Learning Weighted Least Squares Regression with Python: A Practical Guide

Authored by

Mohammed looti

October 26, 2025

The Foundational Role of Homoscedasticity in OLS

A cornerstone assumption underpinning classical linear regression models, particularly the Ordinary Least Squares method, is that of **homoscedasticity**. This critical concept dictates that the variability of the residuals--the vertical distances between the observed data points and the predicted regression line--must be uniform across all values of the predictor variable. In simpler terms, the spread or **variance** of the errors should remain constant, irrespective of the magnitude of the independent variable being analyzed.

When the condition of **homoscedasticity** is satisfied, the Ordinary Least Squares (OLS) procedure yields the most desirable estimates: the Best Linear Unbiased Estimators (BLUE). This guarantees that the calculated regression coefficients are not only unbiased but also possess the smallest possible **variance** among all linear unbiased estimators. Achieving this efficiency is paramount because it ensures that statistical inferences--such as the construction of confidence intervals and the execution of hypothesis testing--can be performed with maximum reliability and precision.

Furthermore, the presence of **homoscedasticity** implies that every single observation in the dataset contributes equally to the final determination of the regression coefficients. This equal weighting prevents any specific range of data points from disproportionately influencing the model parameters due to unusually large or small error variance. This uniformity is crucial for producing robust and trustworthy regression analysis results, confirming the validity of the model's structure.

Identifying and Mitigating the Impact of Heteroscedasticity

However, this ideal scenario is not always met in real-world applications. When the constant **variance** of the residuals assumption is violated, the model is said to exhibit **heteroscedasticity**. This situation commonly manifests visually in residual plots as a non-uniform spread of errors, often taking on a recognizable fan or cone shape, where the error magnitude increases or decreases systematically with the predictor variable.

The presence of **heteroscedasticity** carries significant consequences for model interpretation. Although the regression coefficients estimated by OLS remain mathematically unbiased, they lose their efficiency. The most critical issue, however, lies in the standard errors of these coefficients becoming biased, which directly leads to inaccurate p-values. Consequently, it becomes challenging to accurately assess the statistical significance of the independent variables, potentially resulting in flawed conclusions about which variables truly drive the relationship.

When **heteroscedasticity** is detected, relying solely on standard OLS output can lead to unreliable findings and misguided predictive decisions. To restore the efficiency of the estimates and validate the statistical inference, we must employ alternative regression analysis techniques that

specifically account for the varying error variance. The most powerful and effective solution is **Weighted Least Squares (WLS) regression**, which is the primary focus of the subsequent steps in this tutorial.

Introducing Weighted Least Squares (WLS) Regression

Weighted Least Squares (WLS) regression is an advanced statistical methodology specifically engineered to effectively model data characterized by **heteroscedasticity**. In contrast to OLS, which assumes parity in the influence of every data point, WLS strategically assigns a unique weight to each observation. These weights are fundamentally defined as being inversely proportional to the estimated error variance of that observation, thereby ensuring that data points associated with higher measurement precision (lower variance) exert a stronger influence on the final model fit.

The core mechanism of WLS involves systematically "downweighting" observations that possess larger estimated error variance and "upweighting" those that exhibit smaller error variance. This sophisticated approach effectively corrects for the disparate levels of precision across the dataset. By prioritizing the most reliable data points, WLS ensures that the resulting regression coefficients are estimated with greater efficiency, guaranteeing the validity of the statistical inferences drawn from the model, despite the persistent presence of non-constant error.

This tutorial is structured to provide a practical, step-by-step walkthrough detailing how to execute **Weighted Least Squares regression** utilizing the robust capabilities of the Python programming language, particularly leveraging the specialized statsmodels library. We will first prepare a synthetic dataset, establish a baseline using a standard OLS model, and then apply WLS to demonstrate its superiority in improving model reliability and performance.

Step 1: Constructing the Dataset in Python

Our initial task involves setting up the necessary data infrastructure within Python, relying on the indispensable pandas library for efficient data manipulation and structuring. We will create a pandas DataFrame to house our sample data, which is designed to simulate a scenario common in research: examining the correlation between student study time and final exam performance.

The resulting DataFrame will feature two key columns: '**hours**', designated as our predictor variable representing study time, and '**score**', serving as the response variable reflecting the final exam results. We will generate data for 16 hypothetical students, ensuring a sufficient range of study hours and corresponding scores to clearly illustrate the potential patterns of **heteroscedasticity** that WLS is designed to correct.

The following Python code snippet executes the creation of this DataFrame and displays its initial

rows. This preliminary check is vital for quickly verifying the structure and content of our data before proceeding to the complex stages of regression analysis.

import pandas as pd

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

#view first five rows of DataFrame
print(df.head())

hours score
0 1 48
1 1 78
2 2 72
3 2 70
4 2 66

Step 2: Establishing the Baseline with Ordinary Least Squares (OLS)

With the data successfully loaded, the next logical step is to fit a standard Ordinary Least Squares (OLS) model. This initial model will serve as the crucial baseline against which we measure the performance and improvement delivered by the **Weighted Least Squares regression**. We utilize the statsmodels library in Python, specifying '**hours**' as the independent variable and '**score**' as the dependent variable.

The **OLS()** function within statsmodels is employed for this purpose. A necessary prerequisite before fitting the model is the addition of a constant term to our predictor variables using **sm.add_constant(X)**. This term is mathematically essential as it represents the intercept ($beta_0$) in the linear equation, ensuring the regression line is allowed to cross the y-axis at a point other than zero, thereby accurately modeling the underlying relationship.

Following the model fitting, we generate and display the comprehensive model summary. This output provides vital statistics regarding model quality and significance. Key metrics to scrutinize include the R-squared value, which quantifies the proportion of **variance** explained, the p-values associated with the coefficients, and diagnostic statistics such as the Durbin-Watson statistic and the Jarque-Bera test, which help in assessing the initial assumptions regarding residuals.

import statsmodels.api as sm

```
#define predictor and response variables
y = df
X = df

#add constant to predictor variables
X = sm.add_constant(X) # Corrected from x to X

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

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

OLS Regression Results

==========

===

Dep. Variable: score R-squared: 0.630

Model: OLS Adj. R-squared: 0.603

Method: Least Squares F-statistic: 23.80

Date: Mon, 31 Oct 2022 Prob (F-statistic): 0.000244

Time: 11:19:54 Log-Likelihood: -57.184

No. Observations: 16 AIC: 118.4

Df Residuals: 14 BIC: 119.9

Df Model: 1

Covariance Type: nonrobust

==========

===

coef std err t P>|t|

---

const 60.4669 5.128 11.791 0.000 49.468 71.465

hours 5.5005 1.127 4.879 0.000 3.082 7.919

==========

===

Omnibus: 0.041 Durbin-Watson: 1.910

Prob(Omnibus): 0.980 Jarque-Bera (JB): 0.268

Skew: -0.010 Prob(JB): 0.875

Kurtosis: 2.366 Cond. No. 10.5

From the printed summary, we note an initial R-squared value of **0.630**. This suggests that the OLS model explains 63% of the variation in exam scores based on study hours, indicating a

reasonable fit. However, model reliability hinges on the unverified assumption of **homoscedasticity**. If this assumption is violated, the standard errors and p-values presented here are compromised, regardless of the seemingly strong R-squared value. This potential flaw necessitates proceeding to the WLS method to obtain truly efficient estimates.

Step 3: Implementing Weighted Least Squares (WLS) for Efficiency

To correct for the potential bias introduced by **heteroscedasticity** and secure more efficient coefficient estimates, we move to implement **Weighted Least Squares regression**. The statsmodels library facilitates this with its dedicated **WLS()** function. The crucial part of WLS is defining appropriate weights; these weights are typically inversely proportional to the true, underlying error variance of each observation.

Since the true error variance is typically unknown, a common and effective procedure is to estimate it using the residuals derived from the initial OLS model. The methodology employed below involves running an auxiliary regression analysis where the absolute OLS residuals are regressed against the OLS fitted values. The calculated weights (wt) are then defined as the inverse of the squared fitted values from this auxiliary regression. This precise weighting scheme successfully diminishes the influence of observations with high error variance while amplifying the impact of more reliable, low-variance data points.

Once these customized weights are computed, we proceed to fit the WLS model using **sm.WLS()**, feeding in the response variable (y), the predictor variables (X), and the calculated weights (wt). The summary of this refined model is then printed, allowing a direct comparison with the baseline OLS results to quantify the improvement achieved by accounting for the variable error structure.

import statsmodels.formula.api as smf # Import for ols function

#define weights to use
wt = 1 / smf.ols('fit.resid.abs() ~ fit.fittedvalues', data=df).fit().fittedvalues**2

#fit weighted least squares regression model
fit_wls = sm.WLS(y, X, weights=wt).fit()

#view summary of weighted least squares regression model
print(fit_wls.summary())

WLS Regression Results
==========
===
Dep. Variable: score R-squared: 0.676
Model: WLS Adj. R-squared: 0.653

Method: Least Squares F-statistic: 29.24
Date: Mon, 31 Oct 2022 Prob (F-statistic): 9.24e-05
Time: 11:20:10 Log-Likelihood: -55.074
No. Observations: 16 AIC: 114.1
Df Residuals: 14 BIC: 115.7
Df Model: 1
Covariance Type: nonrobust
==========
===
coef std err t P>|t|
---
const 63.9689 5.159 12.400 0.000 52.905 75.033
hours 4.7091 0.871 5.407 0.000 2.841 6.577
==========
===
Omnibus: 2.482 Durbin-Watson: 1.786
Prob(Omnibus): 0.289 Jarque-Bera (JB): 1.058
Skew: 0.029 Prob(JB): 0.589
Kurtosis: 1.742 Cond. No. 17.6
==========
===

Analyzing WLS Results and Model Improvement

A thorough analysis of the **Weighted Least Squares model** summary reveals clear evidence of enhanced performance compared to the initial OLS baseline. Most notably, the model's R-squared value increased substantially to **0.676**, marking an improvement over the OLS value of 0.630. This quantifiable increase signifies that the WLS model is now capable of explaining a greater proportion of the total **variance** observed in the exam scores, which translates to a superior overall fit to the data distribution.

Beyond the coefficient of determination, the paramount benefit of WLS lies in its ability to produce more efficient and statistically reliable estimates for the coefficients and their standard errors. By directly mitigating the effects of **heteroscedasticity** through strategic weighting, the model ensures that the statistical inferences--such as the calculated p-value for the 'hours' coefficient, which remains highly significant--are trustworthy and robust. The improved precision solidifies the strong, positive relationship between study hours and exam scores.

The shift from a standard OLS approach to employing WLS demonstrates a practical method for

enhancing model validity when the assumption of **homoscedasticity** is violated. By appropriately weighting observations, we gain a more accurate understanding of the relationships within our data, leading to more robust statistical conclusions.

Conclusion: Mastering Robust Regression Techniques

This comprehensive tutorial has successfully navigated the essential concepts of **homoscedasticity** in linear regression and provided a practical framework for addressing its violation, known as **heteroscedasticity**, using **Weighted Least Squares (WLS) regression** within the Python environment. We executed a clear sequence: data preparation, OLS baseline fitting, and finally, the rigorous implementation of WLS via the careful construction of weights based on estimated error variance.

The empirical evidence confirmed the advantages of the WLS approach. The observed increase in the R-squared value from 0.630 to 0.676 demonstrated a superior model fit. Crucially, WLS delivered more efficient and reliable estimates of the coefficients and their standard errors, ensuring that our statistical inferences are valid, even in the presence of non-constant error variance.

For any professional engaging in statistical modeling or data science, mastering techniques like WLS is paramount. By diligently recognizing and correcting violations of core assumptions, we build more trustworthy regression models, leading to more accurate insights and facilitating better-informed decision-making based on analyzed data. This methodology is a fundamental skill for advanced statistical analysis in Python.

Additional Resources

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