An Introduction to R

R Workshop 2: Linear and Logistic Regression

MBA542: Baker University

January 15th, 2020

Contents

1 Introduction

In this applied homework assignment you will learn how to use ordinary least squares regression and logistic regression with real data. You can find the homework questions at the end of this document.

2 Workshop Procedures

The R code for ordinary least squares regression and logistic regression that will be used in two respective case studies (cars dataset and breast cancer dataset) is presented below.

2.1 Ordinary Least Squares Regression

Call in the libraries:

```
library(ggplot2)
library(dplyr)
```

Note: if R indicates that it cannot find the package, you likely will need to run the install.packages() command for that package again.

R has a pre-loaded dataset on cars. This data has the speed (*speed*) a car is going and the corresponding stopping distance (*dist*). Use the assignment code to call the data "cars". The code to do so follows:

```
cars <- cars
```

It is possible to look at the first five rows of data with a *head* command.

```
cars %>% head
```

You can plot data from this dataset with *speed* on the x-axis and *dist* on the y-axis. The *geom_point* command plots each point separately. The *geom_smooth* command plots a fitted line to the data.

```
cars %>% ggplot(aes(x = speed, y= dist)) +
  geom_point()+
  geom_smooth(method="lm")
```

2.1.1 Linear Model in R

The fitted line is a regression using an ordinary least squares approach which minimizes the error terms between the line and each observation. We can run a regression to see the summary information on this fitted line. Here, the command first calls the dataset (cars), runs a regression (the lm command which stands for linear model), and then produces a summary of the regression output.

```
cars %>% lm(dist ~ speed, data = .) %>% summary
```

The summary command produces output, reproduced below, which shows the coefficients, as well as the statistical significance of the coefficients. The output also shows the R-squared and F-statistic of overall significance. The R-squared number shows the ratio of explained variation to total variation. As such, it can be interpreted as an indication of how well the model explains the variation in the observed data.

The F-statistic of overall significance compares a model with no predictors to the specified model. The status quo (null hypothesis) is that the intercept-only model and the specified model do equally good jobs explaining the variation in the data. The alternative hypothesis is that the explanatory power of the specified model is better than the intercept-only model. You can look at the p-value and if it is below your specified level of significance, you can reject the null hypothesis in favor of the alternative.

```
Call:
lm(formula = dist ~ speed, data = .)

Residuals:
    Min      1Q  Median      3Q     Max
-29.069  -9.525  -2.272   9.215  43.201

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -17.5791     6.7584  -2.601   0.0123 *
speed         3.9324     0.4155   9.464 1.49e-12 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 15.38 on 48 degrees of freedom
Multiple R-squared:  0.6511,Adjusted R-squared:  0.6438
F-statistic: 89.57 on 1 and 48 DF,  p-value: 1.49e-12
```

2.1.2 Residuals from OLS

We know from the theory about linear regression that there are assumption made about the residuals. We can examine these assumptions by plotting the residuals. R makes this a relatively accessible process. The code below shows you how to do this:

```
model <- lm(dist ~ speed, data = cars)
residuals <- residuals(model)
hist(residuals)
```

The first command saves the linear model as an object called model in the R environment. The second command extracts the error terms, also called residuals. The third command graphs the histogram of those residuals. You can see that this histogram is slightly skewed, indicating that there may be some problem with the assumption of linearity in the data on cars.

2.1.3 Range of Possible Coefficients

We know that the estimate about the coefficient from the linear regression is only a statistical guess of the true parameter. As such, it can be important to think about a possible range of values that the coefficient might take. To do this, we can compute confidence intervals. The code to do this is given in the following fashion:

```
cars %>% lm(dist ~ speed, data = .) %>% coefficients
cars %>% lm(dist ~ speed, data = .) %>% confint
```

Slightly more work with thinking about different coefficients. This code gives an example of what it would look like if we considered realized values of a range of coefficients on the explanatory variable. Not necessary for the homework questions.

```
predict_dist <- function(speed, theta_1)
  data.frame(speed = speed,
             dist = theta_1 * speed,
             theta = as.factor(theta_1))

cars %>% ggplot(aes(x = speed, y = dist, colour = theta)) +
  geom_point(colour = "black")+
  geom_line(data = predict_dist(cars$speed, 2))+
  geom_line(data = predict_dist(cars$speed, 3))+
  geom_line(data = predict_dist(cars$speed, 4))+
  scale_color_discrete(name=expression(theta[1]))

##Plot errors to find out best estimate for theta_1

thetas <- seq(0,5, length.out = 50)
fitting_error <- Vectorize(function(theta)
  sum((theta * cars$speed - cars$dist)**2))

data.frame(thetas = thetas, errors = fitting_error(thetas)) %>%
  ggplot(aes(x = thetas, y=errors))+
  geom_line()+
  xlab(expression(theta[1])) + ylab("")
```

2.1.4 Evaluating Regression Models

Suppose we want to look at a model which is linear and a model with a non-linear term and compare the goodness of fit. That is to say, we want to validate the models. First, lets look at two possible models:

```
linear <- cars %>% lm(dist ~ speed, data = .)
non_linear <- cars %>% lm(dist ~ speed + I(speed^2), data = .)
```

To compare the two models, we need a measure of how well the curve they generate fits the data. To do this, we need to look at something called the root mean squared error. First, you take each error term for each data point, square it to avoid negative numbers, and then add each of the squared error terms together and divide by the number of observations. The final step is to take the square root. In R, this would look like:

```
rmse <- function(x,t) sqrt(mean(sum((t-x)^2)))
rmse(predict(linear, cars), cars$dist)
rmse(predict(non_linear, cars), cars$dist)
```

Running this code, you can see that the non-linear model produces a slightly smaller root mean squared error. This indicates that the non-linear model fits slightly better than the linear model. This information is consistent with the error term histogram we plotted earlier which showed that the linear model was not quite appropriate for the data.

2.2 Logistic Regression

First, load in the relevant packages and call in the Breast Cancer data. Recall that if you receive an error message about the package not being found, you may have to use the install.packages() command to first get it into your library.

```
library(mlbench)
library(dplyr)
data("BreastCancer")
```

We have seen the Breast Cancer data before so are somewhat familiar with the different variables. Notice that the primary outcome variable, called Class in the dataset, is binary. This means that if we want to run a predictive model to see which factors are associated with the Class designation, we cannot use a linear model, which requires the outcome to be continuous. As such, this provides an opportunity to explore the function of a logistic regression, which is designed for binary outcomes.

2.2.1 Plotting the Data with Jitter

First, we should plot the data. We want the variable of interest, Class, to be on the y-axis. This will be our explained variable. We want to explore which possible explanatory variables predict whether Class is designated malignant or benign, the two possible outcomes. To explore this relationship, we can generate an initial plot:

```
BreastCancer %>% ggplot(aes(x = Cl.thickness, y=Class))+
  geom_jitter(height = 0.05, width = 0.3, alpha = 0.4)
```

Here, we have a new command: *geom_jitter*. This is quite a nice option in the set of possible ggplot commands. It is particularly used when you might have data points that overlap. For instance, compare the geom jitter command with the geom point command. These would be the same plot if the data points did not overlap.

```
BreastCancer %>% ggplot(aes(x = Cl.thickness, y=Class))+
  geom_point()
```

In the geom jitter plot, it becomes apparent that tumors which are somewhat less thick are more likely to be designated benign while tumors that are somewhat more think are more likely to be designated malignant. This plot is encouraging and indicates we should look at the log odds ratio to see what a unit change in the cl.thickness variable does to the log odds ratio of a tumor being classified as malignant. To do this, we should first map the Class variable from a binary string variable to a binary numeric variable. This is similar to an exercise we performed in the first applied homework description packet.

```
map_bc <- function(x) {
  ifelse(x == "benign", 0,
         ifelse(x == "malignant", 1, NA))
}
```

Recall that you could also create a dictionary if you had more than two possible outcomes. This will be helpful for one of the homework exercises which asks you to map a categorical string variable with three possible outcomes. If you wanted to use the dictionary approach, the command would look something like this:

```
dict <- c("benign" = "0", "malignant" = "1")
map_bc <- function(x) dict[as.character(x)]
```

It will be important to be able to add this new numeric value into your Breast Cancer dataframe. The code to do this is as follows:

```
BreastCancer$tumor <- map_bc(BreastCancer$Class)
```

Here, you can read the code on the left as first identifying the data frame BreastCancer. The part after the dollar sign is the column. Because a column with this name does not currently exist in the data frame, R will create a new column which is specified in the commands which follow the assignment code. We have put after the assignment code the function *map_bc* which takes the specified column, Class, and runs it through the function converting the string variable into a numeric variable.

2.2.2 Explaining the Binary Outcome

We can now take the newly created numeric outcome variable and graph both the jitter plot as well as a smooth line, similar to the fitted linear model that we saw before. To do this, we can adjust the thickness variable slightly, something that is not necessary in the homework questions given the structure of that data. Then, we can create the ggplot with the jitter and the smooth geometric objects generated. The code looks as follows:

```
BreastCancer %>%
  mutate(Cl.thickness.numeric =
           as.numeric(as.character(Cl.thickness))) %>%
  mutate(IsMalignant = ifelse(Class == "benign", 0, 1)) %>%
  ggplot(aes(x = Cl.thickness.numeric, y = IsMalignant))+
  geom_jitter(height = 0.05, width = 0.3, alpha = 0.4)+
  geom_smooth(method = "glm", method.args = list(family = "binomial"))
```

The mutate command is something we saw in the first week. This allows us to adjust the thickness variable to go from an ordered factor variable to a numeric variable. This makes the plot slightly more aesthetically appealing. It is not necessary for the computation of the logistic function. The subsequent lines create the logistic plots. First, we recreate the jitter plot. Then we add a smoothed plot using a binomial distribution (when the outcome is binary) and a glm (generalized linear model) as opposed to the standard lm (linear model) that we saw in the ordinary least squares regression.

2.2.3 Obtaining the Logistic Regression Coefficient

The final step in this analysis is to obtain the coefficient estimates. The code to do this is as follows:

```
BreastCancer %>%
  mutate(Cl.thickness.numeric =
           as.numeric(as.character(Cl.thickness))) %>%
  glm(tumor ~ Cl.thickness.numeric,
      family = "binomial",
      data = .)
```

The output from this regression is given below.

```
Call:  glm(formula = IsMalignant ~ Cl.thickness.numeric, family = "binomial",
    data = .)

Coefficients:
         (Intercept)  Cl.thickness.numeric
             -5.1602                0.9355
```

```
Degrees of Freedom: 698 Total (i.e. Null);  697 Residual
Null Deviance:     900.5
Residual Deviance: 464.1  AIC: 468.1
```

You can see here a variety of informative calculations. First, notice the coefficients reported. Recall that the interpretation of the coefficient on the explanatory variable has a different interpretation here than it did in the linear regression model. Here, the coefficient on the explanatory variable represents the change in the log odds ratio for a unit change in the explanatory variable.

The output also reports the degrees of freedom, the null deviance, the residual deviance and the AIC number. The degrees of freedom represents the number of values that are free to vary. If you have a dataset with 100 observations, the degrees of freedom is set to 100-1 (99). As you add explanatory variables, you should remove an additional degree of freedom. Here, for instance, we can see that we have 699 observations. That means that our degrees of freedom would be 698 and after accounting for the single explanatory variable, we reduce it another one to 697.

Deviance is a measure of model fit. The null deviance indicates what the deviance would be with an intercept-only model. That is to say, the null deviance indicates how well the model fits the data if the only parameter were the intercept. The residual deviance indicates the deviance for the specified model. When the residual deviance is smaller than the null deviance, the specified model has some explanatory power.

2.2.4 Evaluating a Classification Model

If you want to do classification rather than regression, then the root mean square error is not the function to use to evaluate the model. With classification, you want to know how many data points are classified correctly and how many are not.

We can see this with the example of classifying tumors from the Breast Cancer data. First, let's add the variables of interest to the original dataframe. This time, let's add in one additional explanatory variable so we can see how that works.

```
formatted_data <- BreastCancer %>%
  mutate(Cl.thickness.numeric = as.numeric(as.character(Cl.thickness)),
         Cell.size.numeric =
           as.numeric(as.character(Cell.size))) %>%
  mutate(IsMalignant = ifelse(Class == "benign", 0, 1))

fitted_model <- formatted_data %>%
  glm(IsMalignant ~ Cl.thickness.numeric + Cell.size.numeric, data = .)
```

The first set of commands above adds three new variables to the dataframe: Cl.thickness.numeric, Cell.size.numeric, and IsMalignant. The second set of code creates an object which has the outcomes from the logistic regression model.

We can now construct the prediction matrix. First, we need to take the probabilities created by the model and determine a cutoff for the classification of numeric probabilities to one of the two possible outcomes. Here, the 50 percent demarcation seems to make sense. So, for tumors which are deemed at least 50 percent likely to be malignant, we will assign the malignant category. For tumors which are deemed less than 50 percent likely to be malignant, we will assign the benign category. You can see the outcome of this process in the following code:

```
predict(fitted_model, formatted_data, type = "response") %>% head

classify <- function(probability) ifelse(probability <0.5, 0, 1)
classified_malignant <- classify(predict(fitted_model, formatted_data))

table(formatted_data$IsMalignant, classified_malignant)

table(formatted_data$IsMalignant, classified_malignant,
      dnn=c("Data", "Predictions"))
```

We can take this idea and create a confusion matrix that will help us to calculate the predictive accuracy of the model. This is a somewhat intuitive calculation: you take the number of true positives and true negatives and divide by the total number of observations. In our confusion matrix, the numerator would be the sum of the diagonal: the number of tumors which were malignant and identified as malignant plus the number of tumors which were benign and identified as benign.

```
confusion_matrix <- table(formatted_data$Class, classified_malignant,
                          dnn=c("Data", "Predictions"))

accuracy <- sum(diag(confusion_matrix))/sum(confusion_matrix)
accuracy
```

3 Homework Questions

Each of these homework questions uses the Lasagna dataset provided on Moodle. Import this dataset into your R environment. Remember to run the necessary libraries each time that you start a new R session.

Question 1: Map the categorical string variables DwellType and PayType to categorical numeric variables. Use the table command to obtain the count of each category. Report these counts in your homework submission.

Question 2: Write out the model for a linear regression which tries to explain the level of income using the dwelling type, credit card debt, age, pay type and car value. Explain the interpretation of each of the coefficients on the respective variables.

Question 3: Run the linear model and obtain estimates for the coefficient on each explanatory variable. Report these coefficients in your homework submission and include an interpretation of their meaning.

Question 4: Obtain an histogram of the error terms in the linear model from question 3. Does it appear that the error terms are normally distributed? Why is this important? Include the histogram in your homework submission.

Question 5: Create a new variable called outcome which is equal to one if the HaveTried variable is yes and is equal to zero if the HaveTried variable is equal to no. This sets up a binary outcome variable which will be excellent for a logistic regression. Generate a count using the table command of the number of ones and the number of zeros. Report this in your homework submission. (Recall: this kind of mapping was introduced in the first lecture - look back at those notes for map class if you have forgotten how to do this).

Question 6: Use the ggplot command combined with a jitter geom to visualize the outcome variable against age, weight, income, car value, mall trips, and credit card debt each in a separate plot. Include your plots in your homework submission. Which of these seems to be the most predictive of whether someone has tried lasagna?

Question 7: Incorporate the GLM line using the geom smooth command. What does this line indicate about each variable's predictive power regarding whether the individual was likely to have tried lasagna? Include the plots in your homework submission.

Question 8: Run the logistic regression and obtain the coefficients for each explanatory variable. Run these regressions separately for each variable (report these coefficients in your homework submission) and run a full model (report these coefficients in your homework submission). Provide an interpretation of each coefficient.