# STAT 3355 - HW 4

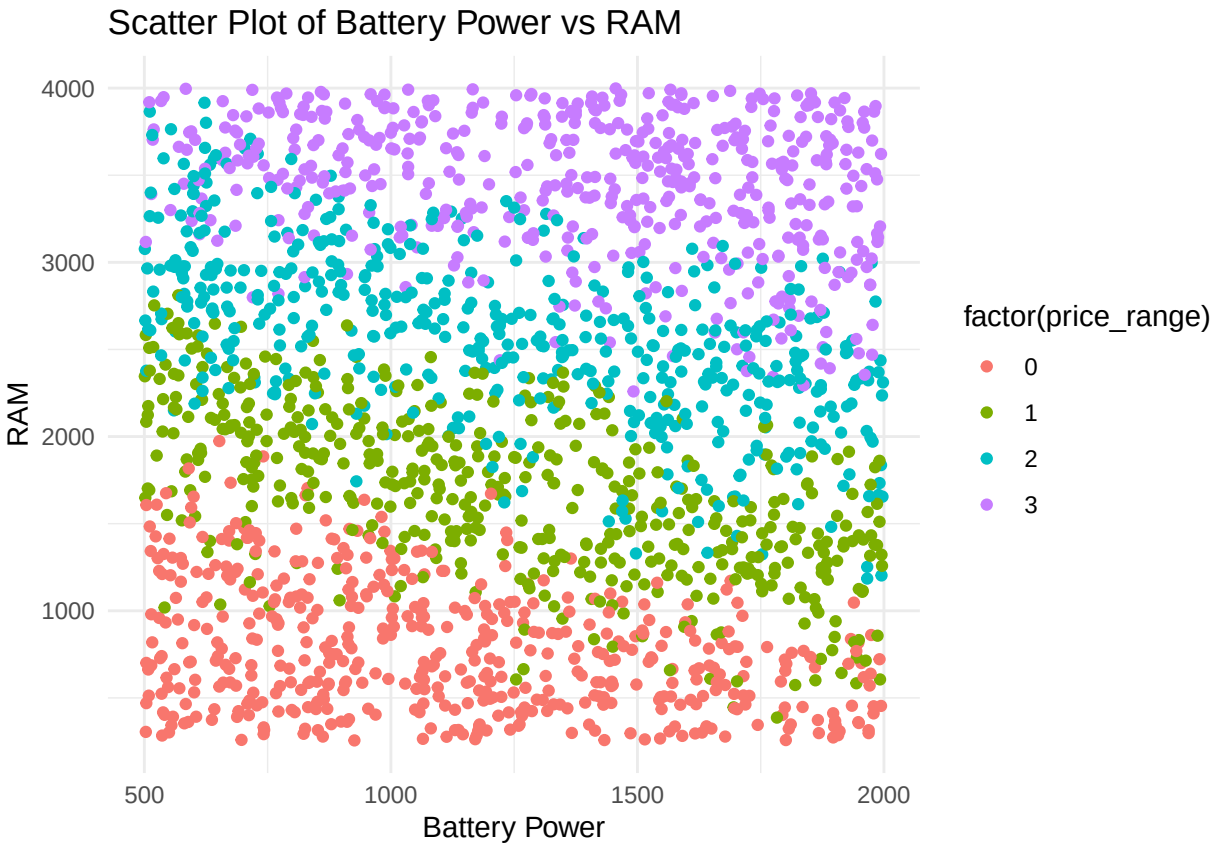
2024-03-19

## Problem 1

```
library(ggplot2)
data <- read.csv("/Users/springkim/downloads/train.csv")
```

## Problem 1 - (a)

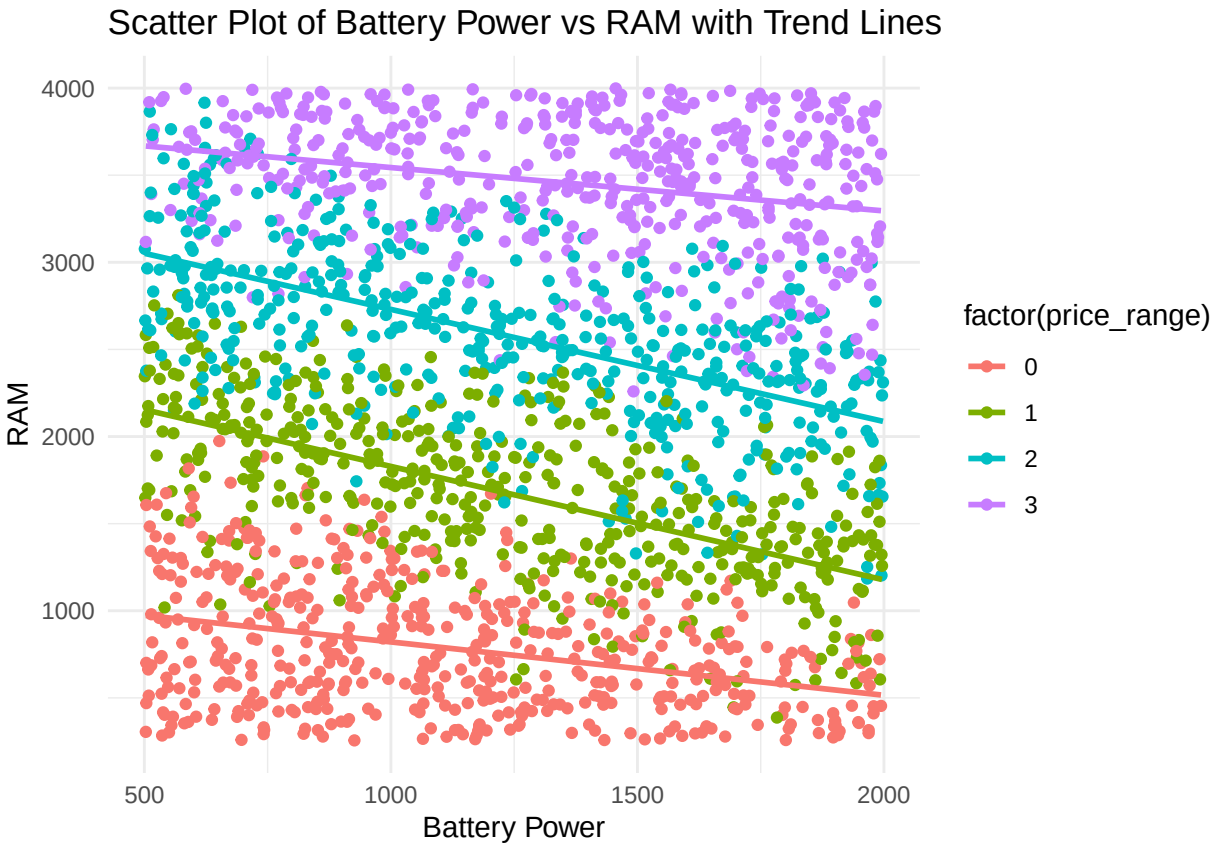
```
ggplot(data, aes(x = battery_power, y = ram, color = factor(price_range))) +
  geom_point() +
  labs(x = "Battery Power", y = "RAM") +
  ggtitle("Scatter Plot of Battery Power vs RAM") +
  theme_minimal()
```




## Problem 1 - (b)

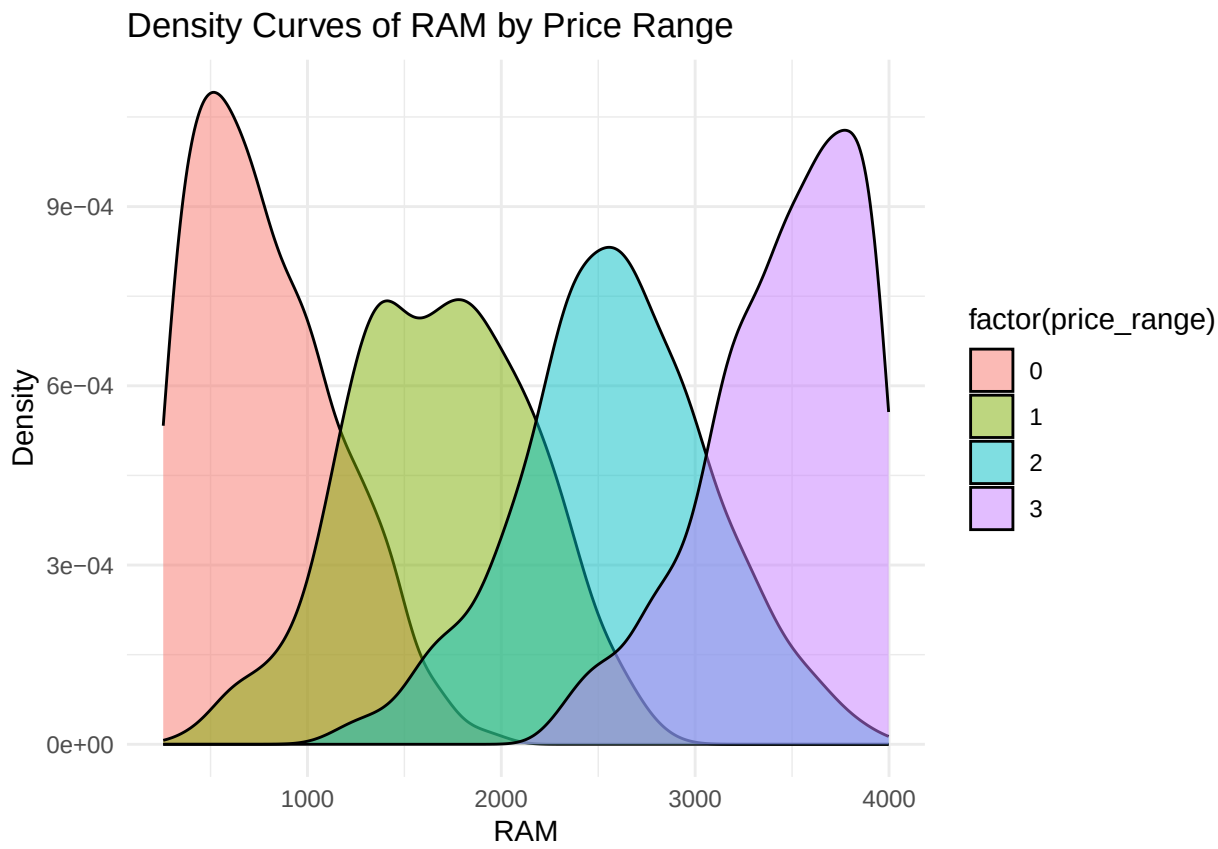
```
ggplot(data, aes(x = battery_power, y = ram, color = factor(price_range))) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE) +
  labs(x = "Battery Power", y = "RAM") +
  ggtitle("Scatter Plot of Battery Power vs RAM with Trend Lines") +
  theme_minimal()
```

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



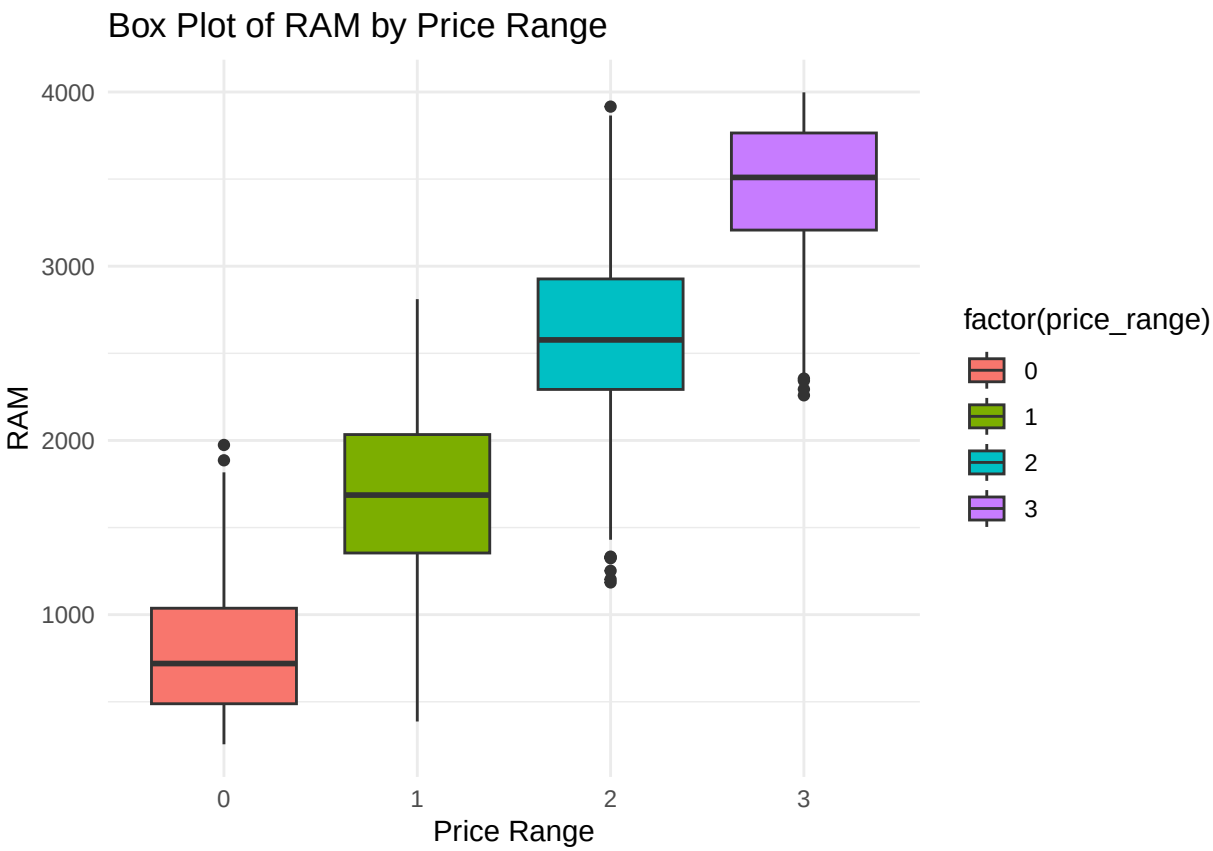

## Problem 1 - (c)

```
ggplot(data, aes(x = ram, fill = factor(price_range))) +
  geom_density(alpha = 0.5) +
  labs(x = "RAM", y = "Density") +
  ggtitle("Density Curves of RAM by Price Range") +
  theme_minimal()
```



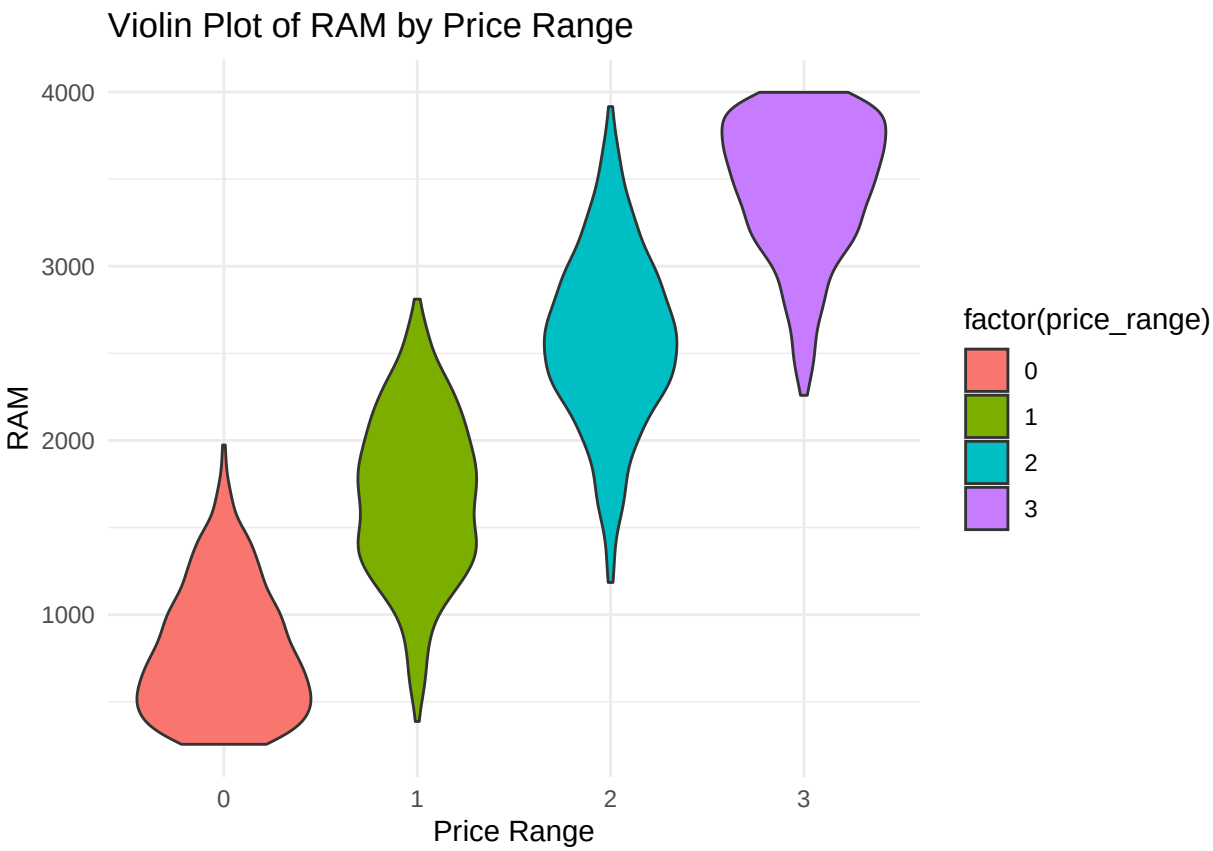

## Problem 1 - (d)

```
ggplot(data, aes(x = factor(price_range), y = ram, fill = factor(price_range))) +
  geom_boxplot() +
  labs(x = "Price Range", y = "RAM") +
  ggtitle("Box Plot of RAM by Price Range") +
  theme_minimal()
```



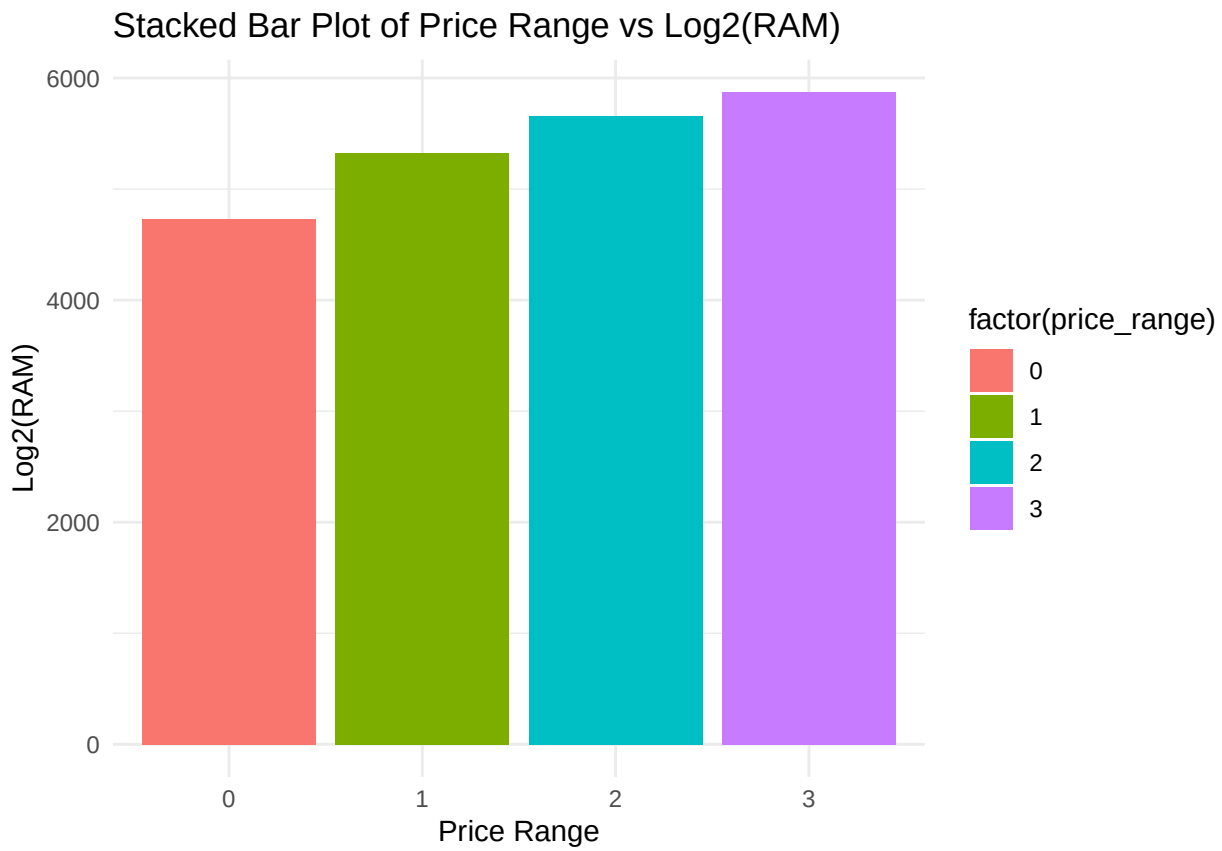

## Problem 1 - (e)

```
ggplot(data, aes(x = factor(price_range), y = ram, fill = factor(price_range))) +
  geom_violin() +
  labs(x = "Price Range", y = "RAM") +
  ggtitle("Violin Plot of RAM by Price Range") +
  theme_minimal()
```




## Problem 1 - (f)

```
ggplot(data, aes(x = factor(price_range), fill = factor(price_range))) +
  geom_bar(aes(y = log2(ram)), stat = "identity", position = "stack") +
  labs(x = "Price Range", y = "Log2(RAM)") +
  ggtitle("Stacked Bar Plot of Price Range vs Log2(RAM)") +
  theme_minimal()
```




## Problem 2

```
library(UsingR)
```

```
## Loading required package: MASS

## Loading required package: HistData

## Loading required package: Hmisc

##
## Attaching package: 'Hmisc'

## The following objects are masked from 'package:base':
##
##     format.pval, units
```

```
library(ggplot2)
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 lubridate 1.9.3     v tibble    3.2.1
## v purrr     1.0.2     v tidyr     1.3.1
```

```
## -- Conflicts ---------------------------------------- tidyverse_conflicts() --
## x dplyr::filter()    masks stats::filter()
## x dplyr::lag()       masks stats::lag()
## x dplyr::select()    masks MASS::select()
## x dplyr::src()       masks Hmisc::src()
## x dplyr::summarize() masks Hmisc::summarize()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

### Problem 2 - (a)

```
cereal_data <- data("UScereal")
full_names <- c("General Mills", "Kelloggs", "Nabisco", "Post", "Quaker Oats", "Ralston Purina")
UScereal$mfr <- factor(UScereal$mfr, labels = full_names)
levels(UScereal$mfr)
```

```
## [1] "General Mills"  "Kelloggs"       "Nabisco"        "Post"
## [5] "Quaker Oats"    "Ralston Purina"
```

### Problem 2 - (b)

```
UScereal$shelf <- factor(UScereal$shelf)
levels(UScereal$shelf) <- c("Low", "Middle", "Upper")
levels(UScereal$shelf)
```

```
## [1] "Low"    "Middle" "Upper"
```

### Problem 2 - (c)

```
UScereal$Product <- rownames(UScereal)
```

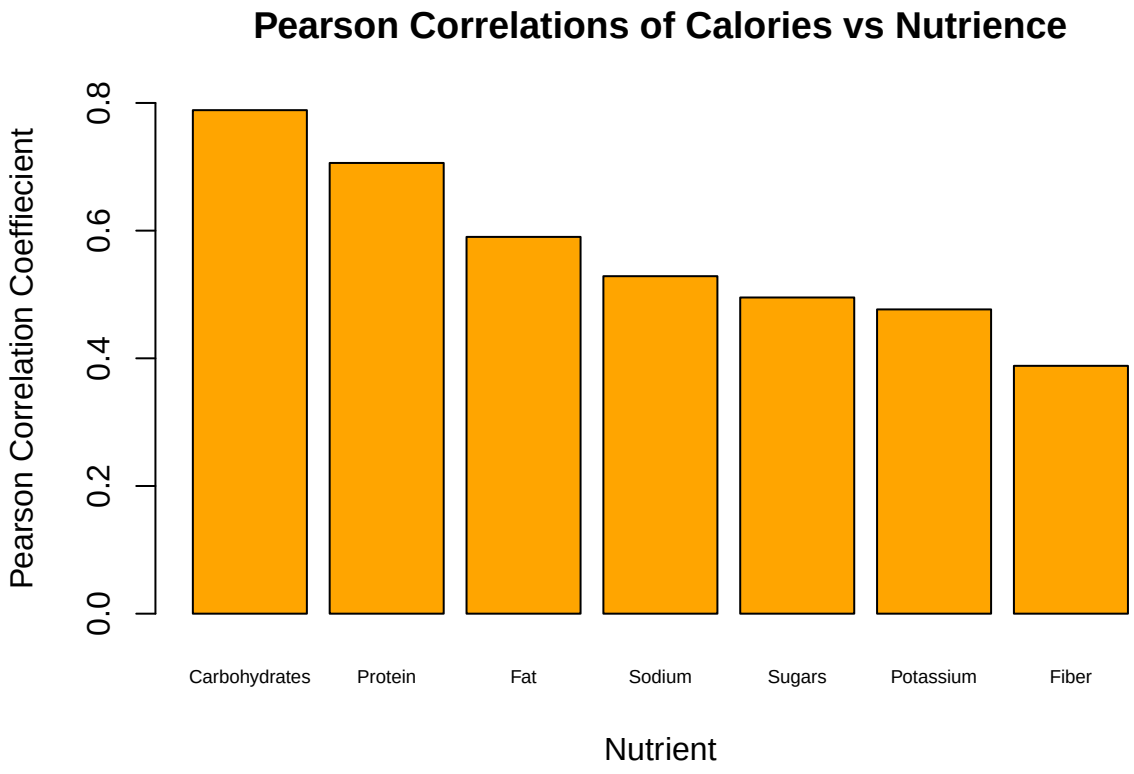
### Problem 2 - (d)

```
proteinCor <- cor(UScereal$calories, UScereal$protein)
fatCor <- cor(UScereal$calories, UScereal$fat)
sodiumCor <- cor(UScereal$calories, UScereal$sodium)
fiberCor <- cor(UScereal$calories, UScereal$fibre)
carboCor <- cor(UScereal$calories, UScereal$carbo)
sugarCor <- cor(UScereal$calories, UScereal$sugars)
potassCor <- cor(UScereal$calories, UScereal$potassium)
cor_df <- data.frame(Nutrience = c("Protein", "Fat", "Sodium", "Fiber", "Carbohydrates", "Sugars", "Pota
                     Calorie_Correlation=c(proteinCor, fatCor, sodiumCor, fiberCor, carboCor, sugarCor,
print(cor_df)
```

```
##       Nutrience Calorie_Correlation
## 1       Protein           0.7060105
## 2           Fat           0.5901757
## 3        Sodium           0.5286552
## 4         Fiber           0.3882179
## 5 Carbohydrates           0.7887227
## 6        Sugars           0.4952942
## 7     Potassium           0.4765955
```

### Problem 2 - (e)

```r
ordered_cor_df <- cor_df[order(-cor_df$Calorie_Correlation),]
barplot(ordered_cor_df$Calorie_Correlation,
        names.arg = ordered_cor_df$Nutrience,
        main = "Pearson Correlations of Calories vs Nutrience",
        xlab = "Nutrient",
        ylab = "Pearson Correlation Coeffiecient",
        col = "orange",
        cex.names = 0.6,
        ylim = range(pretty(c(0, ordered_cor_df$Calorie_Correlation))))
```

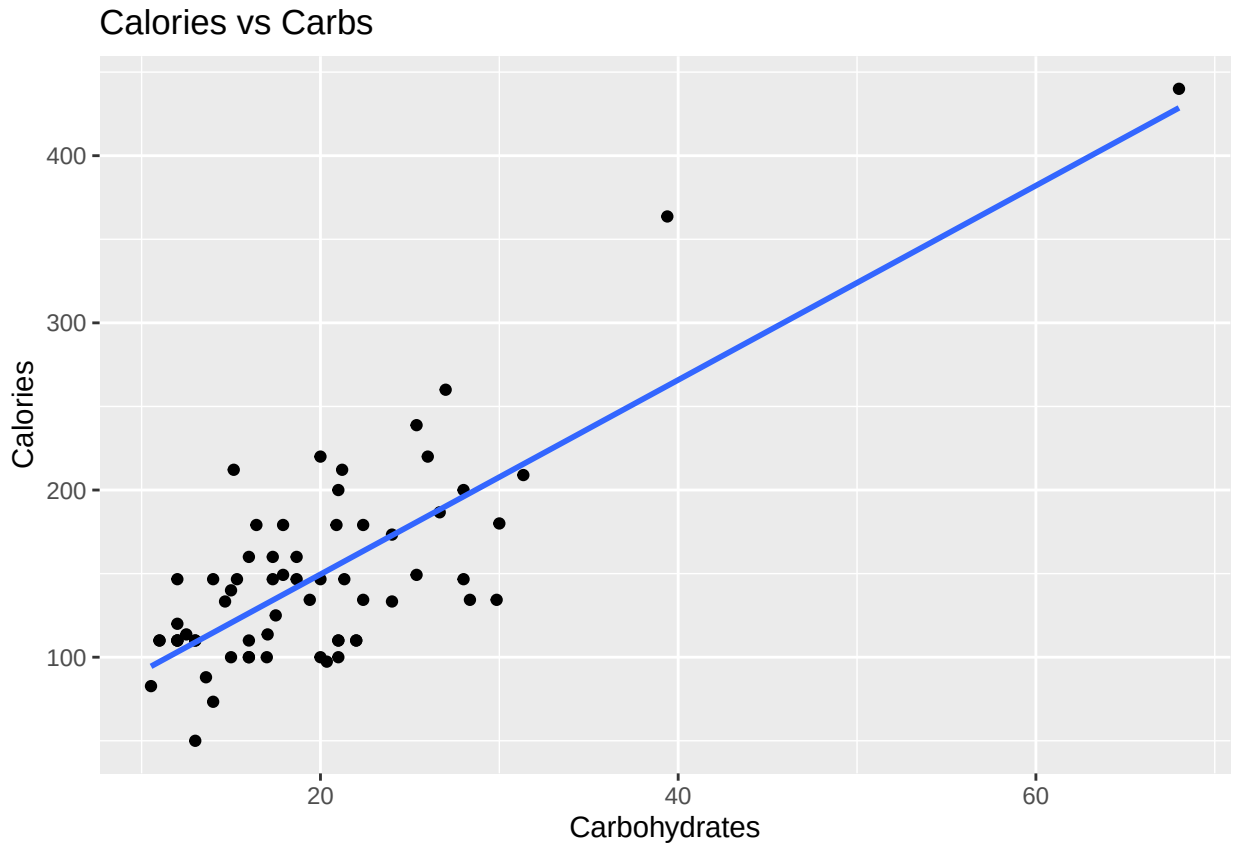



```r
# Highest value is carbohydrates.
```

### Problem 2 - (f)

```
ggplot(UScereal, aes(x = carbo , y = calories )) + geom_point() +
labs(title = "Calories vs Carbs", x = "Carbohydrates", y = "Calories") + geom_smooth(method = "lm", se =
```

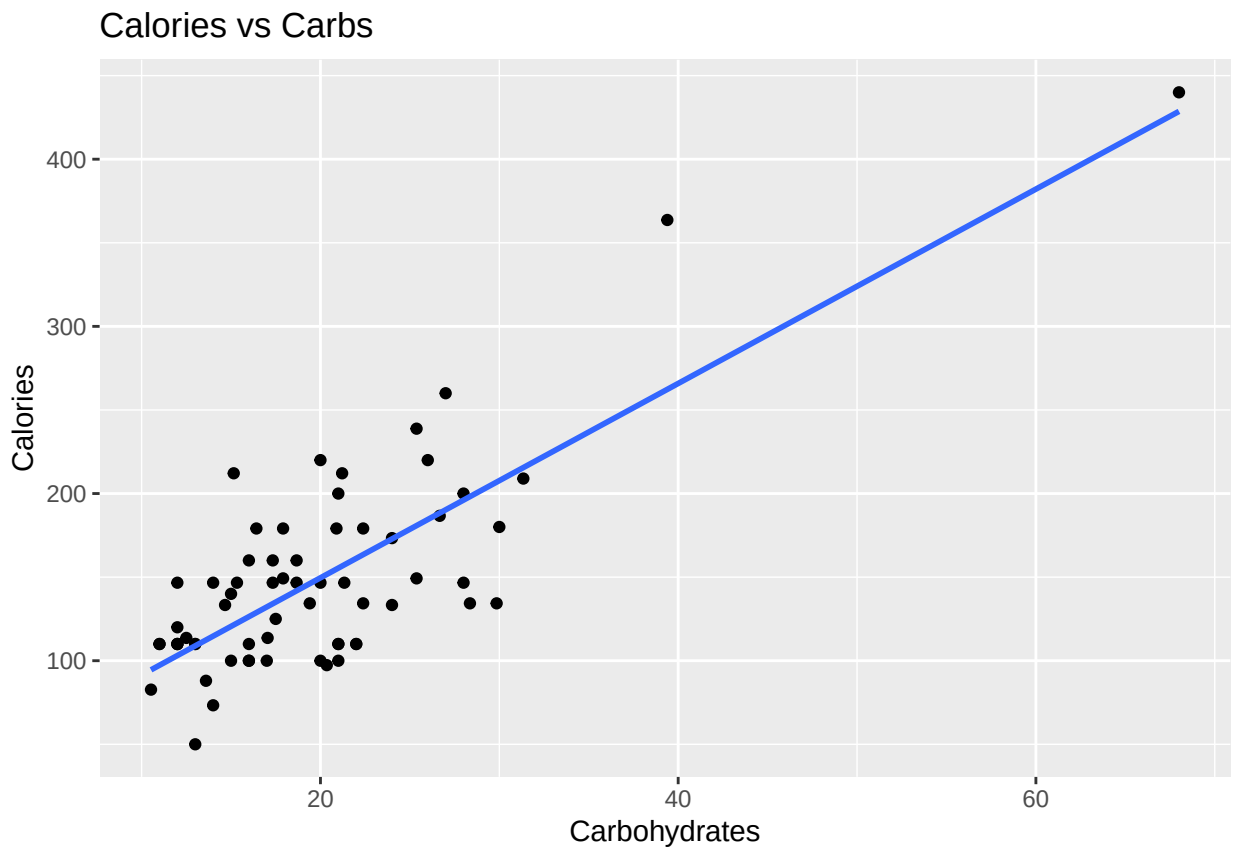
```
## `geom_smooth()` using formula = 'y ~ x'
```




## Problem 2 - (f)

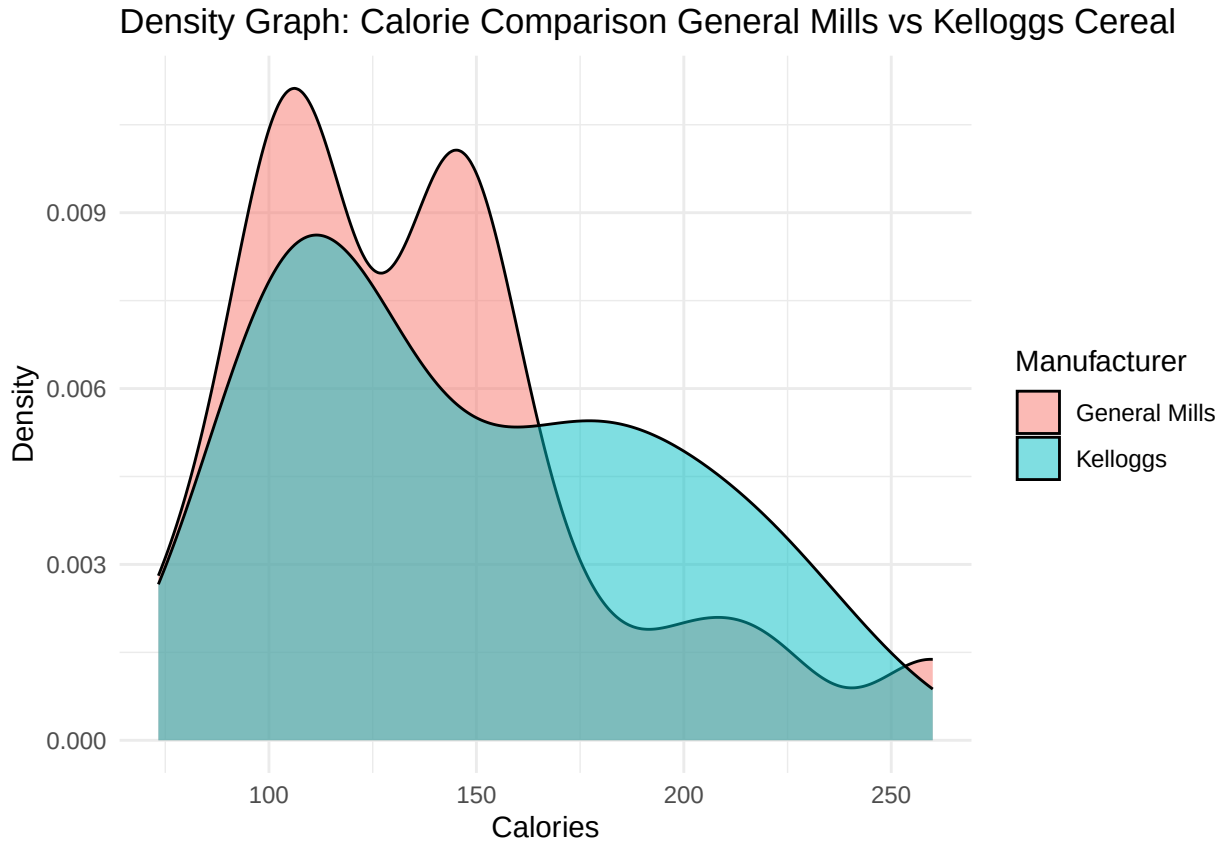
```
ggplot(UScereal, aes(x = carbo , y = calories )) + geom_point() +
labs(title = "Calories vs Carbs",
     x = "Carbohydrates",
     y = "Calories") + geom_smooth(method = "lm", se = FALSE)
```

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



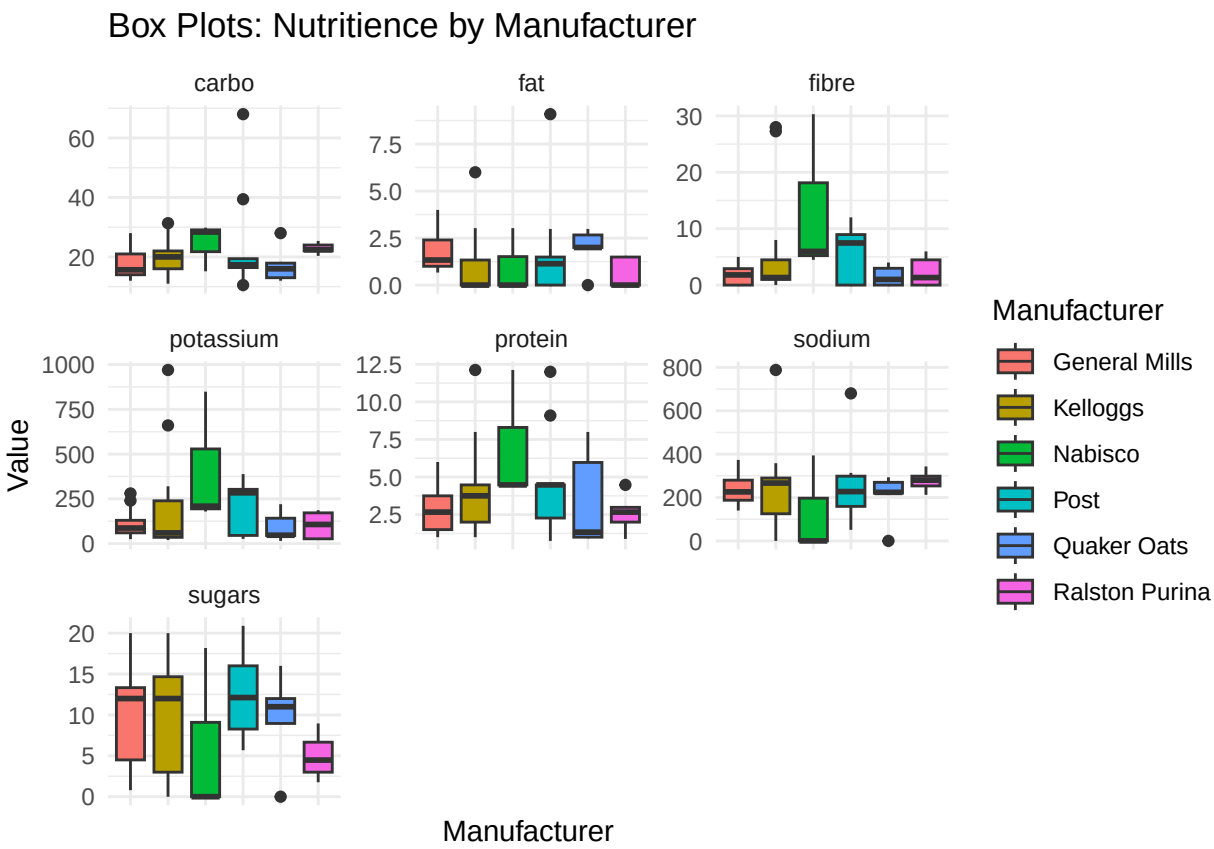

## Problem 2 - (h)

```
subGandK <- subset(UScereal, mfr %in% c("General Mills", "Kelloggs"))
ggplot(subGandK, aes(x = calories, fill = mfr)) +
  geom_density(alpha = 0.5) +
  labs(title = "Density Graph: Calorie Comparison General Mills vs Kelloggs Cereal",
       x = "Calories",
       y = "Density",
       fill = "Manufacturer") +
  theme_minimal()
```




## Problem 2 - (i)

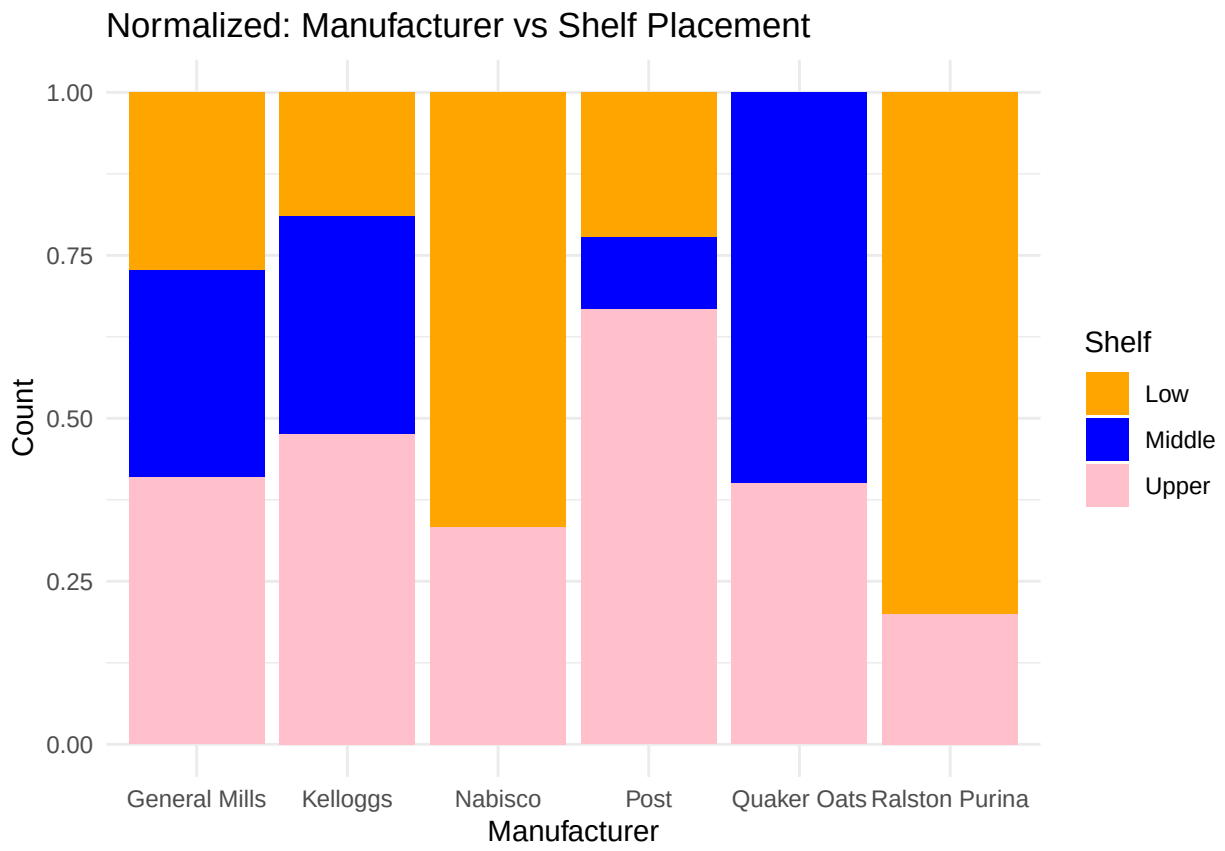
```
nutVSmfr <- UScereal[, c("mfr", "protein", "fat", "sodium", "fibre", "carbo", "sugars", "potassium")]
nutVSmfr_data <- gather(nutVSmfr, Nutrition, Value, -mfr)
ggplot(nutVSmfr_data, aes(x = mfr, y = Value, fill = mfr)) + geom_boxplot() +
facet_wrap(~ Nutrition, scales = "free", ncol = 3) + labs(title = "Box Plots: Nutritience by Manufactur
x = "Manufacturer",
y = "Value",
fill = "Manufacturer") +
theme_minimal() +
theme(axis.text.x = element_blank())
```




```
## Nabisco looks good for healthy diets. Low sugar and high protein and fibre.
```

### Problem 2 - (j)

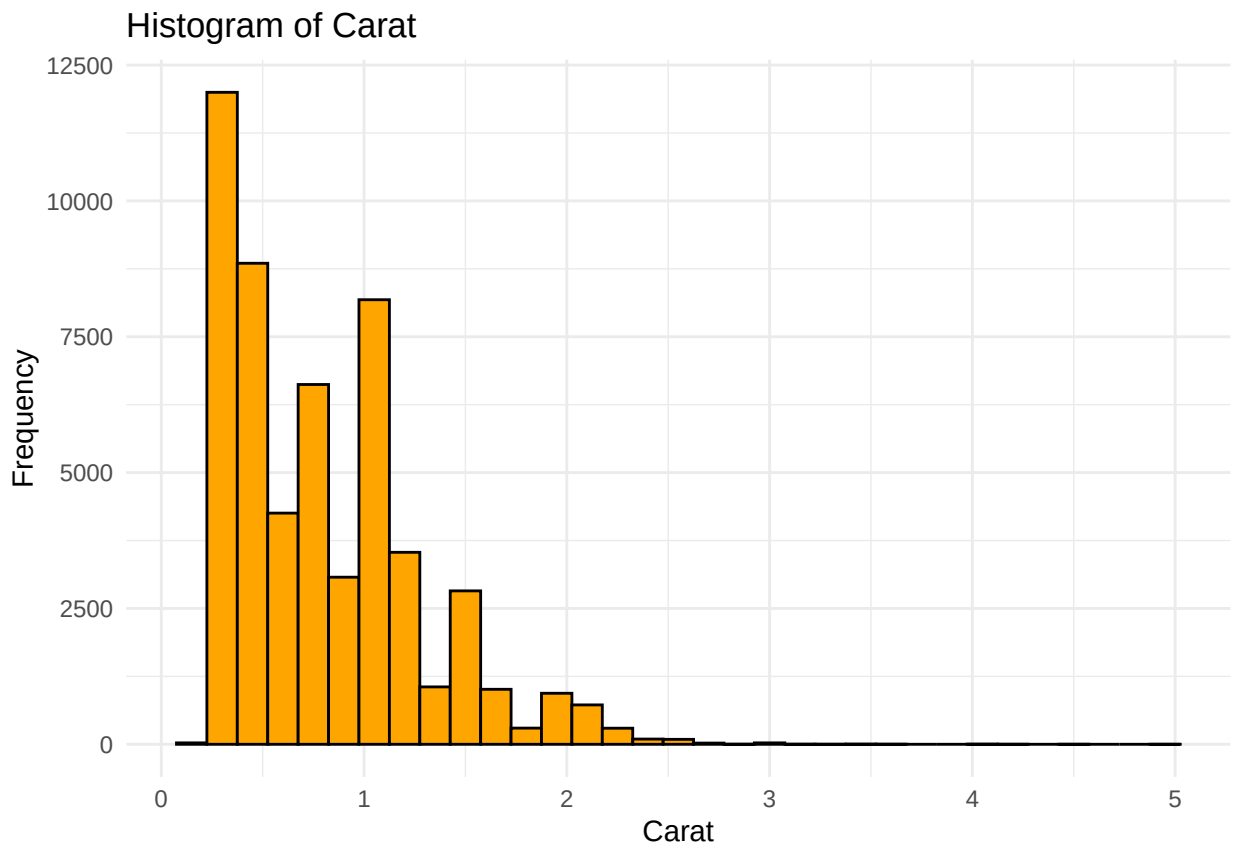
```
ggplot(UScereal, aes(x = mfr, fill = shelf)) + geom_bar(position = "fill") +
scale_fill_manual(values = c("orange", "blue", "pink")) + labs(title = "Normalized: Manufacturer vs Shel
x = "Manufacturer", y = "Count",
fill = "Shelf") +
theme_minimal()
```




## Problem 3 - (a)

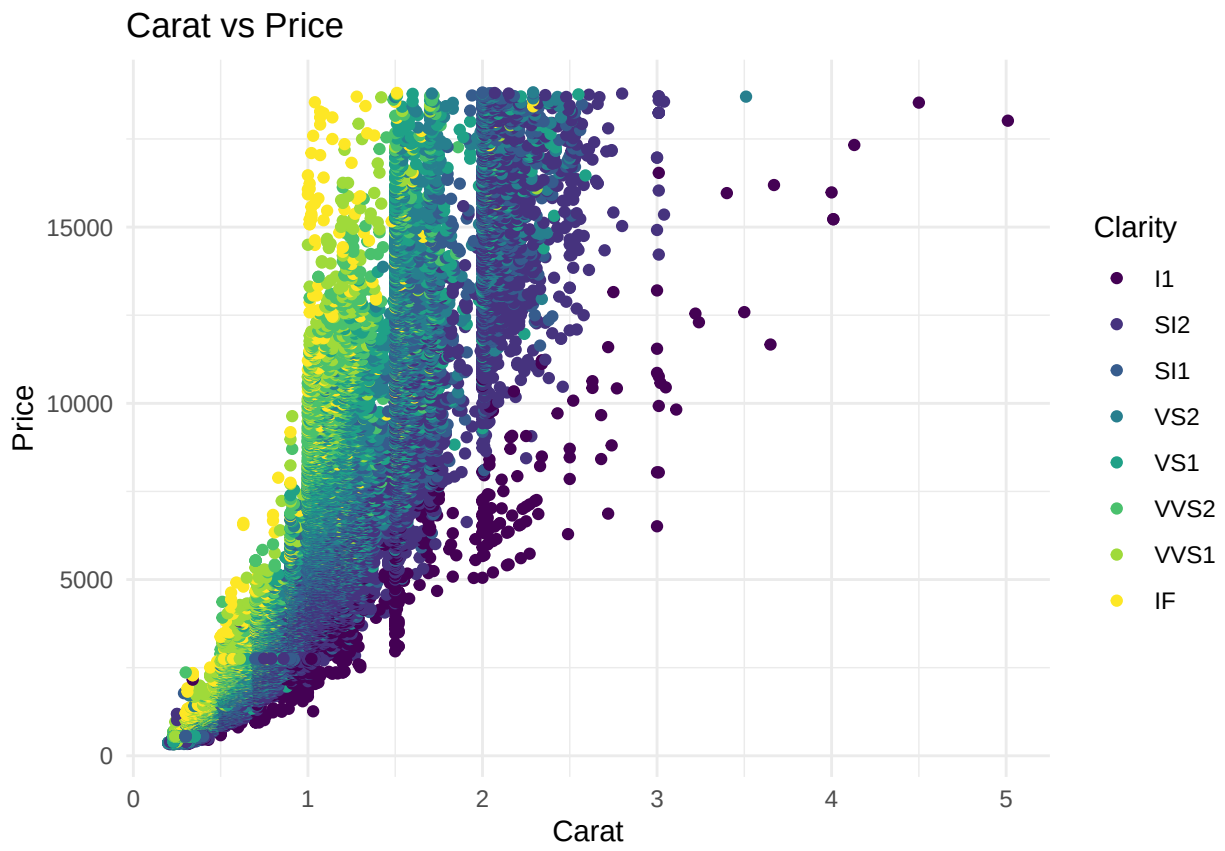
```
library(UsingR)
library(ggplot2)
library(tidyverse)
library(dplyr)

data("diamonds")
ggplot(diamonds, aes(x = carat)) +
geom_histogram(binwidth = 0.15, fill = "orange", color = "black") + labs(x = "Carat", y = "Frequency",
```




## Problem 3 - (b)

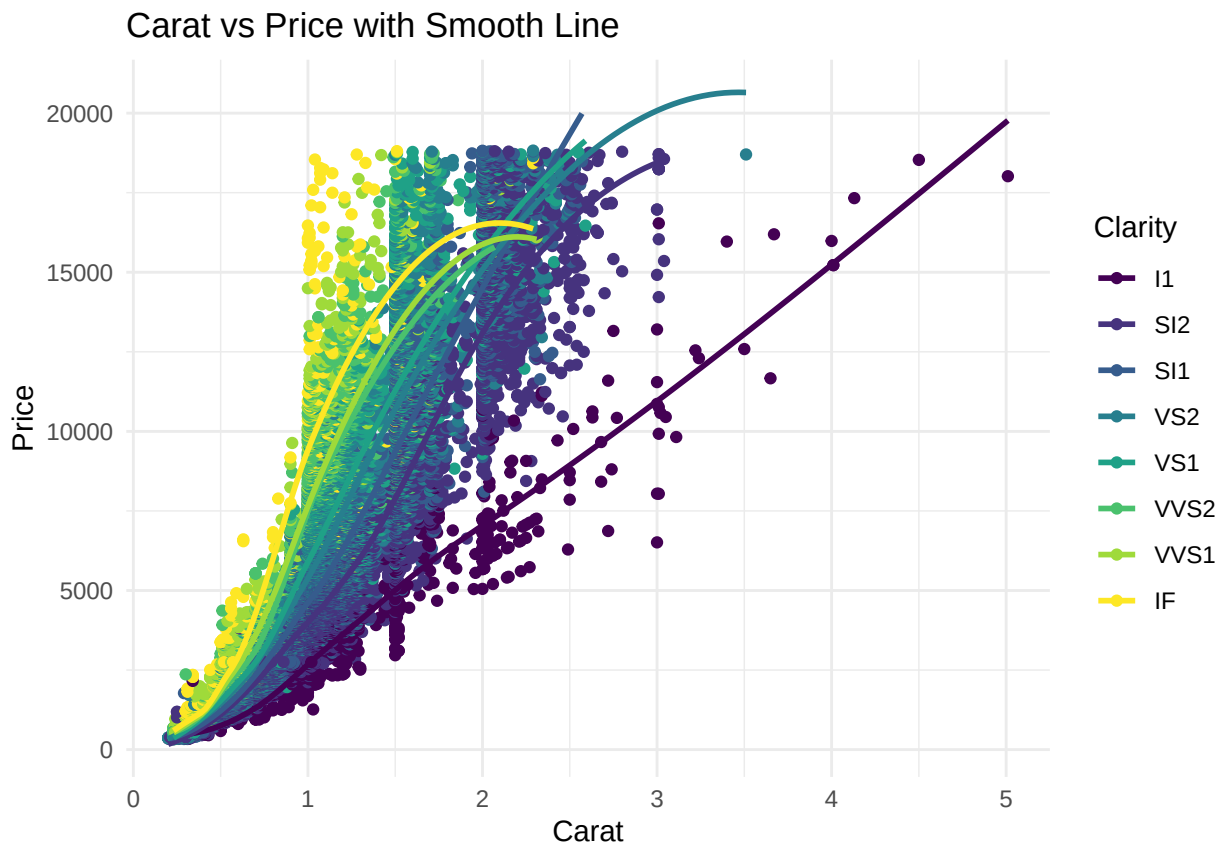
```
ggplot(diamonds, aes(x = carat, y = price, color = clarity)) +
geom_point() +
labs(x = "Carat", y = "Price", color = "Clarity", title = "Carat vs Price") + theme_minimal()
```




## Problem 3 - (c)

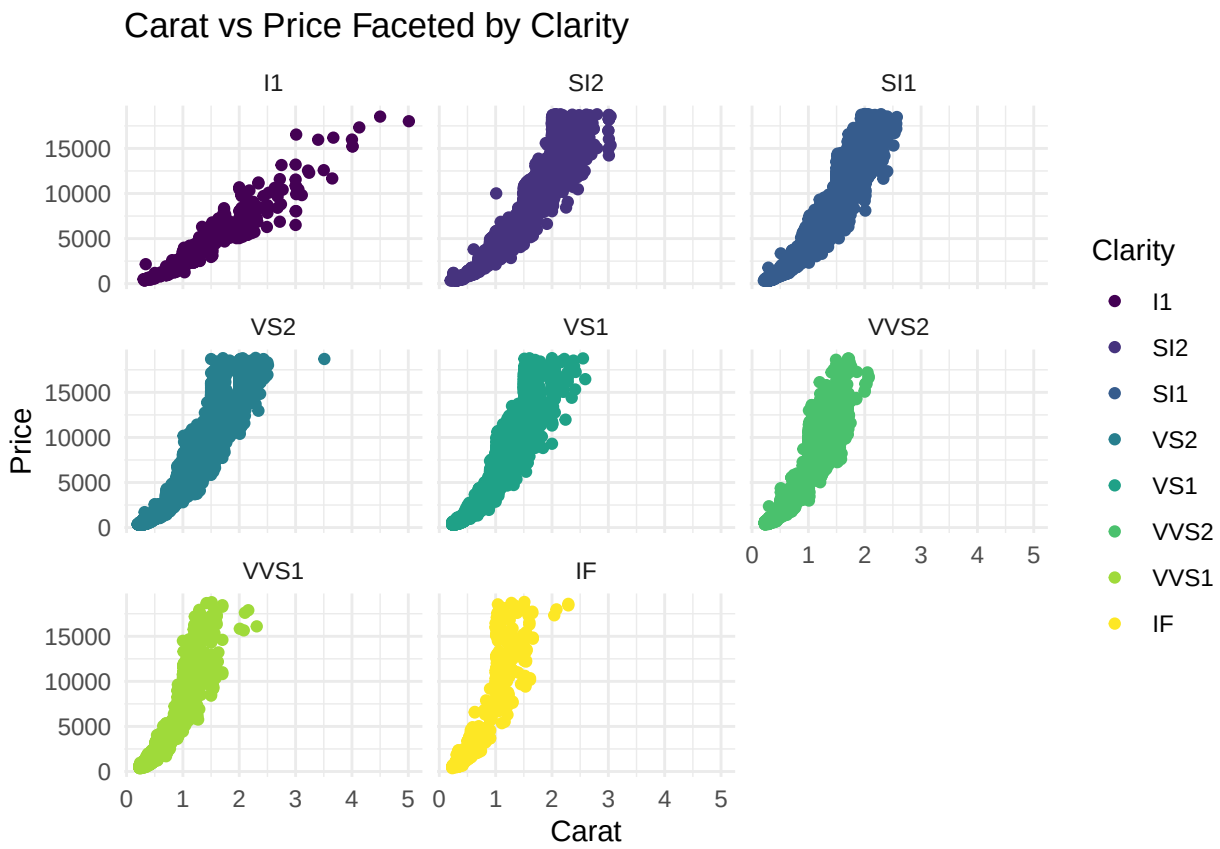
```
ggplot(diamonds, aes(x = carat, y = price, color = clarity)) +
  geom_point() +
  geom_smooth(method = "loess", se = FALSE) +  # geom_smooth inside ggplot
  labs(x = "Carat", y = "Price", color = "Clarity", title = "Carat vs Price with Smooth Line") +
  theme_minimal()
```

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



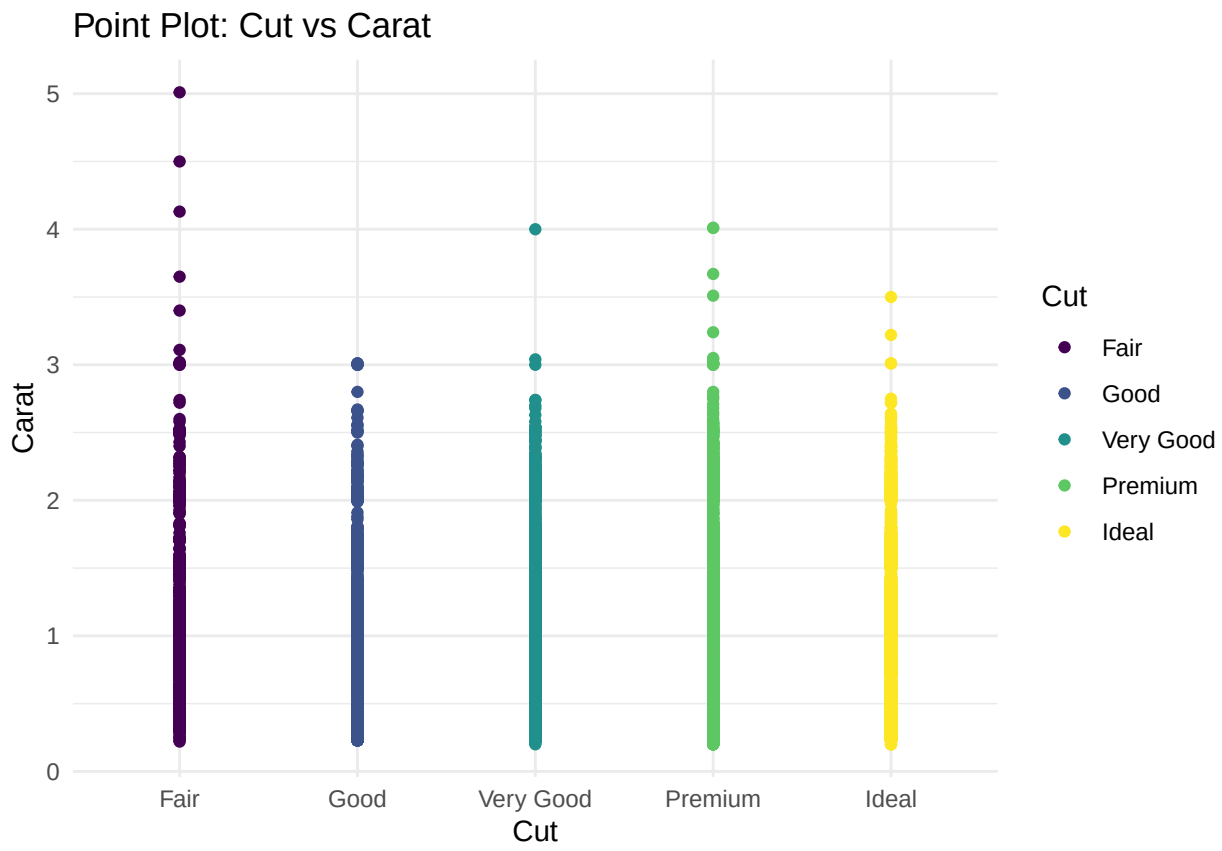

## Problem 3 - (d)

```
ggplot(diamonds, aes(x = carat, y = price, color = clarity)) +
geom_point() +
facet_wrap(~ clarity) +
labs(x = "Carat", y = "Price", color = "Clarity", title = "Carat vs Price Faceted by Clarity") +
theme_minimal()
```

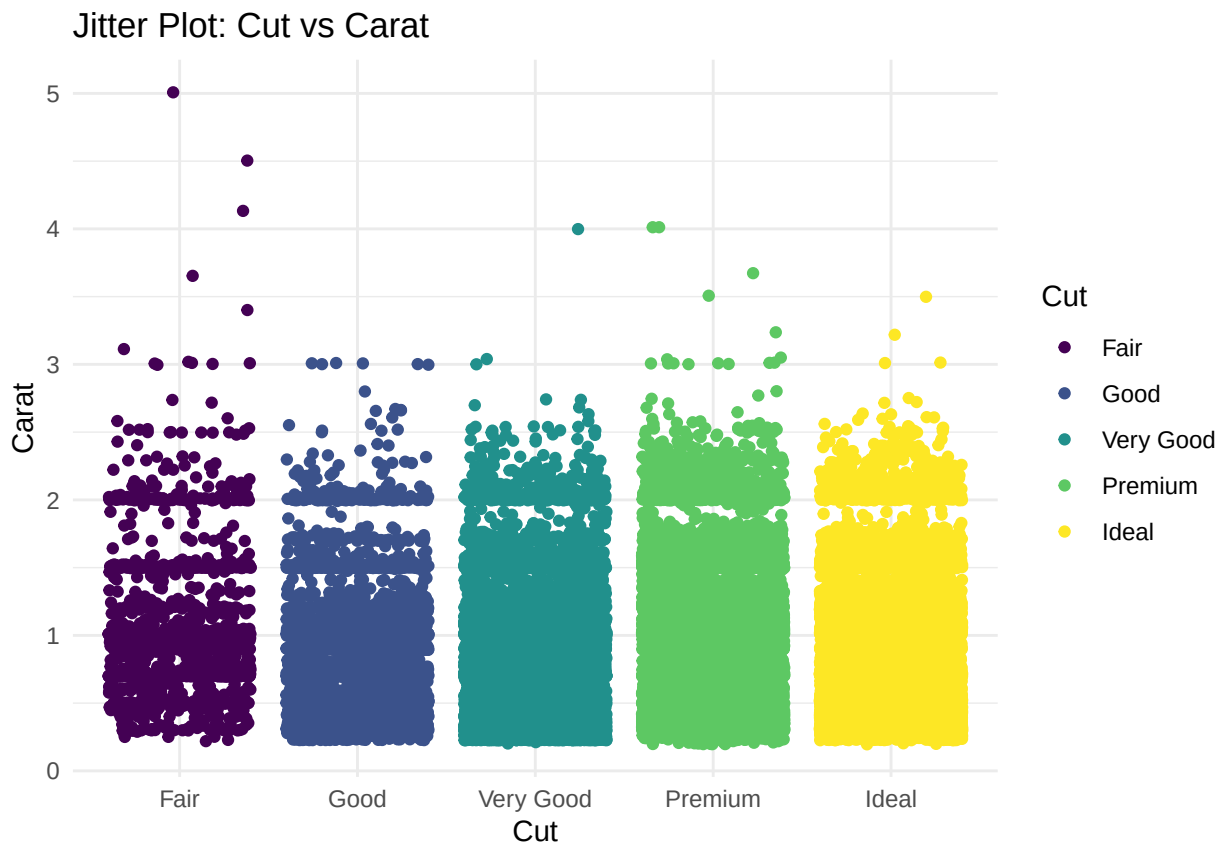



## Problem 3 - (e)

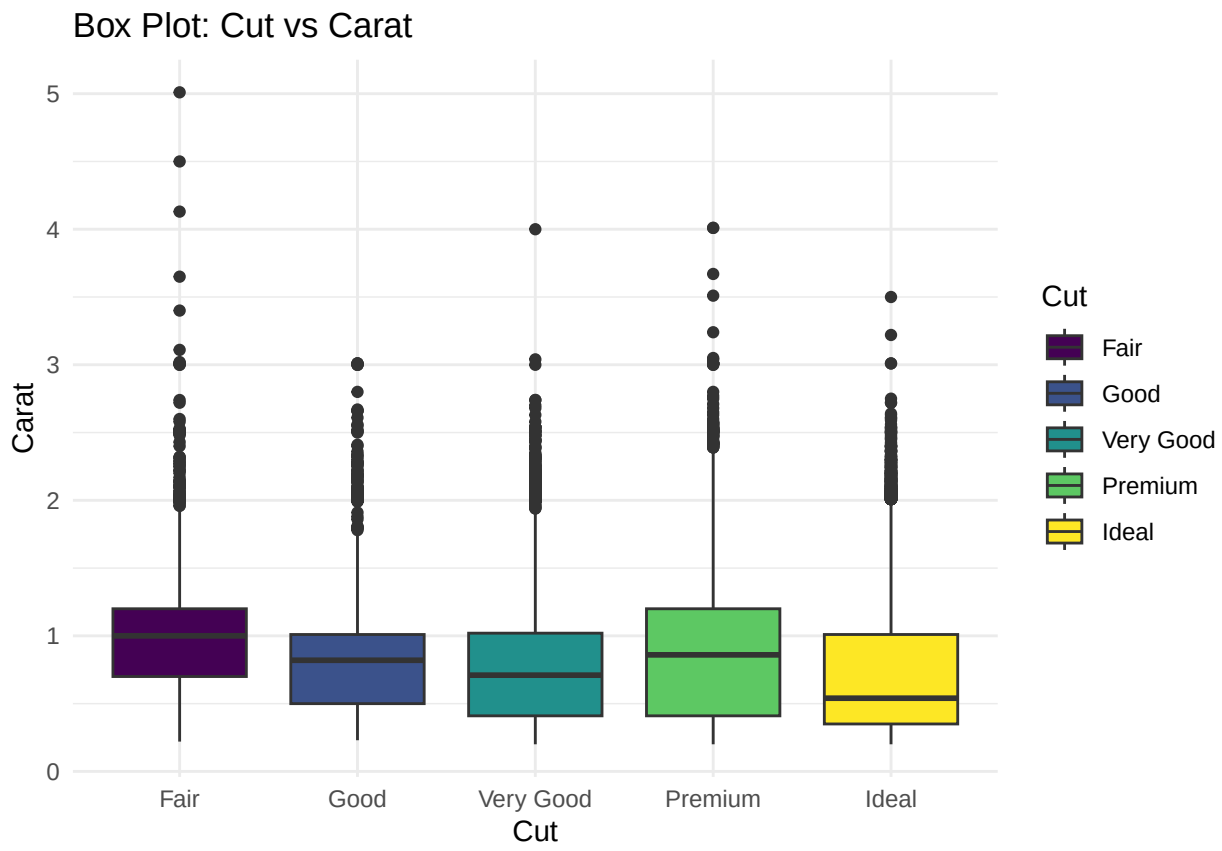
```
## Point Plot
ggplot(diamonds, aes(x = cut, y = carat, color = cut)) +
geom_point() +
labs(x = "Cut", y = "Carat", color = "Cut", title = "Point Plot: Cut vs Carat") + theme_minimal()
```



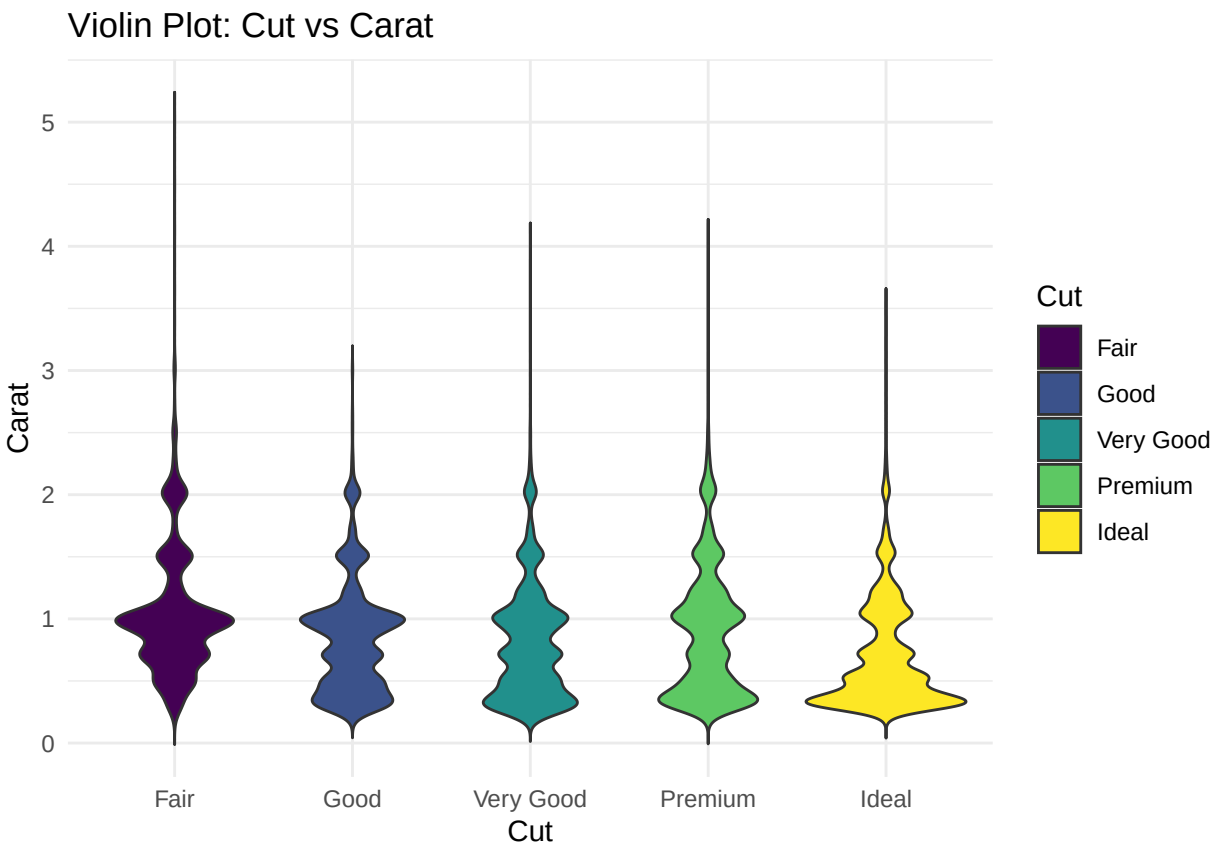

```
## Jitter Plot
ggplot(diamonds, aes(x = cut, y = carat, color = cut)) +
geom_jitter() +
labs(x = "Cut", y = "Carat", color = "Cut", title = "Jitter Plot: Cut vs Carat") + theme_minimal()
```




```
## Box Plot
ggplot(diamonds, aes(x = cut, y = carat, fill = cut)) +
geom_boxplot() +
labs(x = "Cut", y = "Carat", fill = "Cut", title = "Box Plot: Cut vs Carat") + theme_minimal()
```

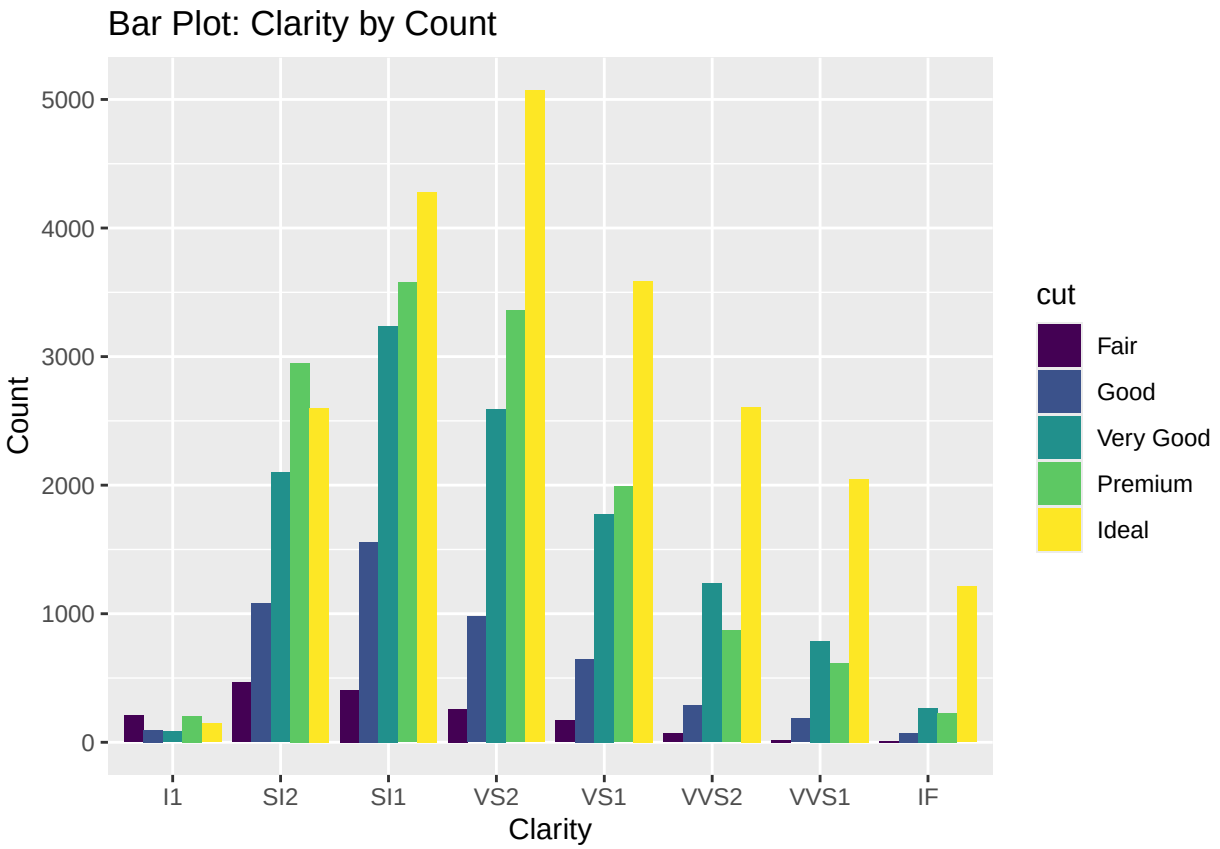



```
## Violin Plot
ggplot(diamonds, aes(x = cut, y = carat, fill = cut)) +
geom_violin(trim = FALSE) +
labs(x = "Cut", y = "Carat", fill = "Cut", title = "Violin Plot: Cut vs Carat") + theme_minimal()
```




## Problem 3 - (f)

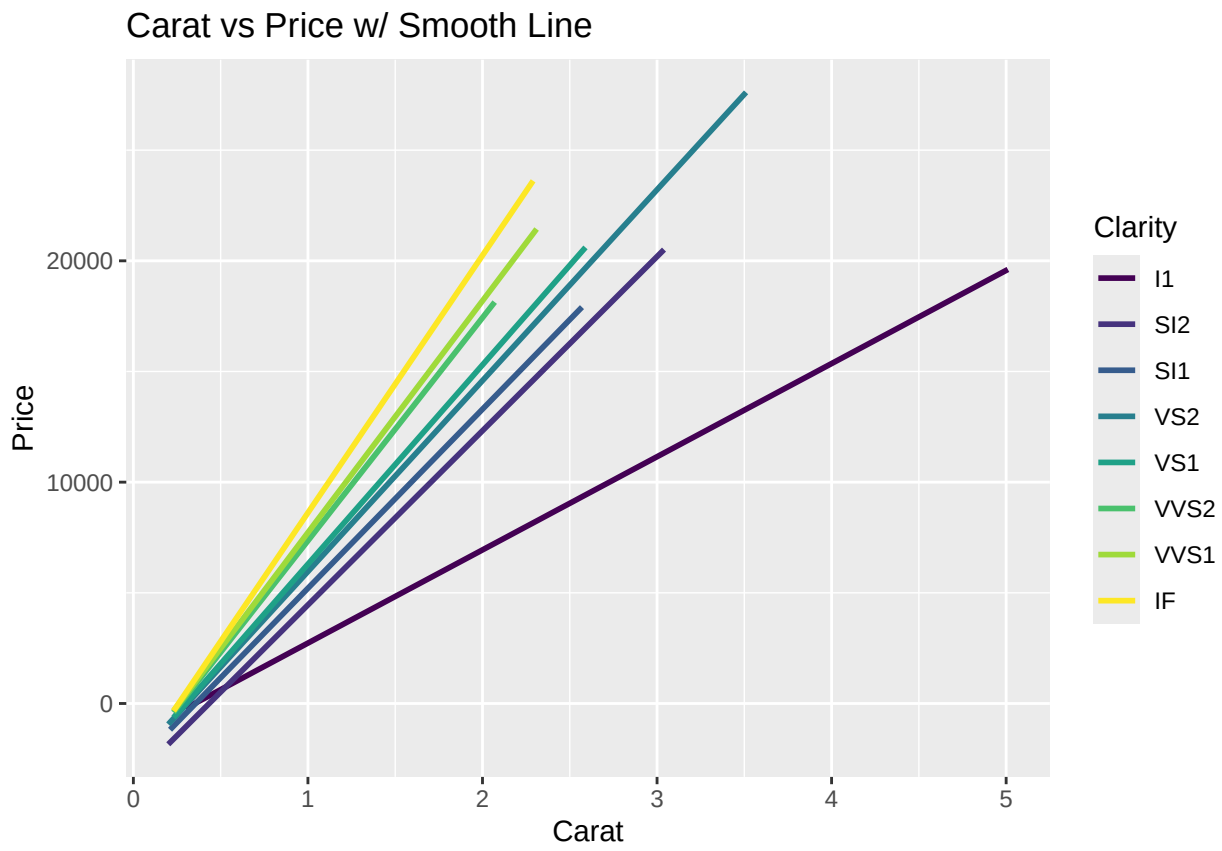
```
ggplot(diamonds, aes(x = clarity, fill = cut)) +
geom_bar(position = "dodge") +
labs(x = "Clarity", y = "Count", title = "Bar Plot: Clarity by Count") + theme(legend.position = "right
```




## Problem 3 - (g)

```
ggplot(diamonds, aes(x = carat, y = price, color = clarity)) +
geom_smooth(method = "lm", se = FALSE) +
labs(x = "Carat", y = "Price", color = "Clarity", title = "Carat vs Price w/ Smooth Line")
```

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




```
theme_minimal()
```

```
## List of 136
##  $ line                            :List of 6
##   ..$ colour       : chr "black"
##   ..$ linewidth    : num 0.5
##   ..$ linetype     : num 1
##   ..$ lineend      : chr "butt"
##   ..$ arrow        : logi FALSE
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_line" "element"
##  $ rect                            :List of 5
##   ..$ fill         : chr "white"
##   ..$ colour       : chr "black"
##   ..$ linewidth    : num 0.5
##   ..$ linetype     : num 1
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_rect" "element"
##  $ text                            :List of 11
##   ..$ family       : chr ""
##   ..$ face         : chr "plain"
##   ..$ colour       : chr "black"
##   ..$ size         : num 11
##   ..$ hjust        : num 0.5
##   ..$ vjust        : num 0.5
```

```
##   ..$ angle        : num 0
##   ..$ lineheight   : num 0.9
##   ..$ margin       : 'margin' num [1:4] 0points 0points 0points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : logi FALSE
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ title                          : NULL
##  $ aspect.ratio                   : NULL
##  $ axis.title                     : NULL
##  $ axis.title.x                   :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : NULL
##   ..$ vjust        : num 1
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 2.75points 0points 0points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.title.x.top               :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : NULL
##   ..$ vjust        : num 0
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 0points 0points 2.75points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.title.x.bottom            : NULL
##  $ axis.title.y                   :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : NULL
##   ..$ vjust        : num 1
##   ..$ angle        : num 90
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 0points 2.75points 0points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.title.y.left              : NULL
```

```
##  $ axis.title.y.right              :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : NULL
##   ..$ vjust        : num 1
##   ..$ angle        : num -90
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 0points 0points 0points 2.75points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.text                       :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : chr "grey30"
##   ..$ size         : 'rel' num 0.8
##   ..$ hjust        : NULL
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : NULL
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.text.x                     :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : NULL
##   ..$ vjust        : num 1
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 2.2points 0points 0points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.text.x.top                 :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : NULL
##   ..$ vjust        : num 0
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 0points 0points 2.2points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
```

```
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.text.x.bottom              : NULL
##  $ axis.text.y                     :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : num 1
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 0points 2.2points 0points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.text.y.left                : NULL
##  $ axis.text.y.right               :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : num 0
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 0points 0points 0points 2.2points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.text.theta                 : NULL
##  $ axis.text.r                     :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : num 0.5
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 0points 2.2points 0points 2.2points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.ticks                      : list()
##   ..- attr(*, "class")= chr [1:2] "element_blank" "element"
##  $ axis.ticks.x                    : NULL
##  $ axis.ticks.x.top                : NULL
##  $ axis.ticks.x.bottom             : NULL
##  $ axis.ticks.y                    : NULL
##  $ axis.ticks.y.left               : NULL
##  $ axis.ticks.y.right              : NULL
```

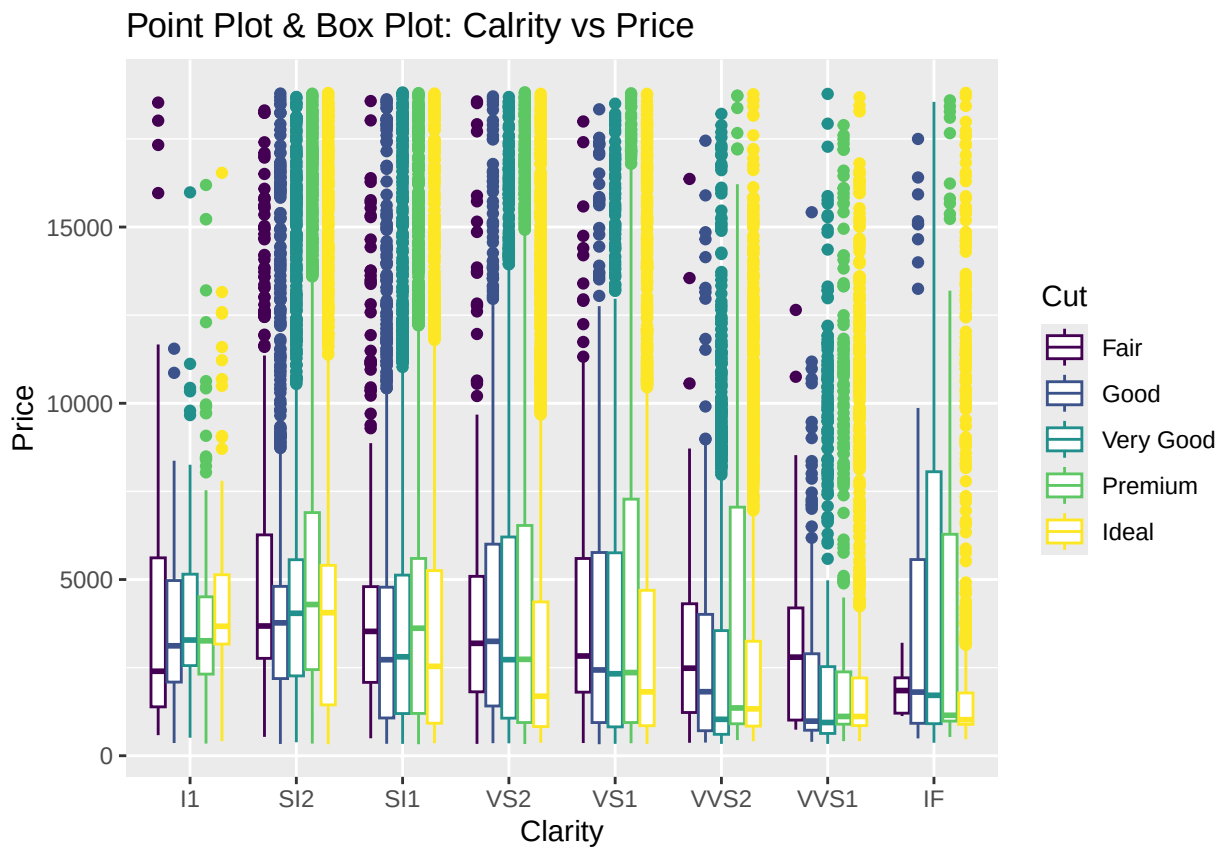
```
##  $ axis.ticks.theta               : NULL
##  $ axis.ticks.r                   : NULL
##  $ axis.minor.ticks.x.top         : NULL
##  $ axis.minor.ticks.x.bottom      : NULL
##  $ axis.minor.ticks.y.left        : NULL
##  $ axis.minor.ticks.y.right       : NULL
##  $ axis.minor.ticks.theta         : NULL
##  $ axis.minor.ticks.r             : NULL
##  $ axis.ticks.length              : 'simpleUnit' num 2.75points
##   ..- attr(*, "unit")= int 8
##  $ axis.ticks.length.x            : NULL
##  $ axis.ticks.length.x.top        : NULL
##  $ axis.ticks.length.x.bottom     : NULL
##  $ axis.ticks.length.y            : NULL
##  $ axis.ticks.length.y.left       : NULL
##  $ axis.ticks.length.y.right      : NULL
##  $ axis.ticks.length.theta        : NULL
##  $ axis.ticks.length.r            : NULL
##  $ axis.minor.ticks.length        : 'rel' num 0.75
##  $ axis.minor.ticks.length.x      : NULL
##  $ axis.minor.ticks.length.x.top  : NULL
##  $ axis.minor.ticks.length.x.bottom: NULL
##  $ axis.minor.ticks.length.y      : NULL
##  $ axis.minor.ticks.length.y.left  : NULL
##  $ axis.minor.ticks.length.y.right : NULL
##  $ axis.minor.ticks.length.theta   : NULL
##  $ axis.minor.ticks.length.r      : NULL
##  $ axis.line                      : list()
##   ..- attr(*, "class")= chr [1:2] "element_blank" "element"
##  $ axis.line.x                    : NULL
##  $ axis.line.x.top                : NULL
##  $ axis.line.x.bottom             : NULL
##  $ axis.line.y                    : NULL
##  $ axis.line.y.left               : NULL
##  $ axis.line.y.right              : NULL
##  $ axis.line.theta                : NULL
##  $ axis.line.r                    : NULL
##  $ legend.background              : list()
##   ..- attr(*, "class")= chr [1:2] "element_blank" "element"
##  $ legend.margin                  : 'margin' num [1:4] 5.5points 5.5points 5.5points 5.5points
##   ..- attr(*, "unit")= int 8
##  $ legend.spacing                 : 'simpleUnit' num 11points
##   ..- attr(*, "unit")= int 8
##  $ legend.spacing.x               : NULL
##  $ legend.spacing.y               : NULL
##  $ legend.key                     : list()
##   ..- attr(*, "class")= chr [1:2] "element_blank" "element"
##  $ legend.key.size                : 'simpleUnit' num 1.2lines
##   ..- attr(*, "unit")= int 3
##  $ legend.key.height              : NULL
##  $ legend.key.width               : NULL
##  $ legend.key.spacing             : 'simpleUnit' num 5.5points
##   ..- attr(*, "unit")= int 8
##  $ legend.key.spacing.x           : NULL
```

```
##  $ legend.key.spacing.y           : NULL
##  $ legend.frame                   : NULL
##  $ legend.ticks                   : NULL
##  $ legend.ticks.length            : 'rel' num 0.2
##  $ legend.axis.line               : NULL
##  $ legend.text                    :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : 'rel' num 0.8
##   ..$ hjust        : NULL
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : NULL
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ legend.text.position           : NULL
##  $ legend.title                   :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : num 0
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : NULL
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ legend.title.position          : NULL
##  $ legend.position                : chr "right"
##  $ legend.position.inside         : NULL
##  $ legend.direction               : NULL
##  $ legend.byrow                   : NULL
##  $ legend.justification           : chr "center"
##  $ legend.justification.top       : NULL
##  $ legend.justification.bottom    : NULL
##  $ legend.justification.left      : NULL
##  $ legend.justification.right     : NULL
##  $ legend.justification.inside    : NULL
##  $ legend.location                : NULL
##  $ legend.box                     : NULL
##  $ legend.box.just                : NULL
##  $ legend.box.margin              : 'margin' num [1:4] 0cm 0cm 0cm 0cm
##   ..- attr(*, "unit")= int 1
##  $ legend.box.background          : list()
##   ..- attr(*, "class")= chr [1:2] "element_blank" "element"
##  $ legend.box.spacing             : 'simpleUnit' num 11points
##   ..- attr(*, "unit")= int 8
##   [list output truncated]
##  - attr(*, "class")= chr [1:2] "theme" "gg"
```

```
##  - attr(*, "complete")= logi TRUE
##  - attr(*, "validate")= logi TRUE
```

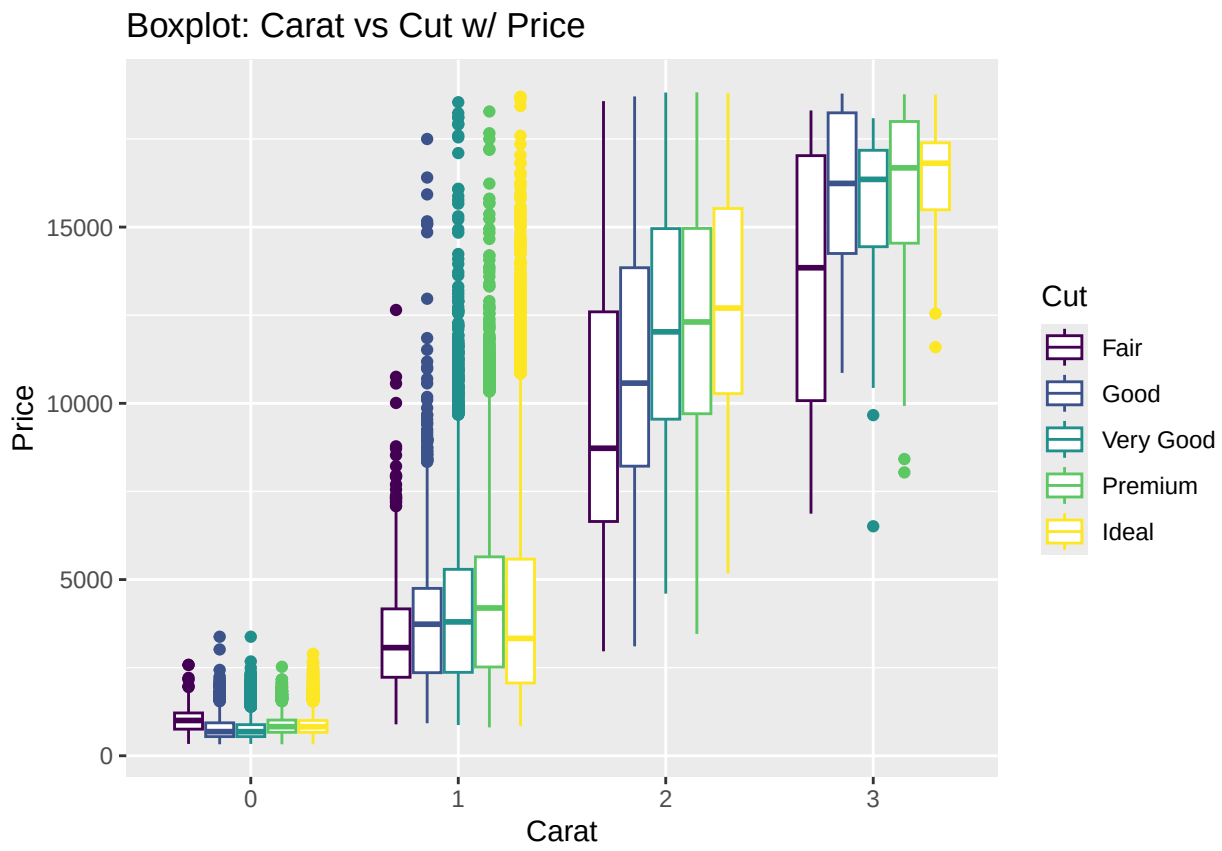
### Problem 3 - (h)

```
ggplot(diamonds, aes(x = clarity, y = price, color = cut)) +
geom_boxplot()+
labs(x = "Clarity", y = "Price", color = "Cut", title = "Point Plot & Box Plot: Calrity vs Price")
```




## Problem 3 - (i)

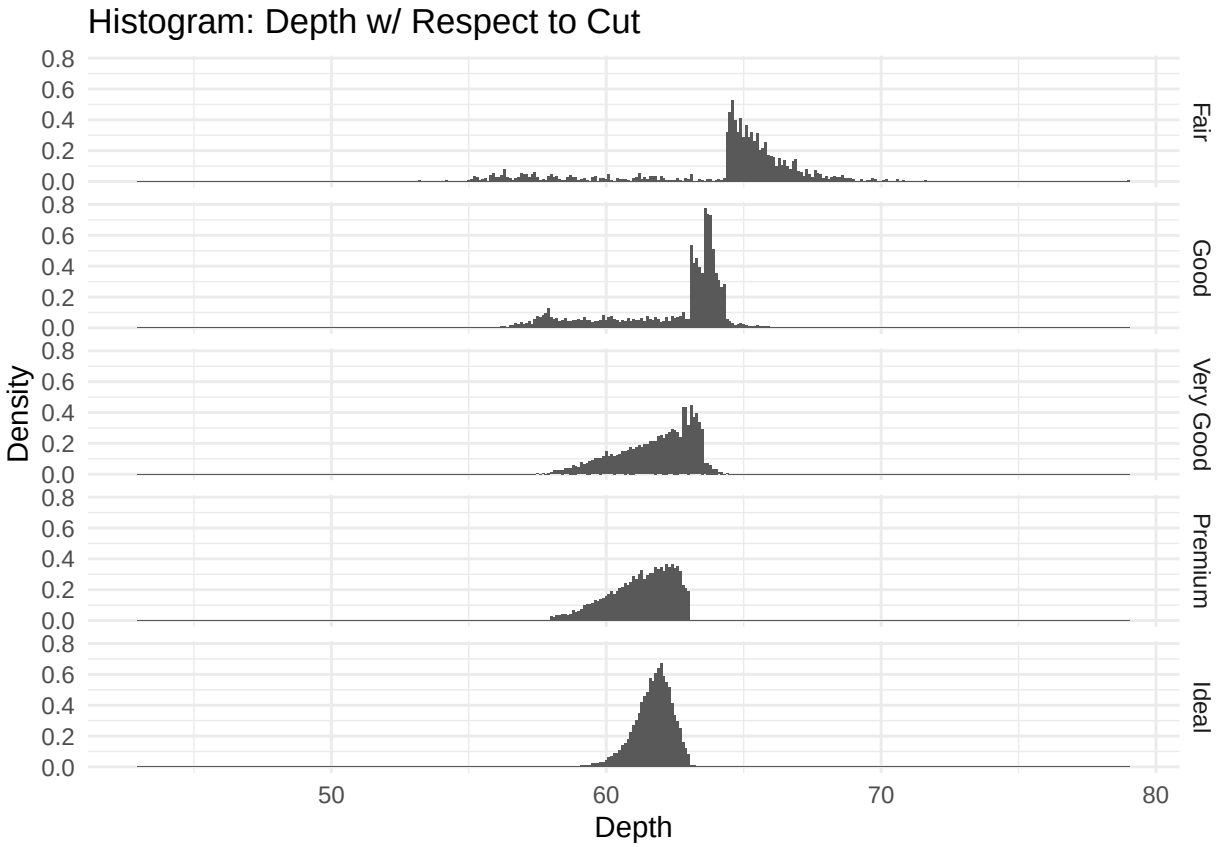
```
diamonds %>% mutate(carat = factor(round(carat),levels = c(0,1,2,3))) %>% filter(!is.na(carat)) %>%
ggplot(aes(x = carat, y = price, color = cut)) +
geom_boxplot() +
labs(x = "Carat", y = "Price", color = "Cut", title = "Boxplot: Carat vs Cut w/ Price")
```




## Problem 3 - (j)

```
ggplot(data = diamonds, aes(x = depth, y = ..density..)) + #Note: sum = 1
geom_histogram(binwidth = .10) +
facet_wrap(~ cut, ncol = 1, strip.position = "right") +
labs(x = "Depth", y = "Density", title = "Histogram: Depth w/ Respect to Cut") + theme_minimal()
```

```
## Warning: The dot-dot notation (‘..density..‘) was deprecated in ggplot2 3.4.0.
## i Please use ‘after_stat(density)‘ instead.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```


Histogram: Depth w/ Respect to Cut
Density
Depth
Fair
Good
Very Good
Premium
Ideal
0.0
0.2
0.4
0.6
0.8
50
60
70
80