LinearRegression

March 22, 2022

[1]:
```
import pandas as pd
```

[4]:
```
from sklearn.datasets import load_boston
import seaborn as sns
```

[7]:
```
data = load_boston()
```

```
/home/niteesh/.local/lib/python3.8/site-
packages/sklearn/utils/deprecation.py:87: FutureWarning: Function load_boston is
deprecated; `load_boston` is deprecated in 1.0 and will be removed in 1.2.

    The Boston housing prices dataset has an ethical problem. You can refer to
    the documentation of this function for further details.

    The scikit-learn maintainers therefore strongly discourage the use of this
    dataset unless the purpose of the code is to study and educate about
    ethical issues in data science and machine learning.

    In this special case, you can fetch the dataset from the original
    source::

        import pandas as pd
        import numpy as np


        data_url = "http://lib.stat.cmu.edu/datasets/boston"
        raw_df = pd.read_csv(data_url, sep="\s+", skiprows=22, header=None)
        data = np.hstack([raw_df.values[::2, :], raw_df.values[1::2, :2]])
        target = raw_df.values[1::2, 2]

    Alternative datasets include the California housing dataset (i.e.
    :func:`~sklearn.datasets.fetch_california_housing`) and the Ames housing
    dataset. You can load the datasets as follows::

        from sklearn.datasets import fetch_california_housing
        housing = fetch_california_housing()

    for the California housing dataset and::
```

```
        from sklearn.datasets import fetch_openml
        housing = fetch_openml(name="house_prices", as_frame=True)

    for the Ames housing dataset.

  warnings.warn(msg, category=FutureWarning)
```

[8]:
```
data.keys()
```

[8]:
```
dict_keys(['data', 'target', 'feature_names', 'DESCR', 'filename',
'data_module'])
```

[9]:
```
feat = data['data']
targ = data['target']
```

[10]:
```
from sklearn.model_selection import train_test_split
```

[12]:
```
X = feat
y = targ
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33,␣
 ↪random_state=42)
```

[13]:
```
from sklearn.linear_model import LinearRegression
```

[14]:
```
model = LinearRegression()
```

[15]:
```
model.fit(X_train, y_train)
```

[15]:
```
LinearRegression()
```

[21]:
```
pred = model.predict(X_test)
```

[17]:
```
from sklearn.metrics import confusion_matrix, classification_report
```

[23]:
```
for i in range(20):
    print(y_test[i], pred[i])
```

```
23.6 28.534694689729807
32.4 36.618700597688
13.6 15.637510787533738
22.8 25.501449600489927
16.1 18.709673401984986
20.0 23.16471591463455
17.8 17.310110346997487
14.0 14.077363672339299
19.6 23.010643881674085
16.8 20.54223481873405
```

```
21.5 24.916323505161287
18.9 18.410980520907415
7.0 -6.520796874789568
21.2 21.833726044450795
18.5 19.14903064319961
29.8 26.05873220348986
18.8 20.302326252257735
10.2 5.749435670226436
50.0 40.33137811065283
14.1 17.457914457821413
```

[24]:
```
model.coef_
```

[24]:
```
array([-1.28749718e-01,  3.78232228e-02,  5.82109233e-02,  3.23866812e+00,
       -1.61698120e+01,  3.90205116e+00, -1.28507825e-02, -1.42222430e+00,
        2.34853915e-01, -8.21331947e-03, -9.28722459e-01,  1.17695921e-02,
       -5.47566338e-01])
```

[ ]:
```

```