SOLUTIONS Notebook: R Workshop Day 3

Regression - Simple, Multiple, Logistic

Your Name Goes Here!

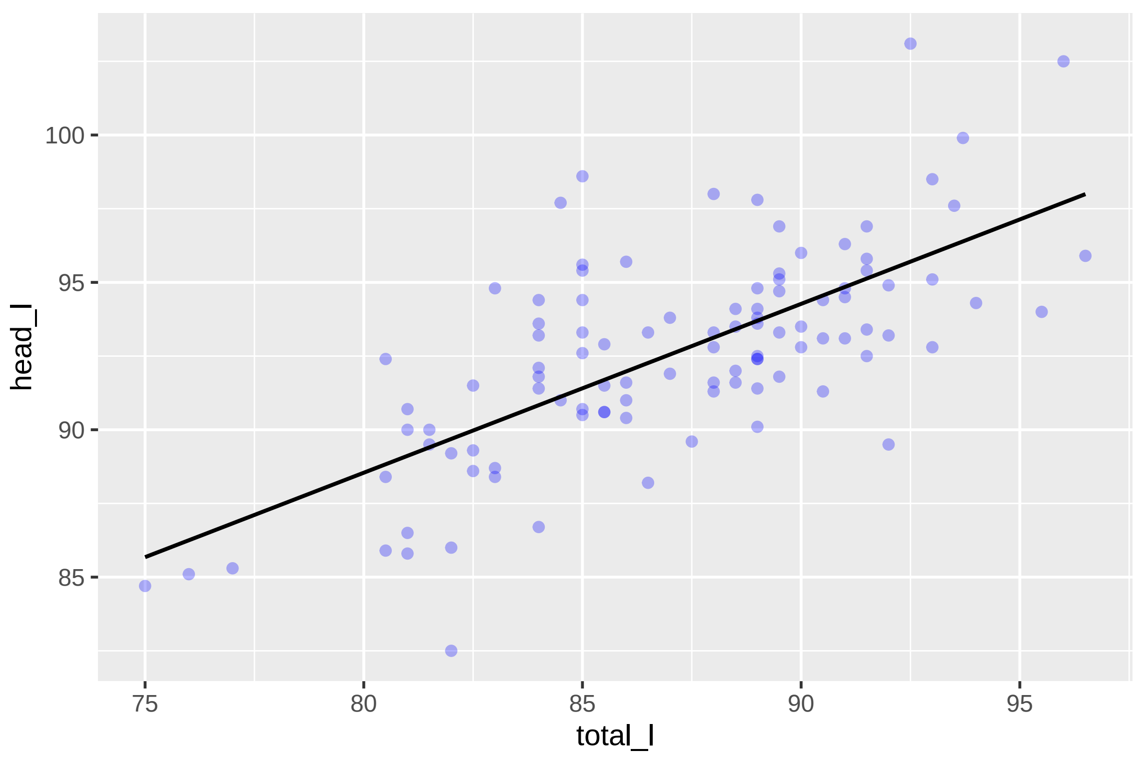
2023-05-26

Simple Linear Regression

In the openintro, view the **possum** data frame.

Make a scatterplot of head length plotted against total length.

```
gf_point(head_l ~ total_l, data = possum, color = "blue", alpha = 0.3) +
  geom_lm(color = "black")
```



Compute the correlation.

```
cor(head_l ~ total_l, data = possum, use = "complete")
```

```
## [1] 0.6910937
```

Fit the linear model using the lm(y ~ x) function.

```
model <-lm(head_l ~ total_l, data = possum)
```

Type just the name of the model to see what has been stored in this variable.

```
model
```

```
##
## Call:
## lm(formula = head_l ~ total_l, data = possum)
##
## Coefficients:
## (Intercept)      total_l
##     42.7098       0.5729
```

Use the summary function to see more details.

```
summary(model)
```

```
##
## Call:
## lm(formula = head_l ~ total_l, data = possum)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.1877 -1.5340 -0.3345  1.2788  7.3968
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 42.70979    5.17281   8.257 5.66e-13 ***
## total_l      0.57290    0.05933   9.657 4.68e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.595 on 102 degrees of freedom
## Multiple R-squared:  0.4776, Adjusted R-squared:  0.4725
## F-statistic: 93.26 on 1 and 102 DF,  p-value: 4.681e-16
```

The function msummary() is a mosaic function that gives terser output than summary().

```
msummary(model)
```

```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 42.70979    5.17281   8.257 5.66e-13 ***
## total_l      0.57290    0.05933   9.657 4.68e-16 ***
##
## Residual standard error: 2.595 on 102 degrees of freedom
## Multiple R-squared:  0.4776, Adjusted R-squared:  0.4725
## F-statistic: 93.26 on 1 and 102 DF,  p-value: 4.681e-16
```

Use makeFun() to make a function of the model and test it out!

```r
f <-makeFun(model)
f(80)
```

```
##       1
## 88.5419
```

Extractor function: coef()

```r
coef(model)
```

```
## (Intercept)     total_l
##  42.7097931   0.5729013
```

Extractor function: confint()

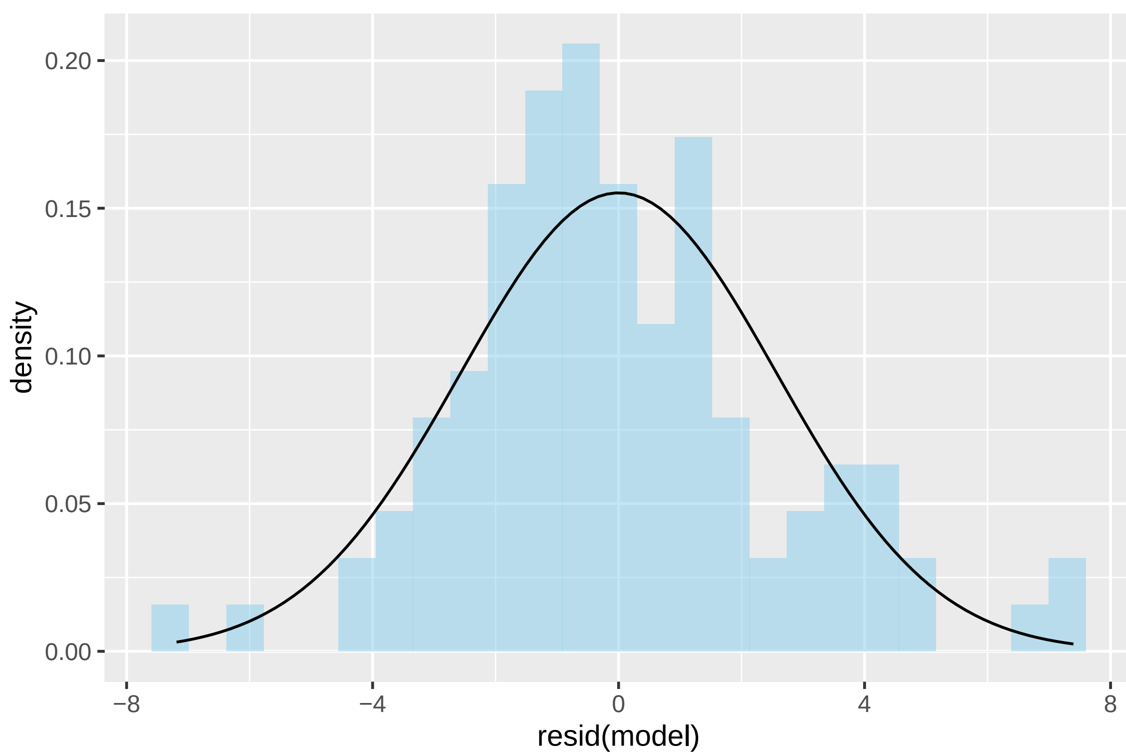
```r
confint(model)
```

```
##                  2.5 %     97.5 %
## (Intercept) 32.4495415 52.9700448
## total_l      0.4552298  0.6905728
```

Model Diagnostics

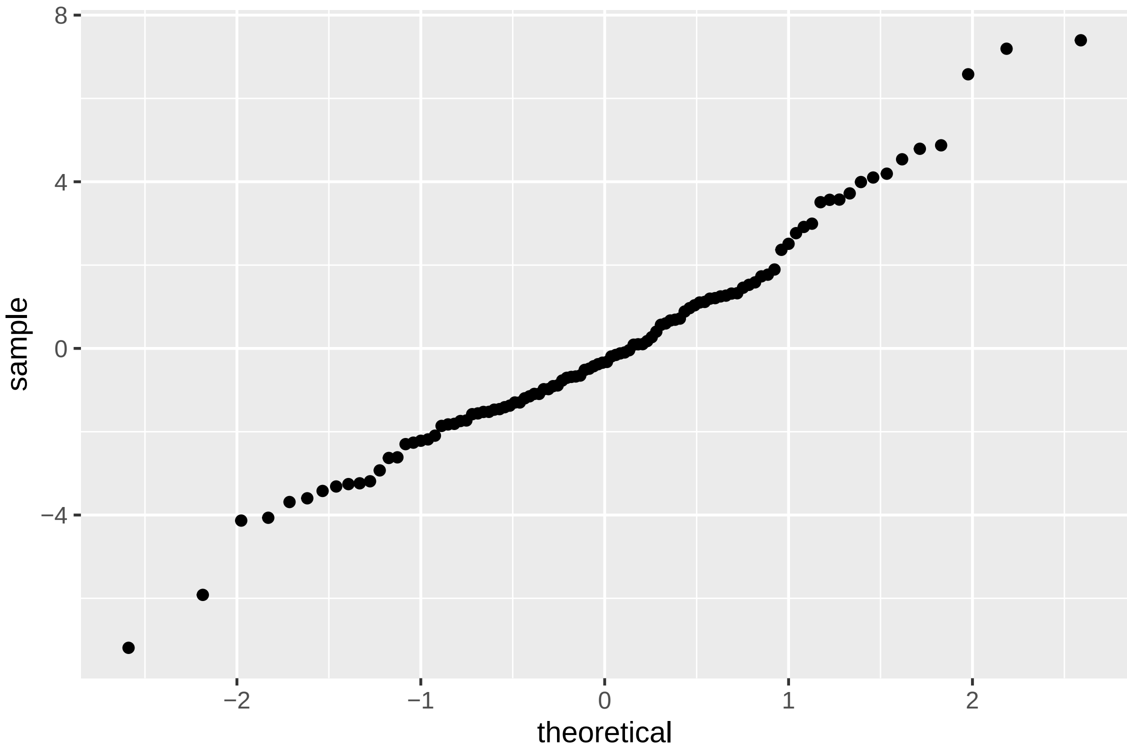
Histogram of residuals.

```r
gf_dhistogram(~ resid(model), fill = "skyblue") %>% gf_fitdistr(dist = "dnorm")
```



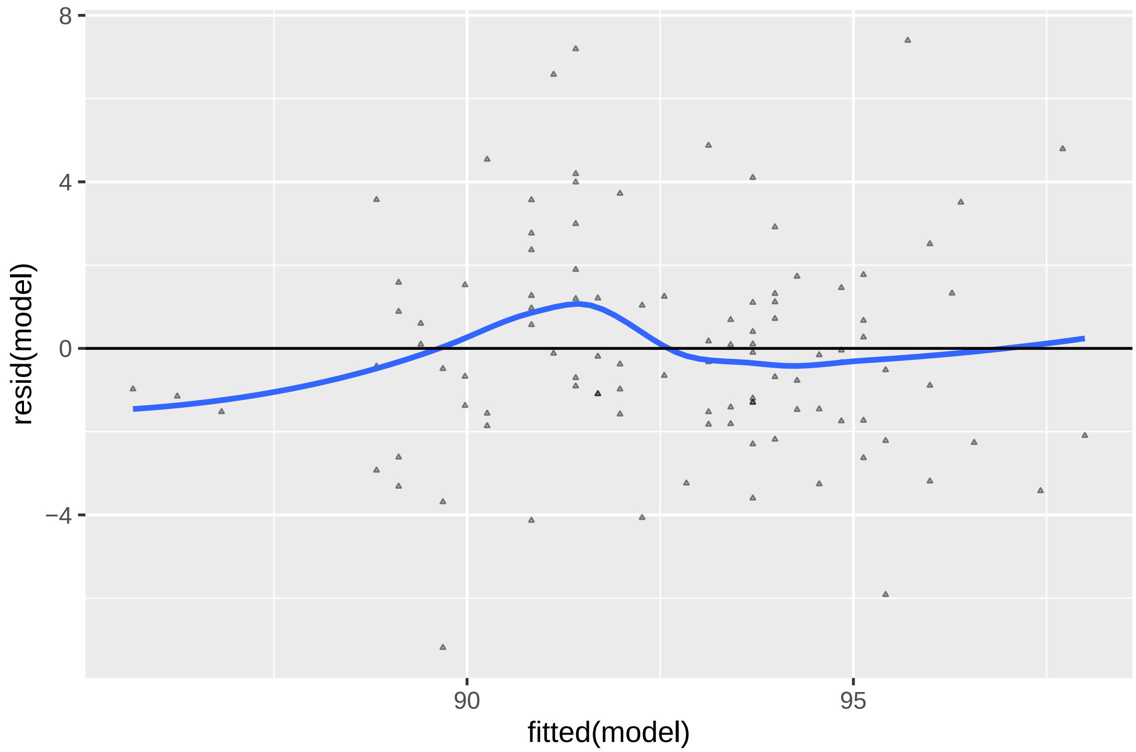
QQ plot.

```
gf_qq(~ resid(model))
```



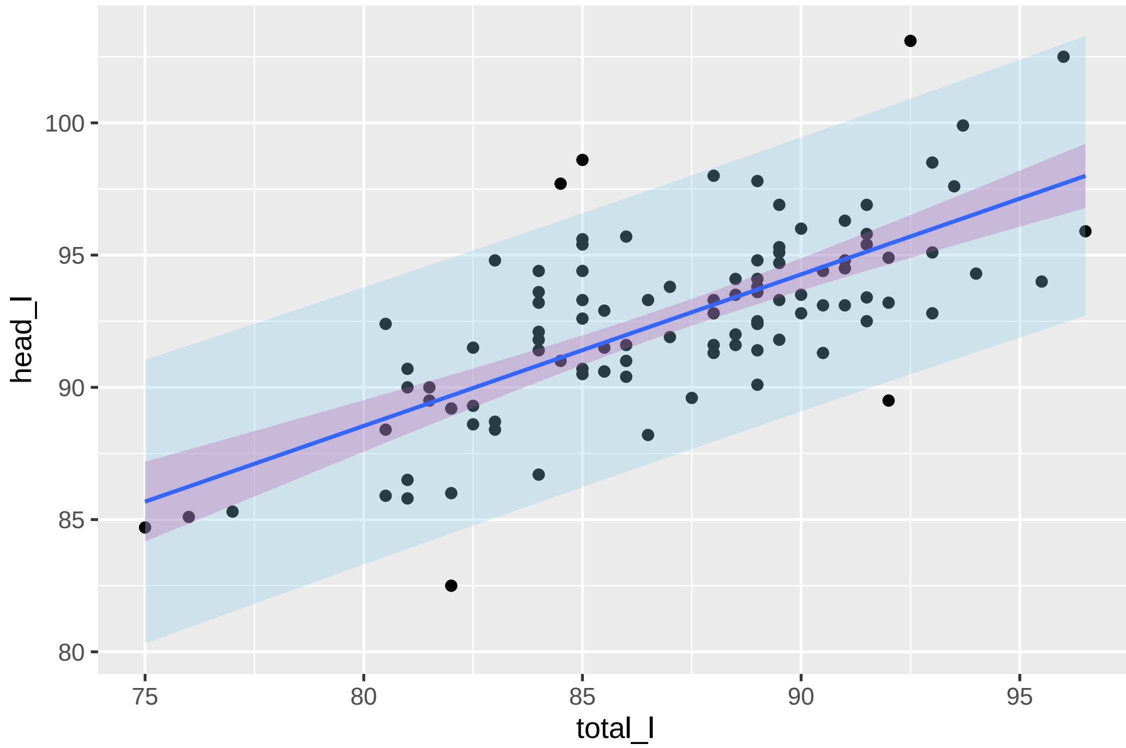
Plot of Residual vs Fitted Values.

```
gf_point(resid(model) ~ fitted(model),
         alpha = 0.5, cex = 0.3, pch = 24) %>%
  gf_smooth(se = FALSE) %>%
  gf_hline(yintercept = 0)
```



Confidence and Prediction Bands.

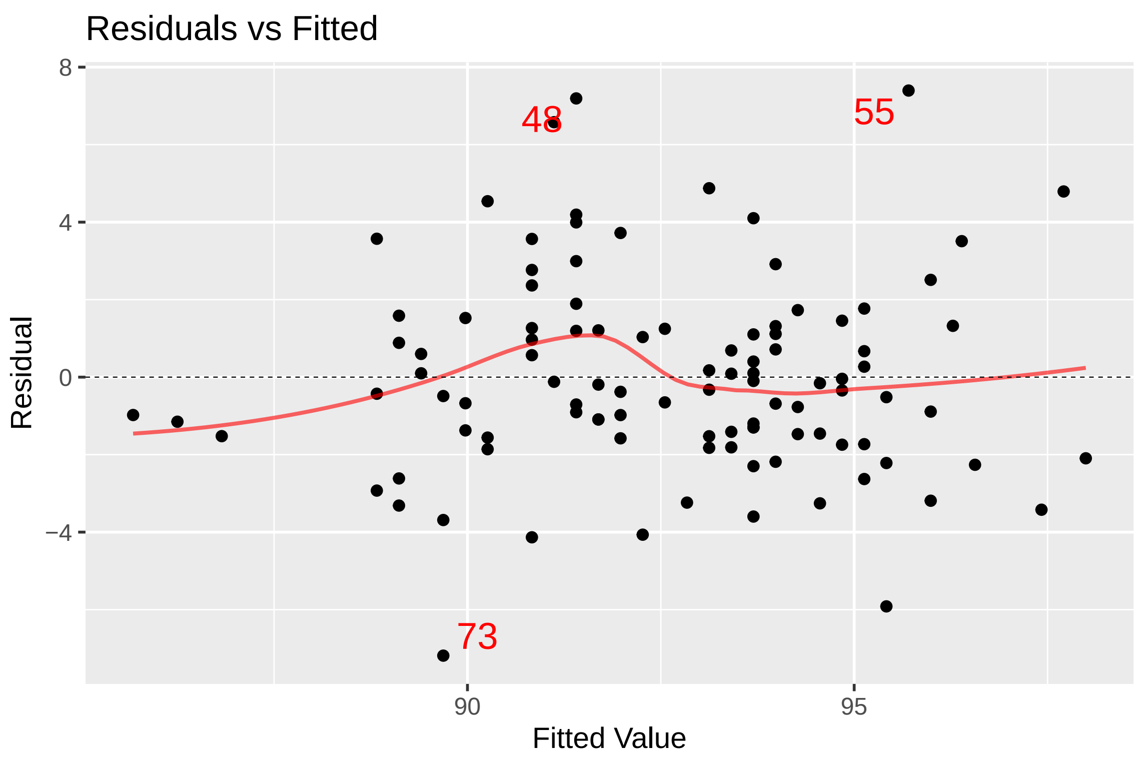
```
gf_point(head_l ~ total_l, data = possum) %>%
  gf_lm(interval = "confidence", fill = "violetred") %>%
  gf_lm(interval = "prediction", fill = "skyblue")
```



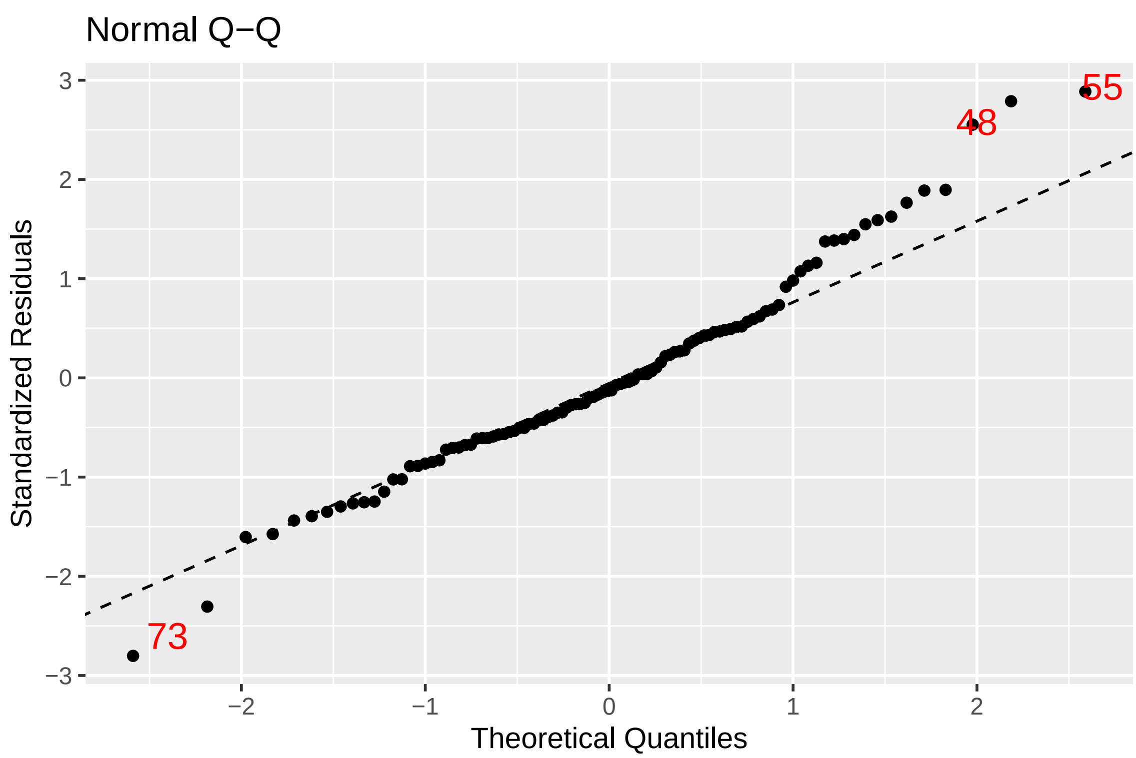
The mplot() function from the mosaic package gives useful model diagnoistic plots.
There is more than one mplot function, so if the function gives you no output then specify the package using mosaic::mplot()

```
mosaic::mplot(model)
```

```
## [[1]]
```

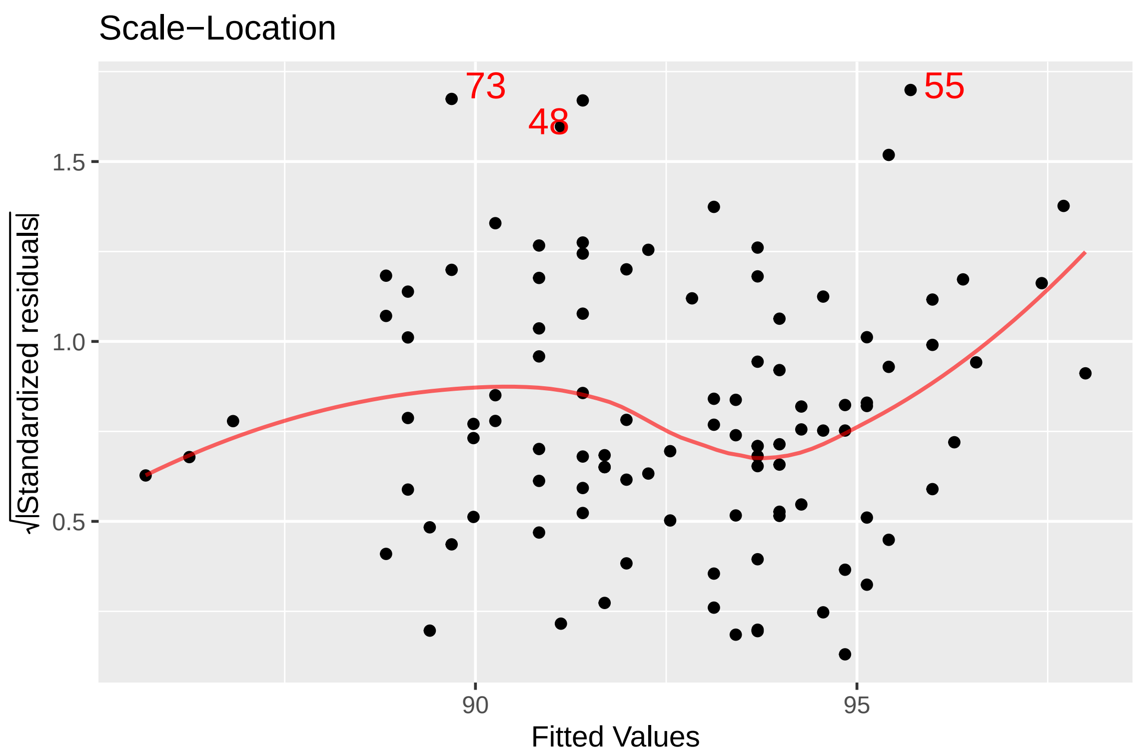

Residuals vs Fitted
8
4
0
−4
Residual
48
55
73
90
95
Fitted Value

```
##
## [[2]]
```

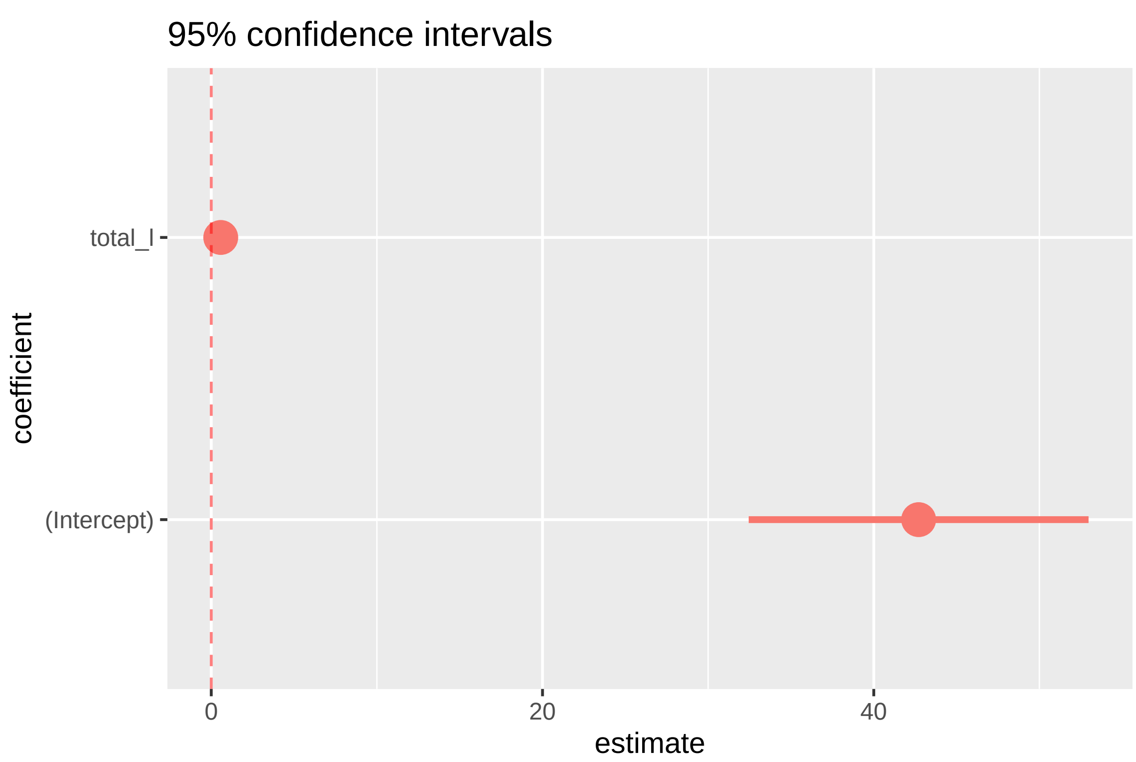

Normal Q−Q
3
2
1
0
−1
−2
−3
Standardized Residuals
48
55
73
−2
−1
0
1
2
Theoretical Quantiles

```
##
```

```
## [[3]]
```


Scale−Location
73
48
55
1.5
1.0
0.5
√|Standardized residuals|
90
95
Fitted Values

```
##
## [[4]]
```


95% confidence intervals
total_l
(Intercept)
coefficient
0
20
40
estimate

Logistic Regression

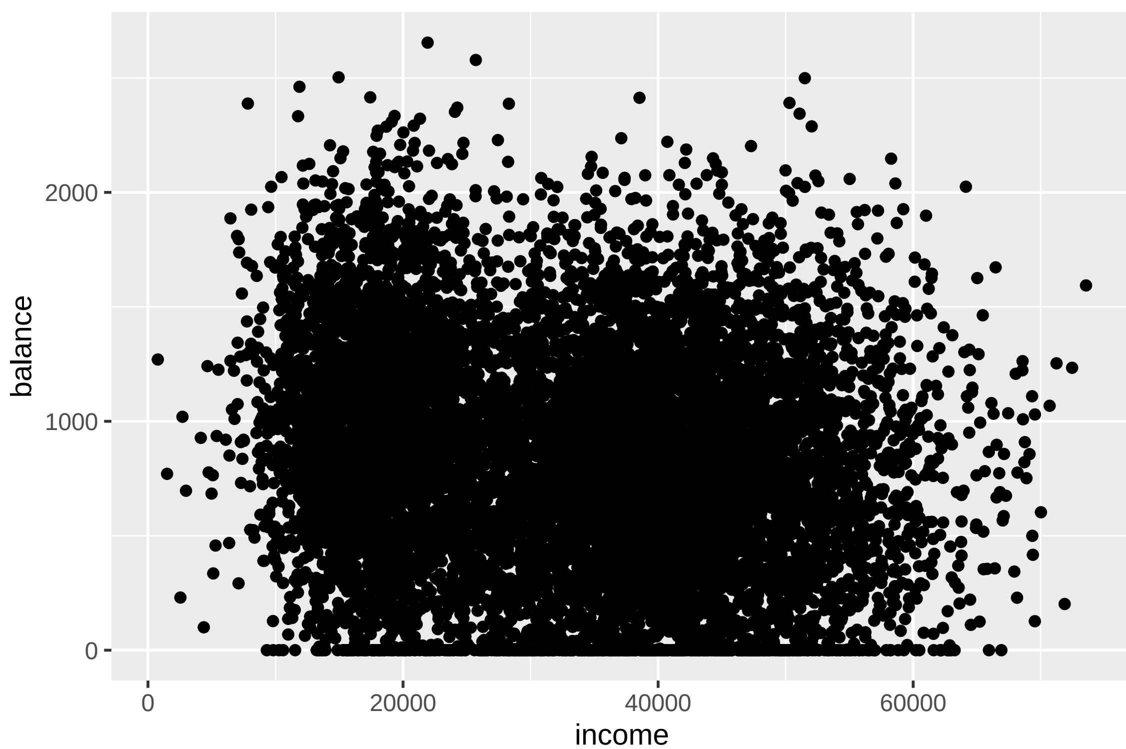
In the ISLR package, there is a data set called Default.
Install the ISLR package, then type ?Default in the console for more information.

Default: "A simulated data set containing information on ten thousand customers. The aim here is to predict which customers will default on their credit card debt."

The response variable of interest is Y = default, a binary categorical variable with levels "Yes" and "No". Let's examine the relation of variables balance and income with default.

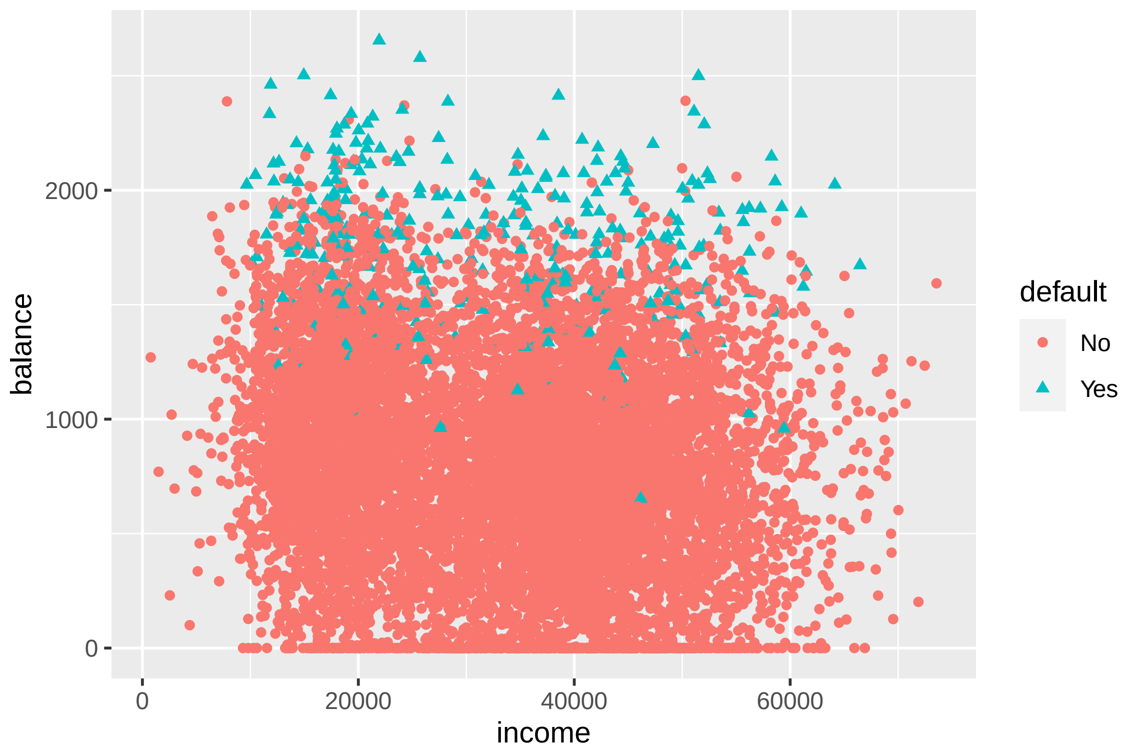
First, a scatterplot of balance vs income.

```
gf_point(balance ~ income, data = Default)
```



Let's set color and shape according to default variable.

```
gf_point(balance ~ income, data = Default, color = ~ default, shape = ~ default)
```

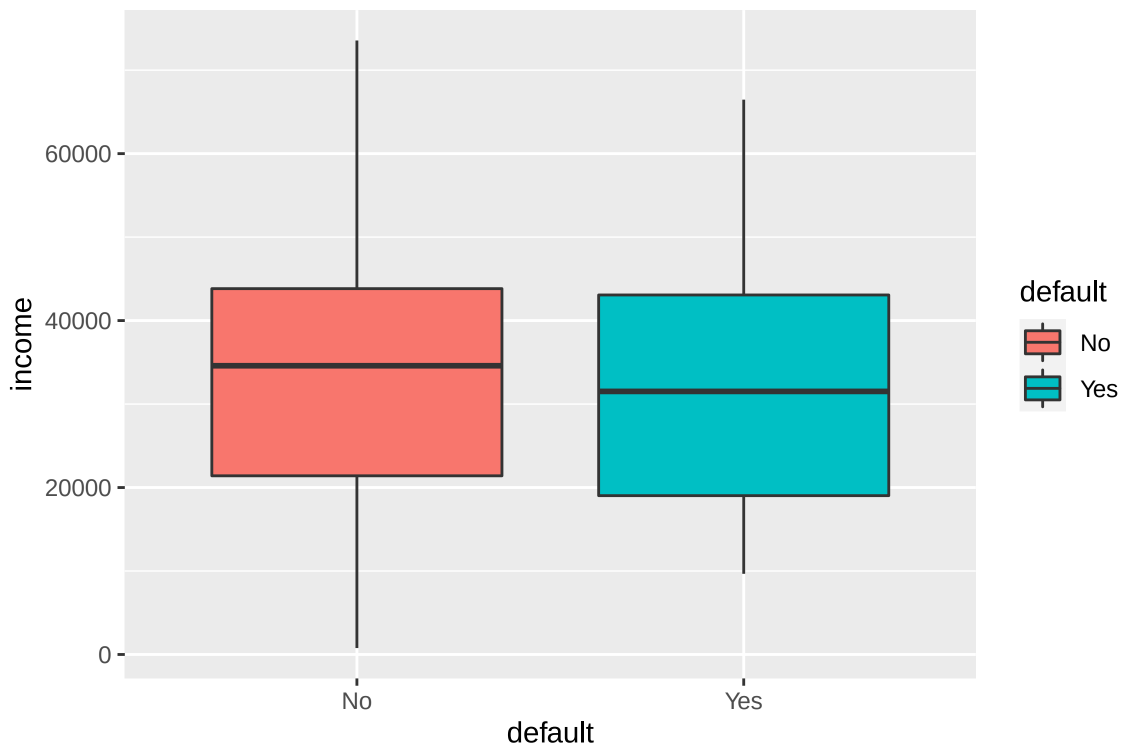


What can we conclude from this?

Let's further confirm with boxplots.

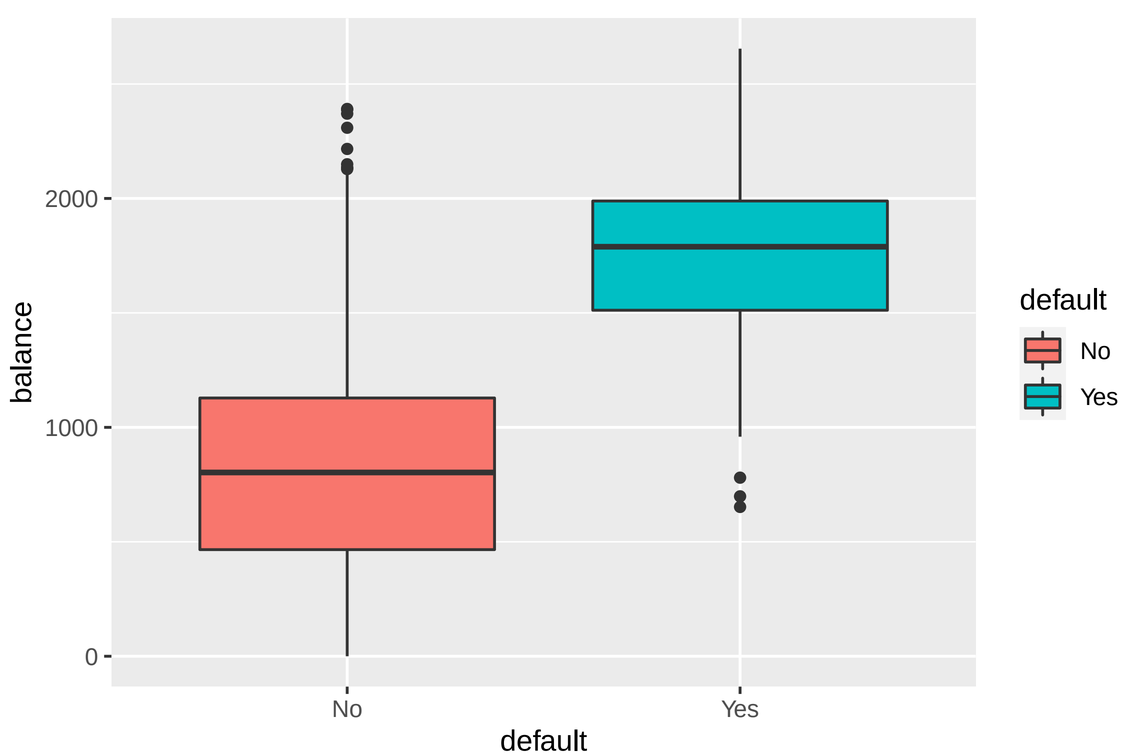
Income broken down default status.

```
gf_boxplot(income ~ default, data = Default, fill = ~ default)
```



Balance broken down default status.

```
gf_boxplot(balance ~ default, data = Default, fill = ~ default)
```



Before proceeding, let's convert "Yes" and "No" into "1" and "0". We use a tidyverse function called mutate to create a new variable (column) called default01. We will need to plot the model.

```
Default01 <- dplyr::mutate(Default, default01 = ifelse(default == "Yes", 1, 0))
tail(Default01, n = 10)
```

```
##       default student   balance   income default01
## 9991       No      No  372.3792 25374.90         0
## 9992       No      No  658.7996 54802.08         0
## 9993       No      No 1111.6473 45490.68         0
## 9994       No      No  938.8362 56633.45         0
## 9995       No     Yes  172.4130 14955.94         0
## 9996       No      No  711.5550 52992.38         0
## 9997       No      No  757.9629 19660.72         0
## 9998       No      No  845.4120 58636.16         0
## 9999       No      No 1569.0091 36669.11         0
## 10000      No     Yes  200.9222 16862.95         0
```

Next, let's fit the model.

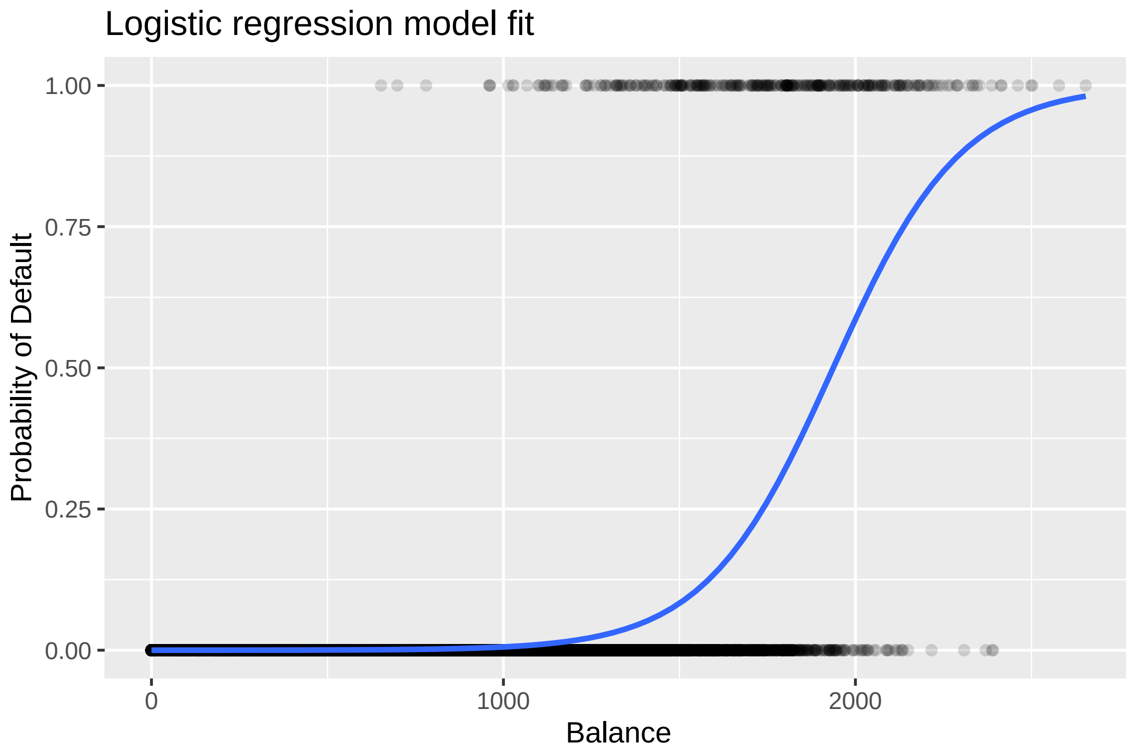
```
options(scipen = 999)
logitmodel <-glm(default01 ~ balance, data = Default01,
                 family = "binomial")
msummary(logitmodel)
```

```
## Coefficients:
##                Estimate  Std. Error z value            Pr(>|z|)
## (Intercept) -10.6513306   0.3611574  -29.49 <0.0000000000000002 ***
## balance       0.0054989   0.0002204   24.95 <0.0000000000000002 ***
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2920.6  on 9999  degrees of freedom
## Residual deviance: 1596.5  on 9998  degrees of freedom
## AIC: 1600.5
##
## Number of Fisher Scoring iterations: 8
```

```
# msummary() is a mosaic function, gives slightly terser output than summary().
```

Plot of Regression Fit Line

```
gf_point(default01 ~ balance, data = Default01,
           alpha = 0.15,
           xlab ="Balance",
           ylab = "Probability of Default",
           title = "Logistic regression model fit") %>%
    gf_smooth(method = "glm", method.args = list(family = "binomial"))
```



Interpreting the Balance Coefficient.
The parameter estimate for balance is on a log-odds scale: it tells us how much of an increase in log-odds is associated with a one-unit increase in balance.
Taking exponential of the coefficients is useful for interpretation.

```
exp(coef(logitmodel))
```

```
##   (Intercept)       balance
## 0.00002366933 1.00551406373
```

For every one dollar increase in monthly balance carried, this tells us the factor by which odds of defaulting increases.

Prediction
Use makeFun() to make a function of the model and test it out! Estimate the probability of default for x = 1000 and x = 2000.

```
f <-makeFun(logitmodel)
f(1000)
```

```
##           1
## 0.005752145
```

```
f(2000)
```

```
##         1
## 0.5857694
```

Using the predict() function:

```r
predict(logitmodel, data.frame(balance = c(1000,2000)), type = "response")
```

```
##           1           2
## 0.005752145 0.585769370
```