Class 6: Interpreting Logistic Regression

```r
### Load packages:
#library(knitr)
library(tidyverse)
#library(data.table)
#library(janitor)
#library(haven)
#library(readxl)
#library(psych)
#library(statar)
library(mktg482)
library(sjPlot)
```

We often want to interpret variables in a model. Let's try this on the Tuango RFM dataset.

```r
load("../../Data/Tuango_rfm.Rdata")
```

First, we estimate our logistic regression model:

```r
lr <- glm(buyer ~ recency + frequency + monetary + platform + category,
          family = binomial, data = tuango)
```

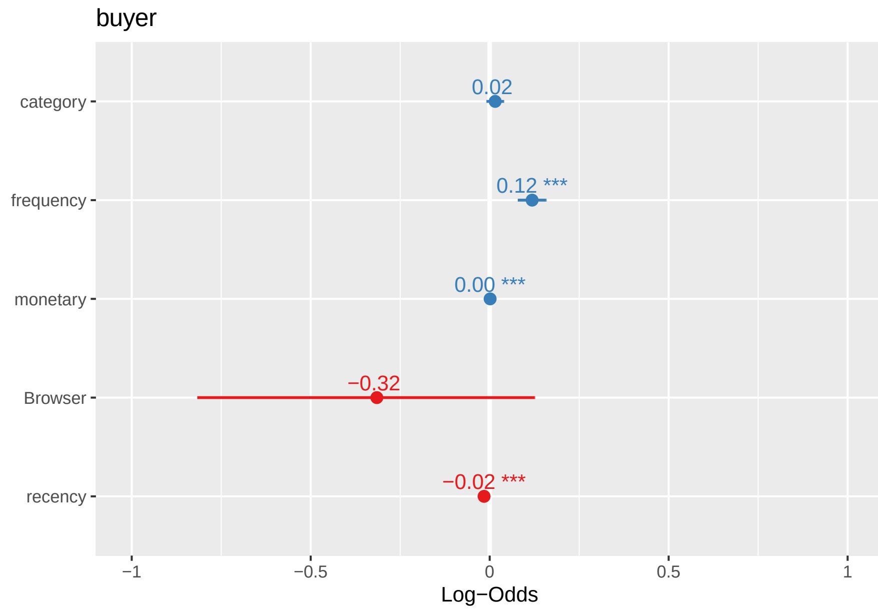
As we discussed, logistic regression coefficient estimates are hard to interpret:

```r
summary(lr)
```

```

Call:
glm(formula = buyer ~ recency + frequency + monetary + platform +
    category, family = binomial, data = tuango)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.8188  -0.2831  -0.2518  -0.1785   4.1222

Coefficients:
                  Estimate Std. Error z value Pr(>|z|)
(Intercept)     -3.4904692  0.1402524 -24.887  < 2e-16 ***
recency         -0.0156296  0.0019678  -7.943 1.98e-15 ***
frequency        0.1186148  0.0201425   5.889 3.89e-09 ***
monetary         0.0012814  0.0002386   5.370 7.88e-08 ***
platformBrowser -0.3151997  0.2390529  -1.319    0.187
category         0.0153914  0.0121877   1.263    0.207
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 3852.0  on 13938  degrees of freedom
Residual deviance: 3651.4  on 13933  degrees of freedom
AIC: 3663.4

Number of Fisher Scoring iterations: 8
```

The `sjPlot` package helps assess the effects of predictor variables, in particular via the `plot_model` command. We can use the package to display the coefficients together with confidence intervals. Make sure you use the options `show.values = TRUE, transform = NULL`.

```
plot_model(lr, show.values = TRUE, transform = NULL)
```



Nonetheless, it is still difficult to interpret what these estimates mean. Instead, we use two approaches to more easily interpret logistic regression coefficient estimates: "Variable Importance" and "Marginal Effects Plots"

Variable Importance

Often we want to know how important a given variable is relative to other variables. One common way to do this is to normalize the coeffients by the standard deviation of the variable. A problem with this idea is that coefficient estimates for continuous and binary variables not comparable. Andrew Gelman proposed a solution in, "Scaling regression inputs by dividing by two standard deviations," Statistics in Medicine (2008), Vol. 27, pp. 2965-2873.

The idea is to measure the effect of continuous variables by considering the effect of a 2 standard deviation change in the variable, and leaving the effect of binary variables unchanged. Since the SD of an evenly-split binary variable (i.e., one with an equal number of 0s and 1s) is 0.5, a 0->1 "one unit change" in a binary variable is equivalent to a 2 SD change. Consequently, the coefficient represents the effect a 2*SD change for such variables (in additional to the convention interpretation as the effect of a one unit change), and now we can compare across the two kinds of variables.

This idea is implemented in R functions I have created for you: `varimp.logistic()` and its companion plotting function `plotimp.logistic()` which can be piped. To use these functions, you need to use load the `mktg482` library as we did using the command `library(mktg482)`; if that command fails for you, you need to install this package by typing:

```
devtools::install_github("fzettelmeyer/mktg482", upgrade = "never", force = TRUE)
```

into the console (you only need to do this once).

The syntax for using the `varimp.logistic()` and `plotimp.logistic()` functions is:

```
varimp.logistic(lrmodel)
```

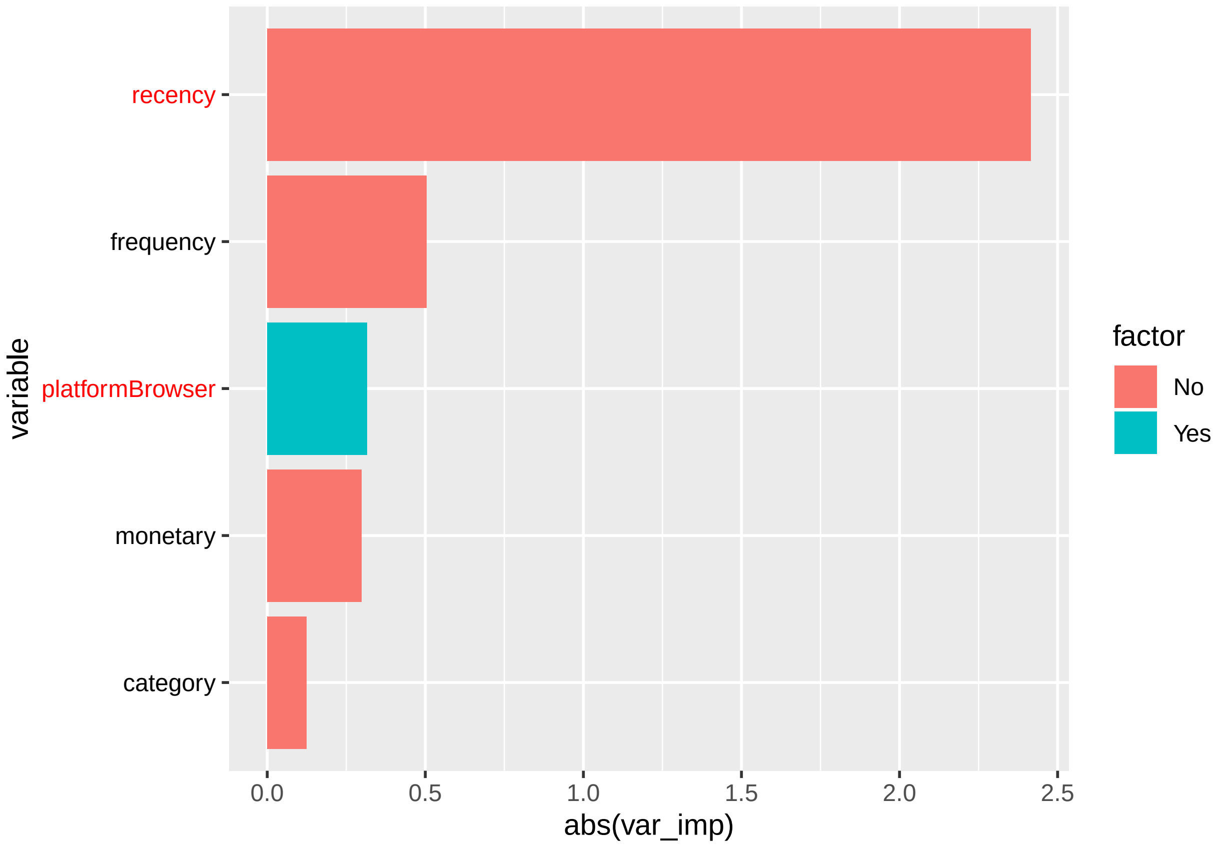
```
plotimp.logistic(lrmodel)
```

Let's try these functions on the Tuango RFM dataset. The first gives a table that lists each variable, whether it is a factor or not, its variable importance before taking the absolute value (so as to preserve information about the sign of the effect), a confidence interval for the variable importance, and the p-value from the logistic regression. The second is a figure which shows the absolute value of variable importance, where information on the sign of the effect is provided via the color of the x-axis label (black=positive; red=negative).

```
varimp.logistic(lr)
```

```
# A tibble: 5 x 6
  variable        factor var_imp var_imp_lower var_imp_upper p_value
  <chr>           <chr>    <dbl>         <dbl>         <dbl>   <dbl>
1 recency         No      -2.41        -3.01          -1.82    0
2 frequency       No       0.504        0.337          0.672   0
3 platformBrowser Yes     -0.315       -0.784          0.153   0.187
4 monetary        No       0.298        0.189          0.406   0
5 category        No       0.124       -0.0685         0.317   0.207
```

```
varimp.logistic(lr) %>% plotimp.logistic()
```



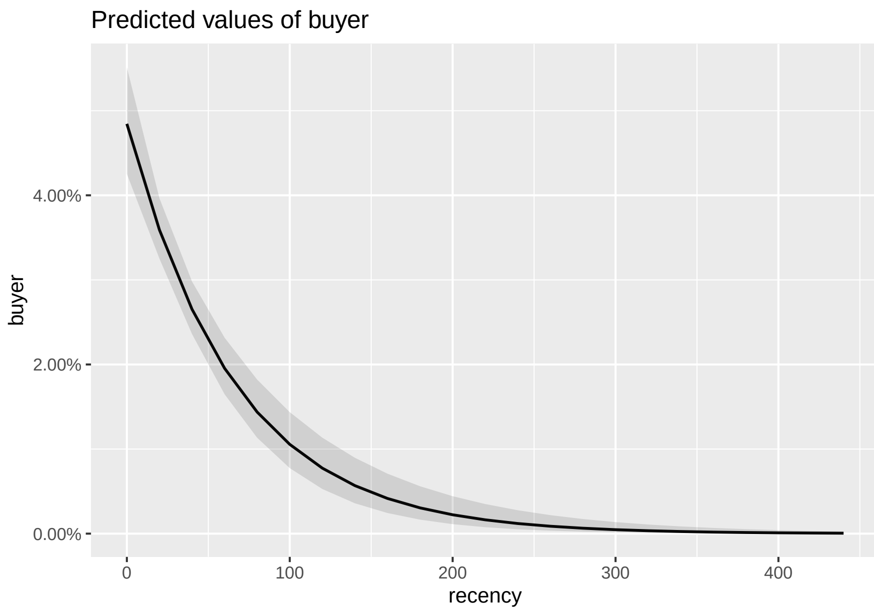

```
# A tibble: 5 x 6
  variable        factor var_imp var_imp_lower var_imp_upper p_value
  <chr>           <chr>    <dbl>         <dbl>         <dbl>   <dbl>
1 recency         No      -2.41        -3.01          -1.82    0
2 frequency       No       0.504        0.337          0.672   0
3 platformBrowser Yes     -0.315       -0.784          0.153   0.187
4 monetary        No       0.298        0.189          0.406   0
5 category        No       0.124       -0.0685         0.317   0.207
```

Marginal Effects

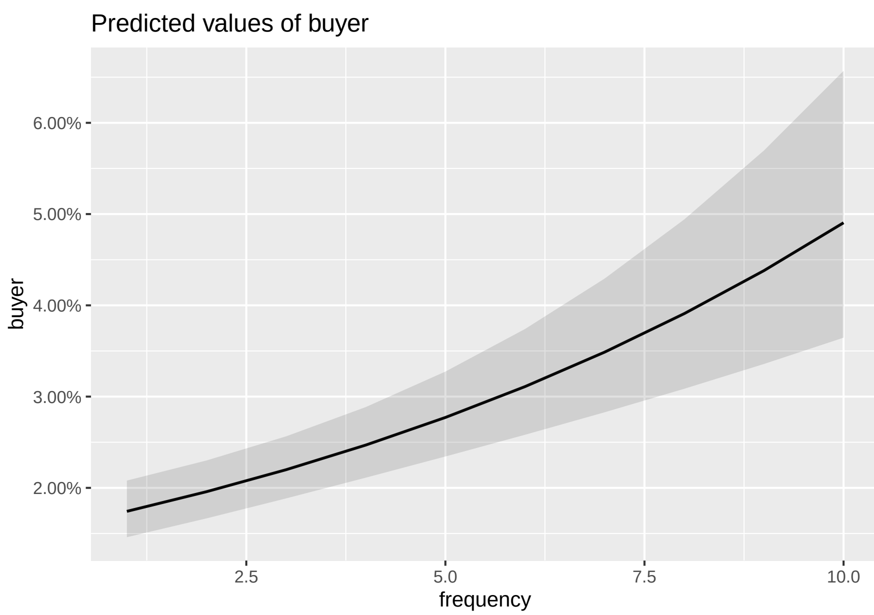
Another way to interpret the coefficient of a variable is to analyze the "marginal effect" of the variable. This corresponds to asking "Holding all other variables at their current values, what is the effect of changing Variable X on the model prediction?" We can get this by using the `plot_model` command with the `type = "eff"` option. This shows the effect of each variable in turn:

```
plot_model(lr, type= "eff")
```

```
$recency
```




```
$frequency
```




```
$monetary
```

$platform

$category

If you want to look at one variable only, use the 'terms = "FEATURE" option where FEATURE is the variable for which you want to see the marginal effect.

```
plot_model(lr, type= "eff", terms = "recency")
```