How to Easily Extract Regression Coefficients from glm() in R

Authored by

stats writer

November 20, 2025

The R function `glm()` is the standard tool for fitting Generalized Linear Models (GLM), which are essential for analyzing response variables that follow distributions other than the Gaussian (normal) distribution. Once a model is fitted, the most critical outputs are the Regression Coefficients. These values quantify the relationship between the predictors and the response variable on the scale of the linear predictor. Successfully extracting, interpreting, and reporting these coefficients requires knowledge of specific indexing techniques within R.

While the basic function `coef()` or `model$coefficients` provides the estimated parameter values, it often lacks the crucial diagnostic statistics necessary for inference, such as the Standard Error and the corresponding P-Value. For a complete inferential report, the `summary()` command is necessary. This article details the robust and clean methods available in R for extracting precisely the information required, whether it is a single coefficient estimate or the full coefficient matrix containing all inferential metrics.

Core Methods for Extracting Regression Coefficients

There are several powerful, yet simple, methods to retrieve the estimated parameters resulting from a fitted `glm()` object. The choice of method depends on whether you need only the parameter estimate itself or the complete statistical package (standard errors, z-statistics, and p-values) required for formal inference.

We outline three fundamental approaches for coefficient extraction from a stored model object, typically named `model`.

Method 1: Extract All Regression Coefficient Estimates

This method provides a named vector containing only the point estimates for all predictors included in the model. This is the fastest way to access the raw coefficient values without any associated statistics like standard errors or p-values. It directly accesses the `coefficients` component stored within the `glm` object.

model$coefficients

Method 2: Extract Regression Coefficient for a Specific Variable

By combining the previous method with array indexing, we can retrieve the specific coefficient estimate associated with a single predictor variable. This is useful when focusing on the marginal effect of one variable, such as `my_variable`, within the larger model structure, allowing for quick verification of a single parameter estimate.

model$coefficients

Method 3: Extract All Regression Coefficients with Standard Error, Z Value & P-Value

For rigorous statistical reporting, you must obtain the full coefficient matrix. This matrix provides the Estimate, Standard Error, Z-Statistic (or Wald statistic), and the corresponding P-Value. This comprehensive output is accessed by applying the `summary()` function to the model object and then extracting the `coefficients` component from the resulting summary object.

summary(model)$coefficients

Analyzing the Components of the Coefficient Matrix

When employing Method 3, the output is a detailed matrix that forms the basis of statistical inference for the GLM. Understanding each column is crucial for correctly interpreting the model results and determining the statistical significance of each predictor variable.

Estimate: This is the predicted change in the linear predictor for a one-unit increase in the predictor variable, holding all other variables constant. In a logistic model, this represents the change in the log-odds of the outcome. The coefficient value indicates the direction and magnitude of the relationship.

Std. Error: The Standard Error measures the precision of the coefficient estimate. It is the estimated standard deviation of the sampling distribution of the coefficient. A smaller standard error indicates a more precise estimate and greater confidence in the observed magnitude of the Regression Coefficients. This value is used directly in calculating the Z-Statistic.

Z Value: This is the Wald Z-statistic, calculated by dividing the Estimate by the Standard Error. It tests the null hypothesis that the true coefficient is zero. A large absolute Z-value suggests strong evidence against the null hypothesis, indicating that the predictor has a significant impact on the outcome.

Pr(>|z|): This is the P-Value associated with the Z-Statistic. It represents the probability of observing a coefficient estimate as extreme as, or more extreme than, the one calculated, assuming the null hypothesis (that the true coefficient is zero) is true. If this value is below a chosen significance level (e.g., 0.05), the coefficient is deemed statistically significant and we reject the null hypothesis.

Practical Demonstration: Using the ISLR Default Dataset

We will now demonstrate how to use these extraction methods in practice using a real-world scenario involving binary classification. Suppose we are fitting a logistic regression model to predict the probability of a customer defaulting on their debt based on their student status, credit

card balance, and annual income. We will utilize the **Default** dataset provided within the ISLR package.

We begin by loading the necessary data and fitting a GLM using the binomial family, which is the link function appropriate for binary outcomes (logistic regression).

#load dataset
data <- ISLR::Default

#view first six rows of data
head(data)

default student balance income
1 No No 729.5265 44361.625
2 No Yes 817.1804 12106.135
3 No No 1073.5492 31767.139
4 No No 529.2506 35704.494
5 No No 785.6559 38463.496
6 No Yes 919.5885 7491.559

#fit logistic regression model
model <- glm(default~student+balance+income, family='binomial', data=data)

#view summary of logistic regression model
summary(model)

Call:
glm(formula = default ~ student + balance + income, family = "binomial",
data = data)

Deviance Residuals:
Min 1Q Median 3Q Max
-2.4691 -0.1418 -0.0557 -0.0203 3.7383

Coefficients:
Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.087e+01 4.923e-01 -22.080 < 2e-16 ***
studentYes -6.468e-01 2.363e-01 -2.738 0.00619 **
balance 5.737e-03 2.319e-04 24.738 < 2e-16 ***
income 3.033e-06 8.203e-06 0.370 0.71152

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

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 2920.6 on 9999 degrees of freedom
Residual deviance: 1571.5 on 9996 degrees of freedom
AIC: 1579.5

Number of Fisher Scoring iterations: 8

Applying Extraction Methods for Estimates Only

To obtain only the estimated values of the Regression Coefficients, excluding all other statistics generated by the model summary, we utilize the `model$coefficients` command. This returns a vector showing the log-odds change associated with each variable, confirming the exact parameter estimates used in the model calculation.

#extract all regression coefficients
model$coefficients

(Intercept) studentYes balance income
-1.086905e+01 -6.467758e-01 5.736505e-03 3.033450e-06

If the analytical goal requires isolation of the coefficient for a specific variable, such as the continuous predictor `balance`, we can apply indexing directly to the resulting vector. This is achieved by placing the variable name (as a string) in square brackets immediately after the `model$coefficients` call. This is a crucial technique for automating the retrieval of specific model parameters.

#extract coefficient for 'balance'
model$coefficients

balance
0.005736505

Extracting and Indexing the Full Diagnostic Matrix

To view the full set of Regression Coefficients along with their corresponding diagnostic statistics--the Standard Error, Z-value, and P-Value--we use the `summary(model)$coefficients` structure. This method yields a matrix, which is the standard output for inferential reporting in R.

#view regression coefficients with standard errors, z values and p-values

summary(model)$coefficients

Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.086905e+01 4.922555e-01 -22.080088 4.911280e-108
studentYes -6.467758e-01 2.362525e-01 -2.737646 6.188063e-03
balance 5.736505e-03 2.318945e-04 24.737563 4.219578e-135
income 3.033450e-06 8.202615e-06 0.369815 7.115203e-01

This detailed matrix confirms that variables like `balance` and `studentYes` are statistically significant predictors of default status, given their extremely low p-values, whereas `income` does not appear to be significant in this particular GLM (P-value = 0.7115).

Advanced Indexing: Targeting Specific Statistical Metrics

Since `summary(model)$coefficients` returns a highly structured matrix object, we can use two-dimensional indexing (row, column) to isolate any specific metric. This is particularly valuable when performing automated reporting, generating dynamic tables, or applying conditional logic based on statistical thresholds.

For example, if we need to access the precise P-Value for the `balance` variable alone, we specify the row name (`'balance'`) and the column name (`'Pr(>|z|)'`).

#view p-value for balance variable
summary(model)$coefficients

4.219578e-135

To extract all p-values simultaneously for all variables in the model, we use matrix slicing: we omit the row index (leaving a blank space before the comma) while specifying the column name `'Pr(>|z|)'`. This technique returns a vector containing all the desired statistics in the order they appear in the model.

#view p-value for all variables
summary(model)$coefficients

(Intercept) studentYes balance income
4.911280e-108 6.188063e-03 4.219578e-135 7.115203e-01

This indexing approach can be adapted to access any column in the matrix, such as the standard errors (`'Std. Error'`) or the Z-values (`'z value'`), providing powerful flexibility for programmatic

data analysis and reporting.

Conclusion: Choosing the Right Extraction Method

In conclusion, working with R's `glm()` function provides multiple pathways to retrieve model parameters. The selection of the appropriate method depends entirely on the required output format and the level of statistical detail necessary for the task at hand. For quick parameter estimates, direct object indexing is the most efficient. However, for formal inference, hypothesis testing, and rigorous reporting, the detailed matrix provided by the `summary()` function is indispensable.

Use `model$coefficients` for raw point estimates of the coefficients only.

Use `summary(model)$coefficients` for the comprehensive matrix including Estimates, Standard Error, Z-values, and P-values.

Use bracket indexing on the summary matrix to isolate specific statistical results for automated workflows, ensuring precision in data extraction.

Mastering these extraction techniques ensures that data scientists can reliably access, manipulate, and report the core results of their generalized linear models, transitioning smoothly from model fitting to rigorous interpretation and publication.