HW1 Submission

Justin Schulberg

5/28/2020

# Linear/Logistic Regression Analysis

In this script, I will employ a few facets of linear and logistic regression analysis to analyze the following three datasets:  
1. AirBnB – This dataset contains a variety of indicators related to housing factors to predict on the price of a house  
2. Direct Marketing – This dataset contains a variety of variables used to determine the Amount Spent on advertising  
3. Titanic – This dataset contains various factors of the individuals aboard the Titanic. I will use those factors to predict whether someone survived

## AirBnB

In the first third of this script, I will analyze the AirBnB dataset, looking at a variety of factors to predict the price of a housing unit. In particular, I will:

* Fit a multiple linear regression model using price as the response variable and all others as predictor variables
* Analyze the results, interpret the coefficients
* Predict the price for a “fake” dataset
* Identify outliers using Cook’s distance approach
* Perform Logarithimic Transformation to better fit the regression models to the dataset and determine which of the four possible log transformations (linear-linear, linear-log, log-linear, and log-log regression) is best

First, let’s take a look at our data.

## # A tibble: 854 x 10  
## room\_id survey\_id host\_id room\_type city reviews overall\_satisfa~  
## <dbl> <dbl> <dbl> <chr> <chr> <dbl> <dbl>  
## 1 1.58e7 1498 1.02e8 Shared r~ Ashe~ 0 0   
## 2 1.83e7 1498 1.26e8 Shared r~ Ashe~ 32 5   
## 3 1.81e7 1498 1.22e8 Shared r~ Ashe~ 4 4.5  
## 4 1.23e7 1498 7.47e5 Shared r~ Ashe~ 24 4.5  
## 5 1.57e5 1498 7.47e5 Shared r~ Ashe~ 152 4.5  
## 6 1.30e7 1498 7.47e5 Shared r~ Ashe~ 20 4.5  
## 7 6.77e6 1498 7.47e5 Shared r~ Ashe~ 52 4.5  
## 8 1.62e7 1498 4.86e7 Shared r~ Ashe~ 14 4.5  
## 9 9.56e6 1498 7.84e6 Entire h~ Ashe~ 3 5   
## 10 1.62e7 1498 1.00e8 Entire h~ Ashe~ 30 5   
## # ... with 844 more rows, and 3 more variables: accommodates <dbl>,  
## # bedrooms <dbl>, price <dbl>

What do the different variables mean? From a website I found [online](tomslee.net/category/airbnb-data), I found the following definitions for our data:

room\_id: A unique number identifying an Airbnb listing. The listing has a URL on the Airbnb web site of <http://airbnb.com/rooms/room_id>  
host\_id: Unique number identifying an Airbnb host. The host’s page has a URL on the Airbnb web site of <http://airbnb.com/users/show/host_id>  
room\_type: One of “Entire home/apt”, “Private room”, or “Shared room”  
city: The city or search area for which the survey is carried out.  
reviews: The number of reviews that a listing has received. Airbnb has said that 70% of visits end up with a review, so the number of reviews can be used to estimate the number of visits. Note that such an estimate will not be reliable for an individual listing (especially as reviews occasionally vanish from the site), but over a city as a whole it should be a useful metric of traffic.  
overall\_satisfaction: The average rating (out of five) that the listing has received from those visitors who left a review.  
accommodates: The number of guests a listing can accommodate.  
bedrooms: The number of bedrooms a listing offers.  
price: The price (in $US) for a night stay. In early surveys, there may be some values that were recorded by month.

Let’s change some of the data types and clean up our dataset.

airbnb\_cleaned <- airbnb\_data %>%  
 # ID columns should be strings, not numbers  
 mutate(room\_id = as.character(room\_id)) %>%  
 mutate(host\_id = as.character(host\_id)) %>%  
 mutate(survey\_id = as.character(survey\_id)) %>%  
 # It looks like room type should be a factor  
 mutate(room\_type = as.factor(room\_type))  
  
# Before we fit a regression model, we actually won't need any  
# of the ID columns or the city, which only has one value (Asheville),  
# so let's remove those  
airbnb\_vars <- airbnb\_cleaned %>%  
 select(-contains("id"), -city)

Now we’ll get set up to run our linear regression model. Before running a linear regression model, let’s set the seed to ensure randomization and reproducibility.

set.seed(123)  
  
# Regardless, let's trek on with a linear regression model  
airbnb\_linreg1 <- lm(price ~ ., data = airbnb\_vars)  
summary(airbnb\_linreg1)

##   
## Call:  
## lm(formula = price ~ ., data = airbnb\_vars)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -367.8 -49.2 3.2 38.6 4032.7   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -23.36172 21.88618 -1.067 0.28609   
## room\_typePrivate room -0.93115 13.21827 -0.070 0.94386   
## room\_typeShared room -76.66780 59.90939 -1.280 0.20099   
## reviews 0.01090 0.09982 0.109 0.91310   
## overall\_satisfaction -10.48160 3.47320 -3.018 0.00262 \*\*   
## accommodates 23.00721 5.23952 4.391 1.27e-05 \*\*\*  
## bedrooms 85.64533 11.45983 7.474 1.95e-13 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 167.1 on 847 degrees of freedom  
## Multiple R-squared: 0.3228, Adjusted R-squared: 0.318   
## F-statistic: 67.3 on 6 and 847 DF, p-value: < 2.2e-16

From this summary, we can see that the R-squared value is .3228, which frankly isn’t too great. We also notice that our room type and reviews variables aren’t too accurate either (low p-value). On the other hand, overall satisfaction, accomodates, and bedrooms are all really good predictors according to the model. We have to be careful though because we identified earlier that accomodates and bedrooms are highly correlated.

Looking at the coefficients in our summary, we also notice that there are two coefficients for Room type (one for private room and one for shared room). But there are three levels to room type and we’re missing “Entire home/apt”. Where’d it go? Well, it’s implicitly included in the analysis. R automatically created indicator variables for room type, and if private room and shared room are both equal to ‘0’, thus indicating that the type of a room is not one of those, we have entire home/apt. The coefficient for room\_type (Shared Room) is -76.7, which can be interpreted as such:

If a room is a shared room, with all else held equal, the price per night is actually -23.36 - 76.67 = -$100.03 cheaper per night.

The coefficient for bedrooms, 85.65, indicates that for every extra bedroom in the house and with all else held equal, the price increases by $85.65 per night.

Now let’s try to predict the price (nearest dollar) per night for a listing with the following factors:  
bedrooms = 1, accommodates = 2, reviews = 70, overall\_satisfaction = 4, and room\_type= ‘Private room’

airbnb\_new <- tibble(  
 bedrooms = 1,   
 accommodates = 2,   
 reviews = 70,   
 overall\_satisfaction = 4,   
 room\_type = "Private room"  
)  
  
(airbnb\_pred <- predict(airbnb\_linreg1, airbnb\_new))

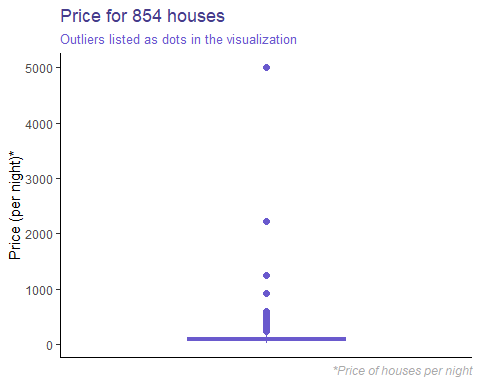
## 1   
## 66.20316

Thus, we predict that the private room would cost 66.20 per night. How does this line up to the rest of our data, where the mean is **126.617096**.

### Outliers

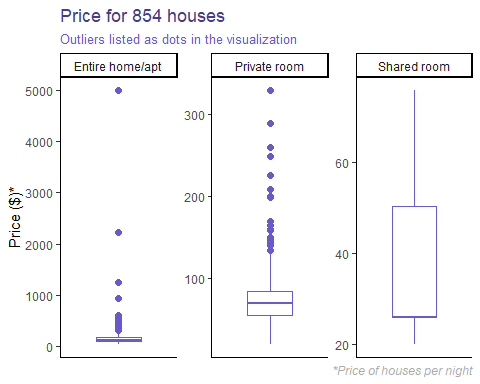
Before looking at outliers, let’s take a look at the boxplot of the price variable.

ggplot(airbnb\_vars, aes(y = airbnb\_vars$price)) +  
 geom\_boxplot(outlier.colour="slateblue3",  
 outlier.size=2,  
 color = "slateblue") +  
 theme\_classic() +  
 # Let's change the names of the axes and title  
 labs(title = paste("Price for", nrow(airbnb\_data), "houses", sep = " "),  
 subtitle = "Outliers listed as dots in the visualization",  
 caption = "\*Price of houses per night") +  
 ylab("Price (per night)\*") +  
 # Center the title and format the subtitle/caption  
 theme(plot.title = element\_text(hjust = 0, color = "slateblue4"),  
 plot.subtitle = element\_text(color = "slateblue", size = 10),  
 plot.caption = element\_text(hjust = 1, face = "italic", color = "dark gray"),  
 # remove the x axis labels because they don't mean much for us  
 axis.text.x = element\_blank()) +  
 # I thought the boxplot was too thick, so let's make it a little skinnier  
 scale\_x\_discrete()



We immediately notice that there are a number of outliers. I have a hunch that it has to do with room type, so let’s break this out further.

ggplot(airbnb\_vars, aes(y = airbnb\_vars$price)) +  
 geom\_boxplot(outlier.colour="slateblue3",  
 outlier.size=2,  
 color = "slateblue") +  
 # Use facet\_wrap to get three boxplots based on the room type  
 # We'll also free up our y axis so everything is easier to see  
 facet\_wrap(~ room\_type, scales = "free\_y") +  
 theme\_classic() +  
 # Let's change the names of the axes and title  
 labs(title = paste("Price for", nrow(airbnb\_data), "houses", sep = " "),  
 subtitle = "Outliers listed as dots in the visualization",  
 caption = "\*Price of houses per night") +  
 ylab("Price ($)\*") +  
 # Center the title and format the subtitle/caption  
 theme(plot.title = element\_text(hjust = 0, color = "slateblue4"),  
 plot.subtitle = element\_text(color = "slateblue", size = 10),  
 plot.caption = element\_text(hjust = 1, face = "italic", color = "dark gray"),  
 # remove the x axis labels because they don't mean much for us  
 axis.text.x = element\_blank()) +  
 # I thought the boxplot was too thick, so let's make it a little skinnier  
 scale\_x\_discrete()

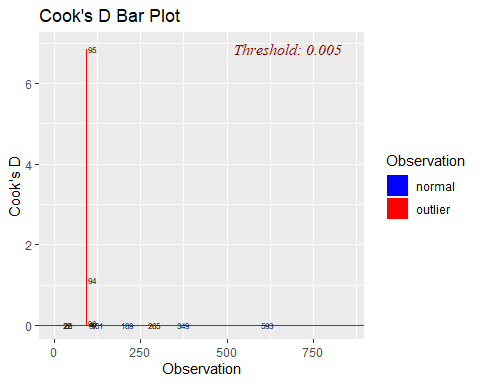


This seems to suggest that our outliers are in the entire home/apt and private room room types, which makes sense since these constitute 98% of our data. However, it looks like the outliers in entire home/apt are REALLY dragging the mean price per night out, whereas the private room outliers seem to be closer to the mean. Let’s use two methodologies: **Cook’s distance and Grubbs Test**.

### Cook’s Distance

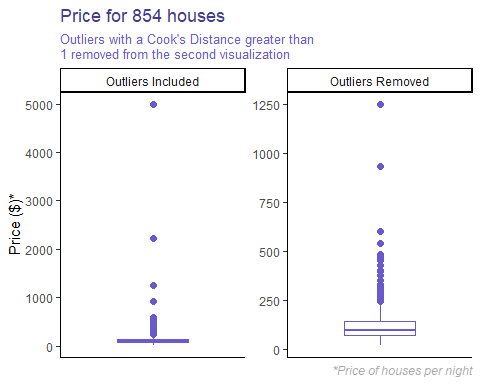
Cook’s distance is a method used to detect outliers that have a lot of influence (leverage) over a model. It does so using the following technique:  
1. Delete one observation, i, from the dataset at a time  
2. Refit our linear regression model on the remaining observations (n-1)  
3. Examine the degree to which the fitted values change when our ith observation is deleted from the model

# Use the olsrr package to plot our Cook's distance  
olsrr::ols\_plot\_cooksd\_bar(airbnb\_linreg1)



We can immediately see that a few points are really pulling the model outwards. Let’s remove these from our dataset and re-run the model on this low leverage dataset.

airbnb\_lowlev <- airbnb\_linreg1 %>%  
 # Calculate the Cook's Distance  
 cooks.distance() %>%  
 # Save it as a tibble  
 as\_tibble() %>%  
 # Rename it something meaningful  
 rename(cooks\_distance = value) %>%  
 # Bring it back into our dataset  
 bind\_cols(airbnb\_vars) %>%  
 # Rearrange our dataset by cook's distance  
 arrange(desc(cooks\_distance)) %>%  
 # Remove the two points with a Cook's Distance over 1  
 filter(cooks\_distance < 1) %>%  
 # Get rid of cooks distance since we don't need it anymore  
 select(-cooks\_distance)  
  
# How has this changed our box plot?  
# Start by creating a new variable in our two datasets that we can  
# eventually use to pivot on.  
airbnb\_combined <- airbnb\_lowlev %>%  
 mutate(outliers = "Outliers Removed")  
  
airbnb\_combined <- airbnb\_vars %>%  
 mutate(outliers = "Outliers Included") %>%  
 bind\_rows(airbnb\_combined)  
   
# Another boxplot viz  
ggplot(airbnb\_combined, aes(y = price)) +  
 geom\_boxplot(outlier.colour="slateblue3",  
 outlier.size=2,  
 color = "slateblue") +  
 # Create separate boxplots for our dataset with and without outliers  
 facet\_wrap(~ outliers, scales = "free\_y") +  
 theme\_classic() +  
 # Let's change the names of the axes and title  
 labs(title = paste("Price for", nrow(airbnb\_data), "houses", sep = " "),  
 subtitle = "Outliers with a Cook's Distance greater than\n1 removed from the second visualization",  
 caption = "\*Price of houses per night") +  
 ylab("Price ($)\*") +  
 # Center the title and format the subtitle/caption  
 theme(plot.title = element\_text(hjust = 0, color = "slateblue4"),  
 plot.subtitle = element\_text(color = "slateblue", size = 10),  
 plot.caption = element\_text(hjust = 1, face = "italic", color = "dark gray"),  
 # remove the x axis labels because they don't mean much for us  
 axis.text.x = element\_blank()) +  
 # I thought the boxplot was too thick, so let's make it a little skinnier  
 scale\_x\_discrete()



Now let’s re-run the model with these two points missing.

airbnb\_linreg2 <- lm(price ~ ., data = airbnb\_lowlev)  
summary(airbnb\_linreg2)

##   
## Call:  
## lm(formula = price ~ ., data = airbnb\_lowlev)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -190.95 -32.43 -7.09 20.35 876.26   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 75.01310 9.09152 8.251 6.01e-16 \*\*\*  
## room\_typePrivate room -32.28201 5.38034 -6.000 2.92e-09 \*\*\*  
## room\_typeShared room -91.69951 24.28958 -3.775 0.000171 \*\*\*  
## reviews -0.05915 0.04047 -1.462 0.144202   
## overall\_satisfaction -6.78957 1.41118 -4.811 1.78e-06 \*\*\*  
## accommodates 11.90698 2.14267 5.557 3.68e-08 \*\*\*  
## bedrooms 35.93177 4.87968 7.364 4.25e-13 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 67.73 on 845 degrees of freedom  
## Multiple R-squared: 0.4249, Adjusted R-squared: 0.4208   
## F-statistic: 104 on 6 and 845 DF, p-value: < 2.2e-16

Our r-squared value jumped to .42! That’s higher than before (.32), which is good. Let’s use Grubbs Test to see if there are any other outliers in our dataset that may be ruining the show.

(grubbs1 <- grubbs.test(airbnb\_lowlev$price))

##   
## Grubbs test for one outlier  
##   
## data: airbnb\_lowlev$price  
## G = 12.71406, U = 0.80983, p-value < 2.2e-16  
## alternative hypothesis: highest value 1250 is an outlier

grubbs1$alternative

## [1] "highest value 1250 is an outlier"

(p\_value <- grubbs1$p.value)

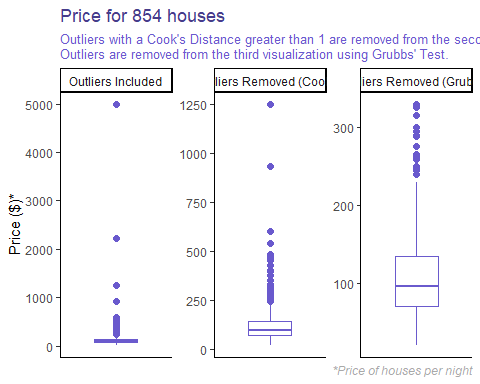
## [1] 0

Since the p-value is below .05, we can say with confidence that there is an outlier in the set. This indicates that the highest value 1250 is an outlier in our dataset. Let’s try removing it and see what happens.

airbnb\_nooutliers <- airbnb\_lowlev %>%  
 # Take out our maximum price  
 filter(price != max(price)) %>%  
 # Arrange our dataset on price  
 arrange(desc(price))  
  
# Let's check our Grubbs Test again and remove any remaining outliers  
# using a recursive function that runs Grubbs Test, checks if the p-value  
# is less than .05, removes the maximum value, and then re-runs Grubbs Test  
# on the remaining values. Once the p-value goes over .05, it'll save our  
# data to a new data frame and stop running  
remove\_outliers <- function(dataframe) {  
 # Save our input dataframe and column of interest as something standard   
 data\_grubbs <- dataframe  
 # Run Grubbs Test  
 grubbs <- grubbs.test(data\_grubbs$price)  
 # Check to see if the p-value is less than .05. If it is, take the highest  
 # value out of our dataset and re-run Grubbs Test.   
 if (grubbs$p.value < .05) {  
 # Re-save our dataset  
 data\_grubbs <- data\_grubbs %>%  
 # Take out our maximum price  
 filter(price != max(price)) %>%  
 # Arrange our dataset on price  
 arrange(desc(price))  
   
 # Re-run this function so it acts recursively  
 return(remove\_outliers(dataframe = data\_grubbs))  
 # If Grubbs Test p-value is greater than or equal to .05, save our final data  
 # frame and exit.  
 } else {  
 airbnb\_nooutliers <<- data\_grubbs  
 cat("Done running Grubbs Test! We successfully removed",   
 nrow(airbnb\_lowlev) - nrow(airbnb\_nooutliers),  
 "outliers.")  
 }  
}  
  
# Now that we have written our function to remove outliers using Grubbs.Test,  
# let's run it on our dataset.  
remove\_outliers(dataframe = airbnb\_nooutliers)

## Done running Grubbs Test! We successfully removed 20 outliers.

# Let's set ourselves up to visualize all of our results. Start by creating a  
# new variable in our two datasets that we can eventually use to pivot on.  
airbnb\_combined2 <- airbnb\_lowlev %>%  
 mutate(outliers = "Outliers Removed (Cook's)")  
  
airbnb\_combined3 <- airbnb\_nooutliers %>%  
 mutate(outliers = "Outliers Removed (Grubbs)") %>%  
 bind\_rows(airbnb\_combined2)  
  
airbnb\_combined3 <- airbnb\_vars %>%  
 mutate(outliers = "Outliers Included") %>%  
 bind\_rows(airbnb\_combined3)  
  
# Another boxplot viz  
ggplot(airbnb\_combined3, aes(y = price)) +  
 geom\_boxplot(outlier.colour="slateblue3",  
 outlier.size=2,  
 color = "slateblue") +  
 # Create separate boxplots for our dataset with and without outliers  
 facet\_wrap(~ outliers, scales = "free\_y") +  
 theme\_classic() +  
 # Let's change the names of the axes and title  
 labs(title = paste("Price for", nrow(airbnb\_data), "houses", sep = " "),  
 subtitle = "Outliers with a Cook's Distance greater than 1 are removed from the second visualization.\nOutliers are removed from the third visualization using Grubbs' Test.",  
 caption = "\*Price of houses per night") +  
 ylab("Price ($)\*") +  
 # Center the title and format the subtitle/caption  
 theme(plot.title = element\_text(hjust = 0, color = "slateblue4"),  
 plot.subtitle = element\_text(color = "slateblue", size = 10),  
 plot.caption = element\_text(hjust = 1, face = "italic", color = "dark gray"),  
 # remove the x axis labels because they don't mean much for us  
 axis.text.x = element\_blank()) +  
 # I thought the boxplot was too thick, so let's make it a little skinnier  
 scale\_x\_discrete()



Now let’s re-run the model with the dataset using Grubbs Test.

airbnb\_linreg3 <- lm(price ~ ., data = airbnb\_nooutliers)  
summary(airbnb\_linreg3)

##   
## Call:  
## lm(formula = price ~ ., data = airbnb\_nooutliers)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -131.64 -25.37 -7.34 17.06 234.73   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 98.41235 6.41958 15.330 < 2e-16 \*\*\*  
## room\_typePrivate room -37.24727 3.70765 -10.046 < 2e-16 \*\*\*  
## room\_typeShared room -86.25907 16.60430 -5.195 2.58e-07 \*\*\*  
## reviews -0.06560 0.02774 -2.365 0.018258 \*   
## overall\_satisfaction -3.97711 0.98586 -4.034 5.99e-05 \*\*\*  
## accommodates 5.69203 1.52073 3.743 0.000194 \*\*\*  
## bedrooms 21.71928 3.43583 6.321 4.24e-10 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 46.29 on 825 degrees of freedom  
## Multiple R-squared: 0.3811, Adjusted R-squared: 0.3766   
## F-statistic: 84.66 on 6 and 825 DF, p-value: < 2.2e-16

Interesting. The R-squared value dropped to .38. One interesting thing to note, is that the reviews explanatory variable has become significant by removing outliers using Grubbs Test. Overall, though, I’d stick to just using Cook’s Distance to remove high leverage points.

### Log Transformation

Next, we’ll revert back to our dataset pre-Grubbs and use a variety of logarithmic transformations, which should help normalize the dataset. We’ll target our energy on price and overall\_satisfaction.

**Linear-linear Model**

airbnb\_linlin <- lm(price ~ overall\_satisfaction, data = airbnb\_lowlev)  
summary(airbnb\_linlin)

##   
## Call:  
## lm(formula = price ~ overall\_satisfaction, data = airbnb\_lowlev)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -129.49 -45.82 -20.82 23.18 1092.51   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 157.489 7.759 20.298 < 2e-16 \*\*\*  
## overall\_satisfaction -9.334 1.710 -5.457 6.35e-08 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 87.53 on 850 degrees of freedom  
## Multiple R-squared: 0.03385, Adjusted R-squared: 0.03271   
## F-statistic: 29.78 on 1 and 850 DF, p-value: 6.345e-08

# Let's store our results so we can visualize them later.  
log\_results <- tibble(transformation = "Linear-Linear",   
 r\_squared = summary(airbnb\_linlin)$r.squared,  
 p\_value = summary(airbnb\_linlin)$coefficients[2, 4])

**Linear-Log Model**  
We’ll first test to see how transforming the explanatory variable, overall satisfaction affects the result. Note that we’ll have to add 1 to the variable before transformation since overall\_satisfaction has a range of 0-5 and log(0) equates to negative infinity, raising quite a number of problems.

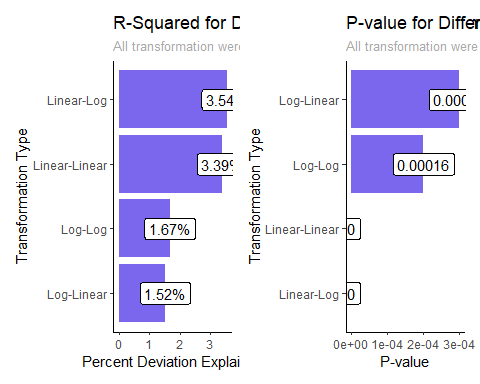
airbnb\_linlog <- lm(price ~ log(overall\_satisfaction + 1), data = airbnb\_lowlev)  
  
# Bind the new results in so we have them for later.  
log\_results <- bind\_rows(log\_results,  
 tibble(transformation = "Linear-Log",  
 r\_squared = summary(airbnb\_linlog)$r.squared,  
 p\_value = summary(airbnb\_linlog)$coefficients[2, 4]))

**Log-Linear Model**  
Next we’ll test to see how transforming the response variable, price, affects the result.

airbnb\_loglin <- lm(log(price) ~ overall\_satisfaction, data = airbnb\_lowlev)  
  
# Bind the new results in so we have them for later.  
log\_results <- bind\_rows(log\_results,  
 tibble(transformation = "Log-Linear",  
 r\_squared = summary(airbnb\_loglin)$r.squared,  
 p\_value = summary(airbnb\_loglin)$coefficients[2, 4]))

**Log-Log Model** Lastly, we’ll test to see how transforming both variables, price and overall\_satisfaction, affects the result. Note that we’ll have to add 1 to the variable before transformation since overall\_satisfaction has a range of 0-5 and log(0) equates to negative infinity, raising quite a number of problems.

airbnb\_loglog <- lm(log(price) ~ log(overall\_satisfaction + 1), data = airbnb\_lowlev)  
  
# Bind the new results in so we have them for later.  
log\_results <- bind\_rows(log\_results,  
 tibble(transformation = "Log-Log",  
 r\_squared = summary(airbnb\_loglog)$r.squared,  
 p\_value = summary(airbnb\_loglog)$coefficients[2, 4]))  
  
# Viz Time  
viz\_rsquared <- ggplot(log\_results,  
 # order by importance  
 aes(x = reorder(transformation, r\_squared), y = round(100\*r\_squared, 4), group = 1), label = log\_results$r\_squared) +  
 # Let's make it a column graph and change the color  
 geom\_col(fill = "slateblue2") +  
 # Add the rounded text labels in for r-squared so it's easier to read  
 geom\_label(label = paste(100\*round(log\_results$r\_squared, 4), "%", sep = "")) +  
 # Change the theme to classic  
 theme\_classic() +  
 # Let's change the names of the axes and title  
 xlab("Transformation Type") +  
 ylab("Percent Deviation Explained") +  
 labs(title = "R-Squared for Different Logarithmic Transformations",  
 subtitle = "All transformation were performed using the log() function.") +  
 # format our title and subtitle  
 theme(plot.title = element\_text(hjust = 0, color = "black"),  
 plot.subtitle = element\_text(color = "dark gray", size = 10)) +  
 # flip the axes and fix the axis  
 coord\_flip()  
  
viz\_pvalue <- ggplot(log\_results,  
 # order by importance  
 aes(x = reorder(transformation, p\_value), y = round(p\_value, 4), group = 1), label = log\_results$p\_value) +  
 # Let's make it a column graph and change the color  
 geom\_col(fill = "slateblue2") +  
 # Add the rounded text labels in for r-squared so it's easier to read  
 geom\_label(label = round(log\_results$p\_value, 5)) +  
 # Change the theme to classic  
 theme\_classic() +  
 # Let's change the names of the axes and title  
 xlab("Transformation Type") +  
 ylab("P-value") +  
 labs(title = "P-value for Different Logarithmic Transformations",  
 subtitle = "All transformation were performed using the log() function.") +  
 # format our title and subtitle  
 theme(plot.title = element\_text(hjust = 0, color = "black"),  
 plot.subtitle = element\_text(color = "dark gray", size = 10)) +  
 # flip the axes and fix the axis  
 coord\_flip()  
  
# Plot each using the patchwork library  
viz\_rsquared + viz\_pvalue



Based on these visualizations, it’s evidently clear that the none of the models are effective. The best transformation, the Linear-Log Model, only has an R-squared of 3.5, which means that 96.5% of the variation in the relationship between price and overall\_satisfaction is unexplained by the model. This makes sense, since I would not expect a large correlation between the price of an Airbnb rental and the ultimate satisfaction with the experience, since experience is usually independent of price.

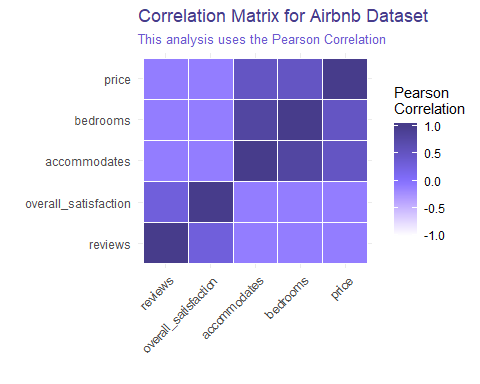
### Correlation

Let’s now look at a correlation matrix, so we have a better grasp of how the variables in our dataset are interacting.

airbnb\_cor <- airbnb\_vars %>%  
 # First remove room type, which is our only non numeric field left  
 select(-room\_type) %>%  
 # compute the correlation table  
 cor() %>%  
 # Round the results  
 round(1) %>%  
 print()

## reviews overall\_satisfaction accommodates bedrooms price  
## reviews 1.0 0.3 -0.1 -0.1 -0.1  
## overall\_satisfaction 0.3 1.0 -0.1 -0.1 -0.1  
## accommodates -0.1 -0.1 1.0 0.8 0.5  
## bedrooms -0.1 -0.1 0.8 1.0 0.5  
## price -0.1 -0.1 0.5 0.5 1.0

# Now let's look at a heat map  
airbnb\_cor %>%  
 # Start by pivoting the correlation table to a tidy format  
 reshape2::melt() %>%  
 ggplot(aes(x = Var1, y = Var2, fill = value)) +  
 # Visualize as tiles  
 geom\_tile(color = "white") +  
 # Change our scale to match the slateblue theme and extend from -1 to 1  
 scale\_fill\_gradient2(low = "white", # color of lowest point  
 high = "slateblue4", # color of highest point  
 mid = "slateblue1", # color of midpoint  
 midpoint = 0, # definition of midpoint  
 limit = c(-1, 1), # definition of range  
 name = "Pearson\nCorrelation" # name of legend  
 ) +  
 # Change the theme  
 theme\_minimal() +  
 labs(title = "Correlation Matrix for Airbnb Dataset",  
 subtitle = "This analysis uses the Pearson Correlation") +  
 ylab("") +  
 xlab("") +  
 # Center the title and format the subtitle/caption  
 theme(plot.title = element\_text(hjust = 0, color = "slateblue4"),  
 plot.subtitle = element\_text(color = "slateblue", size = 10),  
 # Edit the axis text  
 axis.text.x = element\_text(angle = 45, vjust = 1, hjust = 1))



From this we can see that there’s actually quite a bit of correlation between our data points. For example, there exists really high correlation between # of bedrooms and accomodates, which makes sense. With more time, I would recommend taking out highly correlated variables or including interaction terms.

# Direct Marketing

In the second third of this script, I will analyze the marketing dataset, looking at a variety of indicator factors to predict the amount spent on advertising. In particular, I will:

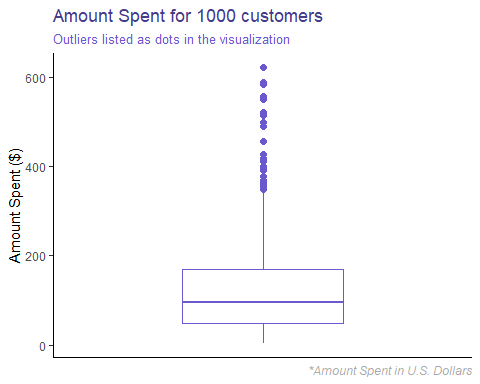
* Fit a multiple linear regression model using AmountSpent as the response variable and all other indicator variables as predictors
* Analyze the results, interpret the coefficients
* Predict the price for a “fake” dataset
* Identify outliers using Cook’s distance approach

First, let’s take a look at our data.

## # A tibble: 1,000 x 10  
## Age Gender OwnHome Married Location Salary Children History Catalogs  
## <fct> <fct> <fct> <fct> <fct> <int> <int> <fct> <int>  
## 1 Old Female Own Single Far 47500 0 High 6  
## 2 Midd~ Male Rent Single Close 63600 0 High 6  
## 3 Young Female Rent Single Close 13500 0 Low 18  
## 4 Midd~ Male Own Married Close 85600 1 High 18  
## 5 Midd~ Female Own Single Close 68400 0 High 12  
## 6 Young Male Own Married Close 30400 0 Low 6  
## 7 Midd~ Female Rent Single Close 48100 0 Medium 12  
## 8 Midd~ Male Own Single Close 68400 0 High 18  
## 9 Midd~ Female Own Married Close 51900 3 Low 6  
## 10 Old Male Own Married Far 80700 0 None 18  
## # ... with 990 more rows, and 1 more variable: AmountSpent <dbl>

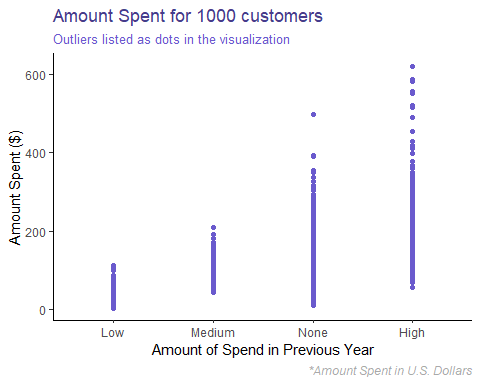
Before jumping into regression, let’s take a look at the boxplot of the amount spent variable.

ggplot(marketing\_data, aes(y = marketing\_data$AmountSpent)) +  
 geom\_boxplot(outlier.colour="slateblue3",  
 outlier.size=2,  
 color = "slateblue") +  
 theme\_classic() +  
 # Let's change the names of the axes and title  
 labs(title = paste("Amount Spent for", nrow(marketing\_data), "customers", sep = " "),  
 subtitle = "Outliers listed as dots in the visualization",  
 caption = "\*Amount Spent in U.S. Dollars") +  
 ylab("Amount Spent ($)") +  
 # Center the title and format the subtitle/caption  
 theme(plot.title = element\_text(hjust = 0, color = "slateblue4"),  
 plot.subtitle = element\_text(color = "slateblue", size = 10),  
 plot.caption = element\_text(hjust = 1, face = "italic", color = "dark gray"),  
 # remove the x axis labels because they don't mean much for us  
 axis.text.x = element\_blank()) +  
 # I thought the boxplot was too thick, so let's make it a little skinnier  
 scale\_x\_discrete()



We immediately notice that there are a number of outliers. I have a hunch that it has to do with amount spent in previous year, so let’s break this out further.

ggplot(marketing\_data, aes(y = AmountSpent, x = reorder(History, AmountSpent))) +  
 geom\_point(color = "slateblue") +  
 theme\_classic() +  
 # Let's change the names of the axes and title  
 labs(title = paste("Amount Spent for", nrow(marketing\_data), "customers", sep = " "),  
 subtitle = "Outliers listed as dots in the visualization",  
 caption = "\*Amount Spent in U.S. Dollars") +  
 ylab("Amount Spent ($)") +  
 xlab("Amount of Spend in Previous Year") +  
 # Center the title and format the subtitle/caption  
 theme(plot.title = element\_text(hjust = 0, color = "slateblue4"),  
 plot.subtitle = element\_text(color = "slateblue", size = 10),  
 plot.caption = element\_text(hjust = 1, face = "italic", color = "dark gray"))



#### One-hot Encoding

First we’ll start by running a regression model with AmountSpent as our response variable and History and Salary as our explanatory variables. Before we run a linear regression model on the data, let’s create indicator variables also known as “One-hot Encoding” on the History column. Our base case will be when there is no recent history on the customer’s purchases.

marketing\_encoded <- marketing\_data %>%  
 # Only bring in the variables of interest  
 select(AmountSpent, Salary, History) %>%  
 # One-hot Encoding  
 mutate(LowHistory = if\_else(History == "Low", 1, 0)) %>%  
 mutate(MediumHistory = if\_else(History == "Medium", 1, 0)) %>%  
 mutate(HighHistory = if\_else(History == "High", 1, 0)) %>%  
 # We'll also create three interaction variables between the history indicators and  
 # the salary of the customer.  
 mutate(LowSalary = LowHistory \* Salary) %>%  
 mutate(MediumSalary = MediumHistory \* Salary) %>%  
 mutate(HighSalary = HighHistory \* Salary) %>%  
 # Since the History variable is now redundant since we one-hot encoded it, let's  
 # get rid of our original variable. In situations where each of our indicators is  
 # '0', we know that there is no spending history for the customer.  
 select(-History)

Now that we have our variables of interest, we’ll fit our first linear regression model.

marketing\_linreg1 <- lm(AmountSpent ~ ., data = marketing\_encoded)  
summary(marketing\_linreg1)

##   
## Call:  
## lm(formula = AmountSpent ~ ., data = marketing\_encoded)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -214.33 -25.47 -6.46 20.64 352.50   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 1.9622199 6.3880253 0.307 0.758777   
## Salary 0.0023641 0.0001071 22.083 < 2e-16 \*\*\*  
## LowHistory 25.4466733 8.9203292 2.853 0.004426 \*\*   
## MediumHistory 79.2984388 12.8982169 6.148 1.14e-09 \*\*\*  
## HighHistory 72.6735221 15.2270169 4.773 2.09e-06 \*\*\*  
## LowSalary -0.0021069 0.0001890 -11.150 < 2e-16 \*\*\*  
## MediumSalary -0.0021153 0.0002182 -9.693 < 2e-16 \*\*\*  
## HighSalary -0.0006408 0.0001926 -3.328 0.000908 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 55.79 on 992 degrees of freedom  
## Multiple R-squared: 0.6654, Adjusted R-squared: 0.6631   
## F-statistic: 281.9 on 7 and 992 DF, p-value: < 2.2e-16

Based on this model, the R-squared value is .67, which isn’t too bad. The model we created explains about 2/3 of the variation in the data.

**What do the coefficients mean?** It looks like the interaction between salary and history of spend has an extremely small effect on amount spent. This would be surprisy, but individual salaries tend to be extremely high (in the tens of thousands). Given a large salary, the effect should still be significant.

The history variables mean quite a bit more, however. The LowHistory variable has a coefficient of 25.45, meaning that, all else held constant, if someone is categorized as having a low spend in the previous year, they will spend 1.96 + 25.45 = $27.41 in the current year.

The MediumHistory variable has a coefficient of 79.30, meaning that, all else held constant, if someone is categorized as having a medium spend in the previous year, they will spend 1.96 + 79.30 = $81.26 in the current year.

The HighHistory variable has a coefficient of 72.67, meaning that, all else held constant, if someone is categorized as having a high spend in the previous year, they will spend 1.96 + 72.67 = $74.63 in the current year.

What if someone has no history of spending in our dataset? Then all three History variables 0 out and we’re left with our intercept. Namely, that someone with no spend in the previous year will spend $1.96 in the current year.

*If salary were 10,000, the amount spent by history type would be as follows:*  
LowHistory Scenario: 1.9622 + 10000x.0023641 + 25.4467 - 10000x.0021069 = *$29.98* MediumHistory Scenario: 1.9622 + 10000x.0023641 + 79.2984 - 10000x0.0021153 = *$83.75* HighHistory Scenario: 1.9622 + 10000x.0023641 + 72.6735 - 10000x0.0006408 = *$91.87*

## Titanic

In the last third of this script, I will analyze the titanic dataset, looking at a variety of indicator factors related to individuals who were aboard the Titanic to predict whether or not they surved (1 = Survived, 0 = Did not Survive). In particular, I will:

* Convert the survived variable to a 0, 1 scale
* Perform a logistic regression model on the dataset, using ‘Survived’ as the response
* Analyze and interpret the model results
* Determine the probability of survival based on gender

First, let’s take a look at our data using read\_tsv, which works on tab-separated values, like the text file I got off this [site](http://math.ucdenver.edu/RTutorial/).

## Parsed with column specification:  
## cols(  
## Name = col\_character(),  
## PClass = col\_character(),  
## Age = col\_double(),  
## Sex = col\_character(),  
## Survived = col\_double()  
## )

## Warning: 4 parsing failures.  
## row col expected actual file  
## 1314 NA 5 columns 1 columns 'C:/Users/jschulberg/Documents/EdX/Data Analytics for Business/Homework/Airbnb/Data/titanic.txt'  
## 1315 Age a double Age 'C:/Users/jschulberg/Documents/EdX/Data Analytics for Business/Homework/Airbnb/Data/titanic.txt'  
## 1315 Survived a double Survived 'C:/Users/jschulberg/Documents/EdX/Data Analytics for Business/Homework/Airbnb/Data/titanic.txt'  
## 2629 NA 5 columns 1 columns 'C:/Users/jschulberg/Documents/EdX/Data Analytics for Business/Homework/Airbnb/Data/titanic.txt'

## # A tibble: 2,629 x 5  
## Name PClass Age Sex Survived  
## <chr> <chr> <dbl> <chr> <dbl>  
## 1 Allen, Miss Elisabeth Walton 1st 29 female 1  
## 2 Allison, Miss Helen Loraine 1st 2 female 0  
## 3 Allison, Mr Hudson Joshua Creighton 1st 30 male 0  
## 4 Allison, Mrs Hudson JC (Bessie Waldo Daniels) 1st 25 female 0  
## 5 Allison, Master Hudson Trevor 1st 0.92 male 1  
## 6 Anderson, Mr Harry 1st 47 male 1  
## 7 Andrews, Miss Kornelia Theodosia 1st 63 female 1  
## 8 Andrews, Mr Thomas, jr 1st 39 male 0  
## 9 Appleton, Mrs Edward Dale (Charlotte Lamson) 1st 58 female 1  
## 10 Artagaveytia, Mr Ramon 1st 71 male 0  
## # ... with 2,619 more rows

Here is a description of all the variables:  
Name - name of the individual  
pclass Ticket class 1 = 1st, 2 = 2nd, 3 = 3rd  
Age Age in years  
sex gender  
survival Survival 0 = No, 1 = Yes

# What percentage of each column is null?  
sapply(titanic\_data, function(x) paste(100\*round(sum(is.na(x))/2629, 3), "%", sep = ""))

## Name PClass Age Sex Survived   
## "0%" "0.1%" "42.5%" "0.1%" "0.1%"

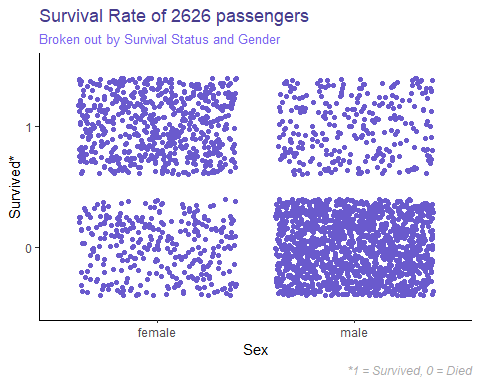
The fact that almost half of our age data is null is concerning. We’ll either have to take these out or impute the values.

I noticed that one of the rows has a weird value in it, where PClass = PClass and the other values are NA, so let’s remove it. We’ll also do some other data prep, like converting to factors and rounding out the ages.

titanic\_prepped <- titanic\_data %>%   
 filter(!(PClass == "PClass")) %>%  
 # Make Sex, PClass and Survived factors  
 mutate(Sex = as.factor(Sex),  
 PClass = as.factor(PClass),  
 Survived = as.factor(Survived)) %>%  
 # Round our age values  
 mutate(Age = round(Age))

Now we’ll look at the relationship between gender and survival rate. First, let’s take a look at the data.

ggplot(aes(x = Sex, y = Survived), data = titanic\_prepped) +   
 geom\_jitter(colour = "slateblue") +  
 theme\_classic() +  
 # Let's change the names of the axes and title  
 labs(title = paste("Survival Rate of", nrow(titanic\_prepped), "passengers", sep = " "),  
 subtitle = "Broken out by Survival Status and Gender",  
 caption = "\*1 = Survived, 0 = Died") +  
 ylab("Survived\*") +  
 # Center the title and format the subtitle/caption  
 theme(plot.title = element\_text(hjust = 0, color = "slateblue4"),  
 plot.subtitle = element\_text(color = "slateblue2", size = 10),  
 plot.caption = element\_text(hjust = 1, face = "italic", color = "dark gray"))



From this, we can clearly see that more females tended to survive than die, and more males tended to die than survive. Additionally, of those who survived, more were females and of those who died, more were male.

100\*round(prop.table(table(titanic\_prepped$Sex)), 3)

##   
## female male   
## 35.2 64.8

Before we jump to conclusions, just note that 65% of our passengers are male, but even with genders, it’s clear to see a pattern in survival rate.

### Logistic Regression

Now, we’ll run a logistic regression model on our dataset, using Sex as the explanatory variable and Survived as the response variable.

# Start by setting the seed to ensure randomization and reproducibility  
set.seed(123)  
  
# Let's run a logistic regression on our dataset, with Survived as the response   
# variable and Sex as the explanatory variable  
titanic\_logmod <- glm(Survived ~ Sex, data = titanic\_prepped, family = "binomial")  
# let's see how we did  
summary(titanic\_logmod)

##   
## Call:  
## glm(formula = Survived ~ Sex, family = "binomial", data = titanic\_prepped)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.4823 -0.6042 -0.6042 0.9005 1.8924   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 0.69315 0.06979 9.932 <2e-16 \*\*\*  
## Sexmale -2.30118 0.09538 -24.128 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 3376.1 on 2625 degrees of freedom  
## Residual deviance: 2711.1 on 2624 degrees of freedom  
## AIC: 2715.1  
##   
## Number of Fisher Scoring iterations: 4

From this model, we can see that the intercept is .69. This represents the log-odds of a female surviving on the Titanic. The results of logistic regression can be tough to understand, but here’s my take:  
Female survival rate is the value when  
*log(p/(1-p)) = B0*  
To find p, the probability that a female survives, we solve for p and assume a log-base of e. Thus:  
*p = exp(B0)/(1+exp(B0))*

(female\_survival <- exp(titanic\_logmod$coefficients[[1]])/(1 + exp(titanic\_logmod$coefficients[[1]])))

## [1] 0.6666667

We can apply the same logic for males, this time adding the two coefficients together before exponentiating. The math looks a little messy, but we get the right results. This will, in effect, be  
*p = exp(B0 + B1)/(1+exp(B0 + B1))*

(male\_survival <- exp(titanic\_logmod$coefficients[[1]] + titanic\_logmod$coefficients[[2]])/(1 + exp(titanic\_logmod$coefficients[[1]] + titanic\_logmod$coefficients[[2]])))

## [1] 0.1668625

Thus, the probability that a female survives is 66.7% and the probability that a male survives is 16.7%. This makes sense because, at least in the movie, they tended to put women and children on life boats. If we brought in age too, I’d expect that most of the males who survived were young (children).

# Let's save our results so we can visualize them  
survival\_rates <- tibble(gender = c("Male", "Female"),  
 rate = c(male\_survival, female\_survival))  
# Viz Time  
ggplot(survival\_rates, aes(x = gender, y = 100\*rate, group = 1), label = rate) +  
 # Let's make it a column graph and change the color  
 geom\_col(fill = "slateblue2") +  
 # Add the rounded text labels in so it's easier to read  
 geom\_label(label = paste(100\*round(survival\_rates$rate, 3), "%", sep = "")) +  
 # Change the theme to classic  
 theme\_classic() +  
 # Force the axes to be 0 to 100  
 ylim(0, 100) +   
 # Let's change the names of the axes and title  
 xlab("Gender") +  
 ylab("Probability of Survival (%)") +  
 labs(title = "Probability of Surviving the Fatal Titanic",  
 subtitle = "Probabilistic results, broken out by gender, are\ncalculated using a logistic regression model") +  
 # format our title and subtitle  
 theme(plot.title = element\_text(hjust = 0, color = "slateblue4"),  
 plot.subtitle = element\_text(color = "slateblue1", size = 10)) +  
 # flip the axes  
 coord\_flip()

