Illustrations - Simple and Multiple Linear Regression

Steele H. Valenzuela

February 18, 2015

Illustrations for Simple and Multiple Linear Regression

February 2015

Simple Linear Regression


Setting: Calls the New York Auto Club are possibly related to the weather, with more calls occurring during bad weather. This example illustrates descriptive analyses and simple linear regression to explore this hypothesis in a data set containing information on calendar day, weather, and numbers of calls.

Libraries that we will use in R

```r
# REMEMBER, if these do not upload, it is because you have not used the command 'install.packages()
library(foreign) # to upload the State data set
library(ggplot2) # to plot
library(Hmisc) # for the command describe
```

Stata Data Set that we will use in R

Let’s load the data.

```r
url <- "http://people.umass.edu/biep640w/datasets/ers.dta"
dat <- read.dta(file = url)
```

Simple Linear Regression Variables:

Outcome Y = calls Predictor X = low

2. Preliminaries: Descriptives  After some searching of the web, several packages in R contain similar output to Stata’s describe command. Below is that exact command, but using the Hmisc package in R. Additionally, we’ll check for missing data because if we have any, it’s a minor nuisance.

```r
describe(dat)
```

```r
## dat
##
## 12 Variables 28 Observations
## -----------------------------------------------
## day
## n missing unique Info Mean .05 .10 .25 .50
## 28 0 28 1 12258 12070 12072 12076 12258
## .75 .90 .95
## 12440 12444 12446
##
```
### lowest : 12069 12070 12071 12072 12073
### highest: 12443 12444 12445 12446 12447
### "---------------------------------------------------------------------"

#### calls

<table>
<thead>
<tr>
<th>n</th>
<th>missing</th>
<th>unique</th>
<th>Info</th>
<th>Mean</th>
<th>.05</th>
<th>.10</th>
<th>.25</th>
<th>.50</th>
</tr>
</thead>
<tbody>
<tr>
<td>28</td>
<td>0</td>
<td>27</td>
<td>1</td>
<td>4319</td>
<td>1698</td>
<td>1739</td>
<td>1842</td>
<td>3062</td>
</tr>
<tr>
<td>.75</td>
<td>.90</td>
<td>.95</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6498</td>
<td>8132</td>
<td>8821</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### "---------------------------------------------------------------------"

#### fhigh

<table>
<thead>
<tr>
<th>n</th>
<th>missing</th>
<th>unique</th>
<th>Info</th>
<th>Mean</th>
<th>.05</th>
<th>.10</th>
<th>.25</th>
<th>.50</th>
</tr>
</thead>
<tbody>
<tr>
<td>28</td>
<td>0</td>
<td>21</td>
<td>1</td>
<td>34.96</td>
<td>15.70</td>
<td>21.90</td>
<td>29.75</td>
<td>35.00</td>
</tr>
<tr>
<td>.75</td>
<td>.90</td>
<td>.95</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>41.75</td>
<td>47.30</td>
<td>48.00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### "---------------------------------------------------------------------"

#### flow

<table>
<thead>
<tr>
<th>n</th>
<th>missing</th>
<th>unique</th>
<th>Info</th>
<th>Mean</th>
<th>.05</th>
<th>.10</th>
<th>.25</th>
<th>.50</th>
</tr>
</thead>
<tbody>
<tr>
<td>28</td>
<td>0</td>
<td>19</td>
<td>1</td>
<td>24.46</td>
<td>7.05</td>
<td>11.10</td>
<td>18.75</td>
<td>27.00</td>
</tr>
<tr>
<td>.75</td>
<td>.90</td>
<td>.95</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>32.00</td>
<td>34.30</td>
<td>38.25</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### "---------------------------------------------------------------------"

#### high

<table>
<thead>
<tr>
<th>n</th>
<th>missing</th>
<th>unique</th>
<th>Info</th>
<th>Mean</th>
<th>.05</th>
<th>.10</th>
<th>.25</th>
<th>.50</th>
</tr>
</thead>
<tbody>
<tr>
<td>28</td>
<td>0</td>
<td>19</td>
<td>1</td>
<td>37.46</td>
<td>15.00</td>
<td>19.20</td>
<td>32.00</td>
<td>39.50</td>
</tr>
<tr>
<td>.75</td>
<td>.90</td>
<td>.95</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>43.25</td>
<td>50.20</td>
<td>54.30</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### "---------------------------------------------------------------------"

#### low

<table>
<thead>
<tr>
<th>n</th>
<th>missing</th>
<th>unique</th>
<th>Info</th>
<th>Mean</th>
<th>.05</th>
<th>.10</th>
<th>.25</th>
<th>.50</th>
</tr>
</thead>
<tbody>
<tr>
<td>28</td>
<td>0</td>
<td>22</td>
<td>0.99</td>
<td>21.75</td>
<td>0.0</td>
<td>2.1</td>
<td>10.5</td>
<td>26.0</td>
</tr>
<tr>
<td>.75</td>
<td>.90</td>
<td>.95</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>31.0</td>
<td>36.6</td>
<td>39.3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### "---------------------------------------------------------------------"

#### rain

<table>
<thead>
<tr>
<th>n</th>
<th>missing</th>
<th>unique</th>
<th>Info</th>
<th>Sum</th>
<th>Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>28</td>
<td>0</td>
<td>2</td>
<td>0.66</td>
<td>9</td>
<td>0.3214</td>
</tr>
</tbody>
</table>

### "---------------------------------------------------------------------"

#### snow

<table>
<thead>
<tr>
<th>n</th>
<th>missing</th>
<th>unique</th>
<th>Info</th>
<th>Sum</th>
<th>Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>28</td>
<td>0</td>
<td>2</td>
<td>0.51</td>
<td>6</td>
<td>0.2143</td>
</tr>
</tbody>
</table>
Let's summarize the y-variable and x-variable, \texttt{calls} and \texttt{low}. This time around, we will create a function to sweep through these summary statistics rather than typing every command line by line.

```
# Let's make a scatterplot
ggplot(data = dat, aes(x = low, y = calls)) + geom_point() + ggtitle("Calls to NY Auto Club 1993-1994")
```
As you can see, the first chunk of commands `ggplot` captures the data as well as the aesthetics that will be overplotted on the plot. The next chunk indicates what kind of geometric object we would like to overlay our aesthetics, our x and y variables. Here, we are specifying `geom_point()` which is a simple scatterplot. Lastly, the command `ggtitle()` indicates the main title. You’ll see all of these chunks/arguments evolve throughout this handout, so please, stay tuned.

Next, we’re going to make another scatterplot, overlayed with a Lowess Regression.

```r
ggplot(data = dat, aes(x = low, y = calls)) + geom_point() + stat_smooth(method = "loess", se = FALSE)
```
Let us test for normality implementing the Shapiro-Wilk Test of Normality on the y-variable.

```r
shapiro.test(dat$calls)
```

```
##
## Shapiro-Wilk normality test
##
## data:  dat$calls
## W = 0.829, p-value = 0.0003628
```

Because of a small p-value of $3.6278 \times 10^{-4}$, we reject the null hypothesis of normality.

For now, let’s make a histogram of the variable **calls** overlayed with a normal curve.

```r
# first, let’s find the mean and standard deviation, which will assist us in creating a normal curve
mean(dat$calls); sd(dat$calls)
```

```
## [1] 4319
## [1] 2693
```

```
#Now, this is tricky but extremely helpful
ggplot(dat, aes(x = calls)) + geom_histogram(aes(y = ..density..), binwidth = 1454) +
stat_function(fun = dnorm, colour = "red", arg = list(mean = 4318.75, sd = 2692.564))
```
Prof. Bigelow mentions on her handout: > No surprise here, given that the Shapiro Wilk test reject normality. This graph confirms non-linearity of the distribution of \( Y = \text{calls} \).

3. Model Fitting (Estimation) We are now going to fit the model.

```r
fit1 <- lm(calls ~ low, data = dat)
summary(fit1)
```

```
## Call:
## lm(formula = calls ~ low, data = dat)
##
## Residuals:  
##    Min     1Q   Median     3Q    Max
## -3112 -1468  -214  1144  3588

## Coefficients:  
##            Estimate Std. Error t value Pr(>|t|)
## (Intercept) 7475.8      704.6   10.61  6.1e-11 ***  
## low         -145.2       27.8  -5.22  1.9e-05 ***  
## ---
## Signif. codes:  
##  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

## Residual standard error: 1920 on 26 degrees of freedom
## Multiple R-squared: 0.512, Adjusted R-squared: 0.493
## F-statistic: 27.3 on 1 and 26 DF,  p-value: 1.86e-05
```
From the `summary()` command, we see vital information, no doubt, but we are missing some other pieces of information that Stata executes wonderfully with only one line of code. We are missing the ANOVA table as well as the confidence intervals.

```r
anova(fit1)
```

## Analysis of Variance Table

# Response: calls

Df | Sum Sq | Mean Sq | F value | Pr(>F) |
--- | --- | --- | --- | --- |
low | 1.00e+08 | 1.00e+08 | 27.3 | 1.9e-05 *** |
Residuals | 9.55e+07 | 3.67e+06 | |
--- |
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```r
confint(fit1)
```

2.5 % 97.5 %
(Intercept) 6027.5 8924.24
low -202.3 -88.03

If you recall from assignment 3, we did create a function to call of this information.

```r
stata.regress <- function(x) {
  a <- summary(x)  # we are going to pass x to three commands as listed
  b <- anova(x)
  d <- confint(x)
  print(a); print(b); print(d)  # the last line typically specifies what one wants the function to print
}
```

```r
stata.regress(fit1)
```

## Call:

```
  lm(formula = calls ~ low, data = dat)
```

## Residuals:

```
  Min 1Q Median 3Q Max
  -3112 -1468 -214 1144 3588
```

## Coefficients:

```
  Estimate Std. Error t value Pr(>|t|)
(Intercept) 7475.8 704.6 10.61 6.1e-11 ***
low -145.2 27.8 -5.22 1.9e-05 ***
---
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

## Residual standard error: 1920 on 26 degrees of freedom
## Multiple R-squared: 0.512, Adjusted R-squared: 0.493
## F-statistic: 27.3 on 1 and 26 DF, p-value: 1.86e-05

## Analysis of Variance Table
Let's get back to the problem. From the summary output, given $y = \text{calls}$ and $x = \text{low}$, you should be able to draw up the fitted line:

$$\hat{y} = 7475.85 - 145.15 \times x$$

The 0.5121 indicates that 51% of the variability in calls is explained. Also, the overall F test significance level, $\Pr(>F)$, 27.2849, suggests that the straight line fit performs better in explaining variability in calls than does $\bar{Y} = \text{average # calls}$. 

4. Model Examination  Let’s examine the model by fitting a confidence interval band on the scatterplot.
5. Checking Model Assumptions and Fit  Let’s check the model with several residual plots.

```r
# nice ggplot2 function that displays a nice chunk of information. This will not be evaluated here but do run it during your own session. Also, we will run it in the last section.
fortify(fit1)

qplot(.fitted, .resid, data = fit1) + geom_hline(yintercept = 0) + geom_smooth(se = F) + ggtitle("Fitted Estimates against Residuals")
```

Next, let’s plot a qqplot, to take a closer look at the normality of our model (is that a proper phrase?).

```r
qplot(sample = .resid, data = fit1, stat = "qq") + geom_abline() + ggtitle("Normality of Residuals of Y = calls on X = low")
```
Next, let’s plot Cook’s Distance using two different geometric objects: points and bars.

```r
qplot(x = seq_along(.cooksd), y = .cooksd, data = fit1) + geom_point(stat = "identity") + ggtitle("Cook’s Distance")
```
That was points (obviously), and now, here are bars.

```r
qplot(x = seq_along(.cooksd), y = .cooksd, data = fit1) + geom_bar(stat = "identity") + ggtitle("Cook's Distance")
```
Professor Bigelow’s remarks: > For straight line regression, the suggestion is to regard Cook’s Distance values $> 1$ as significant. > Here, there are no unusually large Cook Distance values. > Not shown but useful, too, are examinations of leverage and jackknife residuals.

Lastly, here are our jackknife residual plot.

```r
qplot(x = .fitted, y = .stdresid, data = fit1) + geom_point(stat = "identity") + ggtitle("Jackknife Residuals v Predicted") + xlab("Linear Prediction") + ylab("Studentized Residuals")
```
Per Professor Bigelow’s remarks: > Recall - a jackknife residual for an individual is a modification for the solution for a studentized residual in which the mean square error is replaced by the mean square error obtained after deleting that individual from the analysis. > Departures of this plot from a parallel band about the horizontal line at zero are significant. > The plot here is a bit noisy but not too bad considering the small sample size.

Multiple Linear Regression

1. **A General Approach for Model Development** There are no rules nor single best strategy. In fact, different study designs and different research questions call for different approaches for model development. **Tip** – before you begin model development, make a list of your study design, research aims, outcome variable, primary predictor variables, and covariates.

As a general suggestion, the following approach as the advantages of providing a reasonably thorough exploration of the data and relatively little risk of missing something important.

**Preliminary** - be sure you have: 1. Checked, cleaned, and described your data 2. Screened the data for multivariate associations 3. Thoroughly explored the bivariate relationships

**Step 1** - Fit the “maximal” model The maximal model is the large model that contains all the explanatory variables of interest as predictors. This model also contains all the covariates that might be of interest. It also contains all the interactions that might be of interest. Note the amount of variation explained.

**Step 2** - Begin simplifying the model Inspect each of the terms in the “maximal” model with the goal of removing the predictor that is the least significant. Drop from the model the predictors that are the least significant, beginning with the higher order interactions. **Tip** - interactions are complicated and we are aiming for a simple model). Fit the reduced model. Compare the amount of variation explained by the reduced model with the amount of variation explained by the “maximal” model.

If the deletion of a predictor has little effect on the variation explained. Then leave that predictor out of the model. And inspect each of the terms in the model again.
If the deletion of a predictor has a significant effect on the variation explained, then put that predictor back into the model.

**Step 3** - Keep simplifying the model Repeat step 2, over and over, until the model remaining contains nothing but significant predictor variables.

**Beware of some important caveats** > Sometimes, you will want to keep a predictor in the model regardless of its statistical significance (an example is randomization assignment in a clinical trial). > The order in which you delete terms from the model matters. > You still need to be flexible to considerations of biology and what makes sense.

### 2. Introduction to Example

**Source:** Matthews et al. Parity Induced Protection Against Breast Cancer 2007.

**Research Question:** What is the relationship of \( Y = 53 \) expression to parity and age at first pregnancy, after adjustments for the potentially confounding effects of current age and menopausal status. Age at first pregnancy has been grouped and is either \( \leq 24 \) years or \( > 24 \) years.

Now, let’s get started and load the data.

```r
url2 <- "http://people.umass.edu/biep691f/data/p53paper_small.dta"
dat2 <- read.dta(file = url2)
```

### 3. (Further) Introduction to Example

Let’s explore the structure of the data as well as the data itself.

```r
str(dat2[1:5])

## 'data.frame': 67 obs. of 5 variables:
## $ agecurr : int 43 53 39 45 30 35 54 40 37 57 ... 
## $ pregnum : int 3 2 1 0 1 3 2 2 0 2 ... 
## $ menop : Factor w/ 2 levels "no","yes": 2 2 1 1 1 1 1 1 1 2 ... 
## $ p53 : num 3 2.4 2.2 3.5 5.5 ... 
## $ agefirst: Factor w/ 3 levels "never pregnant",...: 3 3 3 1 3 2 3 1 2 ... 
```

I cheated by looking at the environment window pane and I had seen that the dataset `dat2` contained 5 variables, hence the specified subset of one through five ([1:5]). Anyway, back to the point. We see that we have 5 variables, two of which are integers (`agecurr` & `pregnum`) while another is a number, decimals per se (`p53`). The last two variables, `menop` and `agefirst`, are factored variables with two and three levels, respectively.

Let’s look at the first 10 rows.

```r
head(dat2, 10)

## agecurr pregnum menop p53 agefirst
## 1 43 3 yes 3.0 age > 24 
## 2 53 2 yes 2.4 age > 24 
## 3 39 1 no 2.2 age > 24 
## 4 45 0 no 3.5 never pregnant 
## 5 30 1 no 5.5 age > 24 
## 6 35 3 no 6.0 age le 24 
## 7 54 2 no 4.0 age le 24 
## 8 40 2 no 4.0 age > 24 
## 9 37 0 no 2.0 never pregnant 
## 10 57 2 yes 5.0 age le 24 
```
And now, let's look at the Stata like function we previously loaded, `describe`.

```
describe(dat2) # once again, this output is not as nice as Stata's but it gets the job done

## dat2
## 5 Variables 67 Observations
## -------------------------------
agecurr
## n missing unique Info Mean .05 .10 .25 .50
## 67 0 38 1 39.63 16.6 20.0 28.5 40.0
## .75 .90 .95
## 49.5 58.0 60.0
##
## lowest: 15 16 18 19 20, highest: 58 60 62 63 75
## -------------------------------
pregnum
## n missing unique Info Mean
## 67 0 4 0.92 1.657
##
## 0 (16, 24%), 1 (9, 13%), 2 (24, 36%), 3 (18, 27%)
## -------------------------------
menop
## n missing unique
## 67 0 2
##
## no (48, 72%), yes (19, 28%)
## -------------------------------
p53
## n missing unique Info Mean .05 .10 .25 .50
## 67 0 19 0.99 3.251 2.000 2.000 2.500 3.000
## .75 .90 .95
## 4.000 4.625 5.000
##
## 1 2 2.20000004768372 2.25 2.40000009536743 2.5 2.75 3
## Frequency 3 7 1 1 1 5 7 9
## % 4 10 1 1 1 7 10 13
## 3.20000004768372 3.29999995231628 3.5 3.70000004768372 4
## Frequency 1 1 10 1 9
## % 1 1 15 1 13
## 4.30000019073486 4.375 5 5.19999980926514 5.5 6
## Frequency 2 2 4 1 1 1
## % 3 3 6 1 1
## -------------------------------
agefirst
## n missing unique
## 67 0 3
##
## never pregnant (16, 24%), age le 24 (32, 48%)
## age > 24 (19, 28%)
## -------------------------------
```

Let's check for pairwise correlations.
# Please follow the code as the factors variables tend to mess with the correlation function

```r
menop.new <- as.numeric(as.factor(dat2$menop))
agefirst.new <- as.numeric(as.factor(dat2$agefirst))
dat2.mat <- matrix(data = c(dat2$agecurr, dat2$pregnum, menop.new, dat2$p53, agefirst.new), nrow = 67, ncol = 5)
colnames(dat2.mat) <- c("agecurr", "pregnum", "menop", "p53", "agefirst")
cor(dat2.mat)  # unfortunately, p-values are not provided as well as column names. +1 to Stata

## agecurr pregnum menop p53 agefirst
## agecurr 1.0000 0.5416 0.72848 0.13397 0.4765
## pregnum 0.5416 1.0000 0.40205 0.44190 0.5765
## menop 0.7285 0.4021 1.00000 0.04498 0.2823
## p53 0.1340 0.4419 0.04498 1.00000 0.2021
## agefirst 0.4765 0.5765 0.28227 0.20206 1.0000
```

Because the p-values here are not displayed, we will quote from Professor Bigelow’s handout. > Only one correlation with Y = 53 is statistically significant, r(p53, pregnum) = 0.44 and a p-value of 0.0002. Note, that some of the predictors are statistically significantly correlated with each other: r(agefirst, pregnum) = 0.58 with a p-value less than 0.0001.

Now, let’s examine a pairwise scatterplot.

```r
pairs(dat2.mat, gap = 0, pch = ".")
```

Please note that my columns are arranged differently than Prof. Bigelow’s. Additionally, this plot is difficult to take in. It’s also not as useful as we were hoping.

Let’s take a closer look at it fitting a linear model and a lowess/loess model.
This is a lot of R code to take in so just trust me and plot it.

```r
ggplot(data = dat2, aes(x = pregnum, y = p53)) + geom_point() +
  stat_smooth(method = "lm", se = FALSE, colour = "darkblue", size = 1) +
  stat_smooth(method = "loess", se = FALSE, colour = "darkred", size = 1) +
  ggtitle(expression(atop("Assessment of Linearity", atop("Y = p53, X = pregnum"))))
```

## Warning: pseudoinverse used at 3.015
## Warning: neighborhood radius 2.015
## Warning: reciprocal condition number 3.1562e-16
## Warning: There are other near singularities as well.

The red line represents a lowess/loess model while the blue line represents a linear model. From here, we can confirm that we have a linear relationship.

Now, let’s fit the variable `agefirst` against `p53`.

```r
ggplot(data = dat2, aes(x = agefirst, y = p53)) + geom_point() +
  stat_smooth(method = "lm", se = FALSE, colour = "darkblue", size = 1) +
  stat_smooth(method = "loess", se = FALSE, colour = "darkred", size = 1) +
  ggtitle(expression(atop("Assessment of Linearity", atop("Y = p53, X = agefirst"))))
```

## geom_smooth: Only one unique x value each group. Maybe you want aes(group = 1)?
Assessment of Linearity

Y = p53, X = agefirst

We see that we get a warning and we cannot plot a linear model. We'll have to use dummy variables for this model.

Moving on, let's keep the same y-variable \textit{p53} and add the x-variable \textit{agecurr}, instead.

ggplot(data = dat2, aes(x = agecurr, y = p53)) + geom_point() +
  stat_smooth(method = "lm", se = FALSE, colour = "darkblue", size = 1) +
  stat_smooth(method = "loess", se = FALSE, colour = "darkred", size = 1) +
  ggtitle(expression(atop("Assessment of Linearity", atop("Y = p53, X = agecurr")))))
have a linear model.

4. Handling of Categorical Predictors: Indicator Variables  Let’s create dummy variables for age at first pregnancy: early and late. The best way to go about this in R is to add these new dummy variables to the original data frame, dat2.

```r
# we are adding new variables early and late to dat2 using the ifelse function
# it is a basic function in R as well as other programs. Enter ?ifelse in the console to find out more
dat2$early <- ifelse(dat2$agefirst == "age le 24", 1, 0)
dat2$late <- ifelse(dat2$agefirst == "age > 24", 1, 0)

table(dat2$agefirst, dat2$early)

##
##          0 1
## never pregnant 16 0
## age le 24 0 32
## age > 24 19 0

table(dat2$agefirst, dat2$late)

##
##          0 1
## never pregnant 16 0
## age le 24 32 0
## age > 24 0 19
```

And from the `table()` function, we see that our new variables are well-defined.
5. Model Fitting and Estimation

Model Estimation Set 1: Determination of best model in the predictors of interest. Goal is to obtain best parameterization before considering covariates.  

**MAXIMAL MODEL:** Regression of $Y = p53$ on all variables: $pregnum + [early, late]$

```r
# First, let's fit the pre-specified model and then run the our version of Stata's regression function on that model
fit2 <- lm(p53 ~ pregnum + early + late, data = dat2)

stata.regress(fit2)
```

```
## Call:
## lm(formula = p53 ~ pregnum + early + late, data = dat2)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -2.8603 -0.5703  0.0161  0.5161  2.6210
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept) 2.5703 0.2409 10.67 9.4e-16 ***
## pregnum 0.3764 0.2009 1.87 0.066 .
## early 0.1608 0.5556 0.29 0.773
## late -0.0677 0.5017 -0.13 0.893
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.964 on 63 degrees of freedom
## Multiple R-squared: 0.203, Adjusted R-squared: 0.165
## F-statistic: 5.35 on 3 and 63 DF,  p-value: 0.0024

## Analysis of Variance Table
##
## Response: p53
##              Df Sum Sq Mean Sq  F value  Pr(>F)
## pregnum      1 14.333  14.333 15.44733 8.5e-04 ***
## early        1  0.556  0.556  0.58724 0.4447
## late         1  0.020  0.020  0.020 0.89307
## Residuals   63 58.549  0.933
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Intercept) 2.0890 3.0517
## pregnum -0.0250 0.7778
## early -0.9495 1.2710
## late -1.0704 0.9349
```

The fitted line is:

$$p53 = 2.57 + (0.38) \cdot pregnum + (0.16) \cdot early - (0.07) \cdot late$$

20\% of the variability in $Y = 53$ is explained by this model ($R^2 = 0.20$). This model is statistically significantly better than the null model ($p$-value of F test $= 0.0024$).
Note!! We see a consequence of the multi-collinearity of our predictors \([\text{early}, \text{late}]\) and \(\text{pregnum}\) have NON-significant t-statistic p-values: \text{early} and \(\text{late pregnum}\) has a t-statistical p-value that is only marginally significant.

Now, let’s run a partial F-test on the variables \(\text{early}\) and \(\text{late}\).

```r
library(car) # May not need
library(survey) # be sure to install this package first before loading the library

## Wald test for early late
## in lm(formula = p53 ~ pregnum + early + late, data = dat2)
## F = 0.3052 on 2 and 63 df: p= 0.74

I had to search the web for this function as apparently, not many in the R community use this function. Anyway, we see that \text{early} and \(\text{late}\) are not statistically significant (p-value = 0.74). We may conclude that, in the adjusted model containing \(\text{pregnum}\), \([\text{early}, \text{late}]\) are not statistically significantly associated with \(Y = p53\).

Next, let’s run a partial F-test on the variable \(\text{pregnum}\).

```r
regTermTest(fit2, ~pregnum)
```

## Wald test for pregnum
## in lm(formula = p53 ~ pregnum + early + late, data = dat2)
## F = 3.511 on 1 and 63 df: p= 0.066

We see that it is marginally statistically significant (p-value = 0.0656). The null hypothesis is rejected and we may conclude that in the model that contains \([\text{early}, \text{late}]\), \(\text{pregnum}\) is marginally statistically significantly associated with \(Y = p53\).

Now that we have \(\text{pregnum}\) under our model belt, let’s fit a new model with only this variable.

```r
fit3 <- lm(p53 ~ pregnum, data = dat2)
stata.regress(fit3)
```
## Residuals:
## Min 1Q Median 3Q Max
## -2.8093 -0.5635 0.0212 0.6060 2.5212
##
## Coefficients:
## Estimate Std. Error t value Pr(>|t|)
## (Intercept) 2.564 0.209 12.28 < 2e-16 ***
## pregnum 0.415 0.105 3.97 0.00018 ***
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.953 on 65 degrees of freedom
## Multiple R-squared: 0.195, Adjusted R-squared: 0.183
## F-statistic: 15.8 on 1 and 65 DF, p-value: 0.000181
##
## Analysis of Variance Table
##
## Response: p53
## Df Sum Sq Mean Sq F value Pr(>F)
## pregnum 1 14.3 14.33 15.8 0.00018 ***
## Residuals 65 59.1 0.91
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## The fitted line is:
## \[ p53 = 2.56 + 0.41 \times \text{pregnum} \]

19.5% of the variability in \( Y = p53 \) is explained by this model (R-squared = .195). This model is statistically significantly more explanatory that the null model (p-value = 0.0002).

Now let’s run a model with the variables early and late.

```r
fit4 <- lm(p53 ~ early + late, data = dat2)
stata.regress(fit4)
```

## Call:
## lm(formula = p53 ~ early + late, data = dat2)
##
## Residuals:
## Min 1Q Median 3Q Max
## -2.613 -0.592 -0.113 0.558 2.387
##
## Coefficients:
## Estimate Std. Error t value Pr(>|t|)
## (Intercept) 2.570 0.246 10.47 1.7e-15 ***
## early 1.043 0.301 3.47 0.00094 ***
## late 0.645 0.333 1.94 0.05720 .
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## 22
Residual standard error: 0.982 on 64 degrees of freedom
Multiple R-squared: 0.159, Adjusted R-squared: 0.132
F-statistic: 6.03 on 2 and 64 DF, p-value: 0.00399

Analysis of Variance Table

Response: p53

<table>
<thead>
<tr>
<th>Df</th>
<th>Sum Sq</th>
<th>Mean Sq</th>
<th>F value</th>
<th>Pr(&gt;F)</th>
</tr>
</thead>
<tbody>
<tr>
<td>early</td>
<td>1</td>
<td>8.0</td>
<td>8.02</td>
<td>8.31</td>
</tr>
<tr>
<td>late</td>
<td>1</td>
<td>3.6</td>
<td>3.62</td>
<td>3.75</td>
</tr>
<tr>
<td>Residuals</td>
<td>64</td>
<td>61.7</td>
<td>0.96</td>
<td></td>
</tr>
</tbody>
</table>

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

(Intercept) 2.07975 3.061
early 0.44216 1.644
late -0.02033 1.311

Now, we're picking up speed and we are going to create a regression table. Once again, R does not have such a feature, so, +1 to Stata, again. Luckily, someone has created a package. Let’s check it out.

```r
library(stargazer)
```

Please cite as:

```r
stargazer(fit2, fit3, fit4, type = "text")
```

Dependent variable: p53

<table>
<thead>
<tr>
<th></th>
<th>(1)</th>
<th>(2)</th>
<th>(3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>pregnum</td>
<td>0.376*</td>
<td>0.415***</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(0.201)</td>
<td>(0.105)</td>
<td></td>
</tr>
<tr>
<td>early</td>
<td>0.161</td>
<td></td>
<td>1.043***</td>
</tr>
<tr>
<td></td>
<td>(0.556)</td>
<td></td>
<td>(0.301)</td>
</tr>
<tr>
<td>late</td>
<td>-0.068</td>
<td></td>
<td>0.645*</td>
</tr>
<tr>
<td></td>
<td>(0.502)</td>
<td></td>
<td>(0.333)</td>
</tr>
<tr>
<td>Constant</td>
<td>2.570***</td>
<td>2.564***</td>
<td>2.570***</td>
</tr>
<tr>
<td></td>
<td>(0.241)</td>
<td>(0.209)</td>
<td>(0.246)</td>
</tr>
</tbody>
</table>

Observations 67 67 67
During my short stint as a undergraduate political science minor, many of the models they fit in their papers are printed in this format, which must be read from vertically. Now, some of the epidemiological papers that I have stumbled upon are in this format. Back to the table.

Let’s choose model 2 as a good “minimally adequate” model: \( Y = p53 \) and \( X = \text{pregnum} \). This is why.

1. Model 1 is the maximal model with a R-squared value of 0.20
2. Model 2 drops \{early, late\} and it has an R-squared value that is minimally lower of 0.195
3. Model 3 drops \text{pregnum} and the R-squared drop is more substantial with a value of 0.159.

Model Estimation Set 2: Regression of \( Y = p53 \) on parity with adjustment for covariates
We’re going to pick up speed on this section. Let’s create a new maximal model.

```r
fit5 <- lm(p53 ~ pregnum + agecurr + menop, data = dat2)
stata.regress(fit5)
```

```r
##
## Call:
## lm(formula = p53 ~ pregnum + agecurr + menop, data = dat2)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -2.6056 -0.6065 -0.0172  0.4972  2.4465
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)                2.70431   0.44035   6.14  6.1e-08 ***
## pregnum                   0.49233   0.12457   3.95  0.0002 ***
## agecurr                   -0.00477   0.01364  -0.35  0.7276
## menopyes                  -0.27978   0.37769  -0.74  0.4616
##
## Signif. codes:  
## " "  \( p<0.1 \);  "*"  \( p<0.05 \);  "**"  \( p<0.01 \)
##
## Residual standard error: 0.955 on 63 degrees of freedom
## Multiple R-squared: 0.218, Adjusted R-squared: 0.181
## F-statistic: 5.85 on 3 and 63 DF,  p-value: 0.00137
##
## Analysis of Variance Table
##
## Response: p53
##                 Df Sum Sq Mean Sq  F value  Pr(>F)
## pregnum          1  14.3  14.33 15.73000 0.00019 ***
## agecurr          1   1.2   1.15  1.27000 0.26496
## menopyes         1   0.5   0.50  0.50000 0.46158
## Residuals       63  57.4   0.91
##
```
Let’s create a model with the variable `agecurr` dropped.

```r
fit6 <- lm(p53 ~ pregnum + menop, data = dat2)
```

And let’s pull everything together and create a regression table.

```r
stargazer(fit5, fit6, type = "text")
```
### Dependent variable:

<table>
<thead>
<tr>
<th></th>
<th>(1)</th>
<th>(2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>p53</td>
<td>0.492***</td>
<td>0.475***</td>
</tr>
<tr>
<td></td>
<td>(0.125)</td>
<td>(0.114)</td>
</tr>
<tr>
<td>pregnum</td>
<td>-0.005</td>
<td>-0.280</td>
</tr>
<tr>
<td></td>
<td>(0.014)</td>
<td>(0.378)</td>
</tr>
<tr>
<td>menopyes</td>
<td>-0.367</td>
<td>-0.281</td>
</tr>
<tr>
<td></td>
<td>(0.281)</td>
<td>(0.378)</td>
</tr>
<tr>
<td>Constant</td>
<td>2.704***</td>
<td>2.569***</td>
</tr>
<tr>
<td></td>
<td>(0.440)</td>
<td>(0.208)</td>
</tr>
<tr>
<td>Observations</td>
<td>67</td>
<td>67</td>
</tr>
<tr>
<td>R2</td>
<td>0.218</td>
<td>0.216</td>
</tr>
<tr>
<td>Adjusted R2</td>
<td>0.181</td>
<td>0.192</td>
</tr>
<tr>
<td>Residual Std. Error</td>
<td>0.955 (df = 63)</td>
<td>0.948 (df = 64)</td>
</tr>
<tr>
<td>F Statistic</td>
<td>5.847*** (df = 3; 63)</td>
<td>8.831*** (df = 2; 64)</td>
</tr>
</tbody>
</table>

---

Note: *p<0.1; **p<0.05; ***p<0.01

Finally, only using R-squared values, we see that both values of these models are similar, around 0.20. This was also similar to the model we used in the last section **Model Estimation Set 1** with only the variable *pregnum*. This is the model we will choose.

6. Checking Model Assumptions and Fit

Continuing from where we left off, let’s display the model we are going to check.

```
stata.regress(fit3)
```

```
##
## Call:
## lm(formula = p53 ~ pregnum, data = dat2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.8093 -0.5635  0.0212  0.6060  2.5212
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.5640     0.2089   12.28  < 2e-16 ***
## pregnum       0.4149     0.1056    3.94  0.00018 ***
## ---
## Signif. codes:  < 0.001 *** 0.001 ** 0.01 * 0.05 . 0.1   1
##
## Residual standard error: 0.953 on 65 degrees of freedom
## Multiple R-squared:  0.195,  Adjusted R-squared:  0.183
## F-statistic: 15.8 on 1 and 65 DF,  p-value: 0.000181
```
# Analysis of Variance Table

## Response: p53

<table>
<thead>
<tr>
<th>Df</th>
<th>Sum Sq</th>
<th>Mean Sq</th>
<th>F value</th>
<th>Pr(&gt;F)</th>
</tr>
</thead>
<tbody>
<tr>
<td>pregnum</td>
<td>1</td>
<td>14.3</td>
<td>14.33</td>
<td>15.8</td>
</tr>
<tr>
<td>Residuals</td>
<td>65</td>
<td>59.1</td>
<td>0.91</td>
<td></td>
</tr>
</tbody>
</table>

---

## Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

((Intercept) 2.1467 2.9804

pregnum 0.2064 0.6241

To access the predicted values of $Y$ in a new variables called $\hat{y}$, we can pull it from a string (aka the dollar sign, $§$).

```r
fit3$fitted.values
```

Or we can view the fitted values in a simple function from `ggplot2` package.

```r
fortify(fit3)  # remember, the `ggplot2` package must be installed and the library must be loaded
```

## p53 pregnum .hat .sigma .cooksd .fitted .resid .stdresid

| 1 | 3.000 | 0 0.03664 0.9550 1.423e-02 3.809 -0.80929 -0.86506 |
| 2 | 2.400 | 2 0.01634 0.9524 9.185e-03 3.394 -0.99404 -1.05152 |
| 3 | 2.200 | 1 0.02011 0.9555 6.993e-03 2.979 -0.77879 -0.82540 |
| 4 | 3.500 | 0 0.04795 0.9531 2.553e-02 2.564 0.93646 1.00692 |
| 5 | 5.500 | 1 0.02011 0.9063 7.329e-02 2.979 2.52121 2.67212 |
| 6 | 6.000 | 3 0.03664 0.9192 1.043e-01 3.809 2.19071 2.34166 |
| 7 | 4.000 | 2 0.01634 0.9575 3.413e-03 3.394 0.60596 0.64100 |
| 8 | 8.000 | 2 0.01634 0.9575 3.413e-03 3.394 0.60596 0.64100 |
| 9 | 2.000 | 0 0.04795 0.9579 9.247e-03 2.979 -0.56354 -0.60594 |
| 10 | 5.000 | 2 0.01634 0.9390 2.398e-02 3.394 1.60596 1.69882 |
| 11 | 5.000 | 3 0.03664 0.9485 3.080e-02 3.809 1.19071 1.27275 |
| 12 | 5.000 | 3 0.03664 0.9485 3.080e-02 3.809 1.19071 1.27275 |
| 13 | 1.000 | 0 0.04795 0.9395 7.118e-02 2.564 -1.56354 -1.68117 |
| 14 | 4.000 | 2 0.01634 0.9575 3.413e-03 3.394 0.60596 0.64100 |
| 15 | 4.000 | 2 0.01634 0.9575 3.413e-03 3.394 0.60596 0.64100 |
| 16 | 2.000 | 2 0.01634 0.9444 1.807e-02 3.394 -1.39404 -1.47465 |
```r
fit3$yhat <- fortify(fit3)$fitted
```
Moving on, let’s plot the model of the observed values against the fitted values. Ideally, the points will fall on the \( X = Y \) line (a diagonal line).

```
ggplot(data = dat2, aes(x = p53, y = fit3.yhat)) + geom_point() + stat_smooth(method = "lm") +
ggtitle(expression(atop("Model Check", atop("Plot of Observed vs Predicted"))))) +
ylab("Fitted Values")
```

![Model Check Plot of Observed vs Predicted](image)

Now, let’s check for the normality of residuals by plotting a standardized normality plot. This code will be a bit complicated but try to follow along. Unfortunately, after an hour, I was not able to figure out the code for this plot using the ggplot2 package.

```
fit3.resid <- fortify(fit3)$resid  # pulling the residuals out of the model
resid.stdnorm <- pnorm(fit3.resid)
plot(ppoints(length(resid.stdnorm)), sort(resid.stdnorm), main = "Model Check", sub = "Standardized Normality Plot of Residuals", xlab = "Observed Probability", ylab = "Expected Probability")
abline(0, 1)
```
Everything looks reasonable, so let’s move on.

Now, let’s do a quantile normal plot of the residuals. This, I can do in `ggplot2`.

```r
qplot(sample = fit3.resid, stat = "qq") + geom_abline() + xlab("Inverse Normal") + ylab("Residuals") + ggtitle(expression(atop("Model Check", atop("Quantile-Normal Plot of Residuals")))))
```
We are almost done. Let’s run the Shapiro Wilk test of normality on the residuals.

```r
shapiro.test(fit3.resid) # this does not add up to the same result in STATA...???
```

```r
## Shapiro-Wilk normality test
## data: fit3.resid
## W = 0.9849, p-value = 0.5943
```

Lastly, after searching the web for model misspecification and the linktest, I have had no such luck. Let’s run the Cook-Weisberg Test for Homogeneity of variance of the residuals. In R, this test is known as the Score Test for Non-Constant Error Variance.

```r
ncvTest(fit3)
```

```r
## Non-constant Variance Score Test
## Variance formula: ~ fitted.values
## Chisquare = 1.561  Df = 1  p = 0.2115
```

We see that the null hypothesis of homogeneity of variance of the residuals is not reject as we have a p-value of 0.21.

Next, let’s plot the fitted values against the residuals. This is a graphical assessment of constant variance of the residuals.
We see that the variability of the residuals looks reasonably homogeneous, confirming the Cook-Weisberg test result.

Last of the last, let’s check for outlying, leverage, and influential points - look for values > 4.
Well, we’re done. Pat yourself on the back for making it through. The above plot looks nice. Not only are there no Cook distances greater than 4, they are all less than 1!