LT7: Categorical Data Analysis

Learning Outcomes

By the end of this workshop, you should be able to:

1. Run a chi-square goodness of fit test
2. Run a chi-square indpendence test
3. Understand how to run a logstic regression

Install required packages

```
install.packages("tidyverse")
install.packages("car")
install.packages("readr")
```

Load required packages

The required packages are the same as the installed packages. Write the code needed to load the required packages in the below R chunk.

```
library(car)
library(tidyverse)
library(readr)
```

Chi-square

In general, the chi-square (x2) test is a useful statistical test to look at differences with categorical variables (e.g., political preference, gender). We can use the x2 test in two similar, but subtly difference situations:

1. When estimating how similar an observed distribution is to an expected distribution (e.g., are the same proportion of people supporting the three political parties?). Here, we use a goodness of fit test.
2. When estimating whether two variables are independent (e.g., is the likelihood of getting aid from large companies vs. small companies the same for all three political parties?). Here we use an independent test.

Because of the relative simplicty of chi-square, we'll first compute x2 by hand, and then we will do it using R!

Chi-square goodness of fit test

If you have one categorical variable from a single population, and you'd like to determine whether the sample is consistent with a hypothesized distribution, you can use a x2 goodness of fit test!

Null hypothesis (H0): The data follow a specified distribution (typically a truly equal distribution)

Alternate hypothesis (Ha): The data don't follow the specified distribution (typically a non-equal distribution)

For example, imagine that there are 3 political parties (Democrats, Republicans, and Independent), and you take a poll from a sample of US citizens asking which category they support (i.e., their likely vote). You might wonder whether the people's likely votes are equally distributed between the 3 political parties (your H0).

The research question is:

Are US citizen's political party preferences equally distrubuted?

To answer this, lets take a random sample of US citizens and ask them which party they support. This data can be found in the votes dataframe. Write the code to load it below:

```
votes <- read_csv("C:/Users/tc560/Dropbox/Work/LSE/PB130/LT7/Workshop/votes.csv")
```

```
## Warning: Missing column names filled in: 'X1' [1]
```

```
## Parsed with column specification:
## cols(
##   X1 = col_character(),
##   x = col_double()
## )
```

```
votes
```

```
## # A tibble: 3 x 2
##   X1        x
##   <chr> <dbl>
## 1 dem     587
## 2 rep     552
## 3 ind     480
```

Incidently, given the simplicity of this data, we can also enter it directly in R pretty easily by creating a new R object and concanating the three cases:

```
votes <- c(dem = 587, rep = 552, ind = 480)
votes
```

```
## dem rep ind
## 587 552 480
```

This may came in useful if you have very simple data you wish to run analyses on as it saves alot of time writing it into excel and then transferring it over to R.

Anyway, I digress.

Calculate observed proportions

Returning to the analyses at hand, it can be helpful to reframe these observed frequencies as proportions of the total sample. To do that, we just divide each observation by the sample size (N). Lets first create the sample size object N

```
N <- sum(votes)
N
```

```
## [1] 1619
```

There are 1619 paritcipants in our sample. Lets now divide each party count by the sample size to get an indication of the proportion of voters for each political party in our sample.

```
votes_freq <- votes/N
votes_freq
```

```
##       dem       rep       ind
## 0.3625695 0.3409512 0.2964793
```

Determine expected frequencies

Now, we want to determine the expected frequencies for each political party if the spread of votes was distributed equally (i.e., the null hypothesis). Since we have 3 categories (Democrat, Republican, Independent), and our null hypothesis is that voters are equally distributed across the categories, our hypothesized proportion p of voters/party = 1/3 or 0.333333. Lets first create that proportion object in R and call it p:

```
p <- 1/3
p
```

```
## [1] 0.3333333
```

Then, lets multiply that proportion by the sample size to arrive at an expected proportion of voters for each party if the null hypothesis were true. We will call this R object e for expected:

```
e <- N * p
e
```

```
## [1] 539.6667
```

Since we have 3 categories (Democrat, Republican, Independent), and our null hypothesis is that voters are equally distributed across the categories, our hypothesized proportion p of voters/party = 1/3. Thus, our expected frequencies for each party should be Np = *1619* 13 = 539.6667.

Calculate x2

Recall from the lecture that the calcualtion for chi-square is just the sum of the observed counts for each cell minus the expected counts squared divided by the expected count. So lets do that calculation now by hand in R and save it as a new object called chisq_gf:

```
chisq_gf <- (587 - 539.67)^2/539.67 + (552-539.67)^2/539.67 + (480 - 539.67)^2/539.67
chisq_gf
```

```
## [1] 11.0302
```

Here we have a chi-square of 11.03. Is that large enough to be considered statistically significant? For this we need to know the degrees of freedom.

Calculate df and significance

Degrees of freedom in chi-square is k-1, where k is the number of categories. So in this case our df = 3-1 = 2.

We can compare a chi-square of 11.03 with a df of 2 to a table of critical values - like in the lecture. Or we can directly calcualte the p value in R using pchsq(). To do this, we just create a new R object called p_val that contains this calculation.

```
p_val <- pchisq(chisq_gf, df = 2, lower.tail = F)
p_val
```

```
## [1] 0.00402553
```

So, we calculated x2 (df=2) = 11.03, p = 0.004.

Calculating chi-square goodness-of-fit test with R

Now, the easy way to do this in one line of code with the chisq.test() function!

```r
gf_test <- chisq.test(votes)
gf_test
```

```
##
##  Chi-squared test for given probabilities
##
## data:  votes
## X-squared = 11.03, df = 2, p-value = 0.004025
```

This is the simple way to do chi-square, but I wanted to show you the process underpinning this output. We can see that the chi-sqaure is indeed large enough to be statistically significant (i.e., $p < .05$). The voters are not equally distributed across the political parties.

Writing up results

We can reject the null hypothesis that the voters are equally distributed across political parties, x2 (df=2) = 11.03, p = 0.004. Thus, the votes are significantly different than we would expect if an equal number of voters ahd voted for each party. From looking at the observed frequencies compared to those expected, it looks like fewer voters were supporting the Independent party (~30%), compared to the Democrats or Republicans (~35%).

Chi-square test of independence

If you have two categorical variables, and you'd like to determine whether the variables are independent (i.e., that there is no relationship between them), you can use a x2 test of inderpendence.

Null hypothesis (H0): The 2 categorical variables are independent (there is no relationship between the variables)

Alternate hypothesis (Ha): The 2 categorical variables are dependent (there is a relationship between the variables)

Calulating the independence test by hand

In this example, we have data about juvenile deaths in the UK (N=350), specifically about the time of death (time = 1, 2, 3; corresponding to 1990-91, 1992-93, 1994-95), and the cause of death (cause = maltreatment, or other). We want to know if there is any evidence that, in recent years, we have been seeing an increase in maltreatment cases, many of which result in death.

Our research question, then, is:

Is there a relationship between time and cause of death among juveniles in the UK (dependent)?

To answer this, lets create a table in R containing the data. This is quite straightforward with cell data like this, we just list the counts and the number of columns:

```r
## write the data out and list the number of columns
death_table <- matrix(c(26, 31, 45, 68, 80, 100), ncol=2)

# label the columns and rows
colnames(death_table) <- c('maltreat', 'other')
rownames(death_table) <- c('1', '2', '3')
```

For ease, we can convert this dataframe into a table and add sums to the margins for each category - which will helps us when calculating the expected values.

```r
# convert to a table
death_table <- as.table(death_table)
```

```r
# add margins to calculate the sums for each cell
addmargins(death_table)
```

```
##     maltreat other Sum
## 1         26    68  94
## 2         31    80 111
## 3         45   100 145
## Sum      102   248 350
```

Calculate expected values

For each cell, we can calculate the expected value by multiplying that cells row and column totals and dividing by our total sample size.

For instance, using the margins from above, the expected value for 1, maltreat = (94*102)/350=27.39.

The expected value for 2, maltreat = (111*102)/350=32.35.

And so on..

Once we calculate an expected value for each cell, we can then use the same x2 function using the observed and expected values to calculate our x2 statistic.

Try filling in a blank code with the expected values, and then calculate out x2!

```r
## write the data out and list the number of columns
expected <- matrix(c(27.39, 32.35, ??, ??, ??, ??), ncol=2)

# label the columns and rows
colnames(expected) <- c('maltreat', 'other')
rownames(expected) <- c('1', '2', '3')

# convert to a table
expected <- as.table(expected)
expected
death_table
```

I've outputted both expected and observed Tables here. Expected is top and observed is bottom. Lets use these values and the chi-square formula to calculate chi-square by hand.

Remember, the calcualtion for chi-square is:

X2 = sum(observed-expcected)^2/expected

So for each cell take the observed from the expected, square it, and then divide by the expected. Then, add them all up.

```r
chisq_in <- (26 - 27.39)^2/27.39 + (31-32.35)^2/32.35 + (?? - ??)^2/?? + (?? - ??)^2/?? + (80 - ??)^2/??
chisq_in
```

You should calculate that x2 = 0.43

Assess significance

The degrees of freedom in an chi-square test of independence is; (number of rows-1)*(number of columns-1)

Since we have 3 rows and 2 columns, our df = (3-1)*(2-1) = 2

And then use the pchisq() used above to determine our p-value/significance.

```r
pchisq(0.43, df=2, lower.tail = F)
```

```
## [1] 0.8065414
```

We fail to reject the null hypothesis that cause of death is independent of time (i.e., that there is no relationship between the variables), X2(df=2) = 0.43, p = 0.81, and thus we don't have any evidence that maltreatment deaths are increasing with time. The causes of deaths for juveniles appears to be pretty stable over time.

Calculating chi-square independence test with R

Like above, with R we can calculate all this in one line of code:

```r
chisq.test(death_table)
```

```
##
##  Pearson's Chi-squared test
##
## data:  death_table
## X-squared = 0.43075, df = 2, p-value = 0.8062
```

Easy!

Writing up results

We can accept the null hypothesis that the effect of maltreatment on juvenile death is independent of time, x2 (df=2) = 0.43, p = 0.81.

Logistic regression

Logistic regression, also called a logit model, is used to model categorical outcome variables with continuous predictors. In the logistic regression model the log odds of the outcome is modeled as a linear consequence of the predictor(s).

The data

We are going to go through a very simple example of logisitc regression. That is one categorical outcome and one continuous predictor. At the end, we will see that logisitc regression is just like normal regression and it can be exended to more than one predictor. But for now, its best to start with the simplest design.

A researcher is interested in how GPA (grade point average) effects admission into graduate school. The outcome variable, admit/don't admit, is a categorical variable.

I have generated some hypothetical data on this topic in the college dataframe. The dataframe has three variables:

1. admit - is our categorical college admission outcome variable (admitted = 1, not admitted = 0)
2. gpa - is our focal continuous predictor and is the mean GPA score for each student in our sample
3. pres - is our second predictor, which we will add later. This is the prestigue of the school each student attended and goes from 1 = low prestige to 4 = high prestige.

Before we get into the analysis, lets remind ourselves of the research question and hypotheses.

Research Question:

Is there a relationship between GPA and college admission in students from the US?

Null hypothesis: The relationship between GPA and college admission will be zero

Alternative hypothesis: There will be a positive relationship between GPA and college admission.

Lets load the college dataframe now or just import it from the LT7 folder:

```
college <- read_csv("C:/Users/tc560/Dropbox/Work/LSE/PB130/LT7/Workshop/college.csv")
```

```
## Parsed with column specification:
## cols(
##   admit = col_double(),
##   gpa = col_double(),
##   pres = col_double()
## )
```

```
college
```

```
## # A tibble: 400 x 3
##    admit   gpa  pres
##    <dbl> <dbl> <dbl>
##  1     0  3.61     3
##  2     1  3.67     3
##  3     1  4        1
##  4     1  3.19     4
##  5     0  2.93     4
##  6     1  3        2
##  7     1  2.98     1
##  8     0  3.08     2
##  9     1  3.39     3
## 10     0  3.92     2
## # ... with 390 more rows
```

Setting up the logistic regression model

To build the logistic regression model we will use glm() rather than lm(). This is becuase logistic regression is a special case of the linear model - one in which the outcome is estimated in terms of log(odds) rather than raw units. We need to do this becuase the outcome only goes from 0 to 1 and a linear model will make predicted values that fall outside of these boundarys without this transformation (see lecture). But otherwise, the coding is exactly the same as we have been working with to date.

For the simple logisitic regression model, lets create a new R object called college.model and tell R to run a logistic regression on the categorical outcome. The code of the glm function is similar to that of lm, except that we must pass the argument family = binomial in order to tell R to run a logistic regression rather than some other type of generalized linear model.

```
college.model <- glm(admit ~ 1 + gpa, data = college, family = "binomial")
summary(college.model)
```

```
##
## Call:
## glm(formula = admit ~ 1 + gpa, family = "binomial", data = college)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.1131  -0.8874  -0.7566   1.3305   1.9824
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -4.3576     1.0353  -4.209 2.57e-05 ***
## gpa           1.0511     0.2989   3.517 0.000437 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 499.98  on 399  degrees of freedom
## Residual deviance: 486.97  on 398  degrees of freedom
## AIC: 490.97
##
## Number of Fisher Scoring iterations: 4
```

In the output above, the first thing we see is the call, this is R reminding us what the model we ran was, what options we specified, etc. Next we see the deviance residuals, which are a measure of model fit. This part of output shows the distribution of the deviance residuals for individual cases used in the model - which is like the liklihood estiamtes we calcuated in the lecture. Below we discuss how to use summaries of the deviance statistic to assess model fit.

The next part of the output shows the coefficients, their standard errors, the z-statistic (sometimes called a Wald z-statistic), and the associated p-values. Both the intercept and gpa slope are statistically significant (i.e., $p < .05$). The logistic regression coefficients give the change in the log odds of the outcome for a one unit increase in the predictor variable.

So, for every one unit change in gpa, the log odds of admission (versus non-admission) increases by 1.05.

Below the table of coefficients are fit indices, including the null and deviance residuals and the AIC. Later we show an example of how you can use these values to help assess model fit. However, in short, deviance is analogous to the sum of squares calculations in linear regression and is a measure of the lack of fit to the data in a logistic regression model. The null deviance represents the difference between a model with only the intercept (which means "no predictors") and a saturated model (a model with a theoretically perfect fit). The goal is for the model deviance (noted as Residual deviance) to be lower; smaller values indicate better fit. In this respect, the null model provides a baseline upon which to compare predictor models.

Bootstrapping the estimate

As with linear regression, we can bootstrap the gpa estimate to arrive at a 95% confidence interval that makes no assumption about the distribution of the variables or sampling distribution of the estimate.

```
college.boot <- Boot(college.model, f=coef, R = 5000)
```

```
## Loading required namespace: boot
```

```
confint(college.boot, level = .95, type = "norm")
```

```
## Bootstrap normal confidence intervals
##
##                 2.5 %    97.5 %
## (Intercept) -6.4303138 -2.213068
## gpa          0.4346727  1.650000
```

The confidence interval for the gpa slope does not include zero and therefore this support the normal theory test. The relationship between gpa and college attendance is statistivcally significant.

Odds ratios

You can also exponentiate the coefficients and interpret them as odds-ratios. R will do this computation for you. To get the exponentiated coefficients, you tell R that you want to exponentiate (exp), and that the object you want to exponentiate is called coefficients and it is part of college.model (coef(college.model)). We can use the same logic to get odds ratios and their confidence intervals, by exponentiating the confidence

intervals from the bootstrapping procedure. To put it all in one table, we use cbind to bind the coefficients and confidence intervals column-wise.

```
## odds ratios only
exp(coef(college.model))
```

```
## (Intercept)         gpa
##  0.01280926  2.86082123
```

```
## odds ratios and 95% CI
exp(cbind(OR = coef(college.model), confint(college.boot)))
```

```
##                      OR       2.5 %     97.5 %
## (Intercept) 0.01280926 0.001411825 0.09889761
## gpa         2.86082123 1.588886261 5.38403434
```

Now we can say that for a one unit increase in gpa, the odds of being admitted to graduate school (versus not being admitted) increase by a factor of 2.86. Thats a big increase! This is far more interpretable than expressing this relationship in terms of log(odds).

Model fit

We may also wish to see measures of how well our model fits. The output produced by summary(college.model) included indices of fit (shown below the coefficients), including the null and deviance residuals and the AIC.

One measure of model fit is the significance of the overall model. This test asks whether the model with predictors fits significantly better than a model with just an intercept (i.e., the empty model).

The test statistic is the difference between the residual deviance for the model with predictors and the null model. The test statistic is distributed chi-squared with degrees of freedom equal to the differences in degrees of freedom between the current and the null model (i.e., the number of predictor variables in the model).

To find the difference in deviance for the two models (i.e., the test statistic) we can use the command:

```
with(college.model, null.deviance - deviance)
```

```
## [1] 13.0089
```

The degrees of freedom for the difference between the two models is equal to the number of predictor variables in the mode, and can be obtained using:

```
with(college.model, df.null - df.residual)
```

```
## [1] 1
```

Finally, the p-value can be obtained using:

```
with(college.model, pchisq(null.deviance - deviance, df.null - df.residual, lower.tail = F))
```

```
## [1] 0.0003100148
```

The chi-square of 13.01 with 1 degrees of freedom and an associated p-value of less than 0.001 tells us that our model as a whole fits significantly better than an empty model. This is sometimes called a likelihood ratio test (the deviance residual is -2log likelihood - just as we saw in the lecture). To see the model's log likelihood, we type:

```
logLik(college.model)
```

```
## 'log Lik.' -243.4838 (df=2)
```

Interestingly, we can also use the anova() function to see the same information:

```
anova(college.model, test="Chisq")
```

```
## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: admit
##
## Terms added sequentially (first to last)
##
##
##      Df Deviance Resid. Df Resid. Dev Pr(>Chi)
## NULL                   399     499.98
## gpa   1   13.009       398     486.97  0.00031 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The print out above shows the same overall reduction in deviance, from 499.98 to 486.97 on 1 degree of freedom. In anova, much like with each main effect we saw in LT5, the reduction in deviance is shown for each term (here we only have gpa), added sequentially first to last. Of note is the gpa term, which produced a significant reduction in deviance of 13.01 - just as we calculated above.

How its written

We used logistic regression to test whether the college admission could be predicted by gpa. The chi-square test statistic for the overall logistic regression model was statistically significant: x2(1) = 13.01, p < .05. Therefore, our model does fits significantly better than an empty intercept-only model. The logistic regression coefficient indicted that gpa was positively correlated with admission status (b = 1.05, SE = .30, z = 3.10, p < .05). With 5,000 resamples, the bootstrap 95% confidence interval associated with the regression coeffient was .44 to 1.64. The odds ratio associated with the regression coefficient of 2.86 indicated that a one unit increase in gpa enhances the chance of attending college by a factor of 2.86.

Adding an additional predictor

As we saw in the lecture, logistic regression is just an extention of the linear model. Specifically, it is a general linear model because the outcome is categorical and therefore we must transform the y-axis to log(odds) rather than raw units. But otherwise, the analyses are the same - our transformation linearlizes the relationship and permits tests using the straight line equation (i.e., y = a + bx + e)

And becuase logistic regression is equivalent to the linear model, we can also do fancy extentions with more than one predictor variable. Just like multiple regression or ANOVA for the linear model.

Lets go ahead and extend the simple logsitic regression model we just built to include an additional predictor - school prestige (lablled 'pres') - and call it 'multiple.college.model'. By doing so, we will have a multiple logistic regression model - one that includes two predictors.

```
multiple.college.model <- glm(admit ~ 1 + gpa + pres, data = college, family = "binomial")
summary(multiple.college.model)
```

```
##
## Call:
## glm(formula = admit ~ 1 + gpa + pres, family = "binomial", data = college)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.4599  -0.8974  -0.6657   1.1516   2.1781
```

```
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -2.8826     1.0916  -2.641 0.008273 **
## gpa           1.0270     0.3064   3.352 0.000801 ***
## pres         -0.5822     0.1263  -4.609 4.05e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 499.98  on 399  degrees of freedom
## Residual deviance: 463.93  on 397  degrees of freedom
## AIC: 469.93
##
## Number of Fisher Scoring iterations: 3
```

As we now know what all the estimates in this table mean, lets focus here on the slopes. The slope coefficients, their standard errors, the z-statistic (sometimes called a Wald z-statistic), and the associated p-values are listed for gpa and pres. Both the gpa and pres slopes are statistically significant (i.e., $p < .05$).

The logistic regression coefficients in this multiple model are interpreted in the same way as in multiple regression. That is, they give the change in the log odds of the outcome for a one unit increase in the predictor variable, holding the other predictor constant.

So holding pres constant, for every one unit change in gpa the log odds of admission (versus non-admission) increases by 1.03.

And holding gpa constant, for every one unit change in pres the log odds of admission (versus non-admission) decreases by 0.58.

And just like multiple regression, we can add as many predictors as we see fit to our logisitic regression model.

Pretty neat!

Like above, we can also call the model fit for this multiple model using anova(). Here the overal reduction in error is outputted after the last predictor and every previous reduction in fit with the addition of each predictor.

```
anova(multiple.college.model, test="Chisq")
```

```
## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: admit
##
## Terms added sequentially (first to last)
##
##
##      Df Deviance Resid. Df Resid. Dev  Pr(>Chi)
## NULL                   399     499.98
## gpa   1   13.009       398     486.97   0.00031 ***
## pres  1   23.034       397     463.93 1.591e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Here we can see that the reduction in error attributable to gpa is 13 and the addition of prestige reduces the error further by around 10 to 23.03. 23.03 is the overall model fit and is a chi-sqaure statistic on 2 df. We

can see that this is significant so the overall model is better than the empty model. We can reject the null hypothesis.

Activity

Okay, lets now give you a chance to set up and test a logistic regression. We are going to examine whether personality effect whether or not someone volunteers for psychological research.

The data for this task are from Cowles and Davis (1987). They were interstest to know whether students' extraversion and neuroticism influenced their tendency to volunteer for research projects.

The outcome variable - volunteering - is a categorical outcome with two levels. These are yes and no, with no coded 0 and yes coded 1.

The continuous predictor variables neuroticism and extraversion are scale measurements from the Eysenck personality inventory.

Before we delve into this topic, lets just clarify the research question and null hypothesis we are testing:

Research Question - Is there a relationship between personality and tendency to volunteer for psychology research among college students

Null Hypothesis - The relationship between neuroticism and tendency to volunteer for psychology research will be zero

Null hypothesis - The relationship between extraversion and tendency to volunteer for psychology research will be zero

Before we start, lets load the cowles dataframe to the R environment. Go to the LT7 folder, and then to the workshop folder, and find the "cowles.csv" file. Click on it aand then select "import dataset". In the new window that appears, click "update" and then when the dataframe shows, click import. If you want, you can try running the code below and it might do the same thing (if not put your hand up).

```
cowles <- read_csv("C:/Users/tc560/Dropbox/Work/LSE/PB130/LT7/Workshop/cowles.csv")
```

```
## Parsed with column specification:
## cols(
##   id = col_double(),
##   neuroticism = col_double(),
##   extraversion = col_double(),
##   volunteer = col_double()
## )
```

Task 1 - Build the multiple logistic regression model

Use the glm() function to build the multiple logistic regression model volunteering. Place the 'volunteer' variable as the outcome in this model and the 'neuroticism' and 'extraversion' variables as the predictors. Save this logisitic regression model in a new R object called cowles.model and call the summary.

```
cowles.model <- glm(?? ~ 1 + ?? + ??, data = ??, family = "binomial")
summary(??)
```

What is the effect of neuroticism on volunteering and how is it interpreted?

Is the effect of neuroticism on volunteering statisticall significant?

What is the effect of extraversion on volunteering and how is it interpreted?

Is the effect of extraversion on volunteering statisticall significant?

Task 2 - Bootstrap 95% confidence interval for the model estimates

Lets go ahead generate a bootstrap sampling distribution of the estimates and name it as a new R object; cowles.boot. Then use it to call the bootsrap 95% confidence interval using the confint() function.

```
cowles.boot <- Boot(??, f=coef, R = ??)
summary(??) ## add the cowles.model and stipulate how many resamples you want to run (typically 5,000)

confint(??, level = .95, type = "norm") # add the bootstrap model you just created to this code
```

What is the bootstrap 95% confidence interval for the effect of neuroticism on volunteering?

Do we accept or reject the null hypothesis?

What is the bootstrap 95% confidence interval for the effect of extraversion on volunteering?

Do we accept or reject the null hypothesis?

Task 3 - The odds ratio

Knowing the estimates in units of log(odds) is not overly informative, so it is also a good idea to also express effects in terms of the odds ratio. This is the increase/decrease in the odds of finding the outcome for every unit increase in the predictor. To do this, we simply exponentiate (exp) the slope coefficients, and tell R that the object you want to exponentiate is called coefficients and it is part of cowles.model (coef(cowles.model)). We can use the same logic to get odds ratios and their confidence intervals by exponentiating the confidence intervals from the bootstrapping procedure. To put it all in one table, we use cbind to bind the coefficients and confidence intervals column wise. Using the code above as a guide, fill in the missing code belof for the odds ratios and the confidence intervals.

```
exp(coef(??)

exp(cbind(OR = coef(??), confint(??)))
```

What is the effect of neuroticism on volunteering expressed in units of odds ratio and how is it interpreted?

What is the effect of neuroticism on volunteering expressed in units of odds ratio and how is it interpreted?

Task 4 - chi-sqaure model fit

Use anova() to calcualte the chi-sqaure for the overall model fit

```
anova(??, test="Chisq")
```

What is the chi-square and degrees of freedom for the overall model and is it statistically significant?