What is the complete guide to linear regression in Python?

Authored by

stats writer

April 17, 2024

The complete guide to linear regression in Python is a comprehensive resource that provides a detailed explanation of the concept of linear regression and how it can be implemented using the Python programming language. This guide covers all the necessary steps involved in performing linear regression, from data preparation and visualization to building, evaluating, and interpreting the regression model. It also includes various techniques and tools that can be used to improve the accuracy and efficiency of the regression model. Whether you are new to linear regression or looking to enhance your skills, this guide serves as a valuable reference for understanding and applying linear regression in Python.

A Complete Guide to Linear Regression in Python

Linear regression is a method we can use to understand the relationship between one or more predictor variables and a response variable.

This tutorial explains how to perform linear regression in Python.

Example: Linear Regression in Python

Suppose we want to know if the number of hours spent studying and the number of prep exams taken affects the score that a student receives on a certain exam.

To explore this relationship, we can perform the following steps in Python to conduct a multiple linear regression.

Step 1: Enter the data.

First, we'll create a pandas DataFrame to hold our dataset:

```
import pandas as pd

#create data
df = pd.DataFrame({'hours': ,
'exams': ,
'score': })

#view data
df
```

	hours	exams	score
0	1	1	76
1	2	3	78
2	2	3	85
3	4	5	88
4	2	2	72
5	1	2	69
6	5	1	94
7	4	1	94
8	2	0	88
9	4	3	92
10	4	4	90

11 3 3 75

12 6 2 96

13 5 4 90

14 3 4 82

15 4 4 85

16 6 5 99

17 2 1 83

18 1 0 62

19 2 1 76

Step 2: Perform linear regression.

Next, we'll use the **OLS() function** from the statsmodels library to perform ordinary least squares regression, using "hours" and "exams" as the predictor variables and "score" as the response variable:

```
import statsmodels.api as sm

#define response variable
y = df

#define predictor variables
x = df]
```

```
#add constant to predictor variables
x = sm.add_constant(x)

#fit linear regression model
model = sm.OLS(y, x).fit()

#view model summary
print(model.summary())
```

```
OLS Regression Results
==============================================================================
Dep. Variable: score R-squared: 0.734
Model: OLS Adj. R-squared: 0.703
Method: Least Squares F-statistic: 23.46
Date: Fri, 24 Jul 2020 Prob (F-statistic): 1.29e-05
Time: 13:20:31 Log-Likelihood: -60.354
No. Observations: 20 AIC: 126.7
Df Residuals: 17 BIC: 129.7
Df Model: 2
Covariance Type: nonrobust
==============================================================================
coef std err t P>|t|
------------------------------------------------------------------------------
```

const 67.6735 2.816 24.033 0.000 61.733 73.614

hours 5.5557 0.899 6.179 0.000 3.659 7.453

exams -0.6017 0.914 -0.658 0.519 -2.531 1.327

==========

==========

Omnibus: 0.341 Durbin-Watson: 1.506

Prob(Omnibus): 0.843 Jarque-Bera (JB): 0.196

Skew: -0.216 Prob(JB): 0.907

Kurtosis: 2.782 Cond. No. 10.8

==========

==========

Step 3: Interpret the results.

Here is how to interpret the most relevant numbers in the output:

R-squared: 0.734. This is known as the coefficient of determination. It is the proportion of the variance in the response variable that can be explained by the predictor variables. In this example, 73.4% of the variation in the exam scores can be explained by the number of hours studied and the number of prep exams taken.

F-statistic: 23.46. This is the overall F-statistic for the regression model.

Prob (F-statistic): 1.29e-05. This is the p-value associated with the overall F-statistic. It tells us whether or not the regression model as a whole is statistically significant. In other words, it tells us if the two predictor variables combined have a statistically significant association with the response variable. In this case the p-value is less than 0.05, which indicates that the predictor variables "hours studied" and "prep exams taken" combined have a statistically significant association with exam score.

coef: The coefficients for each predictor variable tell us the average expected change in the response variable, assuming the other predictor variable remains constant. For example, for each additional hour spent studying, the average exam score is expected to increase by 5.56, assuming that prep exams taken remains constant.

We interpret the coefficient for the intercept to mean that the expected exam score for a student who studies zero hours and takes zero prep exams is 67.67.

P>|t|. The individual p-values tell us whether or not each predictor variable is statistically significant. We can see that "hours" is statistically significant (p = 0.00) while "exams" (p = 0.52) is not statistically significant at α = 0.05. Since "exams" is not statistically significant, we may end up deciding to remove it from the model.

Estimated regression equation: We can use the coefficients from the output of the model to create the following estimated regression equation:

exam score = 67.67 + 5.56*(hours) - 0.60*(prep exams)

We can use this estimated regression equation to calculate the expected exam score for a student, based on the number of hours they study and the number of prep exams they take. For example, a student who studies for three hours and takes one prep exam is expected to receive a score of 83.75:

Keep in mind that because prep exams taken was not statistically significant (p = 0.52), we may decide to remove it because it doesn't add any improvement to the overall model. In this case, we could perform simple linear regression using only hours studied as the

predictor variable.

Step 4: Check model assumptions.

Once you perform linear regression, there are several assumptions you may want to check to ensure that the results of the regression model are reliable. These assumptions include:

Assumption #1: There exists a linear relationship between the predictor variables and the response variable.

Check this assumption by generating a **residual plot** that displays the fitted values against the residual values for a regression model.

Assumption #2: Independence of residuals.

Check this assumption by performing a **Durbin-Watson Test**.

Assumption #3: Homoscedasticity of residuals.

Check this assumption by performing a **Breusch-Pagan Test**.

Assumption #4: Normality of residuals.

Check this assumption visually using a Q-Q plot.Check this assumption with formal tests like a Jarque-Bera Test or an Anderson-Darling Test.

Assumption #5: Verify that multicollinearity doesn't exist among predictor variables.

Check this assumption by calculating the VIF value of each predictor variable.

If these assumptions are met, you can be confident that the results of your multiple linear regression model are reliable.

You can find the complete Python code used in this tutorial here.