Week 5 Quiz

Bryan Gibson - brg2130

In [1]:
```
# import the datasets module from sklearn
from sklearn import datasets
```

In [2]:
```
# use datasets.load_boston() to load the Boston housing dataset
boston = datasets.load_boston()
```

In [3]:
```
# print the description of the dataset in boston.DESCR
print(boston.DESCR)
```

```
.. _boston_dataset:

Boston house prices dataset
---------------------------

**Data Set Characteristics:**

    :Number of Instances: 506

    :Number of Attributes: 13 numeric/categorical predictive. Median
Value (attribute 14) is usually the target.

    :Attribute Information (in order):
        - 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 centr
es
        - 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 bl
acks by town
        - LSTAT    % lower status of the population
        - MEDV     Median value of owner-occupied homes in $1000's

    :Missing Attribute Values: None

    :Creator: Harrison, D. and Rubinfeld, D.L.

This is a copy of UCI ML housing dataset.
https://archive.ics.uci.edu/ml/machine-learning-databases/housing/


This dataset was taken from the StatLib library which is maintained a
t Carnegie Mellon University.

The Boston house-price data of Harrison, D. and Rubinfeld, D.L. 'Hedo
nic
prices and the demand for clean air', J. Environ. Economics & Managem
ent,
vol.5, 81-102, 1978.   Used in Belsley, Kuh & Welsch, 'Regression dia
gnostics
...', Wiley, 1980.   N.B. Various transformations are used in the tab
le on
pages 244-261 of the latter.

The Boston house-price data has been used in many machine learning pa
pers that address regression
problems.
```

```
.. topic:: References

   - Belsley, Kuh & Welsch, 'Regression diagnostics: Identifying Infl
uential Data and Sources of Collinearity', Wiley, 1980. 244-261.
   - Quinlan,R. (1993). Combining Instance-Based and Model-Based Lear
ning. In Proceedings on the Tenth International Conference of Machine
Learning, 236-243, University of Massachusetts, Amherst. Morgan Kaufm
ann.
```

In [4]:
```
# copy the dataset features from boston.data to X
X = boston.data
```

In [5]:
```
# copy the dataset labels from boston.target to y
y = boston.target
```

In [6]:
```
# import the LinearRegression model from sklearn.linear_model
from sklearn.linear_model import LinearRegression
```

In [7]:
```
# initialize a linear regression model as lr with the default argumen
ts
lr = LinearRegression()
```

In [8]:
```
# fit the lr model using the entire set of X features and y labels
lr.fit(X,y)
```

Out[8]:
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normal
ize=False)
```

In [9]:
```
# score the lr model on entire set of X features and y labels
lr.score(X,y)
```

Out[9]:
```
0.7406426641094095
```

In [10]:
```
# import the DecisionTreeRegressor from sklearn.tree
from sklearn.tree import DecisionTreeRegressor
```

In [11]:
```
# initialize a decision tree model as dt with the default arguments
dt = DecisionTreeRegressor()
```

In [12]:
```
# fit the dt model using the entire set of X features and y labels
dt.fit(X,y)
```

Out[12]:
```
DecisionTreeRegressor(criterion='mse', max_depth=None, max_features=N
one,
                      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=None, splitter='bes
t')
```

In [13]:
```
# score the dt model on the entire set of X features and y labels
dt.score(X,y)
```

Out[13]: 1.0

What are we doing wrong here?!
Why shouldn't we trust these scores to tell us how the models with generalize?

We're training and evaluating on the same data set.