Snowshoe Analysis

¶ Garrett Springer ¶

Data and Description

A company that sells outdoor equipment collected data on 371 customers. The company sells a variety of products and is considering selling snowshoes. The following variables were recorded:

Variable	Description
age	Age of the customer in years
product	The primary type of product the customer has previously purchased. One of: "winterSports" (used as the baseline), "mountainSports", "waterSports"
quantity	The total number of unique products the customer has previously purchased
tenure	The number of days since the customers' first purchase
snowshoes	Indication of purchasing snowshoes in the future, if the company were to sell them. One of: 1 (the customer indicated they would buy snowshoes), 0 (the customer indicated they would not buy snowshoes)

Download the snowshoe.txt file from Canvas (Files -> DATA SETS), and put it in the same folder as this R Markdown file.

PART 1 (PREDICT SNOWSHOES)

For Part 1 of this analysis, you will address the company's first goal of using the data from their current customers to create a model to predict whether a particular customer will buy snowshoes.

Complete your exploratory data analysis (EDA) in this section. You may use multiple code chunks, if you wish, to organize your code.

```
snowshoes <- read_table("snowshoe.txt")
```

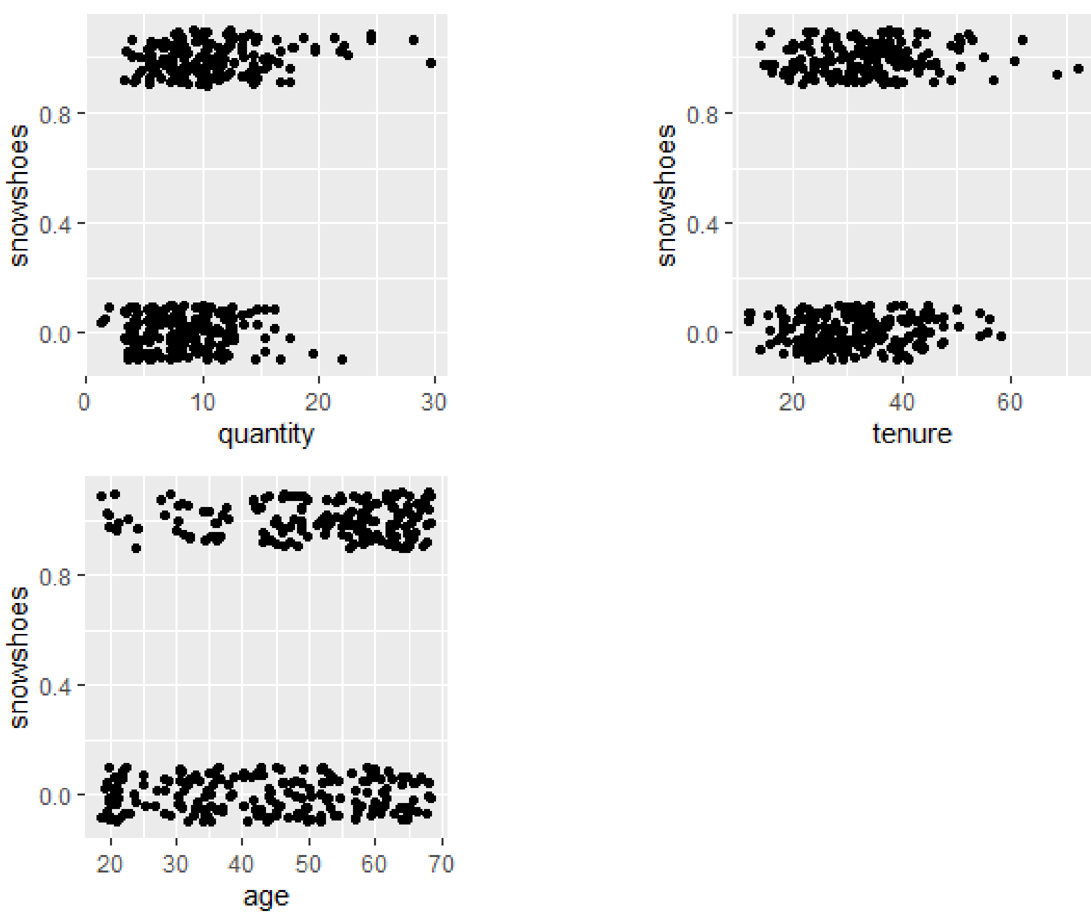
```
##
## -- Column specification --------------------------------------------------------
## cols(
##   quantity = col_double(),
##   tenure = col_double(),
##   age = col_double(),
##   product = col_character(),
##   snowshoes = col_double()
## )
```

```
snowshoes$product <- as.factor(snowshoes$product)
summary(snowshoes)
```

```
##     quantity         tenure           age                    product
##  Min.   : 2.000   Min.   :12.00   Min.   :19.00   mountainSports:105
##  1st Qu.: 6.000   1st Qu.:25.00   1st Qu.:34.50   waterSports   : 49
##  Median : 8.000   Median :32.00   Median :49.00   winterSports  :217
##  Mean   : 9.402   Mean   :32.22   Mean   :46.58
##  3rd Qu.:12.000   3rd Qu.:38.00   3rd Qu.:59.00
##  Max.   :30.000   Max.   :72.00   Max.   :68.00
##    snowshoes
##  Min.   :0.0000
##  1st Qu.:0.0000
##  Median :0.0000
##  Mean   :0.4501
```

```
## 3rd Qu.:1.0000
## Max.   :1.0000
```

```
quan_plot <- snowshoes %>%
ggplot(mapping = aes(x = quantity, y = snowshoes)) +
geom_point(position = position_jitter(height = .1)) +
theme(aspect.ratio = 1)
tenure_plot <- snowshoes %>%
ggplot(mapping = aes(x = tenure, y = snowshoes)) +
geom_point(position = position_jitter(height = .1))+
theme(aspect.ratio = 1)
age_plot <- snowshoes %>%
ggplot(mapping = aes(x = age, y = snowshoes)) +
geom_point(position = position_jitter(height = .1))+
theme(aspect.ratio = 1)
grid.arrange(quan_plot, tenure_plot, age_plot, ncol = 2)
```




```
table(snowshoes$age, snowshoes$snowshoes)
```

```
   ##
## 	0  1
##   19  2  1
##   20 12  3
##   21 12  3
##   22  7  0
##   23  1  1
##   24  2  2
##   25  3  0
##   27  2  0
##   28  3  2
##   29  2  1
##   30  2  3
##   31  8  2
##   32  3  3
##   33  6  0
##   34  5  2
##   35  6  2
##   36  5  4
##   37  3  2
##   38  3  2
##   39  1  0
##   40  3  0
##   41  2  0
##   42  5  5
##   43  4  5
##   44  6  3
##   45  4  3
##   46  4  5
##   47  3  4
##   48  4  3
##   49  5  7
##   50  6  2
##   51  0  1
##   52  6  3
##   53  7  5
##   54  4  2
##   55  1  5
##   56  1  7
##   57  6  5
##   58  2  6
##   59  6  8
##   60  5  4
##   61  6  1
##   62  3  9
##   63  2 12
```

```
##   64  6  4
##   65  4  9
##   66  4  4
##   67  2  5
##   68  5  7
```

```
snowshoes %>%
group_by(product) %>%
summarise(percent_rented = mean(snowshoes))
```

```
## # A tibble: 3 x 2
##   product        percent_rented
##   <fct>                   <dbl>
## 1 mountainSports          0.343
## 2 waterSports             0.163
## 3 winterSports            0.567
```

```
print("Baseline")
```

```
## [1] "Baseline"
```

```
mean(snowshoes$snowshoes)
```

```
## [1] 0.4501348
```

Perform variable selection in this section. You may use multiple code chunks, if you wish, to organize your code.

```
#we need to make indicator variables
snowshoes_new <- snowshoes %>%
mutate(winterSports = ifelse(product == "winterSports", 1, 0)) %>%
mutate(waterSports = ifelse(product == "waterSports", 1, 0)) %>%
mutate(mountainSports = ifelse(product == "mountainSports", 1, 0))
```

```
snowshoes_new$snowshoes <- as.factor(snowshoes_new$snowshoes)
head(snowshoes_new)
```

```
## # A tibble: 6 x 8
##   quantity tenure   age product        snowshoes winterSports waterSports
##      <dbl>  <dbl> <dbl> <fct>          <fct>            <dbl>       <dbl>
## 1        6     33    31 waterSports    0                    0           1
## 2       10     35    47 mountainSports 0                    0           0
## 3       12     36    55 mountainSports 0                    0           0
## 4       16     24    54 mountainSports 0                    0           0
## 5       16     31    20 winterSports   1                    1           0
## 6       10     27    65 winterSports   1                    1           0
## # ... with 1 more variable: mountainSports <dbl>
```

```
snowshoes_x <- as.matrix(snowshoes_new[,c(1,2,3,6,7,8)])
snowshoes_y <- as.matrix(snowshoes_new[, 5])
# use cross validation to pick the "best" (based on MSE) lambda
snowshoes_en <- cv.glmnet(x = snowshoes_x,
                          y = snowshoes_y,
                          family = "binomial",
                          type.measure = "deviance",
                          alpha = .5)
snowshoes_en$lambda.min
```

```
## [1] 0.02097845
```

```
snowshoes_en$lambda.1se
```

```
## [1] 0.09294755
```

```
coef(snowshoes_en, s = "lambda.1se")
```

```
## 7 x 1 sparse Matrix of class "dgCMatrix"
##                         s1
## (Intercept)    -1.89656888
## quantity        0.05469138
## tenure          .
```

```
## age              0.01901768
## winterSports     0.56778095
## waterSports     -0.39667011
## mountainSports   .
```

```
snowshoes_best_subsets_bic <- bestglm(as.data.frame(snowshoes),
                                      IC = "BIC",
                                      method = "exhaustive",
                                      TopModels = 1,
                                      family = binomial)
```

```
## Morgan-Tatar search since family is non-gaussian.
```

```
## Note: factors present with more than 2 levels.
```

```
summary(snowshoes_best_subsets_bic$BestModel)
```

```
##
## Call:
## glm(formula = y ~ ., family = family, data = Xi, weights = weights)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.1021  -0.9530  -0.4264   0.9828   2.1632
##
## Coefficients:
##                      Estimate Std. Error z value Pr(>|z|)
## (Intercept)         -3.611703   0.525985  -6.867 6.58e-12 ***
## quantity             0.115997   0.032145   3.609 0.000308 ***
## age                  0.037884   0.008733   4.338 1.44e-05 ***
## productwaterSports  -1.103465   0.459509  -2.401 0.016332 *
## productwinterSports  1.073872   0.268948   3.993 6.53e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 510.62  on 370  degrees of freedom
## Residual deviance: 425.48  on 366  degrees of freedom
## AIC: 435.48
```

```
##
## Number of Fisher Scoring iterations: 4
```

Fit a model using the variables you selected from the prevous section, and determine in any interaction(s) are needed for this model in this section. You may use multiple code chunks, if you wish, to organize your code.

```r
snowshoes_logistic <- glm(snowshoes ~
                          quantity + age + product,
                          data = snowshoes,
                          family = binomial(link = "logit"))
snow_all_int <- glm(snowshoes ~
                          quantity * age * product,
                          data = snowshoes,
                          family = binomial(link = "logit"))
#no interactions does better than all interactions
anova(snowshoes_logistic, snow_all_int, test = "Chisq")
```

```
## Analysis of Deviance Table
##
## Model 1: snowshoes ~ quantity + age + product
## Model 2: snowshoes ~ quantity * age * product
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1       366     425.48
## 2       359     417.68  7   7.7987   0.3507
```

```r
snow_int_age <- glm(snowshoes ~
                    quantity * age + product * age,
                    data = snowshoes,
                    family = binomial(link = "logit"))
#no interactions does better than interactions with age
anova(snowshoes_logistic, snow_int_age, test = "Chisq")
```

```
## Analysis of Deviance Table
##
```

```
## Model 1: snowshoes ~ quantity + age + product
## Model 2: snowshoes ~ quantity * age + product * age
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1       366     425.48
## 2       363     420.14  3   5.3369   0.1487
```

```
snow_int_quan <- glm(snowshoes ~
        quantity * age + product * quantity,
              data = snowshoes,
        family = binomial(link = "logit"))
#no interactions does better than interactions with quantity
anova(snowshoes_logistic, snow_int_quan, test = "Chisq")
```

```
## Analysis of Deviance Table
##
## Model 1: snowshoes ~ quantity + age + product
## Model 2: snowshoes ~ quantity * age + product * quantity
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1       366     425.48
## 2       363     423.44  3   2.0335   0.5655
```

```
snow_int_prod <- glm(snowshoes ~
        product * age + product * quantity,
              data = snowshoes,
        family = binomial(link = "logit"))
#no interactions does better than interactions with product
anova(snowshoes_logistic, snow_int_prod, test = "Chisq")
```

```
## Analysis of Deviance Table
##
## Model 1: snowshoes ~ quantity + age + product
## Model 2: snowshoes ~ product * age + product * quantity
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1       366     425.48
## 2       362     419.71  4   5.7708   0.2169
```

```
snow_int_quan_age <- glm(snowshoes ~
              quantity * age + product,
                 data = snowshoes,
```

```
                   family = binomial(link = "logit"))
#no interactions does better than interaction between quantity and age
anova(snowshoes_logistic, snow_int_quan_age, test = "Chisq")
```

```
## Analysis of Deviance Table
##
## Model 1: snowshoes ~ quantity + age + product
## Model 2: snowshoes ~ quantity * age + product
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1       366     425.48
## 2       365     424.37  1   1.1059    0.293
```

```
snow_int_quan_prod <- glm(snowshoes ~
                 quantity * product + age,
                   data = snowshoes,
                 family = binomial(link = "logit"))
#no interactions does better than interaction between quantity and product
anova(snowshoes_logistic, snow_int_quan_prod, test = "Chisq")
```

```
## Analysis of Deviance Table
##
## Model 1: snowshoes ~ quantity + age + product
## Model 2: snowshoes ~ quantity * product + age
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1       366     425.48
## 2       364     424.61  2  0.87207   0.6466
```

```
snow_int_age_prod <- glm(snowshoes ~
                 quantity + product * age,
                   data = snowshoes,
                 family = binomial(link = "logit"))
#no interactions does better than interaction between age and product
anova(snowshoes_logistic, snow_int_age_prod, test = "Chisq")
```

```
## Analysis of Deviance Table
##
## Model 1: snowshoes ~ quantity + age + product
## Model 2: snowshoes ~ quantity + product * age
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1       366     425.48
```

```
## 2       364     420.71  2   4.7669  0.09223 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
#after all tests, the original model performs the best without any
#interactions
summary(snowshoes_logistic)
```

```
##
## Call:
## glm(formula = snowshoes ~ quantity + age + product, family = binomial(link = "
##     data = snowshoes)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.1021  -0.9530  -0.4264   0.9828   2.1632
##
## Coefficients:
##                      Estimate Std. Error z value Pr(>|z|)
## (Intercept)         -3.611703   0.525985  -6.867 6.58e-12 ***
## quantity             0.115997   0.032145   3.609 0.000308 ***
## age                  0.037884   0.008733   4.338 1.44e-05 ***
## productwaterSports  -1.103465   0.459509  -2.401 0.016332 *
## productwinterSports  1.073872   0.268948   3.993 6.53e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 510.62  on 370  degrees of freedom
## Residual deviance: 425.48  on 366  degrees of freedom
## AIC: 435.48
##
## Number of Fisher Scoring iterations: 4
```

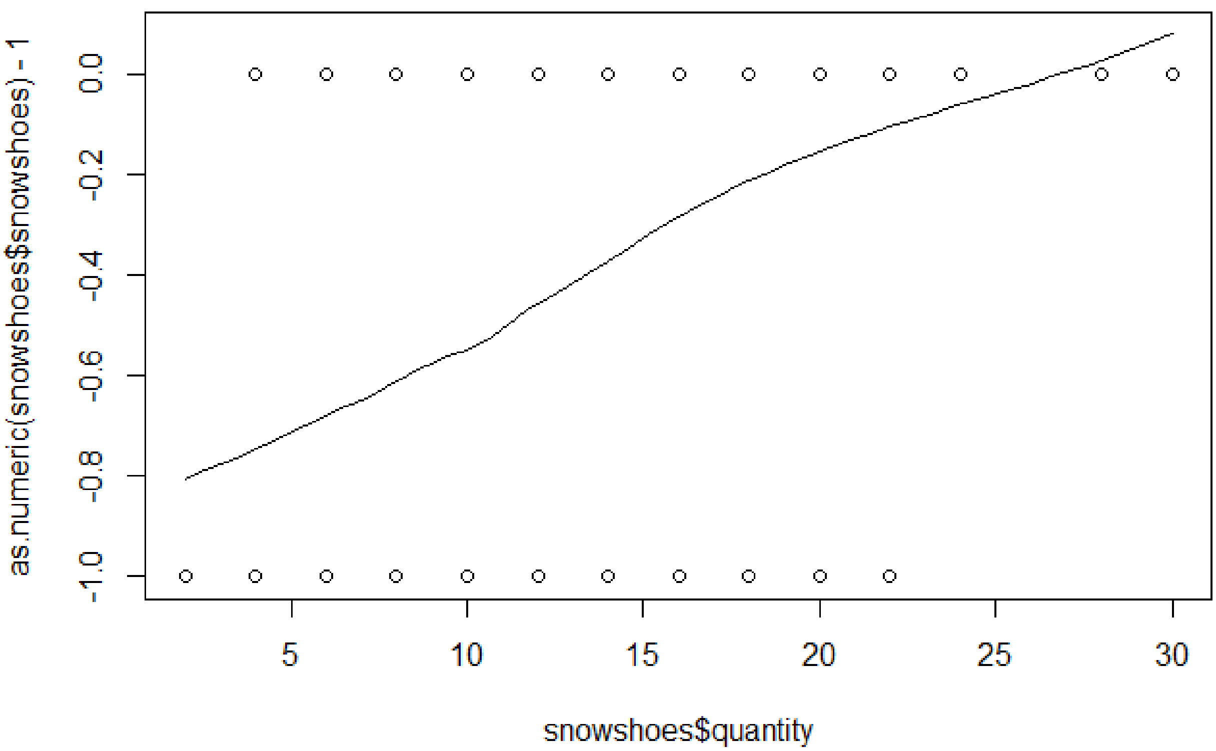
After all tests, the original model performs the best without any interactions

Based on your results above, fit an appropriate ("final") model and check model assumptions in this

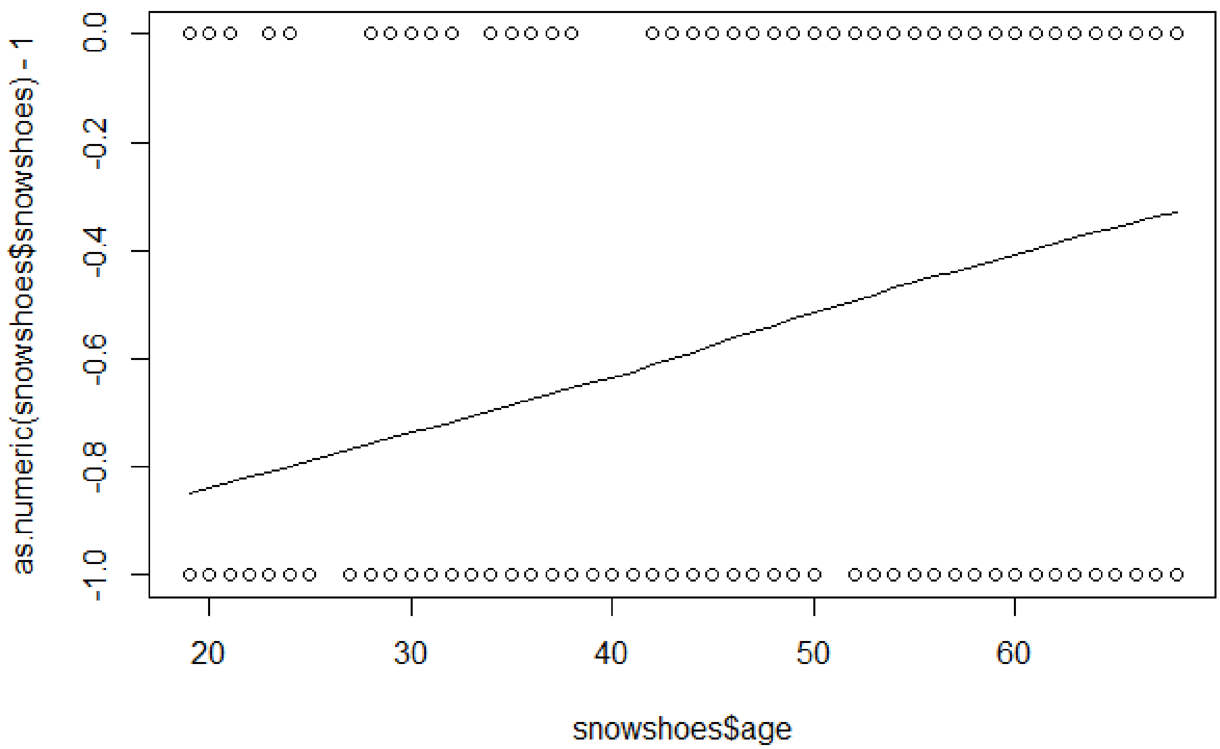
section. You may use multiple code chunks, if you wish, to organize your code.

X vs log odds is linear

```
scatter.smooth(x = snowshoes$quantity,
       y = as.numeric(snowshoes$snowshoes) - 1)
```




```
scatter.smooth(x = snowshoes$age,
       y = as.numeric(snowshoes$snowshoes) - 1)
```



This assumption is met because the lines are strictly increasing.

Observations are independent

It does not say that the data was randomly sampled, however, one persons data should not affect the other. There may be some problems with people being family members or friends. So I would say this assumption is not met but we will move on with our analysis anyway.

No influential points

```
calc_df_fits <- function(df, lm){

df$dffits <- dffits(lm)

plot <- ggplot(data = df) +
geom_point(mapping = aes(x = as.numeric(rownames(df)),
y = abs(dffits))) +
ylab("Absolute Value of DFFITS for Y") +
xlab("Observation Number") +
geom_hline(mapping = aes(yintercept = 2 *  sqrt(length(lm$coefficients) / len
color = "red",
```

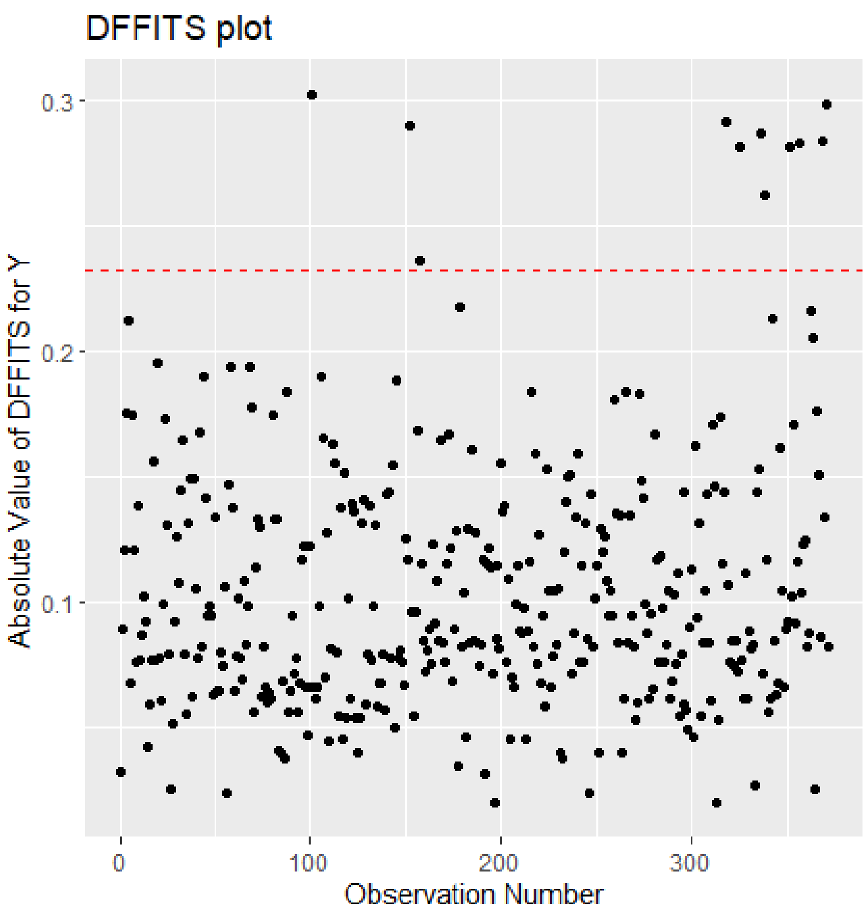
```
                linetype = "dashed") +
    theme(aspect.ratio = 1) +
    labs(title = "DFFITS plot")

  thing <- df %>%
    mutate(rowNum = row.names(df)) %>%
    filter(abs(dffits) > 2 * sqrt(length(lm$coefficients) /
                                    length(dffits))) %>%
    arrange(desc(abs(dffits)))

  out <- list(plot, thing)
  return(out)

}
calc_df_fits(snowshoes, snowshoes_logistic)
```

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




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

```
## # A tibble: 11 x 7
##    quantity tenure   age product        snowshoes dffits rowNum
##       <dbl>  <dbl> <dbl> <fct>              <dbl>  <dbl> <chr>
##  1       22     39    52 mountainSports         0 -0.302 101
##  2        8     44    68 waterSports            1  0.299 370
##  3       14     72    50 waterSports            1  0.292 318
##  4       24     38    30 mountainSports         1  0.290 152
##  5        8     27    63 waterSports            1  0.287 336
##  6       12     31    68 waterSports            1  0.284 368
##  7       10     23    63 waterSports            1  0.283 356
##  8       12     38    63 waterSports            1  0.282 325
##  9       10     29    62 waterSports            1  0.281 351
## 10        8     24    49 waterSports            1  0.263 338
## 11       20     47    61 winterSports           0 -0.237 157
```

This assumption is met. We can see no influential points in the dffits plot.

Additional predictor variables are not required

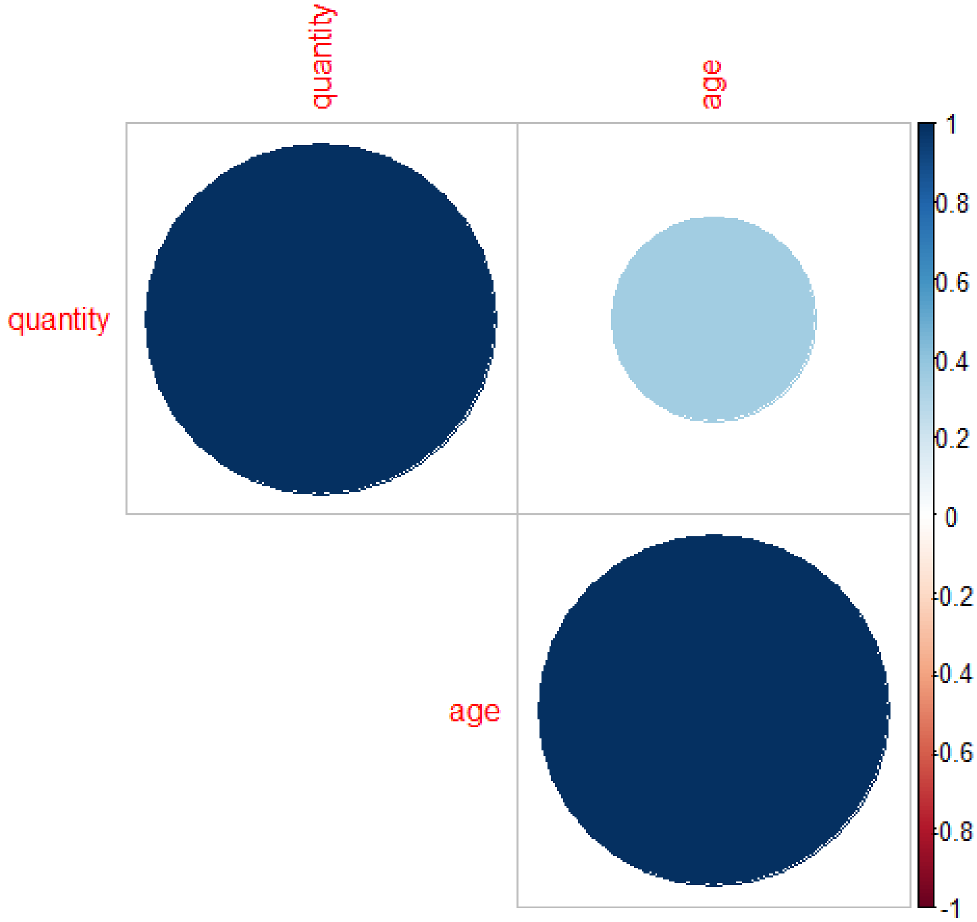
There could possibly be more factors to a person renting snowshoes, but in our model we already removed some of our variables so we should not need more. This assumption is met.

Multicollinearity

```
vif(snowshoes_logistic)
```

```
##              GVIF Df GVIF^(1/(2*Df))
## quantity 1.084980  1        1.041624
## age      1.091764  1        1.044875
## product  1.053354  2        1.013080
```

```
corrplot(cor(snowshoes %>%
  dplyr::select(-snowshoes, -product, -tenure)), type = "upper")
```



The vifs look great. They are all close to 1. This assumption is met.

Complete statistical inference based on the best model you chose in this section. You may use multiple code chunks, if you wish, to organize your code.

```
exp(confint(snowshoes_logistic))
```

```
## Waiting for profiling to be done...
```

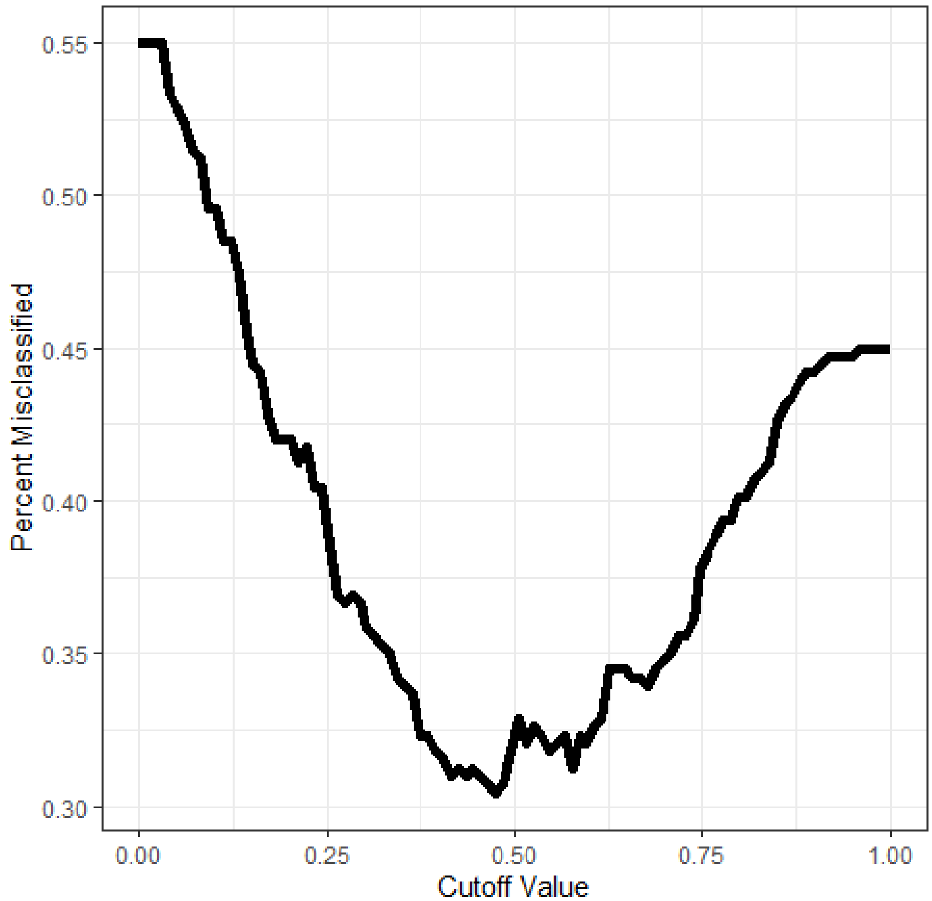
```
##                           2.5 %    97.5 %
## (Intercept)         0.009247604 0.0730338
## quantity            1.056424372 1.1987322
## age                 1.021248050 1.0568982
## productwaterSports  0.127595263 0.7865917
## productwinterSports 1.741103330 5.0083443
```

```r
new_person = data.frame(age = 23,
                        product = "waterSports",
                        quantity = 6)
new_pred <- predict(snowshoes_logistic,
                    newdata = new_person,
                    type = 'response',
                    se.fit = T)
new_log_odds  <- predict(snowshoes_logistic,
                    newdata = new_person,
                    se.fit = T)
xbar <- new_pred$fit
me <- qnorm(p = .975) * new_pred$se.fit
xbar + c(-1,0,1) * me
```

```
## [1] 0.00337649 0.04117618 0.07897588
```

```r
snowshoes_preds <- predict(snowshoes_logistic, type = "response")
# create a sequence from 0 to 1 to represent all possible cut-off values that
# we could choose:
possible_cutoffs <- seq(0, 1, length = 100)
# transform heart$chd from a factor with levels "yes" and "no" to a factor with
# levels 1 and 0:
snowshoes_binary <- snowshoes$snowshoes
# create an empty vector where we will store the percent misclassified for each
# possible cut-off value we created:
percent_misclass <- rep(NA, length(possible_cutoffs))
# for each possible cut-off value, (1) grab the cut-off value, (2) for all 757
# patients, store a 1 in "classify" if their predicted probability is larger
# than the cut-off value, and (3) compute the average percent misclassified
# across the 757 patients when using that cut-off by averaging the number of
# times "classify" (0 or 1 based on how that cut-off classified a person) is
# not the same as heart_binary (the truth):
for(i in 1:length(possible_cutoffs)) {
  cutoff <- possible_cutoffs[i]  # (1)
  classify <- ifelse(snowshoes_preds > cutoff, 1, 0)  # (2)
  percent_misclass[i] <- mean(classify != snowshoes_binary)  # (3)
}
# percent_misclass holds the average misclassification rates for each cut-off
# put this information in a dataframe so we can plot it with ggplot:
misclass_data <- as.data.frame(cbind(percent_misclass, possible_cutoffs))
# plot the misclassification rate against the cut-off value:
ggplot(data = misclass_data) +
  geom_line(mapping = aes(x = possible_cutoffs, y = percent_misclass),
            size = 2) +
```

```
theme_bw() +
xlab("Cutoff Value") +
ylab("Percent Misclassified") +
theme(aspect.ratio = 1)
```




```
# choose the "best" cut-off that minimizes the percent misclassified:
cutoff <- possible_cutoffs[which.min(percent_misclass)]
cutoff
```

```
## [1] 0.4747475
```

```
pred <- snowshoes_preds > cutoff
conf_matrix <- table("truth" = snowshoes$snowshoes,
                     "prediction" = pred)
addmargins(conf_matrix)
```

```
##       prediction
## truth FALSE TRUE Sum
```

```
##   0     146   58  204
##   1      55  112  167
##   Sum   201  170  371
```

```
my_roc <- roc(snowshoes$snowshoes,
                snowshoes_preds)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

```
auc(my_roc)
```

```
## Area under the curve: 0.7593
```

PART 2 (PREDICT TENURE)

For Part 2 of this analysis, you will address the company's second goal of using this data to create a model to predict a customer's tenure (the number of days since the customers' first purchase).

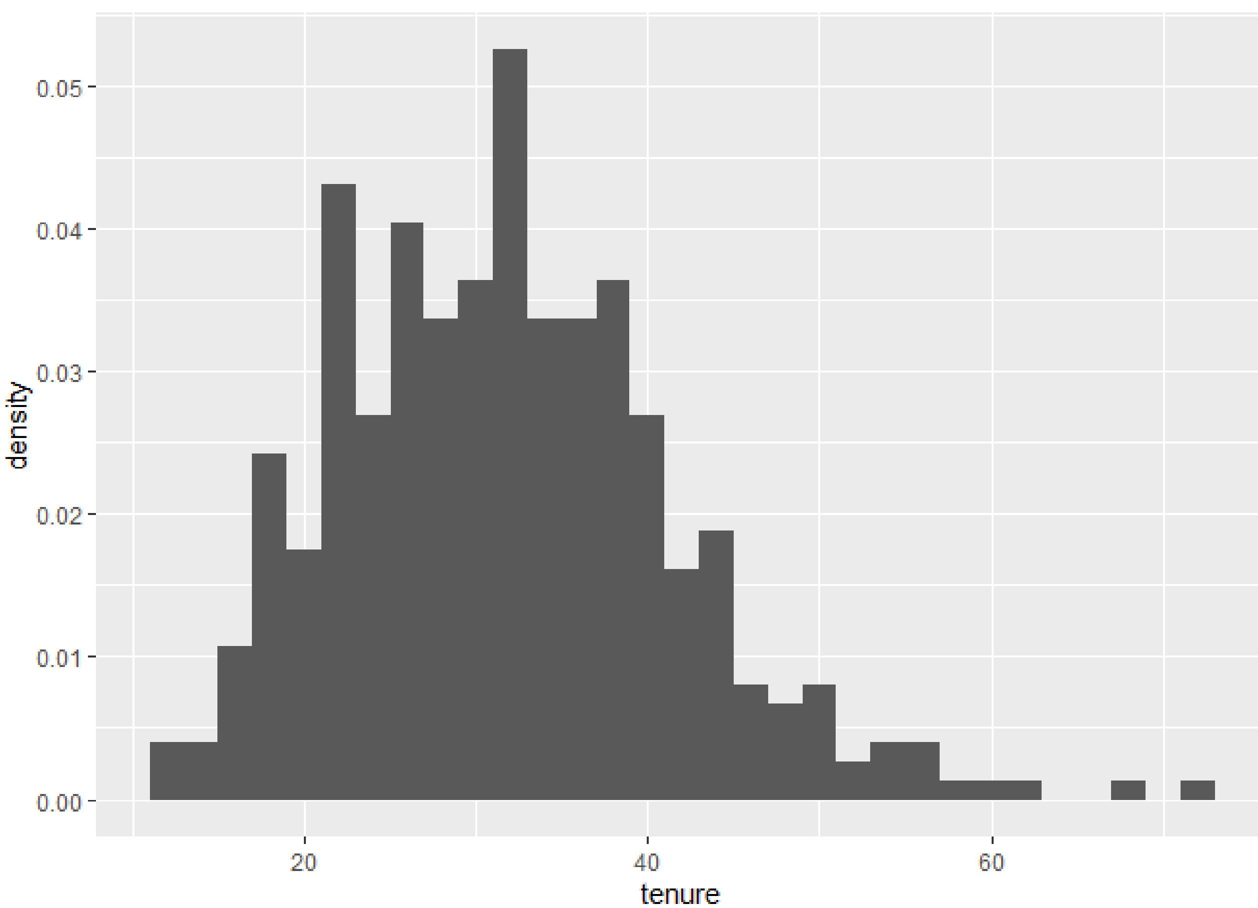
```
snowshoes <- read_table("snowshoe.txt")
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   quantity = col_double(),
##   tenure = col_double(),
##   age = col_double(),
##   product = col_character(),
##   snowshoes = col_double()
## )
```

```
snowshoes <- as_tibble(snowshoes)
snowshoes$product <- as.factor(snowshoes$product)
```

Complete your exploratory data analysis (EDA) in this section. You may use multiple code chunks, if you wish, to organize your code.

```
snowshoes %>%
  ggplot() +
  geom_histogram(mapping = aes(x = tenure, y = ..density..),
                 binwidth = 2)
```



Perform variable selection in this section. You may use multiple code chunks, if you wish, to organize your code.

```r
snowshoes <- snowshoes %>%
  dplyr::select(quantity, age, product, snowshoes, tenure)
snowshoes_best_subsets_bic <- bestglm(as.data.frame(snowshoes),
                                      IC = "BIC",
                                      method = "exhaustive",
                                      TopModels = 1,
                                      family = poisson(link = 'log'))
```

```
## Morgan-Tatar search since family is non-gaussian.
```

```
## Note: factors present with more than 2 levels.
```

```r
BIC(snowshoes_best_subsets_bic$BestModel)
```

```
## [1] 2856.835
```

```r
summary(snowshoes_best_subsets_bic$BestModel)
```

```
##
## Call:
## glm(formula = y ~ ., family = family, data = Xi, weights = weights)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.8712  -1.0869  -0.1285   0.8581   5.3750
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) 3.049584   0.033732  90.405  < 2e-16 ***
## quantity    0.018828   0.002174   8.660  < 2e-16 ***
## age         0.005110   0.000670   7.626 2.42e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 1078.48  on 370  degrees of freedom
## Residual deviance:  884.01  on 368  degrees of freedom
## AIC: 2845.1
```

```
##
## Number of Fisher Scoring iterations: 4
```

Fit a model using the variables you selected from the prevous section, and determine in any interaction(s) are needed for this model in this section. You may use multiple code chunks, if you wish, to organize your code.

```
snowshoes_poisson <- glm(tenure ~ quantity + age,
                         data = snowshoes,
                         family = poisson(link = 'log'))
snowshoes_poisson_int <- glm(tenure ~ quantity * age,
                         data = snowshoes,
                         family = poisson(link = 'log'))
#the model with the interaction term does better (has a low pvalue)
anova(snowshoes_poisson, snowshoes_poisson_int,
      test = 'Chisq')
```

```
## Analysis of Deviance Table
##
## Model 1: tenure ~ quantity + age
## Model 2: tenure ~ quantity * age
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1       368     884.01
## 2       367     876.67  1   7.3421 0.006736 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

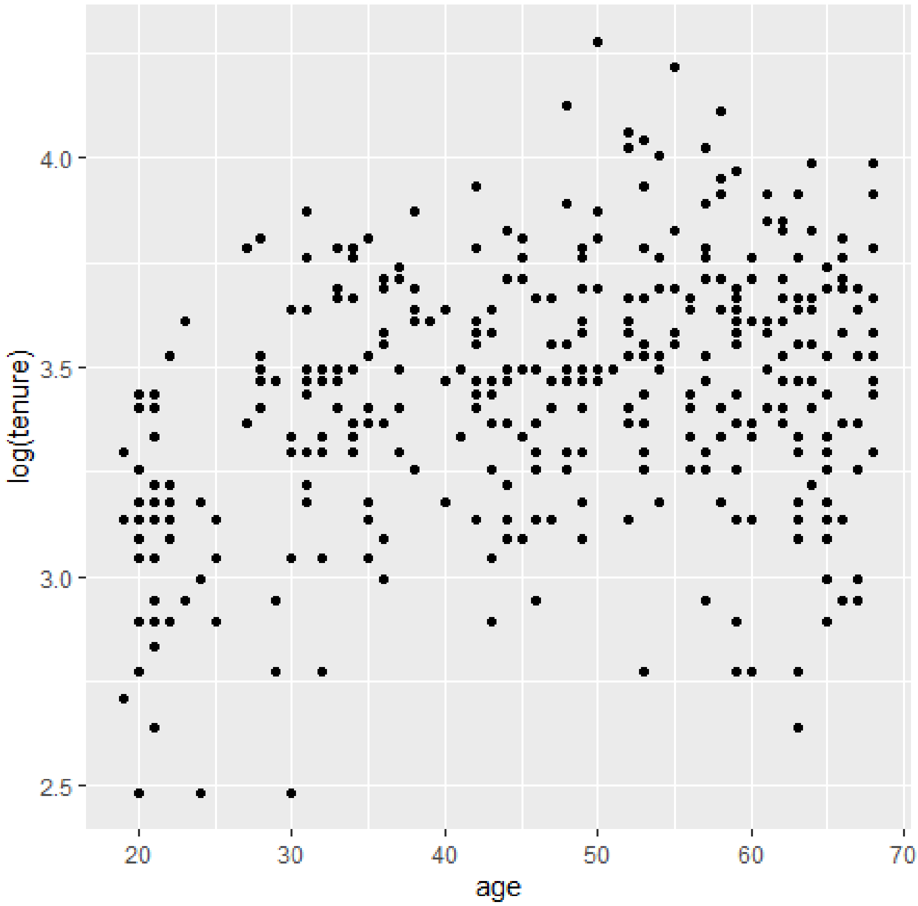
```
summary(snowshoes_poisson_int)
```

```
##
## Call:
## glm(formula = tenure ~ quantity * age, family = poisson(link = "log"),
##     data = snowshoes)
##
## Deviance Residuals:
```

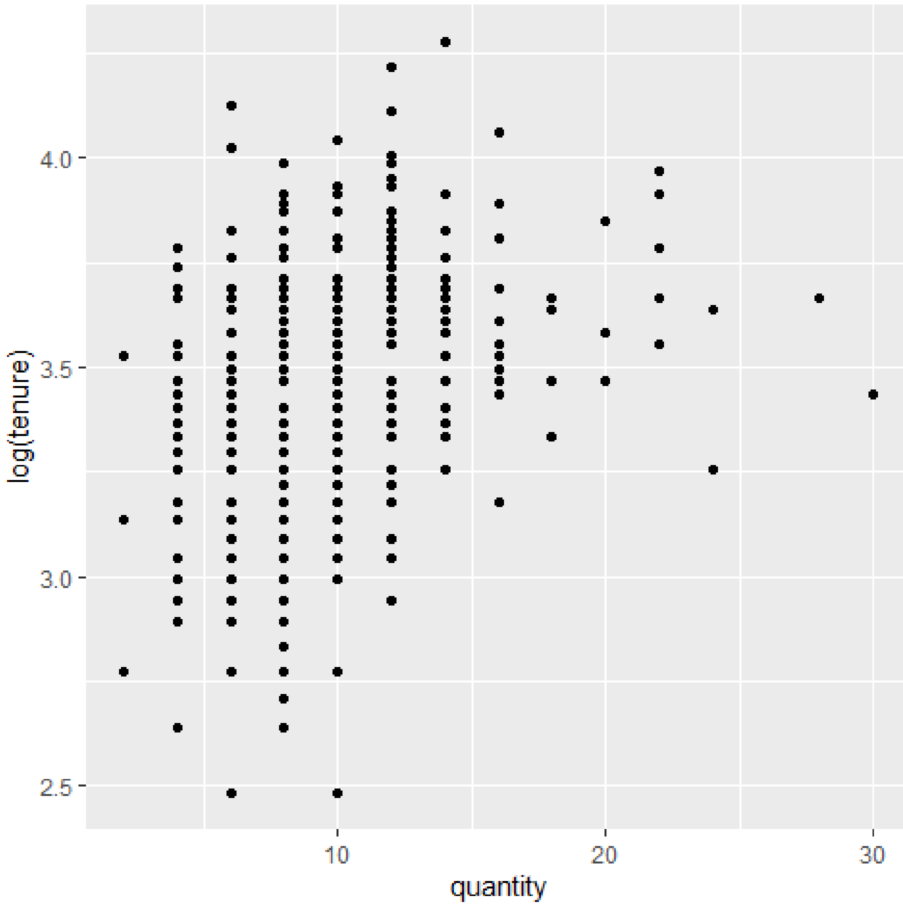
```
##     Min       1Q   Median       3Q      Max
## -3.9231  -1.0386  -0.1259   0.8513   5.3196
##
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)   2.8652906  0.0761152  37.644  < 2e-16 ***
## quantity      0.0415927  0.0086370   4.816 1.47e-06 ***
## age           0.0089309  0.0015649   5.707 1.15e-08 ***
## quantity:age -0.0004533  0.0001670  -2.714  0.00664 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 1078.48  on 370  degrees of freedom
## Residual deviance:  876.67  on 367  degrees of freedom
## AIC: 2839.7
##
## Number of Fisher Scoring iterations: 4
```

Based on your results above, fit an appropriate ("final") model and check model assumptions in this section. You may use multiple code chunks, if you wish, to organize your code.

```
snowshoes %>%
ggplot(mapping = aes(x = age, y = log(tenure))) +
geom_point() +
theme(aspect.ratio = 1)
```




```
snowshoes %>%
ggplot(mapping = aes(x = quantity, y = log(tenure))) +
geom_point() +
theme(aspect.ratio = 1)
```




```
mean(snowshoes$tenure)
```

```
## [1] 32.22372
```

```
var(snowshoes$tenure)
```

```
## [1] 95.35792
```

```
snowshoes_quasipoisson_int <- glm(tenure ~ quantity * age,
                                  data = snowshoes,
                                  family = quasipoisson(link = 'log'))
summary(snowshoes_quasipoisson_int)
```

```
##
## Call:
## glm(formula = tenure ~ quantity * age, family = quasipoisson(link = "log"),
```

```
##     data = snowshoes)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.9231  -1.0386  -0.1259   0.8513   5.3196
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.8652906  0.1187040  24.138  < 2e-16 ***
## quantity      0.0415927  0.0134697   3.088  0.00217 **
## age           0.0089309  0.0024405   3.660  0.00029 ***
## quantity:age -0.0004533  0.0002605  -1.740  0.08263 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for quasipoisson family taken to be 2.432135)
##
##     Null deviance: 1078.48  on 370  degrees of freedom
## Residual deviance:  876.67  on 367  degrees of freedom
## AIC: NA
##
## Number of Fisher Scoring iterations: 4
```

```
pchisq(q = snowshoes_poisson_int$deviance, df = snowshoes_poisson_int$df.residual
       lower.tail = FALSE)
```

```
## [1] 1.106365e-43
```

Complete statistical inference based on the best model you chose in this section. You may use multiple code chunks, if you wish, to organize your code.

```
lr <- snowshoes_poisson_int$null.deviance - snowshoes_poisson_int$deviance
print(lr)
```

```
## [1] 201.8039
```

```
pchisq(lr, df = length(snowshoes_poisson_int$coefficients)-1,
       lower.tail = F)
```

```
## [1] 1.719378e-43
```

```
new_customer <- data.frame(age = 52,
                           quantity = 10)
pred <- predict(snowshoes_poisson_int, newdata = new_customer,
                se.fit = T)
pred$se.fit * qnorm(p = .975)
```

```
## [1] 0.01912317
```