Lab 12. Logistic Regression

# Introduction

Logistic regression uses one or more numeric variables to predict the probability of a binomial y variable - e.g. does the price of a shirt predict if people will (1) or won’t (0) buy a shirt, does the amount of water you give a parrot every day determine if it will (1) or will not (0) curse at you, etc. In this lab you will walk through interpreting regression with a fake data set, and then you will use a real dataset to solve the mystery of why so many people in Florida let their pet reptiles go loose in the wild.

Well, okay, the United States as a whole but like… it’s mostly a Florida problem TBH.

# Learning Outcomes

By the end of this lab you should be able to:

* Use glm() to make a binomial regression
* Use predict() and round() to predict on new data
* Use the confusionMatrix() function from the caret package to determine accuracy of your predictions
* Determine if a continuous variable can predict a binomial variable
* Interpret the meaning of a positive or negative slope in a logistic regression
* Find good statistical talking points to yell at the pet trade industry?

# Part 1: Fake Data

## Part 1.1. Make The Data

For a logistic regression we need to make two groups - one that is a “positive” result (1) and one that is a “negative” result (0). We also need some sort of predictor x variable.

positive <- data.frame(y = 1, x = rnorm(n = 50, mean = 50, sd = 3))  
negative <- data.frame(y = 0, x = rnorm(n = 50, mean = 42, sd = 3))  
together <- rbind(positive, negative)

Let’s take a look at them first, using a density diagram. Use the as.character() function to remind R that 0/1 is a category, and the alpha=.7 argument to make things see-through.

library(ggplot2)  
  
ggplot(together, aes(x = x, fill = as.character(y)))+  
 geom\_density(alpha = .7)

This data could be anything, but you can see pretty clearly that the 0 and 1 categories are different. Some examples of what this data could be:

* People who get paid more are more likely to be happy (1) than unhappy (0).
* Reptiles that are bigger are more likely to be released to the wild (1) than kept forever (0).
* People with longer hair are more likely to be hippies (1) than not hippies (0).
* Greater amounts of vitamin D intake during the winter is more likely to make you happy (1) than unhappy (0).

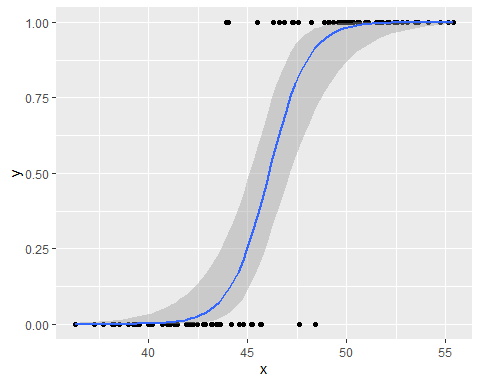
In this sense, **a logistic regression is very much like a t-test, but instead of saying “these are different” you’re asking “can I use x to predict these categories?”**

## Part 1.2 Plot the Data as A Bivariate Distribution

You can also use ggplot to view this data as a scatterplot just like you would otherwise, but to use the geom\_smooth() function you will need to do a little bit of manipulation. Specifically, you are using the glm() function to build this model. This is a general function that is similar to lm() but more “general” hence the name **general linear model**.

Because glm() can take more arguments, you have to specify that this is a **binomial** function, where you only have two options.

ggplot(together, aes(x = x, y = y))+  
 geom\_point()+   
 stat\_smooth(method="glm", method.args=list(family="binomial"))



## Part 1.3 Model Building

Now you can see that there is a relationship! Let’s build a model to test that.You can use the glm() function for real here, making sure to specify that this is a **binomial** model. Just like before, you can use the summary() function to get more information.

model <- glm(y~x, data = together, family=binomial)  
summary(model)

Interpreting this information is very similar to how you interpret the output for linear regression - there are the estimated slopes and the *p* values just as before. The biggest difference is that it does not output an R2 value.

In this example there is a statistically significant relationship - the *p* value for x is very small. Additionally, the slope for x is positive - that means that as x increased, so did y.

## Part 1.4 How Accurate?

R doesn’t give you an R2 value for logistic regressions. This is typical for most functions in R that deal with non-linear information, because the R2 value is no longer very helpful.

So how do we measure the accuracy and goodness of fit of the model? For most of these models, we look at the actual accuracy of the model - essentially, does the model predict the 0’s and 1’s correctly when it is used on the original data.

To start with, use the predict() function to create a new column of what the model would predict for each x value. You have to add type = "response" here to get a logistic regression result.

together$predict <- predict.glm(model, newdata=together, type = "response")

You’ll notice it hasn’t rounded the values. That’s fine, it likes to tell you how certain it is essentially of being a 1. You can use the round() function to make a column of rounded values.

together$predict2 <- round(together$predict)

Now, we’re going to make a **confusion matrix** - that’s a fancy term for tallying up how many of these predictions were wrong in either direction. To do this, you will need to use the install.packages() function to install caret and e1071. Once installed, load them using library():

library(caret)  
library(e1071)

Note that there is a dplyr workaround if you cannot get caret to install - skip to part 1.5 if that remains a problem.

For the confusion matrix, we have to do a little bit of wiggling. The confusionMatrix() function gets fussy when given what it thinks is the wrong kind of data - it wants factors, not numbers. Use the as.factor() function to get it to stop being silly.

confusionMatrix(data=as.factor(together$predict2), reference=as.factor(together$y))

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 48 3  
## 1 2 47  
##   
## Accuracy : 0.95   
## 95% CI : (0.8872, 0.9836)  
## No Information Rate : 0.5   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.9   
##   
## Mcnemar's Test P-Value : 1   
##   
## Sensitivity : 0.9600   
## Specificity : 0.9400   
## Pos Pred Value : 0.9412   
## Neg Pred Value : 0.9592   
## Prevalence : 0.5000   
## Detection Rate : 0.4800   
## Detection Prevalence : 0.5100   
## Balanced Accuracy : 0.9500   
##   
## 'Positive' Class : 0   
##

First off, overall accuracy here is pretty high! It has calculated the confidence intervals for you, and the accuracy is between 0.887 and 0.984 (your results will slightly differ).

As far as which groups were wrong, look at the little square of values at the top. 3 values were predicted as 0 when they were actually 1, and 2 values were predicted as 0 when they should have been 1.

## Part 1.5 Workaround

This version of looking at a confusion matrix is a bit more difficult and time consuming, but if you can’t get caret to install correctly the alternate option is to use dplyr and reshape2.

library(dplyr)  
library(reshape2)

To make a confusion matrix using dplyr, you’ll want to get rid of any NA values using filter() - not that important here, but it will be for the reptiles dataset. THen you’ll want to create a column that determines if the prediction was correct or not (**Wrong**) using mutate(). Then group your data by the

long <- together %>%  
 filter(!is.na(predict2)) %>%  
 mutate(Wrong = ifelse(y == predict2, "Right", "Wrong")) %>%  
 group\_by(y, predict2, Wrong) %>%  
 tally()

You can interpret your confusion matrix from the above table, but if you want it to look like the confusion matrix, you’ll need to use the dcast() function:

wide <- long %>%  
 dcast(y~predict2, value.var = "n")

Both way show you that this model is very accurate, with 47 of the 0’s being predicted as 0’s, and only 3 of the 0 values being incorrectly predicted as 1.

But it can be helpful to have a percentage accuracy as well. This setup is very similar to how we calculated above, but instead of grouping by **y**, **predict2**, and **Wrong**, we’re just tallying up how many predictions were correct or not.

acc <- together %>%  
 filter(!is.na(predict2)) %>%  
 mutate(Wrong = ifelse(y == predict2, "Right", "Wrong")) %>%  
 group\_by(Wrong) %>%  
 tally()   
  
AccuracyPercentage <- 100\*(acc[1,2]/(acc[1,2]+acc[2,2]))

## Part 1.6 Summarize

If you were asked to summarize this model, you would want to report three different things:

* The slope *p* value, for significance
* The accuracy of the model, as a measure of goodness of fit
* Whether or not the categories were evenly predicted.

This last point is very important, especially in datasets that are uneven. If you have 4 points that are 0, and 96 points that are 1, then a model that just predicts everything as 1 would have a very high accuracy (96%). However, that doesn’t mean it’s very good at predicting how many 0’s there should be.

A good summary might say something like:

The x value was a strong predictor of y, and was statistically significant (p <0.001). The overall model accuracy was high (0.95), and it seems that this continuous variable has a strong relationship with the binomial variable. Misclassifications were few, and no category was preferred over another.

## Part 2: Real Data

You will be using data from a 2019 study called “Pet problems: biological and economic factors that influence the release of alien reptiles and amphibians by pet owner,” available on Data Dryad: <https://datadryad.org/stash/dataset/doi:10.5061/dryad.j2n732c>

I have placed a mildly modified version of this data on D2L, called **reptile.release.csv**. The authors very helpfully provided a short summary of what each of the column names were, listed below with a few modifications to make herpetology terms more accessible.

1. col\_sp: species name, referenced to Catalogue of Life (<http://www.catalogueoflife.org/col/>)
2. col\_sp\_id: catalogue of life unique ID
3. col\_class: class of species, referenced to Catalogue of Life
4. col\_order: order of species, referenced to Catalogue of Life
5. clade: order/suborder of species (used in manuscript). Derived by authors using Catalogue of Life
6. col\_family: family of species, referenced to Catalogue of Life
7. col\_genus: genus of species, referenced to Catalogue of Life
8. sum\_qty: total number of imports for a given species from LEMIS database from 1999 to 2012 (see manuscript for details)
9. median\_price: median price of a given species from 2012 - 2016. Data comes from online vendors. This has been adjusted for inflation to December 2016 US dollars.
10. resid\_time: number of years a given species was either imported or listed for sale by an online vendor
11. max\_life\_yr: longevity of species (years)
12. adult\_mass\_g: adult mass (grams)
13. age\_maturity\_d: number of days species takes to reach sexual maturity
14. adult\_svl\_cm: adult Snout Vent length (nose to tail tip) in centimeters
15. clutch\_size: number of eggs laid or offspring birthed in one clutch\_size
16. clutches\_per\_year: number of clutches laid per year
17. fecundity: ‘clutch\_size’ multiplied by ‘clutches\_per\_year’, derived by authors
18. lht\_source: description of what database each life history traits was derived from
19. kraus\_c\_edd\_1999: whether a species has been recorded as free-living/released (1) or not (0)

### QUESTION 1: For any logistic regression to determine if a particular x variable influenced reptile release, which column would be your y axis?

1. clutch\_size
2. sum\_qty
3. resid\_time
4. kraus\_c\_edd\_1999

### QUESTION 2: You will not use col\_sp, col\_sp\_id, col\_class, col\_order, clade, col\_family, col\_genus, or lht\_source as variables in any of your logistic regressions. Why?

1. Because you told me not to (this is not the answer)
2. Because they have too many digits
3. Because they are categorical data

The authors of this paper wanted to set out to find specific, easily identifiable features of the pet trade that made a reptile or amphibian more likely to be released. They thought that certain aspects of different species could change the cost of care and how much a person valued their pet, both of which would influence the likelihood of a person releasing a pet. Your job will be to run two different logistic regressions, one on a variable you think would increase someone’s likelihood of releasing a pet and one you think would decrease it.

### QUESTION 3: Which numeric variable do you think is the most likely to increase the liklihood someone might release a reptile? Explain your answer.

*(note - you do not have to be right! Your explanation just has to be reasonable.)*

### QUESTION 4: Which variable do you think is the most likely to decrease the liklihood someone might release a reptile? Explain your answer.

*(note - you do not have to be right! Your explanation just has to be reasonable.)*

Now that you have chosen two numeric variables, you will run two separate logistic regressions and analyze the results. It’s okay if your results are not significant, or if they are different than what you expected from questions 3 and 4 - understanding the test is the point, not getting statistical significance or having a weirdly omniscient level of understanding of what makes a person dip out of snake ownership.

### QUESTION 5: Copy-paste your glm code for each of the variables you listed in the previous question.

*Note - you should have two glm codes with one x variable each, you do not need to run a multiple logistic regression.*

### QUESTION 6: Is there a statistically significant relationship between your x variables and the liklihood of reptile release?

*Hint: look at the p value for your x variable! And also… report your p values*

### QUESTION 7: Look at your answers for questions 3 & 4. Do your analyses support your answers? Use your slope estimates to answer your question.

*Remember that non-significant p values mean a slope could be 0!*

### QUESTION 8: Use the predict() and confusionMatrix() functions for each of your logistic regressions. Which of your x variables had the highest accuracy in prediction?

*You don’t have to give us your code, but do report your accuracy!*

### QUESTION 9: If someone asked you what they could do to stop people from releasing reptiles into the wild, what (according to your analyses) would you tell them was the most effective way of doing so?

*Psst if neither of your results were significant or good at predicting release, you can tell them that you need to do more analyses…*

The End! Upload your answers to D2L.