Calculate Standardized Residuals in Python

Authored by

Mohammed looti

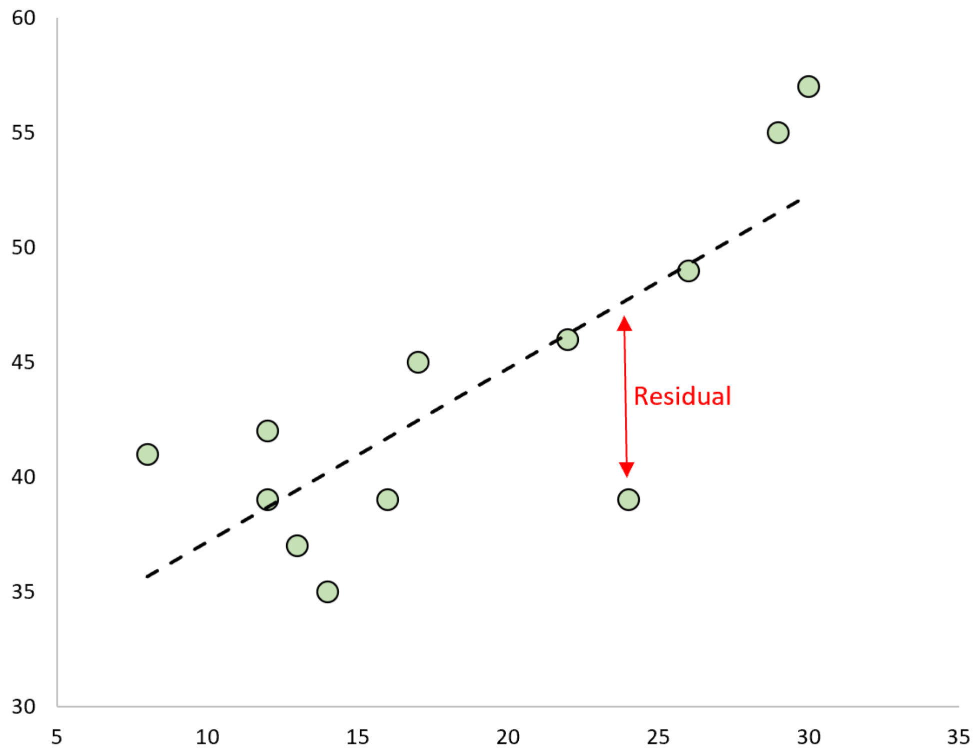
November 6, 2025

A **residual** represents the fundamental difference between an observed data point and the value predicted by a statistical **regression model**. Understanding residuals is critical for assessing the overall fit and validity of any predictive model.

Mathematically, the residual for a given observation is calculated simply as:

Residual = Observed Value - Predicted Value

When visualizing a fitted regression line overlaid on the original data, the residual for each data point is the vertical distance separating that point from the line of best fit. This visual representation helps to illustrate the model's error for individual observations:



Understanding Standardized Residuals

While raw residuals are useful, they do not provide a standardized measure of error that accounts for variability across the dataset. This is where the **standardized residual** becomes an essential diagnostic tool. Standardized residuals normalize the error by dividing the raw residual by an estimate of its standard deviation.

This standardization is crucial because it allows us to compare residuals across different observations and models, making it far easier to identify influential data points or potential **outliers**.

The general formula for the standardized residual is expressed as:

ri = ei / s(ei) = **ei / RSE√1-hii**

In this formula, the components are defined as follows:

ei: The ith raw **residual**.
RSE: The **Residual Standard Error** of the model, which estimates the standard deviation of the error term.
hii: The **leverage** of the ith observation, which measures how far the observation's explanatory variable value is from the mean of the explanatory variable values.

In statistical practice, a common rule of thumb is that any standardized residual with an absolute value greater than 3 is generally considered a significant **outlier** that warrants further investigation, as it suggests the observation is three or more standard deviations away from the fitted line. This tutorial demonstrates the precise steps to calculate these residuals using **Python**.

Step 1: Define and Prepare the Data

Our first step is to establish the necessary data structure within the **Python** environment. We will utilize the powerful `pandas` library to create and manage our dataset, which consists of a predictor variable (x) and a response variable (y).

This small, illustrative dataset will serve as the foundation for fitting our linear **regression model** and calculating the subsequent residuals.

import pandas as pd

```
#create dataset
df = pd.DataFrame({'x': ,
'y': })
```

Step 2: Fit the Linear Regression Model using Statsmodels

With the data prepared, the next phase involves fitting the appropriate statistical model. We rely on the `statsmodels` library, a robust package in **Python** designed for estimating statistical models and performing statistical tests.

We must first clearly define the response variable (y) and the explanatory variable (x). Crucially, for **Ordinary Least Squares (OLS)** regression, we must use the `sm.add_constant(x)` function to include the intercept term in the model calculation.

import statsmodels.api as sm

#define response variable
y = df

#define explanatory variable
x = df

#add constant to predictor variables (for the intercept)
x = sm.add_constant(x)

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

Step 3: Calculate and Interpret the Standardized Residuals

Once the **regression model** is fitted, we can access diagnostic statistics, including the **standardized residuals**. The `statsmodels` package provides the `get_influence()` method, which is specifically designed to calculate various influence statistics needed for model diagnostics.

We use the `resid_studentized_internal` attribute from the influence object to retrieve the standardized residuals (often interchangeably referred to as internally studentized residuals). These values quantify how many standard deviations each observation's **residual** is away from zero.

#create instance of influence object
influence = model.get_influence()

#obtain standardized residuals (internally studentized)
standardized_residuals = influence.resid_studentized_internal

#display standardized residuals
print(standardized_residuals)

Upon reviewing the output, we observe that none of the standardized residuals have an absolute value exceeding 3. Based on the conventional statistical benchmark, this suggests that none of the observations in this dataset qualify as extreme **outliers** that significantly distort the model fit.

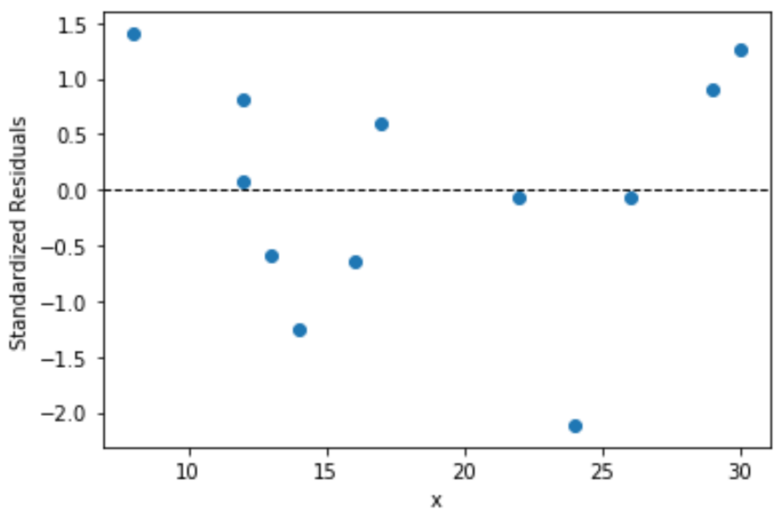
Step 4: Visualize the Standardized Residuals for Diagnostic Checks

A crucial component of **regression model** diagnostics is the visualization of the residuals. Plotting the standardized residuals against the predictor variable (x) allows us to visually check key assumptions, particularly the assumption of homoscedasticity (constant variance of errors).

A well-fitting model should display residuals scattered randomly around the zero horizontal line without any clear pattern (such as a cone shape or a curve). We use `matplotlib` in **Python** to generate this diagnostic plot.

import matplotlib.pyplot as plt

```
plt.scatter(df.x, standardized_residuals)
plt.xlabel('x')
plt.ylabel('Standardized Residuals')
plt.axhline(y=0, color='black', linestyle='--', linewidth=1)
plt.show()
```



The resulting scatterplot visually confirms our numerical findings from Step 3: the points are scattered randomly around the zero line, indicating that the assumption of constant error variance is likely met and that there are no obvious structural issues or problematic **outliers** skewing the **standardized residual** distribution.

Additional Resources for Statistical Diagnostics

For those interested in delving deeper into statistical diagnostic methods, the following resources provide further context on **standardized residuals** and related concepts.

What Are Standardized Residuals?