tu18_LinearRegression_II_HW

April 6, 2023

1 Linear Regression II: Homework

1.0.1 Question 1:

- Load the Boston Housing Dataset
- Find the feature with highest correlation with the Median House Value
- Predict the Median House Value with the highest correlated feature
- How does the prediction of the this model (R2) compare with the prediction of the model used in the exercise in class that used all the features instead of just the most correlated feature?

[28]:
```
# Import libraries
import pandas as pd
import seaborn as sns
from sklearn.datasets import load_boston
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error,mean_squared_error, r2_score
```

[76]:
```
boston_dataset = load_boston()
```

```
/Users/jesadathavornfung/opt/anaconda3/lib/python3.9/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)
```

[4]:
```
print(boston_dataset.keys())
```

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

[5]:
```
print(boston_dataset.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 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 black people
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.
```

[7]:
```
# Create Data Frame
boston = pd.DataFrame(boston_dataset.data, columns=boston_dataset.feature_names)
boston.head()
```

[7]:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	\
0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1.0	296.0	
1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2.0	242.0	
2	0.02729	0.0	7.07	0.0	0.469	7.185	61.1	4.9671	2.0	242.0	
3	0.03237	0.0	2.18	0.0	0.458	6.998	45.8	6.0622	3.0	222.0	
4	0.06905	0.0	2.18	0.0	0.458	7.147	54.2	6.0622	3.0	222.0	

	PTRATIO	B	LSTAT
0	15.3	396.90	4.98
1	17.8	396.90	9.14
2	17.8	392.83	4.03
3	18.7	394.63	2.94
4	18.7	396.90	5.33

[86]:
```python
# Split the data into features (X) and target (y)
X = df.drop('MEDV', axis=1)
y = df['MEDV']

# Initialize a Linear Regression model
lr = LinearRegression()

# Iterate over all features and calculate the R-squared value for each
for feature in X.columns:
    X_feature = X[[feature]]
    lr.fit(X_feature, y)
    y_pred = lr.predict(X_feature)
    mae = mean_absolute_error(y, y_pred)
    mse = mean_squared_error(y, y_pred)
    r2 = r2_score(y, y_pred)
    print(f"R-squared for {feature}: {r2:.4f}")
```

```
R-squared for CRIM: 0.1508
R-squared for ZN: 0.1299
R-squared for INDUS: 0.2340
R-squared for CHAS: 0.0307
R-squared for NOX: 0.1826
R-squared for RM: 0.4835
R-squared for AGE: 0.1421
R-squared for DIS: 0.0625
R-squared for RAD: 0.1456
R-squared for TAX: 0.2195
R-squared for PTRATIO: 0.2578
R-squared for B: 0.1112
R-squared for LSTAT: 0.5441
```

The higest correlation (R^2) is LSTAT of 0.5441.

[81]:
```
# Convert the dataset into a Pandas DataFrame
df = pd.DataFrame(boston.data, columns=boston.feature_names)
df['MEDV'] = boston.target

# Find the feature with the highest correlation with the target (MEDV)
corr_matrix = df.corr()
highest_corr_feature = corr_matrix['MEDV'].sort_values(ascending=False).index[1]

# Split the data into features (X) and target (y)
X = df[[highest_corr_feature]]
y = df['MEDV']

# Initialize a Linear Regression model
lr = LinearRegression()

# Fit the model to the data
lr.fit(X, y)

# Predict the Median House Value using the highest correlated feature
X_new = pd.DataFrame({highest_corr_feature: [df[highest_corr_feature].max()]}) ␣
 ↪# using the max value of the feature for demonstration purposes
y_pred = lr.predict(X_new)

# Print the predicted value
print(f"Predicted Median House Value using feature {highest_corr_feature}:␣
 ↪${y_pred[0]:,.2f}")
```

```
Predicted Median House Value using feature RM: $45.25
```

[83]:
```
# Convert the dataset into a Pandas DataFrame
df = pd.DataFrame(boston.data, columns=boston.feature_names)
df['MEDV'] = boston.target

# Split the data into features (X) and target (y) using LSTAT
X = df[['LSTAT']]
y = df['MEDV']

# Initialize a Linear Regression model
lr = LinearRegression()

# Fit the model to the data
lr.fit(X, y)

# Predict the Median House Value using LSTAT
X_new = [[df['LSTAT'].max()]]  # using the max value of LSTAT
y_pred = lr.predict(X_new)
```

```
# Print the predicted value
print(f"Predicted Median House Value using LSTAT: ${y_pred[0]:,.2f}")
```

```
Predicted Median House Value using LSTAT: $-1.52
```

```
/Users/jesadathavornfung/opt/anaconda3/lib/python3.9/site-
packages/sklearn/base.py:450: UserWarning: X does not have valid feature names,
but LinearRegression was fitted with feature names
  warnings.warn(
```

The predicted R^2 give different median value. Although LSTAT have the highest R^2, it does not have the highest median value. Instead, it appears that RM has the highest median value. Also, I think that the exampl in class is more simple and easy to work with comapring to this excercise.

1.0.2 Question 2:

- Load the Boston Housing Dataset
- Find the feature with lowest correlation with the Median House Value
- Predict the Median House Value with all the features except the one with lowet correlation
- How does the prediction of the this model (R2) compare with the prediction of the full model model used in the exercise in class?

[84]:
```
# Import libraries
import pandas as pd
import seaborn as sns
from sklearn.datasets import load_boston
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error,mean_squared_error, r2_score
```

[85]:
```
# Load the Boston Housing Datset.
boston_dataset = load_boston()
```

```
/Users/jesadathavornfung/opt/anaconda3/lib/python3.9/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)
```

[88]:
```
# Split the data into features (X) and target (y)
X = df.drop('MEDV', axis=1)
y = df['MEDV']

# Initialize a Linear Regression model
lr = LinearRegression()

# Iterate over all features and calculate the R-squared value for each
for feature in X.columns:
    X_feature = X[[feature]]
    lr.fit(X_feature, y)
    y_pred = lr.predict(X_feature)
    mae = mean_absolute_error(y, y_pred)
    mse = mean_squared_error(y, y_pred)
    r2 = r2_score(y, y_pred)
    print(f"R-squared for {feature}: {r2:.4f}")
```

```
R-squared for CRIM: 0.1508
R-squared for ZN: 0.1299
R-squared for INDUS: 0.2340
R-squared for CHAS: 0.0307
R-squared for NOX: 0.1826
R-squared for RM: 0.4835
R-squared for AGE: 0.1421
```

```
R-squared for DIS: 0.0625
R-squared for RAD: 0.1456
R-squared for TAX: 0.2195
R-squared for PTRATIO: 0.2578
R-squared for B: 0.1112
R-squared for LSTAT: 0.5441
```

The higest correlation (R^2) is CHAS of 0.0307.

[90]:
```
# Convert the dataset into a Pandas DataFrame
df = pd.DataFrame(boston.data, columns=boston.feature_names)
df['MEDV'] = boston.target

# Remove the feature with the lowest correlation (CHAS)
df = df.drop('CHAS', axis=1)

# Split the data into features (X) and target (y)
X = df.drop('MEDV', axis=1)
y = df['MEDV']

# Initialize a Linear Regression model
lr = LinearRegression()

# Fit the model to the data
lr.fit(X, y)

# Predict the Median House Value using all features except CHAS
X_new = pd.DataFrame(X.iloc[0, :]).T  # using the first row of X for␣
 ↪demonstration purposes
y_pred = lr.predict(X_new)

# Print the predicted value
print(f"Predicted Median House Value using all features except CHAS:␣
 ↪${y_pred[0]:,.2f}")
```

```
Predicted Median House Value using all features except CHAS: $30.21
```

I think that comparison between the R^2 model is more accurate than comparing to all of the class.