Perform Linear Regression with Categorical Variables in R

Authored by

Mohammed looti

October 28, 2025

Linear regression is a fundamental statistical method used to model the relationship between a **dependent variable** (often called the response variable) and one or more **independent variables** (also known as predictor variables). This powerful technique allows researchers and analysts to quantify how changes in predictors are associated with shifts in the response, enabling both prediction and inference across various fields.

While many classic applications of this model focus solely on continuous variables (numerical quantities measured on a scale), real-world datasets frequently necessitate the inclusion of categorical variables. These variables--which represent groups, types, or qualities--are often critical predictors. Ignoring them, or incorporating them incorrectly, can lead to severely biased or inaccurate regression models.

This comprehensive tutorial provides an expert, step-by-step methodology for executing linear regression in the statistical environment R when your model specification demands the inclusion of categorical predictor variables. We will navigate the essential stages, including data structure validation, appropriate model fitting using R's specialized functions, and the crucial process of interpreting the resulting coefficients.

Understanding Categorical Variables in Regression

Before launching into the coding, it is essential to establish how categorical variables are mathematically integrated into a linear regression framework. Unlike continuous variables, which have inherent numerical meaning (e.g., height or temperature), categorical variables classify observations into distinct, non-ordered groups (e.g., gender, geographical region, or program type). A regression equation cannot simply treat these labels as numerical inputs.

To overcome this limitation, categorical variables must be transformed using a technique called **dummy coding**. This involves converting the single categorical variable into a series of binary variables, known as dummy variables or indicator variables, where a value of 1 indicates presence in a category and 0 indicates absence. Crucially, if a categorical variable contains k distinct categories (or levels), we only require $k-1$ dummy variables in the model.

The kth category that is omitted from the set of dummy variables is designated as the reference category. The effect of this reference group is implicitly contained within the model's intercept (β?). The coefficients for the remaining $k-1$ dummy variables then represent the difference in the expected response value compared to that reference category, holding all other predictors constant. Fortunately, R's core modeling functions, particularly the `lm()` function, automate this complex process seamlessly when the variable is correctly specified as a `factor`.

Practical Example: Modeling Basketball Performance

To illustrate the practical application of including categorical predictors, we will analyze a hypothetical dataset concerning basketball players. Our objective is to build a linear regression model predicting the total points scored by a player, based on their training commitment and the specific program they utilized. This scenario perfectly combines a continuous predictor (hours) and a categorical predictor (program).

Our dataset, which we will create in R, includes 12 observations (players) and three essential variables:

points: The total points scored, serving as our **response variable**.
hours: The number of hours the player spent practicing, a **continuous predictor**.
program: The specific training program used (1, 2, or 3), which is our **categorical predictor variable**.

The initial step is to construct this data frame within R and confirm its structure. Although the `program` variable is initially numeric (1, 2, 3), we must explicitly tell R that these numbers represent distinct, non-ordered categories--a critical step before fitting the model.

#create data frame
df <- data.frame(points=c(7, 7, 9, 10, 13, 14, 12, 10, 16, 19, 22, 18),
hours=c(1, 2, 2, 3, 2, 6, 4, 3, 4, 5, 8, 6),
program=c(1, 1, 1, 1, 2, 2, 2, 2, 3, 3, 3, 3))

#view data frame
df

points hours program
1 7 1 1
2 7 2 1
3 9 2 1
4 10 3 1
5 13 2 2
6 14 6 2
7 12 4 2
8 10 3 2
9 16 4 3
10 19 5 3
11 22 8 3
12 18 6 3

Preparing Data and Fitting the Model in R

The goal is to estimate a linear regression model where points scored are a function of both practice hours and the training program. The conceptual model is: **points = β? + β?(hours) + β?(program)**. Since the **program** variable must be treated as a set of indicator variables rather than a single numerical column, proper data preparation is mandatory.

We use the `as.factor()` function in R to convert the `program` column from an integer type to a factor. This transformation signals to the `lm()` function to automatically apply the necessary dummy coding. Once the data type is corrected, we fit the model using the standard `lm()` syntax and request the summary output.

By executing the code below, we instruct R to treat Program 1 as the implicit reference category and calculate the effects of Program 2 and Program 3 relative to Program 1, while simultaneously estimating the effect of continuous practice hours.

#convert 'program' to factor
df$program <- as.factor(df$program)

#fit linear regression model
fit <- lm(points ~ hours + program, data = df)

#view model summary
summary(fit)

Call:
lm(formula = points ~ hours + program, data = df)

Residuals:
Min 1Q Median 3Q Max
-1.5192 -1.0064 -0.3590 0.8269 2.4551

Coefficients:
Estimate Std. Error t value Pr(>|t|)
(Intercept) 6.3013 0.9462 6.660 0.000159 ***
hours 0.9744 0.3176 3.068 0.015401 *
program2 2.2949 1.1369 2.019 0.078234 .
program3 6.8462 1.5499 4.417 0.002235 **

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

Residual standard error: 1.403 on 8 degrees of freedom

Multiple R-squared: 0.9392, Adjusted R-squared: 0.9164
F-statistic: 41.21 on 3 and 8 DF, p-value: 3.276e-05

Detailed Interpretation of Regression Output

The output generated by the `summary()` function contains all the information needed to evaluate the model's performance and interpret the predictors. We will focus primarily on the "Coefficients" table, which allows us to write out the specific estimated equation: **points = 6.3013 + 0.9744(hours) + 2.2949(program 2) + 6.8462(program 3)**.

The interpretation of these coefficients must carefully consider the concept of the reference category (Program 1). The coefficients for `program2` and `program3` do not represent the absolute expected points for those programs, but rather the estimated difference in points compared to Program 1, holding hours constant. We also rely on the p-value column to assess the statistical significance of each predictor.

Detailed breakdown of the estimated parameters:

Intercept (6.3013): This is the baseline prediction. It estimates the expected points scored for a player who has practiced 0 hours and is enrolled in Program 1 (the reference group). With a very low p-value (0.000159), this intercept is highly statistically significant.
hours (Coefficient: 0.9744): This indicates that for every one-hour increase in practice time, the player's predicted points scored increase by **0.9744**, regardless of the program they are in. The p-value of 0.015401 confirms that practice hours is a statistically significant predictor (at $\alpha = 0.05$).
program2 (Coefficient: 2.2949): Players in Program 2 are expected to score **2.2949** more points than those in Program 1, assuming identical practice hours. However, the p-value (0.078234) is slightly above the conventional 0.05 threshold, suggesting this difference is not conventionally statistically significant.
program3 (Coefficient: 6.8462): Players utilizing Program 3 are predicted to score a substantial **6.8462** more points, on average, than those in Program 1, again holding hours constant. The low p-value (0.002235) confirms that Program 3 yields a highly statistically significant difference in performance compared to the reference group.

In addition to the individual coefficients, the model summary provides crucial overall metrics. The Multiple R-squared value of 0.9392 signifies that nearly 94% of the variability in points scored is explained by the combination of hours and training program. Furthermore, the highly significant F-statistic and its associated p-value (3.276e-05) confirm that our model, taken as a whole, provides a much better fit than a model with no predictors.

Making Predictions with the Fitted Model

A core utility of a properly fitted linear regression model is its ability to forecast outcomes for new data points. We can easily use our derived equation to predict the expected points for a hypothetical player, given their specific practice hours and assigned program. R streamlines this prediction process using the versatile `predict()` function, which requires a new data frame containing the predictor values.

Let us define a new player who has practiced for 5 hours and is enrolled in the highly effective Program 3. We create a new data frame structure, ensuring that the `program` variable is explicitly treated as a factor (matching the structure of our model data), and then pass this new data to the `predict()` function.

#define new player
new <- data.frame(hours=c(5), program=as.factor(c(3)))

#use the fitted model to predict the points for the new player
predict(fit, newdata=new)

1
18.01923

The model predicts that this new player will score approximately **18.02** points. We can manually verify this prediction by plugging the values (Hours=5, Program 3=1, Program 2=0) back into our equation, confirming our understanding of how the coefficients for the categorical variable are applied:

points = 6.3013 + 0.9744(hours) + 2.2949(program 2) + 6.8462(program 3)
For Program 3:

points = 6.3013 + 0.9744(5) + 2.2949(0) + 6.8462(1)
points = 6.3013 + 4.872 + 0 + 6.8462
points = 18.0195

This manual check confirms the accuracy of R's calculation and solidifies the interpretation of the model's structure when handling categorical predictors.

Conclusion and Further Steps

This guide has successfully demonstrated the methodology for performing linear regression in R when the predictor set includes categorical variables. The key to accurate modeling lies in correctly

specifying these variables as factors, allowing the `lm()` function to automatically generate and manage the necessary dummy variables.

Mastery of interpreting the resulting coefficients is crucial, particularly understanding how the choice of reference category affects the magnitude and meaning of the estimates. Moving forward in your statistical journey, we highly recommend exploring advanced topics such as checking regression assumptions, analyzing interaction terms (where the effect of one predictor depends on the level of another), and applying various diagnostics to build even more robust predictive models.

Additional Resources

The following tutorials offer further guidance on performing other common statistical tasks in R: