W271 Assignment 1

Due 11:59pm Pacific Time, Sunday October 4, 2020

Amber Chen

1. Confidence Intervals (2 points)

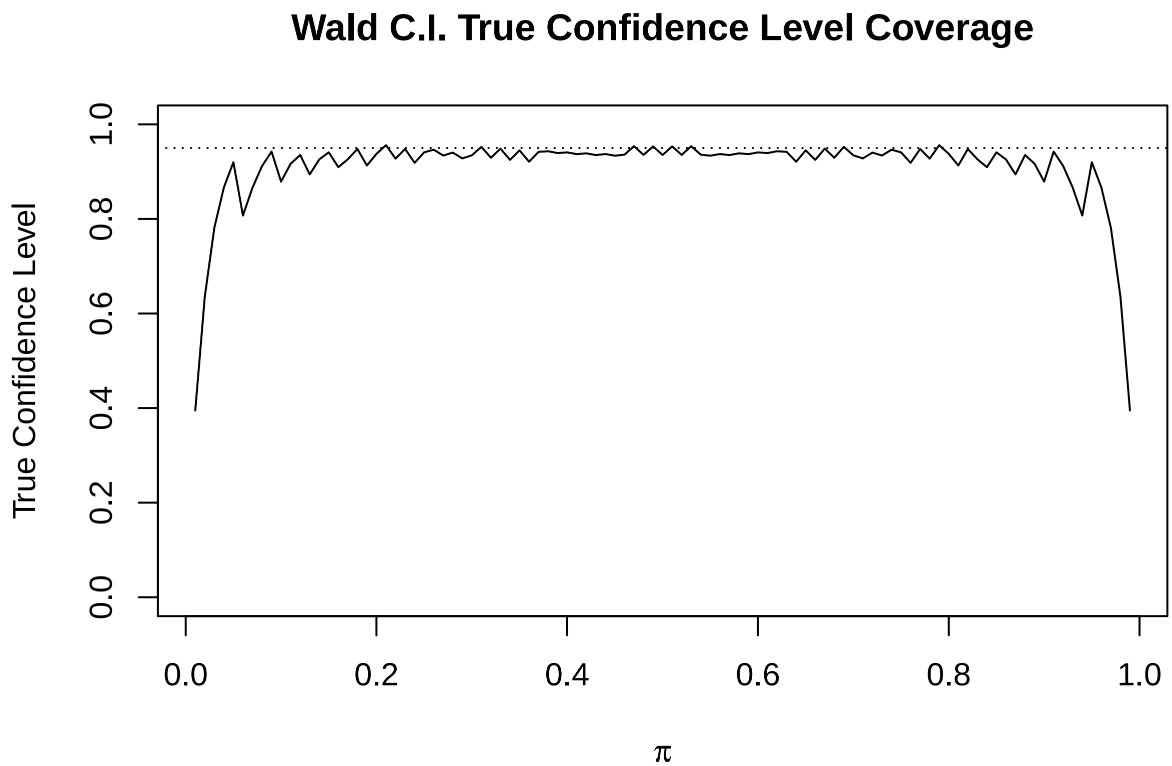
A Wald confidence interval for a binary response probability does not always have the stated confidence level, $1 - \alpha$, where α (the probability of rejecting the null hypothesis when it is true) is often set to 0.05%. This was demonstrated with code in the week 1 live session file.

Question 1.1: Use the code from the week 1 live session file and: (1) redo the exercise for `n=50, n=100, n=500`, (2) plot the graphs, and (3) describe what you have observed from the results. Use the same `pi.seq` as in the live session code.

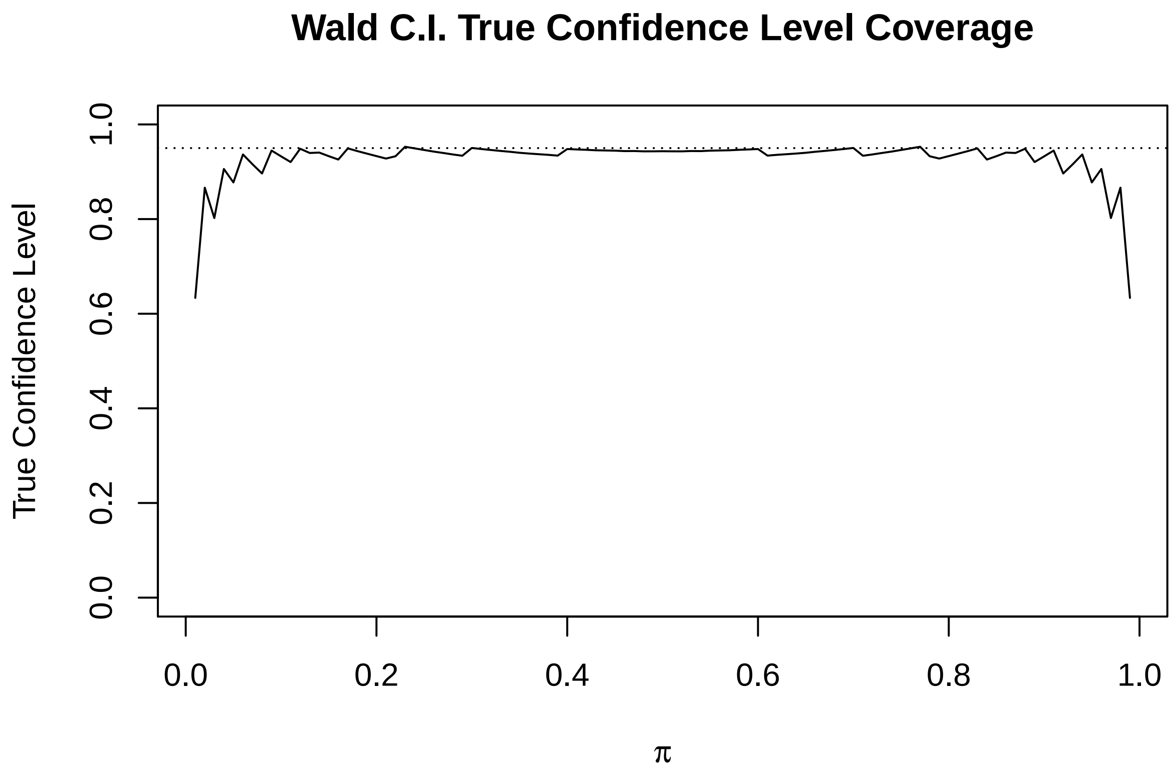
```
pi = 0.6
alpha = 0.05
n.list = c(50, 100, 500)

wald.CI.true.coverage = function(pi, alpha = 0.05, n) {

    w = 0:n

    pi.hat = w/n
    pmf = dbinom(x = w, size = n, prob = pi)

    var.wald = pi.hat * (1 - pi.hat)/n
    wald.CI_lower.bound = pi.hat - qnorm(p = 1 - alpha/2) * sqrt(var.wald)
    wald.CI_upper.bound = pi.hat + qnorm(p = 1 - alpha/2) * sqrt(var.wald)

    covered.pi = ifelse(test = pi > wald.CI_lower.bound, yes = ifelse(test = pi <
        wald.CI_upper.bound, yes = 1, no = 0), no = 0)

    wald.CI.true.coverage = sum(covered.pi * pmf)

    wald.df = data.frame(w, pi.hat, round(data.frame(pmf, wald.CI_lower.bound,
        wald.CI_upper.bound), 4), covered.pi)

    return(wald.df)
}

for (n in n.list) {
```

```
    w = 0:n
    wald.df = wald.CI.true.coverage(pi = pi, alpha = 0.05, n = n)
    wald.CI.true.coverage.level = sum(wald.df$covered.pi * wald.df$pmf)

    # Let's compute the ture coverage for a sequence of pi
    pi.seq = seq(0.01, 0.99, by = 0.01)
    wald.CI.true.matrix = matrix(data = NA, nrow = length(pi.seq),
        ncol = 2)
    counter = 1
    for (pi in pi.seq) {
        wald.df2 = wald.CI.true.coverage(pi = pi, alpha = 0.05,
            n = n)
        wald.CI.true.matrix[counter, ] = c(pi, sum(wald.df2$covered.pi *
            wald.df2$pmf))
        counter = counter + 1
    }
    str(wald.CI.true.matrix)
    wald.CI.true.matrix[1:5, ]

    # Plot the true coverage level (for given n and alpha)
    plot(x = wald.CI.true.matrix[, 1], y = wald.CI.true.matrix[,
        2], ylim = c(0, 1), main = "Wald C.I. True Confidence Level Coverage",
        xlab = expression(pi), ylab = "True Confidence Level",
        type = "l")
    abline(h = 1 - alpha, lty = "dotted")
}
```

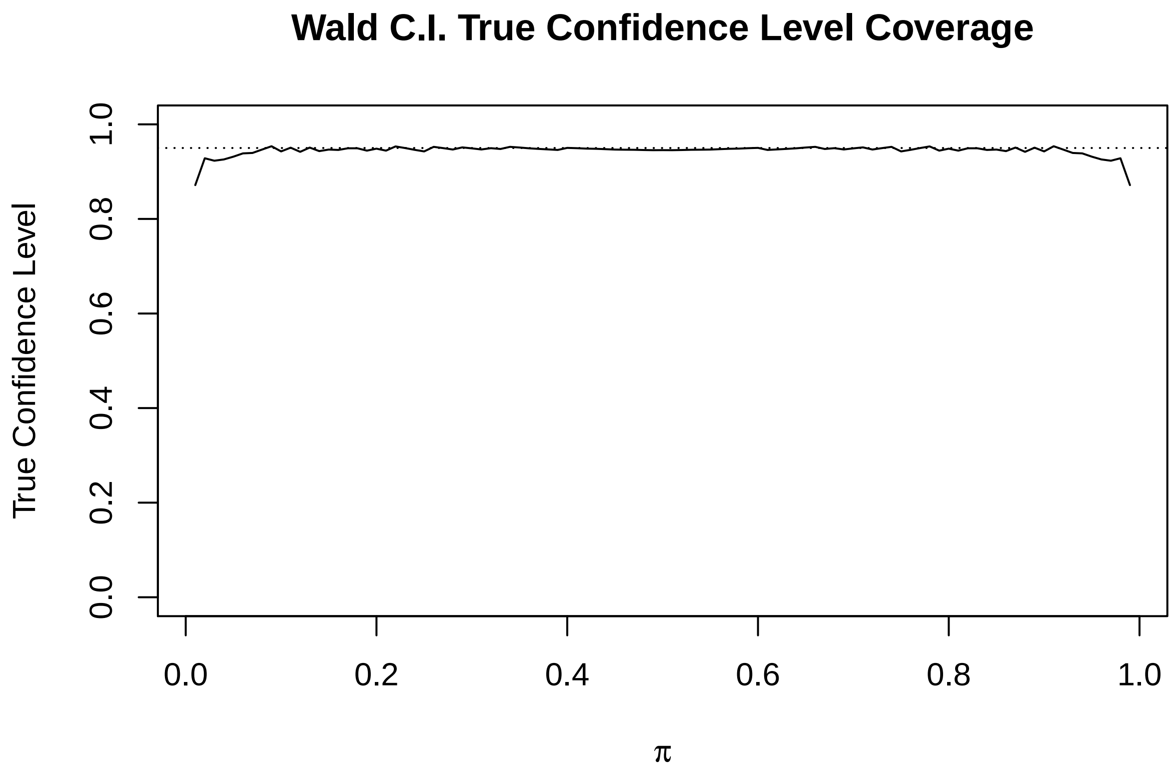
```
##  num [1:99, 1:2] 0.01 0.02 0.03 0.04 0.05 0.06 0.07 0.08 0.09 0.1 ...
```




```
##  num [1:99, 1:2] 0.01 0.02 0.03 0.04 0.05 0.06 0.07 0.08 0.09 0.1 ...
```




```
##  num [1:99, 1:2] 0.01 0.02 0.03 0.04 0.05 0.06 0.07 0.08 0.09 0.1 ...
```



Observations from the three graphs - Because of $\sqrt{\hat{\pi}(1-\hat{\pi})/n}$ in Wald Confidence Interval calculation, the lower and upper limits of the approximation are exactly $\hat{\pi}$ when w = 0 or 1. - Also Wald CI limits are symmetric with respect to $\pi = 0.5$ given $\sqrt{\hat{\pi}(1-\hat{\pi})/n}$ is a symmetric function - The true CI level has a greater coverage, and even equal to $1-\alpha$ when π is closer to 0.5 than when π is away from 0.5 - we can see that a larger sample size, n, does help a better approximation for π. When sample size, n, is small, the true CI coverage is more conservative. As n increases, the coverage approaches to $1-\alpha$ from below. This trend is more obvious when π is near 0 or 1

Question 1.2: (1) Modify the code for the Wilson Interval. (2) Do the exercise for `n=10, n=50, n=100, n=500`. (3) Plot the graphs. (4) Describe what you have observed from the results and compare the Wald and Wilson intervals based on your results. Use the same `pi.seq` as in the live session code.

**Wilson Confidence Interval:

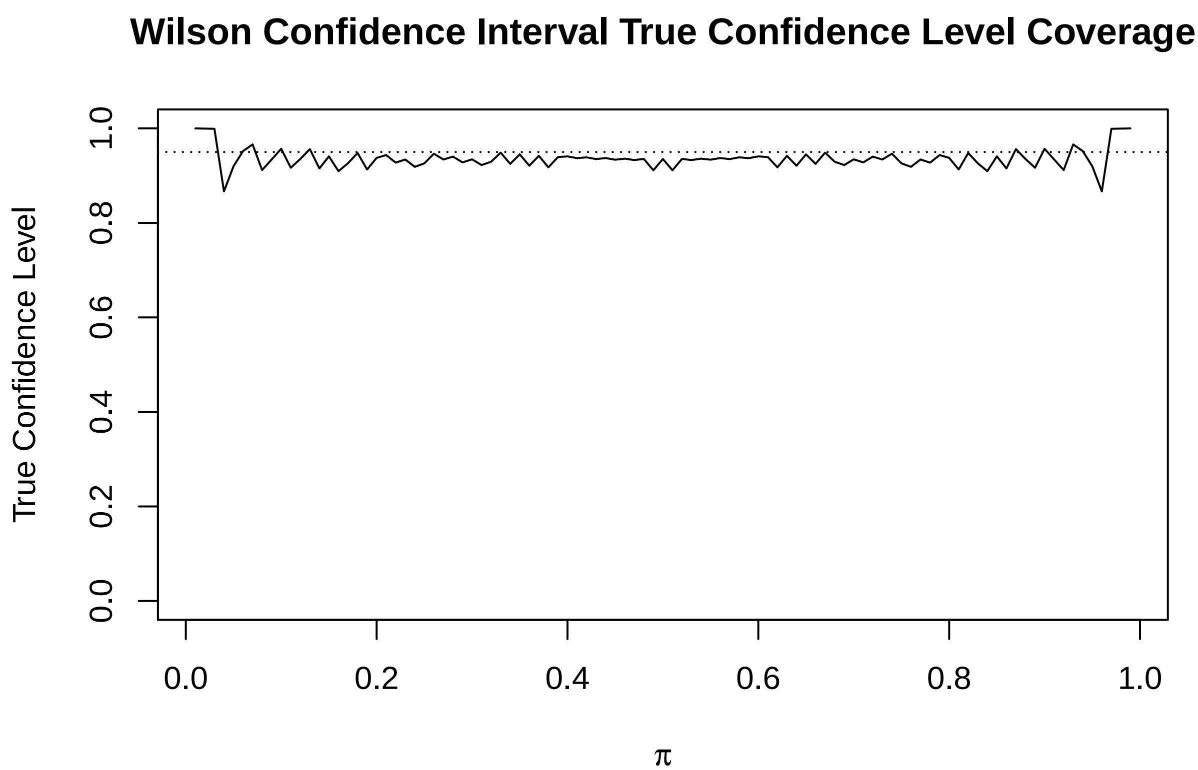
$$\tilde{\pi} \pm \frac{Z_{1-\frac{\alpha}{2}} n^{1/2}}{n + Z^2_{1-\frac{\alpha}{2}}} \sqrt{\hat{\pi}(1-\hat{\pi}) + \frac{Z^2_{1-\frac{\alpha}{2}}}{4n}}$$

```
pi = 0.6
alpha = 0.05
n.list = c(50, 100, 500)

wilson.CI.true.coverage = function(pi, alpha = 0.05, n) {
```

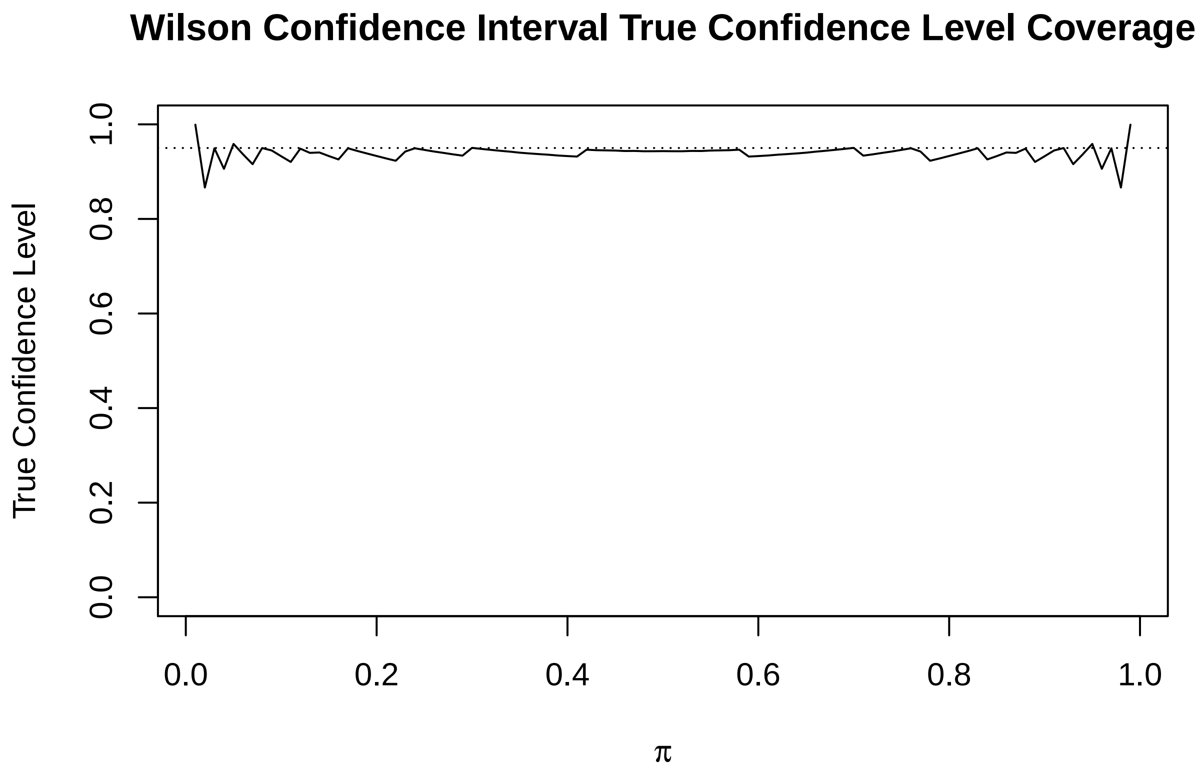
```
    w = 0:n

    pi.hat = w/n
    pmf = dbinom(x = w, size = n, prob = pi)

    var.wilson = pi.hat * (1 - pi.hat) + (qnorm(p = 1 - alpha/2)^2)/(4 *
        n)
    wilson.CI_lower.bound = pi.hat - qnorm(p = 1 - alpha/2) *
        sqrt(n)/(n + qnorm(p = 1 - alpha/2)^2) * sqrt(var.wilson)
    wilson.CI_upper.bound = pi.hat + qnorm(p = 1 - alpha/2) *
        sqrt(n)/(n + qnorm(p = 1 - alpha/2)^2) * sqrt(var.wilson)

    covered.pi = ifelse(test = pi > wilson.CI_lower.bound, yes = ifelse(test = pi <
        wilson.CI_upper.bound, yes = 1, no = 0), no = 0)

    wilson.CI.true.coverage = sum(covered.pi * pmf)

    wilson.df = data.frame(w, pi.hat, round(data.frame(pmf, wilson.CI_lower.bound,
        wilson.CI_upper.bound), 4), covered.pi)

    return(wilson.df)
}

for (n in n.list) {
    w = 0:n
    wilson.df = wilson.CI.true.coverage(pi = pi, alpha = 0.05,
        n = n)
    wilson.CI.true.coverage.level = sum(wilson.df$covered.pi *
        wilson.df$pmf)

    # Let's compute the ture coverage for a sequence of pi
    pi.seq = seq(0.01, 0.99, by = 0.01)
    wilson.CI.true.matrix = matrix(data = NA, nrow = length(pi.seq),
        ncol = 2)
    counter = 1
    for (pi in pi.seq) {
        wilson.df2 = wilson.CI.true.coverage(pi = pi, alpha = 0.05,
            n = n)
        # print(paste('True Coverage is',
        # sum(wald.df2$covered.pi*wald.df2$pmf)))
        wilson.CI.true.matrix[counter, ] = c(pi, sum(wilson.df2$covered.pi *
            wilson.df2$pmf))
        counter = counter + 1
    }
    str(wilson.CI.true.matrix)
    wilson.CI.true.matrix[1:5, ]
```

```
    # Plot the true coverage level (for given n and alpha)
    plot(x = wilson.CI.true.matrix[, 1], y = wilson.CI.true.matrix[,
        2], ylim = c(0, 1), main = "Wilson Confidence Interval True Confidence Level Coverage",
        xlab = expression(pi), ylab = "True Confidence Level",
        type = "l")
    abline(h = 1 - alpha, lty = "dotted")
}
```

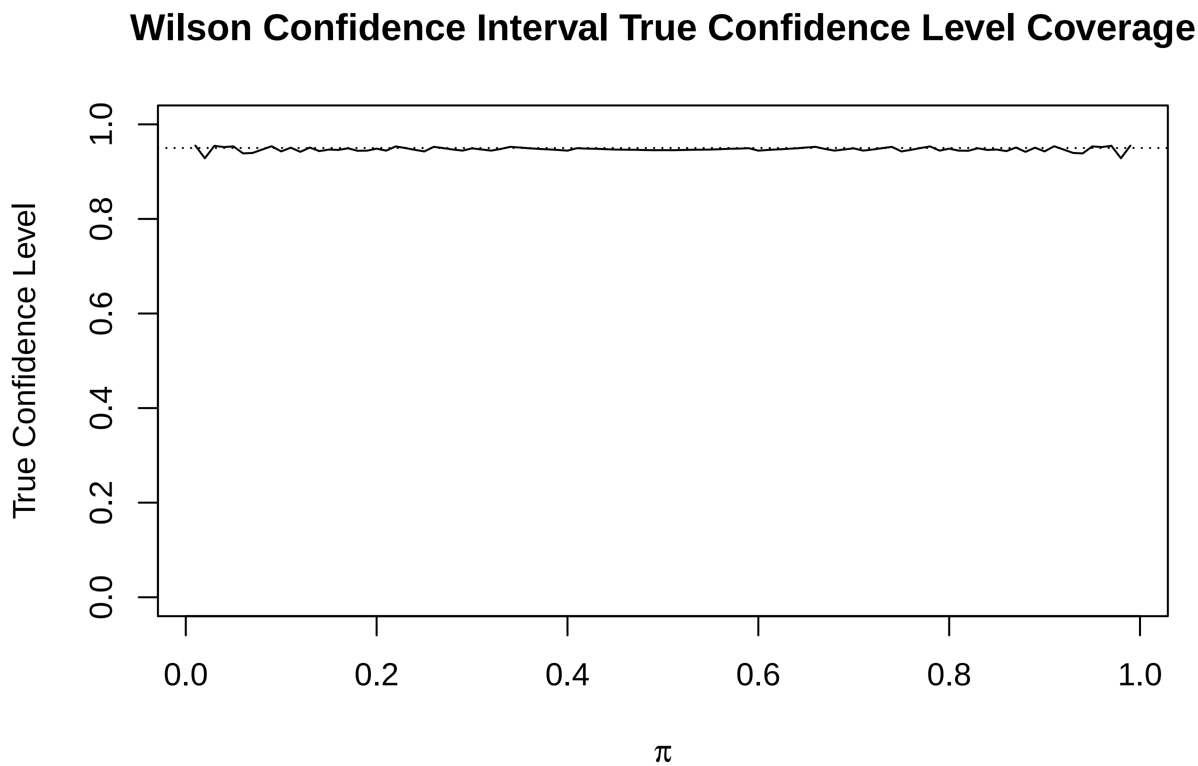
```
##  num [1:99, 1:2] 0.01 0.02 0.03 0.04 0.05 0.06 0.07 0.08 0.09 0.1 ...
```




```
##  num [1:99, 1:2] 0.01 0.02 0.03 0.04 0.05 0.06 0.07 0.08 0.09 0.1 ...
```




```
##  num [1:99, 1:2] 0.01 0.02 0.03 0.04 0.05 0.06 0.07 0.08 0.09 0.1 ...
```



Observations from the above three graphs - Compared to the Wald intervals, the Wilson intervals are closer to $1 - \alpha$ for same sample size. For n = 500, the Wilson interval is almost at 0.95 from $\pi = 0$ to $\pi = 0$, whereas the Wald interval is still a bit far way from 0.95 when π is close to 0 or 1 - The Wilson intervals are aggressive for a small sample size. When n = 50 or 100, we can clearly see the coverage is above $1 - \alpha = 0.95$ when π is close to 0 or 1

2: Binary Logistic Regression (2 points)

Do Exercise 8 a, b, c, and d on page 131 of Bilder and Loughin's textbook. Please write down each of the questions. The dataset for this question is stored in the file *"placekick.BW.csv"* which is provided to you.

In general, all the R codes and datasets used in Bilder and Loughin's book are provided on the book's website: chrisbilder.com

For **question 8b**, in addition to answering the question, re-estimate the model in part (a) using "*Sun*" as the base level category for *Weather*.

Continuing Exercise 7, use the Distance, Weather, Wind15, Temperature, Grass, Pressure, and Ice explanatory variables as linear terms in a new logistic regression model and complete the following:
(a) Estimate the model and properly define the indicator variables used within it.

```r
# read in data set
placekick <- read.csv("placekick.BW.csv")
glimpse(placekick)
```

```
## Rows: 2,003
## Columns: 10
## $ GameNum     <fct> 2002-0101, 2002-0101, 2002-0101, 2002-0101, 2002-0101, ...
## $ Kicker      <fct> Bryant, Bryant, Cortez, Cortez, Cortez, Cortez, Cortez,...
## $ Good        <fct> Y, Y, N, Y, N, Y, Y, Y, Y, N, Y, Y, N, Y, N, N, Y, Y, Y...
## $ Distance    <int> 29, 33, 25, 23, 48, 33, 36, 34, 45, 48, 33, 52, 50, 27,...
## $ Weather     <fct> Sun, Sun, Sun, Sun, Sun, Sun, Sun, Sun, Sun, Sun, Sun, ...
## $ Wind15      <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
## $ Temperature <fct> Nice, Nice, Nice, Nice, Nice, Nice, Nice, Hot, Hot, Hot...
## $ Grass       <int> 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0...
## $ Pressure    <fct> N, N, N, N, N, N, Y, N, N, N, N, N, N, N, N, N, N, N, N...
## $ Ice         <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
```

```r
describe(placekick)
```

```
## placekick
##
##  10  Variables      2003  Observations
## --------------------------------------------------------------------------------
## GameNum
##        n  missing distinct
##     2003        0      523
##
## lowest : 2002-0101 2002-0102 2002-0103 2002-0104 2002-0105
## highest: 2003-P203 2003-P204 2003-P301 2003-P302 2003-P401
## --------------------------------------------------------------------------------
## Kicker
##        n  missing distinct
```

```
##     2003        0       52
##
## lowest : Akers      Andersen   Anderson   Boyd       Brien
## highest: Stover     Tuthill    Vanderjagt Vinatieri  Wilkins
## --------------------------------------------------------------------------------
## Good
##        n  missing distinct
##     2003        0        2
##
## Value          N     Y
## Frequency    438  1565
## Proportion 0.219 0.781
## --------------------------------------------------------------------------------
## Distance
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     2003        0       43    0.999    36.35    11.11       22       23
##      .25      .50      .75      .90      .95
##       28       37       44       49       52
##
## lowest : 18 19 20 21 22, highest: 56 57 58 60 62
## --------------------------------------------------------------------------------
## Weather
##        n  missing distinct
##     2003        0        4
##
## Value        Clouds   Inside SnowRain      Sun
## Frequency       717      385      171      730
## Proportion    0.358    0.192    0.085    0.364
## --------------------------------------------------------------------------------
## Wind15
##        n  missing distinct     Info      Sum     Mean      Gmd
##     2003        0        2    0.343      264   0.1318    0.229
##
## --------------------------------------------------------------------------------
## Temperature
##        n  missing distinct
##     2003        0        3
##
## Value       Cold   Hot  Nice
## Frequency    259   198  1546
## Proportion 0.129 0.099 0.772
## --------------------------------------------------------------------------------
## Grass
##        n  missing distinct     Info      Sum     Mean      Gmd
##     2003        0        2     0.68     1308    0.653   0.4534
##
## --------------------------------------------------------------------------------
## Pressure
```

```
##        n  missing distinct
##     2003        0        2
##
## Value          N     Y
## Frequency   1864   139
## Proportion 0.931 0.069
## ---------------------------------------------------------------------------------
## Ice
##        n  missing distinct     Info      Sum     Mean      Gmd
##     2003        0        2    0.056       38  0.01897  0.03724
##
## ---------------------------------------------------------------------------------
```

```r
# Convert indicator variables to binary variables
placekick$Pressure <- revalue(as.factor(placekick$Pressure),
    c(`0` = "N", `1` = "Y"))
```

```
## The following `from` values were not present in `x`: 0, 1
```

```r
placekick$Good <- revalue(as.factor(placekick$Good), c(`0` = "N",
    `1` = "Y"))
```

```
## The following `from` values were not present in `x`: 0, 1
```

The logistic model is defined as follow

$$logit\left(\pi\right) = log\left(\frac{\pi}{1-\pi}\right) = \beta_0 + \beta_1 Distance + \beta_2 Weather + \beta_3 I(Wind15) + \\ \beta_4 Temperature + \beta_5 I(Grass) + \beta_6 I(Pressure) + \beta_7 I(Ice)$$

```r
logreg.mod = glm(formula = Good ~ Distance + Weather + Wind15 +
    Temperature + Grass + Pressure + Ice, family = binomial,
    data = placekick)

# summarize model
stargazer(logreg.mod, type = "latex", summary = F, dep.var.labels = c("Probability of a Success
    title = "Binary Logistic Regression Model", header = F)
```

The estimated regression is

$$logit\left(\pi\right) = log\left(\frac{\pi}{1-\pi}\right) = 5.7402 - 0.1096 Distance - 0.08303 WeatherInside - 0.4442 WeatherSnowRain - 0.24$$

$$-0.2438 I(Wind15) + 0.2500 TemperatureHot + 0.2349 TemperatureNice - 0.3284 I(Grass) + 0.2702 I(Pressure)$$

(b) The authors use "Sun" as the base level category for Weather, which is not the default level that R uses. Describe how "Sun" can be specified as the base level in R.

Table 1: Binary Logistic Regression Model

	Dependent variable:
	Probability of a Success in Kick
Distance	−0.110***
	(0.007)
WeatherInside	−0.083
	(0.215)
WeatherSnowRain	−0.444**
	(0.218)
WeatherSun	−0.248*
	(0.140)
Wind15	−0.244
	(0.176)
TemperatureHot	0.250
	(0.248)
TemperatureNice	0.235
	(0.181)
Grass	−0.328**
	(0.160)
PressureY	0.270
	(0.263)
Ice	−0.876*
	(0.451)
Constant	5.740***
	(0.370)
Observations	2,003
Log Likelihood	−895.643
Akaike Inf. Crit.	1,813.286
Note:	*p<0.1; **p<0.05; ***p<0.01

```r
levels(as.factor(placekick$Weather))
```

```
## [1] "Clouds"   "Inside"   "SnowRain" "Sun"
```

```r
placekick$SortedWeather = factor(as.factor(placekick$Weather),
    levels = c("Sun", "Clouds", "Inside", "SnowRain"))
levels(placekick$SortedWeather)
```

```
## [1] "Sun"      "Clouds"   "Inside"   "SnowRain"
```

"Sun" can be specified as the base leve by applying factor function with a definition of levels = c("Sun", "Clouds", "Inside", "SnowRain")

(c) Perform LRTs for all explanatory variables to evaluate their importance within the model. Discuss the results.

To test the existence of effect of an explanatory variable on all response categories, we set the hypotheses as follow:

$$H_0 : \beta_{jr} = 0, \quad j = 2, \ldots, J \quad \text{assuming j=1 is the base category}$$
$$H_a : \beta_{jr} \neq 0, \quad \text{for some } j$$

```r
anova(logreg.mod, test = "LR")
```

```
## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: Good
##
## Terms added sequentially (first to last)
##
##
##             Df Deviance Resid. Df Resid. Dev  Pr(>Chi)
## NULL                         2002     2104.0
## Distance     1  287.047      2001     1817.0 < 2.2e-16 ***
## Weather      3   13.424      1998     1803.6  0.003804 **
## Wind15       1    2.090      1997     1801.5  0.148228
## Temperature  2    1.831      1995     1799.7  0.400249
## Grass        1    4.659      1994     1795.0  0.030884 *
## Pressure     1    0.003      1993     1795.0  0.954510
## Ice          1    3.698      1992     1791.3  0.054479 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The LRT results show that Distance, Weather and Grass are statistically significant explanatory variables given their p-values are less than 0.05.

(d) Estimate an appropriate odds ratio for distance, and compute the corresponding confidence interval. Interpret the odds ratio.

The odds ratio for distance is defined as follow:

$$OR = \frac{Odds_{x_1+c}}{Odds_{x_1}} = exp(c\beta_1)$$

```
c = 5
OR = exp(c * logreg.mod$coefficients["Distance"])
OR
```

```
## Distance
## 0.578106
```

```
OR.rev = exp(-c * logreg.mod$coefficients["Distance"])
OR.rev
```

```
## Distance
## 1.729787
```

The odds of a successful kick change by 1.7298 times for every 5 yards derease in distance

```
alpha = 0.05
beta.distance.CI <- confint(object = logreg.mod, parm = "Distance",
    level = 1 - alpha)
```

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

```
beta.distance.CI
```

```
##       2.5 %      97.5 %
## -0.12394589 -0.09575447
```

```
OR.rev.CI = exp(-c * beta.distance.CI)
OR.rev.CI
```

```
##    2.5 %   97.5 %
## 1.858425 1.614092
```

With 95% confidence, the odds of a success change by an amount between 1.8584 and 1.6141 times for every 5 yards decrease in distance.

3: Binary Logistic Regression (2 points)

The dataset *"admissions.csv"* contains a small sample of graduate school admission data from a university. The variables are specificed below:

1. admit - the depenent variable that takes two values: $0, 1$ where 1 denotes *admitted* and 0 denotes *not admitted*
2. gre - GRE score
3. gpa - College GPA
4. rank - rank in college major

Suppose you are hired by the University's Admission Committee and are charged to analyze this data to quantify the effect of GRE, GPA, and college rank on admission probability. We will conduct this analysis by answering the follwing questions:

Question 3.1: Examine the data and conduct EDA

```r
# read in data set
admissions <- read.csv("admissions.csv")
glimpse(admissions)
```

```
## Rows: 400
## Columns: 5
## $ X     <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18...
## $ admit <int> 0, 1, 1, 1, 0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0...
## $ gre   <int> 380, 660, 800, 640, 520, 760, 560, 400, 540, 700, 800, 440, 7...
## $ gpa   <dbl> 3.61, 3.67, 4.00, 3.19, 2.93, 3.00, 2.98, 3.08, 3.39, 3.92, 4...
## $ rank  <int> 3, 3, 1, 4, 4, 2, 1, 2, 3, 2, 4, 1, 1, 2, 1, 3, 4, 3, 2, 1, 3...
```

```r
describe(admissions)
```

```
## admissions
##
##  5  Variables      400  Observations
## ---------------------------------------------------------------------------
## X
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##      400        0      400        1    200.5    133.7    20.95    40.90
##      .25      .50      .75      .90      .95
##   100.75   200.50   300.25   360.10   380.05
##
## lowest :   1   2   3   4   5, highest: 396 397 398 399 400
## ---------------------------------------------------------------------------
## admit
```

```
##        n  missing distinct     Info      Sum     Mean      Gmd 
##      400        0        2     0.65      127   0.3175   0.4345 
## 
## --------------------------------------------------------------------------------
## gre 
##        n  missing distinct     Info     Mean      Gmd      .05      .10 
##      400        0       26    0.997    587.7    131.2      399      440 
##      .25      .50      .75      .90      .95 
##      520      580      660      740      800 
## 
## lowest : 220 300 340 360 380, highest: 720 740 760 780 800
## --------------------------------------------------------------------------------
## gpa 
##        n  missing distinct     Info     Mean      Gmd      .05      .10 
##      400        0      132        1     3.39   0.4351    2.758    2.900 
##      .25      .50      .75      .90      .95 
##    3.130    3.395    3.670    3.940    4.000 
## 
## lowest : 2.26 2.42 2.48 2.52 2.55, highest: 3.95 3.97 3.98 3.99 4.00
## --------------------------------------------------------------------------------
## rank 
##        n  missing distinct     Info     Mean      Gmd 
##      400        0        4     0.91    2.485    1.038 
## 
## Value          1     2     3     4
## Frequency     61   151   121    67
## Proportion 0.152 0.378 0.302 0.168
## --------------------------------------------------------------------------------
```

```r
summary(admissions)
```

```
##        X             admit             gre             gpa       
##  Min.   :  1.0   Min.   :0.0000   Min.   :220.0   Min.   :2.260  
##  1st Qu.:100.8   1st Qu.:0.0000   1st Qu.:520.0   1st Qu.:3.130  
##  Median :200.5   Median :0.0000   Median :580.0   Median :3.395  
##  Mean   :200.5   Mean   :0.3175   Mean   :587.7   Mean   :3.390  
##  3rd Qu.:300.2   3rd Qu.:1.0000   3rd Qu.:660.0   3rd Qu.:3.670  
##  Max.   :400.0   Max.   :1.0000   Max.   :800.0   Max.   :4.000  
##       rank      
##  Min.   :1.000  
##  1st Qu.:2.000  
##  Median :2.000  
##  Mean   :2.485  
##  3rd Qu.:3.000  
##  Max.   :4.000
```

Preliminary EDA using above results:

1. There is no missing value in the dataset

2. The responsible variable of interest, admit, is a binary variable where 1 denotes "admitted" and 0 denotes "not admitted"

3. The dataset includes three explanatory variable:

- student's GRE score
- student's college GPA score
- college rank

Then we explored the variables further with visualization:

- From Figure 1 below and above summary of the data, we can see the GRE scores kind of follows a normal distribution, with mean 580 and median 587. However, a large number of students are concentrated at GRE score = 800

- From Figure 2 below and above summary, we can see the college GPA kind of follows a normal distribution, with mean 3.39 and median 3.395. However, it has a heavy right tail as a large number of students are concentrated near and at 4.0 GPA.

- Figure 3 and 4 show that students who are admitted into graduate school have higher average of GRE scores than student who are not admitted. Though the trend is visually not very significant, we will conduct further analysis in the next section to see if this variable has a statistically significant effect to admissions

- Table 1 shows that students in lower ranked college tend to not get admitted into graduate school.

```r
# Histogram of GRE scores
ggplot(admissions, aes(x = gre)) + geom_histogram(aes(y = ..density..),
    fill = "#0072B2", colour = "black") + ggtitle("GRE Score") +
    theme(plot.title = element_text(lineheight = 1, face = "bold"))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```r
# Histogram of college GPA
ggplot(admissions, aes(x = gpa)) + geom_histogram(aes(y = ..density..),
    fill = "#0072B2", colour = "black") + ggtitle("College GPA") +
    theme(plot.title = element_text(lineheight = 1, face = "bold"))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```r
# Boxplot of GRE by admissions
ggplot(admissions, aes(factor(admit), gre)) + geom_boxplot(aes(fill = factor(admit))) +
    geom_jitter() + ggtitle("GRE by admissions") + theme(plot.title = element_text(lineheight =
    face = "bold"))
```

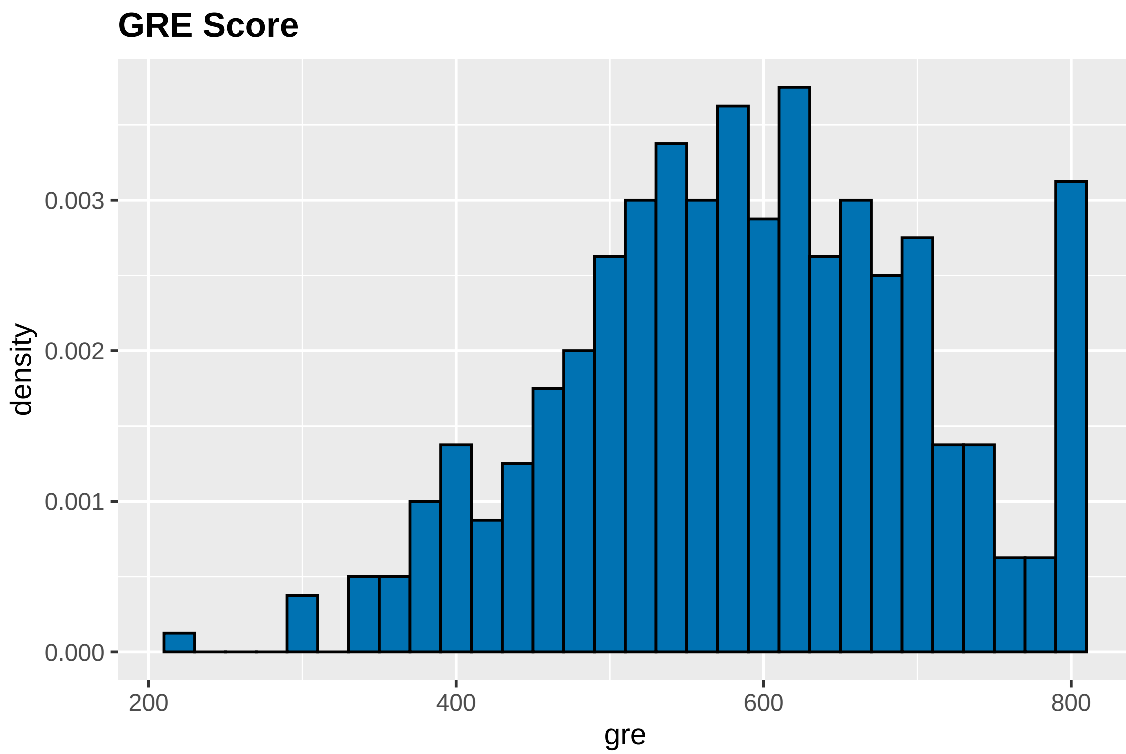


Figure 1: Histogram of GRE Scores

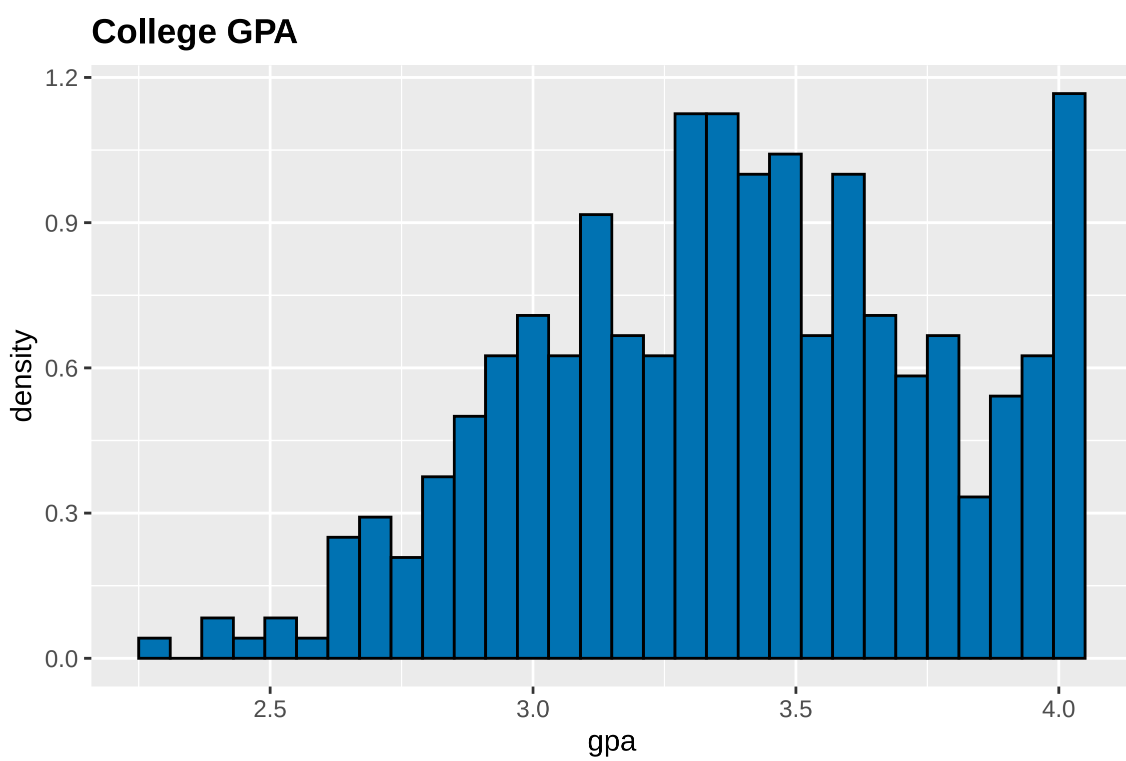


Figure 2: Histogram of college GPA

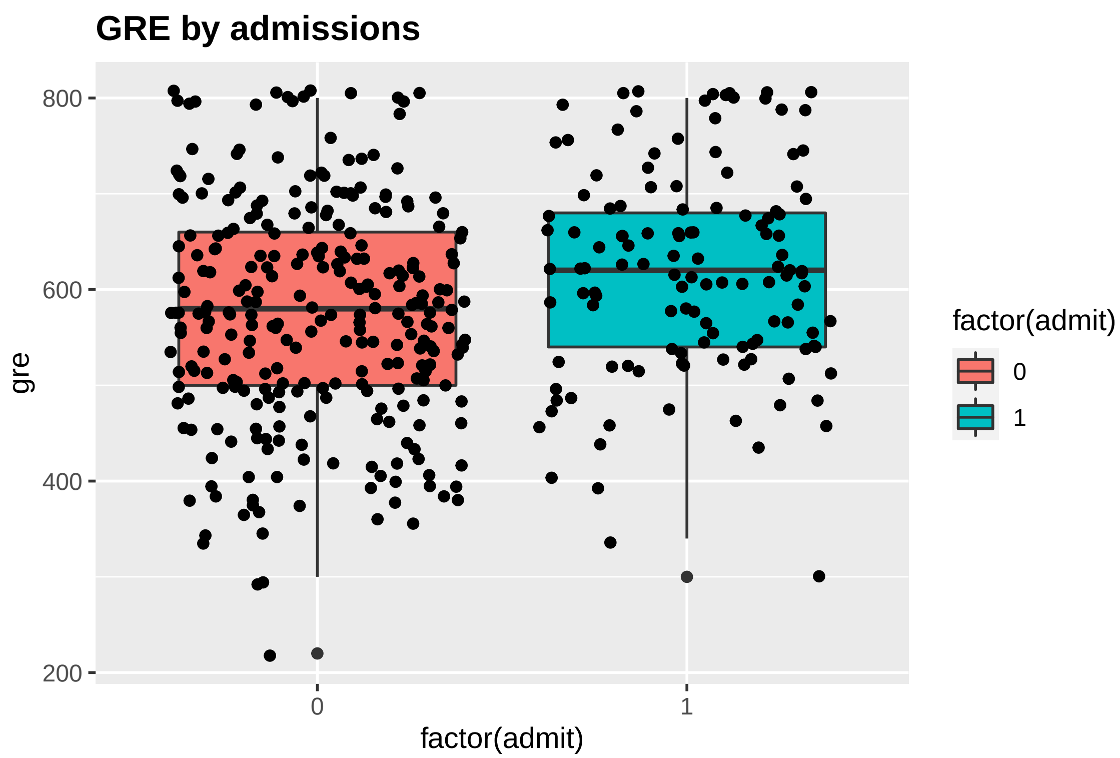


Figure 3: GRE Scores by Admissions Boxplot

```
# GPA by admissions
ggplot(admissions, aes(factor(admit), gpa)) + geom_boxplot(aes(fill = factor(admit))) +
    geom_jitter() + ggtitle("GPA by admissions") + theme(plot.title = element_text(lineheight =
    face = "bold"))
```

```
xtabs(~rank + admit, data = admissions)
```

admit

rank 0 1 1 28 33 2 97 54 3 93 28 4 55 12

```
round(prop.table(xtabs(~rank + admit, data = admissions)), 2)
```

admit

rank 0 1 1 0.07 0.08 2 0.24 0.14 3 0.23 0.07 4 0.14 0.03

Question 3.2: Estimate a binary logistic regression using the following set of explanatory variables: gre, gpa, $rank$, gre^2, gpa^2, and $gre \times gpa$, where $gre \times gpa$ denotes the interaction between gre and gpa variables

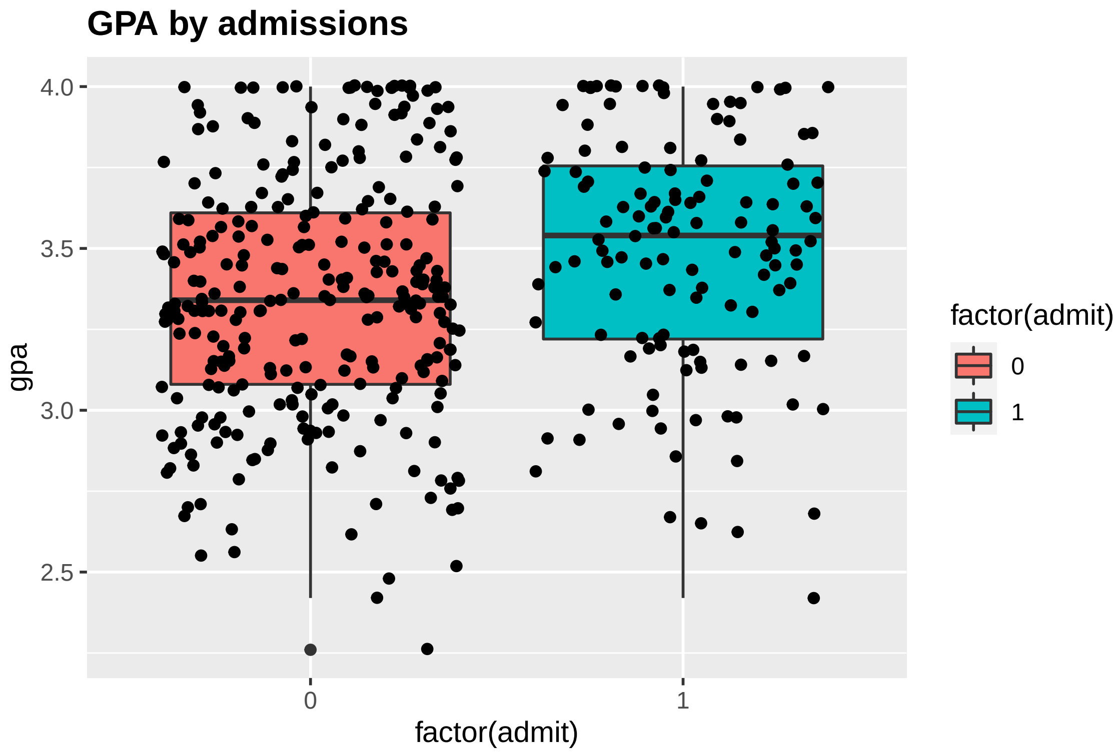


Figure 4: College GPA by Admissions Boxplot

The logistic model is defined as follow

$$logit\,(\pi) = log\left(\frac{\pi}{1-\pi}\right) = \beta_0 + \beta_1 GRE + \beta_2 GPA + \beta_3 Rank + \beta_4 I(GRE^2) + \beta_5 I(GPA^2) + \beta_6 GRE:GPA$$

```
admissions.mod = glm(formula = admit ~ gre + gpa + rank + I(gre^2) +
    I(gpa^2) + gre:gpa, family = "binomial", data = admissions)

# summarize model
stargazer(admissions.mod, type = "latex", summary = F, dep.var.labels = c("Probability of gradu
    title = "Binary Logistic Regression Model", header = F)
```

Based on the results, we get the estimated regression

$$logit\,(\pi) = log\left(\frac{\pi}{1-\pi}\right) = -7.092 + 0.01845GRE - 0.00796GPA - 0.5643Rank + 0.000003495I(GRE^2) + -0.6511I(G$$

Question 3.3: Test the hypothesis that GRE has no effect on admission using the likelihood ratio test

We use LRT for hypothesis testing, and set our hypothesis as follow

$H_0 : \beta_1 = 0$ $H_a : \beta_1 \neq 0$

Table 2: Binary Logistic Regression Model

	Dependent variable:
	Probability of graduate school admissions
gre	0.018
	(0.012)
gpa	−0.008
	(4.933)
rank	−0.564***
	(0.128)
I(gre^2)	0.00000
	(0.00001)
I(gpa^2)	0.651
	(0.761)
gre:gpa	−0.006*
	(0.003)
Constant	−7.092
	(9.024)
Observations	400
Log Likelihood	−227.861
Akaike Inf. Crit.	469.723
Note:	*p<0.1; **p<0.05; ***p<0.01

To do this, we create a reduced model that has all variables from the above model but not the GRE variable. Then we apply anova() on the two models

```
admissions.mod2 = glm(formula = admit ~ gpa + rank + I(gre^2) +
    I(gpa^2) + gre:gpa, family = "binomial", data = admissions)
summary(admissions.mod2)
```

```
##
## Call:
## glm(formula = admit ~ gpa + rank + I(gre^2) + I(gpa^2) + gre:gpa,
##     family = "binomial", data = admissions)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6135  -0.8871  -0.6432   1.1592   2.1619
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.933e+00  8.222e+00  -0.235    0.814
## gpa          1.747e-01  4.871e+00   0.036    0.971
## rank        -5.653e-01  1.275e-01  -4.433 9.28e-06 ***
## I(gre^2)     9.540e-06  6.949e-06   1.373    0.170
## I(gpa^2)     3.379e-01  7.321e-01   0.462    0.644
## gpa:gre     -2.721e-03  2.434e-03  -1.118    0.264
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 499.98  on 399  degrees of freedom
## Residual deviance: 458.39  on 394  degrees of freedom
## AIC: 470.39
##
## Number of Fisher Scoring iterations: 4
```

```
anova(admissions.mod, admissions.mod2, test = "LR")
```

```
## Analysis of Deviance Table
##
## Model 1: admit ~ gre + gpa + rank + I(gre^2) + I(gpa^2) + gre:gpa
## Model 2: admit ~ gpa + rank + I(gre^2) + I(gpa^2) + gre:gpa
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1       393     455.72
## 2       394     458.39 -1  -2.6669   0.1025
```

Based on the result, we cannot reject the null hypothesis as the p-value is 0.1025. Therefore GRE is not statistically significant.

Question 3.4: What is the estimated effect of college GPA on admission?

```
c = 0.1
exp(c * admissions.mod$coefficients["gpa"])
```

```
##       gpa
## 0.9992044
```

The estimated effect of college GPA on admission is 0.999. The odds of a success in graduate school admission change by 0.999 times for every 0.1 increase in GPA.

Question 3.5: Construct the confidence interval for the admission probability for the students with $GPA = 3.3$, $GRE = 720$, and $rank = 1$

```
alpha = 0.05

predict.data <- data.frame(gpa = 3.3, gre = 720, rank = 1)

# predict probability of admission where gpa = 3.3, gre =
# 720, rank = 1
linear.pred <- predict(object = admissions.mod, newdata = predict.data,
    type = "link", se = TRUE)
pi.hat <- exp(linear.pred$fit)/(1 + exp(linear.pred$fit))

# Calculate the CI
CI.lin.pred <- linear.pred$fit + qnorm(p = c(alpha/2, 1 - alpha/2)) *
    linear.pred$se
CI.pi <- exp(CI.lin.pred)/(1 + exp(CI.lin.pred))
data.frame(predict.data, pi.hat, lower = CI.pi[1], upper = CI.pi[2])
```

```
##   gpa gre rank    pi.hat     lower     upper
## 1 3.3 720    1 0.5692897 0.4366982 0.6926379
```

Therefore, the confidence interval is (0.4366982 , 0.6926379) where gpa = 3.3, gre = 720, rank = 1 and the estimated probability is 0.5692897.

4. Binary Logistic Regression (2 points)

Load the `Mroz` data set that comes with the *car* library (this data set is used in the week 2 live session file).

Question 4.1: Estimate a linear probability model using the same specification as in the binary logistic regression model estimated in the week 2 live session. Interpret the model results. Conduct model diagnostics. Test the CLM model assumptions.

```r
# load Mroz data
data(Mroz)
str(Mroz)
```

```
## 'data.frame':    753 obs. of  8 variables:
##  $ lfp : Factor w/ 2 levels "no","yes": 2 2 2 2 2 2 2 2 2 2 ...
##  $ k5  : int  1 0 1 0 1 0 0 0 0 0 ...
##  $ k618: int  0 2 3 3 2 0 2 0 2 2 ...
##  $ age : int  32 30 35 34 31 54 37 54 48 39 ...
##  $ wc  : Factor w/ 2 levels "no","yes": 1 1 1 1 2 1 2 1 1 1 ...
##  $ hc  : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
##  $ lwg : num  1.2102 0.3285 1.5141 0.0921 1.5243 ...
##  $ inc : num  10.9 19.5 12 6.8 20.1 ...
```

```r
describe(Mroz)
```

```
## Mroz
##
##  8  Variables      753  Observations
## --------------------------------------------------------------------------------
## lfp
##        n  missing distinct
##      753        0        2
##
## Value         no   yes
## Frequency    325   428
## Proportion 0.432 0.568
## --------------------------------------------------------------------------------
## k5
##        n  missing distinct     Info     Mean      Gmd
##      753        0        4    0.475   0.2377   0.3967
##
## Value          0     1     2     3
## Frequency    606   118    26     3
## Proportion 0.805 0.157 0.035 0.004
## --------------------------------------------------------------------------------
## k618
##        n  missing distinct     Info     Mean      Gmd
```

```
##      753        0        9    0.932    1.353     1.42
##
## lowest : 0 1 2 3 4, highest: 4 5 6 7 8
##
## Value          0     1     2     3     4     5     6     7     8
## Frequency    258   185   162   103    30    12     1     1     1
## Proportion 0.343 0.246 0.215 0.137 0.040 0.016 0.001 0.001 0.001
## ---------------------------------------------------------------------------------
## age
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##      753        0       31    0.999    42.54    9.289     30.6     32.0
##      .25      .50      .75      .90      .95
##     36.0     43.0     49.0     54.0     56.0
##
## lowest : 30 31 32 33 34, highest: 56 57 58 59 60
## ---------------------------------------------------------------------------------
## wc
##        n  missing distinct
##      753        0        2
##
## Value         no   yes
## Frequency    541   212
## Proportion 0.718 0.282
## ---------------------------------------------------------------------------------
## hc
##        n  missing distinct
##      753        0        2
##
## Value         no   yes
## Frequency    458   295
## Proportion 0.608 0.392
## ---------------------------------------------------------------------------------
## lwg
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##      753        0      676        1    1.097   0.6151   0.2166   0.4984
##      .25      .50      .75      .90      .95
##   0.8181   1.0684   1.3997   1.7600   2.0753
##
## lowest : -2.054124 -1.822531 -1.766441 -1.543298 -1.029619
## highest:  2.905078  3.064725  3.113515  3.155581  3.218876
## ---------------------------------------------------------------------------------
## inc
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##      753        0      621        1    20.13    11.55    7.048    9.026
##      .25      .50      .75      .90      .95
##   13.025   17.700   24.466   32.697   40.920
##
## lowest : -0.029  1.200  1.500  2.134  2.200, highest: 77.000 79.800 88.000 91.000 96.000
```

```
## ----------------------------------------------------------------------------
```

```r
# convert lfp to a binary variable with 1 = 'yes' and 0 =
# 'no'
Mroz$lfp.binary <- ifelse(Mroz$lfp == "yes", 1, 0)

# Estimate the linear probability model
mroz.lm <- lm(lfp.binary ~ k5 + k618 + age + wc + hc + lwg +
    inc, data = Mroz)

# summarize model
stargazer(mroz.lm, type = "latex", summary = F, dep.var.labels = c("U.S Women's Labor-Force Pa
    title = "Linear Regression Model", header = F)
```

Based on Table 3, we get the estimated linear model:

$$y = 1.1435 - 0.294836K5 - 0.011215K618 - 0.012741Age + 0.163679WCyes+$$
$$0.018951HCyes + 0.12274LWG - 0.00676Inc$$

The p-values of K618 and HC are greater than 0.05, so the two variables are not statistically significant. The rest of the variables are statistically significant as their p-values are less than 0.05. The R^2 numbers indicate a fairly poor fitness-of-fit for the model.

CLM Assumptions

Assumption 1: Linear in parameters

In this model, each of our β coefficients have linear relationships to the explanatory variables. Additionally, we have not constrained the error term in any way, so this assumption is met.

Assumption 2: Random sampling The data includes samples of married women in U.S. from the Panel Study of Income Dynamics (PSID). The data is collected by Panel Study of Income Dynamics (PSID) and the methodology published on their website seems to be a fairly representation of the population. However, we don't know whether they applied weights to the sample, so we cannot conclude that the data points are independently and identically distributed.

Assumption 3: No perfect collinearity Since the model has only one independent variable, $Temperature$, there is no violation of multi-collinearity assumption.

```r
mroz.fmla = as.formula("lfp.binary ~ k5 + k618 + age+ wc + hc + lwg + inc")
mroz.X <- as.matrix(model.matrix(mroz.fmla, data = Mroz))
imcdiag(mroz.X, Mroz$lfp.binary)
```

```
##
## Call:
## imcdiag(x = mroz.X, y = Mroz$lfp.binary)
##
##
## All Individual Multicollinearity Diagnostics Result
##
```

Table 3: Linear Regression Model

	Dependent variable:
	U.S Women's Labor-Force Participation
k5	-0.295^{***}
	(0.036)
k618	-0.011
	(0.014)
age	-0.013^{***}
	(0.003)
wcyes	0.164^{***}
	(0.046)
hcyes	0.019
	(0.043)
lwg	0.123^{***}
	(0.030)
inc	-0.007^{***}
	(0.002)
Constant	1.144^{***}
	(0.127)
Observations	753
R^2	0.150
Adjusted R^2	0.142
Residual Std. Error	0.459 (df = 745)
F Statistic	18.827^{***} (df = 7; 745)
Note:	*p<0.1; **p<0.05; ***p<0.01

```
##                VIF    TOL       Wi       Fi Leamer   CVIF Klein   IND1   IND2
## (Intercept) 1.9998 0.5000 106.4083 124.3097     NA 1.9339     1 0.0047 1.8356
## k5          1.2631 0.7917  27.9986  32.7089 0.8898 1.2214     1 0.0064 0.7647
## k618        1.2122 0.8249  22.5875  26.3875 0.9083 1.1723     1 0.0066 0.6428
## age         1.4979 0.6676  52.9949  61.9104 0.8171 1.4485     1 0.0054 1.2205
## wcyes       1.5183 0.6586  55.1660  64.4468 0.8116 1.4683     1 0.0053 1.2534
## hcyes       1.5407 0.6491  57.5412  67.2215 0.8057 1.4899     1 0.0052 1.2885
## lwg         1.1232 0.8903  13.1095  15.3149 0.9436 1.0861     0 0.0072 0.4027
## inc         1.1921 0.8388  20.4484  23.8884 0.9159 1.1528     1 0.0067 0.5917
##
## 1 --> COLLINEARITY is detected by the test
## 0 --> COLLINEARITY is not detected by the test
##
## k5 , wcyes , coefficient(s) are non-significant may be due to multicollinearity
##
## R-square of y on all x: 0.1503
##
## * use method argument to check which regressors may be the reason of collinearity
## ===================================
```

VIFs for all the variables are just above 1 and less than 2, indicating there is no multicollinearity among the 7 explanatory variables.

Assumption 4: Zero conditional mean Examining the following residuals vs fitted plot, a zero conditional mean would be an approximately flat line on the 0-residual. This model differs greatly from that. We can conclude that the model does not satisfy the assumption of zero conditional mean. The figure clearly shows that the residuals have some negative relationship with estimated response values, so there are factors correlated with either our explanatory or dependent variables that are not controlled for in this model.

```r
# convert to data frame
model_df <- fortify(mroz.lm)

# residuals vs. fitted
ggplot(data = model_df, aes(x = .fitted, y = .resid)) + geom_point(shape = 21,
    size = 3, colour = "black", fill = "grey", alpha = 0.3) +
    geom_smooth(se = F, aes(y = .stdresid), alpha = 0.5, size = 0.5,
        method = "loess", span = 5, formula = y ~ x) + labs(title = "Residuals vs. Fitted",
    x = "Fitted values", y = "Residuals") + theme_classic() +
    theme(plot.title = element_text(hjust = 0.5, face = "bold"),
        plot.subtitle = element_text(hjust = 0.5))
```

Assumption 5: Constant variance in the error term (Homoscedasticity)

Below scale location plot (Fig. 6) shows a trend line sloping up from the left to the middle and then slides down to the right. and the $\sqrt{standardizedresiduals}$ vs fitted values scatter plot shows a “x” trend. This indicates the variance is not constant.

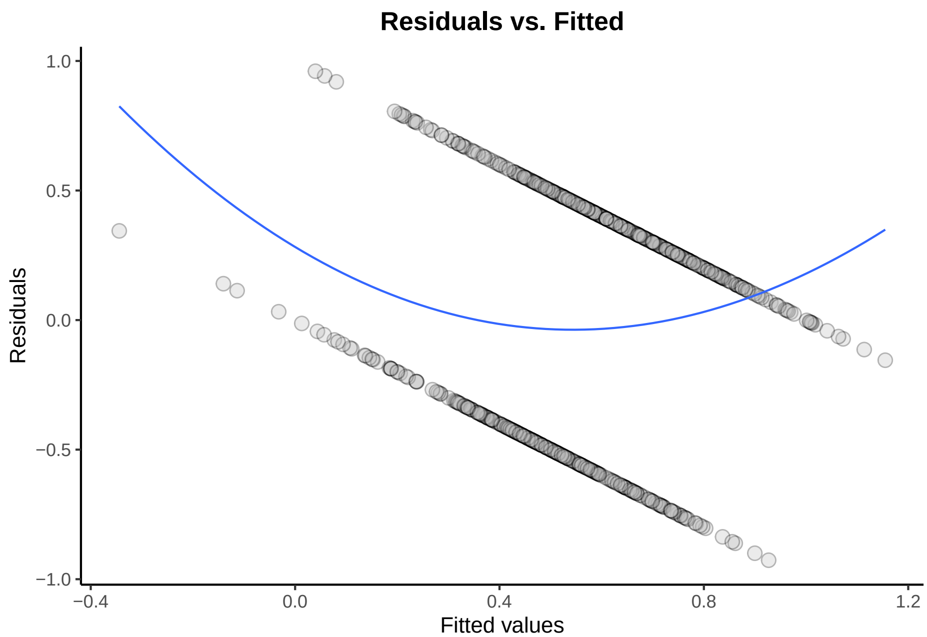


Figure 5: Residuals vs. Fitted

```r
# fitted vs. sqrt of std. resid
ggplot(data = model_df, aes(x = .fitted, y = sqrt(abs(.stdresid)))) +
    geom_point(shape = 21, size = 3, colour = "black", fill = "grey",
        alpha = 0.3) + geom_smooth(se = F, alpha = 0.5, size = 0.5,
    method = "loess", formula = y ~ x) + labs(title = "Scale-Location",
    x = "Fitted values", y = expression(sqrt("Standardized Residuals"))) +
    theme_classic() + theme(plot.title = element_text(hjust = 0.5,
    face = "bold"), plot.subtitle = element_text(hjust = 0.5))
```

```
##
##  studentized Breusch-Pagan test
##
## data:  mroz.lm
## BP = 97.603, df = 7, p-value < 2.2e-16
```

We further conducted a Breusch-Pagan test. With a p-value considerably less than 0.05, we can reject the null hypothesis of homoscedasticity.

Assumption 6: Normal distribution of error terms

The histogram of residuals and the Normal Q-Q plot (Fig. 7) show the residuals do not have a normal distribution and have heavy tails on both side.

```r
# residuals hist
p1 <- ggplot(data = model_df, aes(x = .resid)) + geom_histogram(stat = "bin",
    bins = 20, alpha = 0.3, fill = "blue", color = "black") +
```

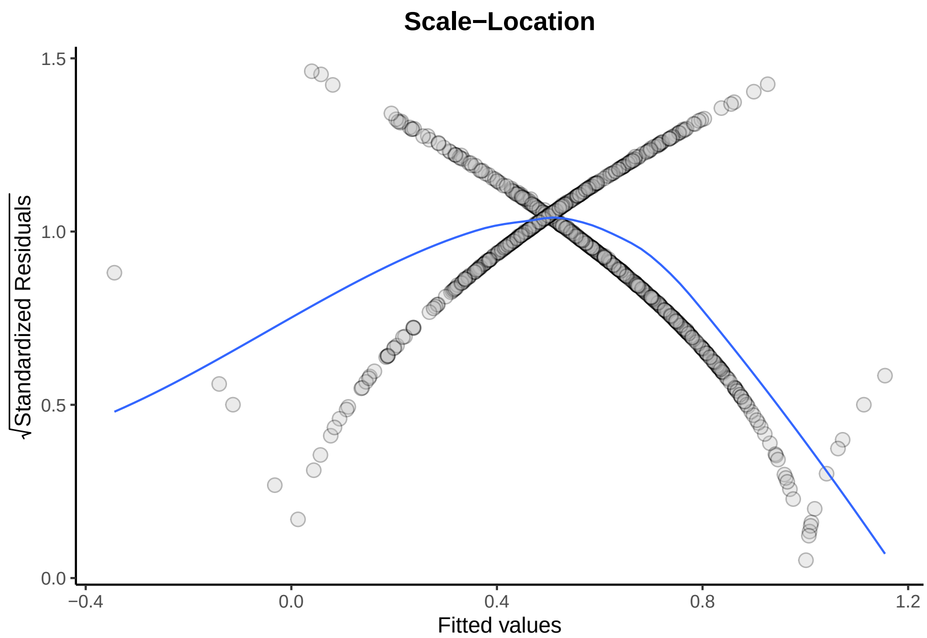


Figure 6: Scale-Location

```
    labs(title = "Histogram of Residuals", x = "Residuals", y = "Frequency") +
    theme_classic() + theme(plot.title = element_text(hjust = 0.5,
    face = "bold"), plot.subtitle = element_text(hjust = 0.5))

# Q-Q plot
p2 <- ggplot(data = model_df, aes(sample = .stdresid)) + geom_qq(shape = 21,
    size = 3, colour = "black", fill = "grey", alpha = 0.3) +
    geom_qq_line(alpha = 0.5, size = 0.5, color = "blue") + labs(title = "Q-Q Plot",
    x = "Theoretical Quantiles", y = "Standardized Residuals") +
    theme_classic() + theme(plot.title = element_text(hjust = 0.5,
    face = "bold"), plot.subtitle = element_text(hjust = 0.5))


grid.arrange(p1, p2, ncol = 2)
```

Question 4.2: Estimate a binary logistic regression with `lfp`, which is a binary variable recoding the participation of the females in the sample, as the dependent variable. The set of explanatory variables includes `age`, `inc`, `wc`, `hc`, `lwg`, `totalKids`, and a quadratic term of `age`, called `age_squared`, where `totalKids` is the total number of children up to age 18 and is equal to the sum of `k5` and `k618`.

We define the binary logistic model:

$$log(\frac{\pi}{1-\pi}) = \beta_0 + \beta_1 Age + \beta_2 Inc + \beta_3 WC + \beta_4 HC + \beta_5 LWG + \beta_6 totalKids + \beta_7 Age^2$$

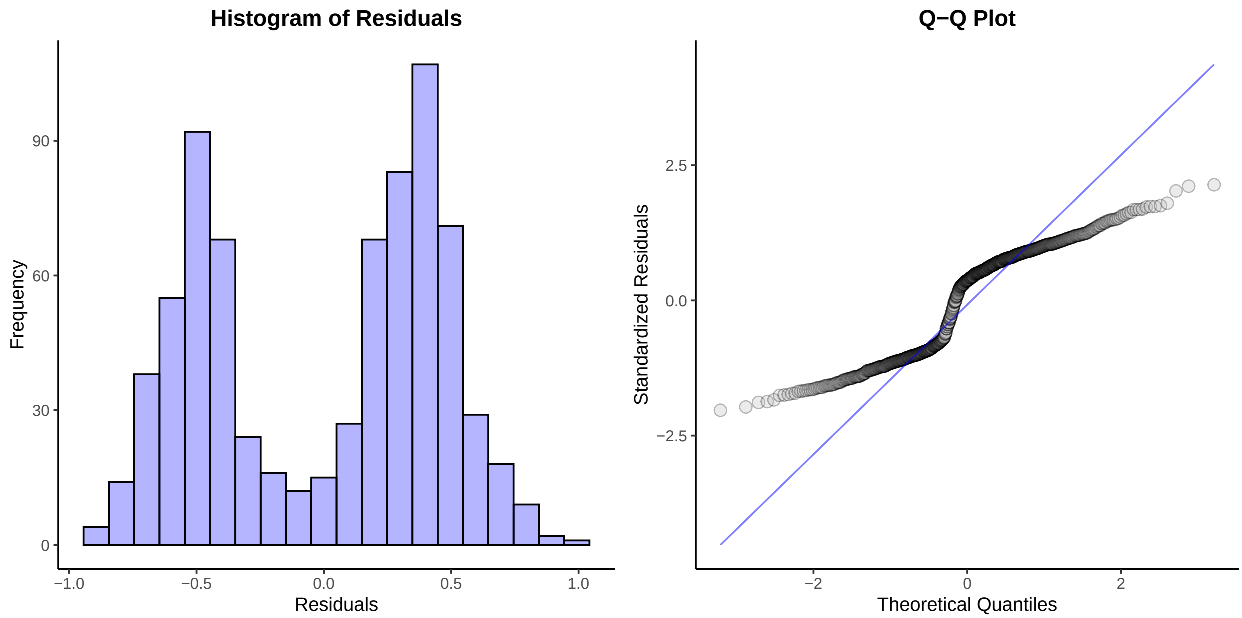


Figure 7: Normality of Errors

```
Mroz$totalKids = Mroz$k5 + Mroz$k618
Mroz$age_squared = Mroz$age^2
mroz.glm <- glm(lfp ~ age + inc + wc + hc + lwg + totalKids +
    age_squared, family = binomial, data = Mroz)

# summarize model
stargazer(mroz.lm, type = "latex", summary = F, dep.var.labels = c("U.S Women's Labor-Force Pa
    title = "Binary Logistic Regression Model", header = F)
```

Based on the results, we get the estimated regression model

$$log(\frac{\pi}{1-\pi}) = -5.294073 + 0.318014Age - 0.034561Inc + 0.666013WC + \\ 0.09826HC + 0.549976LWG - 0.22249totalKids + \beta_7 Age^2$$

Question 4.3: Is the age effect statistically significant? From above result summary table, the variable age has a p-value of less than 0.05. Therefore, the age effect is statistically significant.

Question 4.4: What is the effect of a decrease in age by 5 years on the odds of labor force participation for a female who was 45 years of age.

We derived below OR formula

$$OR = exp(c\beta_1 + c\beta_7(2 \times Age + c))$$

```
c = -5
age = 45
age_effect = exp(c * mroz.glm$coefficient["age"] + c * mroz.glm$coefficient["age_squared"] *
```

Table 4: Binary Logistic Regression Model

	Dependent variable:
	U.S Women's Labor-Force Participation
k5	-0.295^{***}
	(0.036)
k618	-0.011
	(0.014)
age	-0.013^{***}
	(0.003)
wcyes	0.164^{***}
	(0.046)
hcyes	0.019
	(0.043)
lwg	0.123^{***}
	(0.030)
inc	-0.007^{***}
	(0.002)
Constant	1.144^{***}
	(0.127)
Observations	753
R^2	0.150
Adjusted R^2	0.142
Residual Std. Error	0.459 (df = 745)
F Statistic	18.827^{***} (df = 7; 745)
Note:	*p<0.1; **p<0.05; ***p<0.01

```r
    (2 * age + c))
age_effect
```

```
##      age
## 1.171602
```

The odds of a labour force participation for a 45 year-old married woman increase by 1.1716 times for a decrease in age by 5 years.

Question 4.5: Estimate the profile likelihood confidence interval of the probability of labor force participation for females who were 40 years old, had income equal to 20, did not attend college, had log wage equal to 1, and did not have children.

```r
alpha = 0.05

# create a function for generating profile likelihood CI on
# predicted probability
mroz.CI.pi <- function(obj.glm, predict.data, alpha) {
    linear.pred <- predict(object = mroz.glm, newdata = predict.data,
        type = "link", se = TRUE)
    pi.hat <- exp(linear.pred$fit)/(1 + exp(linear.pred$fit))
    CI.lin.pred <- linear.pred$fit + qnorm(p = c(alpha/2, 1 -
        alpha/2)) * linear.pred$se
    CI.pi <- exp(CI.lin.pred)/(1 + exp(CI.lin.pred))
    data.frame(pi.hat = pi.hat, CI.pi.lower = CI.pi[1], CI.pi.upper = CI.pi[2])
}

# Calulcate the CI for females who were 40 years old, had
# income equal to 20, did not attend college, had log wage
# equal to 1, did not have children, and have husband who did
# not attend college
mroz.predict.data1 <- data.frame(age = 40, inc = 20, wc = "no",
    hc = "no", lwg = 1, totalKids = 0, age_squared = 40^2)
data.frame(mroz.predict.data1, mroz.CI.pi(obj.glm = mroz.glm,
    predict.data = mroz.predict.data1, alpha = alpha))
```

```
##   age inc wc hc lwg totalKids age_squared   pi.hat CI.pi.lower CI.pi.upper
## 1  40  20 no no   1         0        1600 0.668822   0.5861286   0.7422584
```

The profile likelihood CI of the probability of labor force participation is (0.5861286, 0.7422584) for females who were 40 years old, had income equal to 20, did not attend college, had log wage equal to 1, did not have children, and have husband who did not attend college

```r
# Use mcprofile() to do the calculation, but yield a
# different result
K <- matrix(data = c(1, 40, 20, 0, 0, 1, 0, 1600), nrow = 1,
```

```r
    ncol = 8)
linear.combo <- mcprofile(object = mroz.glm, CM = K)
confint(object = linear.combo, level = 1 - alpha)
```

```
##
##    mcprofile - Confidence Intervals
##
## level:         0.95
## adjustment:    single-step
##
##    Estimate lower upper
## C1    0.703 0.351  1.06
```

```r
# Calulcate the CI for females who were 40 years old, had
# income equal to 20, did not attend college, had log wage
# equal to 1, did not have children, and have husband who
# attended college
mroz.predict.data2 <- data.frame(age = 40, inc = 20, wc = "no",
    hc = "yes", lwg = 1, totalKids = 0, age_squared = 40^2)
data.frame(mroz.predict.data2, mroz.CI.pi(obj.glm = mroz.glm,
    predict.data = mroz.predict.data2, alpha = alpha))
```

```
##   age inc wc  hc lwg totalKids age_squared    pi.hat CI.pi.lower CI.pi.upper
## 1  40  20 no yes   1         0        1600 0.6902144   0.5849864   0.7788481
```

For females whose husbands did attend college (hc = "yes"), the profile likelihood CI of the probability of labor force participation is (0.5849864, 0.7788481) The results are very similar for hc = "no" and hc = "yes".

```r
# Use mcprofile() to do the calculation, but yield a
# different result
K <- matrix(data = c(1, 40, 20, 0, 1, 1, 0, 1600), nrow = 1,
    ncol = 8)
linear.combo <- mcprofile(object = mroz.glm, CM = K)
confint(object = linear.combo, level = 1 - alpha)
```

```
##
##    mcprofile - Confidence Intervals
##
## level:         0.95
## adjustment:    single-step
##
##    Estimate lower upper
## C1    0.801 0.346  1.26
```

5: Maximum Likelihood (2 points)

Question 18 a and b of Chapter 3 (page 192,193)

For the wheat kernel data (*wheat.csv*), consider a model to estimate the kernel condition using the density explanatory variable as a linear term.

Question 5.1 Write an R function that computes the log-likelihood function for the multinomial regression model. Evaluate the function at the parameter estimates produced by multinom(), and verify that your computed value is the same as that produced by logLik() (use the object saved from multinom() within this function).

$$\sum_{i=1}^{N} healthy * log(pi_{healthy})$$

```
wheat <- read.csv("wheat.csv")

logL <- function(beta, x, Y) {
    # compute pi_1.hat (healthy)
    pi_1 = (1 + exp(x %*% beta[1:7]) + exp(x %*% beta[8:14]))^(-1)

    # compute pi_2.hat (scab)
    pi_2 = exp(x %*% beta[1:7]) * pi_1

    # compute pi_3.hat (sprout)
    pi_3 = exp(x %*% beta[8:14]) * pi_1

    # combine together to form a pi_hat matrix
    pi_hat_mtx = cbind(pi_1, pi_2, pi_3)

    # compute log likelihood
    sum(Y * log(pi_hat_mtx))
}

# Confirm healthy is the base category
levels(wheat$type)
```

```
## [1] "Healthy" "Scab"    "Sprout"
```

```
fmla = as.formula("type ~ class + density + hardness + size + weight + moisture")

# Create a matrix for all explanatory variables
x <- as.matrix(model.matrix(fmla, data = wheat))

wheat.mod <- multinom(formula = fmla, data = wheat)
```

```
## # weights:  24 (14 variable)
```

```
## initial  value 302.118379
## iter  10 value 234.991271
## iter  20 value 192.127549
## final  value 192.112352
## converged
```

```r
summary(wheat.mod)
```

```
## Call:
## multinom(formula = fmla, data = wheat)
##
## Coefficients:
##        (Intercept)   classsrw   density    hardness      size     weight
## Scab      30.54650 -0.6481277 -21.59715 -0.01590741 1.0691139 -0.2896482
## Sprout    19.16857 -0.2247384 -15.11667 -0.02102047 0.8756135 -0.0473169
##           moisture
## Scab    0.10956505
## Sprout -0.04299695
##
## Std. Errors:
##        (Intercept)  classsrw  density    hardness      size     weight
## Scab      4.289865 0.6630948 3.116174 0.010274587 0.7722862 0.06170252
## Sprout    3.767214 0.5009199 2.764306 0.008105748 0.5409317 0.03697493
##          moisture
## Scab   0.1548407
## Sprout 0.1127188
##
## Residual Deviance: 384.2247
## AIC: 412.2247
```

```r
# Create a binary matrix for the response variable
type.healthy = ifelse(test = wheat$type == "Healthy", yes = 1,
    no = 0)
type.scab = ifelse(test = wheat$type == "Scab", yes = 1, no = 0)
type.sprout = ifelse(test = wheat$type == "Sprout", yes = 1,
    no = 0)
type.binary = cbind(type.healthy, type.scab, type.sprout)

beta_hat = c(summary(wheat.mod)$coefficients[1, ], summary(wheat.mod)$coefficients[2,
    ])

logL(beta = beta_hat, x = x, Y = type.binary)
```

```
## [1] -192.1124
```

```r
# LogLik() verifies above computed value
logLik(wheat.mod)
```

```
## 'log Lik.' -192.1124 (df=14)
```

Question 5.2 Maximize the log-likelihood function using optim() to obtain the MLEs and the estimated covariance matrix. Compare your answers to what is obtained by multinom(). Note that to obtain starting values for optim(), one approach is to estimate separate logistic regression models for $log\left(\frac{\pi_2}{\pi_1}\right)$ and $log\left(\frac{\pi_3}{\pi_1}\right)$. These models are estimated only for those observations that have the corresponding responses (e.g., a $Y = 1$ or $Y = 2$ for $log\left(\frac{\pi_2}{\pi_1}\right)$).

```r
# Create binary response variable for the two separate
# logistic regression models
wheat$type.healthy_scab <- ifelse(test = wheat$type != "Sprout",
    yes = 1, no = 0)
wheat$type.healthy_sprout <- ifelse(test = wheat$type != "Scab",
    yes = 1, no = 0)

wheat.mod1 = glm(formula = type.healthy_scab ~ class + density +
    hardness + size + weight + moisture, data = wheat, family = binomial)
wheat.mod2 = glm(formula = type.healthy_sprout ~ class + density +
    hardness + size + weight + moisture, data = wheat, family = binomial)

# use the beta values from above logistic regression model as
# initial beta values
beta.start = c(wheat.mod1$coefficients, wheat.mod2$coefficients)
MLE = optim(beta.start, fn = logL, x = x, Y = type.binary, hessian = TRUE)
MLE
```

```
## $par
## (Intercept)    classsrw     density    hardness        size      weight
##  -1.8462880  -1.1863566  -1.9003612  -1.6025724  -4.9173841   0.2592875
##    moisture (Intercept)    classsrw     density    hardness        size
##  -1.5381006 -27.8409375  -1.0611941   5.0237824   0.7885141  11.8579302
##      weight    moisture
##  14.5747590   2.2866368
##
## $value
## [1] -82835.61
##
## $counts
## function gradient
##      501       NA
##
## $convergence
## [1] 1
```

```
##
## $message
## NULL
##
## $hessian
##               (Intercept)      classsrw       density      hardness
## (Intercept) -1.618901e-03  0.000000e+00 -5.195034e-03  2.910383e-04
## classsrw     0.000000e+00  0.000000e+00  0.000000e+00  1.818989e-06
## density     -5.195034e-03  0.000000e+00 -3.747118e-03  3.550667e-03
## hardness     2.910383e-04  1.818989e-06  3.550667e-03 -6.439222e-04
## size        -8.840289e-04  0.000000e+00 -1.662556e-03 -3.137757e-03
## weight       1.640728e-03  0.000000e+00 -2.495653e-03  3.790774e-03
## moisture     1.244189e-03  0.000000e+00  2.284651e-03 -3.659807e-03
## (Intercept)  1.618901e-03  0.000000e+00  5.196853e-03 -2.892193e-04
## classsrw    -3.637979e-06  0.000000e+00 -1.818989e-06  0.000000e+00
## density      5.198672e-03  0.000000e+00  3.745299e-03 -3.550667e-03
## hardness    -2.910383e-04  0.000000e+00 -3.547029e-03  6.421033e-04
## size         8.840289e-04  0.000000e+00  1.666194e-03  3.139576e-03
## weight      -1.637090e-03 -1.818989e-06  2.495653e-03 -3.783498e-03
## moisture    -1.242370e-03  0.000000e+00 -2.282832e-03  3.661626e-03
##                      size        weight      moisture   (Intercept)
## (Intercept) -8.840289e-04  1.640728e-03  0.0012441888  0.0016189006
## classsrw     0.000000e+00  0.000000e+00  0.0000000000  0.0000000000
## density     -1.662556e-03 -2.495653e-03  0.0022846507  0.0051968527
## hardness    -3.137757e-03  3.790774e-03 -0.0036598067 -0.0002892193
## size        -3.292371e-03 -1.044100e-03 -0.0009858923  0.0008803909
## weight      -1.044100e-03 -6.293703e-03  0.0004183676 -0.0016407284
## moisture    -9.858923e-04  4.183676e-04 -0.0045838533 -0.0012441888
## (Intercept)  8.803909e-04 -1.640728e-03 -0.0012441888 -0.0016189006
## classsrw     3.637979e-06  3.637979e-06  0.0000000000  0.0000000000
## density      1.658918e-03  2.493834e-03 -0.0022846507 -0.0051950337
## hardness     3.139576e-03 -3.783498e-03  0.0036634447  0.0002910383
## size         3.296009e-03  1.040462e-03  0.0009858923 -0.0008840289
## weight       1.044100e-03  6.299160e-03 -0.0004165486  0.0016425474
## moisture     9.877112e-04 -4.165486e-04  0.0045820343  0.0012460077
##                  classsrw       density      hardness          size
## (Intercept) -3.637979e-06  5.198672e-03 -0.0002910383  0.0008840289
## classsrw     0.000000e+00  0.000000e+00  0.0000000000  0.0000000000
## density     -1.818989e-06  3.745299e-03 -0.0035470293  0.0016661943
## hardness     0.000000e+00 -3.550667e-03  0.0006421033  0.0031395757
## size         3.637979e-06  1.658918e-03  0.0031395757  0.0032960088
## weight       3.637979e-06  2.493834e-03 -0.0037834980  0.0010404619
## moisture     0.000000e+00 -2.284651e-03  0.0036634447  0.0009858923
## (Intercept)  0.000000e+00 -5.195034e-03  0.0002910383 -0.0008840289
## classsrw     0.000000e+00  3.637979e-06  0.0000000000  0.0000000000
## density      3.637979e-06 -3.743480e-03  0.0035543053 -0.0016661943
## hardness     0.000000e+00  3.554305e-03 -0.0006439222 -0.0031359377
## size         0.000000e+00 -1.666194e-03 -0.0031359377 -0.0032923708
```

```
## weight      -1.818989e-06 -2.497472e-03  0.0037889549 -0.0010440999
## moisture     1.818989e-06  2.282832e-03 -0.0036634447 -0.0009858923
##                   weight      moisture
## (Intercept) -1.637090e-03 -1.242370e-03
## classsrw    -1.818989e-06  0.000000e+00
## density      2.495653e-03 -2.282832e-03
## hardness    -3.783498e-03  3.661626e-03
## size         1.044100e-03  9.877112e-04
## weight       6.299160e-03 -4.165486e-04
## moisture    -4.165486e-04  4.582034e-03
## (Intercept)  1.642547e-03  1.246008e-03
## classsrw    -1.818989e-06  1.818989e-06
## density     -2.497472e-03  2.282832e-03
## hardness     3.788955e-03 -3.663445e-03
## size        -1.044100e-03 -9.858923e-04
## weight      -6.297341e-03  4.165486e-04
## moisture     4.165486e-04 -4.583853e-03
```

```
# the estimated covariance matrix
vcov(wheat.mod)
```

```
##                     Scab:(Intercept) Scab:classsrw  Scab:density Scab:hardness
## Scab:(Intercept)        18.402938332  0.5432053570 -1.197620e+01 -1.517788e-03
## Scab:classsrw            0.543205357  0.4396946554 -1.668212e-01  2.747157e-03
## Scab:density           -11.976200979 -0.1668211816  9.710541e+00 -1.115248e-03
## Scab:hardness           -0.001517788  0.0027471574 -1.115248e-03  1.055671e-04
## Scab:size               -0.313977363 -0.0289294890 -1.188236e-01 -2.242729e-03
## Scab:weight             -0.024552919  0.0081092793  1.039371e-02  2.553441e-04
## Scab:moisture           -0.238085442 -0.0680664781  2.389726e-02 -3.026129e-04
## Sprout:(Intercept)      12.807763893  0.2363817827 -9.001142e+00 -3.176215e-03
## Sprout:classsrw          0.210558508  0.1628421767 -7.776222e-02  1.050589e-03
## Sprout:density          -8.810479801 -0.0962164423  6.959394e+00  8.603524e-04
## Sprout:hardness         -0.001957409  0.0010640657 -1.319396e-05  4.867791e-05
## Sprout:size             -0.053738680  0.0116200639 -1.134686e-01 -9.181111e-04
## Sprout:weight           -0.014943303  0.0005338826  8.388231e-03  1.127364e-04
## Sprout:moisture         -0.113503671 -0.0238294132  3.308393e-02 -8.419978e-05
##                         Scab:size   Scab:weight Scab:moisture Sprout:(Intercept)
## Scab:(Intercept)   -0.3139773634 -0.0245529193 -2.380854e-01      12.8077638930
## Scab:classsrw      -0.0289294890  0.0081092793 -6.806648e-02       0.2363817827
## Scab:density       -0.1188235869  0.0103937054  2.389726e-02      -9.0011424289
## Scab:hardness      -0.0022427294  0.0002553441 -3.026129e-04      -0.0031762155
## Scab:size           0.5964259736 -0.0347753262  1.336683e-02      -0.0001378718
## Scab:weight        -0.0347753262  0.0038072011 -1.942459e-03      -0.0101634786
## Scab:moisture       0.0133668332 -0.0019424589  2.397566e-02      -0.1192794398
## Sprout:(Intercept) -0.0001378718 -0.0101634786 -1.192794e-01      14.1919038468
## Sprout:classsrw     0.0114278839  0.0004195723 -2.306453e-02       0.3383598328
## Sprout:density     -0.1481996753  0.0034213267  3.811244e-02      -9.6155373269
```

```
## Sprout:hardness    -0.0009650957  0.0001073025 -7.046648e-05      -0.0013491315
## Sprout:size         0.2150518165 -0.0100923196  1.644835e-03      -0.1369568476
## Sprout:weight      -0.0105234221  0.0010297479 -4.390577e-04      -0.0098215495
## Sprout:moisture     0.0020299034 -0.0004509080  8.406835e-03      -0.1552286726
##                    Sprout:classsrw Sprout:density Sprout:hardness   Sprout:size
## Scab:(Intercept)      0.2105585084  -8.8104798009   -1.957409e-03 -0.0537386797
## Scab:classsrw         0.1628421767  -0.0962164423    1.064066e-03  0.0116200639
## Scab:density         -0.0777622185   6.9593936748   -1.319396e-05 -0.1134685971
## Scab:hardness         0.0010505894   0.0008603524    4.867791e-05 -0.0009181111
## Scab:size             0.0114278839  -0.1481996753   -9.650957e-04  0.2150518165
## Scab:weight           0.0004195723   0.0034213267    1.073025e-04 -0.0100923196
## Scab:moisture        -0.0230645254   0.0381124407   -7.046648e-05  0.0016448350
## Sprout:(Intercept)    0.3383598328  -9.6155373269   -1.349131e-03 -0.1369568476
## Sprout:classsrw       0.2509207203  -0.0921286113    1.605835e-03  0.0125313646
## Sprout:density       -0.0921286113   7.6413870533   -1.184597e-04 -0.1251924336
## Sprout:hardness       0.0016058346  -0.0001184597    6.570315e-05 -0.0012250336
## Sprout:size           0.0125313646  -0.1251924336   -1.225034e-03  0.2926070656
## Sprout:weight         0.0001646941   0.0022936057    1.275157e-04 -0.0133206436
## Sprout:moisture      -0.0374122280   0.0368856221   -1.669261e-04  0.0032090270
##                    Sprout:weight Sprout:moisture
## Scab:(Intercept)   -0.0149433028   -1.135037e-01
## Scab:classsrw       0.0005338826   -2.382941e-02
## Scab:density        0.0083882311    3.308393e-02
## Scab:hardness       0.0001127364   -8.419978e-05
## Scab:size          -0.0105234221    2.029903e-03
## Scab:weight         0.0010297479   -4.509080e-04
## Scab:moisture      -0.0004390577    8.406835e-03
## Sprout:(Intercept) -0.0098215495   -1.552287e-01
## Sprout:classsrw     0.0001646941   -3.741223e-02
## Sprout:density      0.0022936057    3.688562e-02
## Sprout:hardness     0.0001275157   -1.669261e-04
## Sprout:size        -0.0133206436    3.209027e-03
## Sprout:weight       0.0013671457   -6.169973e-04
## Sprout:moisture    -0.0006169973    1.270553e-02
```