Excersises 1 N

Data Visualization: Flights at ABIA

Our question was "How does the arrival delay/departure delay vary by month?" and "How does the arrival delay/departure delay vary by day of the week?".

```
library(mosaic)
```

```
## Warning: package 'mosaic' was built under R version 3.5.2
## Loading required package: dplyr
## Warning: package 'dplyr' was built under R version 3.5.2
##
## Attaching package: 'dplyr'
## The following objects are masked from 'package:stats':
##
##     filter, lag
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
## Loading required package: lattice
## Loading required package: ggformula
## Warning: package 'ggformula' was built under R version 3.5.2
## Loading required package: ggplot2
## Warning: package 'ggplot2' was built under R version 3.5.2
## Loading required package: ggstance
## Warning: package 'ggstance' was built under R version 3.5.2
##
## Attaching package: 'ggstance'
## The following objects are masked from 'package:ggplot2':
##
##     geom_errorbarh, GeomErrorbarh
##
## New to ggformula?  Try the tutorials:
##  learnr::run_tutorial("introduction", package = "ggformula")
##  learnr::run_tutorial("refining", package = "ggformula")
## Loading required package: mosaicData
## Loading required package: Matrix
##
## The 'mosaic' package masks several functions from core packages in order to add
## additional features.  The original behavior of these functions should not be affected by this.
##
## Note: If you use the Matrix package, be sure to load it BEFORE loading mosaic.
```

```
##
## Attaching package: 'mosaic'

## The following object is masked from 'package:Matrix':
##
##     mean

## The following object is masked from 'package:ggplot2':
##
##     stat

## The following objects are masked from 'package:dplyr':
##
##     count, do, tally

## The following objects are masked from 'package:stats':
##
##     binom.test, cor, cor.test, cov, fivenum, IQR, median, prop.test,
##     quantile, sd, t.test, var

## The following objects are masked from 'package:base':
##
##     max, mean, min, prod, range, sample, sum
```

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 3.5.2

## -- Attaching packages ---------------------------------------- tidyverse 1.3.0 --

## v tibble  2.1.3     v purrr   0.3.3
## v tidyr   1.0.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0

## Warning: package 'tibble' was built under R version 3.5.2

## Warning: package 'tidyr' was built under R version 3.5.2

## Warning: package 'purrr' was built under R version 3.5.2

## Warning: package 'stringr' was built under R version 3.5.2

## Warning: package 'forcats' was built under R version 3.5.2

## -- Conflicts ------------------------------------------- tidyverse_conflicts() --
## x mosaic::count()            masks dplyr::count()
## x purrr::cross()             masks mosaic::cross()
## x mosaic::do()               masks dplyr::do()
## x tidyr::expand()            masks Matrix::expand()
## x dplyr::filter()            masks stats::filter()
## x ggstance::geom_errorbarh() masks ggplot2::geom_errorbarh()
## x dplyr::lag()               masks stats::lag()
## x tidyr::pack()              masks Matrix::pack()
## x mosaic::stat()             masks ggplot2::stat()
## x mosaic::tally()            masks dplyr::tally()
## x tidyr::unpack()            masks Matrix::unpack()
```

```
library(knitr)
```

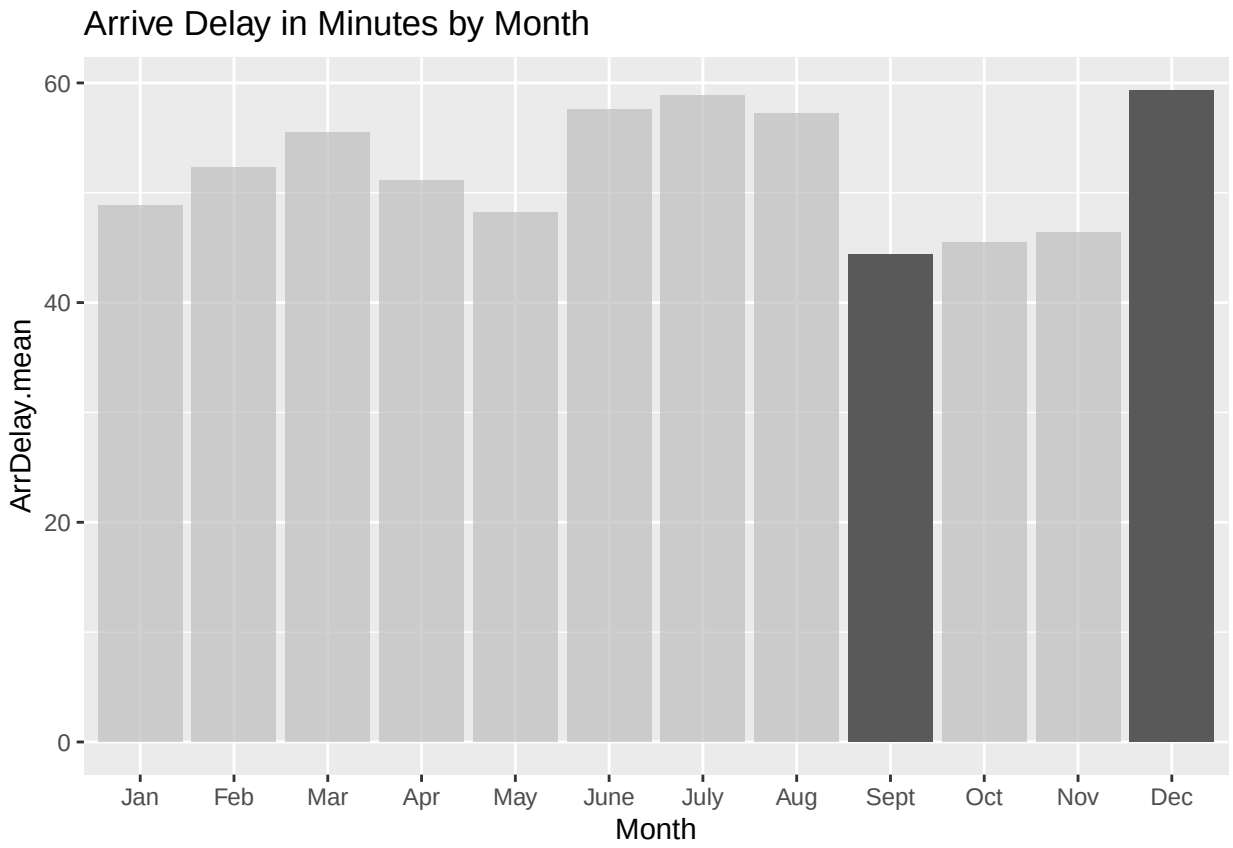
```
## Warning: package 'knitr' was built under R version 3.5.2
```

```r
#install.packages(???gghighlight???)
library(gghighlight)
```

```
## Warning: package 'gghighlight' was built under R version 3.5.2
```

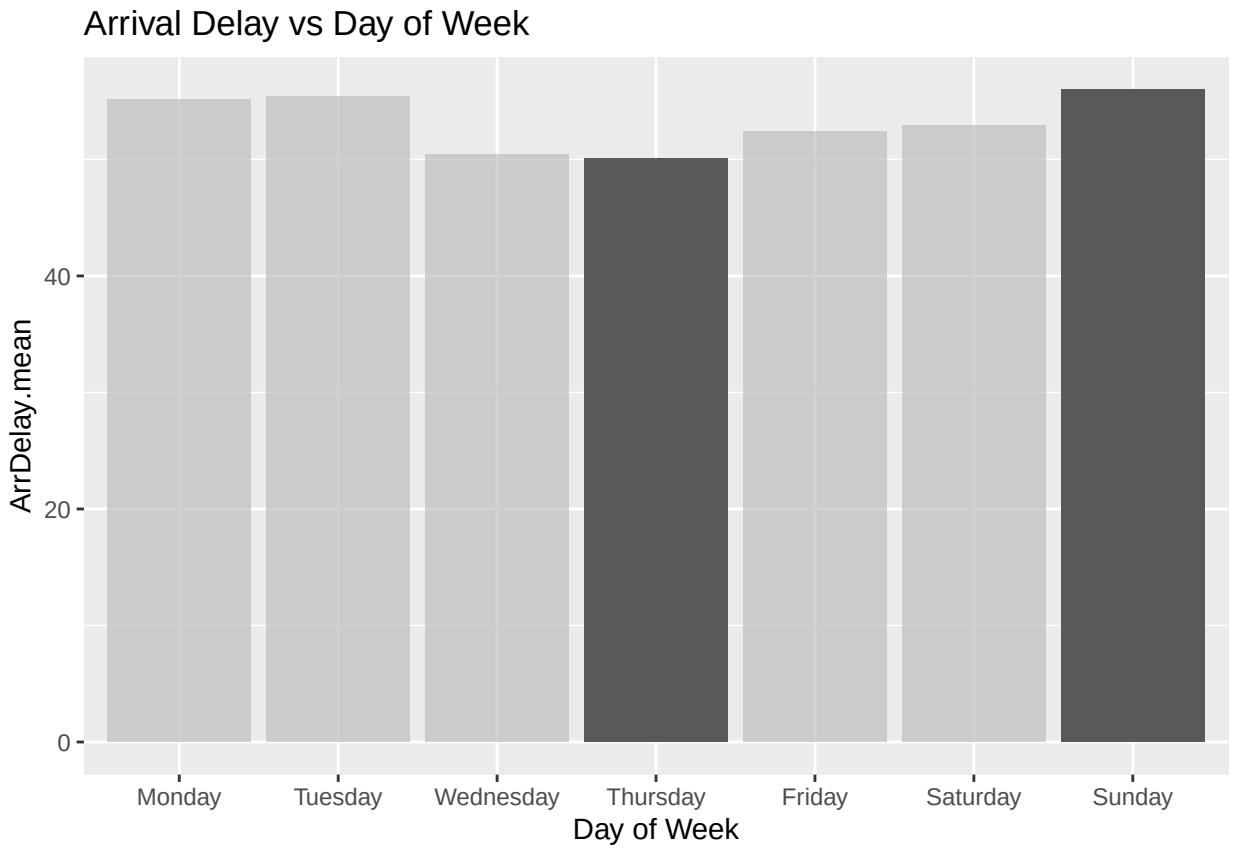
```r
ABIA <- read.csv("ABIA.csv")

ABIA_New = na.omit(ABIA)

#Recode DayOfWeek Variable
ABIA_New$DayOfWeek[ABIA_New$DayOfWeek == "1"] <- "Monday"
ABIA_New$DayOfWeek[ABIA_New$DayOfWeek == "2"] <- "Tuesday"
ABIA_New$DayOfWeek[ABIA_New$DayOfWeek == "3"] <- "Wednesday"
ABIA_New$DayOfWeek[ABIA_New$DayOfWeek == "4"] <- "Thursday"
ABIA_New$DayOfWeek[ABIA_New$DayOfWeek == "5"] <- "Friday"
ABIA_New$DayOfWeek[ABIA_New$DayOfWeek == "6"] <- "Saturday"
ABIA_New$DayOfWeek[ABIA_New$DayOfWeek == "7"] <- "Sunday"


#Recode Month Variable
ABIA_New$Month[ABIA_New$Month == "1"] <- "Jan"
ABIA_New$Month[ABIA_New$Month == "2"] <- "Feb"
ABIA_New$Month[ABIA_New$Month == "3"] <- "Mar"
ABIA_New$Month[ABIA_New$Month == "4"] <- "Apr"
ABIA_New$Month[ABIA_New$Month == "5"] <- "May"
ABIA_New$Month[ABIA_New$Month == "6"] <- "June"
ABIA_New$Month[ABIA_New$Month == "7"] <- "July"
ABIA_New$Month[ABIA_New$Month == "8"] <- "Aug"
ABIA_New$Month[ABIA_New$Month == "9"] <- "Sept"
ABIA_New$Month[ABIA_New$Month == "10"] <- "Oct"
ABIA_New$Month[ABIA_New$Month == "11"] <- "Nov"
ABIA_New$Month[ABIA_New$Month == "12"] <- "Dec"

#Taking the means of ArrDelay by Month
Arr_summ_Month = ABIA_New %>%
  group_by(Month)  %>%
  summarize(ArrDelay.mean = mean(ArrDelay))

#ArrDelay vs Month Barchart
ggplot(data = Arr_summ_Month) +
  geom_bar(mapping = aes(x = Month, y = ArrDelay.mean), stat='identity') +
  scale_x_discrete(name ="Month",
                   limits=c("Jan","Feb","Mar", "Apr",
                            "May", "June", "July", "Aug", "Sept",
                            "Oct", "Nov", "Dec")) +
  ggtitle("Arrive Delay in Minutes by Month") +
  gghighlight(ArrDelay.mean < 45 | ArrDelay.mean > 59)
```

```
## label_key: Month
```




```
#Taking the means of ArrDelay by DayOfWeek
Arr_summ_DOW = ABIA_New %>%
  group_by(DayOfWeek)  %>%
  summarize(ArrDelay.mean = mean(ArrDelay))

#ArrDelay vs DayOfWeek Barchart
ggplot(data = Arr_summ_DOW) +
  geom_bar(mapping = aes(x = DayOfWeek, y = ArrDelay.mean),
           stat='identity') +
  scale_x_discrete(name ="Day of Week",
                   limits=c("Monday","Tuesday","Wednesday", "Thursday",
                            "Friday", "Saturday", "Sunday")) +
  ggtitle("Arrival Delay vs Day of Week") +
  gghighlight(ArrDelay.mean < 50.1 | ArrDelay.mean > 55.9)
```

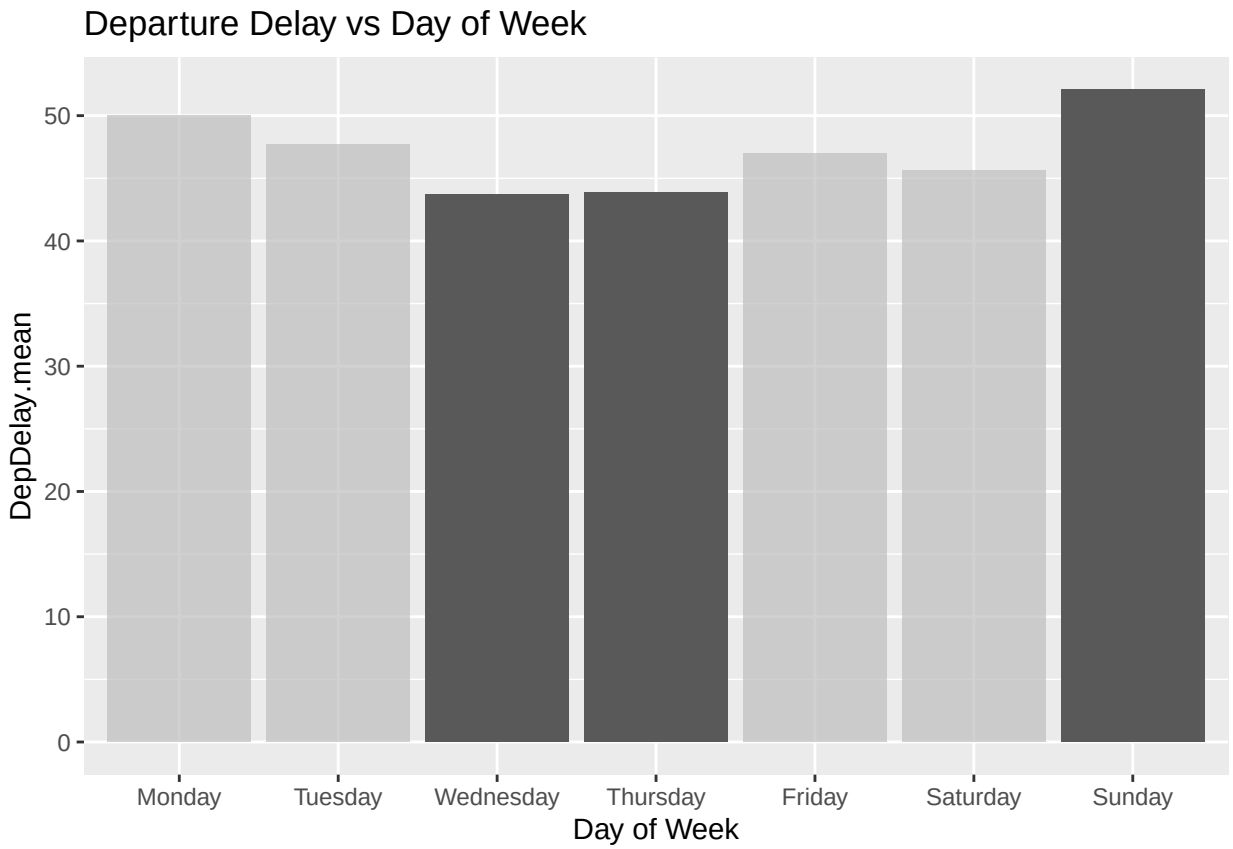
```
## label_key: DayOfWeek
```



For arrival delays, September had the smallest average delay time in minutes while December had the highest average delay time in minutes for all months. For days of the week, Sunday averaged the highest minutes of delay while Thursday averaged the lowest.

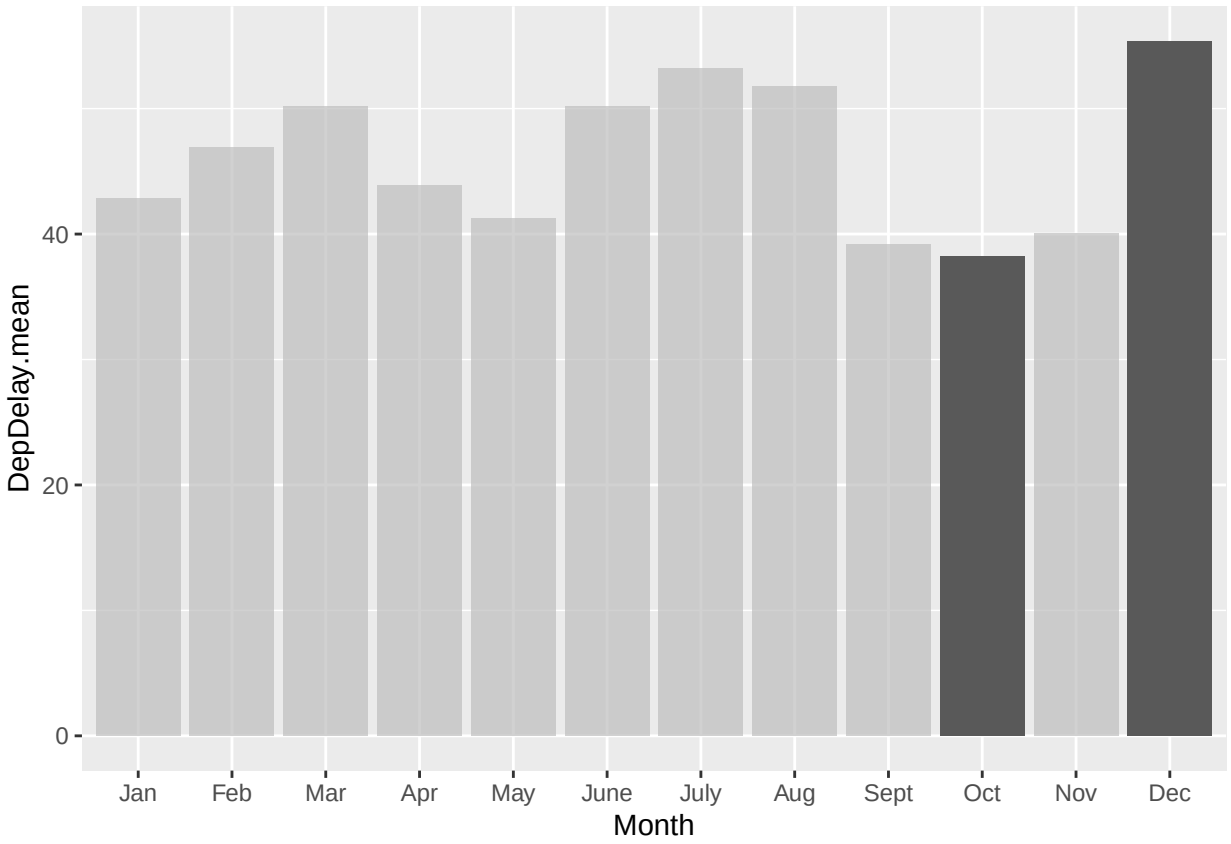
```
#Taking the means of DepDelay by DayOfWeek
Dep_summ_DOW = ABIA_New %>%
  group_by(DayOfWeek)  %>%
  summarize(DepDelay.mean = mean(DepDelay))

#DepDelay vs DayOfWeek Barchart
ggplot(data = Dep_summ_DOW) +
  geom_bar(mapping = aes(x = DayOfWeek, y = DepDelay.mean),
           stat='identity') +
  scale_x_discrete(name ="Day of Week",
                   limits=c("Monday","Tuesday","Wednesday", "Thursday",
                            "Friday", "Saturday", "Sunday")) +
  ggtitle("Departure Delay vs Day of Week") +
  gghighlight(DepDelay.mean < 44 | DepDelay.mean > 51)
```

```
## label_key: DayOfWeek
```




```
#Taking the means of DepDelay by Month
Dep_summ_Month = ABIA_New %>%
  group_by(Month)  %>%
  summarize(DepDelay.mean = mean(DepDelay))

#DepDelay vs Month chart
ggplot(data = Dep_summ_Month) +
  geom_bar(mapping = aes(x = Month, y = DepDelay.mean), stat='identity') +
  scale_x_discrete(name ="Month",
                   limits=c("Jan","Feb","Mar", "Apr",
                            "May", "June", "July", "Aug", "Sept",
                            "Oct", "Nov", "Dec")) +
    gghighlight(DepDelay.mean < 39 | DepDelay.mean > 55)
```

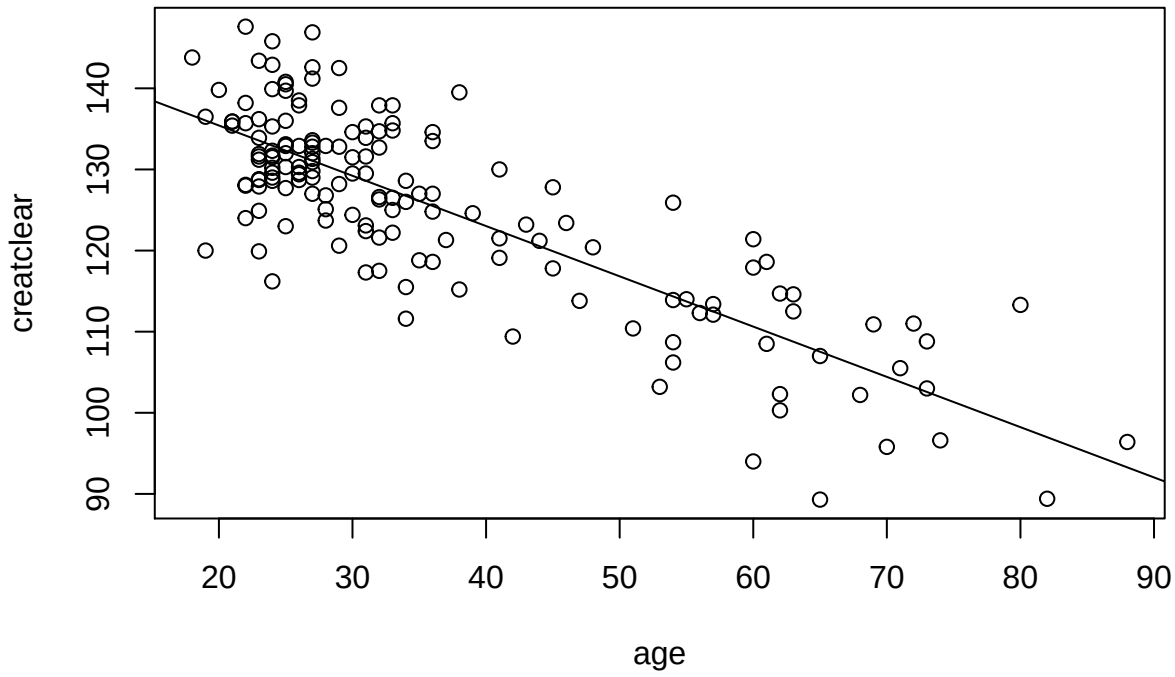
```
## label_key: Month
```



As for departure delays, Wednesday and Thursday had the lowest average delay time in minutes while Sunday had the highest. Departure Delays differed from Arrival Delays per month. The lowest average delay time in minutes was in October and the highest was in December. One of the interesting facts was that all of the days of the week had a similar average delay time.

Regression Practice

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

cremodel<-lm(creatclear~age, data = creatinine)
plot(creatclear~age, data = creatinine)
abline(cremodel)
```




```
#Slope = -0.61982
summary(cremodel)
```

```
##
## Call:
## lm(formula = creatclear ~ age, data = creatinine)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -18.2249  -4.6175   0.2221   4.7212  15.8221
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 147.81292    1.37965  107.14   <2e-16 ***
## age          -0.61982    0.03475  -17.84   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.911 on 155 degrees of freedom
## Multiple R-squared:  0.6724, Adjusted R-squared:  0.6703
## F-statistic: 318.2 on 1 and 155 DF,  p-value: < 2.2e-16
```

```
predict.lm(cremodel, list(age = 55))
```

```
##       1
## 113.723
```

```
predict.lm(cremodel, list(age = 40))
```

```
##        1
## 123.0203
```

```
predict.lm(cremodel, list(age = 60))
```

```
##       1
```

```
## 110.624
```

After creating a linear regression model of creatinine clearance rate vs age, we estimated the average creatinine clearance rate for a 55-year-old to be approximately 114.

From our model, we get a slope of -0.61982 which means that the average creatinine clearance rate decreases by approximately 0.62 ml/minute per 1 year increase in age.

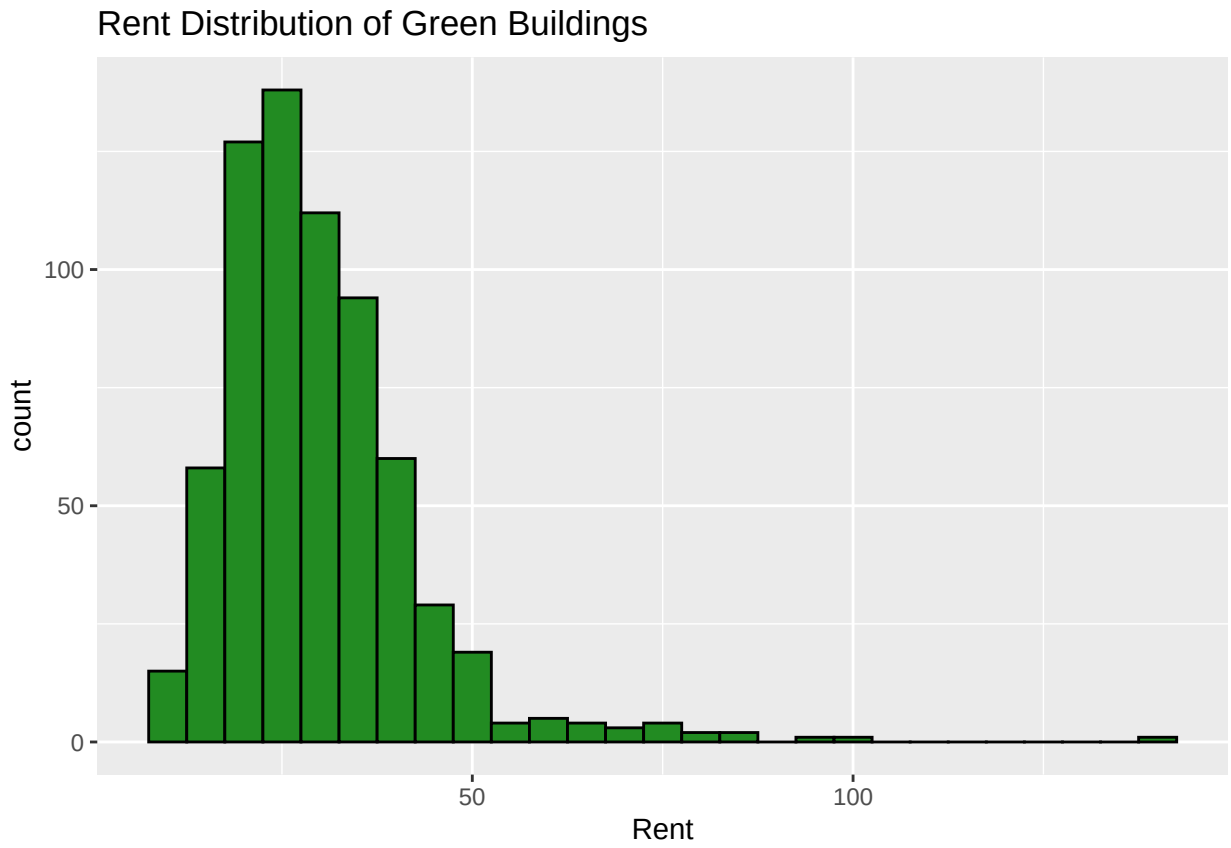
Whose creatinine clearance rate is healthier (higher) for their age: a 40-year-old with a rate of 135, or a 60-year-old with a rate of 112? Our model predicted that the creatinine clearance rate for a 40-year-old and 60-year-old would be 123.02 ml/minute and 110.2 ml/minute, respectively. Based on these values, the 40-year-old is healthier since he/she has a higher clearance rate for his/her age than the 60-year-old.

Green Buildings

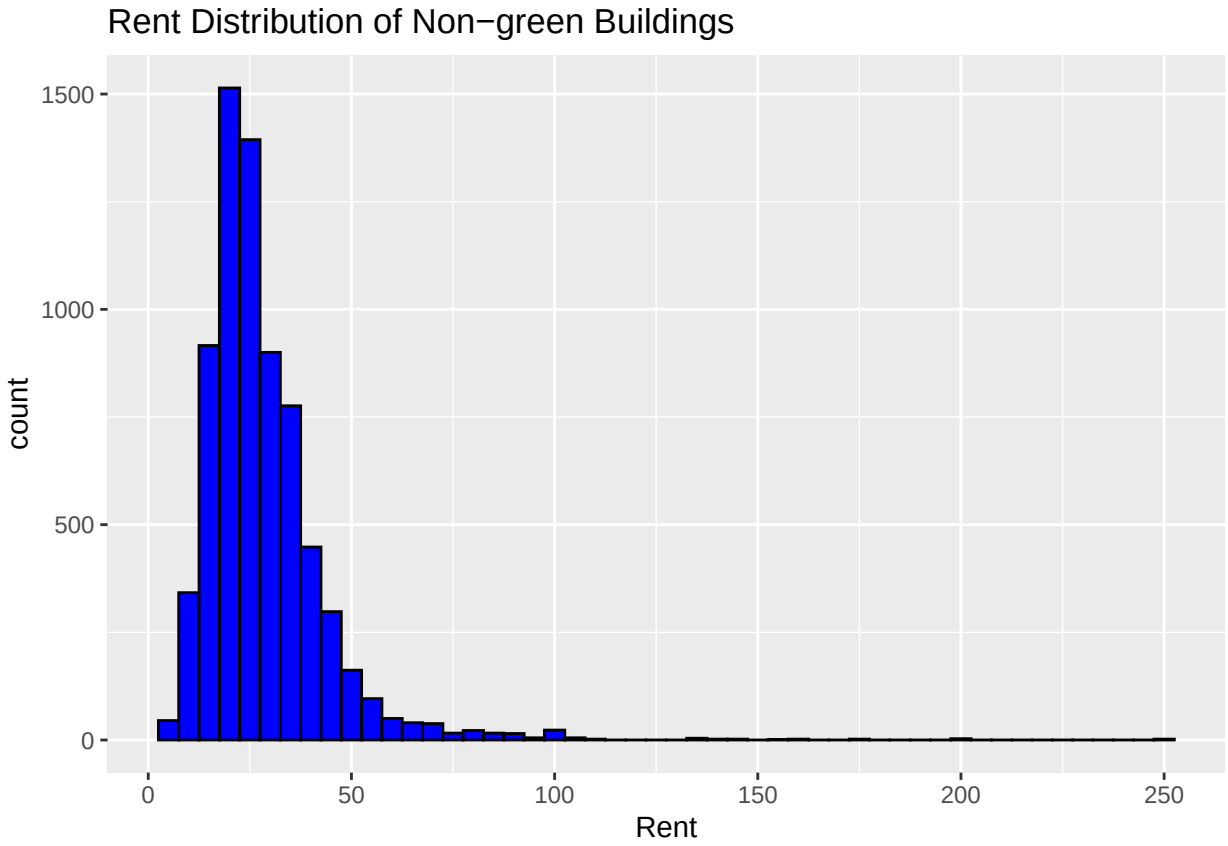
```
#install.packages("tidyverse")
library(tidyverse)
library(mosaic)
greenbuildings <- read.csv("greenbuildings.csv")
GB = read.csv("greenbuildings.csv", header = T)

GB_New = na.omit(GB)
```

To start our analysis, we first wanted to compare the cost of both types of buildings. We plotted the distribution of rent for both buildings to see if mean or median was the appropriate statistic to report.

```
#distribution of rent for green buildings
ggplot(data = GB_New[GB_New$green_rating == 1,]) +
  geom_histogram(mapping = aes(x = Rent), binwidth = 5, fill = "forestgreen",
                 color = "black") + scale_fill_manual(values=fill) +
  ggtitle("Rent Distribution of Green Buildings")
```




```
#distribution of rent for non-green buildings
ggplot(data = GB_New[GB_New$green_rating == 0,]) +
  geom_histogram(mapping = aes(x = Rent), binwidth = 5, fill = "blue",
                 color = "black") +
  ggtitle("Rent Distribution of Non-green Buildings")
```



Both distributions appeared skewed, so we agreed with the data guru and used median to compare the rent of both buildings.

```
#median rent for green and non-green buildings
GB_New %>%
  group_by(green_rating) %>%
  summarize(median_rent = median(Rent))
```

```
## # A tibble: 2 x 2
##   green_rating median_rent
##          <int>       <dbl>
## 1            0        25
## 2            1        27.6
```

We found the median rent for green buildings to be $27.60 per square foot per year and $25.00 per square foot per year for non-green buildings. We then repeated the steps of the data guru to remove the buildings with less than 10% leasing rate since these buildings could have extenuating circumstances.

```
#copying analysis of data guru, removing leasing rate below 10 does not change median
greenbuildings %>%
  group_by(green_rating) %>%
  filter(leasing_rate >= 10) %>%
  summarize(median_rent = median(Rent))
```

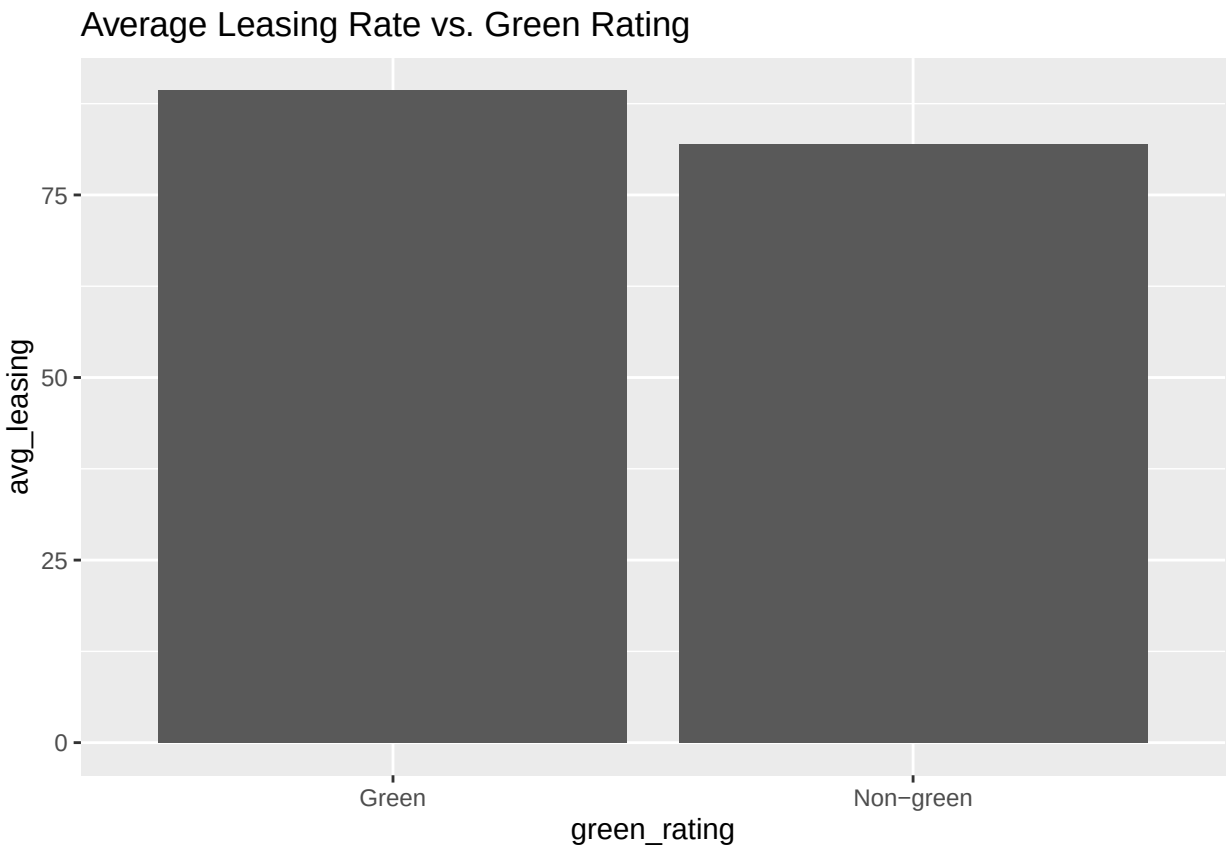
```
## # A tibble: 2 x 2
##   green_rating median_rent
##          <int>       <dbl>
## 1            0        25.0
## 2            1        27.6
```

We discovered removing these buildings hardly changed the median rent. It only increased the median rent of non-green buildings by 3 cents. We concluded this step of their analysis was useless.

```
gleasing <- greenbuildings %>%
  group_by(green_rating) %>%
  summarize(avg_leasing = mean(leasing_rate))
gleasing
```

```
## # A tibble: 2 x 2
##   green_rating avg_leasing
##          <int>       <dbl>
## 1            0        82.0
## 2            1        89.3
```

```
gleasing$green_rating[gleasing$green_rating == "0"] <- "Non-green"
gleasing$green_rating[gleasing$green_rating == "1"] <- "Green"

ggplot(data = gleasing) +
  geom_bar(mapping = aes(x = green_rating, y = avg_leasing), stat = "identity") +
  ggtitle("Average Leasing Rate vs. Green Rating")
```



One variable to consider when deciding between green and non-green buildings is to look at the leasing rate. Green buildings have a leasing rate that is about 8% higher than non-green buildings.

```
#percent class a and class b, separated by green and non-green
greenbuildings %>%
  group_by(green_rating) %>%
  summarize(pct_a = mean(class_a) * 100, pct_b = mean(class_b) * 100)
```

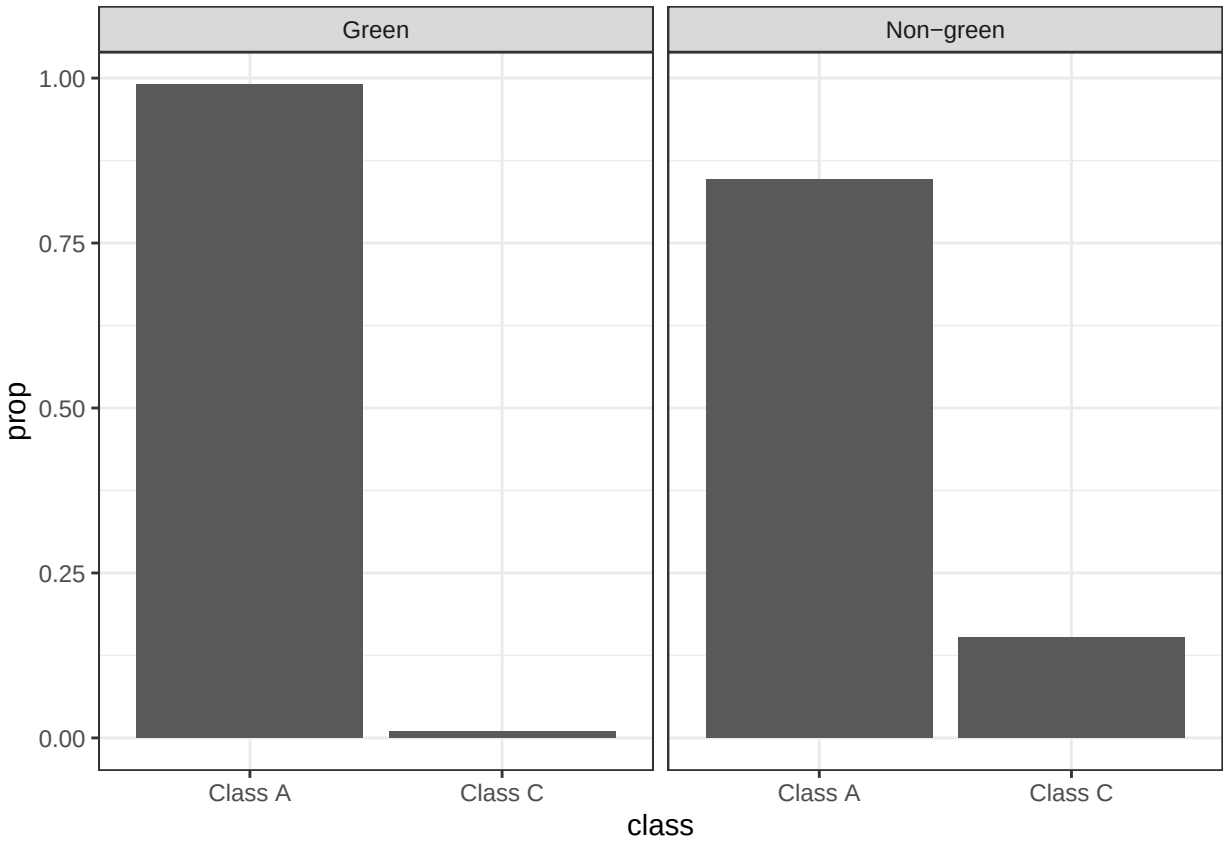
```
## # A tibble: 2 x 3
```

```
##   green_rating pct_a pct_b
##          <int> <dbl> <dbl>
## 1            0  36.2  48.5
## 2            1  79.7  19.3
```

```
#converted class b to the number 2 to combine into one variable
GB_New$class_b[GB_New$class_b == "1"] <- 2

#created new variable that included class b and class a ratings
g1 = greenbuildings %>%
  mutate(class = class_a + class_b)

#changed numbers to words in variable and included class c; changed green_rating to words for visual
g1$class[g1$class == "2"] <- "Class B"
g1$class[g1$class == "1"] <- "Class A"
g1$class[g1$class == "0"] <- "Class C"

xtabs(~class + green_rating, data=g1) %>%
  prop.table(margin=2)
```

```
##          green_rating
## class               0         1
##   Class A 0.84699681 0.98978102
##   Class C 0.15300319 0.01021898
```

```
g1$green_rating[g1$green_rating == "0"] <- "Non-green"
g1$green_rating[g1$green_rating == "1"] <- "Green"

ggplot(data = g1) +
  geom_bar(mapping = aes(x = class, y = ..prop.., group = 1)) +
  facet_wrap(green_rating~.) +
  theme_bw()
```

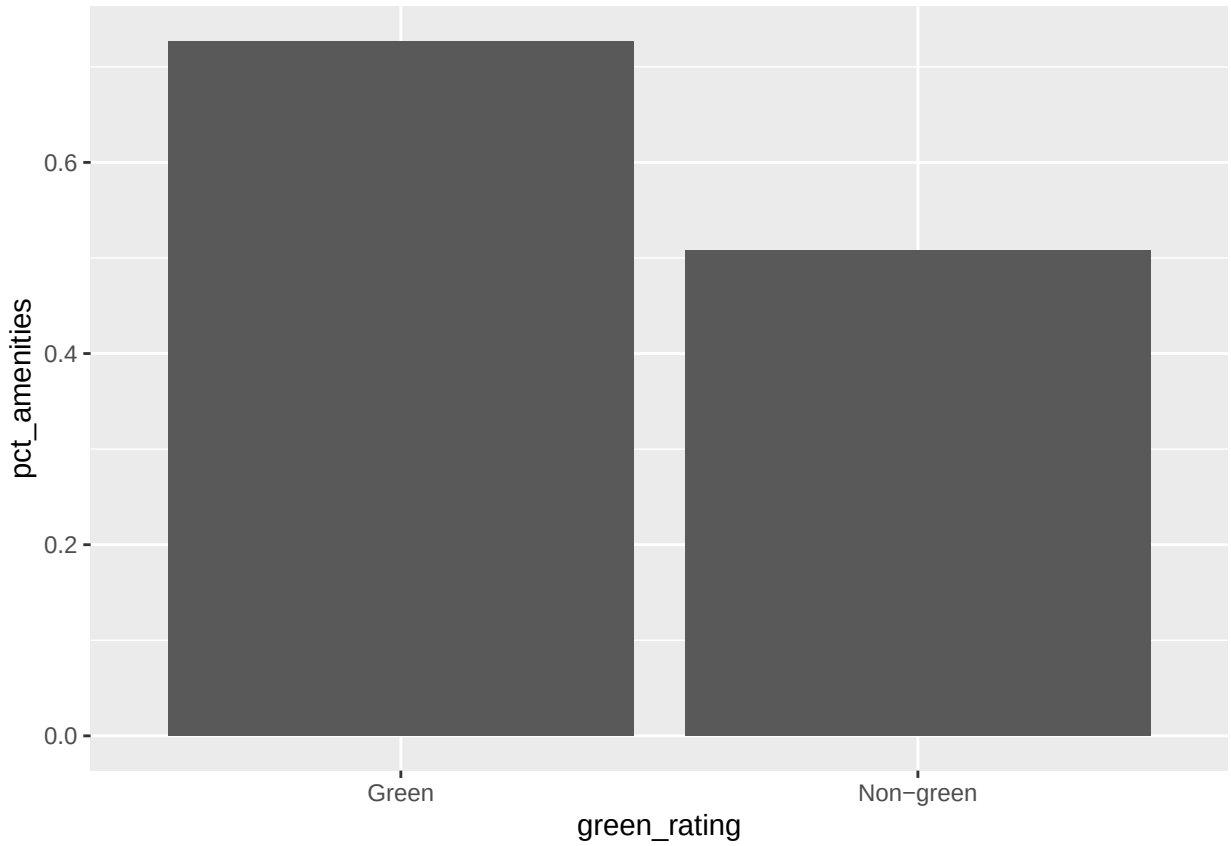


We found that 79.7% of green buildings are rated as class a, while only 36.2% of non-green buildings are rated as class a. This could be one reason the rent in green buildings is higher than non-green buildings. The quality of green buildings are generally higher and could lead to higher customer satisfaction.

```
#percent amenities for green and non-green buildings
g2 = GB %>%
  group_by(green_rating) %>%
  summarize(pct_amenities = mean(amenities), na.rm =TRUE)
g2
```

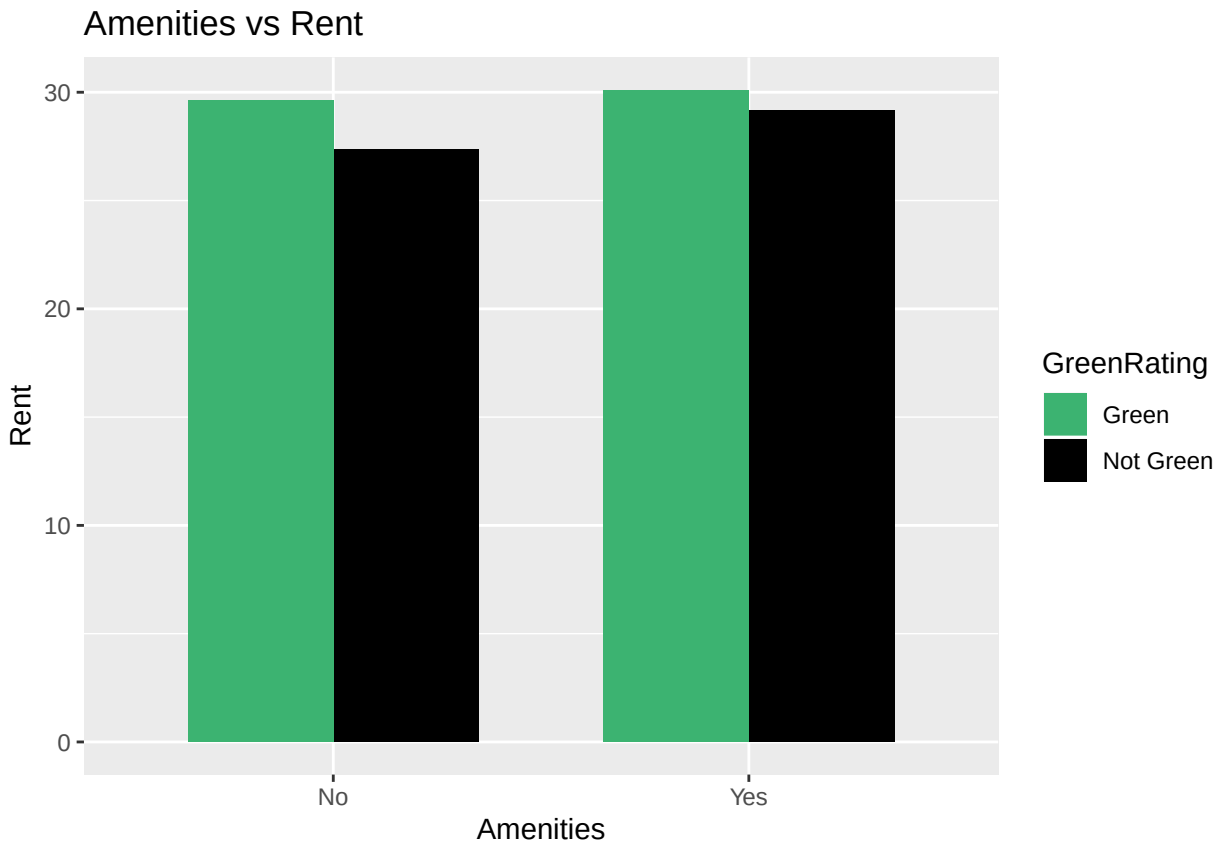
```
## # A tibble: 2 x 3
##   green_rating pct_amenities na.rm
##          <int>         <dbl> <lgl>
## 1            0         0.508 TRUE
## 2            1         0.727 TRUE
```

```
g2$green_rating[g2$green_rating == "0"] <- "Non-green"
g2$green_rating[g2$green_rating == "1"] <- "Green"

#plot for percent amenities
ggplot(data = g2) +
  geom_bar(mapping = aes(x = green_rating, y = pct_amenities), stat = 'identity')
```



We looked at another potential confounding variable, which is the presence of amenities at the building location. 72.7% of green buildings have amenities available, while only 50% of non-green buildings have them. This again could account for the difference in rent between green and non-green buildings and could be a factor that increases the leasing rate and overall profits from the building. We then went on to include rent in this comparison.

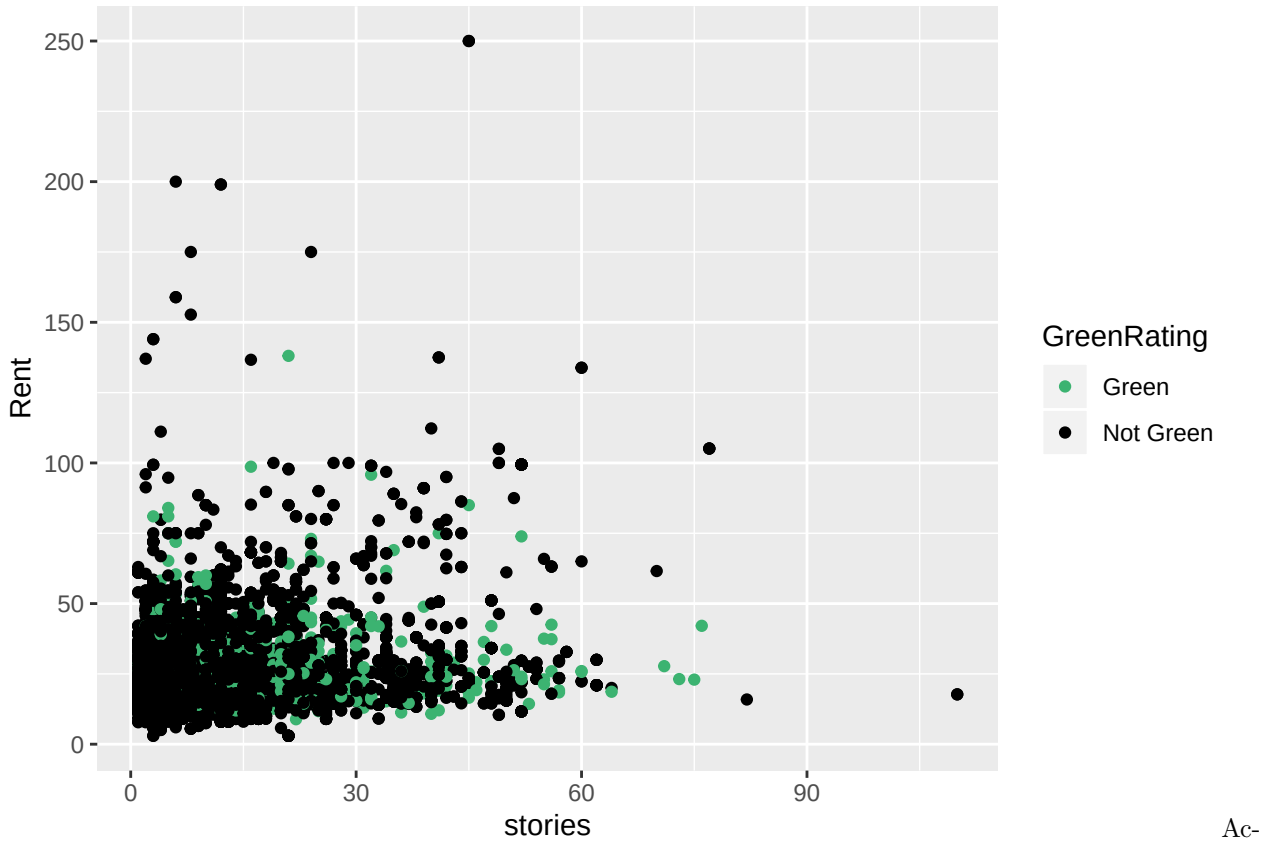
```
#RENT VS AMENITIES
GB_New$GreenRating <- rep(0,nrow(GB_New))
GB_New$GreenRating[which(GB_New$green_rating==1)]  <- "Green"
GB_New$GreenRating[which(GB_New$green_rating==0)] <- "Not Green"
fill <- c("mediumseagreen", "black")

GB_New$Amenities <- rep(0,nrow(GB_New))
GB_New$Amenities[which(GB_New$amenities==1)] <- "Yes"
GB_New$Amenities[which(GB_New$amenities==0)] <- "No"

AmenitiesChart<- ggplot(data=GB_New, aes(x=Amenities, y=Rent, fill=GreenRating)) + ggtitle("Amenities vs
  geom_bar(stat="summary", fun.y = "mean",position = position_dodge(),
           width =0.7) +  scale_fill_manual(values=fill)
AmenitiesChart
```



We compared the rent to the presence/absence of amenities. Other factors such as class, we considered if having amenities was considered when the data analyst compared the rent between green buildings and non-green buildings. However, when accounting for if a building has amenities or not, the green building does get more revenue from rent. Overall, that alone is not convincing since the other things we looked at(stories and classes) don't have green buildings as having higher rent.

The next part of our analysis compares the rent charged to tenants in the building in dollars per square foot per calendar year and the height of the building in stories. The green dots signify the green buildings and the black dots signify the non-green buildings.

```
#SCATTERPLOT
#use color  "forestgreen", "lightgoldenrod4"
fill2 <- c("mediumseagreen", "black")

GB_New$GreenRating <- rep(0,nrow(GB_New))
GB_New$GreenRating[which(GB_New$green_rating==1)]  <- "Green"
GB_New$GreenRating[which(GB_New$green_rating==0)] <- "Not Green"

ggplot(GB_New, aes(x=stories, y=Rent, color = GreenRating )) + geom_point() +
  scale_color_manual(values = fill2)
```

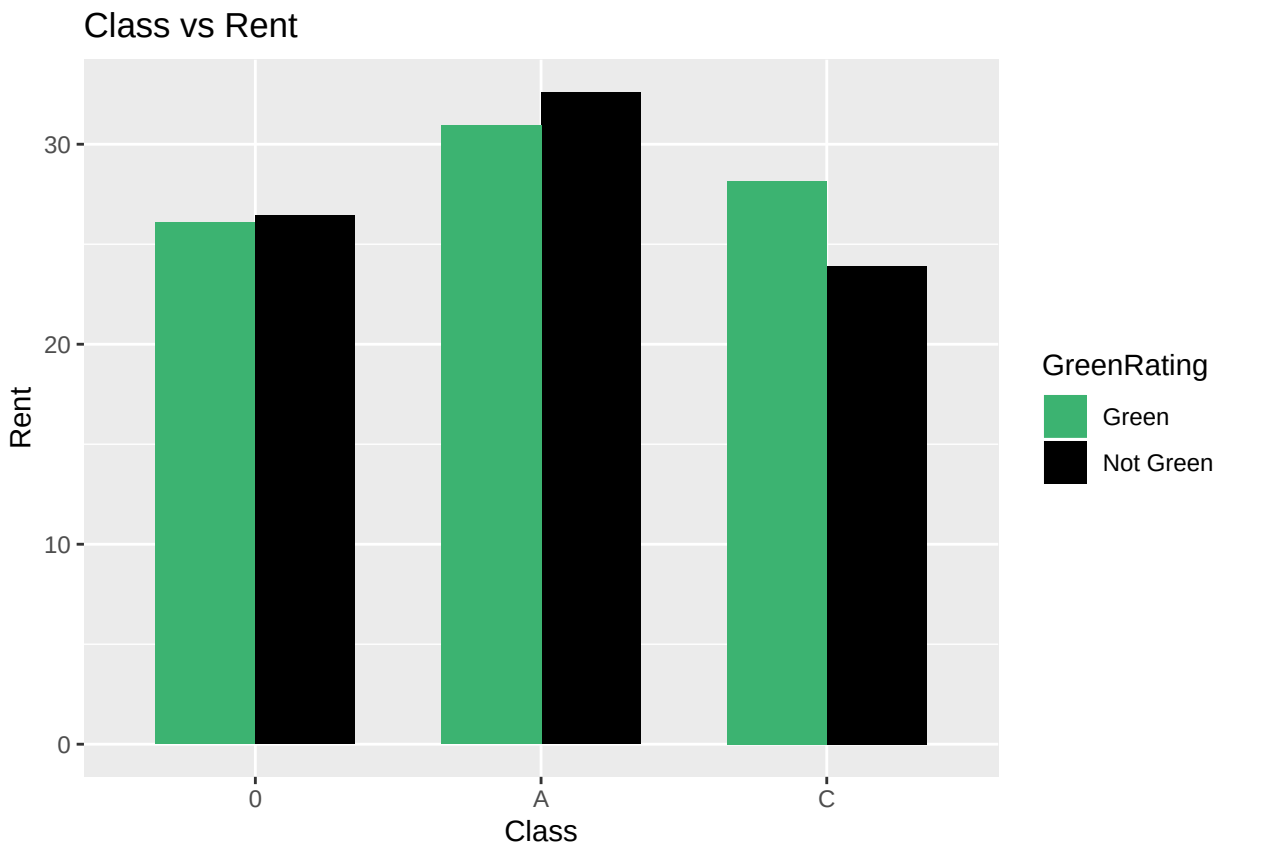


According to the graph, there is no relation between Rent and stories for both green and not green. If there was a difference in the slope of green and not green, the investment in the green building would be profitable. However, since we cannot calculate the slope at all, it is inefficient to invest in the green building.

Next, we tried to see if there was a distinguishable difference in the amount of rent paid per square foot after breaking the data down into classes. When the data analyst was comparing the rent between green buildings and non-green buildings, he may not have made a direct comparison. Other factors, such as class, should have been considered. If tenants were really willing to pay more to live in a green building, then green buildings should have higher rent in every class (A, B, and C).

```
#Rent by Class Bar Chart
#Changing green.rating to string variable
GB_New$green_rating <- rep(0,nrow(GB_New))
Green = GB_New$Green_Rating[which(GB_New$green_rating==1)]
NotGreen = GB_New$Green_Rating[which(GB_New$green_rating==0)]

#Creating ClassC Variable
GB_New$ClassC <- rep(0,nrow(GB_New))
GB_New$ClassC <- GB_New$class_a + GB_New$class_b

#Creating Class Variable
GB_New$Class <- rep(0,nrow(GB_New))
GB_New$Class[which(GB_New$class_a==1)] <- "A"
GB_New$Class[which(GB_New$class_b==1)] <- "B"
GB_New$Class[which(GB_New$ClassC==0)] <- "C"

#Bar Chart
fill <- c("mediumseagreen", "black")
```

```
ClassChart1<- ggplot(data=GB_New, aes(x=Class, y=Rent, fill=GreenRating)) + ggtitle("Class vs Rent") +
  geom_bar(stat="summary", fun.y = "mean",position = position_dodge(),
           width =0.7) +  scale_fill_manual(values=fill)
ClassChart1
```



Only in class C is there an instance of green buildings having higher rent. This shows that when you account for class, green buildings overall do not produce more revenue from rent than non-green buildings.

In conclusion, we did not fully agree with the analysis of the data guru. They did not include several variables that could account for differences in cost and leasing rate.

Milk Prices

In microeconomics, the power law is used to model how consumer demand (number of sales) changes as price changes. The power law states the following: y = alpha$x$ˆ*beta This model can be applied to the milk sales data to determine the best price to maximize profit. The following equation can be used to describe the relationship between net profit (N), price per unit (P), cost per unit (c), and number of sales (Q).* $N = (P\text{-}c)$Q In order to find an equation to predict the number of sales to expect at a given price, the power law is followed. The following equation can be used to describe the relationship between Q, P, price elasticity of demand (E), and a constant (K). Q = KPˆE Then, simplifying the power law by taking the log of both sides yields the following equation: log(y)= log(alpha) + beta-1*log(x) log(y) = beta-0 + beta-1*log(x)

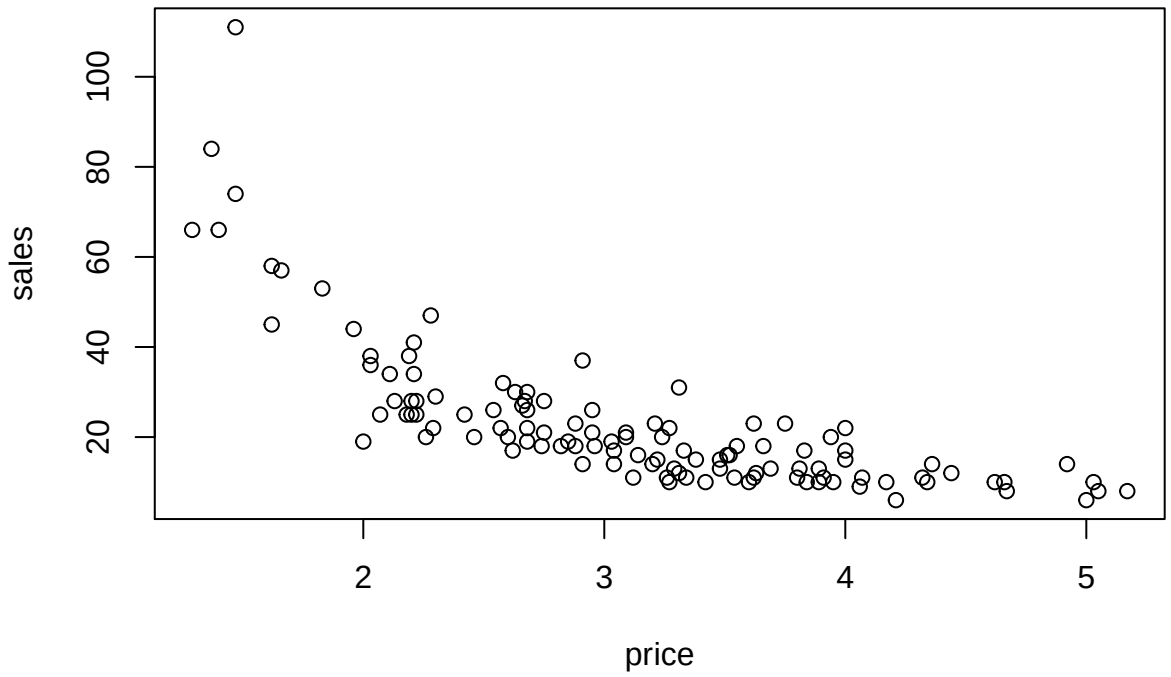
Milk Prices

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

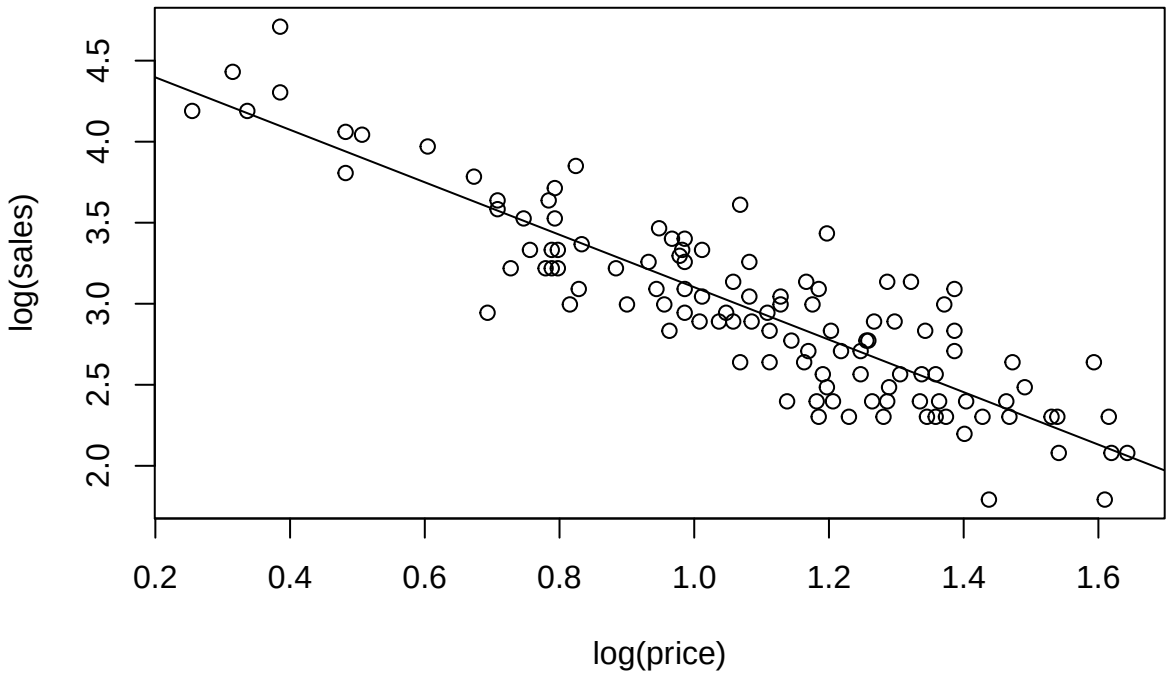
```r
#create linear model following power rule with log(x) and log(y)
milk_loglm = lm(log(sales) ~ log(price), data = milk)
coef(milk_loglm)
```

```
## (Intercept)  log(price)
##    4.720604   -1.618578
```

```r
#plot of sales vs price
plot(sales ~ price, data = milk)
```




```r
#plot of log(sales) vs log(price) with line from linear model
plot(log(sales) ~ log(price), data = milk) + abline(milk_loglm)
```




```
## integer(0)
```

```
#calculating alpha and beta-1 from linear model; cost = $1 per unit
alpha = exp(coef(milk_loglm)[1])
rate = coef(milk_loglm)[2]
c = 1

alpha
```

```
## (Intercept)
##     112.236
```

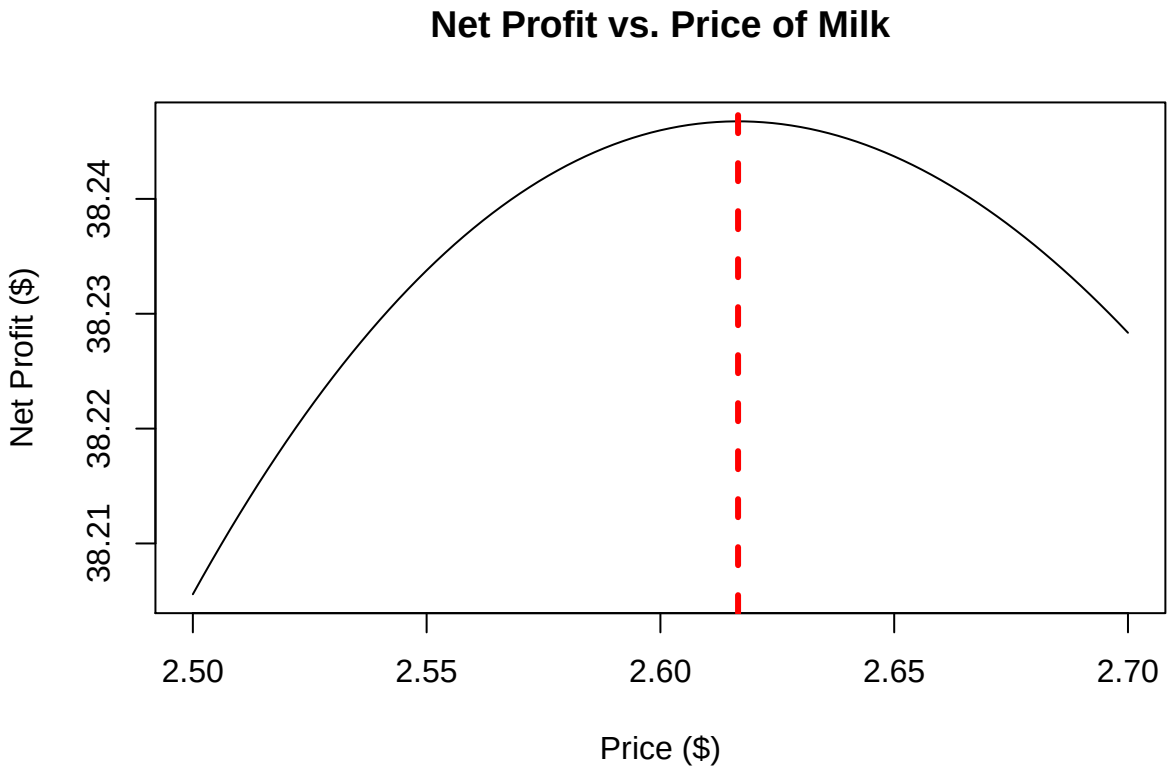
```
rate
```

```
## log(price)
##  -1.618578
```

```
#equation for net profit
eq = function(x){(x-c)*alpha*x^(rate)}

#maximum value for net profit
maximum_price <- optimize(eq, interval=c(2.5, 2.7), maximum=TRUE)
maximum_price
```

```
## $maximum
## [1] 2.616611
##
## $objective
## (Intercept)
##    38.24675
```

```
#create curve to find value that would maximize profit
plot(eq, 2.5, 2.7, xlab = "Price ($)", ylab = "Net Profit ($)", main = "Net Profit vs. Price of Milk")
abline(v = maximum_price, col = 'red', lwd=3, lty=2)
```




```
#maximum price is 2.616611 dollars
eq(2.61)
```

```
## (Intercept)
##    38.24663
```

```
eq(2.62)
```

```
## (Intercept)
##    38.24672
```

```
#$2.62 generates slightly more profit than $2.61

#plot equation onto original scale
plot(sales ~ price, data=milk, main = "Price of Milk vs. Number of Sales")
curve(alpha*(x**rate), add=TRUE, col='red')
```

A linear regression model was then created using the log of both sales and price from the milk data. The model gave us values for the intercept (????0) and rate (????1). ????0 is equal to log(????) so the leading constant, K, in the equation is e4.72. Using ????1 as the elasticity of demand then produces the following equation: Q =e4.72P-1.619 We can then plug this equation into the net profit equation as Q to produce a curve for net profit vs. price. From this curve, we found the profit was at its maximum when the price was equal to $2.616611. Rounding to a full cent, the $2.62 had slightly higher profit than $2.61. The final answer for the best price to maximize profits was determined to be $2.62.