Visualization Examples

SDS 291

4/20/2020

Contents

Visualizing Distributions & Frequencies

You likely want to offer your reader or poster viewer some descriptive statistics or sense of the distribution/frequency of your variables of interest.

```
library(Stat2Data)
data("Titanic")
newTitanic <- Titanic %>% filter(PClass != "*") %>% mutate(Survived2 = as.factor(if_else(Survived ==
    1, "Yes", if_else(Survived == 0, "No", NA_character_))))
```

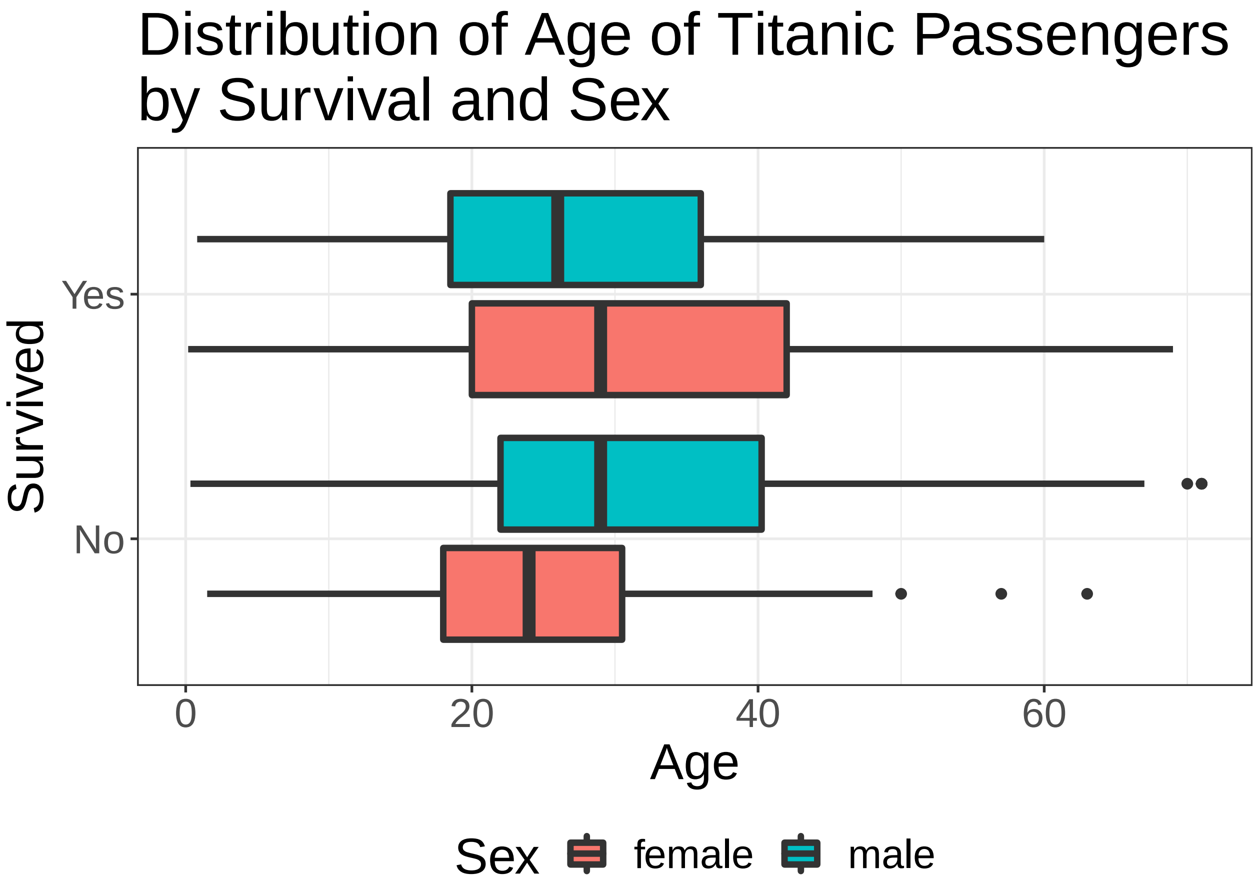
Binary Response Variable, Quantitative Explanatory Variable

Boxplot

You might want to illustrate the same kind of boxplot we've seen for logistic regression *by* some other variable – probably a variable you plan as an interaction hypothesis.

```
Survival_Age_Box <- newTitanic %>% ggplot(aes(y = Age, x = Survived2,
    fill = Sex)) + # Making Thicker Lines for the Boxplot
geom_boxplot(position = position_dodge(0.9), lwd = 1.2) + coord_flip() +
    theme_bw() + labs(y = "Age", x = "Survived", title = "Distribution of Age of Titanic Passengers \nb
    # Making the font big so it's easy to see on a poster
theme(legend.position = "bottom", text = element_text(size = 20))

Survival_Age_Box
```

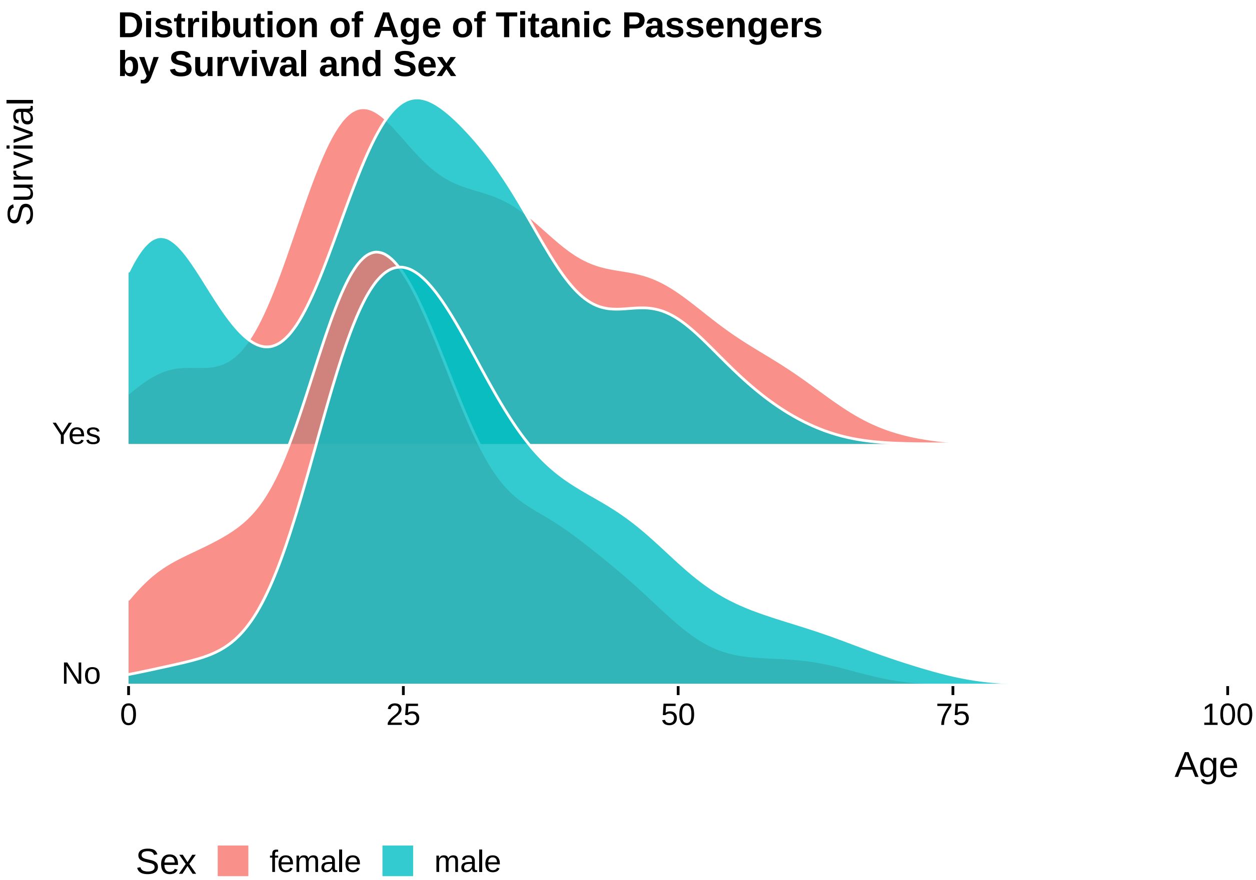
```
## Warning: Removed 556 rows containing non-finite values (stat_boxplot).
```



Distributions

You could also use a ridge plot from **ggridges** package (you might need to install it) to show the same pattern as a smoothed distribution, like a density plot, that shows not just the median and quantiles like a boxplot does.

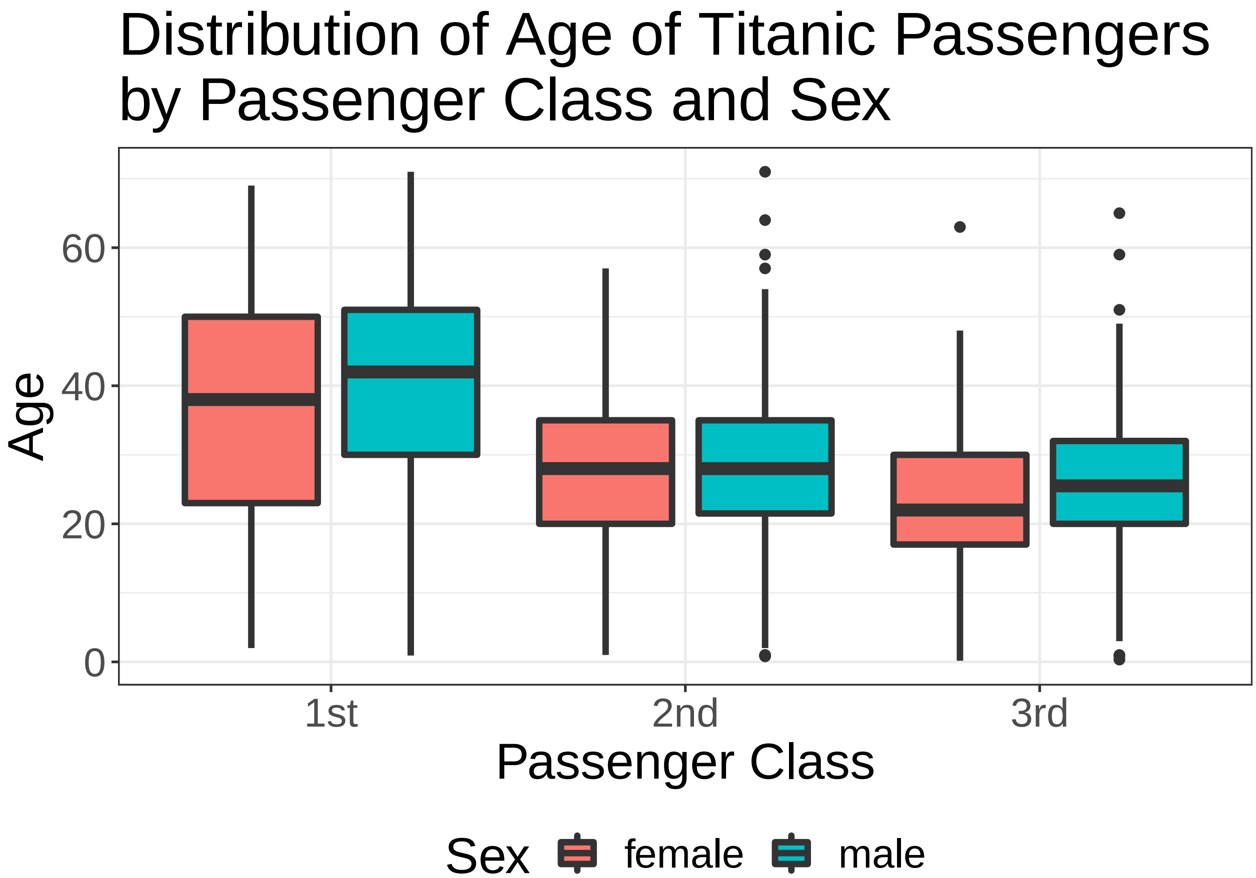
```
library(ggridges)
Survival_Age_Ridges <- newTitanic %>% ggplot(aes(y = Survived2)) +
    geom_density_ridges(aes(x = Age, fill = Sex), alpha = 0.8, color = "white",
        from = 0, to = 100) + labs(x = "Age", y = "Survival", title = "Distribution of Age of Titanic Pa
    scale_y_discrete(expand = c(0.01, 0)) + scale_x_continuous(expand = c(0.01,
    0)) + theme_ridges(grid = FALSE) + theme(legend.position = "bottom")

Survival_Age_Ridges
```



Quantitative response variable and categorical/binary explanatory variable

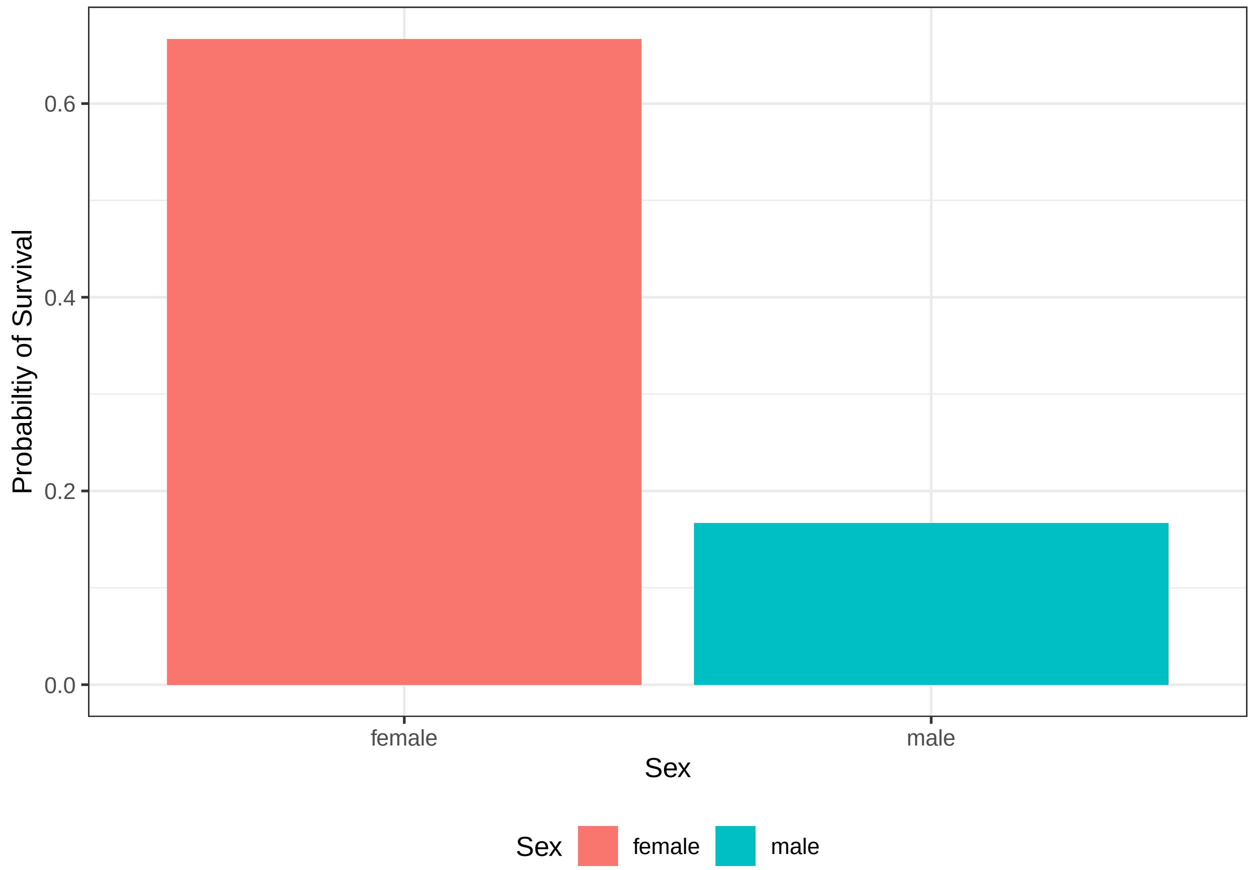
Remember that you could also use a boxplot for if you have a quantitative response variable (let's say `age` just to keep the Titanic data going). It's like the one above, but without flipping the y and x axis.

```
PClass_Age_Box <- newTitanic %>% ggplot(aes(y = Age, x = PClass, fill = Sex)) +
    geom_boxplot(position = position_dodge(0.9), lwd = 1.2) + theme_bw() +
    labs(y = "Age", x = "Passenger Class", title = "Distribution of Age of Titanic Passengers \nby Pass
    theme(legend.position = "bottom", text = element_text(size = 20))

PClass_Age_Box
```



Binary Response and Binary Explanatory Variables

```
# Here the explanatory variable is Sex and the response variable
# is Survived (yes/no) saving the plot as binary_by_binary,
# working from the Whickham data set
binary_by_binary <- Titanic %>% # by smoking status
group_by(Sex) %>% # count how many are survived
count(Survived) %>% # calculate the count as a proportion: (proprtion Survival of
# males; survival for females)
mutate(pi = n/sum(n)) %>% # only keep the survival outcomes (dead is just 1-pi)
filter(Survived == 1) %>% # starting the plot of the probability of survival by sex
ggplot(aes(y = pi, x = Sex, fill = Sex)) + # Just naming the y-xis so it's clear which outcome we're pl
# (alive or dead)
ylab("Probabiltiy of Survival") + # plot the proportion/probability for Male and Female
geom_bar(stat = "identity") + theme_bw() + theme(legend.position = "bottom")

binary_by_binary
```



Visualizing Regression Models

```
library(tidyverse)
library(Stat2Data)
data("Titanic")
```

Binary Repsonse Variables / Logistic Regression Models

We're using the Titanic data to estimate the simple logistic regression of differences in survival by sex (Model 1), and then a multiple logistic regression model adjusted for age (Model 2).

```
model_1 <- glm(Survived ~ SexCode, family = binomial, data = Titanic)
model_2 <- glm(Survived ~ SexCode + Age, family = binomial, data = Titanic)
```

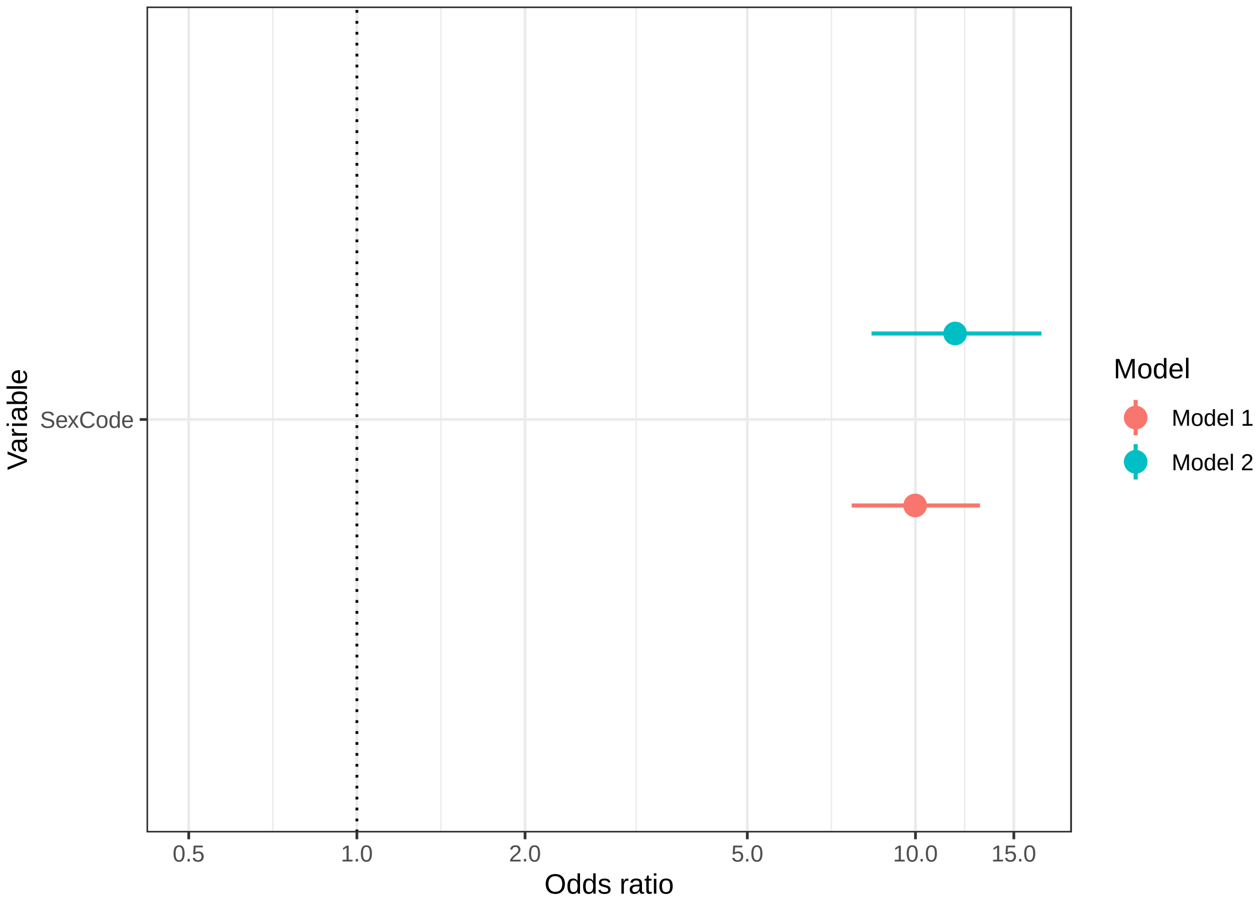
Then we're going to use the `broom` package to exponentiate the coefficients (i.e., calculating the OR), the confidence interval, and a new variable to indicate which model it was from – all saved into a dataset. Once for each model, and then we're going to combine them together into a new dataframe called oddsratios. And then we're going to keep only the ORs and CIs for SexCode variable, since that's the only one we want to illustrate – this would likely be your main explanatory variable from your main hypothesis that you want to depict across multiple models.

```
library(broom)
preds_1 <- tidy(model_1, conf.int = TRUE, exponentiate = TRUE) %>%
    mutate(Model = "Model 1")
preds_2 <- tidy(model_2, conf.int = TRUE, exponentiate = TRUE) %>%
    mutate(Model = "Model 2")
```

```
oddsratios <- bind_rows(preds_1, preds_2)
oddsratios <- oddsratios %>% filter(term %in% c("SexCode"))
```

Plotting the Odds Ratios

```
# Saving this as OR_plot, and working from the odds ratio dataset
OR_Plot<- oddsratios %>%
# Defining the Y as your estimated OR, x as the name of the OR
# (here, it's for SexCode) and coloring the Models differently
  ggplot(aes(y = estimate, x = term, colour = Model)) +
  # making the OR a point, with the CI as a line through the OR
        geom_pointrange(aes(ymin = conf.low, ymax = conf.high),
  # dodged/offset horizontally to make it w
                       position = position_dodge(width = 0.5),
                       size = .75) +
  # putting in a dotted line at the OR null = 1
        geom_hline(yintercept = 1.0, linetype = "dotted", size = .5) +
  # Making the axis scale be on the log scale
        scale_y_log10(breaks = c(0,0.5, 1.0, 2,5,10, 15)) +
  # Labeling the x and y axes
        labs(y = "Odds ratio", x = "Variable") +
   # Flipping the y and x axes (similar to what we had to do with the boxplots)
        coord_flip(ylim = c(0.5, 16)) +
  # Making the background white instead of grey
        theme_bw()
OR_Plot
```




```
ggsave("OR_plot.jpg")
```

```
## Saving 7 x 5 in image
```

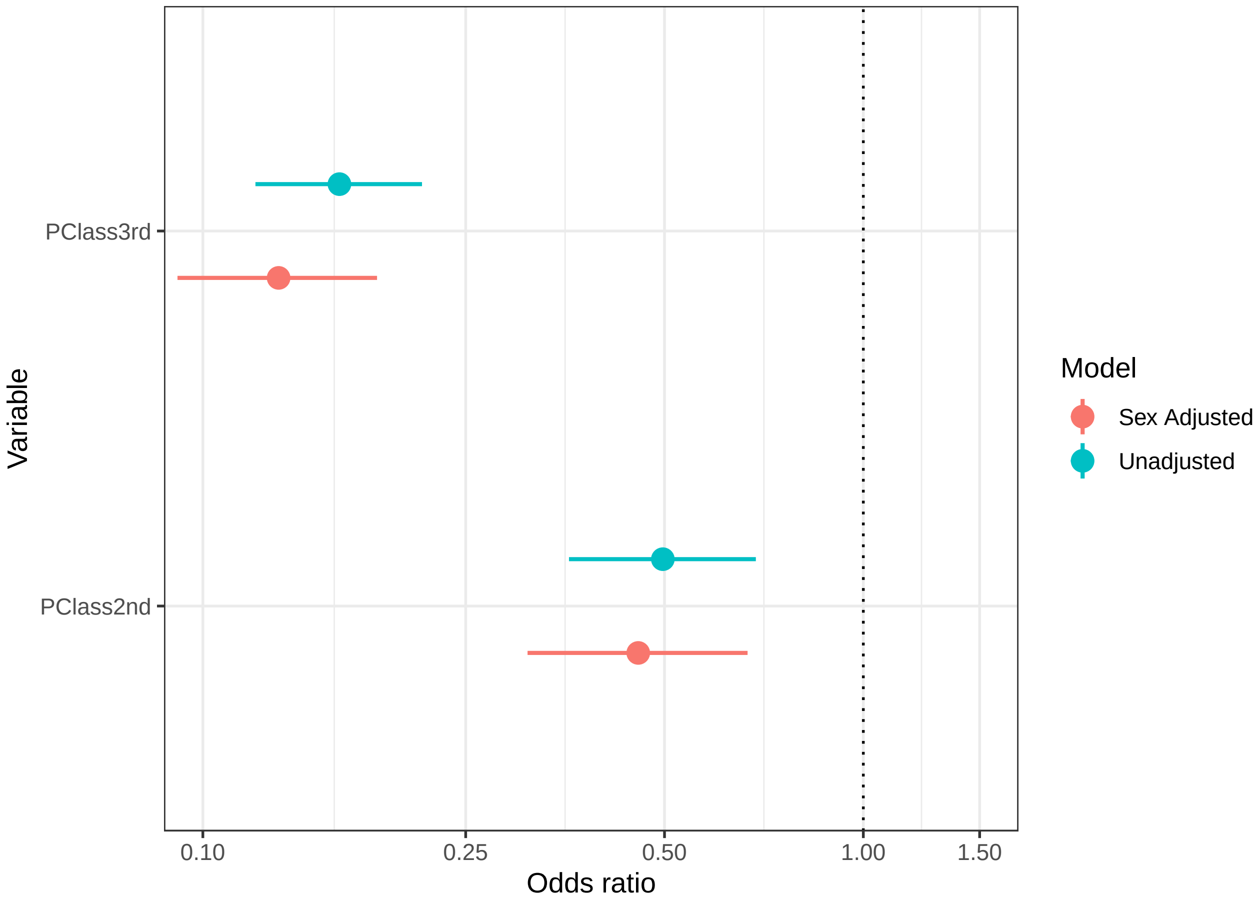
For more insight and discussion of why you might want to depict the odds ratio on the log scale, read a brief description here. [note: this refers to "risk" which is similar to odds, and also "relative odds" which is a synonmym for "odds ratio"]. The piece a letter to the editor of the American Journal of Epidemiology, which, as a journal, expects that any visual of ORs to be on the log scale. This letter to the editor, from some former and current editors, explains why.

Plotting ORs for Categorical Explanatory Variable

Now we're going to do a similar process but for a categorical explanatory variable rather than binary: PClass as a simple logistic regression model and then adjusted for Sex.

```
newTitanic <- Titanic %>% filter(PClass != "*")
PClass_1 <- glm(Survived ~ PClass, family = binomial, data = newTitanic)
PClass_preds_1 <- tidy(PClass_1, conf.int = TRUE, exponentiate = TRUE) %>%
    mutate(Model = "Unadjusted")

PClass_2 <- glm(Survived ~ PClass + SexCode, family = binomial, data = newTitanic)
PClass_preds_2 <- tidy(PClass_2, conf.int = TRUE, exponentiate = TRUE) %>%
    mutate(Model = "Sex Adjusted")

PClass_OR <- bind_rows(PClass_preds_1, PClass_preds_2)
PClass_OR <- PClass_OR %>% filter(term %in% c("PClass2nd", "PClass3rd"))
```

```
# Elements like pointrange and position_dodge only work when the
# outcome is mapped to y, need to go through with OR set as y then
# flip at the end
PClass_OR_plot <- PClass_OR %>% ggplot(aes(y = estimate, x = term,
    colour = Model)) + geom_pointrange(aes(ymin = conf.low, ymax = conf.high),
    position = position_dodge(width = 0.5), size = 0.75) + geom_hline(yintercept = 1,
    linetype = "dotted", size = 0.5) + scale_y_log10(breaks = c(0,
    0.1, 0.25, 0.5, 1, 1.5)) + labs(y = "Odds ratio", x = "Variable") +
    coord_flip(ylim = c(0.1, 1.5)) + theme_bw()

PClass_OR_plot
```




```
# This saves the figure, which is helpful for bringing into a
# poster or document.
ggsave("PClass_OR_plot.jpg")
```

```
## Saving 7 x 5 in image
```

Predicted Probabilities

Maybe instead of odds ratios you want predicted probabilities - they're more intuitive, etc. To do that, we're going to take slightly different approach and use the `predict()` function like we have in homeworks to get the predicted probability. Remember that the predict function requires a "new" dataset with sample values for whom we want the predicted probabilities to be calculated. We're going to use the same model above: Survival as a function of PClass, adjusted for Sex.

Calculate the probabilities

First, we will generate a new dataset with the values of 1st, 2nd, and 3rd Class to apply to the PClass variable. We repeat it twice so that there is a separate probability for men and women (this is similar to what we did above with mean age, but there's no mean sex, so illustrating both male and female probabilities makes more sense than "holding sex constant at the mean"). These are saved in a newdataset called "newdat_Titanic"

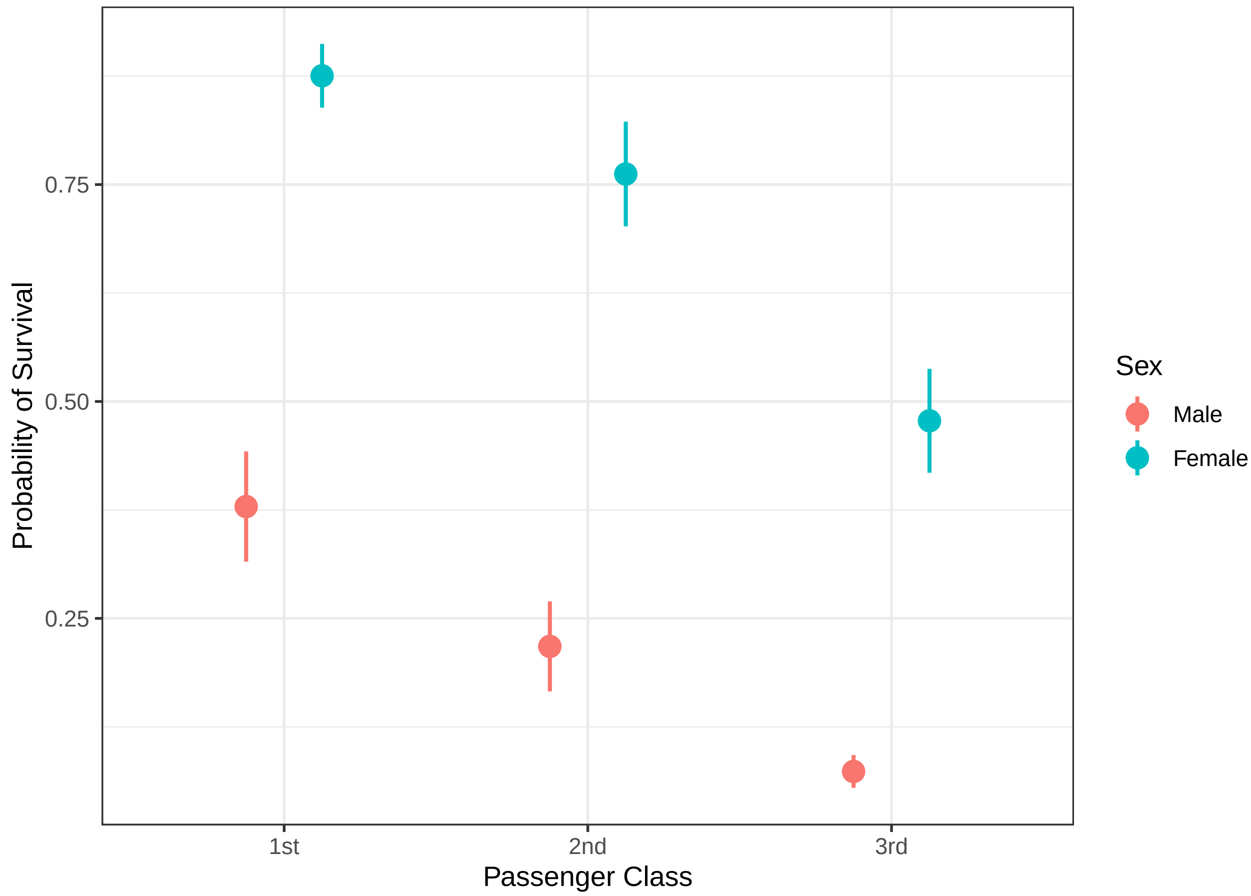
Then we predict the probability (type="response") of survival for men and women in each passenger class from the regression coefficients stored in PClass_2. We also ask for the standard error (se.fit=TRUE) so that we can calculate a confidence interval.

Lastly, we put those both together – the sample data and the estimated probability and standard error – in a dataset called `allpred_Titanic`.

```
newdat_Titanic <- data.frame(PClass = c("1st", "2nd", "3rd", "1st",
    "2nd", "3rd"), SexCode = c(0:1))
pred_Titanic <- as.data.frame(predict(PClass_2, newdat_Titanic, se.fit = TRUE,
    type = "response"))
allpred_Titanic <- bind_cols(newdat_Titanic, pred_Titanic)
```

Plot the probabilities

```
# Elements like pointrange and position_dodge only work when the outcome
#   is mapped to y, need to go through with OR set as y then flip at the
#   end
PClass_Prob_Plot <- allpred_Titanic %>%
  ggplot(aes(y = fit, x = PClass, colour = as.factor(SexCode))) +
        geom_pointrange(aes(ymin = fit-(1.96*se.fit), ymax = fit+(1.96*se.fit)),
                        position = position_dodge(width = 0.5),
                        size = .75) +
      # Legend label
        scale_color_hue(name="Sex",
                      # Defining the colors by your variable categories
                     breaks=c("0", "1"),
                     # Making longer, sensible variable labels for the legend
                     labels=c("Male",
                       "Female")) +
       # geom_hline(yintercept = 1.0, linetype = "dotted", size = .5) +
        #scale_y_log10(breaks = c(0,0.1,0.25,0.5, 1.0, 1.5)) +
        labs(y = "Probability of Survival", x = "Passenger Class") +
        #coord_flip(ylim = c(0.1, 1.5)) +
        theme_bw()
PClass_Prob_Plot
```




```
ggsave("PClass_Prob_Plot.jpg")
```

```
## Saving 7 x 5 in image
```

Binary Explanatory and Quantitative Response

Similar to what we did above with the predicted probability, if you have a quantiative response variable, we can do a similar process to estimate the predicted value ($\hat{y}$) of the response variable for binary explanatory variables. (If you have quantitative explanatory variables, you could use the example code from the travel time example and show a linear slope for that quantitative variable).

Here, the data are an extension of our original PorschePrice example – now with 2 car types: Porsche or Jaguars – used to predict the price of a used car. I am also including another binary variable – how old the used car is – to mimic a binary*binary interaction that some groups are having.

First I'm creating those variables, and estimating a model of $y = \beta_0 + \beta_1 Type + \beta_2 Age + \beta_3 Type * Age + \beta_4 Mileage + \epsilon$ Then I'm calculating the mean number of miles to predict the estimated price of the car for by type and age of the average number of miles. You'll want / need to do this for your confounding variables.

Then we're doing a very similar process as above (see that section for details) with the predict function. Here we're able to calculate the confidence interval around that prediction directly.

```
data("PorscheJaguar")
PJ <- PorscheJaguar %>% mutate(age_le5 = as.factor(if_else(Age < 5,
    "< 5yo", if_else(Age >= 5, "5+yo", NA_character_))))

m0 <- lm(Price ~ Car * age_le5 + Mileage, data = PJ)
summary(m0)
```

```
##
## Call:
## lm(formula = Price ~ Car * age_le5 + Mileage, data = PJ)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -21.3237  -5.3639  -0.6174   5.3603  18.8844
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)             54.75720    2.85205  19.199  < 2e-16 ***
## CarPorsche              15.60613    3.22845   4.834 1.12e-05 ***
## age_le55+yo             -8.43185    3.71450  -2.270   0.0271 *
## Mileage                 -0.50197    0.06816  -7.365 9.53e-10 ***
## CarPorsche:age_le55+yo   2.62758    4.65649   0.564   0.5749
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.866 on 55 degrees of freedom
## Multiple R-squared:  0.7877, Adjusted R-squared:  0.7723
## F-statistic: 51.03 on 4 and 55 DF,  p-value: < 2.2e-16
```

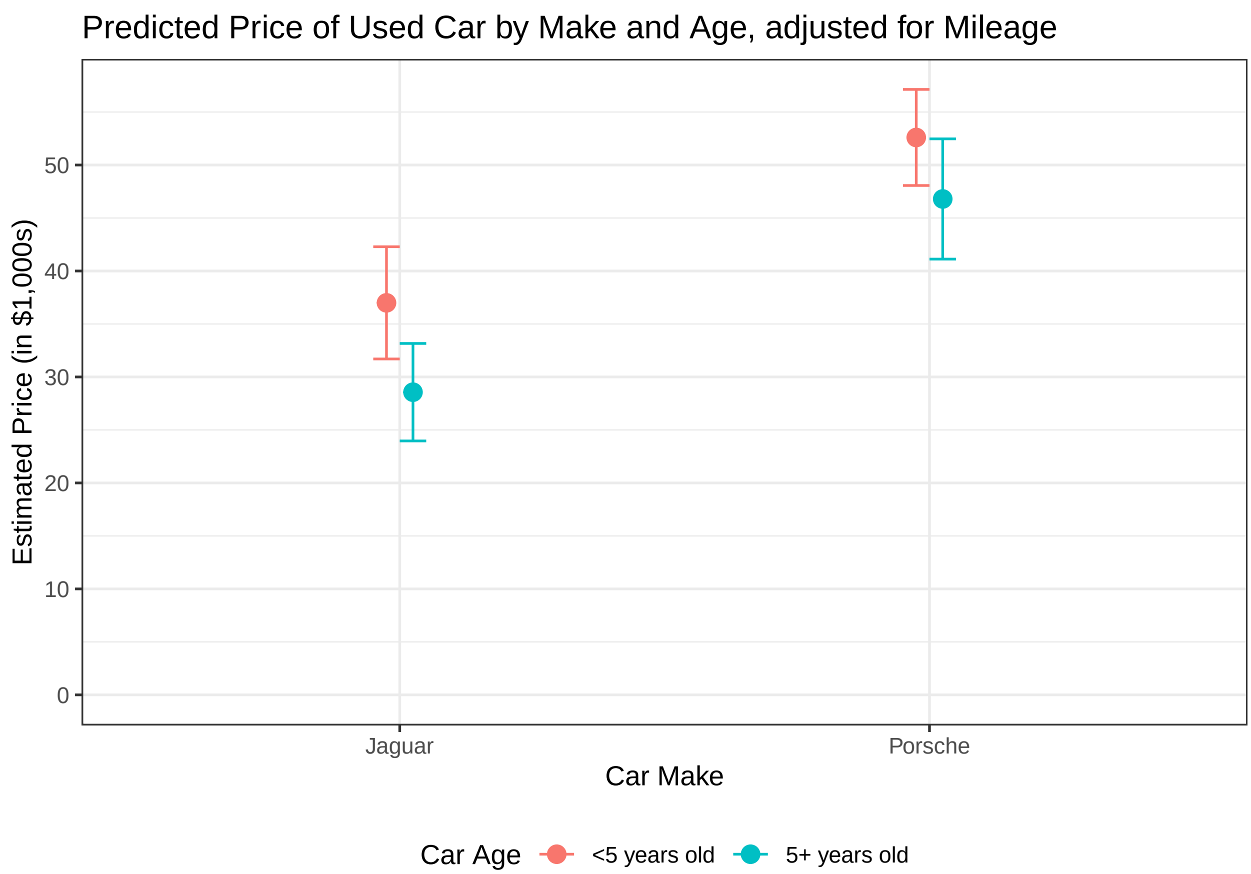
```
meanMiles <- mean(PJ$Mileage)
newdat <- data.frame(Car = c("Porsche", "Porsche", "Jaguar", "Jaguar"),
    age_le5 = c("< 5yo", "5+yo"), Mileage = meanMiles)
preds <- as.data.frame(predict(m0, newdat, interval = "confidence",
    level = 0.95))
allpreds <- bind_cols(newdat, preds)
```

Plotting

With a point and CI for the prediction around the point

```
Price_point<-allpreds %>% ggplot(aes(x=Car, y=fit,
                                     colour=age_le5, group=age_le5)) +
    geom_errorbar(aes(ymin=lwr, ymax=upr),
 # Defining
      width=.1, position=position_dodge(0.1)) +
  # Slightly offsetting the points by group
    geom_point(position=position_dodge(0.1),
  # Making the points smaller
      size=3) +
    # Changing x-axis label
    xlab("Car Make") +
   # Changing y-axis label
    ylab("Estimated Price (in $1,000s)") +
  # Legend label
    scale_color_hue(name="Car Age",
  # Defining the colors by your variable categories
                    breaks=c("< 5yo", "5+yo"),
   # Making longer, sensible variable labels for the legend
                    labels=c("<5 years old",
                      "5+ years old")) +
```

```
  # Title for the overall graph
    ggtitle("Predicted Price of Used Car by Make and Age, adjusted for Mileage") +
  # Expand y range to start at 0
    expand_limits(y=0) +
  # Set tick every 5
    scale_y_continuous(breaks=0:60*10) +
   # Making
    theme_bw() +
  # Position legend in bottom centered
    theme(legend.position="bottom")

Price_point
```



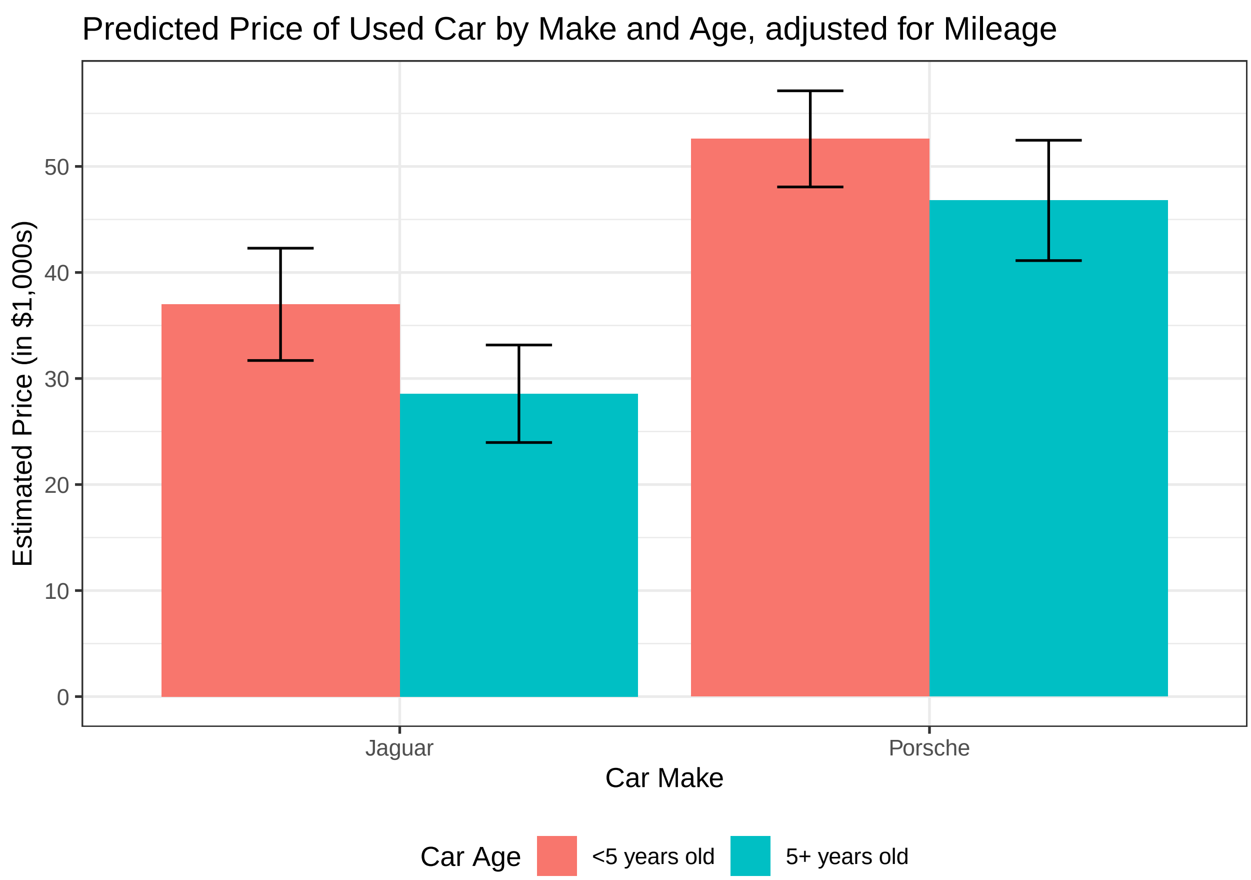

```
ggsave("Price_point.jpg")

## Saving 7 x 5 in image
```

Same as above but as a bar

Here the top of the bar is the predicted price. It's a little goofy to depict as a bar, but you could imagine that the bar height is literally the stack of $1 bills for the price of that car.

```
Price_bar<-allpreds %>%
  ggplot(
# Defining x, y axis and
# the grouping variable that fills in the bar
  aes(x=Car, y=fit, fill=age_le5)) +
```

```r
# Dodge makes the bars be next to each other
    geom_bar(position=position_dodge(.9),
# Making the bars
      stat="identity") +
    geom_errorbar(position=position_dodge(.9),
# Defining the CI levels and line width
      width=.25, aes(ymin=lwr, ymax=upr)) +
# Changing x-axis label
    xlab("Car Make") +
 # Changing y-axis label
    ylab("Estimated Price (in $1,000s)") +
 # Legend label
    scale_fill_discrete(name="Car Age",
# Defining the colors by your variable categories
                     breaks=c("< 5yo", "5+yo"),
# Making longer, sensible variable labels for the legend
                     labels=c("<5 years old",
                       "5+ years old")) +
  # Title for the overall graph
    ggtitle("Predicted Price of Used Car by Make and Age, adjusted for Mileage") +
  # Expand y range to start at 0
    expand_limits(y=0) +
  # Set tick every 5
    scale_y_continuous(breaks=0:60*10) +
   # Making the grey background go away
    theme_bw() +
  # Position legend in bottom centered
    theme(legend.position="bottom")

Price_bar
```




```r
ggsave("Price_bar.jpg")
```

```
## Saving 7 x 5 in image
```