Learning to Extract Fitted Values from Linear Regression Models Using R

Authored by

Mohammed looti

November 13, 2025

The Foundational Concepts of Linear Regression and Prediction

Linear regression stands as a cornerstone in statistical methodology, utilized extensively across disciplines ranging from economics to engineering to model and quantify relationships within data. This powerful technique seeks to summarize the association between a single outcome variable (the response) and one or more **predictor variables**. The fundamental goal is to identify the line or hyperplane that best approximates the observed data points, minimizing the overall prediction error. By defining this line of best fit, we gain crucial insight into how changes in the input variables systematically influence the expected outcome, forming the basis for statistical inference and prediction.

Central to understanding any fitted regression model is the concept of **fitted values**. These values, conventionally denoted as $hat{Y}$ (Y-hat), constitute the specific output predictions generated by the established regression equation for every observation present in the original dataset. Each fitted value is calculated by substituting the corresponding actual predictor values back into the derived model formula. Analyzing these fitted values is indispensable for performing rigorous model diagnostics, verifying the accuracy of the model's predictions on the training data, and pinpointing observations that might be unduly influencing the model's estimates, such as potential **outliers**.

In the realm of statistical computing, particularly within the R programming language, the workflow for statistical modeling is highly structured and efficient. The initial step requires successfully fitting the model to the input data, which produces a complex object containing all necessary computational results. This model object serves as a container for various components, including the estimated regression coefficients, the residuals, and, most importantly for this discussion, the precise set of **fitted values**. Subsequent analysis hinges entirely on correctly accessing and extracting this specific attribute from the resulting model object.

Utilizing the lm() Function for Model Fitting in R

The backbone of linear modeling in R is the built-in `lm()` function. This function is universally applied whether constructing a simple linear model (involving just one predictor) or a complex multiple linear regression model (incorporating several predictors). Executing `lm()` results in the creation of a comprehensive model object--typically of class "lm"--that encapsulates the entirety of the regression analysis results. This object is the definitive source for all derived statistical measures, including coefficients, error estimates, and the required **fitted values**.

Once the model fitting process is successfully completed by the `lm()` function, the resulting object inherently possesses an attribute specifically dedicated to storing the model's predictions. This attribute, named `fitted.values`, holds the vector of predicted response values ($hat{Y}$). The

standard practice in **R** for isolating and retrieving this specific component is through the use of the dollar operator (`$`) appended to the model object name, followed by the attribute identifier. This streamlined approach allows analysts to quickly separate the predictions from the rest of the detailed statistical output for targeted examination.

To solidify this concept, the subsequent sections will detail a complete, practical workflow. We will begin by structuring a dataset, proceed to fit a **multiple linear regression model** using the powerful `lm()` function, and culminate in the precise extraction of the `fitted.values` attribute. This hands-on demonstration is designed to clarify the transition from theoretical statistical modeling to executable and verifiable data analysis steps within the **R programming environment**.

Practical Setup: Preparing the Basketball Performance Dataset

To effectively demonstrate the mechanics of extracting predicted values, we will employ a realistic, albeit hypothetical, dataset focused on basketball player statistics. This dataset, formatted as an **R data frame**, includes several metrics critical to player performance. Our analytical goal is straightforward: we aim to predict a player's total **points** scored based on two input variables: the total **minutes** they played and the number of **fouls** they committed during the game.

The initial step requires defining and structuring the data frame, ensuring that all variables are correctly named and aligned. This structured preparation is paramount for **R**'s modeling functions, as it explicitly defines the roles of the response variable and the **predictor variables**. The sample dataset below comprises 10 distinct observations, offering a compact yet entirely representative sample suitable for illustrating the principles of multiple regression analysis and prediction extraction.

```
#create data frame
df <- data.frame(minutes=c(5, 10, 13, 14, 20, 22, 26, 34, 38, 40),
fouls=c(5, 5, 3, 4, 2, 1, 3, 2, 1, 1),
points=c(6, 8, 8, 7, 14, 10, 22, 24, 28, 30))

#view data frame
df
```

	minutes	fouls	points
1	5	5	6
2	10	5	8
3	13	3	8
4	14	4	7
5	20	2	14

6 22 1 10
7 26 3 22
8 34 2 24
9 38 1 28
10 40 1 30

This prepared dataset establishes the prerequisite structure for fitting our **multiple linear regression model**. The model is designed to capture the combined linear influence of **minutes** played and **fouls** committed on the response variable, **points**. Theoretically, this relationship is formalized by the equation:

points = β0 + β1(minutes) + β2(fouls) + ε

where $beta_0$ is the intercept, $beta_1$ and $beta_2$ are the coefficients representing the impact of each predictor, and $epsilon$ represents the error term. The **fitted values** we seek to extract will be the outcome of this equation once the optimal $beta$ coefficients have been estimated by the `lm()` function.

Implementing and Evaluating the Regression Model

With the data frame correctly defined, the subsequent crucial step involves fitting the model using the `lm()` function. We articulate the model relationship using R's formula notation (`points ~ minutes + fouls`) and specify the source data (`data=df`). The execution of this function efficiently computes the optimal coefficient estimates ($hat{beta}$ values) that define the line of best fit. The output, stored in the `fit` object, is the foundation for all subsequent analysis and prediction tasks.

A standard practice immediately following model fitting is to invoke the `summary()` function on the `fit` object. This action generates a comprehensive statistical report essential for evaluating the model's overall quality and the reliability of the estimated coefficients. The summary details include crucial diagnostic information such as the distribution of **residuals**, the coefficient estimates, their standard errors, t-statistics, and the associated **p-values**. This evaluation ensures that the chosen **predictor variables** are statistically relevant in explaining the variability observed in the response variable.

#fit multiple linear regression model
fit <- lm(points ~ minutes + fouls, data=df)

#view summary of model
summary(fit)

Call:

lm(formula = points ~ minutes + fouls, data = df)

Residuals:
Min 1Q Median 3Q Max
-3.5241 -1.4782 0.5918 1.6073 2.0889

Coefficients:
Estimate Std. Error t value Pr(>|t|)
(Intercept) -11.8949 4.5375 -2.621 0.0343 *
minutes 0.9774 0.1086 9.000 4.26e-05 ***
fouls 2.1838 0.8398 2.600 0.0354 *

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

Residual standard error: 2.148 on 7 degrees of freedom
Multiple R-squared: 0.959, Adjusted R-squared: 0.9473
F-statistic: 81.93 on 2 and 7 DF, p-value: 1.392e-05

Upon reviewing the output summary, particular attention must be paid to the **Pr(>|t|)** column, which presents the associated **p-values** for each coefficient. In this example, the p-values for both the **minutes** (4.26e-05) and **fouls** (0.0354) predictors are substantially below the conventional statistical significance threshold of 0.05. This strong finding confirms that both variables are statistically significant **predictor variables** in modeling the total **points** scored, validating the use of the model for generating accurate predictions. The high Adjusted R-squared value (0.9473) further suggests that the model accounts for nearly 95% of the variance in the response variable.

The Extraction Process: Accessing and Storing Fitted Values

With the statistically significant model object `fit` now residing in the **R programming language** environment, the process of extracting the **fitted values** becomes remarkably simple and direct. These fitted values represent the model's optimal prediction for the response variable (points) given the specific inputs (minutes and fouls) for each corresponding row in the original dataset. We retrieve this vector of predictions by using the dollar operator (`$`) combined with the attribute identifier `fitted.values` on the `fit` object.

For practical analysis and robust visualization, the most efficient technique is to integrate these fitted values directly back into the original **data frame**. By adding the predictions as a new column, analysts create an immediate, side-by-side comparison between the model's estimated outcome and the actual observed outcome. This juxtaposition is invaluable for quickly performing initial diagnostic checks and assessing the model's performance on an observation-by-observation

basis. For clarity, we label this new column `fitted`.

#extract fitted values from regression model into new column in original data frame
df$fitted <- fit$fitted.values

#view updated data frame
df

minutes fouls points fitted
1 5 5 6 3.911127
2 10 5 8 8.798214
3 13 3 8 7.362896
4 14 4 7 10.524098
5 20 2 14 12.021032
6 22 1 10 11.792082
7 26 3 22 20.069321
8 34 2 24 25.704875
9 38 1 28 27.430760
10 40 1 30 29.385594

The resultant `fitted` column contains the precise numerical **fitted values** calculated by the optimized regression equation. These figures represent the expected number of **points** for each player, predicated solely on their corresponding **minutes** and **fouls** data, according to the parameters established by the `lm()` function. This output is the critical link between the theoretical regression model and its tangible predictive performance on the training data.

Model Diagnostics: Comparing Predictions and Residuals

The true utility of extracting **fitted values** is realized when they are benchmarked against the **actual response** values (the `points` column). This comparison directly illuminates the model's predictive accuracy for each observation. The crucial metric derived from this comparison is the **residual**, which is mathematically defined as the difference between the observed value (Y) and the fitted value ($\hat{Y}$). Small residuals signify a close agreement between prediction and reality, indicating a strong fit for that data point. Conversely, large residuals are vital indicators of poor fit, potentially signaling data issues, the presence of influential **outliers**, or a fundamental deficiency in the model's specified structure.

We can analyze specific rows of our enhanced **data frame** to quantify the magnitude and direction of the prediction errors:

The first player recorded **6** points. The model predicted **3.91** points. The **residual** is $6 - 3.91 =

2.09$. This substantial positive residual indicates the model significantly underestimated this player's performance.
The second player scored **8** points. The model predicted **8.80** points. The **residual** is $8 - 8.80 = -0.80$. Here, the model slightly overestimated the player's score.
The third player scored **8** points. The model predicted **7.36** points. The **residual** is $8 - 7.36 = 0.64$. This is a relatively small positive residual, suggesting a highly accurate prediction compared to the previous two examples.

Systematic evaluation of the `fitted` column against the `points` column for all observations provides analysts with deep insight into the model's overall predictive performance and bias. If the model exhibits high fidelity to the underlying data, the **fitted values** will generally cluster tightly around the actual observed values. This vital comparison forms the cornerstone of model diagnostics in the **R programming environment**, enabling analysts to confidently determine how well the chosen **predictor variables** (minutes and fouls) jointly explain the variability in the response.

Conclusion and Next Steps for Regression Analysis

The ability to correctly extract **fitted values** is a foundational, non-negotiable skill for effective statistical computing in R. This process enables researchers to move past the abstract interpretation of estimated coefficients and interact directly with the quantifiable predictions generated by the model. By leveraging the `fitted.values` attribute in conjunction with the model object created by the `lm()` function, analysts gain immediate access to the predicted response for every data point. This capability is essential for performing critical diagnostic checks, such as detailed residual analysis and identifying specific instances of poor model fit.

Mastering the extraction and interpretation of these predicted values is critical for bridging the gap between theoretical statistical modeling and practical, real-world data application. The resulting vector of fitted values provides a quantifiable, objective measure of the model's performance on the training data, ensuring that the final statistical model is both robustly sound and practically useful for future forecasting and inference tasks.

Additional Resources

For analysts seeking to deepen their understanding of statistical modeling and diagnostics in **R**, the following resources provide further guidance on common analytical tasks and techniques, enhancing the ability to build, evaluate, and refine predictive models:

<!--

Featured Posts

-->