Linear Regression: Part 2

Jeremy Oldfather

November 4, 2016

Multiple Linear Regression - Last Time

When we regress y on more than one independent variable, say $(x_1, x_2, ..., x_k)$ rather than simply (x_1), the regression framework is referred to as **multiple linear regression**.

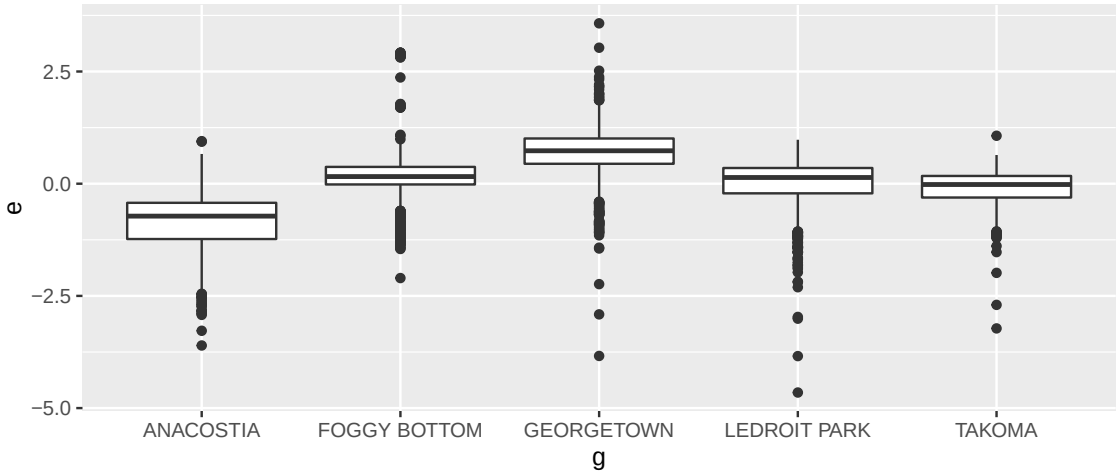
What might motivate us to introduce multiple regressors? One reason might be that we have reason to suspect **omitted variable bias**. In order for our model to be subject to omitted variable bias, we must have both of the following problems:

$E(y|x_1, x_2) \neq E(y|x_1)$: Meaning our expectation about the dependent variable y changes when we include the additional independent variable. If $E(y|x_1, x_2) = E(y|x_1)$, we would say that y is conditionally independent of x_2—meaning either x_2 is completely irrevelant to predicting y, or that any information contained in x_2 is already contained in x_1.

$Cor(x_1, x_2) \neq 0$: Meaning the additional variable x_2 must be correlated with a current dependent variable x_1.

For example, when we plot the distribution of residuals from our previous model by neighborhood, we can see that the variable **neighborhood** meets the first critera for omitted variable bias since for Anacostia we would consistently overstate the impact of a change in living area on the change in sales price.

```
selected.nhoods<-c("FOGGY BOTTOM","ANACOSTIA","LEDROIT PARK","TAKOMA","GEORGETOWN")
data.frame(e=resid(fit),g=data.model$neighborhood) %>%
  filter(g %in% selected.nhoods) %>%
  ggplot(aes(y=e,x=g)) + geom_boxplot()
```



To check the second criteria, that $Cor(living\_area, neighborhood) \neq 0$, we can plug a dummy variable for each neighborhood into R's **cor()** function.

```
cor(data.model$log_living_sqft,data.model$neighborhood=="ANACOSTIA")

## [1] 0.02040842

cor(data.model$log_living_sqft,data.model$neighborhood=="FOGGY BOTTOM")

## [1] -0.1058928

cor(data.model$log_living_sqft,data.model$neighborhood=="GEORGETOWN")

## [1] 0.06697943

cor(data.model$log_living_sqft,data.model$neighborhood=="LEDROIT PARK")

## [1] 0.03272786

cor(data.model$log_living_sqft,data.model$neighborhood=="TAKOMA")

## [1] 0.0146363
```

Anacostia, Ledroit Park, and Takoma are either weakly or not at all correlated with living area, but Foggy Bottom is negatively correlated and Georgetown is weakly positively correlated.

To see the connection between including these neighborhoods in the regression, I will create a new variable that sets all other neighborhoods to "other". I put "aa" at the beginning because R treats the first element of a factor variable as the base value for the regression. So by doing this, we can see how the neighborhoods we analyzed above affect our regression in terms of all other DC neighborhoods.

```
fit2<-data.model %>%
  mutate(nhoods5=ifelse(neighborhood %in% selected.nhoods,
```

```
                        neighborhood,
                        "aa_other")
        ) %>%
  lm(log_sale_price ~ log_living_sqft + factor(nhoods5),data=.)
summary(fit2)

##
## Call:
## lm(formula = log_sale_price ~ log_living_sqft + factor(nhoods5),
##     data = .)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -13.6024  -0.3463   0.1282   0.4260   6.5927
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)                   7.913192   0.033502 236.204  < 2e-16 ***
## log_living_sqft               0.699649   0.004723 148.133  < 2e-16 ***
## factor(nhoods5)ANACOSTIA     -0.832172   0.023094 -36.034  < 2e-16 ***
## factor(nhoods5)FOGGY BOTTOM   0.192594   0.024741   7.784 7.09e-15 ***
## factor(nhoods5)GEORGETOWN     0.715779   0.015717  45.543  < 2e-16 ***
## factor(nhoods5)LEDROIT PARK  -0.007375   0.022117  -0.333 0.738790
## factor(nhoods5)TAKOMA        -0.129247   0.036654  -3.526 0.000422 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7052 on 85290 degrees of freedom
## Multiple R-squared:  0.2355,Adjusted R-squared:  0.2354
## F-statistic:  4378 on 6 and 85290 DF,  p-value: < 2.2e-16
```

Comparing the coefficients for each neighborhood with the distribution of their residuals in the boxplot above, we can see the pattern. The coefficient for a dummy shifts our fitted line up or down by the amount of the coefficient. For example, when all the dummies are turned off (equal zero),

$$\hat{y}_i = 7.913192 + 0.699649 x_i$$

But when Anacostia equals 1,

$$\begin{aligned} \hat{y}_i &= 7.913192 + 0.699649 x_i - 0.832172(1) \\ &= 7.08102 + 0.699649 x_i \end{aligned}$$

Now I will add dummies for the rest of the DC neighborhoods and run the regression again.

```
fit3<-lm(log_sale_price ~ log_living_sqft + factor(neighborhood),data=data.model)
coef(fit3)[1:2]

##     (Intercept) log_living_sqft
##       7.634167        0.716211
```

```
summary(resid(fit3))
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -13.33000  -0.22140   0.07746   0.00000   0.29570   6.22800
```

Multiple Linear Regression - Continuation

Previously, we showed how to fit a line using simple linear regression (one regressor) and then how adding relevant regressors to our model can improve the fit and limit omitted variable bias. We used a box-plot to show that residuals were correlated with a variable that was not yet included in our model. However, we did not discuss how we can prove to ourselves in a rigorous way that the goodness-of-fit in the second model is better than the one in the first.

Goodness-of-Fit (R^2)

Let's start by returning to our definition of the sum of squared residuals. We can re-write it in terms of the residuals in the following way.

$$SSR(\beta_0, ..., \beta_k) = \sum_i (y_i - (\beta_0 + \beta_1 x_{i1} + \cdots + \beta_k x_{ik}))^2 \tag{1}$$

$$= \sum_i (y_i - (\hat{y}_i))^2 \tag{2}$$

$$= \sum_i (\epsilon_i - 0)^2 \tag{3}$$

$$= (n-1)Var(\epsilon) \tag{4}$$

Written this way, we can see that the sum of squared residuals is proportional to the variation of the residuals. In other words, **SSR is the variation in house prices that our model does not explain**.

It turns out that we can also write down a formula for the variation in house prices that our model does explain, the **explained sum of squares (SSE)**, as well as one for the total variation in house prices, or **total sum of squares (SST)**.

$$SSR(\beta_0, ..., \beta_k) = \sum_i (\epsilon_i - 0)^2 = (n-1)Var(\epsilon) \tag{5}$$

$$SSE(\beta_0, ..., \beta_k) = \sum_i (\hat{y}_i - \bar{y})^2 = (n-1)Var(\hat{y}) \tag{6}$$

$$SST = \sum_i (y_i - \bar{y})^2 = (n-1)Var(y) \tag{7}$$

Together, these components of variation have the following relationship:

$$SST = SSE + SSR \tag{8}$$

With this in mind, R^2 can be defined as either the share of total variation explained by the model,

$$R^2 = \frac{SSE}{SST} \tag{9}$$

$$= \frac{Var(\hat{y})}{Var(y)} \tag{10}$$

Or as 1 minus the share of total variation unexplain by the model,

$$R^2 = 1 - \frac{SSR}{SST} \tag{11}$$

$$= 1 - \frac{Var(\epsilon)}{Var(y)} \tag{12}$$

Let's calculate the R-squared for a model and compare it to the regression output of R.

```
# fit the simple model
fit<-lm(log_sale_price ~ log_living_sqft,data=data.model)

# calculate R-squared
1-var(resid(fit))/var(data.model$log_sale_price)

## [1] 0.2039985

# compare to the regression output
summary(fit)

##
## Call:
## lm(formula = log_sale_price ~ log_living_sqft, data = data.model)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -13.6189  -0.3601   0.1332   0.4313   6.5902
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)     7.875186   0.033900   232.3   <2e-16 ***
## log_living_sqft 0.706351   0.004778   147.8   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7196 on 85295 degrees of freedom
## Multiple R-squared:  0.204,Adjusted R-squared:  0.204
## F-statistic: 2.186e+04 on 1 and 85295 DF,  p-value: < 2.2e-16
```

The regression output refers to R^2 as "multiple R-squared", which is simply an allusion to R^2 under the multiple linear regression framework—it is still calculated as described above.

Adjusted R-squared, however, is different from R^2. R^2 does not account for the number of regressors being added to the model. So for small datasets, R^2 will always increase slightly regardless of the explanatory power of the variable being added to the model. Adjusted R-squared takes the number of added regressors into account and makes the necessary adjustment.

Model Selection

Many methods are available for comparing and selecting the best model (from a statistical perspective).

For the following examples, let's consider two models. The first is called our **restricted model**. It is our best model so far that will act as our base model. The second is called our **unrestricted model**. This is our new model that we would like to test against the first. Let's estimate these models now from the house price data:

```
fit.res<-lm(log_sale_price ~ log_living_sqft,data=data.model)
fit.unres<-lm(log_sale_price ~ log_living_sqft + factor(neighborhood),data=data.model)
```

Testing Adjusted R-squared ($\bar{R}^2$)

Comparing the Adjusted R-squared of two model specifications is a popular method of choosing one model over another.

If $\bar{R}^2_{unres} > \bar{R}^2_{res}$, then we reject the hypothesis that the unrestricted model containing no explanatory power over the restricted one. Then the unstricted model becomes our new basis for comparison.

Let's try this test on our house price models.

```
summary(fit.res)$adj.r.squared

## [1] 0.2039892

summary(fit.unres)$adj.r.squared

## [1] 0.4656749
```

In this example, we would reject the hypothesis that the model including the neighborhood dummies contains no explanatory power beyond that of the restricted model without the dummies.

F-test

In general, an F-test is used to test to the equivalence of variances for two normal distributions. Since well-specified linear regression models will have residuals that are normally distributed (we have not dicussed this), the F-test is commonly used to test the equivalence of the unexplained variation between two models.

For this test, we calculate an F-statistic from the SSR of both models and compared it to the critical value from the F-distribution. For the number of regressors in each respective model (p), and the number of observations (n), the F-statistic is defined as:

$$F = \frac{SSR_{res} - SSR_{unres}}{p_{unres} - p_{res}} \Big/ \frac{SSR_{unres}}{n - p_{unres}} \tag{13}$$

This F-statistic has the distribution $F(p_{unres} - p_{res}, n - p_{unres})$. The critical value is then chosen from this distribution based on the desired significance level (usually .05).

Let's try this test on our house price models.

```
anova(fit.res,fit.unres)
```

```
## Analysis of Variance Table
##
## Model 1: log_sale_price ~ log_living_sqft
## Model 2: log_sale_price ~ log_living_sqft + factor(neighborhood)
##   Res.Df   RSS Df Sum of Sq      F    Pr(>F)
## 1  85295 44166
## 2  85241 29628 54     14538 774.58 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Again, we would reject the hypothesis that the model including the neighborhood dummies contains no explanatory power beyond that of the restricted model without the dummies.

However, notice that the F-test provides us with a degree of confidence for rejecting the restricted model. It also shows the number of regressor that were added in the unstricted model– the neighborhood factor results in 54 dummies being added.

Other Methods for Model Selection

Other methods exist for model selection, but will not be covered here. You can read more about them on wikipedia or in your favorite statistics / ecomometrics textbook. Some of the more popular ones are the Likelihood Ratio test (LR), Akaike information criterio (AIC), and the Bayesian informaiton criterion (BIC).

Generalized Linear Models (GLM)

For the oridinary least squares (OLS) regression framework above, we were able to describe the fitted values $\hat{y}_i$ as the expected value of y_i given our regressors (x_i) and the estimated betas ($\hat{\beta}$):

$$\hat{y}_i = E(y_i|\hat{\beta}; x_i) \tag{14}$$

The generalized linear model (GLM) framework expands upon OLS by allowing for the response variable (y) to be of a wider range of distributions. To do this, we allow for inclusion of a **link function** that we will call g. However, in following the notation in Equation 14 above, we will reference its inverse, g^{-1}.

$$\hat{y}_i = E(y_i|g^{-1}, \hat{\beta}; x_i) \tag{15}$$

The link function is that which connects $\hat{y}_i$ to $E(y_i|\beta; x_i)$. For example, if $g(z) = ln(z)$, then $g^{-1}(z) = e^z$ and

$$\begin{aligned} ln(\hat{y}_i) &= E(y_i|\hat{\beta}; x_i) && (16)\\ &= \hat{\beta}_0 + \hat{\beta}_1 x_{i1} + \cdots + \hat{\beta}_1 x_{ik} && (17)\\ \hat{y}_i &= E(y_i|e^z, \beta; x_i) && (18)\\ &= e^{\hat{\beta}_0 + \hat{\beta}_1 x_{i1} + \cdots + \hat{\beta}_1 x_{ik}} && (19) \end{aligned}$$

Then our loss function (analogous to SSR but a more general) can be stated as:

$$\hat{\beta} = \underset{\beta}{argmin}\ ||y_i - E(y_i|g^{-1}, \beta; x_i)|| \quad (20)$$

It not important, in terms of this class, to know how this loss function is minimized. However, you should know that, unlike the SSR in OLS, this loss function does not always have a closed-form solution. So GLM models can take much longer to estimate than OLS models.

GLM Families

With the **glm()** function provided by R, we can estimate GLM models just like we did with **lm()**, we only need to provide the family and link function so it knows the type of response we are trying to model.

I will show examples of the gaussian (normal), logistic, and poisson regression. However, the R documentation for **glm()** provides a full list of possible models and the wikipedia page for "generalized linear models" has a great overview table of the possible models.

Gaussian (identity)

Distribution of y : Normal
Link Function : $g(z) = z$
Inverse Function : $g^{-1}(z) = z$

Notice that because the link is the identity, the GLM gaussian model ends up taking the same form as the OLS regression.

$$\begin{aligned} \hat{y}_i &= E(y_i|g^{-1}, \hat{\beta}; x_i) && (21)\\ &= E(y_i|z, \hat{\beta}; x_i) && (22)\\ &= \hat{\beta}_0 + \hat{\beta}_1 x_{i1} + \cdots + \hat{\beta}_1 x_{ik} && (23) \end{aligned}$$

We can estimate GLM version of our house price model as follows:

```
fit.gaussian<-glm(log_sale_price ~ log_living_sqft,
                  data=data.model,
                  family=gaussian(link="identity"))
summary(fit.gaussian)
```

```
##
## Call:
## glm(formula = log_sale_price ~ log_living_sqft, family = gaussian(link = "identity"),
##     data = data.model)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -13.6189   -0.3601    0.1332    0.4313    6.5902
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)     7.875186   0.033900   232.3   <2e-16 ***
## log_living_sqft 0.706351   0.004778   147.8   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 0.5178044)
##
##     Null deviance: 55485  on 85296  degrees of freedom
## Residual deviance: 44166  on 85295  degrees of freedom
## AIC: 185928
##
## Number of Fisher Scoring iterations: 2
```

Binomial (logit)

Distribution of y : Binomial
Link Function : $g(z) = ln(z/(1-z))$
Inverse Function : $g^{-1}(z) = exp(z)/(1+exp(z))$

$$\hat{y}_i = E(y_i|g^{-1}, \hat{\beta}; x_i) \tag{24}$$

$$= E(y_i|\frac{exp(z)}{1+exp(z)}, \hat{\beta}; x_i) \tag{25}$$

$$= \frac{exp(\hat{\beta}_0 + \hat{\beta}_1 x_{i1} + \cdots + \hat{\beta}_1 x_{ik})}{1 + exp(\hat{\beta}_0 + \hat{\beta}_1 x_{i1} + \cdots + \hat{\beta}_1 x_{ik})} \tag{26}$$

To test this model, we need some data that contains responses that represent successes and failures. Classification type problems can be framed in these terms if we consider one class to be a success and all other classes to be failures.

R provides a toy dataset called "iris" that it famously used to introduce classification. It contains observations on the sepal width/length and petal width/length of 3 species of iris.

Let's create a response variable that is 1 if the species is "setosa" and 0 otherwise, and then fit a model with one of the observed variables.

```
data("iris")
iris<-iris %>% mutate(is.setosa=ifelse(Species=="setosa",1,0))
```

```r
fit.logit<-glm(is.setosa ~ Sepal.Length,
               data=iris,
               family=binomial(link="logit"))
summary(fit.logit)
```

```
##
## Call:
## glm(formula = is.setosa ~ Sepal.Length, family = binomial(link = "logit"),
##     data = iris)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.25787  -0.27914  -0.04605   0.31399   2.14316
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)   27.8285     4.8276   5.765 8.19e-09 ***
## Sepal.Length  -5.1757     0.8934  -5.793 6.90e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 190.954  on 149  degrees of freedom
## Residual deviance:  71.836  on 148  degrees of freedom
## AIC: 75.836
##
## Number of Fisher Scoring iterations: 7
```

The coefficients of the logistic regression can be interpretted as β change in log-odds of a success with a 1 unit increase in the variable of interest.

What are log-odds? Well the odds of a success are defined as:

$$odds = \frac{p(success)}{p(failure)}$$

So the log-odds is simply the odds on log-scale:

$$log(odds) = log\left(\frac{p(success)}{p(failure)}\right) \tag{27}$$

$$= log(p(success)) - log(p(failure)) \tag{28}$$

For example, the probability of drawing a setosa from the dataset at random is $p = .33$. So the odds of drawing a setosa are $.33/.67 = .5$ and the log-odds are $log(.33) - log(.67) = -0.69$

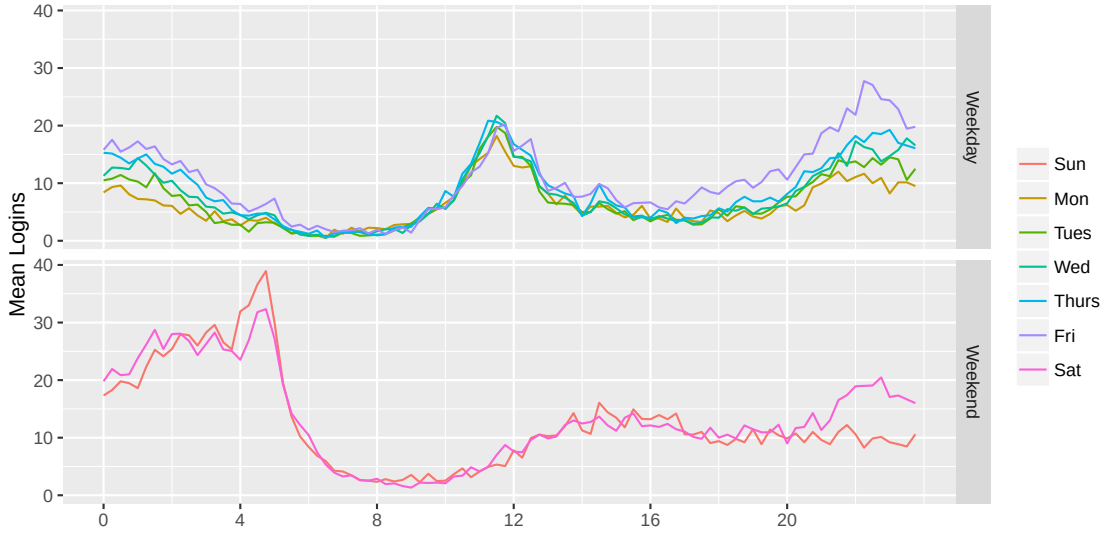
Poisson (log)

Distribution of y : Poisson
Link Function : $g(z) = ln(z)$
Inverse Function : $g^{-1}(z) = exp(z)$

$$\hat{y}_i = E(y_i|g^{-1}, \hat{\beta}; x_i) \tag{29}$$
$$= E(y_i|exp(z), \hat{\beta}; x_i) \tag{30}$$
$$= exp(\hat{\beta}_0 + \hat{\beta}_1 x_{i1} + \cdots + \hat{\beta}_1 x_{ik}) \tag{31}$$

The Poisson allows us to estimate response variables that represent counts over a fix period of time.

For example, I have some Uber logins for a specific geographic region. The original dataset is simply a vector of timestamps captured for 4 months. The data shown below have been aggregated to 15 minute intervals and show the cycles of logins for each day of the week.



Let's try to write down a model that captures the difference in behavior between the weekdays and weekends.

```
logins15<-logins15 %>%
  mutate(split=ifelse((wday=="Sun" | wday=="Sat"),"Weekend","Weekday") )
fit.poisson<-glm(log ~ factor(split) + factor(h) + factor(split):factor(h) ,
                 data=logins15,
                 family=poisson(link="log"))
```

The model above takes into consideration the weekday/weekend split, the hour of the day, and the interaction of the two together. Now let's test the model and see how many logins it predicts for an average **weekend at 4am** and for an average **weekday at 7am**.

```
predict(fit.poisson,
  newdata=data.frame(split=c("Weekend","Weekday"),h=c(4,7)),
  type="response")

##         1         2
## 31.883333  1.545139
```