    0.253994    0.115878    0.702617
min      0.006320    0.000000    0.460000    0.000000    0.385000    3.561000
25%      0.082045    0.000000    5.190000    0.000000    0.449000    5.885500
50%      0.256510    0.000000    9.690000    0.000000    0.538000    6.208500
75%      3.647423   12.500000   18.100000    0.000000    0.624000    6.623500
max     88.976200  100.000000   27.740000    1.000000    0.871000    8.780000

              AGE         DIS         RAD         TAX     PTRATIO           B  \
count  506.000000  506.000000  506.000000  506.000000  506.000000  506.000000
mean    68.574901    3.795043    9.549407  408.237154   18.455534  356.674032
std     28.148861    2.105710    8.707259  168.537116    2.164946   91.294864
min      2.900000    1.129600    1.000000  187.000000   12.600000    0.320000
25%     45.025000    2.100175    4.000000  279.000000   17.400000  375.377500
50%     77.500000    3.207450    5.000000  330.000000   19.050000  391.440000
75%     94.075000    5.188425   24.000000  666.000000   20.200000  396.225000
max    100.000000   12.126500   24.000000  711.000000   22.000000  396.900000

            LSTAT
count  506.000000
mean    12.653063
std      7.141062
min      1.730000
25%      6.950000
50%     11.360000
75%     16.955000
max     37.970000
             MEDV
count  506.000000
mean    22.532806
std      9.197104
min      5.000000
25%     17.025000
50%     21.200000
75%     25.000000
max     50.000000
```

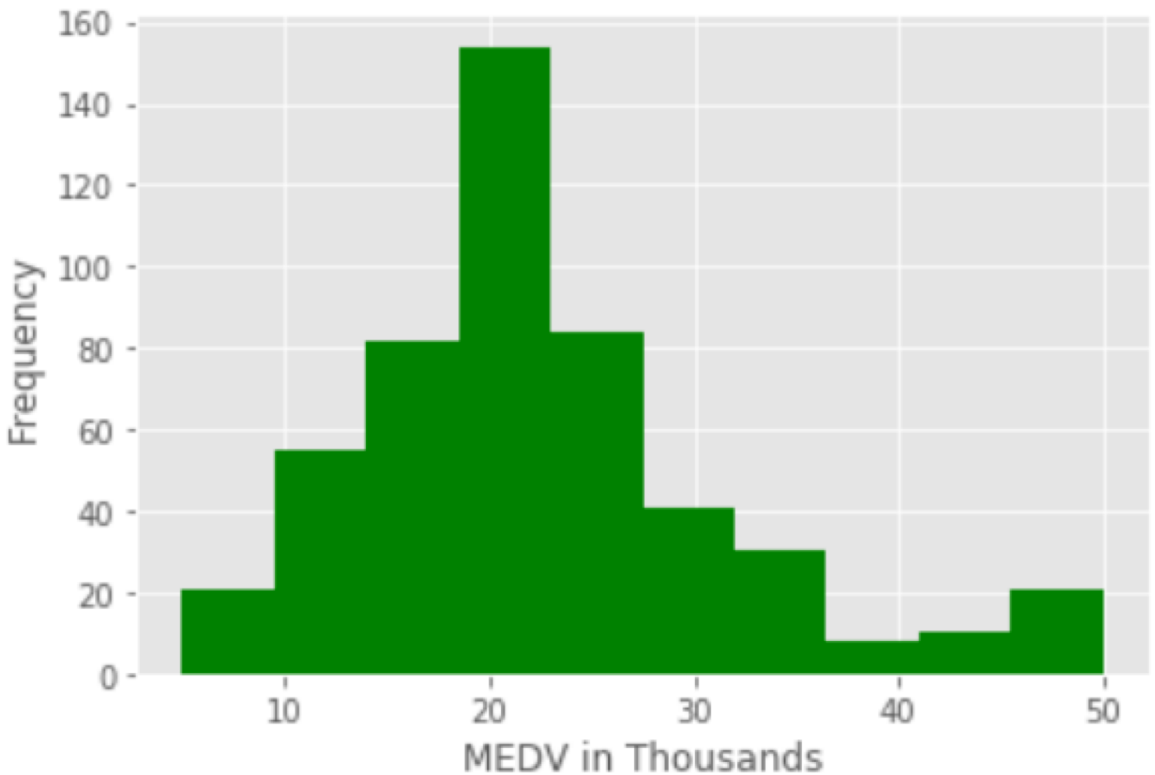
```
#Lets do boxplot of the features
plt.boxplot(X)
plt.xticks(np.arange(1, X.shape[1]+1),headings,rotation=30,ha="right")
plt.show()
```



```
# Lets look at each of the 13 independent features
plt.style.use('ggplot')
fig, axs = plt.subplots(4, 4, figsize=(14,6))
headings = boston.feature_names
for i in range(1,14):
    plt.subplot(4,4,i)
    plt.hist(X[:,i-1], bins=20, color='g')
    plt.xlabel(headings[i-1])
    plt.ylabel('MED')
plt.subplots_adjust(wspace=0.30, hspace=0.25)
fig.delaxes(axs[3][1])
fig.delaxes(axs[3][2])
fig.delaxes(axs[3][3])
plt.show()
```



```
# Lets look at the price distribution (MED) also
plt.hist(Y, color='g')
plt.title('MEDV Histogram')
plt.xlabel('MEDV in Thousands')
plt.ylabel('Frequency')
plt.show()
```



```
# Lets look at now the relationship between the target value and each independent feature
fig, axs = plt.subplots(3, 5, figsize=(16,6))
for i in range(1,14):
    plt.subplot(3,5,i)
    plt.scatter(X[:,i-1],Y,marker ='.', color='g')
    plt.xlabel(headings[i-1])
    plt.ylabel('MED')
plt.subplots_adjust(wspace=0.25, hspace=0.50)
fig.delaxes(axs[2][3])
fig.delaxes(axs[2][4])
plt.show()
```