Learning to Use the coeftest() Function for Statistical Significance Testing in R

Authored by

Mohammed looti

May 5, 2026

RECOMMENDED CITATION

Mohammed looti (2026). *Learning to Use the coeftest() Function for Statistical Significance Testing in R*. PSYCHOLOGICAL STATISTICS. Retrieved from https://statistics.arabpsychology.com/?p=3553

When conducting statistical analyses in **R**, particularly when dealing with **regression models**, it is fundamentally important to assess the statistical significance of each **estimated coefficient**. Determining which factors truly drive the outcome is crucial for creating valid and interpretable models. The **lmtest** package in R offers a specialized and powerful utility, the **coeftest()** function, designed precisely for this task. This function allows researchers and analysts to rigorously perform a **t-test** for every coefficient, providing indispensable insights into the true relationship between **predictor variables** and the **response variable**.

Understanding the significance of individual coefficients is paramount to building robust models that stand up to scrutiny. A significant coefficient suggests that the corresponding predictor has a non-zero impact on the response, while a non-significant coefficient may indicate that the predictor is negligible or redundant in the current model structure. This comprehensive guide will walk you through the essential steps of utilizing **coeftest()**, covering everything from the necessary setup and basic syntax to interpreting the final statistical results in a practical data analysis context.

Core Functionality and Parameters of coeftest()

The **coeftest()** function is an essential component of the **lmtest** package, which is widely recognized for its capabilities in diagnostic checking and testing linear regression models. While standard model summaries in **R** already provide coefficient tests, **coeftest()** is often preferred because it allows for the easy application of various types of robust **standard errors**, such as heteroskedasticity-consistent estimators, which are necessary when model assumptions are violated. Its core role is to efficiently re-estimate these standard errors and subsequently calculate the t-statistics and **p-values** for the coefficients of a fitted model object.

For most standard applications, the basic syntax for calling the **coeftest()** function is remarkably simple, requiring only the fitted regression model object as its main argument:

coeftest(x)

The primary parameter used in this function is defined as follows:

x: This required argument is the name of your **fitted regression model** object. Typically, this object will be the result of model-fitting functions in **R**, such as the **lm()** function for linear models or **glm()** for generalized linear models. **coeftest()** automatically extracts all necessary structural information from this object to correctly execute the coefficient tests.

The simplicity of this interface streamlines the analytical process, enabling quick and accurate assessments of individual coefficient significance. It is important to remember that before you can utilize **coeftest()**, the **lmtest** package must be both installed on your system and loaded into your current **R** session using the `library()` command.

Setting Up the Analysis: Data Preparation in R

To demonstrate the practical utility of the **coeftest()** function, we will construct a realistic scenario involving educational data. Let us assume we are investigating the key factors that influence a group of students' final exam scores. Our collected data includes the final **score** achieved by each student, the total number of **hours** they spent studying, and the number of **prac_exams** (practice exams) they completed. This data must first be organized into a suitable **R data frame** for analysis.

The following **R** code snippet creates this sample **data frame**, providing 10 observations that link the outcome variable (score) to the two **predictor variables** (hours and prac_exams):

```
#create data frame
df <- data.frame(score=c(77, 79, 84, 85, 88, 99, 95, 90, 92, 94),
hours=c(1, 1, 2, 3, 2, 4, 4, 2, 3, 3),
prac_exams=c(2, 3, 3, 2, 4, 5, 4, 3, 5, 4))

#view data frame
df
```

score hours prac_exams
1 77 1 2
2 79 1 3
3 84 2 3
4 85 3 2
5 88 2 4
6 99 4 5
7 95 4 4
8 90 2 3
9 92 3 5
10 94 3 4

This structured **data frame** establishes the necessary groundwork for our subsequent **regression analysis**. By examining this data, we aim to quantify precisely how studying hours and the completion of practice exams jointly predict the variability observed in the final scores. The next analytical step involves defining and fitting the appropriate statistical model using this prepared dataset.

Constructing and Fitting the Multiple Regression Model

Following the preparation of our data, the next critical step is to fit a **multiple linear regression**

model. This model type is the standard choice when the goal is to predict a continuous **response variable** (the score) using the influence of two or more **predictor variables** simultaneously (hours and practice exams).

The underlying conceptual model that guides this analysis is mathematically formulated as:

Exam score = β0 + β1(hours) + β2(practice exams) + ε

In this equation, β0 represents the model's intercept, while β1 and β2 are the **coefficients** for 'hours' and 'practice exams', respectively. These coefficients quantify the expected change in the exam score for a one-unit increase in the corresponding predictor, assuming all other predictors remain constant. To estimate these parameters, we employ the powerful **lm()** function, which is the standard workhorse for linear modeling in **R**:

```
#fit multiple linear regression model
fit <- lm(score ~ hours + prac_exams, data=df)
```

The resulting `fit` object encapsulates all the statistical information derived from our **multiple linear regression model**, including the calculated **estimated coefficients** and their associated **standard errors**. This object serves as the necessary and direct input for the **coeftest()** function, allowing us to proceed with the rigorous testing of individual coefficient significance.

Executing coeftest() and Deciphering the Results Table

With the **regression model** successfully fitted using the **lm()** function, the next step is to perform the individual **t-test** for each **estimated coefficient** using **coeftest()**. Before execution, it is critical to ensure the **lmtest** package has been loaded into the active **R** environment.

The following code executes the test and displays the output summary:

```
library(lmtest)

#perform t-test for each coefficient in model
coeftest(fit)
```

t test of coefficients:

Estimate Std. Error t value Pr(>|t|)
(Intercept) 68.40294 2.87227 23.8150 5.851e-08 ***
hours 4.19118 0.99612 4.2075 0.003998 **
prac_exams 2.69118 0.99612 2.7017 0.030566 *

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

The output generated by **coeftest()** provides a clear and comprehensive summary for every term in the model, including the intercept. Understanding the meaning of each column is vital for statistical interpretation:

Estimate: This column displays the precise **estimated value** for the coefficient (or slope). For instance, the estimate of 4.19118 for 'hours' suggests that, holding 'prac_exams' constant, a one-hour increase in study time is associated with an average increase of 4.19 points in the final exam score.
Std. Error: This is the **standard error** of the estimate, which is a measure of the precision and reliability of the estimated coefficient. Smaller standard errors indicate that the estimate is more precise.
t value: This represents the calculated **t-statistic**. It is computed as the ratio of the coefficient Estimate to its **Standard Error**, quantifying how many standard deviations the estimate is away from zero.
Pr(>|t|): This is the crucial **p-value** associated with the two-sided **t-test**. It represents the probability of observing a **t-statistic** of this magnitude, or greater, if the true coefficient were actually zero (**null hypothesis**).
Signif. codes: These symbols provide a quick visual cue regarding the magnitude of the **p-value**, correlating directly to standard **significance levels** (e.g., *** for high significance).

Based on our example, the resulting individual **t-test** statistics are clearly displayed: t = 4.2075 (p = 0.003998) for 'hours' and t = 2.7017 (p = 0.030566) for 'prac_exams'. These figures are essential for moving to the next stage: formal statistical inference.

Statistical Inference: Hypothesis Testing and Significance

The core purpose of the **t-test** results provided by **coeftest()** is to rigorously test specific hypotheses concerning each **regression coefficient** (βi). For every predictor in our model, we establish a formal pair of competing hypotheses:

H0 (Null Hypothesis): βi = 0. This hypothesis posits that the **predictor variable** has no statistically significant linear relationship with the **response variable**; in practical terms, its slope is zero.
HA (Alternative Hypothesis): βi ? 0. This hypothesis suggests that the **predictor variable** does indeed have a statistically significant relationship with the **response variable**, meaning its true coefficient is not zero.

The decision rule for hypothesis testing relies on comparing the **p-value** derived from the

coeftest() output against a predefined **significance level** (α), which is almost universally set at 0.05. If the calculated **p-value** is smaller than α, we possess sufficient evidence to reject the **null hypothesis**. Rejecting H0 allows us to confidently conclude that the corresponding **coefficient** is statistically significant, confirming the predictor's meaningful contribution to the model.

Applying this rule to our student exam score example, we observe that all **p-values** for the predictors (0.003998 for hours and 0.030566 for prac_exams) are substantially below the chosen **significance level** of 0.05. Consequently, we must reject the **null hypothesis** for both study hours and practice exams. This statistical conclusion affirms that both 'hours spent studying' and the 'number of practice exams taken' are statistically significant, independent predictors of the final exam score in this **multiple linear regression model**.

Summary and Resources for Advanced Analysis

The **coeftest()** function, housed within the robust **lmtest** package, is an indispensable asset for any data scientist or analyst performing **regression analysis** in **R**. It provides a standardized, clear, and efficient methodology for conducting **t-tests** on individual **regression coefficients**, thereby facilitating the crucial process of statistical inference. By systematically examining the **p-values** generated by this function, analysts can confidently determine the statistical significance of each **predictor variable**, leading to a deeper and more trustworthy understanding of the underlying data relationships.

In our practical example, we successfully demonstrated how to utilize **coeftest()** to confirm that both study hours and practice exams significantly contributed to predicting students' final scores. This type of rigorous, coefficient-level testing is paramount for extracting reliable conclusions from statistical models and is foundational for informing any evidence-based decisions in research or business intelligence.

For users interested in advancing their knowledge of linear regression diagnostics and related tools in **R**, particularly concerning model robustness and assumption checking, the following resources are highly recommended for further exploration:

The official **lmtest** package documentation, available on CRAN, which details advanced usage and parameters.
Comprehensive tutorials and academic guides focused on **multiple linear regression** techniques in R environments.
Resources dedicated to understanding and calculating robust **standard errors** in regression, which can be seamlessly integrated with **coeftest()** using its specialized `vcov` argument.