Interpretation of Multiple Regression Coefficients

Brooke Coneeny

The data and code for this lecture can be found at my github: @brooke-coneeny

Multiple Regression Coefficients

A multiple regression can be understood as the following statistical model: $y_i = \beta_0 + \beta_1 x_{i,1} + ... + \beta_{p-1} x_{i,p-1} + \epsilon_i$ for $i = 1, ..., n$. Instead of fitting a line to a two-dimensional plot of points as done in simple regression, we are now fitting a plane to a multi-dimensional plot of points. In this case the residuals now represent the vertical distance between each point and the best-fit plane, and it is our goal to minimize these distances.

The $\beta_0, \beta_1, ..., \beta_{p-1}$ are unknown parameters and the ϵ_i are independent random variables with mean zero and variance σ^2. The β_i can be interpreted as: β_k is the change in the expected value of y if x_k is increased be one unit and the other $x's$ are held fixed.

To find the value of these regression coefficients we use simple regression output. Let us go through this process using example coin data, where the sample contains 25 students who each drew a small handful of coins and recorded x_1: the number of pennies drawn , x_2: the total number of coins draws, and Y: the total value in cents for their handful.

As we can in the output pdf, both of the explanatory variables have linear relationships with the total value of coins in cents

Step 1:

The first step towards calculating the multiple regression coefficients is to fit a simple regression of Y on X_2, value in cents on number of coins, and to save the residuals

```
#Fitting a simple regression of value in cents on number of coins
model1 <- lm(value.in.cents ~ X..of.coins, data = coin_data)
#Saving the residuals of the above model
model1_resids <- resid(model1)
#Adding these residuals to the data set
coin_data <- coin_data %>% cbind(model1_resids)
```

Step 2:

The next step is to fit a regression of X_1 on X_2, number of pennies on number of coins, and save the residuals

```
#Fitting a simple regression of number of pennies on number of coins
model2 <- lm(X..of.pennies ~ X..of.coins, data = coin_data)
#Saving the residuals of the above model
model2_resids <- resid(model2)
#Adding these residuals to the data set
coin_data <- coin_data %>% cbind(model2_resids)
```

Step 3:

The final step is to fit a simple regression of the first set of residuals from model 1 on the second set of residuals from model 2. The fitted slope of this model is the coefficient for X_1, number of pennies, in the multiple regression. This can be see by comparing the summary output of model 3 with model 4.

```
#Fitting a simple regression of the first set of residuals on the second set of residuals
model3 <- lm(model1_resids ~ model2_resids, data = coin_data)

#This is the multiple regression
model4 <- lm(value.in.cents ~ X..of.pennies + X..of.coins, data = coin_data)

#This is the simple regression of x1 on y
model5 <- lm(value.in.cents ~ X..of.pennies, data = coin_data)
```

Interpretation:

Interpreting a multiple regression coefficient is slightly more complex than in simple regression, for multiple regression estimates the effect of one variable x_1 on Y while holding constant other x_i which may be responsible for the observed association between x_1 and Y.

β_0 is still interpreted as the intercept, which is the expected value of Y when all x_i's are equal to 0

β_i in now interpreted as the expected change in Y given a 1-unit increase in x_i while holding all other variables constant.

Continuing to use the coin example, we can see the values of β_1 differ between the simple and multiple regression:

β_1 can be interpreted, in the simple regression, as: A one unit increase in the number of pennies is associated with a 3.16 increase in the total value in cents

β_1 can be interpreted, in the multiple regression, as: A one unit increase in the number of pennies is associated with a -7.03 decrease in the total value in cents, holding the number of coins constant.

As we can see, the values of β_1 differs in the two regressions. This is because some of the predictive power coming from knowing how many pennies is in the handful in the multiple regression is due to the fact that we know how many coins there are.

Simpson's Paradox

Simpson's Paradox is a statistical phenomenon where the relationship between a response variable (Y) and an explanatory variable (X) disappears or reverses when introduced to a third variable (V). This third variable typically divides the sample into subgroups.

The data set we are going to use is from an educational study. It includes information about each state along with the corresponding average SAT score. There is also data on how much money (expenditures) the state spent on SAT resources along with what percent of students in the state take the exam.

At first glance, there seems to be a negative correlation between the average SAT score in a state and the amount of money spent per pupil on education in the state. This relationship is seen in the output pdf and shows us that, supposedly, the states which spend the most amount of money per pupil on education, on average, score worse on their SAT. This should not make sense, we would hope that states which are allocating more money towards education are seeing the payoff.

However, once we introduce the third variable, percent of students taking the SAT, the relationship reverses. The below code shows that I split up the data into two divisions: states where more than 50% of students

take the SAT, and states where less than 50% of students take the SAT. I created the same plot as above for the upper division and the lower. From these plots we can see that the introduction of this third variable flips the relationship for the upper division. This is therefore an example of Simpson's Paradox.

```
grouped_SAT_data <- SAT_data %>%
  mutate(division = case_when(
    Pct_Taking_SAT >= 50 ~ "upper",
    Pct_Taking_SAT < 50 ~ "lower"
  ))
```

Regression Estimates & Standard Errors

So far we have learned the vector form of the last squares estimate $\hat{\beta} = (X^T X)^{-1} X^T Y$, now let us go over how to find it via code. The following R code explains how to compute the fitted regression estimate using the software:

```
#Using the coin data once again
#X1 column vector is the column containing number of pennies in each handful
#X2 column vector is the column containing number of coins in each handful
#Y column vector is the column containing the value of each handful
X1 <- coin_data$X..of.pennies
X2 <- coin_data$X..of.coins
Y <- coin_data$value.in.cents

#Creating (nxp) X matrix
X <- cbind(1,X1,X2)

#Finding the inverse of X-transpose times X
Xinv <- solve(t(X) %*% X)

#Multiplying the above by X-transpose and Y to get beta-hat
bhat <- Xinv %*% t(X) %*% Y
```

Now that we know how to compute $\hat{\beta}$, let us look at the code for finding the root mean square error (s), which is the estimate for sigma. The RMSE is equal to $\sqrt{\frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)}{n-p}}$. This means that the degree of freedom is equal to n-p, which in this case is 25-3 = 22.

```
#Finding the predicted y values using the predicted beta from above
yhat <- X %*% bhat

#Finding the residuals
resids <- Y - yhat

#Finding the number of observations
n <- nrow(coin_data)

#Finding the estimate of sigma
rmse <- sqrt(sum(resids^2)/(n-3))
```

The last computation we are going to cover is finding the standard error estimates. The standard error ($SE(\hat{\beta}_j)$ for $j = 1, ...(p-1)$) is equal to $s\sqrt{(X^T X)^{-1}_{jj}}$

```r
#Finding the standard error
SE <- rmse * sqrt(diag(Xinv))
```

Comparing t-test and f-test

The hypotheses being tested for the t-tests for individual β_j's are different than the whole model F test. In the case of multiple regression, the F-test tends to be most useful because it tests whether all of the betas are significant. This is done by using the null hypothesis: $H_0 : \beta_1 = \beta_2 = ... = \beta_{(p-1)} = 0$ against the alternative hypothesis: $H_1 : \beta_j \neq 0$ for at least one value of j. In order to reject this null hypothesis the F-statistic needs to be high. The t-test is slightly different for it only tests individual betas. The null hypothesis for a t test is: $H_0 : \beta_j = 0$ with the alternative being $H_a : \beta_j \neq 0$. The main difference between the two is that the f-test ensures the significant of the betas as a whole whereas the t-test only ensures one beta at a time.

To conceptualize this, think about the example of a professor giving a student 5 quizzes and a final exam. The professor runs a multiple regression of the final exam score on the quizzes and finds that the F-test is very significant, but none of the individual t tests are significant. This is because the two tests are asking different questions. The F-tests examines whether all the quiz scores, as a whole, are significant when predicting the final exam score. The t-test, for say quiz 5, examines whether or not the 5th quiz score is necessary to predict the final exam score when we already have the 4 other quizzes. It is possible that the professor got a significant F statistic and not t-statistics because the final exam score can not be predicted by only a subset of the quizzes, it needs all five.