How can a Wald test be performed in Python?

Authored by

stats writer

June 23, 2024

A Wald test is a statistical hypothesis test used to determine the significance of a regression coefficient in a linear regression model. In order to perform a Wald test in Python, the first step is to import the necessary libraries such as Statsmodels or Scipy. Next, the data must be loaded into a dataframe and the regression model must be created. Then, the regression coefficient of interest must be specified and a null hypothesis must be defined. The Wald test function, provided by the libraries, can then be used to calculate the test statistic and its associated p-value. Based on the p-value, the null hypothesis can be accepted or rejected, indicating the significance of the regression coefficient. Overall, performing a Wald test in Python involves importing the necessary libraries, loading and preparing the data, creating the regression model, and using the built-in function to calculate the test statistic and make a decision on the null hypothesis.

Perform a Wald Test in Python

A Wald test can be used to test if one or more parameters in a model are equal to certain values.

This test is often used to determine if one or more predictor variables in a regression model are equal to zero.

We use the following null and alternative for this test:

H0: Some set of predictor variables are all equal to zero.HA: Not all predictor variables in the set are equal to zero.

If we fail to reject the null hypothesis, then we can drop the specified set of predictor variables from the model because they don't offer a statistically significant

improvement in the fit of the model.

The following example shows how to perform a Wald test in Python

Example: Wald Test in Python

For this example, we'll the famous mtcars dataset to fit the following multiple linear regression model:

mpg = β0 + β1disp + β2carb + β3hp + β4cyl

The following code shows how to fit this regression model and view the model summary:

```
import statsmodels.formula.apias smf
import pandas as pd
import io

#define dataset as string
mtcars_data="""model,mpg,cyl,disp,hp,drat,wt,qsec,vs,am,gear,carb
Mazda RX4,21,6,160,110,3.9,2.62,16.46,0,1,4,4
Mazda RX4 Wag,21,6,160,110,3.9,2.875,17.02,0,1,4,4
Datsun 710,22.8,4,108,93,3.85,2.32,18.61,1,1,4,1
Hornet 4 Drive,21.4,6,258,110,3.08,3.215,19.44,1,0,3,1
```

Hornet Sportabout,18.7,8,360,175,3.15,3.44,17.02,0,0,3,2
Valiant,18.1,6,225,105,2.76,3.46,20.22,1,0,3,1
Duster 360,14.3,8,360,245,3.21,3.57,15.84,0,0,3,4
Merc 240D,24.4,4,146.7,62,3.69,3.19,20,1,0,4,2
Merc 230,22.8,4,140.8,95,3.92,3.15,22.9,1,0,4,2
Merc 280,19.2,6,167.6,123,3.92,3.44,18.3,1,0,4,4
Merc 280C,17.8,6,167.6,123,3.92,3.44,18.9,1,0,4,4
Merc 450SE,16.4,8,275.8,180,3.07,4.07,17.4,0,0,3,3
Merc 450SL,17.3,8,275.8,180,3.07,3.73,17.6,0,0,3,3
Merc 450SLC,15.2,8,275.8,180,3.07,3.78,18,0,0,3,3
Cadillac Fleetwood,10.4,8,472,205,2.93,5.25,17.98,0,0,3,4
Lincoln Continental,10.4,8,460,215,3,5.424,17.82,0,0,3,4
Chrysler Imperial,14.7,8,440,230,3.23,5.345,17.42,0,0,3,4
Fiat 128,32.4,4,78.7,66,4.08,2.2,19.47,1,1,4,1
Honda Civic,30.4,4,75.7,52,4.93,1.615,18.52,1,1,4,2
Toyota Corolla,33.9,4,71.1,65,4.22,1.835,19.9,1,1,4,1
Toyota Corona,21.5,4,120.1,97,3.7,2.465,20.01,1,0,3,1
Dodge Challenger,15.5,8,318,150,2.76,3.52,16.87,0,0,3,2
AMC Javelin,15.2,8,304,150,3.15,3.435,17.3,0,0,3,2
Camaro Z28,13.3,8,350,245,3.73,3.84,15.41,0,0,3,4
Pontiac Firebird,19.2,8,400,175,3.08,3.845,17.05,0,0,3,2
Fiat X1-9,27.3,4,79,66,4.08,1.935,18.9,1,1,4,1
Porsche 914-2,26,4,120.3,91,4.43,2.14,16.7,0,1,5,2
Lotus Europa,30.4,4,95.1,113,3.77,1.513,16.9,1,1,5,2

```
Ford Pantera L,15.8,8,351,264,4.22,3.17,14.5,0,1,5,4
Ferrari Dino,19.7,6,145,175,3.62,2.77,15.5,0,1,5,6
Maserati Bora,15,8,301,335,3.54,3.57,14.6,0,1,5,8
Volvo 142E,21.4,4,121,109,4.11,2.78,18.6,1,1,4,2"""

#convert string to DataFrame
df = pd.read_csv(io.StringIO(mtcars_data), sep=",")

#fit multiple linear regression model
results = smf.ols('mpg ~ disp + carb + hp + cyl', df).fit()

#view regression model summary
results.summary()
```

```
coef std err t P>|t|
Intercept34.0216 2.523 13.482 0.000 28.844 39.199
disp -0.0269 0.011 -2.379 0.025 -0.050 -0.004
carb -0.9269 0.579 -1.601 0.121 -2.115 0.261
hp 0.0093 0.021 0.452 0.655 -0.033 0.052
cyl -1.0485 0.784 -1.338 0.192 -2.657 0.560
```

Next, we can use the wald_test() function from statsmodels to test if the regression coefficients for the predictor variables "hp" and "cyl" are both equal to zero.

The following code shows how to use this function in practice:

#perform Wald Test to determine if 'hp' and 'cyl' coefficients are both zeroprint(results.wald_test('(hp = 0, cyl = 0)'))

F test: F=array([]), p=0.41403001184235005, df_denom=27, df_num=2

From the output we can see that the of the test is 0.414.

Since this p-value is not less than .05, we fail to reject the null hypothesis of the Wald test.

This means we can assume the regression coefficients for the predictor variables "hp" and "cyl" are both equal to zero.

The following tutorials explain how to perform other common operations in Python: