|  |
| --- |
| **Working in the infer Package Workflow for a Basic Regression** |
| **Obtaining 1000 Permuted Slope Statistics**null\_dist <- <NAME OF DATASET> %>% specify(response = <NAME OF RESPONSE VARIABLE>,  explanatory = <NAME OF EXPLANATORY VARIABLE>) %>%  hypothesize(null = “independence”) %>% generate(reps = 1000, type = "permute") %>% calculate(stat = "slope")***Note:*** You choose the number of reps. I recommend choosing at least 1000, to get a good idea of the shape of the bootstrap distribution – remember we need to verify it is approximately normal.  |
| **Plotting the Null Distribution**visualize(null\_distribution)***Note:*** You can add axis labels to this plot! All you need to do is connect the visualize() step to labs() using a **+** sign.  |
|  **Shading the p-value**visualize(null\_distribution)+  shade\_p\_value(obs\_stat = obs\_slope,  direction = “two-sided”)***Note:*** You can add axis labels to this plot! All you need to do is connect the visualize() step to labs() using a **+** sign.  |
| **Obtaining a p-value**get\_p\_value(null\_dist,  obs\_stat = obs\_slope, direction = “two-sided”) |
| **Parametric Methods for Obtaining a p-value for Basic Regression** |
| my\_model <- lm(<NAME OF RESPONSE VARIABLE> ~ <NAME OF EXPLANATORY VARIABLE>,  data = <NAME OF DATASET>)get\_regression\_table(my\_model,  conf.level = **0.95**)***Note:*** If you want a 90% confidence interval (or an alpha of 0.1), you change conf.level to 0.90 |
| **Working in the infer Package Workflow for a Multiple Regression** |
| **Obtaining the Sample (Observed) Regression Fit**obs\_fit <- <NAME OF DATASET> %>% specify(<RESPONSE> ~ <EXPLANATORY VARIABLE 1> **\*** <EXPLANATORY VARIABLE 2>) %>%  fit()***Note:*** Within specify(), your syntax looks identical to what you used to fit a model with the lm() function. You use a ~ to separate the response from the explanatory variables. ***Note:*** In the above code a **\*** separates the two explanatory variables. This would fit an interaction model! If you want to fit an additive model, you would need to use a **+** sign to separate the variables! |
| **Obtaining 1000 Permuted Fits**null\_dist <- <NAME OF DATASET> %>% specify(<RESPONSE> ~ <EXPLANATORY VARIABLE 1> **\*** <EXPLANATORY VARIABLE 2>) %>%  hypothesize(null = “independence”) %>% generate(reps = 1000, type = "permute") %>% fit()***Note:*** You choose the number of reps. I recommend choosing at least 1000, to get a good idea of the shape of the bootstrap distribution – remember we need to verify it is approximately normal.  |
| **Plotting the Null Distributions**visualize(data = null\_dist)***Note:*** This will create **multiple** histograms, one for each variable included in the multiple regression.  |
| **Shading the p-value**visualize(null\_dist)+  shade\_p\_value(obs\_stat = obs\_fit,  direction = “two-sided”) |
| **Obtaining a p-value**get\_p\_value(null\_dist,  obs\_stat = obs\_fit, direction = “two-sided”) |

|  |
| --- |
| **Parametric Methods for Obtaining a p-value for Multiple Regression** |
| my\_model <- lm(<RESPONSE> ~ <EXPLANATORY VARIABLE 1> **\*** <EXPLANATORY VARIABLE 2>,  data = <NAME OF DATASET>)anova(my\_model)***Note:*** In the above code a **\*** separates the two explanatory variables. This would fit an interaction model! If you want to fit an additive model, you would need to use a **+** sign to separate the variables! |
| **Evaluating Conditions of Residuals for Regression** |
| **Distribution of residuals (to evaluate normality condition)**broom::augment(my\_model) %>%  ggplot(mapping = aes(x = .resid)) + geom\_histogram() + labs(x = "Residual")***Note:*** In the above code my\_model represents the linear model that was fit, this can be either a simple **or** multiple linear regression! |
| **Residuals versus fitted values (to evaluate equal variance condition)**broom::augment(my\_model) %>%  ggplot(mapping = aes(y = .resid, x = `<NAME OF EXPLANATORY VARIABLE>`)) + geom\_point() +  geom\_hline(yintercept = 0, color = "red", linewidth = 3) + labs(x = "<NAME OF EXPLANATORY VARIABLE>",  y = "Residuals")***Note:*** In the above code my\_model represents the linear model that was fit, this can be either a simple **or** multiple linear regression! |