Classification and Regression Testing

Ordinary Least Squares (OLS) using statsmodels

In this article, we will use Python's statsmodels module to implement Ordinary Least Squares(OLS) method of linear regression.

Introduction :

A linear regression model establishes the relation between a dependent variable(y) and at least one independent variable(x) as :

$$\hat{y} = b_1 x + b_0$$

In OLS method, we have to choose the values of b_1 and b_0 such that, the total sum of squares of the difference between the calculated and observed values of y, is minimised.

Formula for OLS:

$$S = \sum_{i=1}^{n} (y_i - \hat{y_i})^2 = \sum_{i=1}^{n} (y_i - b_1 x_1 - b_0)^2 = \sum_{i=1}^{n} (\hat{\epsilon_i})^2 = min$$

Where,

\hat{y_i} = predicted value for the ith observation

y_i = actual value for the ith observation

\epsilon_i = error/residual for the ith observation

n = total number of observations

To get the values of b_0 and b_1 which minimise S, we can take a partial derivative for each coefficient and equate it to zero.

The Algorithm:

Syntax : statsmodels.api.OLS(y, x)

Parameters :

- ***y : the variable which is dependent on x***
- ***x : the independent variable***

1. First we define the variables **x** and **y**. In the example below, the variables are read from a *csv* file using *pandas*. The file used in the example can be downloaded here.
2. Next, We need to add the constant to the equation using the **add_constant()** method.
3. The **OLS()** function of the **statsmodels.api** module is used to perform OLS regression. It returns an OLS object. Then **fit()** method is called on this object for fitting the regression line to the data.
4. The **summary()** method is used to obtain a table which gives an extensive description about the regression results

```
import statsmodels.api as sm
import pandas as pd

# reading data from the csv
data = pd.read_csv('train.csv')

# defining the variables
x = data['x'].tolist()
y = data['y'].tolist()

# adding the constant term
x = sm.add_constant(x)

# performing the regression
# and fitting the model
result = sm.OLS(y, x).fit()

# printing the summary table
print(result.summary())
```

Description of some of the terms in the table :

- **R-squared :** the coefficient of determination. It is the proportion of the variance in the dependent variable that is predictable/explained
- **Adj. R-squared :** Adjusted R-squared is the modified form of R-squared adjusted for the number of independent variables in the model. Value of adj. R-squared increases, when we include extra variables which actually improve the model.
- **F-statistic :** the ratio of mean squared error of the model to the mean squared error of residuals. It determines the overall significance of the model.
- **coef :** the coefficients of the independent variables and the constant term in the equation.
- **t :** the value of t-statistic. It is the ratio of the difference between the estimated and hypothesized value of a parameter, to the standard error

Predicting values:
From the results table, we note the coefficient of x and the constant term. These values are substituted in the original equation and the regression line is plotted using *matplotlib*.

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np

# reading data from the csv
data = pd.read_csv('train.csv')

# plotting the original values
x = data['x'].tolist()
```

```
y = data['y'].tolist()
plt.scatter(x, y)

# finding the maximum and minimum
# values of x, to get the
# range of data
max_x = data['x'].max()
min_x = data['x'].min()

# range of values for plotting
# the regression line
x = np.arange(min_x, max_x, 1)

# the substituted equation
y = 1.0143 * x - 0.4618

# plotting the regression line
plt.plot(y, 'r')
plt.show()
```