# Model Fitting

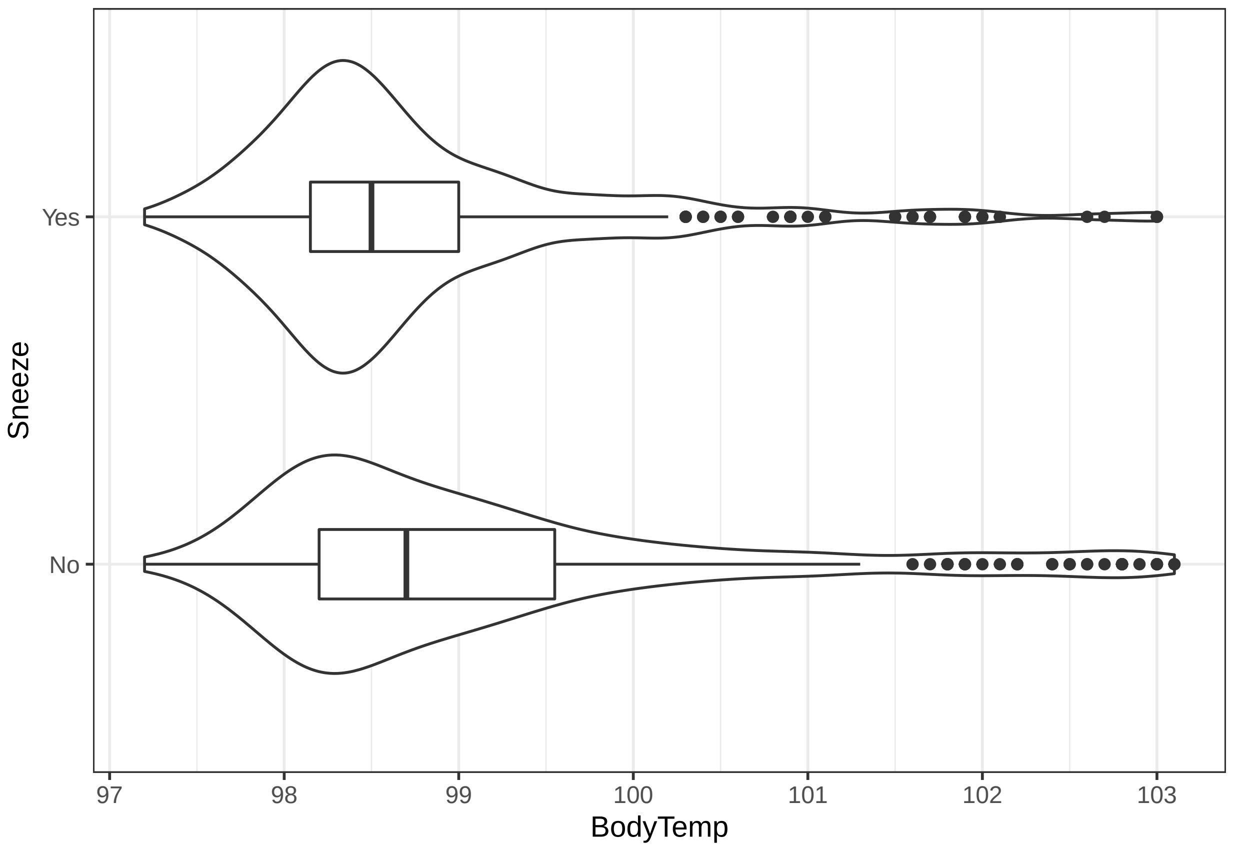
Joe Martin

10/14/2021

## Introduction

To begin modeling the Sneeze variable, I'm adding the boxplot and regression read-out from my exploration:

```r
sneeze_boxplot <- df %>% ggplot(aes(x=BodyTemp, y = Sneeze))+
  geom_violin()+
  geom_boxplot(width = .2)+
  theme_bw()
sneeze_boxplot
```



p-value of .0000006037

```r
temp_sneeze <- lm(BodyTemp ~ Sneeze, data = df)
summary(temp_sneeze)
```

```
##
## Call:
## lm(formula = BodyTemp ~ Sneeze, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.9490 -0.7496 -0.3490  0.3504  4.2504
##
## Coefficients:
##             Estimate Std. Error  t value Pr(>|t|)
## (Intercept) 99.14897    0.06411 1546.478  < 2e-16 ***
## SneezeYes   -0.39935    0.08760   -4.559 6.04e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.18 on 728 degrees of freedom
## Multiple R-squared:  0.02775,    Adjusted R-squared:  0.02642
## F-statistic: 20.78 on 1 and 728 DF,  p-value: 6.037e-06
```

## Models

### Linear Regression

I'll begin running this as a linear regression. This is not the preferred regression to use here because the variable of interest is categorical. However, this will provide an approximate estimate of probability. The code below generates summary stats for this regression.

```r
# Fits a linear model to the continuous outcome using only the main predictor of interest.
lm_mod <- linear_reg() %>%
  set_engine("lm")

lm_fit <- lm_mod %>%
  fit(BodyTemp ~ Sneeze, data = df)

lm_fit
```

```
## parsnip model object
##
## Fit time:  21ms
##
## Call:
## stats::lm(formula = BodyTemp ~ Sneeze, data = data)
##
## Coefficients:
## (Intercept)    SneezeYes
##     99.1490      -0.3994
```

```
tidy(lm_fit)
```

```
## # A tibble: 2 x 5
##   term        estimate std.error statistic    p.value
##   <chr>          <dbl>     <dbl>     <dbl>      <dbl>
## 1 (Intercept)   99.1      0.0641   1546.   0
## 2 SneezeYes     -0.399    0.0876     -4.56 0.00000604
```

Now I'll plot Sneezing alongside other interesting variables, like Weakness and SubjectiveFever. I can review these results and compare models for each variable with summary stats, as well as a dot-and-whisker plot.

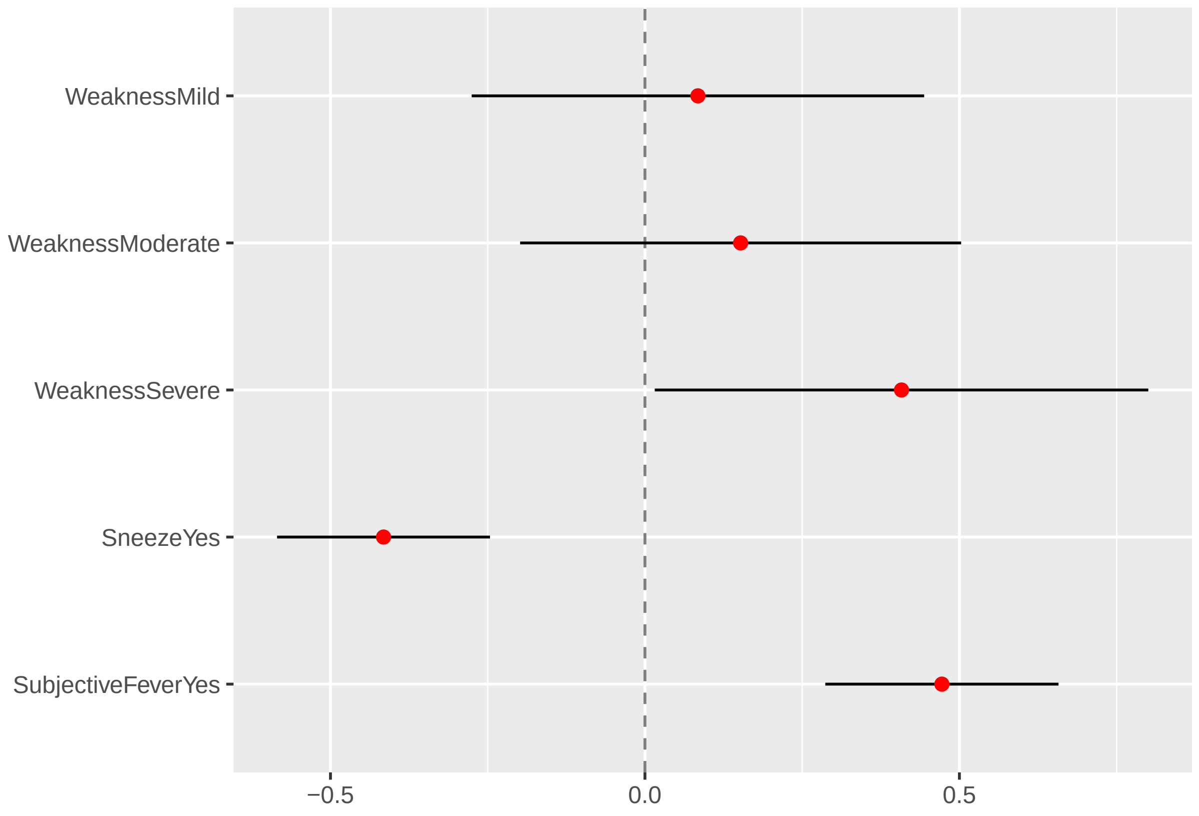
```
# Fits another linear model to the continuous outcome using all (important) predictors of interest.
lm_fit_more <-
  lm_mod %>%
  fit(BodyTemp ~ Weakness + Sneeze + SubjectiveFever, data = df)

lm_fit_more
```

```
## parsnip model object
##
## Fit time:  11ms
##
## Call:
## stats::lm(formula = BodyTemp ~ Weakness + Sneeze + SubjectiveFever,
##     data = data)
##
## Coefficients:
##        (Intercept)        WeaknessMild    WeaknessModerate      WeaknessSevere
##           98.67107             0.08423             0.15213             0.40795
##          SneezeYes  SubjectiveFeverYes
##           -0.41563             0.47214
```

```
tidy(lm_fit_more)
```

```
## # A tibble: 6 x 5
##   term               estimate std.error statistic     p.value
##   <chr>                 <dbl>     <dbl>     <dbl>       <dbl>
## 1 (Intercept)          98.7       0.176    559.    0
## 2 WeaknessMild          0.0842    0.184      0.459 0.646
## 3 WeaknessModerate      0.152     0.179      0.850 0.395
## 4 WeaknessSevere        0.408     0.200      2.04  0.0420
## 5 SneezeYes            -0.416     0.0864    -4.81  0.00000181
## 6 SubjectiveFeverYes    0.472     0.0946     4.99  0.000000751
```

```
tidy(lm_fit_more) %>%
  dwplot(dot_args = list(size = 2, color = "red"),
         whisker_args = list(color = "black"),
         vline = geom_vline(xintercept = 0, color = "grey50", linetype = 2))
```



Next, I'll use glance to view compare the output between the target variable (Sneeze) and the secondary variables I selected.

```
# Compares the model results for the model with just the main predictor and all predictors.
glance(lm_fit)
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared sigma statistic    p.value    df logLik   AIC   BIC
##       <dbl>         <dbl> <dbl>     <dbl>      <dbl> <dbl>  <dbl> <dbl> <dbl>
## 1    0.0278        0.0264  1.18      20.8 0.00000604     1 -1156. 2318. 2332.
## # ... with 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

```
glance(lm_fit_more)
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared sigma statistic  p.value    df logLik   AIC   BIC
##       <dbl>         <dbl> <dbl>     <dbl>    <dbl> <dbl>  <dbl> <dbl> <dbl>
## 1    0.0784        0.0721  1.15      12.3 1.73e-11     5 -1136. 2287. 2319.
## # ... with 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

**Logistic Regression**

A logistic model will likely be a better choice in this case because the Sneeze variable has values of Yes and No. According to ISLR, logistic models will predict the probability that a variable belongs to a certain category. I'll start by viewing summary statistics again.

```r
# Fits a logistic model to the categorical outcome using only the main predictor of interest.
log_mod <- logistic_reg() %>%
  set_engine("glm")

log_fit <-
  lm_mod %>%
  fit(BodyTemp ~ Sneeze, data = df)
```

```r
log_fit
```

```
## parsnip model object
##
## Fit time:  0ms
##
## Call:
## stats::lm(formula = BodyTemp ~ Sneeze, data = data)
##
## Coefficients:
## (Intercept)    SneezeYes
##     99.1490      -0.3994
```

```r
tidy(log_fit)
```

```
## # A tibble: 2 x 5
##   term        estimate std.error statistic    p.value
##   <chr>          <dbl>     <dbl>     <dbl>      <dbl>
## 1 (Intercept)   99.1      0.0641   1546.   0
## 2 SneezeYes     -0.399    0.0876     -4.56 0.00000604
```

Finally, I'll compare the target variable, Sneeze, with the other variables of interest. I'll do this with a summary statistics table, as well as a dot-and-whisker plot.

```r
# Fits another logistic model to the categorical outcome using all (important) predictors of interest.
log_fit_more <-
  log_mod %>%
  fit(Sneeze ~ BodyTemp + Weakness + SubjectiveFever, data = df)

log_fit_more
```

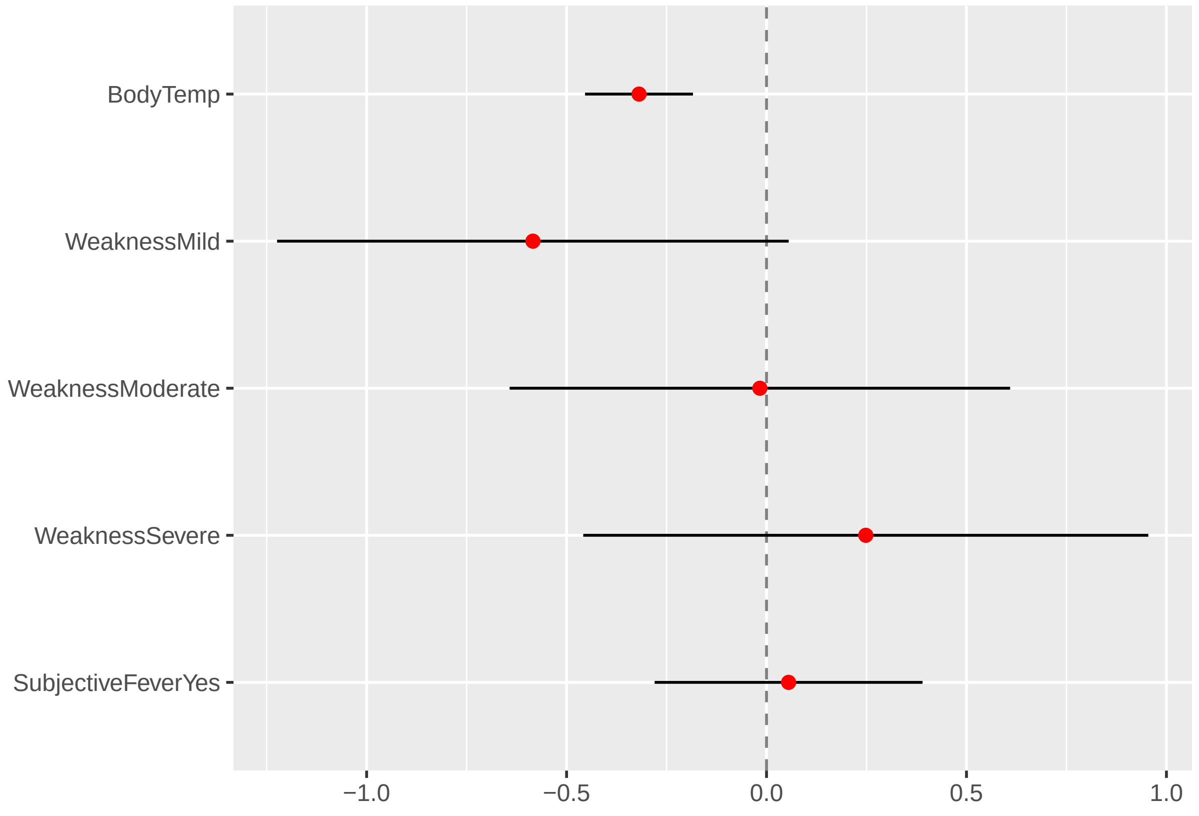
```
## parsnip model object
##
## Fit time:  11ms
##
## Call:  stats::glm(formula = Sneeze ~ BodyTemp + Weakness + SubjectiveFever,
##     family = stats::binomial, data = data)
##
## Coefficients:
##        (Intercept)             BodyTemp         WeaknessMild     WeaknessModerate
##           31.79656             -0.31886             -0.58421             -0.01668
##     WeaknessSevere   SubjectiveFeverYes
##            0.24817              0.05515
```

```
##
## Degrees of Freedom: 729 Total (i.e. Null);  724 Residual
## Null Deviance:       1008
## Residual Deviance: 971.2     AIC: 983.2
```

```
tidy(log_fit_more)
```

```
## # A tibble: 6 x 5
##   term                estimate std.error statistic    p.value
##   <chr>                  <dbl>     <dbl>     <dbl>      <dbl>
## 1 (Intercept)          31.8       6.79      4.69   0.00000279
## 2 BodyTemp             -0.319     0.0689   -4.63   0.00000365
## 3 WeaknessMild         -0.584     0.326    -1.79   0.0734
## 4 WeaknessModerate     -0.0167    0.319    -0.0522 0.958
## 5 WeaknessSevere        0.248     0.360     0.688  0.491
## 6 SubjectiveFeverYes    0.0551    0.171     0.323  0.747
```

```
tidy(log_fit_more) %>%
  dwplot(dot_args = list(size = 2, color = "red"),
         whisker_args = list(color = "black"),
         vline = geom_vline(xintercept = 0, color = "grey50", linetype = 2))
```


BodyTemp
WeaknessMild
WeaknessModerate
WeaknessSevere
SubjectiveFeverYes