How can the Goldfeld-Quandt Test be performed using Python?

Authored by

stats writer

June 26, 2024

The Goldfeld-Quandt Test is a statistical method used to determine if there is heteroscedasticity, or unequal variances, in a dataset. This test can be performed using Python by following these steps:

1. Import the necessary libraries such as statsmodels and scipy.
2. Load the dataset into Python.
3. Divide the dataset into two subsets based on a chosen independent variable.
4. Use the statsmodels library to run a linear regression on both subsets.
5. Calculate the squared residuals for each subset.
6. Use the scipy library to perform an F-test on the squared residuals.
7. Compare the p-value of the F-test to a chosen significance level.
8. If the p-value is lower than the significance level, reject the null hypothesis of equal variances and conclude that there is heteroscedasticity in the dataset.

By following these steps, the Goldfeld-Quandt Test can be easily performed in Python to detect potential heteroscedasticity in a dataset.

Perform the Goldfeld-Quandt Test in Python

The Goldfeld-Quandt test is used to determine if is present in a regression model.

Heteroscedasticity refers to the unequal scatter of at different levels of a in a regression model.

If heteroscedasticity is present, this violates one of the key that the residuals are equally scattered at each level of the response variable.

This tutorial provides a step-by-step example of how to perform the Goldfeld-Quandt test in Python.

Step 1: Create the Dataset

For this example, let's create the following pandas DataFrame that contains information about hours studied, prep exams taken, and final exam score received by 13 students in some class:

```
import pandas as pd

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

#view DataFrame
print(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

Step 2: Fit Linear Regression Model

Next, we'll fit a multiple linear regression model using hours and exams as the predictor variables and score as the response variable:

```
import statsmodels.apias sm

#define predictor and response variables
y = df
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: 0.718
Model: OLS Adj. R-squared: 0.661
Method: Least Squares F-statistic: 12.70
Date: Mon, 31 Oct 2022 Prob (F-statistic): 0.00180
Time: 09:22:56 Log-Likelihood: -38.618
No. Observations: 13 AIC: 83.24
Df Residuals: 10 BIC: 84.93
Df Model: 2
Covariance Type: nonrobust
==============================================================================
coef std err t P>|t|
------------------------------------------------------------------------------
const 71.4048 4.001 17.847 0.000 62.490 80.319
hours 5.1275 1.018 5.038 0.001 2.860 7.395
exams -1.2121 1.147 -1.057 0.315 -3.768 1.344
==============================================================================
Omnibus: 1.103 Durbin-Watson: 1.248
```

Prob(Omnibus): 0.576 Jarque-Bera (JB): 0.803
Skew: -0.289 Prob(JB): 0.669
Kurtosis: 1.928 Cond. No. 11.7
==========

Step 3: Perform the Goldfeld-Quandt test

Next, we will use the het_goldfeldquandt() function from statsmodels to perform the Goldfeld-Quandt test.

Note: The Goldfeld-Quandt test works by removing some number of observations located in the center of the dataset, then testing to see if the spread of residuals is different from the resulting two datasets that are on either side of the central observations.

Typically we choose to remove around 20% of the total observations. In this case, we can use the drop argument to specify that we'd like to remove 20% of observations:

```
#perform Goldfeld-Quandt test
sm.stats.diagnostic.het_goldfeldquandt(y, x, drop=0.2)
```

(1.757450540779035, 0.38270288684680076,

'increasing')

Here is how to interpret the output:

The test statistic is 1.757.The corresponding p-value is 0.383.Null (H0): Homoscedasticity is present.Alternative (HA): Heteroscedasticity is present.

Since the p-value is not less than 0.05, we fail to reject the null hypothesis.

We do not have sufficient evidence to say that heteroscedasticity is a problem in the regression model.

What To Do Next

If you fail to reject the null hypothesis of the Goldfeld-Quandt test then heteroscedasticity is not present and you can proceed to interpret the output of the original regression.

However, if you reject the null hypothesis, this means heteroscedasticity is present in the data. In this case, the standard errors that are shown in the output table of the regression may be unreliable.

There are a couple common ways that you can fix this issue, including:

1. Transform the response variable.

You can try performing a transformation on the response variable, such as taking of the response variable. Typically this can cause heteroscedasticity to go away.

2. Use weighted regression.

Weighted regression assigns a weight to each data point based on the variance of its fitted value. Essentially, this gives small weights to data points that have higher variances, which shrinks their squared residuals.

When the proper weights are used, weighted regression can eliminate the problem of heteroscedasticity.

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

How to Perform White's Test in Python