STAT6021 Linear Models for Data Science - Project 01

Tyler Hinnendael (dcc7qe@virginia.edu) Shrikant Mishra (stm5ne@virginia.edu)
Matt Scheffel (mcs9ff@virginia.edu) Rex Minnis (rlm4cr@virginia.edu)

2022-10-16

Section 1

Executive Summary

Load diamonds4.csv dataset

```
Data <- read.csv("../data/diamonds4.csv", header=TRUE)
head(Data)
```

```
##   carat clarity color       cut price
## 1  0.51     SI2     I Very Good   774
## 2  0.93      IF     H     Ideal  6246
## 3  0.50    VVS2     D Very Good  1146
## 4  0.30     VS1     F     Ideal   538
## 5  0.31     SI1     F     Ideal   502
## 6  1.00     VS1     F     Ideal  7046
```

Section 2

Description of the data and the variables

Question 1 - Data Visualizations

1 Use data visualizations to explore how price is related to the other variables (carat, clarity, color, cut), as well as how the other variables may relate to each other. Address the various claims on the diamond education page on Blue Nile.

```
#ggplot(Data, aes (carat))+
#  geom_bar()+
#  scale_x_binned()
```

```
#skimr::skim(Data)
```

```
#kbl(skimr::skim(Data)) %>%
#  kable_styling(bootstrap_options = c("striped", "hover"))
```

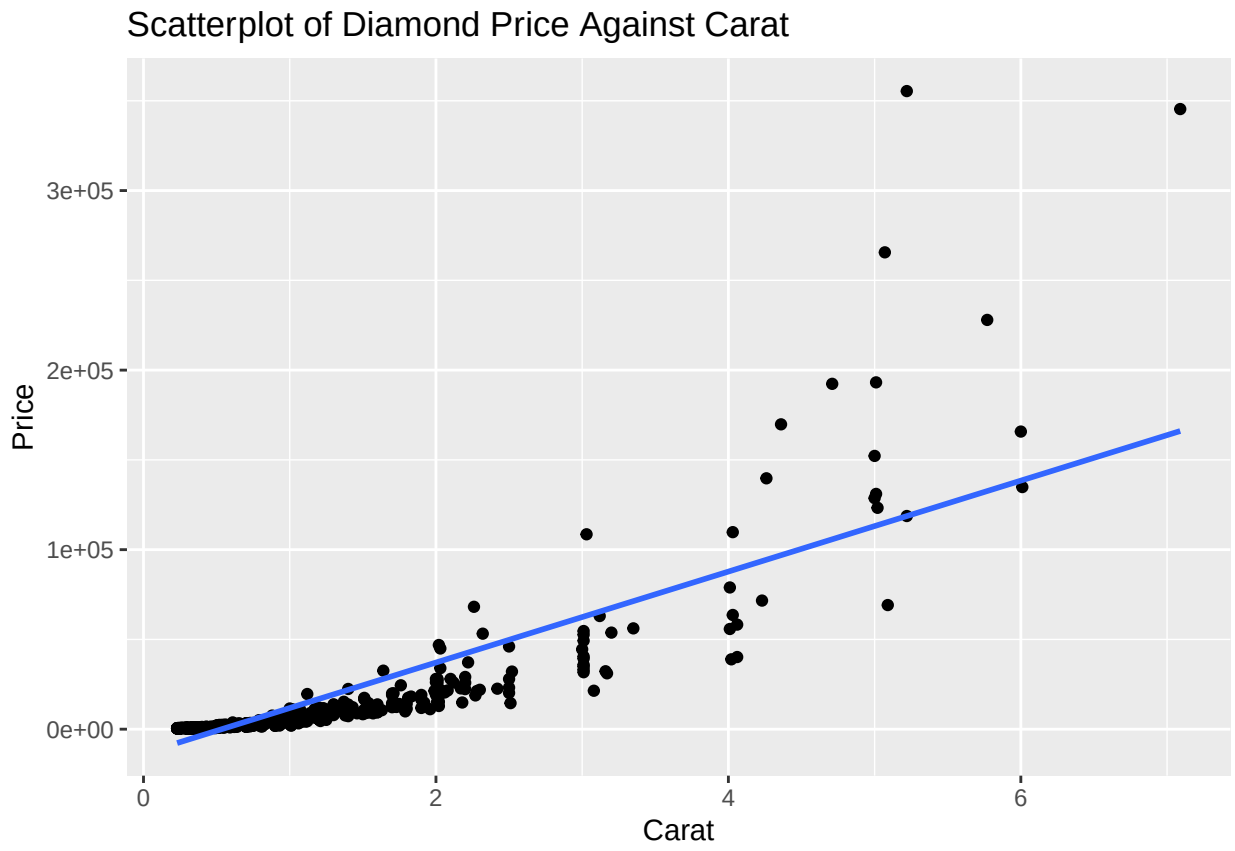
```
#ggplot(Data, aes (clarity))+
#  geom_bar()
```

```
#ggplot(Data, aes (color))+
#  geom_bar()
```

```
#ggplot(Data, aes (cut))+
#  geom_bar()
```

```
ggplot(Data, aes(x = carat, y = price))+
  geom_point()+
  geom_smooth(method = "lm", se=FALSE)+
  labs(y="Price",
       x="Carat",
       title="Scatterplot of Diamond Price Against Carat")
```

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



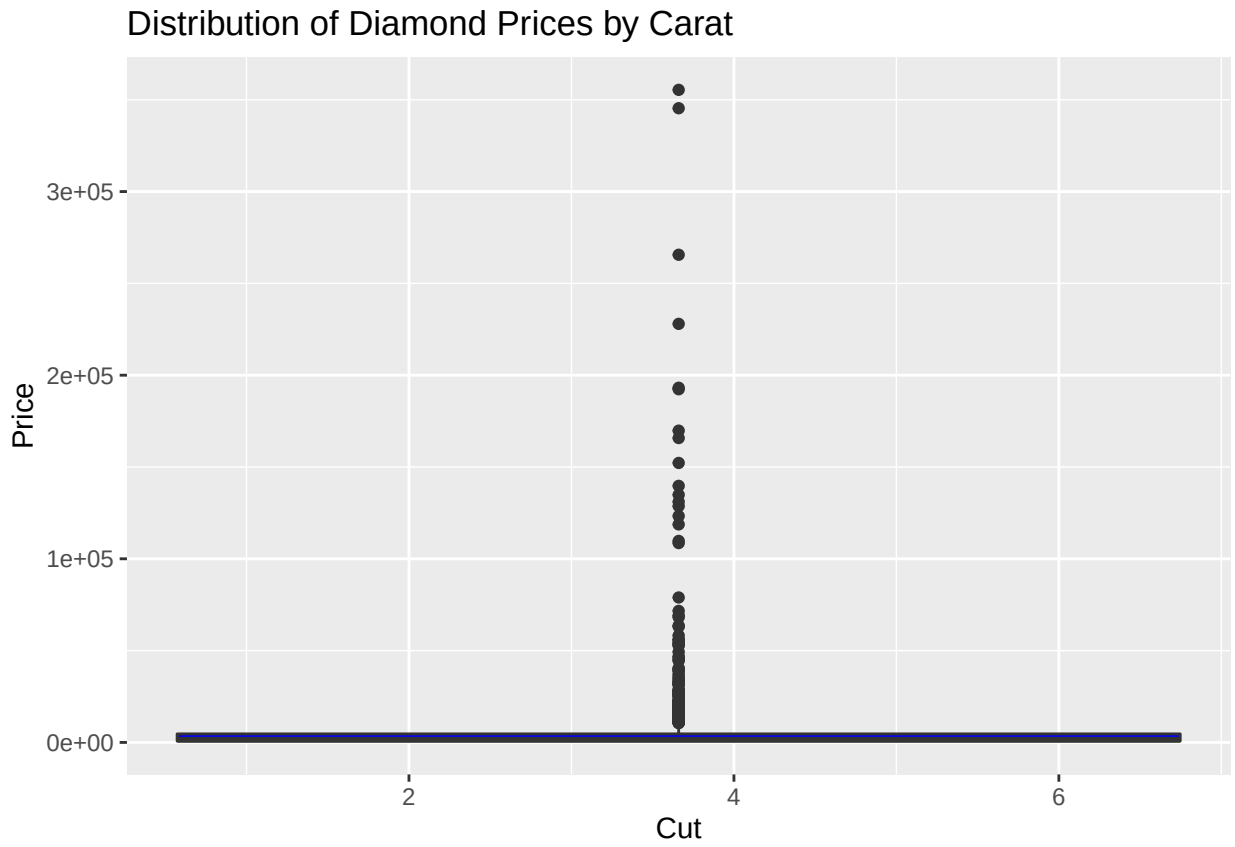

```
#ggplot(Data, aes(x = clarity, y = price))+
#  geom_point()+
#  geom_smooth(method = "lm", se=FALSE)+
#  labs(y="Price",
#       x="Carat",
#       title="Scatterplot of Diamond Price Against Cut")
```

```
#ggplot(Data, aes(x = color, y = price))+
#  geom_point()+
#  geom_smooth(method = "lm", se=FALSE)+
#  labs(y="Price",
#       x="Carat",
#       title="Scatterplot of Diamond Price Against Cut")
```

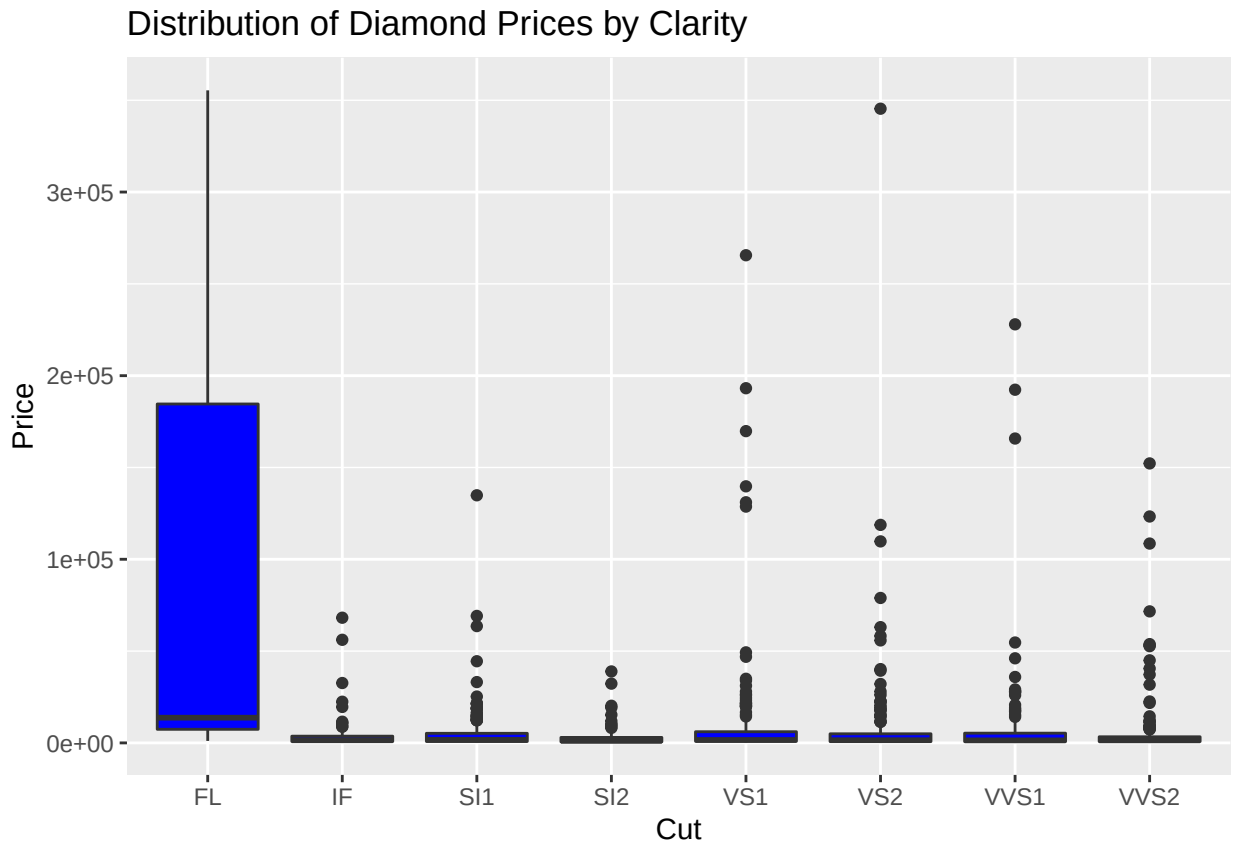
```
#ggplot(Data, aes(x = cut, y = price))+
#  geom_point()+
#  geom_smooth(method = "lm", se=FALSE)+
#  labs(y="Price",
#       x="Carat",
#       title="Scatterplot of Diamond Price Against Cut")
```

```
ggplot(Data, aes(x=carat, y=price))+
     geom_boxplot(fill="Blue")+
     labs(x="Cut", y="Price",
      title="Distribution of Diamond Prices by Carat")
```

```
## Warning: Continuous x aesthetic -- did you forget aes(group=...)?
```

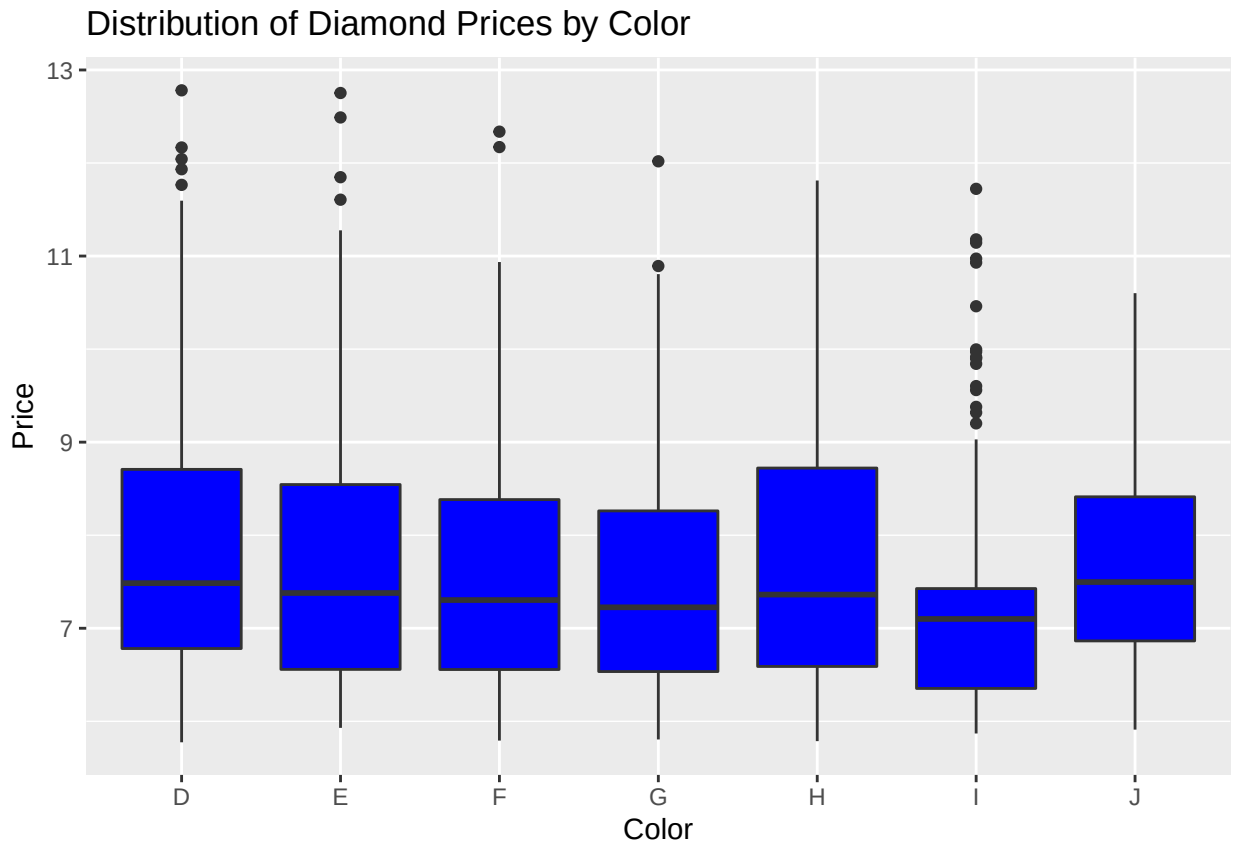



```
ggplot(Data, aes(x=clarity, y=price))+
     geom_boxplot(fill="Blue")+
     labs(x="Cut", y="Price",
      title="Distribution of Diamond Prices by Clarity")
```




```
Data$price_logtf <- log(Data$price)
```

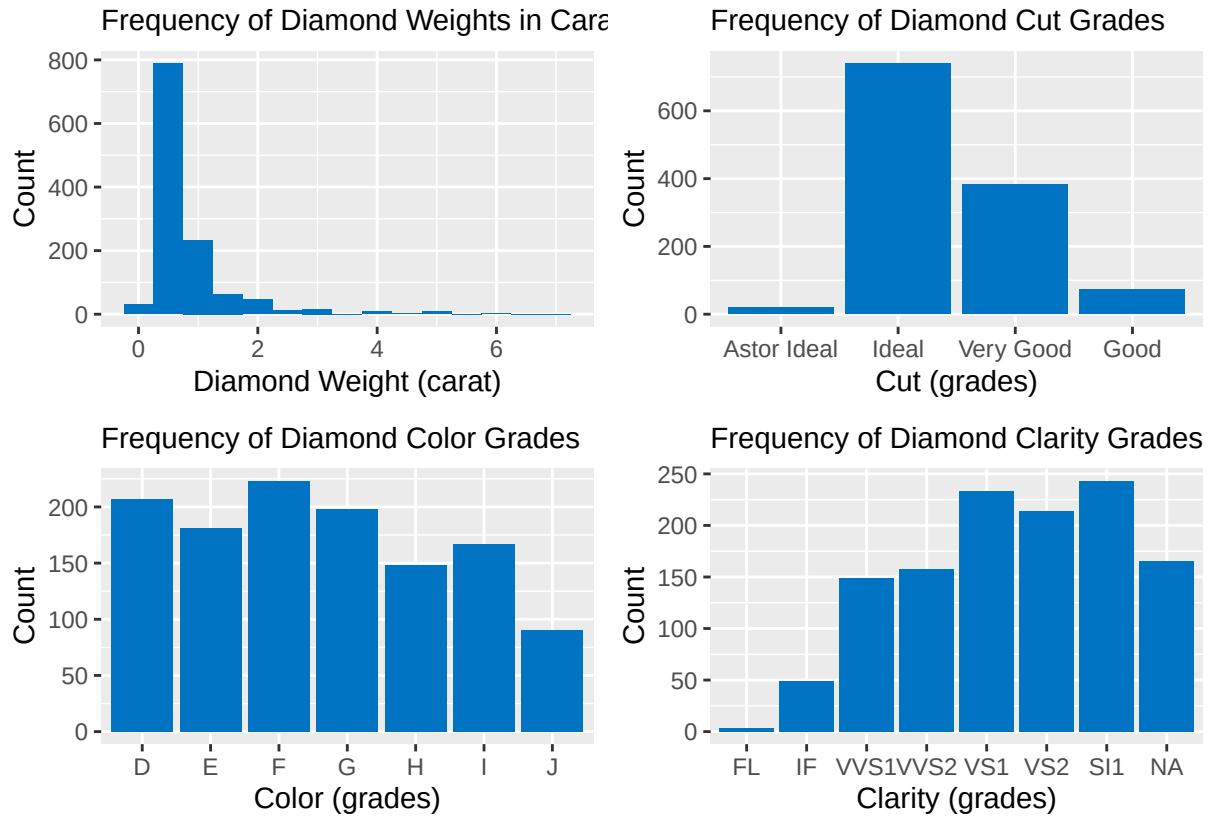
```
ggplot(Data, aes(x=color, y=price_logtf))+
     geom_boxplot(fill="Blue")+
     labs(x="Color", y="Price",
      title="Distribution of Diamond Prices by Color")
```



Question 2 - Univariate Analysis: Frequency Plots

```
#create the log(price) column to transform the y variable
vec <- log(Data$price)
Data["log_price"] <- vec

# reorders these two columns in the diamonds dataframe so it plots in descending order.
Data$clarity <- factor(Data$clarity, levels = c("FL", "IF", "VVS1", "VVS2","VS1","VS2", "SI1"))
Data$cut <- factor(Data$cut, levels = c("Astor Ideal", "Ideal", "Very Good", "Good"))

p1 <- ggplot(Data, aes(carat))+
  geom_histogram(binwidth = 0.5, fill = "#0073C2FF")+
  ggtitle("Frequency of Diamond Weights in Carats")+
  xlab("Diamond Weight (carat)")+
  ylab("Count")+
  theme(plot.title = element_text(size=11))

p2 <- ggplot(Data, aes(cut))+
  geom_bar(fill = "#0073C2FF")+
  ggtitle("Frequency of Diamond Cut Grades")+
  xlab("Cut (grades)")+
  ylab("Count")+
  theme(plot.title = element_text(size=11))
```

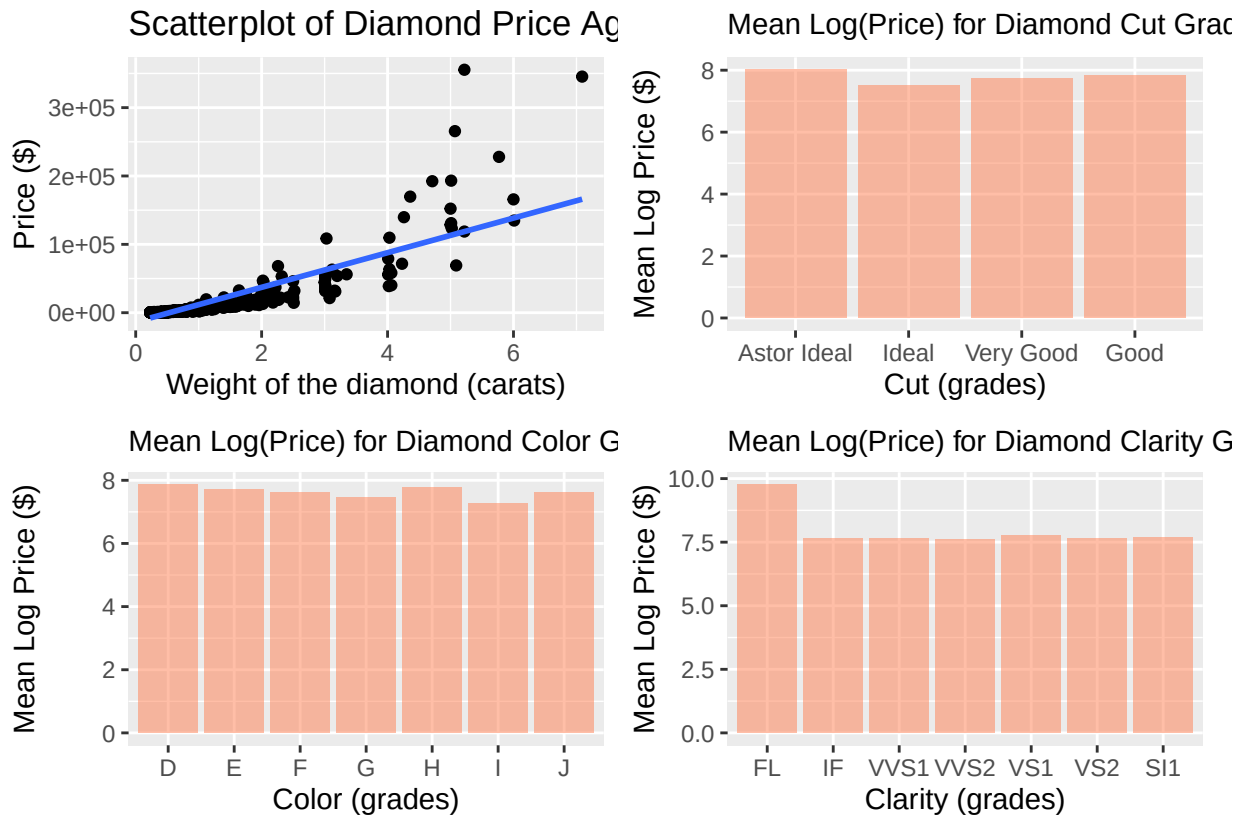
```
p3 <- ggplot(Data, aes(color))+
  geom_bar(fill = "#0073C2FF")+
  ggtitle("Frequency of Diamond Color Grades")+
  xlab("Color (grades)")+
  ylab("Count")+
  theme(plot.title = element_text(size=11))

p4 <- ggplot(Data, aes(clarity))+
  geom_bar(fill = "#0073C2FF")+
  ggtitle("Frequency of Diamond Clarity Grades")+
  xlab("Clarity (grades)")+
  ylab("Count")+
  theme(plot.title = element_text(size=11))

p1 + p2 + p3 + p4 + plot_layout(ncol = 2)
```



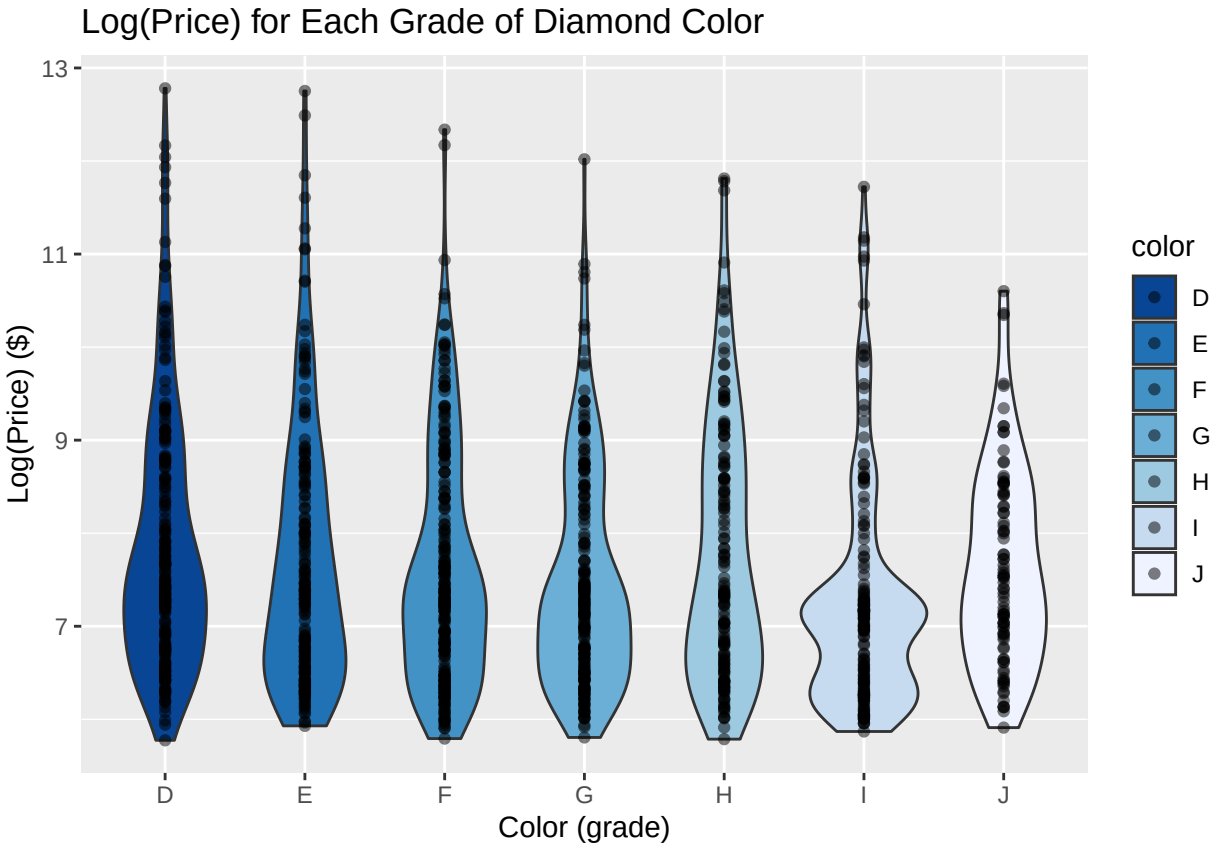

```
cut_mean <- aggregate(Data$log_price, list(Data$cut), FUN=mean)
cols <- c("Cut", "Mean_log_Price")
colnames(cut_mean) <- cols
cut_mean$Cut <- factor(cut_mean$Cut, levels=c("Astor Ideal", "Ideal", "Very Good", "Good"))

color_mean <- aggregate(Data$log_price, list(Data$color), FUN=mean)
cols <- c("Color", "Mean_log_Price")
colnames(color_mean) <- cols
```

```r
clarity_mean <- aggregate(Data$log_price, list(Data$clarity), FUN=mean)
cols <- c("Clarity", "Mean_log_Price")
colnames(clarity_mean) <- cols
clarity_mean$Clarity<- factor(clarity_mean$Clarity, levels=c("FL", "IF", "VVS1", "VVS2", "VS1", "VS2",


g1 <- ggplot(Data, aes(x = carat, y = price))+
        geom_point()+
        geom_smooth(method = "lm", se=FALSE)+
        labs(y="Price ($)", x="Weight of the diamond (carats)",
           title="Scatterplot of Diamond Price Against Carat")

g2 <- ggplot(cut_mean, aes(x=Cut, y=Mean_log_Price))+
  geom_bar(stat = "identity", fill = "coral", alpha = 0.5)+
  ggtitle("Mean Log(Price) for Diamond Cut Grades")+
  xlab("Cut (grades)")+
  ylab("Mean Log Price ($)")+
  theme(plot.title = element_text(size=11))

g3 <- ggplot(color_mean, aes(x=Color, y=Mean_log_Price))+
  geom_bar(stat = "identity", fill = "coral", alpha = 0.5)+
  ggtitle("Mean Log(Price) for Diamond Color Grades")+
  xlab("Color (grades)")+
  ylab("Mean Log Price ($)")+
  theme(plot.title = element_text(size=11))

g4 <- ggplot(clarity_mean, aes(x=Clarity, y=Mean_log_Price))+
  geom_bar(stat = "identity", fill = "coral", alpha = 0.5)+
  ggtitle("Mean Log(Price) for Diamond Clarity Grades")+
  xlab("Clarity (grades)")+
  ylab("Mean Log Price ($)")+
  theme(plot.title = element_text(size=11))

g1 + g2 + g3 + g4 + plot_layout(ncol = 2)
```

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



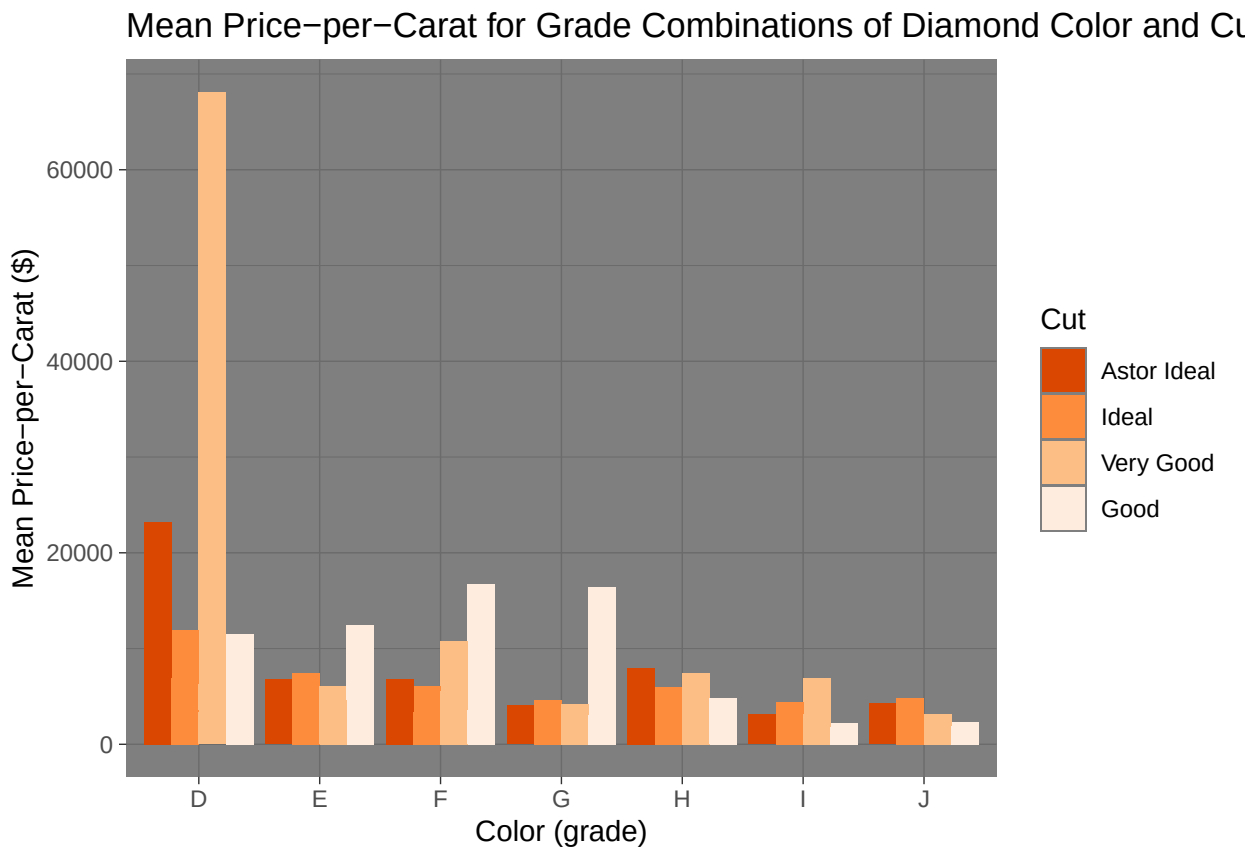

```
#ggplot(Data, aes(x=cut, y=log_price, fill=cut)) +
#  geom_violin() +
#  geom_jitter(width=0, alpha=0.5) +
#  scale_fill_brewer(palette="Blues", direction=-1)

ggplot(Data, aes(x=color, y=log_price, fill=color)) +
  geom_violin() +
  geom_jitter(width=0, alpha=0.5) +
  scale_fill_brewer(palette="Blues", direction=-1)+
  labs(y="Log(Price) ($)", x="Color (grade)",
           title="Log(Price) for Each Grade of Diamond Color")
```



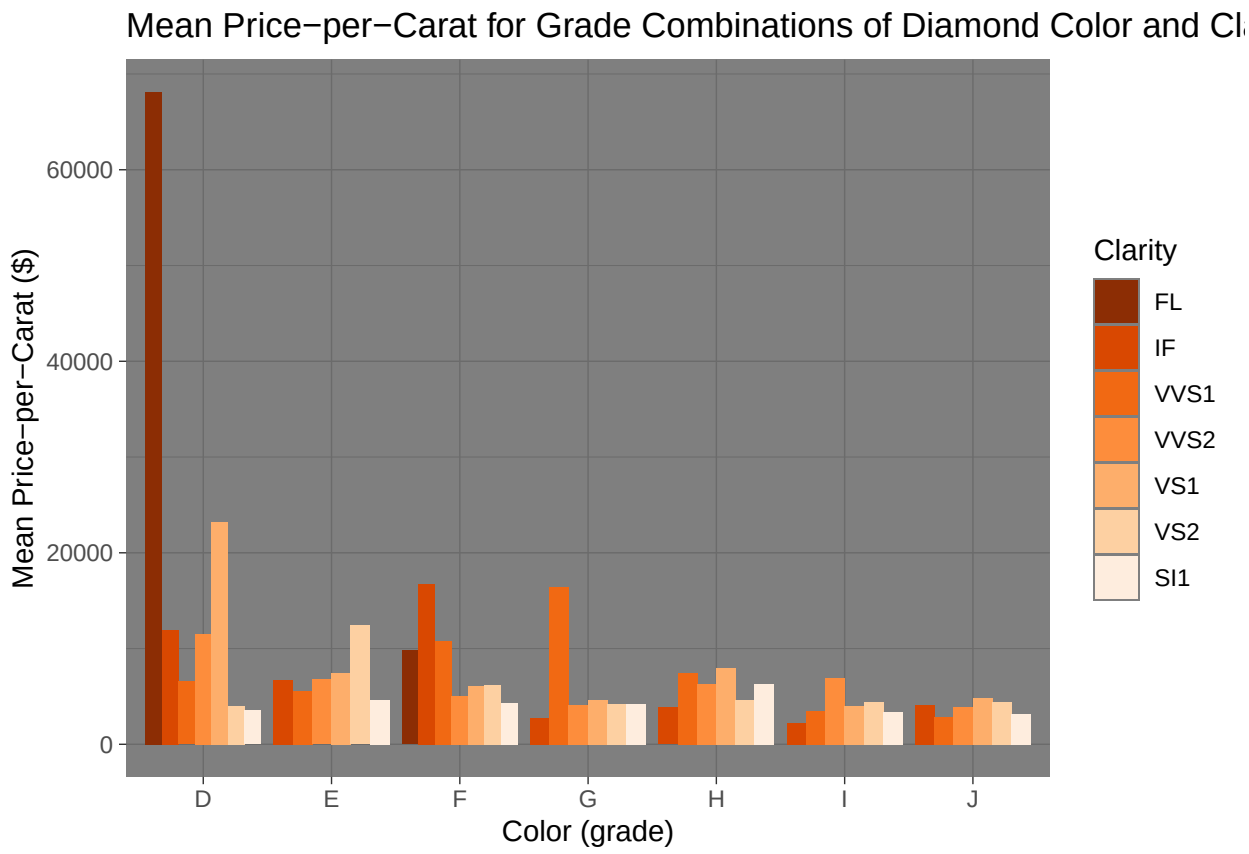

```
#ggplot(Data, aes(x=clarity, y=log_price, fill=clarity)) +
#  geom_violin() +
#  geom_jitter(width=0, alpha=0.5) +
#  scale_fill_brewer(palette="Blues", direction=-1)
```

```
# created 'price per carat" to normalize based on a wide range of carat weights
vec <- (Data$price / Data$carat)
Data["price_per_carat"] <- vec

# A new dataframe 'varmeans' is created to represent the 'mean price_per_carat' for all cut/color/clari
# This dataframe is used for the multivariate plots.
var_means <- aggregate(Data$price_per_carat, list(Data$cut, Data$color, Data$clarity), FUN=mean)
cols <- c("Cut", "Color", "Clarity", "Mean_Price_Per_Carat")
colnames(var_means) <- cols
var_means$Clarity<- factor(var_means$Clarity, levels=c("FL", "IF", "VVS1", "VVS2", "VS1", "VS2", "SI1",
var_means$Cut<- factor(var_means$Cut, levels=c("Astor Ideal", "Ideal", "Very Good", "Good"))
```

```
# Outputs comparisons of Color and Cut combinations
p<-ggplot(var_means, aes(x=Color, y=Mean_Price_Per_Carat, fill=Cut)) +
  geom_bar(position="dodge", stat = "identity") +
  scale_fill_brewer(palette = 7, direction = -1) +
  scale_colour_brewer(palette = 1)
p+theme_dark() + ggtitle("Mean Price-per-Carat for Grade Combinations of Diamond Color and Cut") +
  xlab("Color (grade)") + ylab("Mean Price-per-Carat ($)")
```




```
# Outputs comparisons of Color and Clarity combinations
p<-ggplot(var_means, aes(x=Color, y=Mean_Price_Per_Carat, fill=Clarity)) +
  geom_bar(position="dodge", stat = "identity") +
  scale_fill_brewer(palette = 7, direction = -1) +
  scale_colour_brewer(palette = 1)
p+theme_dark() + ggtitle("Mean Price-per-Carat for Grade Combinations of Diamond Color and Clarity") +
  xlab("Color (grade)") + ylab("Mean Price-per-Carat ($)")
```



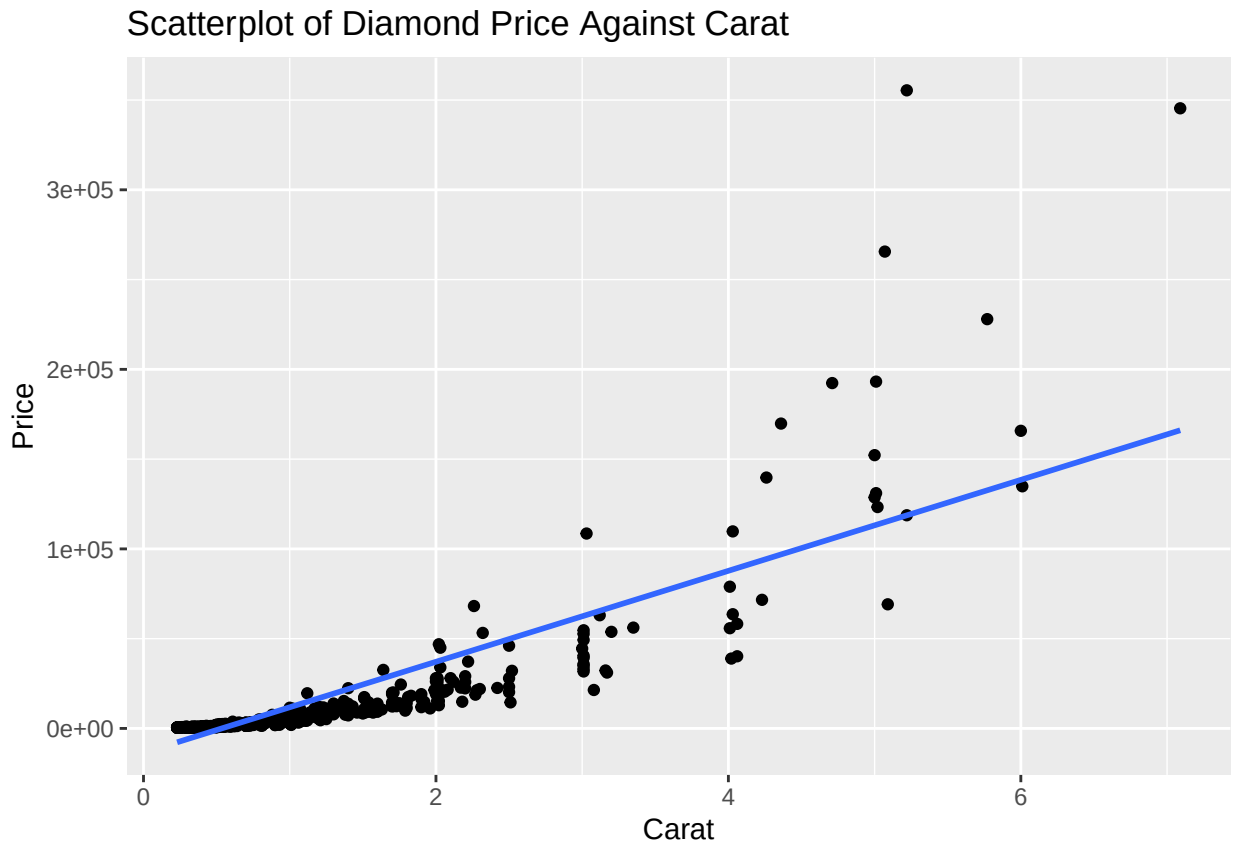
Section 3

A description of how you fitted the regression of price against carat, and the conclusions reached.

Question 2. Fit an appropriate simple linear regression for price against carat.

```
ggplot(Data, aes(x = carat, y = price))+
  geom_point()+
  geom_smooth(method = "lm", se=FALSE)+
    labs(y="Price",
       x="Carat",
       title="Scatterplot of Diamond Price Against Carat")
```

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

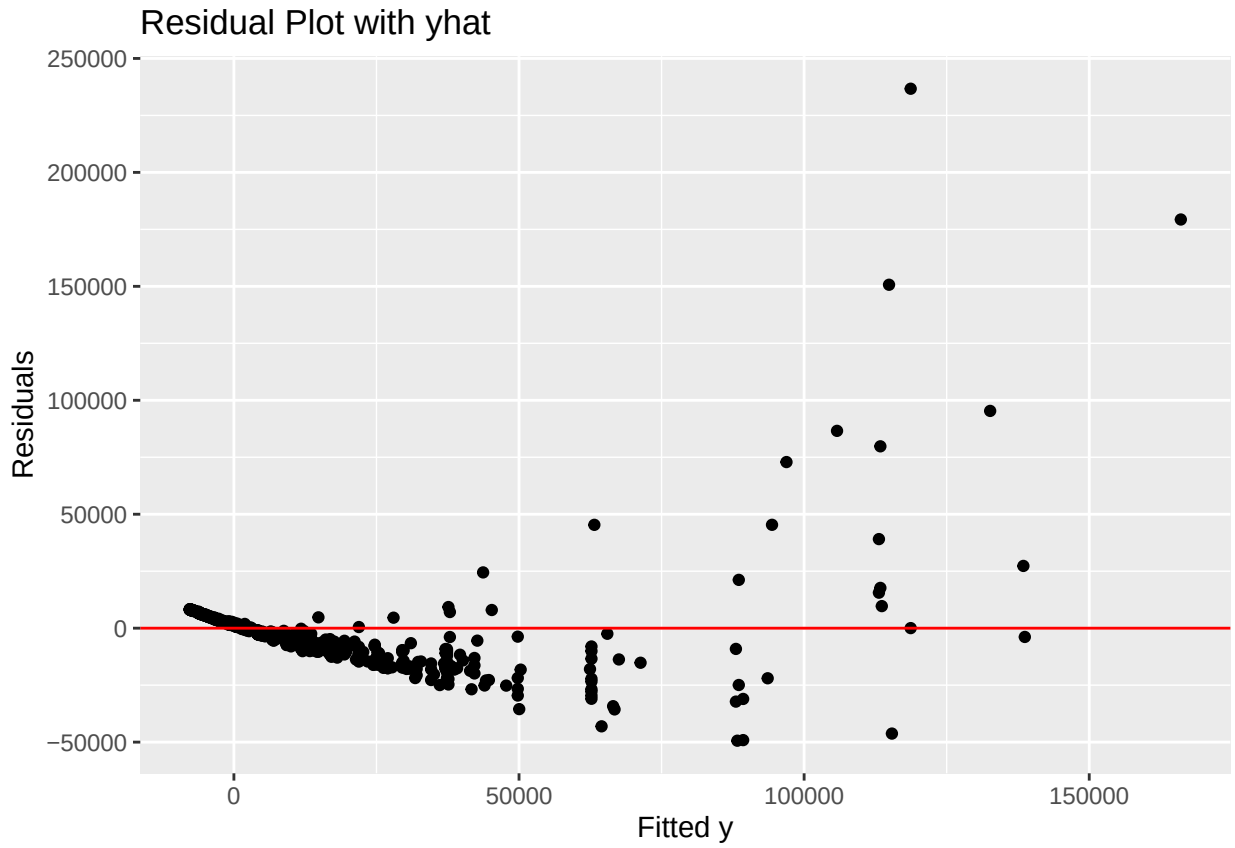



```
results <- lm(price~carat, data=Data)
summary(results)
```

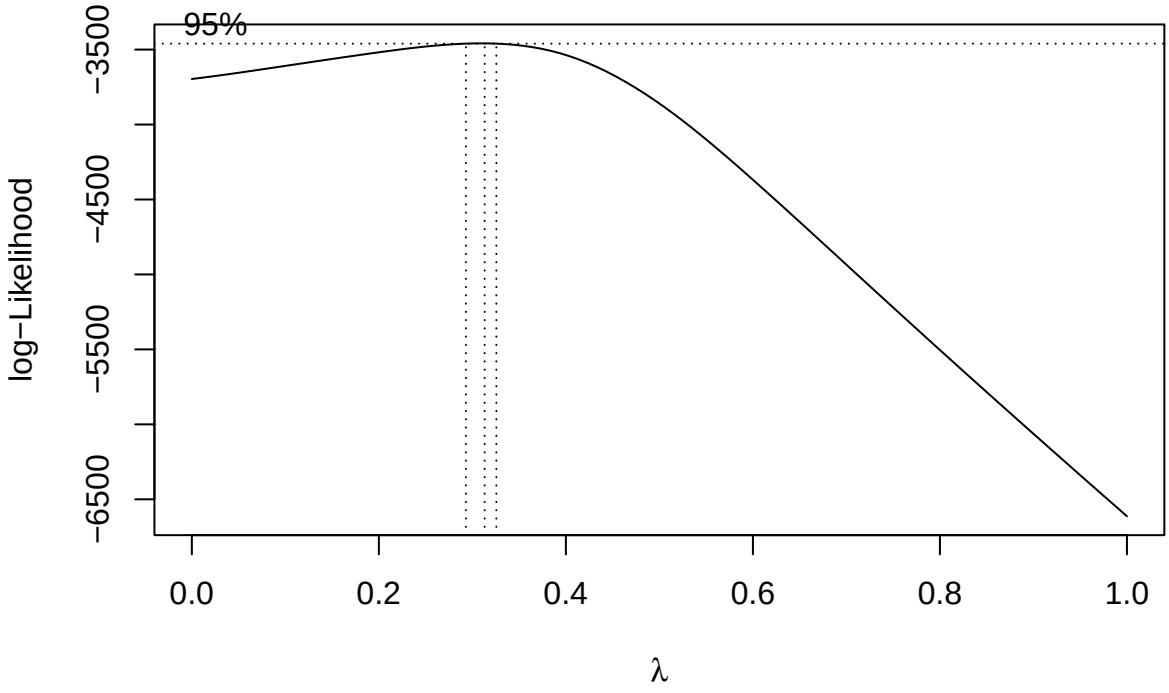
```
##
## Call:
## lm(formula = price ~ carat, data = Data)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -49375  -5048   1867   4965 236711
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -13550.9      559.7  -24.21   <2e-16 ***
## carat        25333.9      494.4   51.24   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 13560 on 1212 degrees of freedom
## Multiple R-squared:  0.6842, Adjusted R-squared:  0.6839
## F-statistic:  2625 on 1 and 1212 DF,  p-value: < 2.2e-16
```

```
yhat <- results$fitted.values
res <- results$residuals

ggplot(Data, aes(x=yhat, y=res))+
  geom_point()+
```

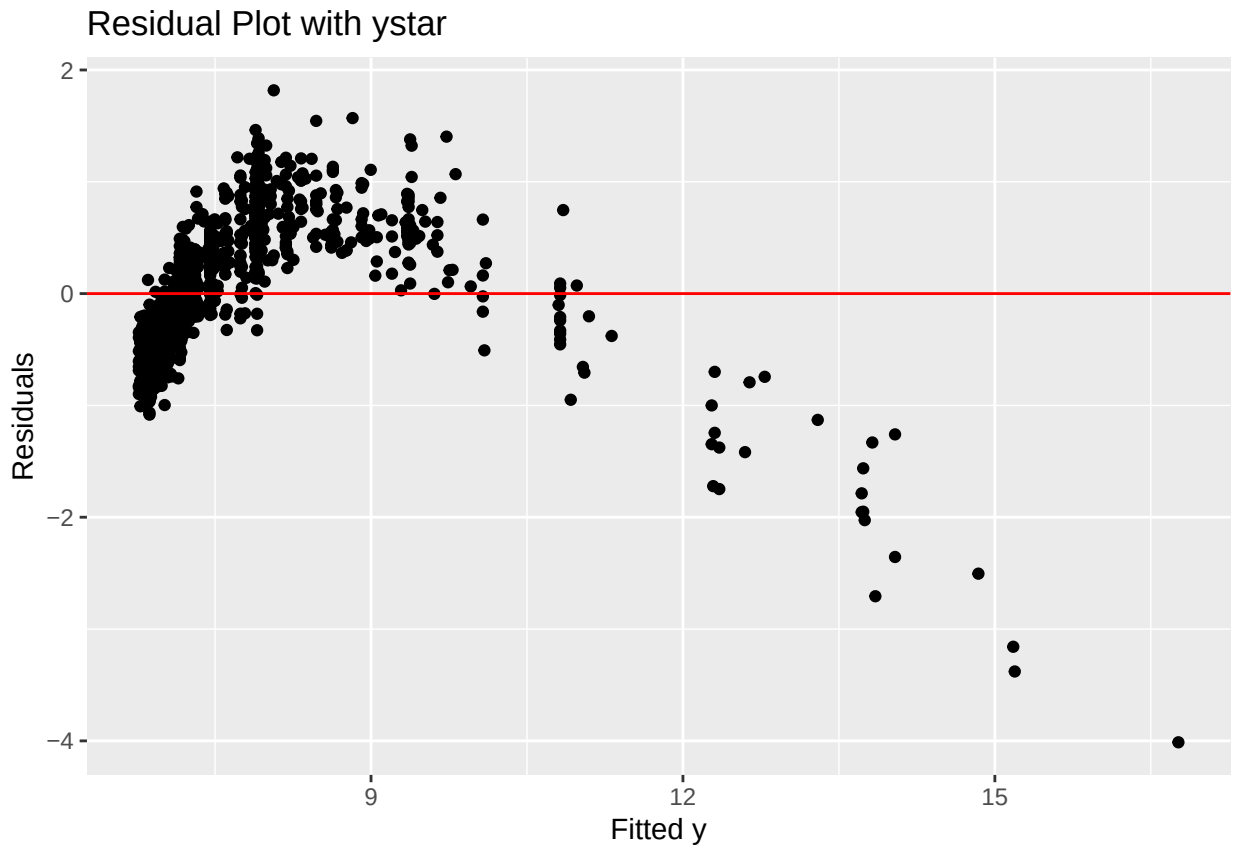
```
  geom_hline(yintercept = 0, color="red")+
  labs(x="Fitted y", y="Residuals", title="Residual Plot with yhat")
```




```
boxcox(results, lambda = seq(0, 1, 0.1))
```

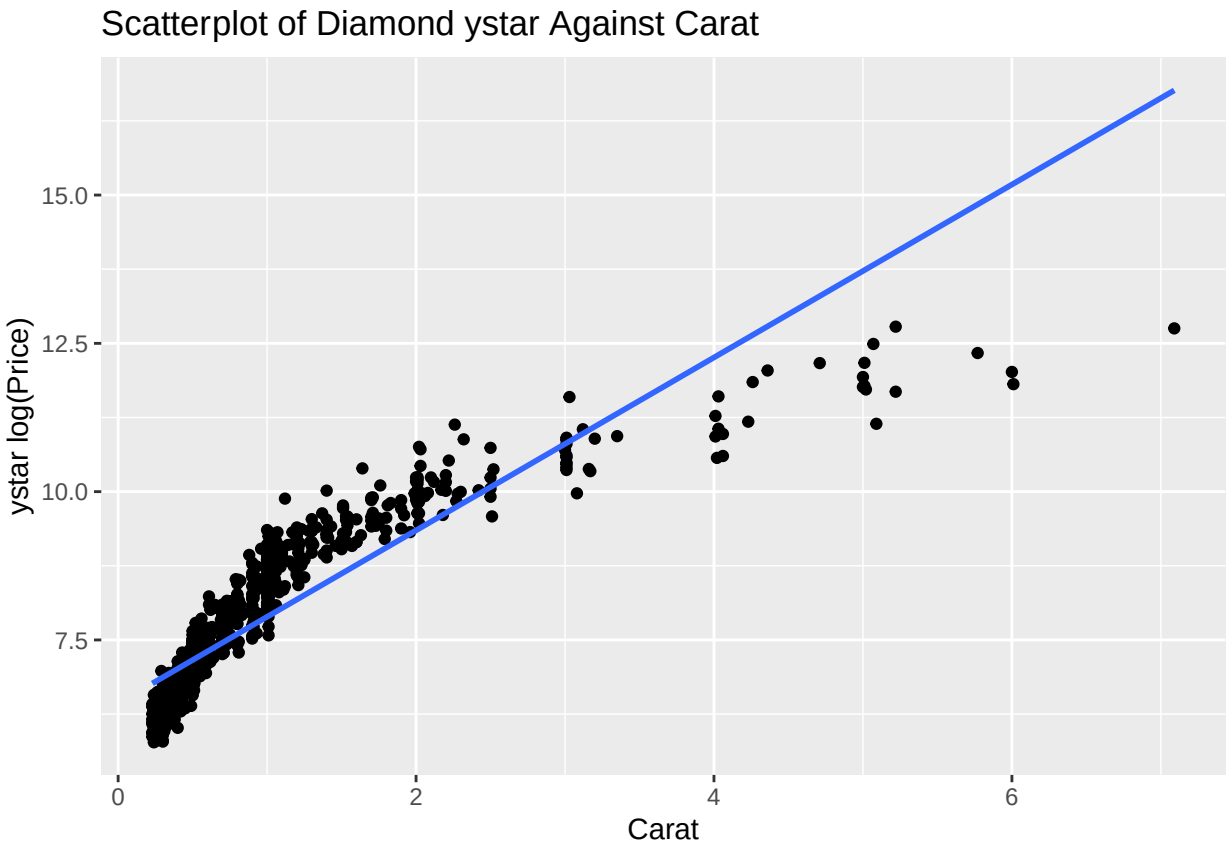



```
ystar <- log(Data$price)
results2 <- lm(ystar~carat, data=Data)
yhat2 <- results2$fitted.values
res2 <- results2$residuals

ggplot(Data, aes(x=yhat2, y=res2))+
  geom_point()+
  geom_hline(yintercept = 0, color="red")+
  labs(x="Fitted y", y="Residuals", title="Residual Plot with ystar")
```

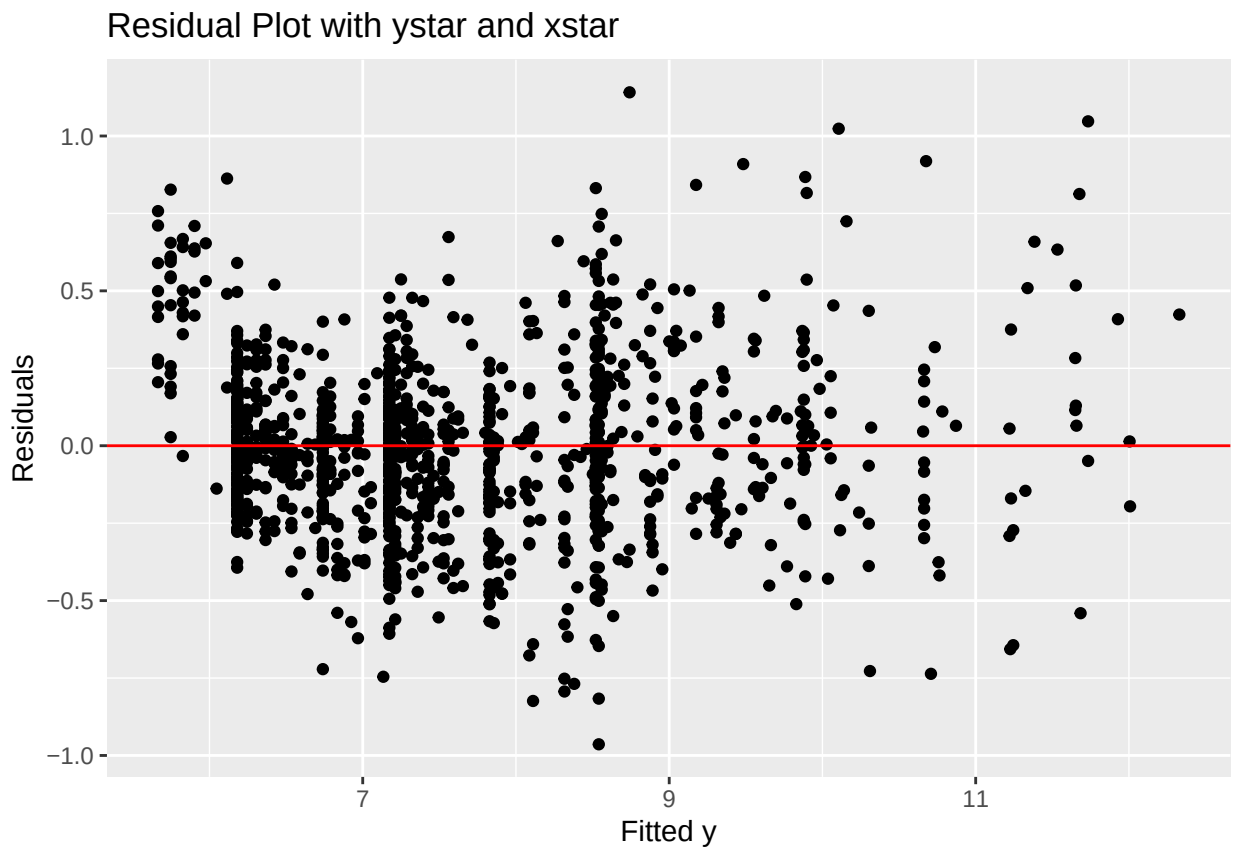



```
ggplot(Data, aes(x=carat, y=ystar))+
  geom_point()+
  geom_smooth(method="lm", se=FALSE)+
  labs(y="ystar log(Price)",
       x="Carat",
       title="Scatterplot of Diamond ystar Against Carat")
```

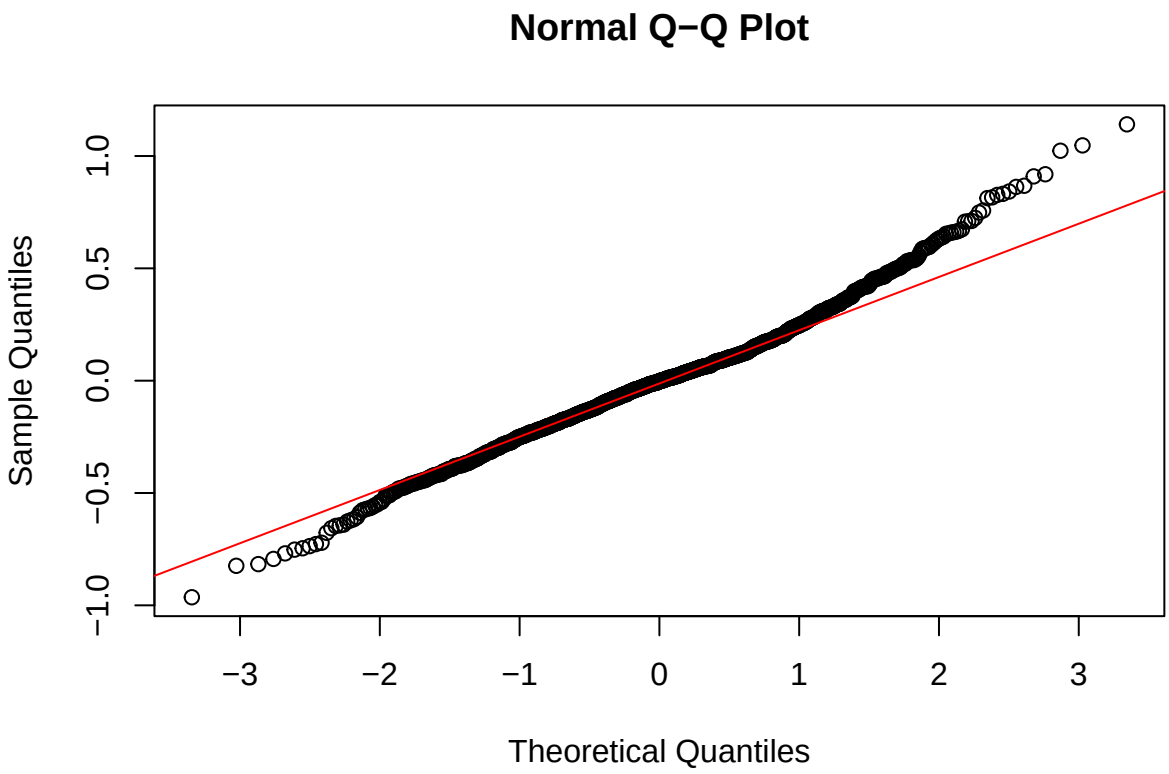
```
## `geom_smooth()` using formula 'y ~ x'
```



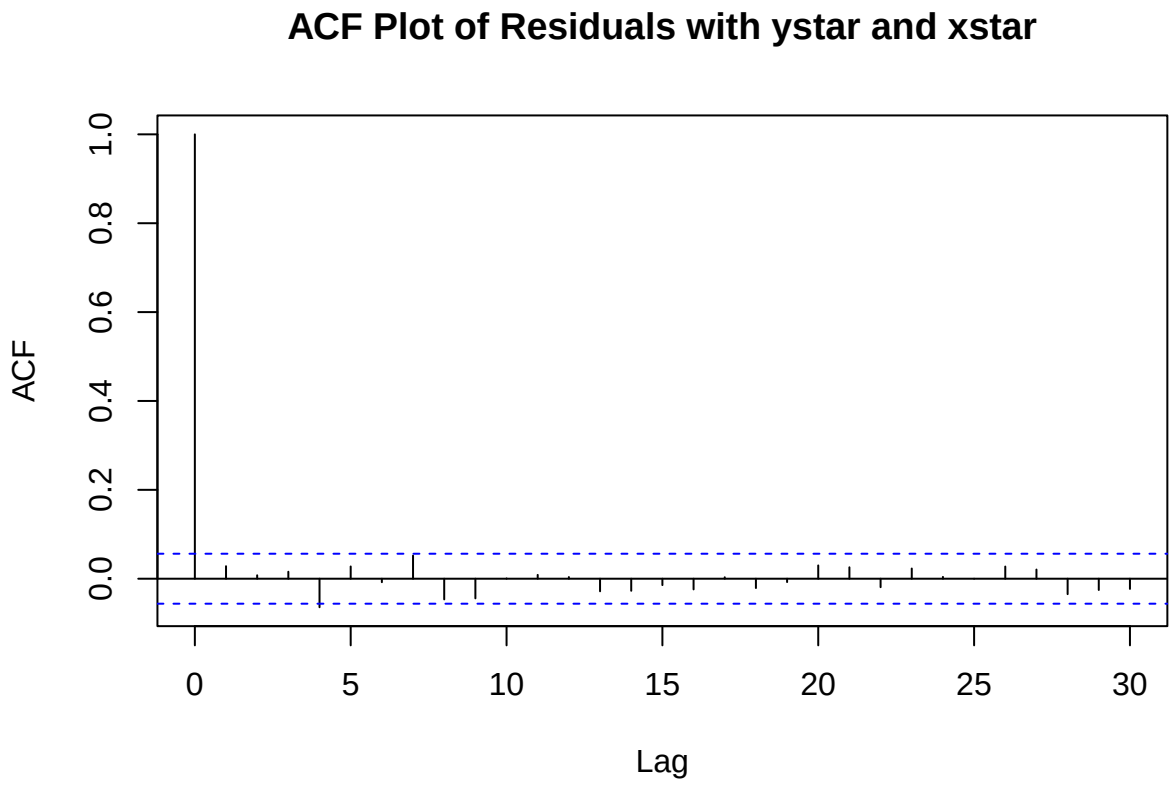

```
xstar <- log(Data$carat)
results3 <- lm(ystar~xstar, data=Data)
yhat3 <- results3$fitted.values
res3 <- results3$residuals

ggplot(Data, aes(x=yhat3, y=res3))+
  geom_point()+
  geom_hline(yintercept = 0, color="red")+
  labs(x="Fitted y", y="Residuals", title="Residual Plot with ystar and xstar")
```




```
qqnorm(res3)
qqline(res3, col = "red")
```

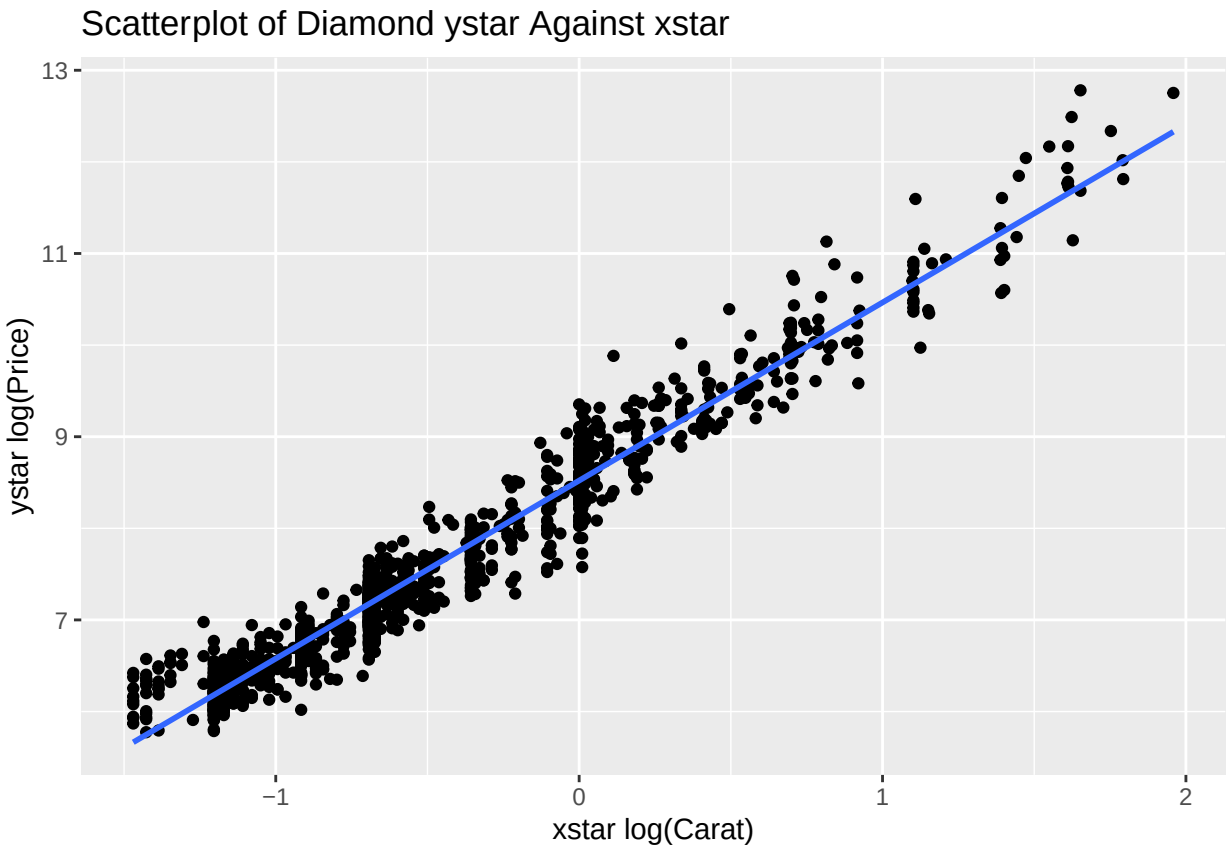



```
a <- acf(res3, main="ACF Plot of Residuals with ystar and xstar")
```




```
ggplot(Data, aes(x=xstar, y=ystar))+
  geom_point()+
  geom_smooth(method="lm", se=FALSE)+
  labs(y="ystar log(Price)",
       x="xstar log(Carat)",
       title="Scatterplot of Diamond ystar Against xstar")
```

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


Scatterplot of Diamond ystar Against xstar
13
11
9
7
ystar log(Price)
−1
0
1
2
xstar log(Carat)