Stat 6021: Homework Set 5

Michael Puchalski

2025-02-24

Set up

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.4     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

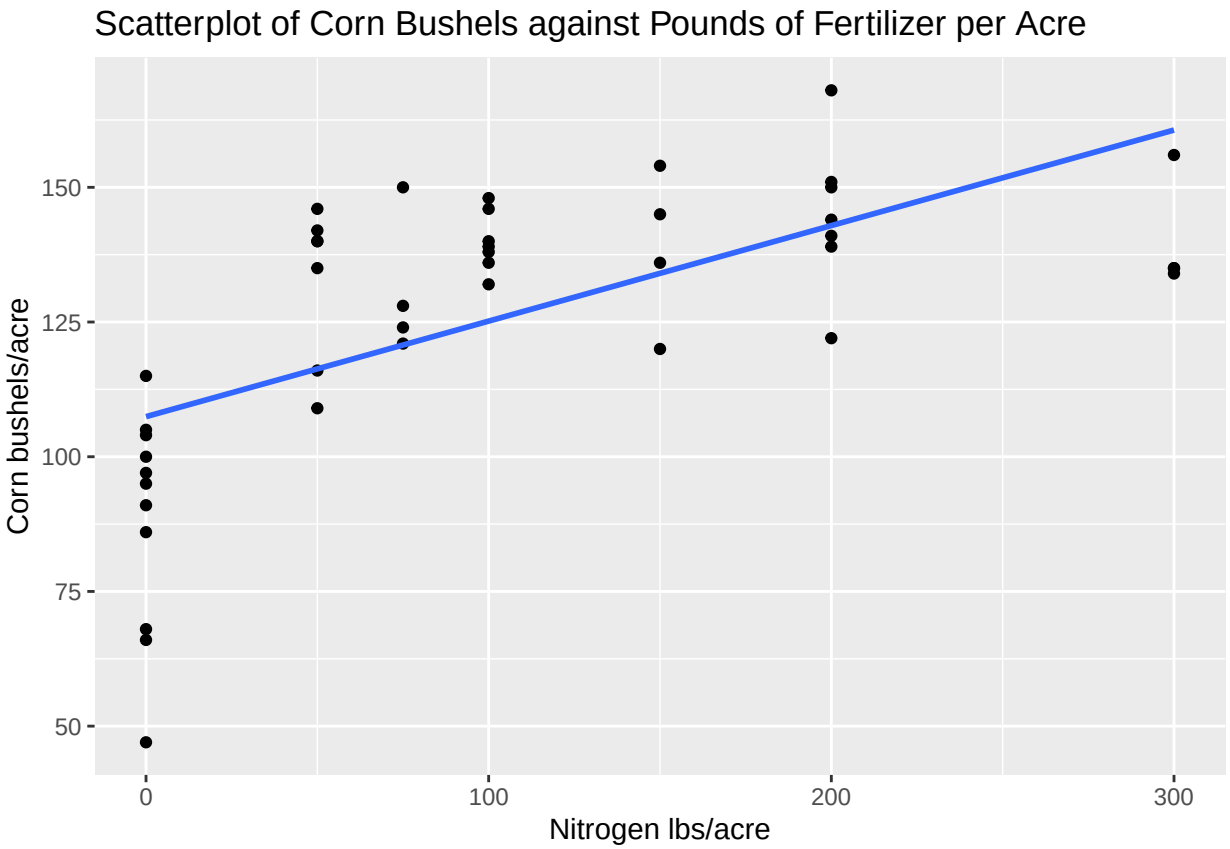
```
library(dplyr)
library(faraway)
```

```
Data = cornnit
```

1(a) The response variable for this study is the corn yield at bushels per acre with the predictor being the nitrogen at pounds per acre.

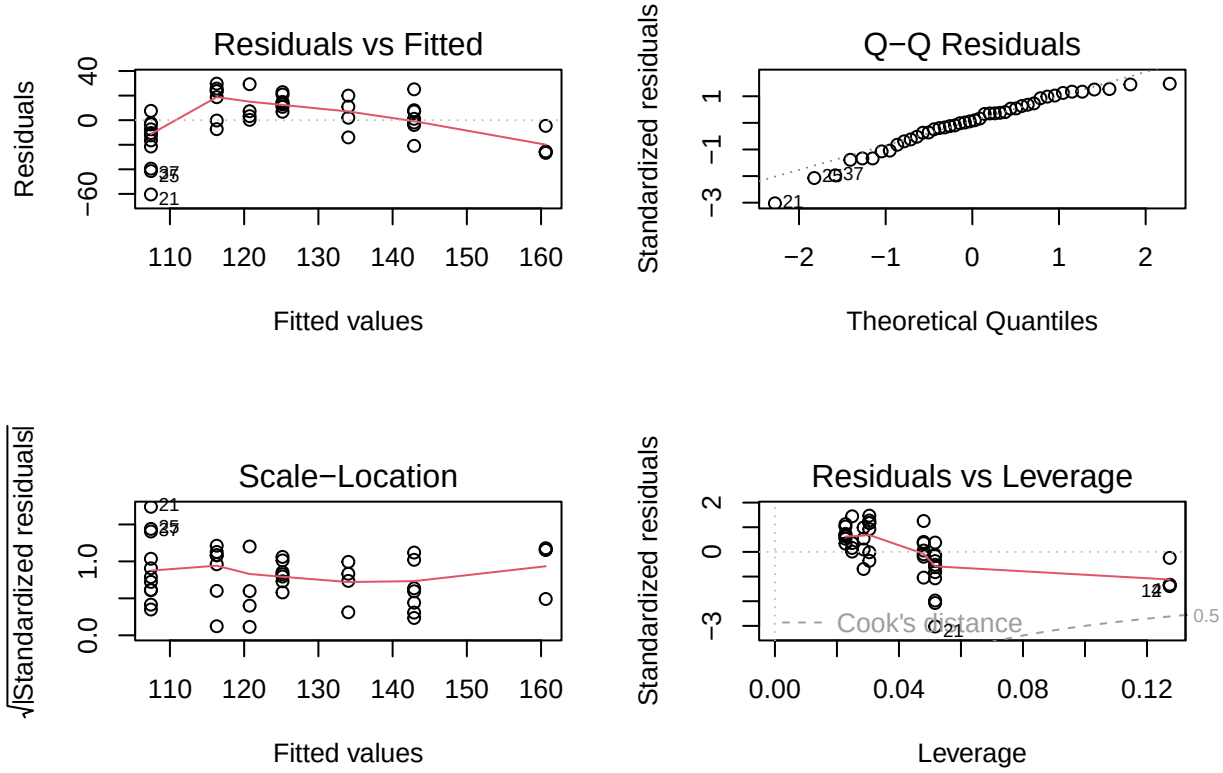
```
ggplot2::ggplot(Data, aes(x=nitrogen, y=yield))+
  geom_point()+
  geom_smooth(method = "lm", se=FALSE)+
  labs(x="Nitrogen lbs/acre", y="Corn bushels/acre", title="Scatterplot of Corn Bushels against Pounds
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



Looking at this plot, this looks like it does not going to fall evenly on both sides of a line nor does it seem like the vertical spread is completely constant as you move from left to right violating both assumption 1 and assumption 2.

```
result<-lm(yield~nitrogen, data=Data)
par(mfrow = c(2,2))
plot(result)
```



1(b)

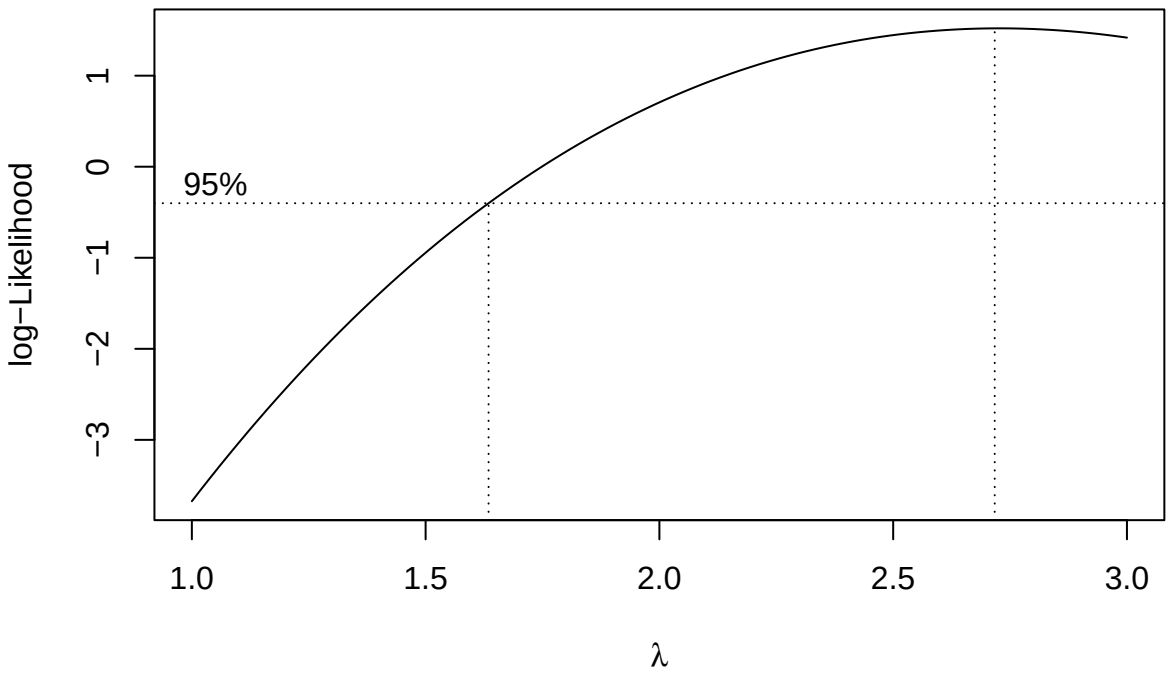
Based on the residuals plot, the red line that is fitted to the average value of the residuals for differing values along the x-axis is curved indicating the current model is not reasonable. Specifically the curvature indicates assumption 1 is violated. For assumption 2, the vertical spread of the residuals appears to be close to, but not completely constant as the move from left to right occurs.

Based on this finding, we should first attempt to address the vertical variance (assumption 2) via a lambda greater than 1 because we see a decrease in variance as we go from left to right. To address this, we will look at a boxcox plot with the ideal being that we can do a logarithmic normalization.

```
library(MASS)
```

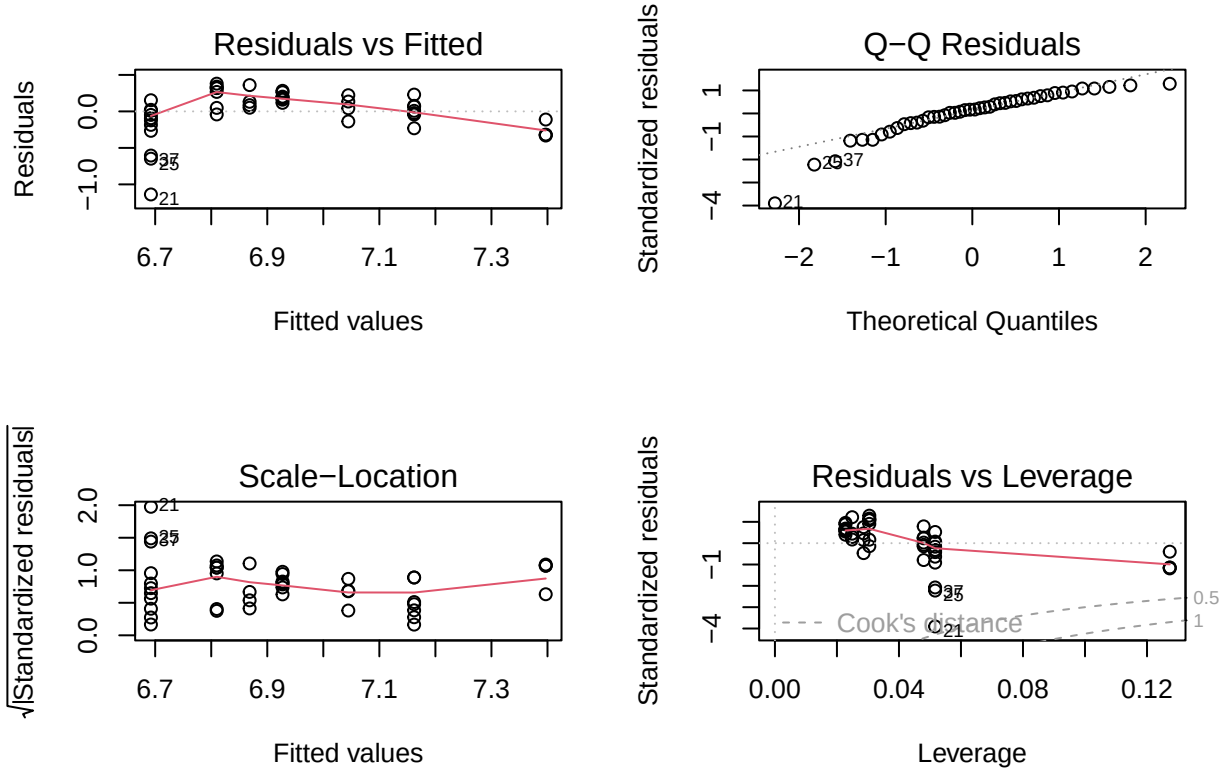
1(c)

```
##
## Attaching package: 'MASS'
## The following object is masked from 'package:dplyr':
##
##     select
MASS::boxcox(result, lambda= seq(1,3,1/10))
```



Here, I am going to choose to set lambda equal to 2 for general ease of use with it falling within the 95% CI range. This means that I am going to transform the y at a factor of log base 2. This plot aids in guiding transformation by giving a range of values in which the log modifier should fall between.

```
ystar<-log2(Data$yield)
Data<- data.frame(Data,ystar)
result.ystar<-lm(ystar~nitrogen, data=Data)
par(mfrow=c(2,2))
plot(result.ystar)
```

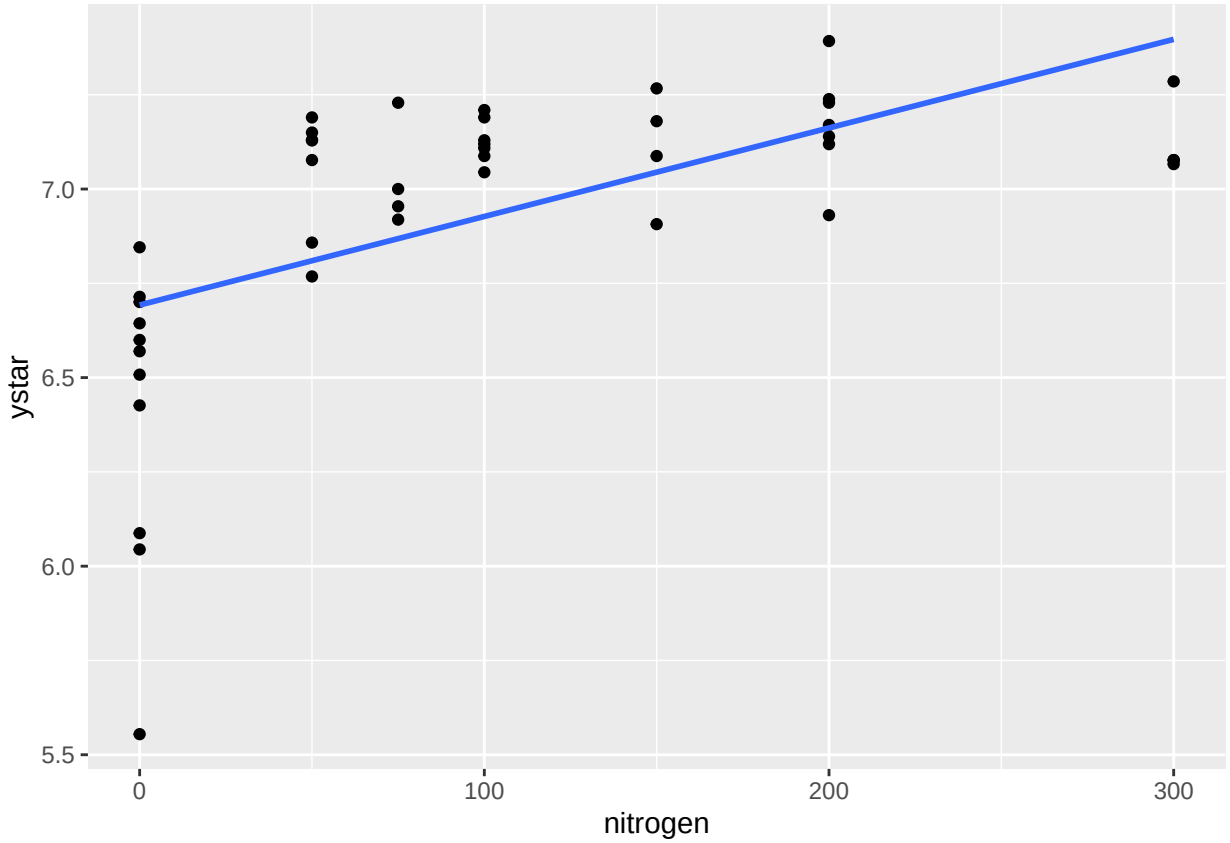


1(d)

Based on the resulting residual plot from the initial log base 2 transformation, it appears the variance has been more normalized; however, the curvature of the residual plot still implies that we need to still adjust for assumption 1. This means we will need to transform it in such a way, that we preserve the changes made to address assumption 2.

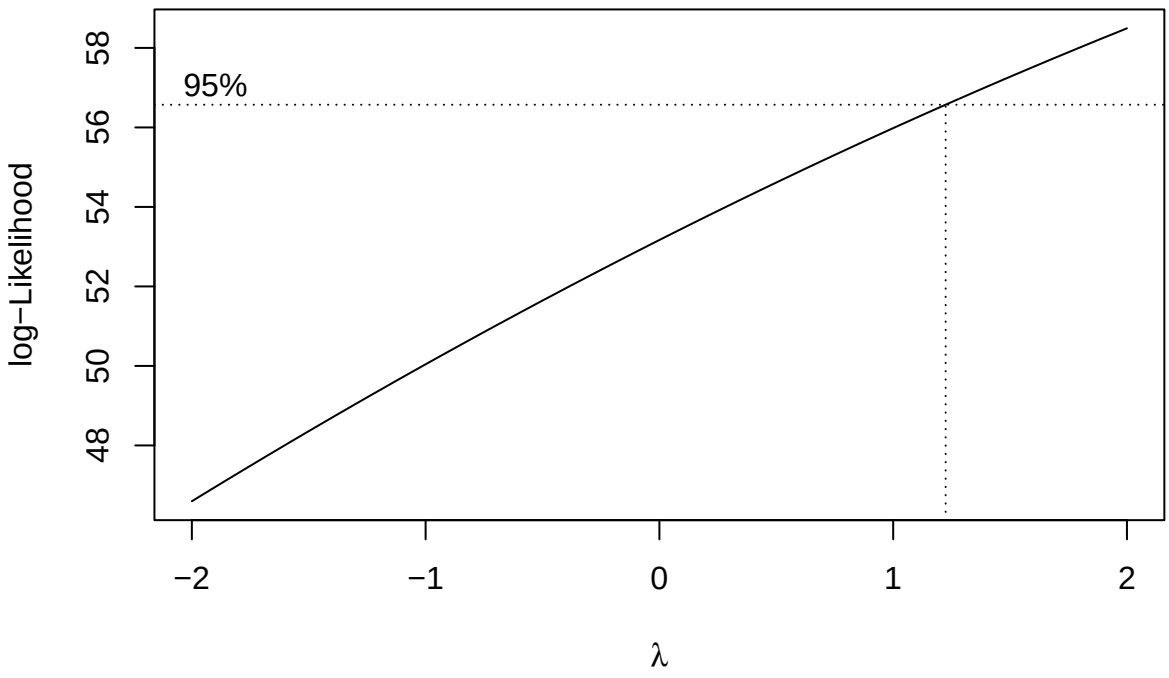
```
ggplot2::ggplot(data=Data, aes(x=nitrogen, y=ystar))+
  geom_point()+
  geom_smooth(method=lm, se=FALSE)
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



Based on the shape of the log2 of y against nitrogen, this is either shaped as an upside down parabola, square root, or log(x). Based on the preference of the log transformation for maintaining the integrity of the interpretation of the model, that is the first transformation that will be attempted.

```
result2<-lm(ystar~nitrogen, data=Data)
MASS::boxcox(result2)
```




```
xstar<-log2(Data$nitrogen)
Data<- data.frame(Data,xstar)
Data
```

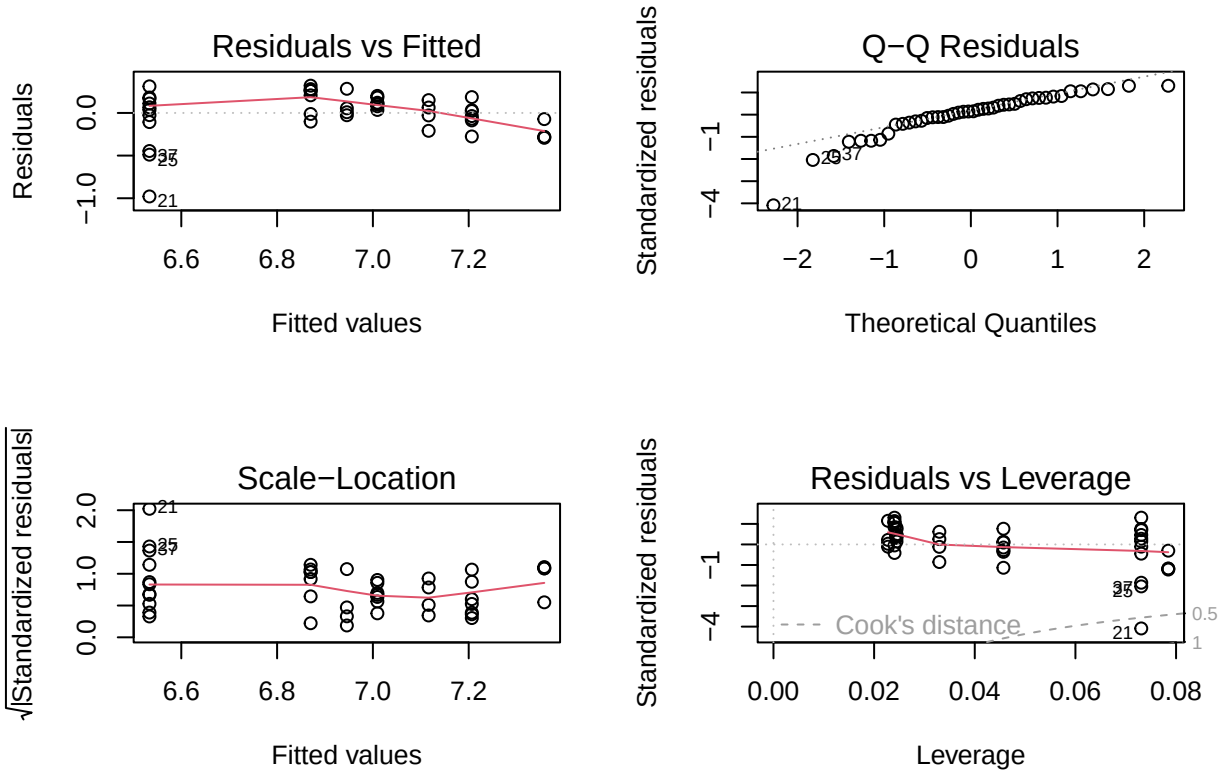
```
##    yield nitrogen    ystar    xstar
## 1    115        0 6.845490     -Inf
## 2    128       75 7.000000 6.228819
## 3    136      150 7.087463 7.228819
## 4    135      300 7.076816 8.228819
## 5     97        0 6.599913     -Inf
## 6    150       75 7.228819 6.228819
## 7    154      150 7.266787 7.228819
## 8    156      300 7.285402 8.228819
## 9     95        0 6.569856     -Inf
## 10   121       75 6.918863 6.228819
## 11   120      150 6.906891 7.228819
## 12   134      300 7.066089 8.228819
## 13    91        0 6.507795     -Inf
## 14   124       75 6.954196 6.228819
## 15   145      150 7.179909 7.228819
## 16   135      300 7.076816 8.228819
## 17   105        0 6.714246     -Inf
## 18   140       50 7.129283 5.643856
## 19   138      100 7.108524 6.643856
## 20   139      200 7.118941 7.643856
## 21    47        0 5.554589     -Inf
## 22   140       50 7.129283 5.643856
```

```
## 23   132      100 7.044394 6.643856
## 24   151      200 7.238405 7.643856
## 25    66        0 6.044394     -Inf
## 26   109       50 6.768184 5.643856
## 27   136      100 7.087463 6.643856
## 28   144      200 7.169925 7.643856
## 29    86        0 6.426265     -Inf
## 30   135       50 7.076816 5.643856
## 31   139      100 7.118941 6.643856
## 32   150      200 7.228819 7.643856
## 33   100        0 6.643856     -Inf
## 34   146       50 7.189825 5.643856
## 35   148      100 7.209453 6.643856
## 36   168      200 7.392317 7.643856
## 37    68        0 6.087463     -Inf
## 38   116       50 6.857981 5.643856
## 39   146      100 7.189825 6.643856
## 40   122      200 6.930737 7.643856
## 41   104        0 6.700440     -Inf
## 42   142       50 7.149747 5.643856
## 43   140      100 7.129283 6.643856
## 44   141      200 7.139551 7.643856
```

```
# result.xstar<-lm(ystar~xstar, data=Data)
# par(mfrow=c(2,2))
# plot(result.xstar)
```

xstar row instead gains a -Inf value if this is given it will not allow for a plot, so instead some other transformations should be attempted.

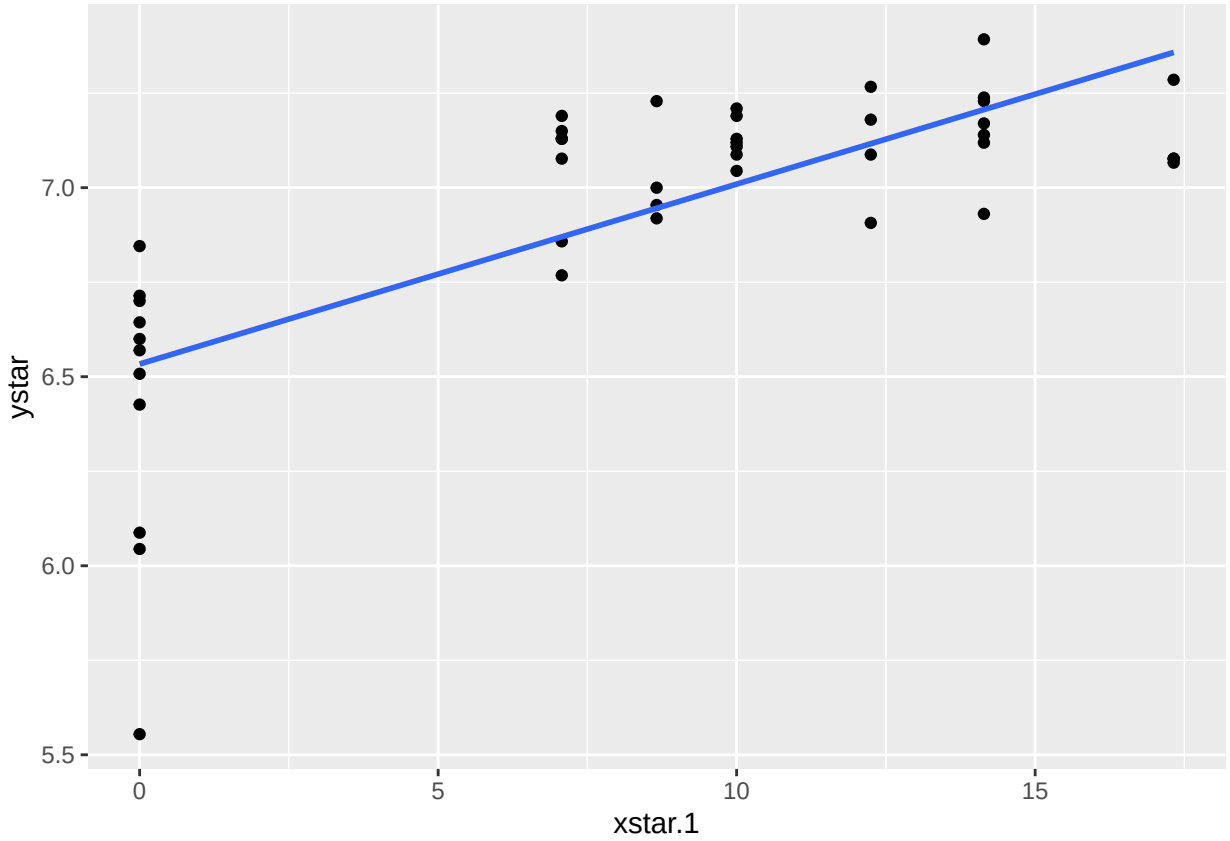
```
xstar.1<-sqrt(Data$nitrogen)
Data<- data.frame(Data,xstar.1)
result.xstar<-lm(ystar~xstar.1, data=Data)
par(mfrow=c(2,2))
plot(result.xstar)
```



Following this transformation, you see variance maintained, and a very close return close to the mean, but still not flat in the residual plot. Along with this, the QQ-residuals plot suggests a form of normal distribution, but there still appears to be some work to be done as some values fall heavily off the main line in the plot.

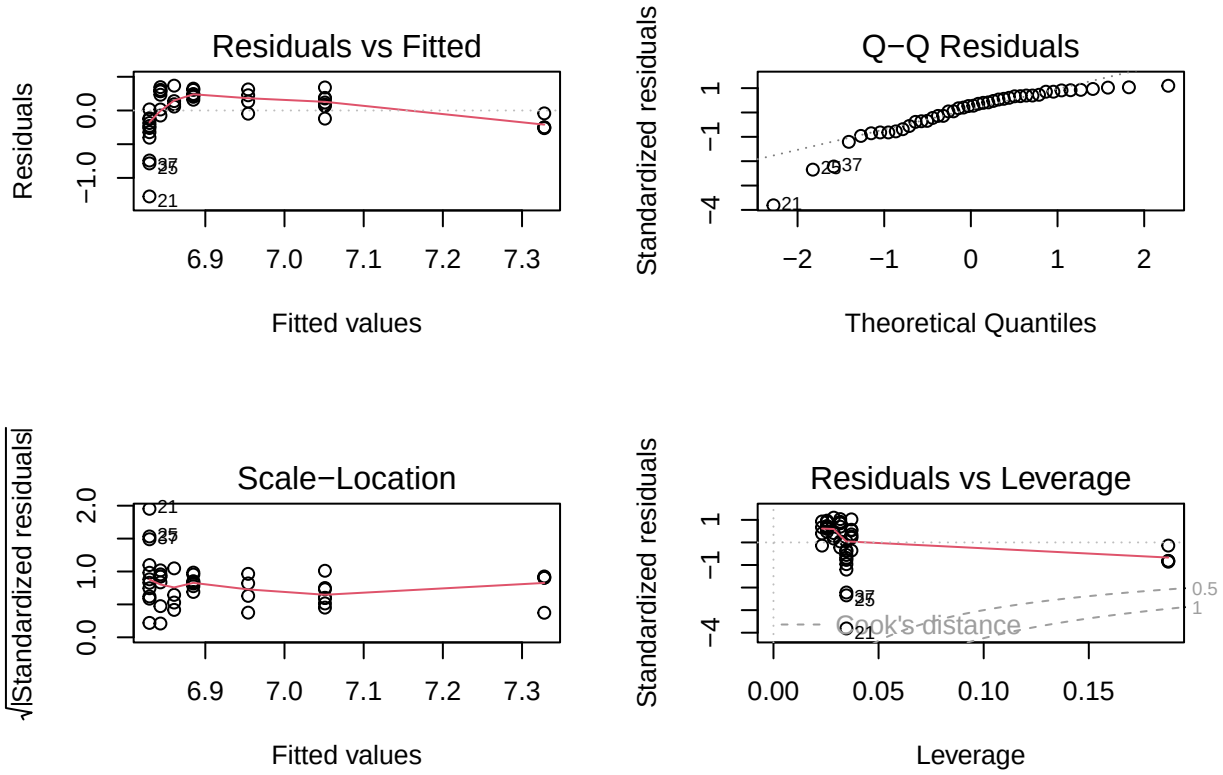
```
ggplot2::ggplot(data=Data, aes(x=xstar.1, y=ystar))+
  geom_point()+
  geom_smooth(method=lm, se=FALSE)
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



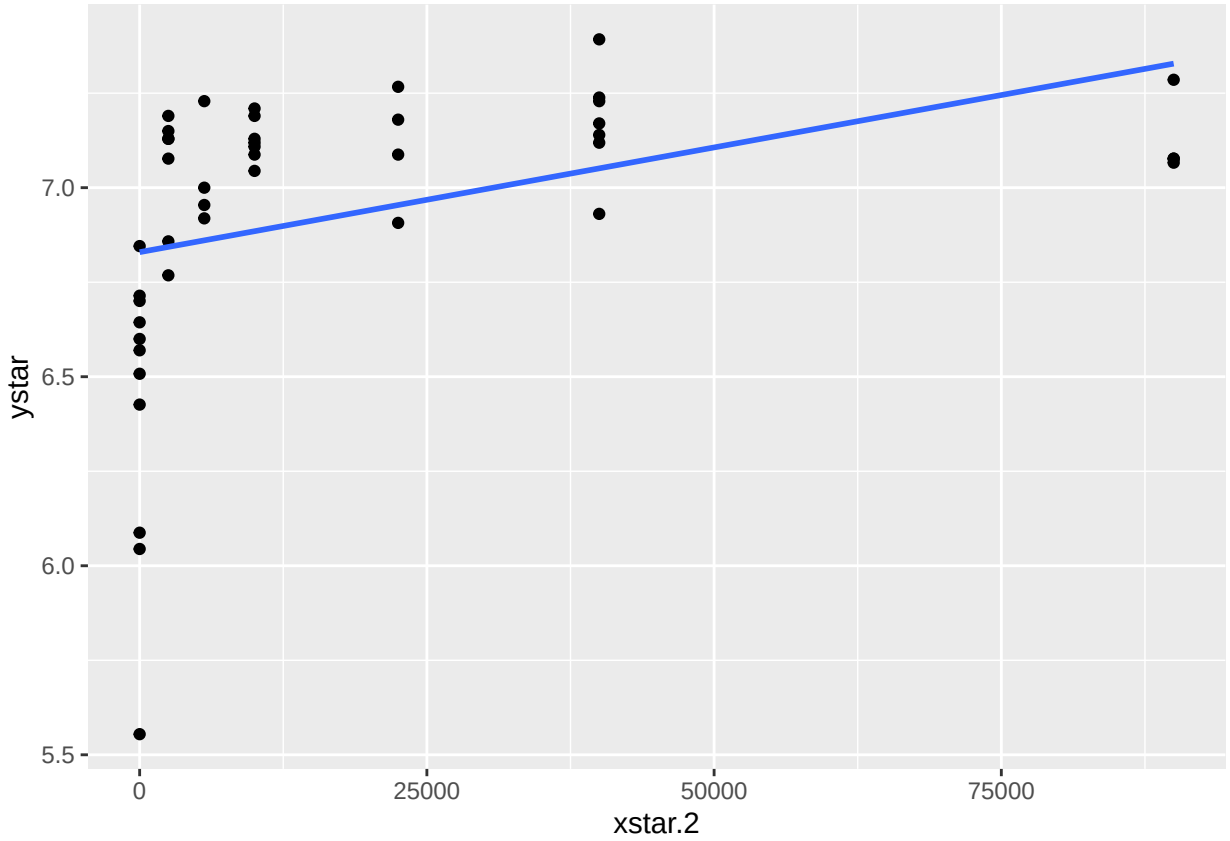
Following that analysis, it can then also be confirmed that this still does not correct the model to fit assumption 1 as the data is still overestimated or underestimated by the line on the right side of the plot. Following this, I will test a negative square due to plot appear to potentially have a negative parabola shap.

```
xstar.2<-(Data$nitrogen)^2
Data<- data.frame(Data,xstar.2)
result.xstar.2<-lm(ystar~xstar.2, data=Data)
par(mfrow=c(2,2))
plot(result.xstar.2)
```



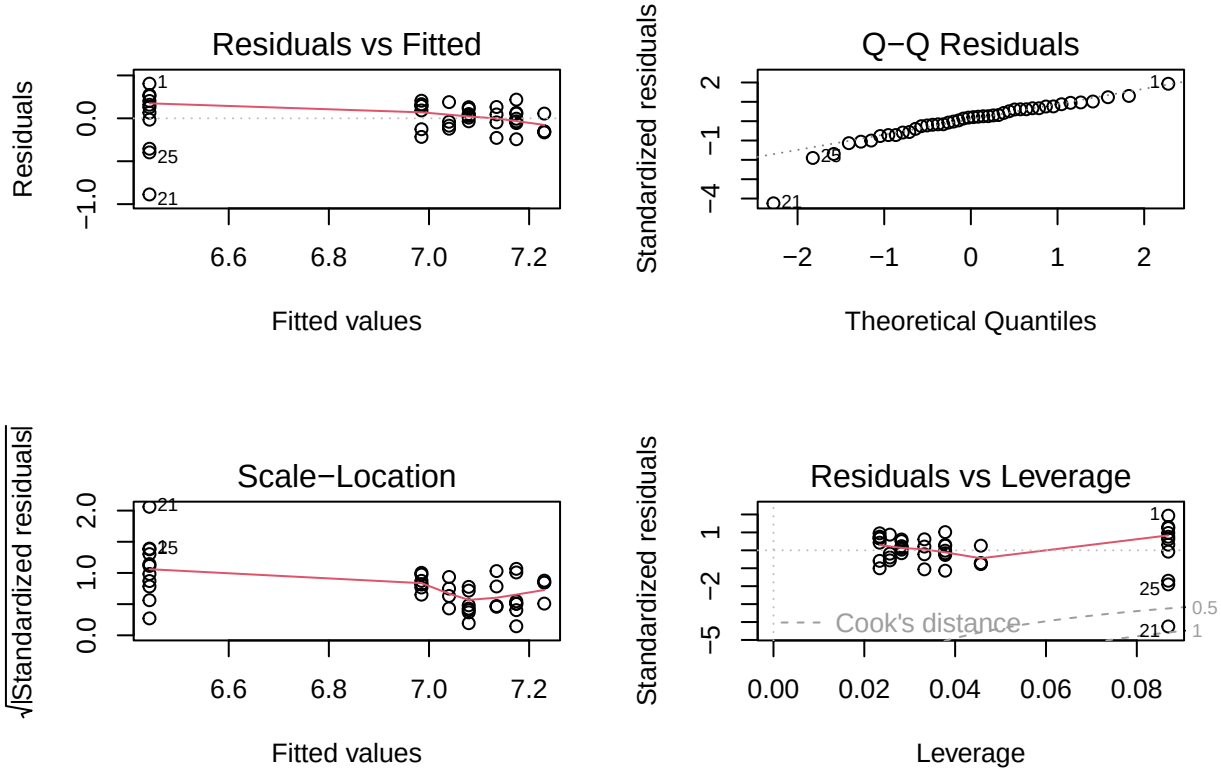

```
ggplot2::ggplot(data=Data, aes(x=xstar.2, y=ystar))+
  geom_point()+
  geom_smooth(method=lm, se=FALSE)
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



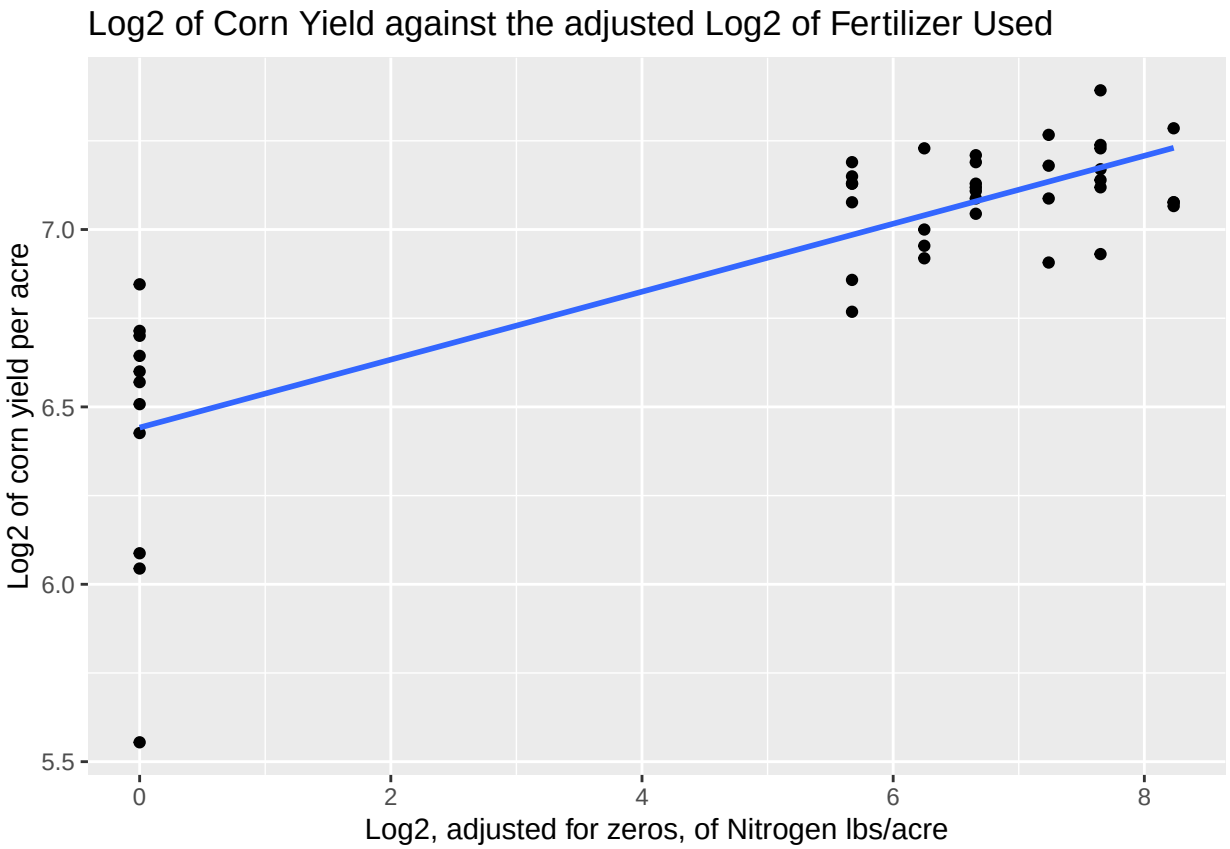
. . . Yeah, so that was wrong and made a worse model. Instead, it seems my first attempt would have been correct and instead I need to compensate for the values of nitrogen that equal 0 in the data frame.

```
nitrogen_adjusted<-Data$nitrogen+abs(min(Data$nitrogen)+1)
xstar.3<-log2(nitrogen_adjusted)
Data<- data.frame(Data,xstar.3)
result.xstar.3<-lm(ystar~xstar.3, data=Data)
par(mfrow=c(2,2))
plot(result.xstar.3)
```




```
ggplot2::ggplot(data=Data, aes(x=xstar.3, y=ystar))+
  geom_point()+
  geom_smooth(method=lm, se=FALSE)+
  labs(x="Log2, adjusted for zeros, of Nitrogen lbs/acre", y="Log2 of corn yield per acre", title="Log2
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



Following this transformation, the values maintained an equal variance along either side of the regression model, however, the distribution along the x-axis of the scatterplot and on the residual plot is incredibly changed with it populating more towards the right side of the x-axis. In this model **x= *log2(x+abs(min(x))+1), something that needed to be done to compensate for the presence of 0's as values for nitrogen. Along with that*** y=log2(y). The Model for this is:

```
lm(formula = ystar~xstar.3, data=Data)
```

```
##
## Call:
## lm(formula = ystar ~ xstar.3, data = Data)
##
## Coefficients:
## (Intercept)      xstar.3
##     6.44166      0.09577
```

In this Interpretation, for a 1% increase in the pounds of nitrogen increased, you see a 0.09577% increase in the yield of bushels of corn. *This analysis feels rocky at best, and I feel like there is a better way to state this.*

2(a) Based on Figure 1, I would advise we adjust the response variable first for vertical variance. This is because based on the residual plot, even with the values appearing mostly up and down, their mean still lies at zero due to the variance of the response variable. This goes along with the general rule that the response variable should be adjusted for first, before you adjust the predictor variable.

2(b) I agree with my classmate. With lambda of 0 falling inside of the boxcox interval, applying the log is a valid option, and is a strong one as a log transformation maintains the integrity of the model when using it for things such as prediction and hypothesis testing.

2(c) The formula of this model would be y-hat* = 1.507892 - 0.44993x, *where* y^*=log(y[the Concentration of the Solution]) and x = time. This can be interpreted as the concentration starts at a value of 1.507892, and decreasing at a rate of (1.01)^0.44993 per 1% increase of time. This is better interpreted as the concentration decreasing by 0.44993% per 1% increase in time it spends in the solution.