DIS

November 18, 2019

[1]:
```
%%html
<style>
.container{width: 100%}
</style>
```

```
<IPython.core.display.HTML object>
```

[2]:
```
%load_ext autoreload
%autoreload 2
```

[3]:
```
import warnings
warnings.filterwarnings("ignore")
```

[4]:
```
import numpy as np
import pandas as pd

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

[5]:
```
import os
os.sys.path.insert(0, "../")
```

0.0.1 Load Data

[6]:
```
from tools import load_boston
data, desc = load_boston("../data_base")
data = data.rename(columns = {"target": "MEDV"})
features = data.drop("MEDV", axis = 1)
prices = data.MEDV
```

[7]:
```
print(desc)
```

```
.. _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 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   % 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 at Carnegie Mellon University.

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

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

.. topic:: References

```
   - Belsley, Kuh & Welsch, 'Regression diagnostics: Identifying Influential
Data and Sources of Collinearity', Wiley, 1980. 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.
```

0.0.2 Feature Engineering

Train Test Splitting

[8]:
```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(features, prices, test_size␣
 ↪= .3, random_state = 42)
X_select, X_train, y_select, y_train = train_test_split(features, prices,␣
 ↪test_size = .4, random_state = 42)
```

DIS

[9]:
```
from tools import cal_benchmark_perf
from sklearn.model_selection import train_test_split
from tools import cal_perf

ALPHA = np.power(10, np.linspace(-1.5, 0.5, 20))
param_grid = {
    "alpha": ALPHA
}
```

[10]:
```
_X_train, _X_test, _y_train, _y_test = train_test_split(X_select, y_select,␣
 ↪test_size = .5, random_state = 42)
before_perf = cal_perf(_X_train, _y_train, _X_test, _y_test, param_grid,␣
 ↪return_grid = False)
```

```
Fitting 5 folds for each of 20 candidates, totalling 100 fits

[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done 100 out of 100 | elapsed:    1.6s finished
C:\ProgramData\Anaconda3\lib\site-
packages\sklearn\model_selection\_search.py:813: DeprecationWarning: The default
of the `iid` parameter will change from True to False in version 0.22 and will
be removed in 0.24. This will change numeric results when test-set sizes are
unequal.
  DeprecationWarning)
```

[11]:
```
def add_logged_dis(features):
    features["logged_dis"] = np.log(features.DIS)
    return
```

```
add_logged_dis(X_select)
```

[12]:
```
_X_train, _X_test, _y_train, _y_test = train_test_split(X_select, y_select,␣
 ↪test_size = .5, random_state = 42)
after_perf = cal_perf(_X_train, _y_train, _X_test, _y_test, param_grid,␣
 ↪return_grid = False)
```

```
Fitting 5 folds for each of 20 candidates, totalling 100 fits

[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done 100 out of 100 | elapsed:    0.1s finished
C:\ProgramData\Anaconda3\lib\site-
packages\sklearn\model_selection\_search.py:813: DeprecationWarning: The default
of the `iid` parameter will change from True to False in version 0.22 and will
be removed in 0.24. This will change numeric results when test-set sizes are
unequal.
  DeprecationWarning)
```

[13]:
```
print(after_perf / before_perf)
```

```
0.8589130044539982
```

Add Features

[14]:
```
for X in [X_select, X_train, X_test]:
    add_logged_dis(X)
```

Calculate Benchmark

[15]:
```
from tools import cal_benchmark_perf
benchmark = cal_benchmark_perf(X_train, y_train, X_test, y_test)
```

Lasso

[16]:
```
ALPHA = np.power(10, np.linspace(-1.5, 0.5, 20))
param_grid = {
    "alpha": ALPHA
}
```

[17]:
```
from tools import cal_perf
mod_perf, grid = cal_perf(X_train, y_train, X_test, y_test, param_grid,␣
 ↪return_grid=True)
```

```
Fitting 5 folds for each of 20 candidates, totalling 100 fits

[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done 100 out of 100 | elapsed:    0.1s finished
C:\ProgramData\Anaconda3\lib\site-
```

```
packages\sklearn\model_selection\_search.py:813: DeprecationWarning: The default
of the `iid` parameter will change from True to False in version 0.22 and will
be removed in 0.24. This will change numeric results when test-set sizes are
unequal.
  DeprecationWarning)
```

[18]:
```
grid.best_estimator_.coef_
```

[18]:
```
array([-1.43203767e-01,  3.92145073e-02, -8.06737738e-02,  8.39251196e-01,
       -1.72424972e+01,  3.74004767e+00,  9.52801719e-03,  5.71369455e-01,
        3.80746994e-01, -1.90651849e-02, -8.84241179e-01,  6.18531597e-03,
       -5.01548006e-01, -9.22170237e+00])
```

[19]:
```
grid.best_params_
```

[19]:
```
{'alpha': 0.03162277660168379}
```

[ ]:

[ ]: