HW2

Yaqi

4/11/2019

1 Tests for normality

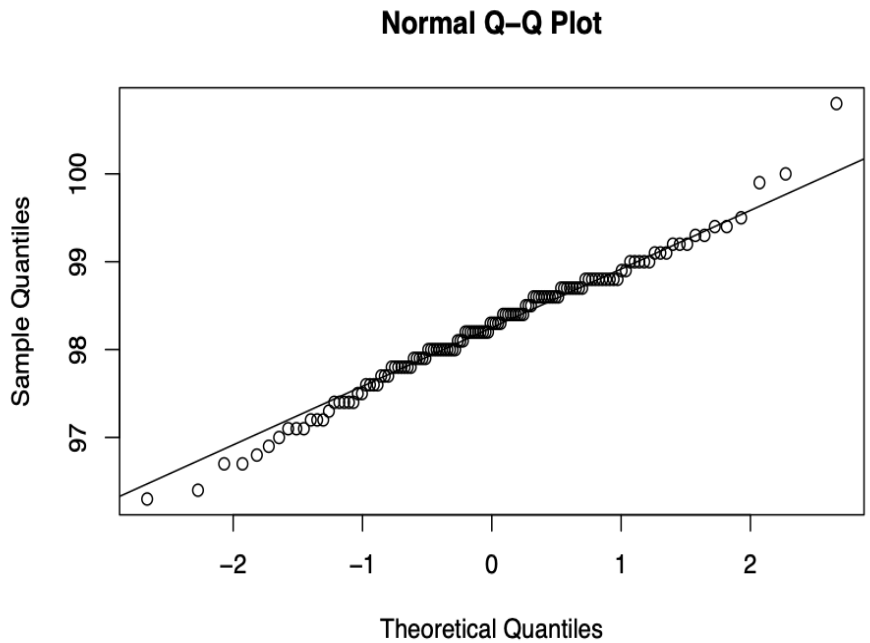
1.1 Are body temperatures normally distributed?

```
bodytemp = read.csv("/Users/yakili/Downloads/HW02 data sets/bodytemp.csv")
attach(bodytemp)
str(bodytemp)
```

```
## 'data.frame':    130 obs. of  3 variables:
##  $ temp: num  96.3 96.7 96.9 97 97.1 97.1 97.1 97.2 97.3 97.4 ...
##  $ sex : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ bpm : int  70 71 74 80 73 75 82 64 69 70 ...
```

1. Create a qqnorm plot

```
qqnorm(temp)
qqline(temp)
```



The qqnorm plot shows nearly a straight line, which indicates the temperature distribution is quite normal.

```
shapiro.test(temp)
```

```
##
##  Shapiro-Wilk normality test
```

```
##
## data:  temp
## W = 0.98658, p-value = 0.2332
```

The p-value of shapiro test is 0.2332, which is high enough to fail to reject the data is normally distributed.

2 T-test

Conduct a t-test for men and women separately to test the null hypothesis that **temperature is 98.6 degrees Fahrenheit**.

First, we should seperate men's temperature and women's temperature from the dataframe.

```
mentemp = bodytemp %>%
  filter(bodytemp$sex == 0) %>% select(temp)
womentemp = bodytemp %>%
  filter(bodytemp$sex == 1) %>% select(temp)
```

Then, we can conduct a t-test for men to test the null hypothesis that temperature is 98.6 degrees Fahrenheit.

```
t.test(mentemp, mu = 98.6)
```

```
##
##  One Sample t-test
##
## data:  mentemp
## t = -5.7158, df = 64, p-value = 3.084e-07
## alternative hypothesis: true mean is not equal to 98.6
## 95 percent confidence interval:
##  97.93147 98.27776
## sample estimates:
## mean of x
##  98.10462
```

The p-value of the t-test is 3.084e-07, small enough to reject the null hypothesis that the body temperature of men is equal to 98.6. Then, we conduct the t-test for women's body temperature.

```
t.test(womentemp, mu = 98.6)
```

```
##
##  One Sample t-test
##
## data:  womentemp
## t = -2.2355, df = 64, p-value = 0.02888
## alternative hypothesis: true mean is not equal to 98.6
## 95 percent confidence interval:
##  98.20962 98.57807
## sample estimates:
## mean of x
##  98.39385
```

The p-value equals 0.02888, which means we fail to reject the null hypothesis, which means women's body temperature is very likely to be 98.6.

Now we can conduct a t-test to test the null hypothesis that men and women have the same body temperature.

```r
t.test(mentemp,womentemp)
```

```
##
##  Welch Two Sample t-test
##
## data:  mentemp and womentemp
## t = -2.2854, df = 127.51, p-value = 0.02394
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.53964856 -0.03881298
## sample estimates:
## mean of x mean of y
##  98.10462  98.39385
```

The p-value is 0.02394. We can reject the null hypothesis at 95% confidence level that men's body temperature is the same as women's body temperature.

3 Effects of gender and heart rate on body temperature

We can run a linear regression to find the relationship between body temperature and genter, and heart rate.

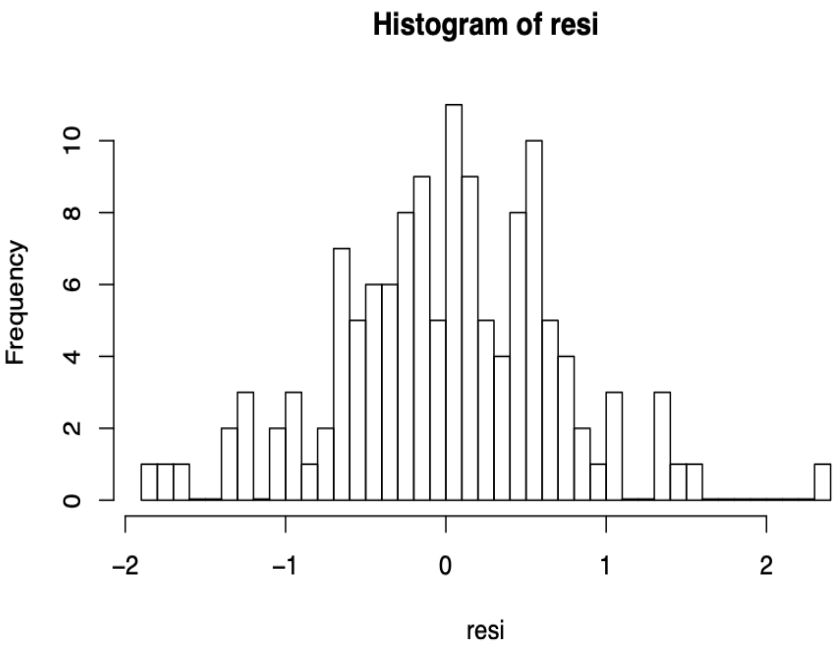
```r
linear = lm(temp~sex+bpm)
lminfo = summary(linear)
resi = lminfo$residuals
summary(linear)
```

```
##
## Call:
## lm(formula = temp ~ sex + bpm)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.86363 -0.45624  0.01841  0.47366  2.33424
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 96.250814   0.648717 148.371  < 2e-16 ***
## sex          0.269406   0.123277   2.185  0.03070 *
## bpm          0.025267   0.008762   2.884  0.00462 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7017 on 127 degrees of freedom
## Multiple R-squared:  0.09825,    Adjusted R-squared:  0.08405
## F-statistic: 6.919 on 2 and 127 DF,  p-value: 0.001406
```

3.1

The intercept of the linear regression is "96.2508140430877" slope for sex is "0.269406", and slope for heart rate is "0.025267" ##3.2 The intercept and slope of both the sex and heart rate are signicicant at the confidence interval $\alpha = 0.05$ ##3.3 We can plot the residuals out to see whether the it's normally distributed.

```r
hist(resi,breaks = 40)
```



The residuals look normally distributed. ##3.4 Shapiro test on the residuals

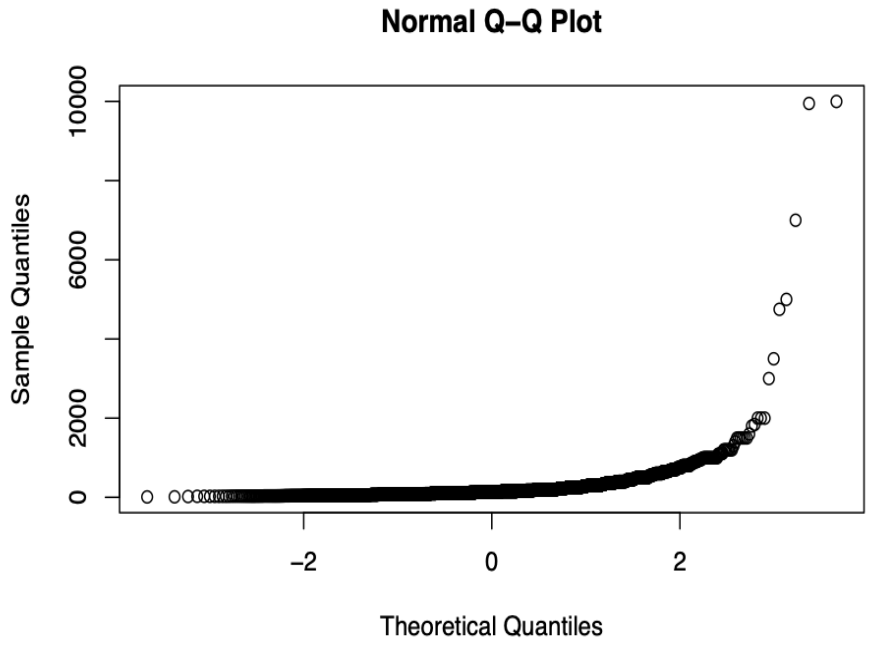
```r
shapiro.test(resi)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  resi
## W = 0.99053, p-value = 0.5222
```

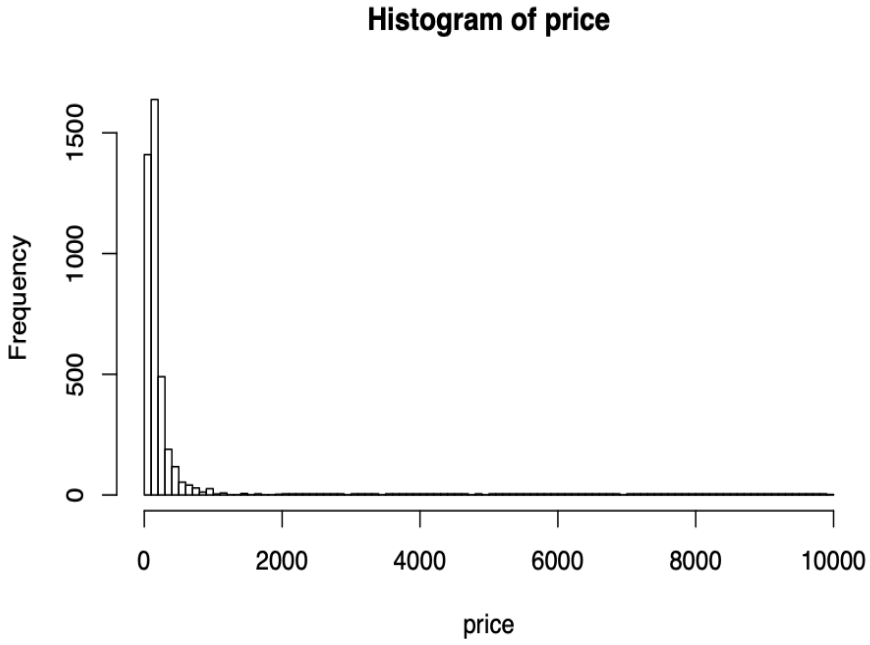
The p-value is 0.52, which is high enough to fail to reject the data is normally distributed.

4 New Orleans Airbnb price analysis

```r
airbnb = read.csv("/Users/yakili/Downloads/HW02 data sets/NOLAlistingsJune2016_subset2.csv")
attach(airbnb)
qqnorm(price)
```




```
hist(price,breaks = 100)
```




```
shapiro.test(price)
```

```
##
```

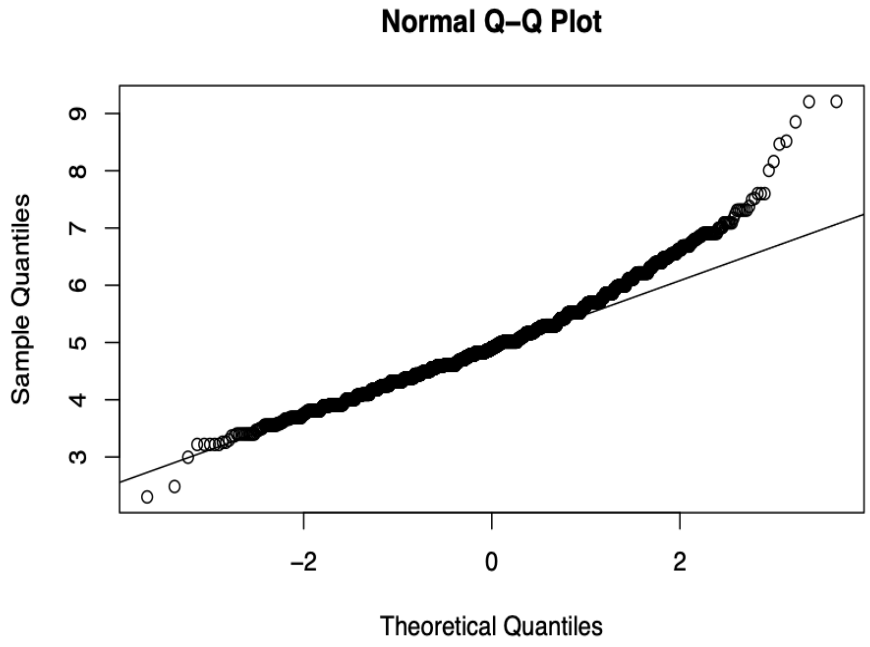
```
##  Shapiro-Wilk normality test
##
## data:  price
## W = 0.29183, p-value < 2.2e-16
```

4.1 Is price normally distributed?

Apparently, price is not normally distributed. We can transform the price to a more normally distributed data. The histogram of the price indicates that a log tranformation would solve the problem.

```
price.trans = log(price)
hist(price.trans)
```


```
qqnorm(price.trans)
qqline(price.trans)
```




```
shapiro.test(price.trans)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  price.trans
## W = 0.97299, p-value < 2.2e-16
```

After the transformation, the qqnorm plot and the shapiro test has shown some improvement on normality.

4.2 Create a linear model

Now, we can creat an OLS model of price as a function of neighourhood, room type, and availability to make simple predictions.

```
lm.price = lm(price.trans ~ neighbourhood + room_type + availability_365)
summary(lm.price)
```

```
##
## Call:
## lm(formula = price.trans ~ neighbourhood + room_type + availability_365)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.5827 -0.4004 -0.0902  0.2842  4.1493
##
## Coefficients:
##                                    Estimate Std. Error t value
## (Intercept)                       4.983e+00  8.902e-02  55.969
## neighbourhoodAudubon              3.135e-01  1.037e-01   3.022
```

```
## neighbourhoodB. W. Cooper              -5.127e-01  5.955e-01  -0.861
## neighbourhoodBayou St. John             1.189e-01  9.898e-02   1.201
## neighbourhoodBlack Pearl                9.196e-03  1.445e-01   0.064
## neighbourhoodBroadmoor                 -4.515e-02  1.115e-01  -0.405
## neighbourhoodBywater                   -8.372e-03  9.650e-02  -0.087
## neighbourhoodCentral Business District  2.696e-01  9.581e-02   2.814
## neighbourhoodCentral City               1.281e-01  9.548e-02   1.342
## neighbourhoodCity Park                  1.057e-01  1.122e-01   0.943
## neighbourhoodEast Carrollton            8.527e-02  1.168e-01   0.730
## neighbourhoodEast Riverside             1.429e-01  1.135e-01   1.259
## neighbourhoodFairgrounds                4.695e-02  9.830e-02   0.478
## neighbourhoodFrench Quarter             4.474e-01  9.259e-02   4.832
## neighbourhoodGarden District            3.302e-01  1.412e-01   2.338
## neighbourhoodGentilly Terrace          -1.250e-01  1.274e-01  -0.981
## neighbourhoodHoly Cross                -2.200e-01  1.332e-01  -1.651
## neighbourhoodIrish Channel              4.457e-02  1.075e-01   0.415
## neighbourhoodLakeview                   3.736e-01  1.356e-01   2.755
## neighbourhoodLeonidas                  -1.090e-01  1.017e-01  -1.072
## neighbourhoodLower Garden District      1.952e-01  9.641e-02   2.024
## neighbourhoodLower Ninth Ward          -1.777e-01  2.256e-01  -0.787
## neighbourhoodMarigny                    2.278e-01  9.443e-02   2.412
## neighbourhoodMarlyville - Fontainbleau  1.394e-01  1.118e-01   1.247
## neighbourhoodMid-City                  -6.642e-03  9.382e-02  -0.071
## neighbourhoodMilan                      2.840e-01  1.074e-01   2.644
## neighbourhoodSeventh Ward               2.191e-02  9.455e-02   0.232
## neighbourhoodSt. Claude                -2.111e-01  9.779e-02  -2.159
## neighbourhoodSt. Roch                  -1.801e-01  1.031e-01  -1.746
## neighbourhoodSt. Thomas Dev             1.749e-01  1.257e-01   1.392
## neighbourhoodTouro                      2.451e-01  1.193e-01   2.055
## neighbourhoodTreme - Lafitte            8.503e-02  9.472e-02   0.898
## neighbourhoodTulane - Gravier           1.738e-02  1.370e-01   0.127
## neighbourhoodUptown                     3.437e-01  1.085e-01   3.166
## neighbourhoodWest Riverside             2.521e-01  1.116e-01   2.258
## room_typePrivate room                  -7.209e-01  2.209e-02 -32.643
## room_typeShared room                   -9.127e-01  8.378e-02 -10.894
## availability_365                        2.714e-04  6.848e-05   3.963
##                                        Pr(>|t|)
## (Intercept)                             < 2e-16 ***
## neighbourhoodAudubon                    0.00252 **
## neighbourhoodB. W. Cooper               0.38927
## neighbourhoodBayou St. John             0.22967
## neighbourhoodBlack Pearl                0.94927
## neighbourhoodBroadmoor                  0.68555
## neighbourhoodBywater                    0.93088
## neighbourhoodCentral Business District  0.00492 **
## neighbourhoodCentral City               0.17964
## neighbourhoodCity Park                  0.34592
## neighbourhoodEast Carrollton            0.46533
## neighbourhoodEast Riverside             0.20804
## neighbourhoodFairgrounds                0.63298
## neighbourhoodFrench Quarter            1.40e-06 ***
## neighbourhoodGarden District            0.01942 *
## neighbourhoodGentilly Terrace           0.32673
## neighbourhoodHoly Cross                 0.09879 .
```

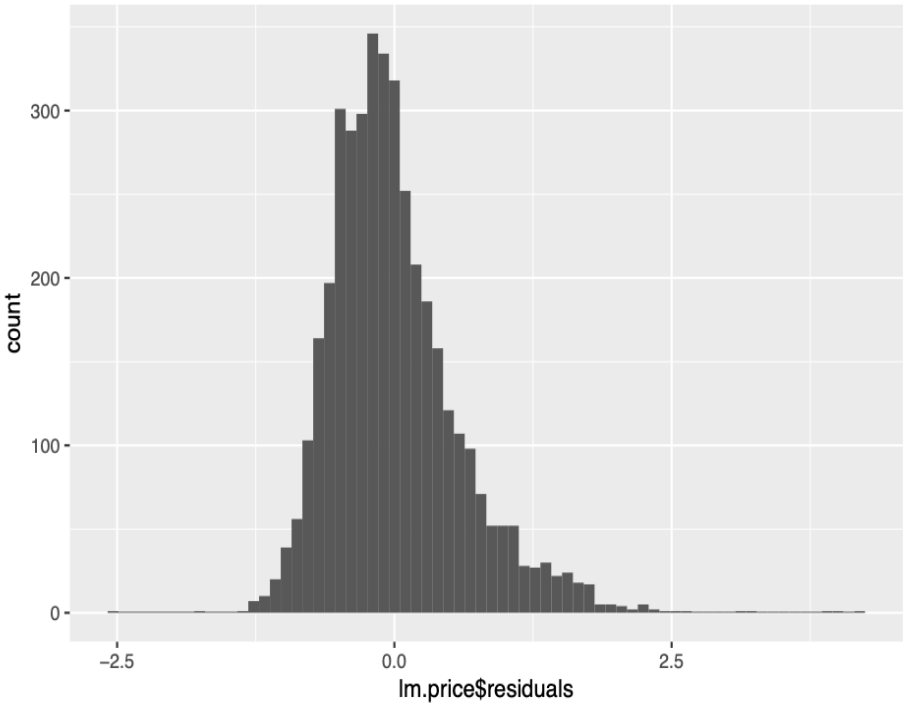
```
## neighbourhoodIrish Channel              0.67831
## neighbourhoodLakeview                   0.00590 **
## neighbourhoodLeonidas                   0.28396
## neighbourhoodLower Garden District      0.04302 *
## neighbourhoodLower Ninth Ward           0.43113
## neighbourhoodMarigny                    0.01590 *
## neighbourhoodMarlyville - Fontainbleau  0.21233
## neighbourhoodMid-City                   0.94357
## neighbourhoodMilan                      0.00822 **
## neighbourhoodSeventh Ward               0.81674
## neighbourhoodSt. Claude                 0.03094 *
## neighbourhoodSt. Roch                   0.08082 .
## neighbourhoodSt. Thomas Dev             0.16406
## neighbourhoodTouro                      0.03996 *
## neighbourhoodTreme - Lafitte            0.36939
## neighbourhoodTulane - Gravier           0.89906
## neighbourhoodUptown                     0.00156 **
## neighbourhoodWest Riverside             0.02397 *
## room_typePrivate room                   < 2e-16 ***
## room_typeShared room                    < 2e-16 ***
## availability_365                       7.53e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5889 on 4000 degrees of freedom
## Multiple R-squared:  0.3128, Adjusted R-squared:  0.3065
## F-statistic: 49.21 on 37 and 4000 DF,  p-value: < 2.2e-16
```

4.3 The interpretation of the model

The significant coefficients of the linear regression are Intercept, In the neighborhood variable, Audubon, Central Business District, French Quarterm,Garden District, dMarigny, St. Claude, Touro, Milan, Uptown, West Riverside are significant coefficients to predict the price. room_type of Private room and Shared room, along with availability_365 are significant coefficients.

Check the normality of residuals of the linear regression.

```
ggplot(lm.price,aes(x= lm.price$residuals))  +geom_histogram(bins = 70)
```




```
shapiro.test(lm.price$residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  lm.price$residuals
## W = 0.93941, p-value < 2.2e-16
```

The residuals seem to be a slightly skewed normal distribution. The shapiro test shows that the residuals are not normally distributed. There is something wrong with the assumptions of the linear regression.

5 Cooking Procedures

```
doe = read.csv("/Users/yakili/Downloads/HW02 data sets/GBDOE-French Fry DOE.csv")
str(doe)
```

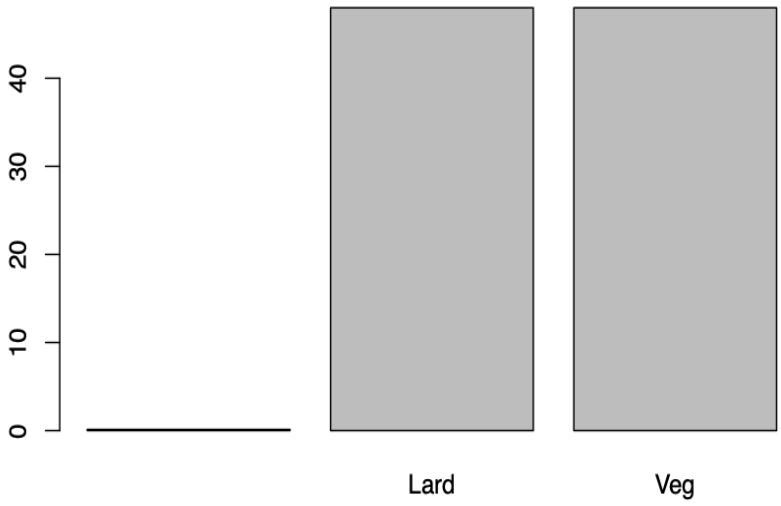
```
## 'data.frame':    120 obs. of  5 variables:
##  $ Potato.Type     : Factor w/ 3 levels "","Idaho","Maine": 3 3 3 3 3 3 3 3 3 3 ...
##  $ Cooking.Oil.Type: Factor w/ 3 levels "","Lard","Veg": 3 3 3 3 3 3 3 3 3 3 ...
##  $ Cooking.Temp    : int  320 320 320 330 330 330 340 340 340 350 ...
##  $ Cooking.Time    : int  10 11 12 10 11 12 10 11 12 10 ...
##  $ Taste.Rating    : num  8.2 8.4 8.8 9.2 9.4 9.2 9.3 9.5 9.3 9.1 ...
```

```
doe = na.omit(doe)
attach(doe)
```

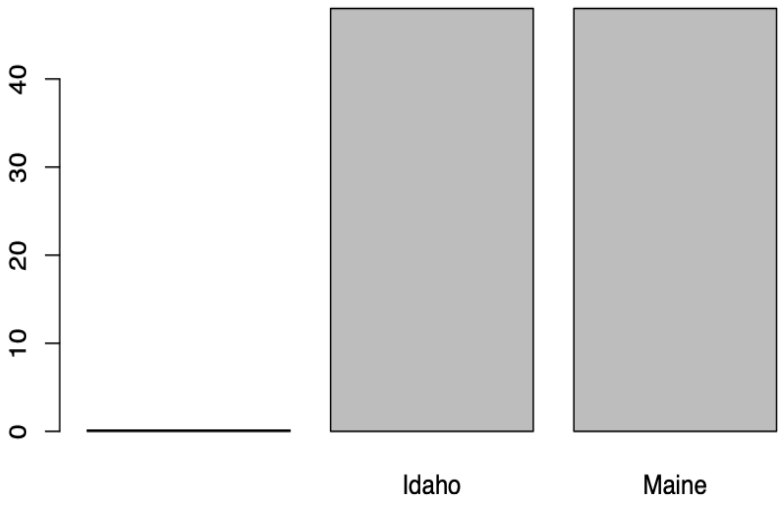
Let's see how these variables are distributed first to meet the normal distribution assumption needed for

linear regression.

```
barplot(table(Cooking.Oil.Type))
```

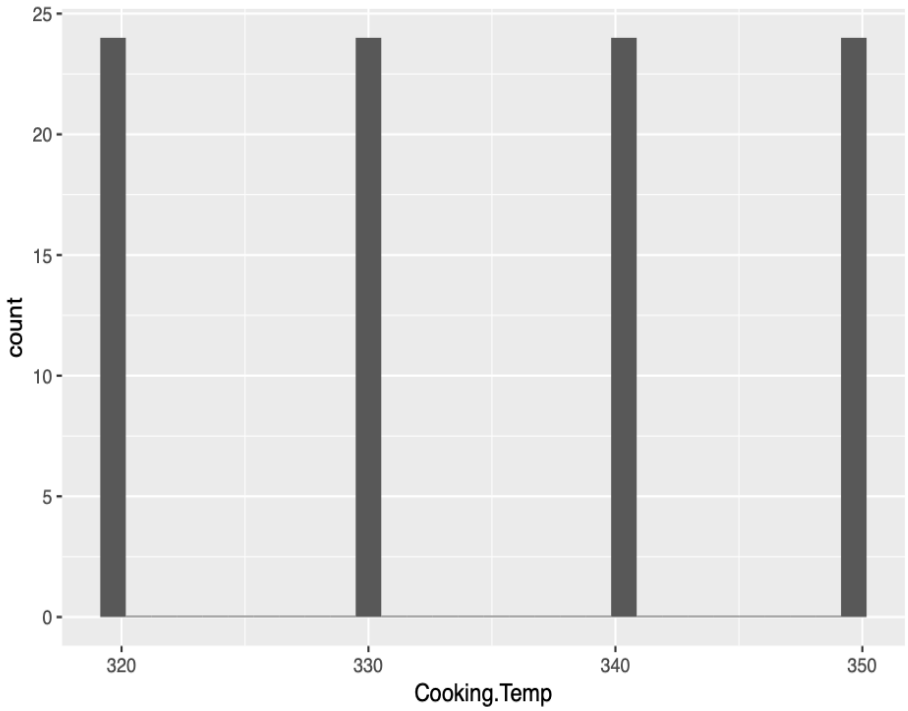



```
barplot(table(Potato.Type))
```



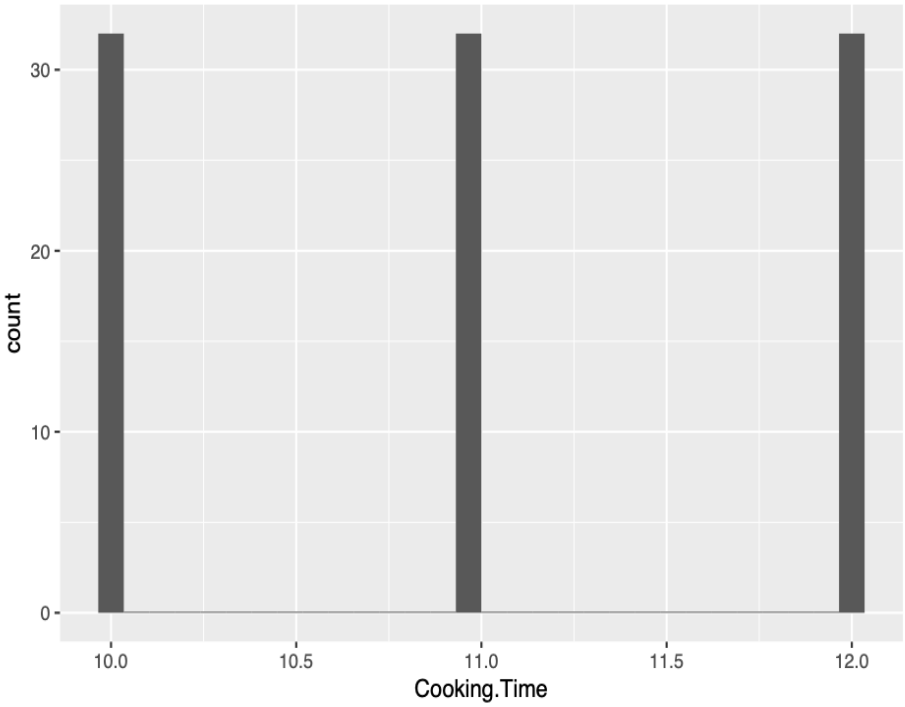

```
ggplot(doe,aes(x=Cooking.Temp)) + geom_histogram()
```

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




```
ggplot(doe,aes(x=Cooking.Time)) + geom_histogram()
```

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



It seems that cooking time and cooking temperature are catogorical variables. We should change them into catogorical values.

```
doe$Cooking.Temp = as.ordered(doe$Cooking.Temp)
doe$Cooking.Time = as.ordered(doe$Cooking.Time)
```

Now we can build a linear model to explain the relationship between taste ratings and other variables.

```
lm.doe = lm(Taste.Rating ~ Cooking.Oil.Type + Cooking.Temp + Cooking.Time + Potato.Type, data = doe)
summary(lm.doe)
```

```
##
## Call:
## lm(formula = Taste.Rating ~ Cooking.Oil.Type + Cooking.Temp +
##     Cooking.Time + Potato.Type, data = doe)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.39854 -0.14552 -0.01354  0.12339  0.40896
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)          8.590729   0.034590 248.356  < 2e-16 ***
## Cooking.Oil.TypeVeg  0.381042   0.039942   9.540 3.17e-15 ***
## Cooking.Temp.L       0.266558   0.039942   6.674 2.14e-09 ***
## Cooking.Temp.Q      -0.672292   0.039942 -16.832  < 2e-16 ***
## Cooking.Temp.C       0.021149   0.039942   0.530  0.59778
## Cooking.Time.L      -0.009723   0.034590  -0.281  0.77931
## Cooking.Time.Q      -0.104614   0.034590  -3.024  0.00327 **
```

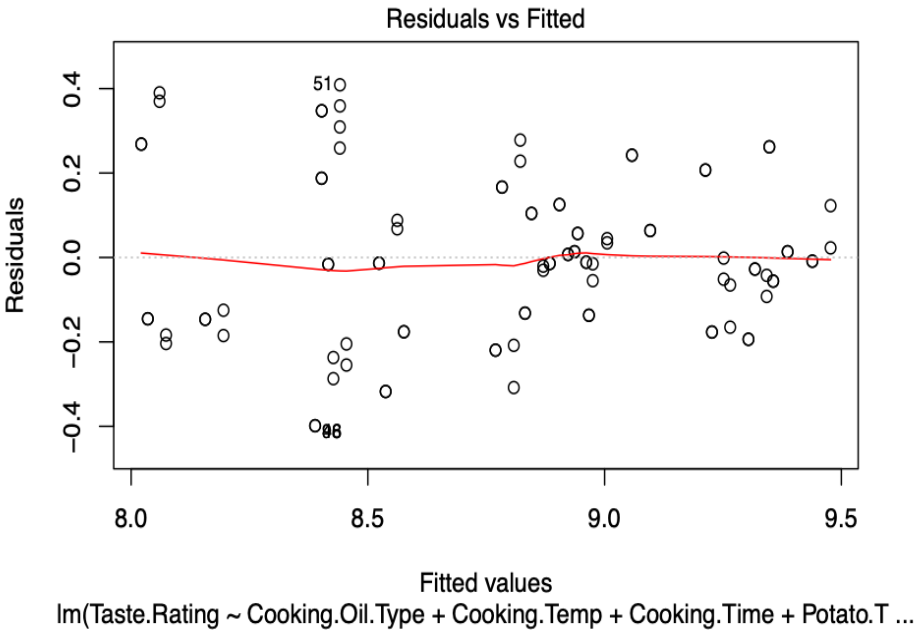
```
## Potato.TypeMaine     0.038542   0.039942   0.965  0.33721
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1957 on 88 degrees of freedom
## Multiple R-squared:  0.8299, Adjusted R-squared:  0.8164
## F-statistic: 61.33 on 7 and 88 DF,  p-value: < 2.2e-16
```

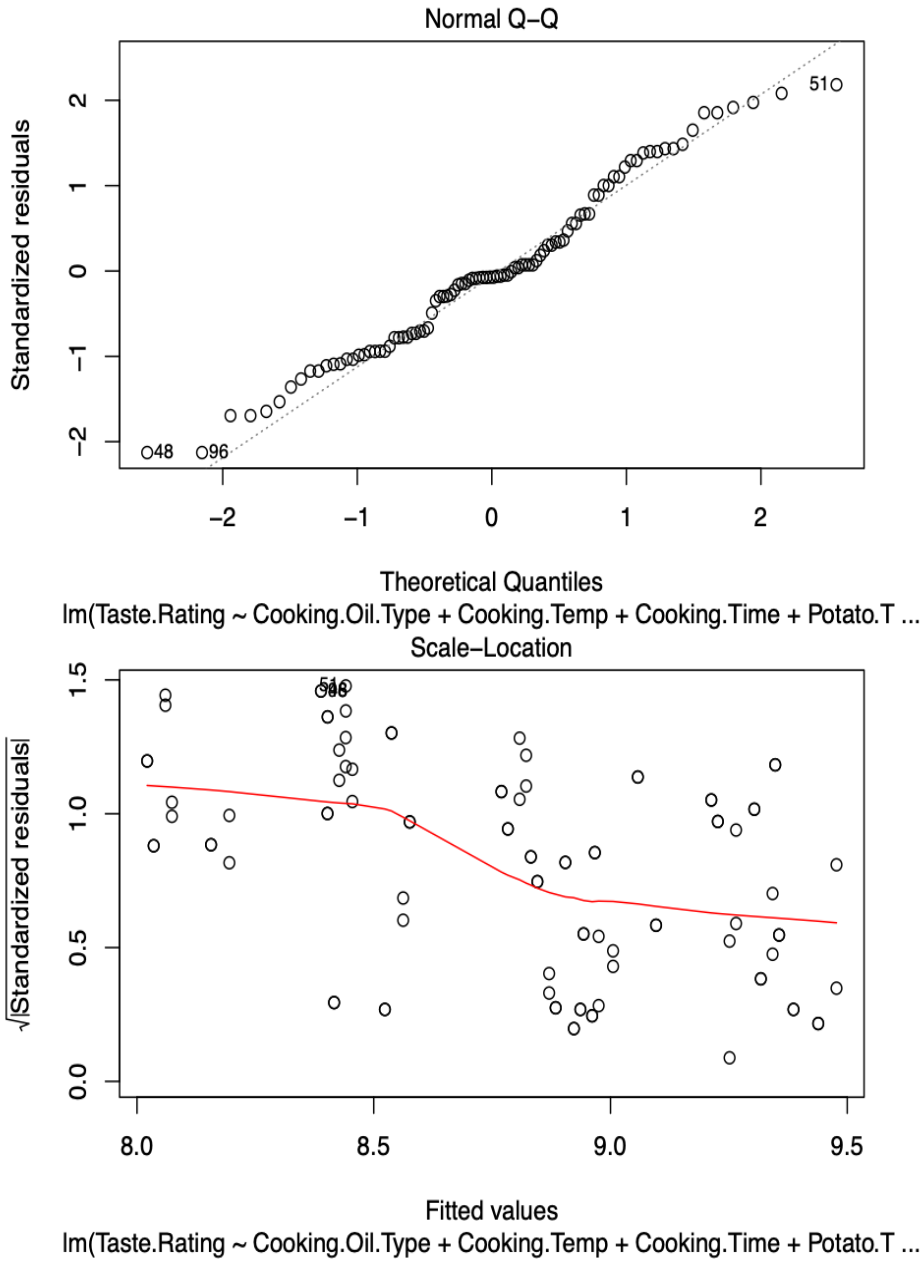
From the linear regression, only Intercept, Veg of the Cooking oil type, cooking temperature, and cooking time are significant enough to explain the taste rating. The Adjusted R-squared is 0.2316, which means the model only explains only 23% of the taste rating and it is actually not a good model. The model explains 81.64% of the taste ratings.

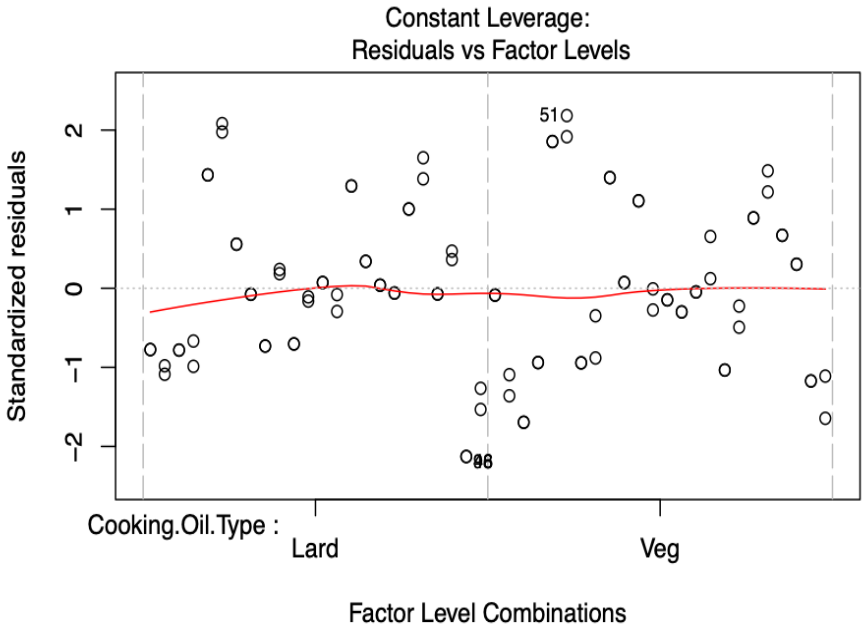
5.2

Present the results of the linear regression model.

```
plot(lm.doe)
```




Normal Q–Q
Standardized residuals
Theoretical Quantiles
lm(Taste.Rating ~ Cooking.Oil.Type + Cooking.Temp + Cooking.Time + Potato.T ...
Scale–Location
√|Standardized residuals|
Fitted values
lm(Taste.Rating ~ Cooking.Oil.Type + Cooking.Temp + Cooking.Time + Potato.T ...

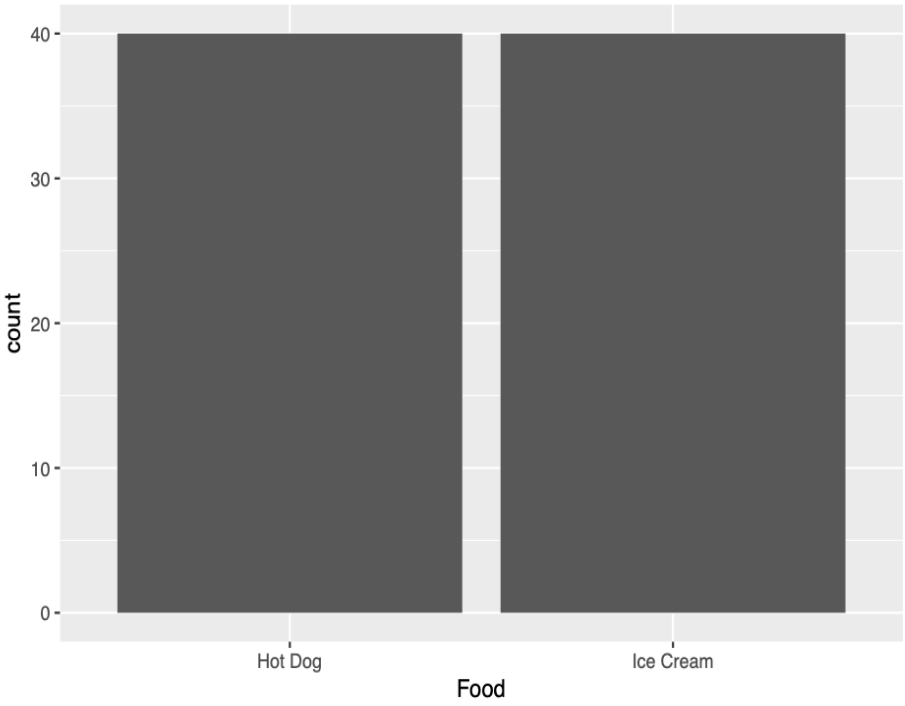


The residuals of the linear regression are distributed normally.

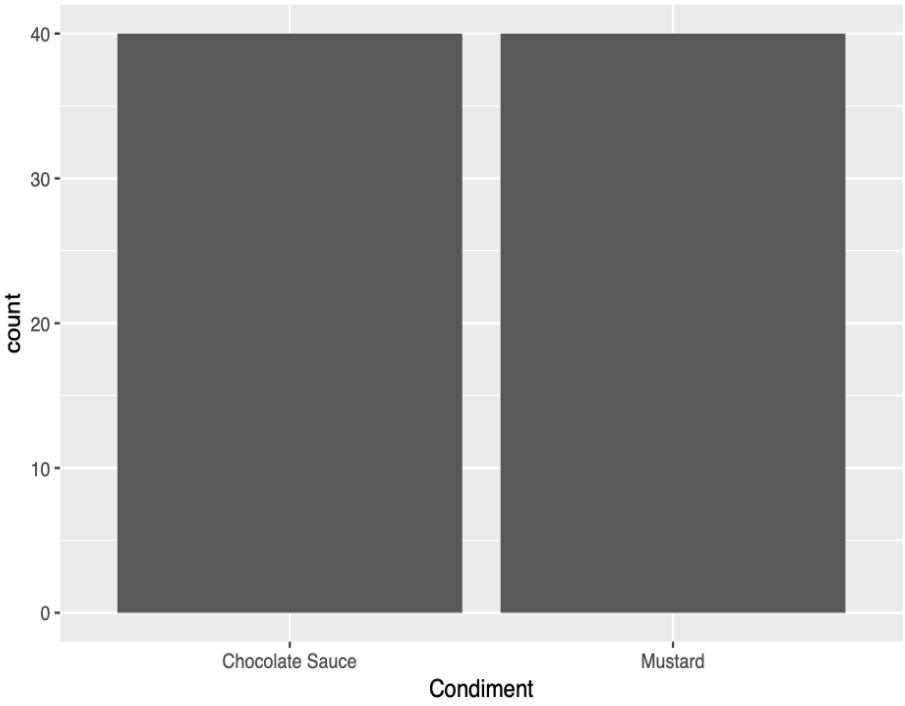
```
fe = read.csv("/Users/yakili/Downloads/HW02 data sets/foodenjoyment.csv")
attach(fe)
glimpse(fe)
```

```
## Observations: 80
## Variables: 3
## $ Enjoyment <dbl> 81.92696, 84.93977, 90.28648, 89.56180, 97.67683, 83...
## $ Food      <fct> Hot Dog, Hot Dog, Hot Dog, Hot Dog, Hot Dog, Hot Dog...
## $ Condiment <fct> Mustard, Mustard, Mustard, Mustard, Mustard, Mustard...
```

```
ggplot(fe, aes(x=Food))+geom_bar()
```



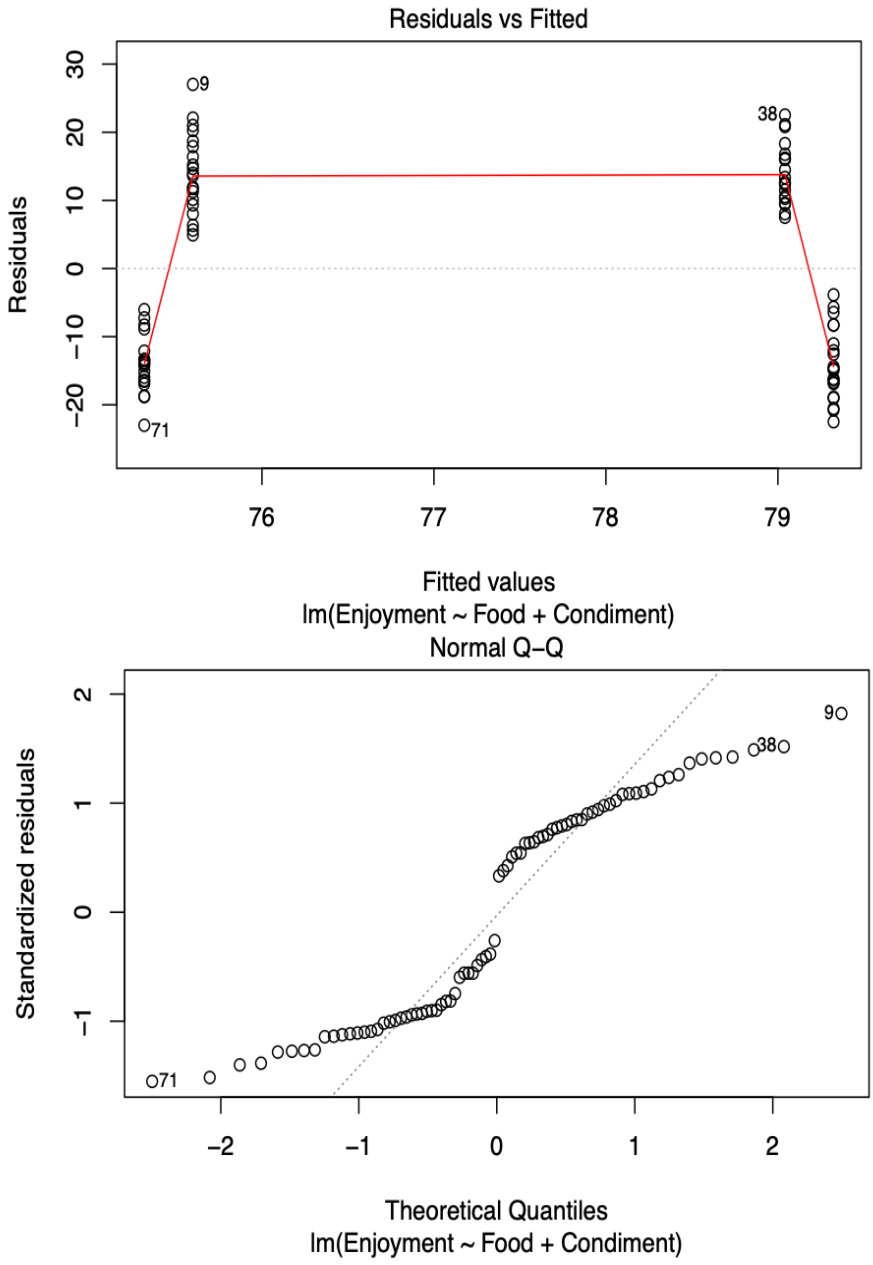

```
ggplot(fe, aes(x=Condiment ))+geom_bar()
```

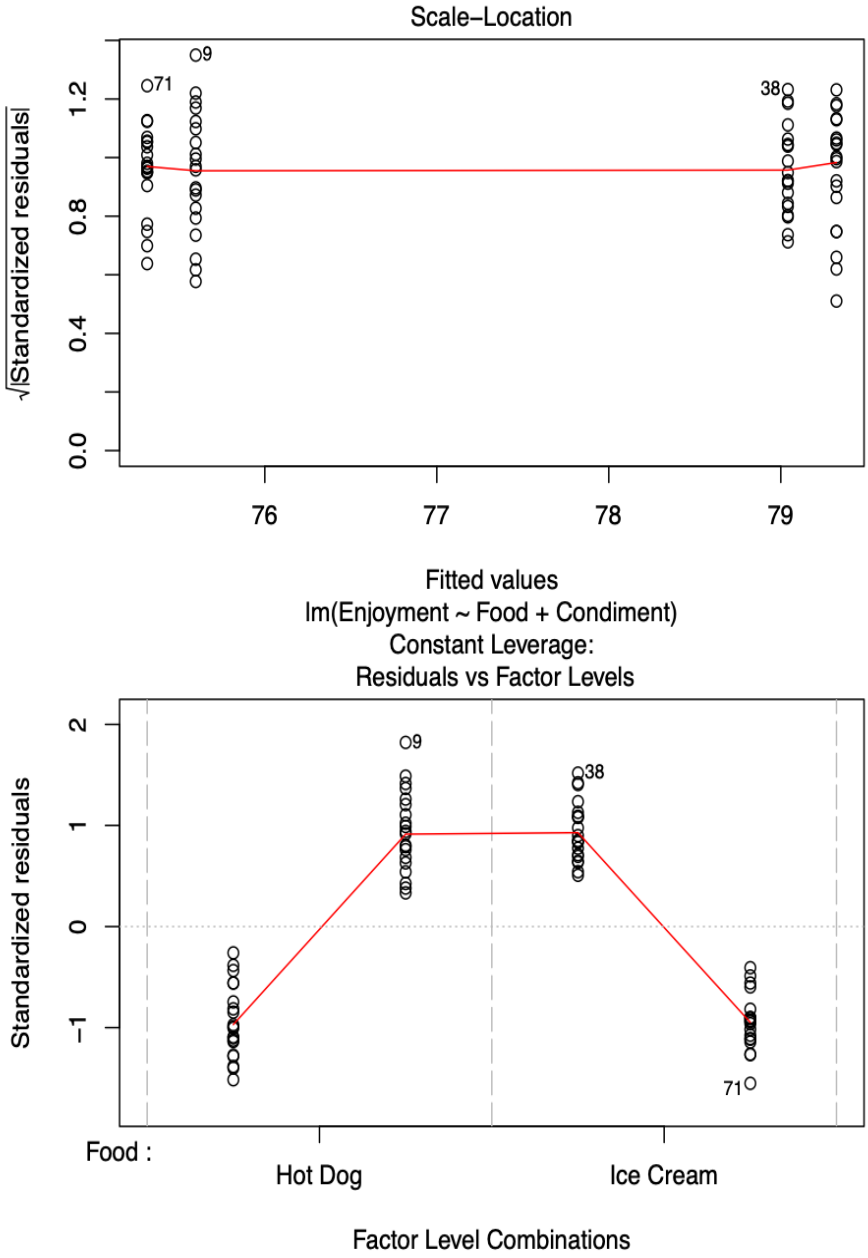



```
lm.fe = lm(Enjoyment ~ Food + Condiment , data = fe)
summary(lm.fe)
```

```
##
## Call:
## lm(formula = Enjoyment ~ Food + Condiment, data = fe)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -23.0067 -14.3016   0.5382  13.4187  27.0218
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)       79.3237     2.9278  27.093   <2e-16 ***
## FoodIce Cream     -0.2826     3.3807  -0.084    0.934
## CondimentMustard  -3.7251     3.3807  -1.102    0.274
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 15.12 on 77 degrees of freedom
## Multiple R-squared:  0.01561,    Adjusted R-squared:  -0.009958
## F-statistic: 0.6105 on 2 and 77 DF,  p-value: 0.5457
```

```
plot(lm.fe)
```


Residuals vs Fitted
Residuals
30
20
10
0
−10
−20
9
38
71
76
77
78
79
Fitted values
lm(Enjoyment ~ Food + Condiment)
Normal Q−Q
Standardized residuals
2
1
0
−1
9
38
71
−2
−1
0
1
2
Theoretical Quantiles
lm(Enjoyment ~ Food + Condiment)



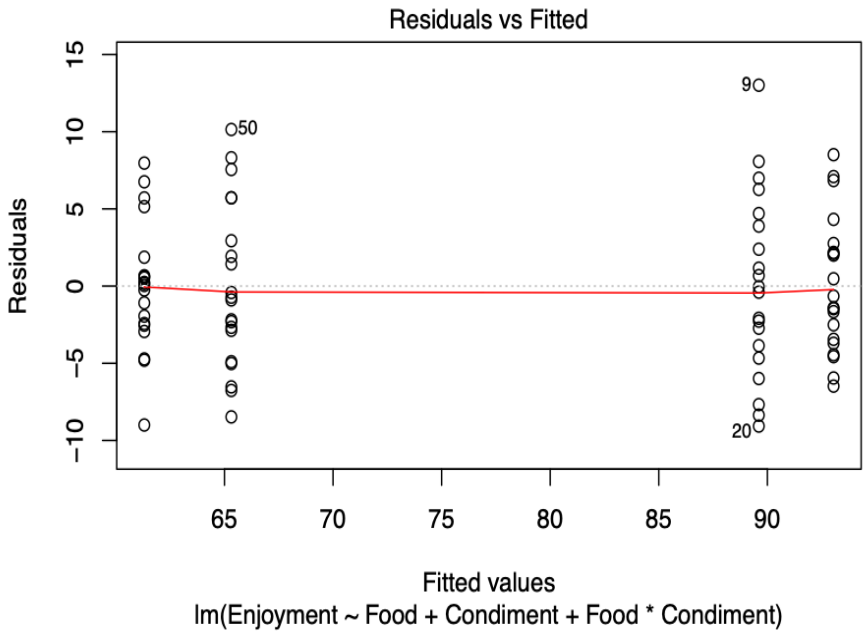
Only the intercept are statistically significant and the model can only explain less than 1 percent of the data, which means this is a pretty bad linear model to see what's driving the enjoyment.

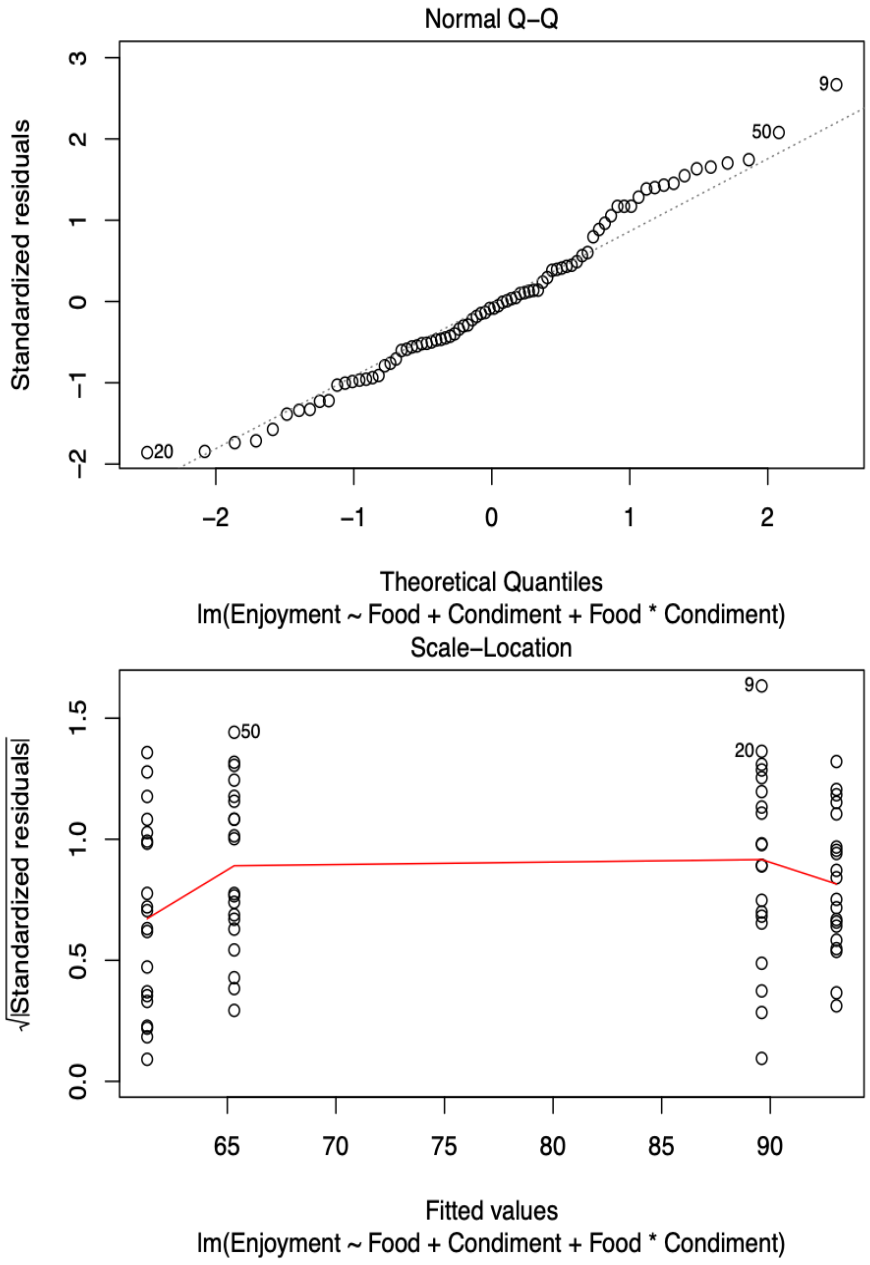
We add an interaction term of food and condiment

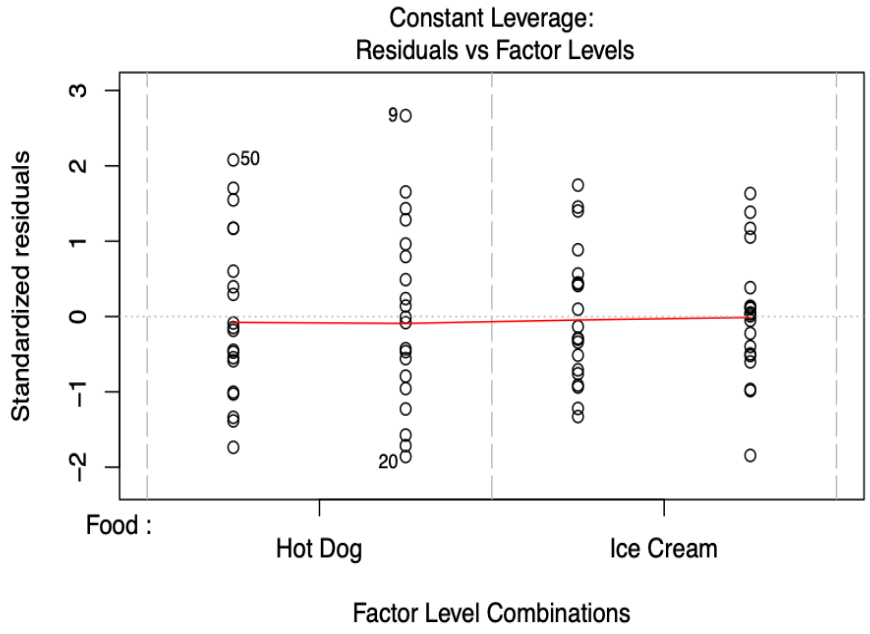
```
lm.fe.int = lm(Enjoyment ~ Food + Condiment + Food * Condiment, data = fe)
summary(lm.fe.int)
```

```
##
## Call:
## lm(formula = Enjoyment ~ Food + Condiment + Food * Condiment,
##     data = fe)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -9.068 -3.068 -0.407  2.802 13.015
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)                     65.317      1.120   58.34   <2e-16 ***
## FoodIce Cream                   27.731      1.583   17.52   <2e-16 ***
## CondimentMustard                24.289      1.583   15.34   <2e-16 ***
## FoodIce Cream:CondimentMustard -56.028      2.239  -25.02   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.007 on 76 degrees of freedom
## Multiple R-squared:  0.8935, Adjusted R-squared:  0.8892
## F-statistic: 212.4 on 3 and 76 DF,  p-value: < 2.2e-16
```

```
plot(lm.fe.int)
```




Normal Q–Q
Standardized residuals
Theoretical Quantiles
lm(Enjoyment ~ Food + Condiment + Food * Condiment)
Scale–Location
√|Standardized residuals|
Fitted values
lm(Enjoyment ~ Food + Condiment + Food * Condiment)



After adding the interaction, the model now can explain 89% of the price now. There are three significant variables: Ice Cream, Mustard, and Mustard Ice Cream. From the fit, Ice Cream and Mustard alone increases the price while an Ice Cream of mustard condiment decreases the price a lot. That's just too much. The distribution of regression residuals shows satisfiable normality.

Homework time

Approximately 2 hours.

NYCAirbnb

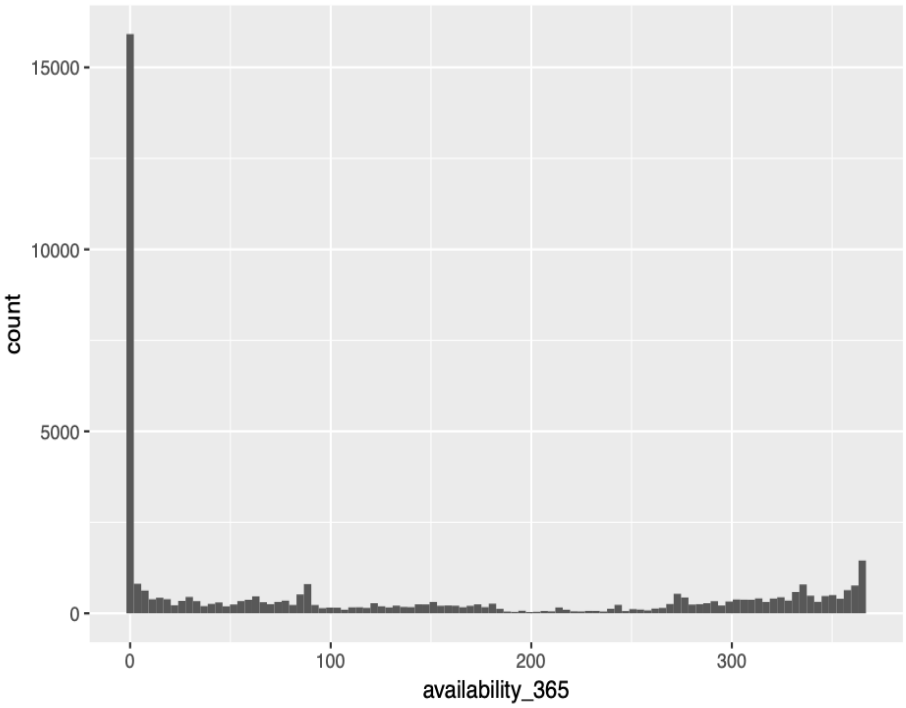
```
nyc = read.csv("/Users/yakili/Downloads/HW02 data sets/NYCairbnb.csv")
glimpse(nyc)
```

```
## Observations: 44,317
## Variables: 16
## $ id                            <int> 18461891, 20702398, 6627449, 19...
## $ name                          <fct> "Bright, comfortable 1B studio ...
## $ host_id                       <int> 916092, 1457680, 13886510, 1149...
## $ host_name                     <fct> Connie Mae, James, Arlene, MoMo...
## $ neighbourhood_group           <fct> Queens, Bronx, Bronx, Bronx, Br...
## $ neighbourhood                 <fct> Ditmars Steinway, City Island, ...
## $ latitude                      <dbl> 40.77414, 40.84919, 40.84977, 4...
## $ longitude                     <dbl> -73.91625, -73.78651, -73.78661...
## $ room_type                     <fct> Entire home/apt, Private room, ...
## $ price                         <int> 110, 50, 125, 100, 300, 69, 150...
## $ minimum_nights                <int> 6, 1, 3, 3, 7, 3, 2, 1, 1, 7, 2...
## $ number_of_reviews             <int> 0, 2, 21, 0, 0, 94, 3, 31, 0, 0...
```

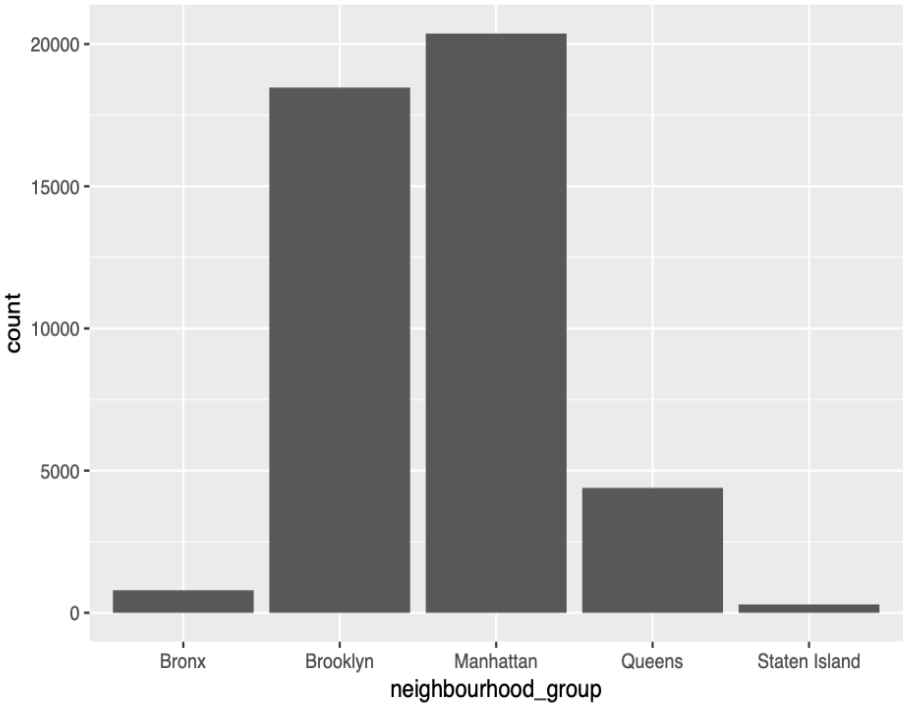
```
## $ last_review                   <fct> , 10/1/17, 9/26/17, , , 7/31/17...
## $ reviews_per_month             <dbl> NA, 2.00, 0.77, NA, NA, 3.27, 1...
## $ calculated_host_listings_count <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1...
## $ availability_365              <int> 0, 169, 363, 90, 365, 325, 74, ...
```

Build the linear regression model with several plausible variables to predict the price. Location, room type, minimum nights to stay, number of reviews, and availability can all affect the price.

```
ggplot(nyc, aes(x = availability_365)) +geom_histogram(bins = 100)
```



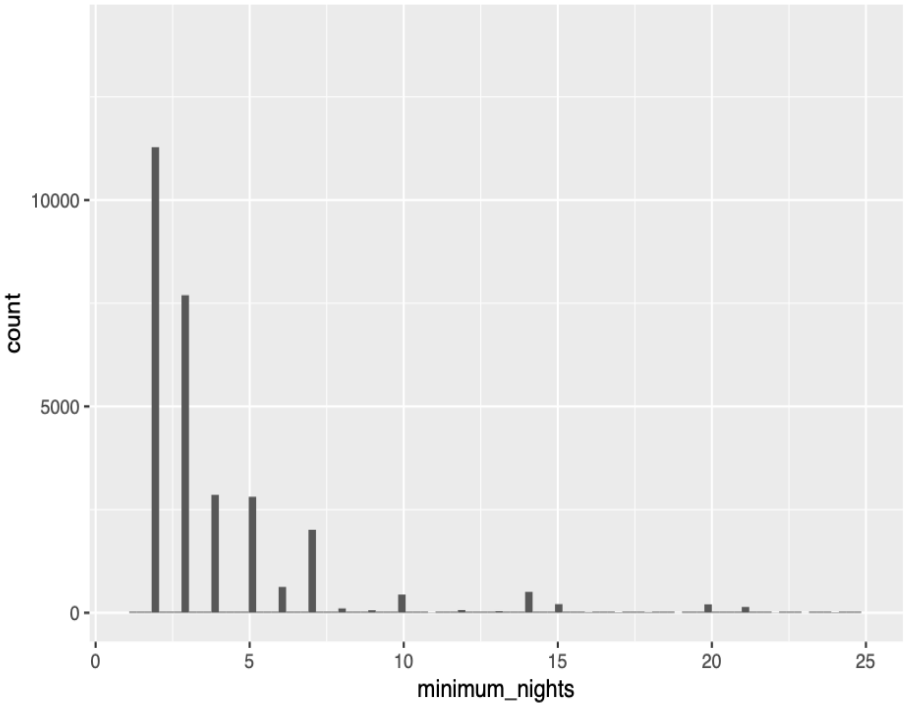

```
ggplot(nyc, aes(x = neighbourhood_group)) +geom_bar()
```




```
ggplot(nyc, aes(x = minimum_nights)) +geom_histogram(bins = 100) + xlim(c(1,25))
```

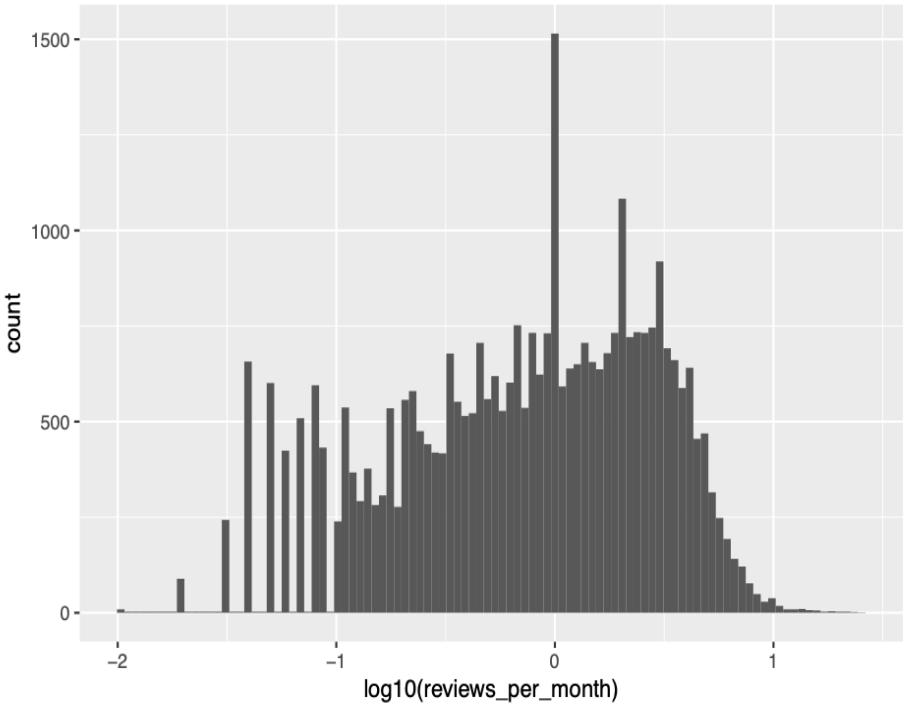
```
## Warning: Removed 1063 rows containing non-finite values (stat_bin).
```

```
## Warning: Removed 2 rows containing missing values (geom_bar).
```



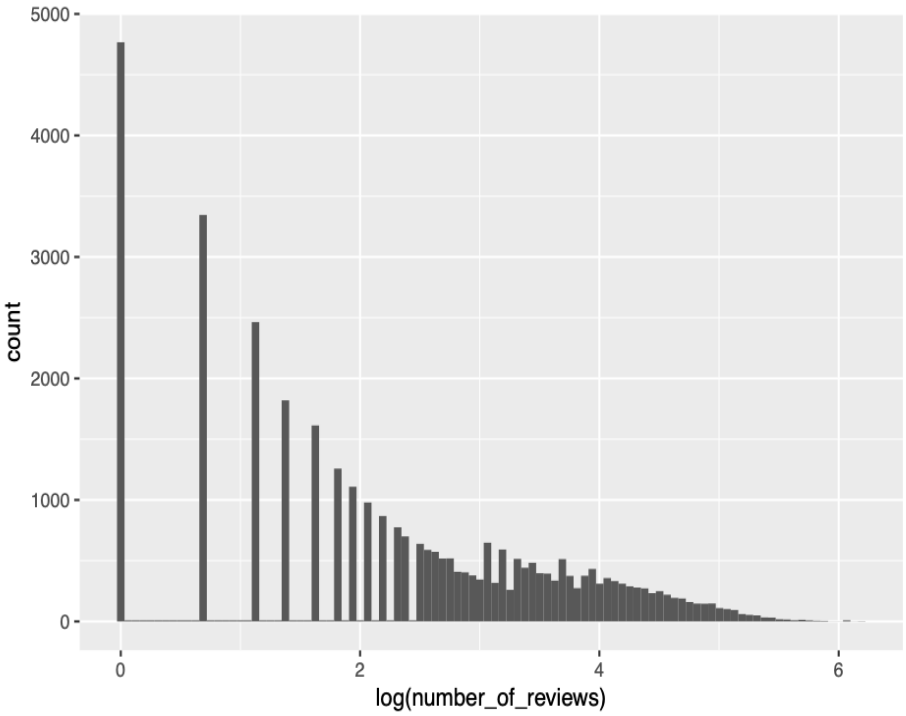

```
ggplot(nyc, aes(x = log10(reviews_per_month))) +geom_histogram(bins = 100)
```

```
## Warning: Removed 9474 rows containing non-finite values (stat_bin).
```




```
ggplot(nyc, aes(x = log(number_of_reviews))) +geom_histogram(bins = 100)
```

```
## Warning: Removed 9432 rows containing non-finite values (stat_bin).
```




```
attach(nyc)
```

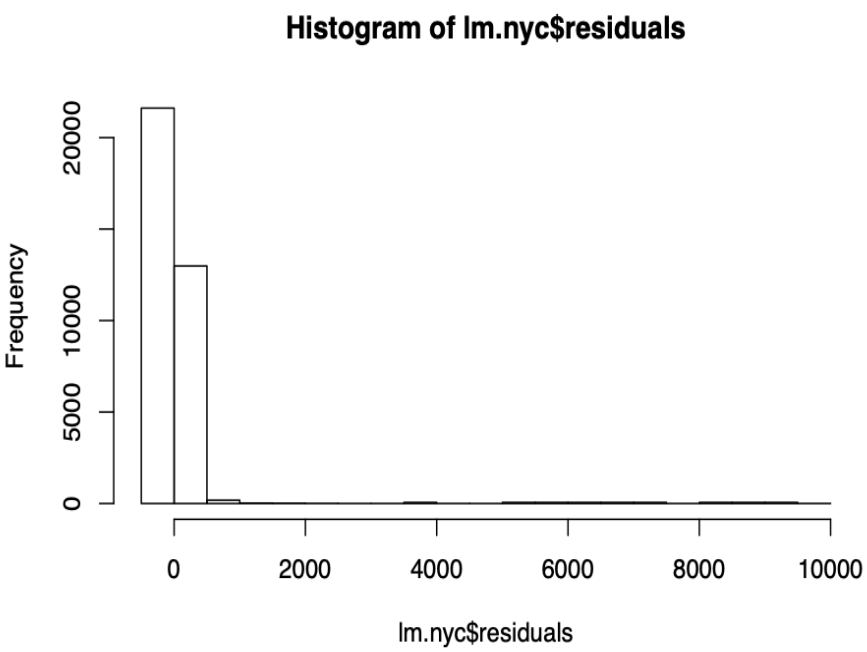
```
## The following objects are masked from airbnb:
##
##     availability_365, latitude, longitude, neighbourhood, price,
##     room_type
```

```
lm.nyc = lm( price ~ neighbourhood_group + room_type +log10(minimum_nights)+log10(reviews_per_month)+ l
summary(lm.nyc)
```

```
##
## Call:
## lm(formula = price ~ neighbourhood_group + room_type + log10(minimum_nights) +
##     log10(reviews_per_month) + log(number_of_reviews) + availability_365)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -260.0  -50.4  -14.7   19.0 9973.7
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                     1.308e+02  7.355e+00  17.788  < 2e-16
## neighbourhood_groupBrooklyn     3.595e+01  6.996e+00   5.139 2.78e-07
## neighbourhood_groupManhattan    8.361e+01  6.999e+00  11.946  < 2e-16
## neighbourhood_groupQueens       1.960e+01  7.402e+00   2.648  0.00811
## neighbourhood_groupStaten Island -2.595e-01  1.314e+01  -0.020  0.98424
## room_typePrivate room           -1.109e+02  1.858e+00 -59.664  < 2e-16
## room_typeShared room            -1.433e+02  5.957e+00 -24.053  < 2e-16
```

```
## log10(minimum_nights)           -8.543e+00  2.668e+00  -3.201  0.00137
## log10(reviews_per_month)        -6.482e+00  2.124e+00  -3.052  0.00227
## log(number_of_reviews)          -4.771e+00  8.752e-01  -5.451 5.03e-08
## availability_365                 1.620e-01  6.977e-03  23.220  < 2e-16
##
## (Intercept)                     ***
## neighbourhood_groupBrooklyn     ***
## neighbourhood_groupManhattan    ***
## neighbourhood_groupQueens       **
## neighbourhood_groupStaten Island
## room_typePrivate room           ***
## room_typeShared room            ***
## log10(minimum_nights)           **
## log10(reviews_per_month)        **
## log(number_of_reviews)          ***
## availability_365                ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 166.8 on 34832 degrees of freedom
##   (9474 observations deleted due to missingness)
## Multiple R-squared:  0.1395, Adjusted R-squared:  0.1393
## F-statistic: 564.9 on 10 and 34832 DF,  p-value: < 2.2e-16
```

```
hist(lm.nyc$residuals)
```



The fit model shows neighbourhood_group, pivate room and shared room, as long as minimum nights, reviews, and availability throuth the year has effects on the price. However, the model can only explain 14% of the price and the residuals are not distributted normally. Further data preprocessing and analysis needed.

```
library("glmnet")
```

```
## Loading required package: Matrix
```

```
##
## Attaching package: 'Matrix'
```

```
## The following object is masked from 'package:tidyr':
##
##     expand
```

```
## Loading required package: foreach
```

```
##
## Attaching package: 'foreach'
```

```
## The following objects are masked from 'package:purrr':
##
##     accumulate, when
```

```
## Loaded glmnet 2.0-16
```

```
library("mvtnorm")
nyc$latitude = NULL
nyc$longitude= NULL
nyc$last_review= NULL
nyc = na.omit(nyc)
x = model.matrix( ~ neighbourhood + neighbourhood_group + room_type + minimum_nights + number_of_review
y = nyc %>%
  select(price) %>%
  unlist() %>%
  as.numeric()
lasso = glmnet(x,y,alpha = 1)
```