How to translate the results from lm() to an equation?

We can use `lm()` to predict a value, but we still need the equation of the result formula in some cases. For example, add the equation to plots.

r | regression | lm

edited Jul 9 '13 at 3:13
gung ♦
103k 34 246 510

asked Jul 8 '13 at 3:20
user27736
116 1 1 4

2 Can you please rephrase your question or add some details? I'm quite familiar with R, `lm` and linear models more generally, but it's not at all clear what, exactly, you want. Can you give an example or something to clarify? Is this for some subject? – Glen_b ♦ Jul 8 '13 at 3:27

1 I guess you want the coefficients of the linear regression formula. Try calling `coef()` on the fitted `lm` object, as in: `mod <- lm(y ~ x); coef(mod)` – Jake Westfall Jul 8 '13 at 3:36

If you type `lm(y~x)$call` it tells you the formula is `y ~ x` . If you mean something different from that, you need to be more specific. – Glen_b ♦ Jul 8 '13 at 6:47

1 Related: How to apply coefficient term for factors and interactive terms in a linear equation? – chl ♦ Jul 8 '13 at 11:26

Worth reading stackoverflow.com/questions/7549694/… – mnel Jul 9 '13 at 4:07

3 Answers

Consider this example:

```
set.seed(5)            # this line will allow you to run these commands on your
                       # own computer & get *exactly* the same output
x = rnorm(50)
y = rnorm(50)

fit = lm(y~x)

summary(fit)

Call:
lm(formula = y ~ x)

Residuals:
     Min       1Q   Median       3Q      Max
-2.04003 -0.43414 -0.04609  0.50807  2.48728

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.00761    0.11554  -0.066    0.948
x            0.09156    0.10901   0.840    0.405

Residual standard error: 0.8155 on 48 degrees of freedom
Multiple R-squared: 0.01449,    Adjusted R-squared: -0.006046
F-statistic: 0.7055 on 1 and 48 DF,  p-value: 0.4051
```

The question, I'm guessing, is how to figure out the regression equation from R's summary output. Algebraically, the equation for a simple regression model is:

$$\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_i + \hat{\varepsilon}_i$$
$$\text{where } \varepsilon \sim \mathcal{N}(0,\ \hat{\sigma}^2)$$

We just need to map the `summary.lm()` output to these terms. To wit:

- $\hat{\beta}_0$ is the `Estimate` value in the `(Intercept)` row (specifically, `-0.00761`)
- $\hat{\beta}_1$ is the `Estimate` value in the `x` row (specifically, `0.09156`)
- $\hat{\sigma}$ is the `Residual standard error` (specifically, `0.8155`)

Plugging these in above yields:

$$\hat{y}_i = -0.00761 + 0.09156 x_i + \hat{\varepsilon}_i$$
$$\text{where } \varepsilon \sim \mathcal{N}(0,\ 0.8155^2)$$

For a more thorough overview, you may want to read this thread: Interpretation of R's lm() output.

Given the OP's mention of a wish to put equations on graphs, I've been pondering whether they actually want a function to take the output of `lm` and produce a character expression like "$\hat{y} = -0.00761 + 0.09156x$" suitable for such a plotting task (hence my repeated call to clarify what they wanted - which hasn't been done, unfortunately). – Glen_b ♦ Jul 9 '13 at 3:14

If what you want is to predict scores using your resulting regression equation, you can construct the equation by hand by typing `summary(fit)` (if your regression analysis is stored in a variable called `fit`, for example), and looking at the estimates for each coefficient included in your model.

For example, if you have a simple regression of the type $y = \beta_0 + \beta_1 x + \epsilon$, and you get an estimate of the intercept (β_0) of +0.5 and an estimate of the effect of x on y (β_1) of +1.6, you would predict an individual's y score from their x score by computing: $\hat{y} = 0.5 + 1.6x$.

However, this is the hard route. R has a built-in function, `predict()`, which you can use to automatically compute predicted values given a model for any dataset. For example: `predict(fit, newdata=data)`, if the x scores you want to use to predict y scores are stored in the variable `data`. (Note that in order to see the predicted scores for the sample on which your regression was performed, you can simply type `fit$fitted` or `fitted(fit)`; these will give you the predicted, a.k.a. fitted, values.)

edited Jul 8 '13 at 4:12

answered Jul 8 '13 at 3:39

Patrick Coulombe
2,240 4 16 25

1 +1, I also changed my `model` to `fit` to parallel your answer. – gung ♦ Jul 8 '13 at 3:56

If you want to show the equation, like to cut/paste into a doc, but don't want to fuss with putting the entire equation together:

```
R> library(MASS)
R> crime.lm <- lm(y~., UScrime)
R> cc <- crime.lm$coefficients
R> (eqn <- paste("Y =", paste(round(cc[1],2), paste(round(cc[-1],2), names(cc[-1]), sep="
* ", collapse=" + "), sep=" + "), "+ e"))
[1] "Y = -5984.29 + 8.78 * M + -3.8 * So + 18.83 * Ed + 19.28 * Po1 + -10.94 * Po2 +
-0.66 * LF + 1.74 * M.F + -0.73 * Pop + 0.42 * NW + -5.83 * U1 + 16.78 * U2 + 0.96 * GDP
+ 7.07 * Ineq + -4855.27 * Prob + -3.48 * Time + e"
```

answered Sep 20 at 17:35

keithpjolley
101 2