How to Easily Calculate SST, SSR, and SSE in Python

Authored by

stats writer

December 2, 2025

ARABPSYCHOLOGY.COM

RECOMMENDED CITATION

stats writer (2025). *How to Easily Calculate SST, SSR, and SSE in Python*. PSYCHOLOGICAL SCALES. Retrieved from https://scales.arabpsychology.com/?p=103697

The evaluation of a statistical model's performance hinges on its ability to explain the variance present in the observed data. In the context of linear regression, three fundamental metrics--the Sum of Squares Total (SST), the Sum of Squares Regression (SSR), and the Sum of Squares Error (SSE)--are used to quantify this performance. These values decompose the total variability of the response variable into parts explained by the model and parts attributed to residual error.

Calculating **SST**, **SSR**, and **SSE** is essential for deriving metrics like the coefficient of determination (R-squared), which indicates how well the linear regression model fits the data. While specialized libraries often provide these outputs directly, understanding the underlying manual calculation using tools like Python, `numpy`, and `statsmodels` offers deeper insight into the mechanics of model diagnostics.

Understanding the Key Metrics: SST, SSR, and SSE

To accurately measure how well a regression model fits a dataset, we utilize a decomposition technique based on squared differences. This approach minimizes the impact of positive and negative deviations canceling each other out, providing a robust measure of overall scatter.

The relationship between these three sums of squares is mathematically defined as: **SST = SSR + SSE**. This decomposition allows statisticians and data scientists to assess the efficacy of the predictive variables (SSR) against the inherent noise or unexplained variance (SSE), relative to the total variance (SST).

Below is a detailed breakdown of each component and its role in evaluating model fit:

Sum of Squares Total (SST)

SST represents the total variation in the dependent variable (Y) around its mean ($bar{y}$). It acts as the benchmark--the amount of variance that must be explained. If the model fails to explain this variance, it performs no better than simply predicting the mean for every observation. This metric is independent of the regression model itself.

SST = Σ(yi - y)2

Sum of Squares Regression (SSR)

SSR, also known as the Sum of Squares Explained, measures the portion of the total variance in the dependent variable that is successfully captured by the regression model. It is the sum of squared differences between the predicted data points ($hat{y}_{i}$) and the mean of the response variable ($bar{y}$). A high SSR relative to SST indicates a strong model fit.

SSR = Σ($hat{y}_{i}$ - y)2

ARABPSYCHOLOGY.COM

Sum of Squares Error (SSE)

SSE, or the Residual Sum of Squares, represents the unexplained variation in the dependent variable. It is the sum of squared differences between the observed data points (y_{i}) and the predicted data points ($hat{y}_{i}$). This value reflects the magnitude of the errors inherent in the model's predictions. The goal in fitting a regression model is typically to minimize **SSE**.

SSE = Σ($hat{y}_{i}$ - yi)2

The following step-by-step example demonstrates how to implement these statistical concepts and calculate each of these crucial metrics for a specific regression model using statsmodels and `numpy` in a Python environment.

Step 1: Preparing the Dataset in Python

Before fitting any model, quality data preparation is mandatory. We begin by creating a sample dataset that simulates a common scenario: examining the relationship between study time and exam performance. This dataset includes the number of hours studied and the resulting exam score for twenty hypothetical students.

We utilize the **Pandas** library to structure this data efficiently into a `DataFrame`, which is the standard format for data manipulation and analysis in Python. This structure allows us to easily isolate the predictor variable ('hours') and the response variable ('score').

The code below sets up our data, ensuring it is ready for the regression analysis in the subsequent steps.

import pandas as pd

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

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

hours score
0 1 68
1 1 76
2 1 74
3 2 80
4 2 76

ARABPSYCHOLOGY.COM

Step 2: Fitting the Simple Linear Regression Model

With the data structured, the next phase involves fitting a simple linear regression model. For this task, we employ the `statsmodels` library, specifically its **OLS()** (Ordinary Least Squares) function. OLS is the most common method for estimating the parameters of a linear regression model, minimizing the sum of the squared residuals (SSE).

In our model definition, 'score' serves as the **response variable (Y)**, which we aim to predict, and 'hours' serves as the **predictor variable (X)**. A critical step when using `statsmodels` is adding a constant term (the intercept) to the predictor variables using `sm.add_constant(x)`. This ensures the regression line is not forced to pass through the origin (0, 0).

The `.fit()` method then executes the OLS estimation, producing a model object containing all necessary statistical information, including the fitted values ($hat{y}_{i}$) required for the sums of squares calculations.

import statsmodels.api as sm

#define response variable
y = df

#define predictor variable
x = df]

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

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

Step 3: Calculating Sums of Squares Using NumPy

Once the model is fitted, we can access the necessary components--the observed scores, the predicted scores (`model.fittedvalues`), and the mean score (`df.score.mean()`)--to calculate **SST**, **SSR**, and **SSE** manually. This step primarily leverages the efficient array manipulation capabilities provided by the numpy library.

Calculating SSE (Sum of Squares Error)

To find the SSE, we compute the squared difference between the predicted values (`model.fittedvalues`) and the actual observed scores (`df.score`). This quantity represents the

ARABPSYCHOLOGY.COM

residual variance that the model cannot explain.

```
import numpy as np

#calculate sse: Sum(Observed - Predicted)^2
sse = np.sum((df.score - model.fittedvalues)**2)
print(sse)
```

331.07488479262696

Calculating SSR (Sum of Squares Regression)

The SSR is calculated by summing the squared differences between the predicted values (`model.fittedvalues`) and the overall mean of the response variable (`df.score.mean()`). This reflects the variance explained by the regression line.

```
#calculate ssr: Sum(Predicted - Mean)^2
ssr = np.sum((model.fittedvalues - df.score.mean())**2)
print(ssr)
```

917.4751152073725

Calculating SST (Sum of Squares Total)

The **SST** can be calculated directly by summing the squared differences between the observed scores and the mean score, or, more simply and as a verification check, by adding the calculated SSR and SSE values.

```
#calculate sst: SSR + SSE
sst = ssr + sse
print(sst)
```

1248.5499999999995

Interpreting the Results: Model Decomposition

The calculations provide a clear quantitative decomposition of the total variation in the exam scores. By examining the magnitudes of **SSR** and **SSE** relative to the **SST**, we can assess the goodness-of-fit of our model.

Summarized results:

ARABPSYCHOLOGY.COM

Sum of Squares Total (SST): 1248.55
Sum of Squares Regression (SSR): 917.4751
Sum of Squares Error (SSE): 331.0749

Since SSR (917.4751) is significantly larger than SSE (331.0749), the majority of the total variability in scores is successfully explained by the hours studied (the regression component). Only a smaller portion remains as residual error.

We can confirm the fundamental relationship of the sums of squares:

SST = SSR + SSE
1248.55 ≈ 917.4751 + 331.0749

This confirmation validates the calculations and provides the necessary inputs for calculating the R-squared value, which is simply $R^2 = SSR / SST$. In this case, $R^2 approx 917.4751 / 1248.55 approx 0.735$, indicating that roughly 73.5% of the variance in exam scores is explained by the variable 'hours studied'.

Conclusion and Further Resources

Understanding and manually calculating the Sums of Squares (SST, SSR, and SSE) is fundamental to statistical modeling and diagnostics. These measures provide the bedrock for determining model utility and calculating key metrics like R-squared, ensuring transparency in evaluating the predictive power of a linear regression model.

While libraries often automate these calculations, the manual implementation using NumPy reinforces the statistical principles at play. For readers interested in exploring automated calculation methods or applying these concepts in other statistical environments, the following resources may be helpful:

You can use the following specialized calculators to automatically determine SST, SSR, and SSE for any simple linear regression line:

The following tutorials explain how to calculate SST, SSR, and SSE in other statistical software:

ARABPSYCHOLOGY.COM