Stat 292 - Recitation 6

R-Markdown - Visualization

Orçun Oltulu

21 / 4 / 2021

First load 'ggplot2' package.

```
# install.packages("ggplot2")
library(ggplot2)
```

Exercise 1:

Part A:

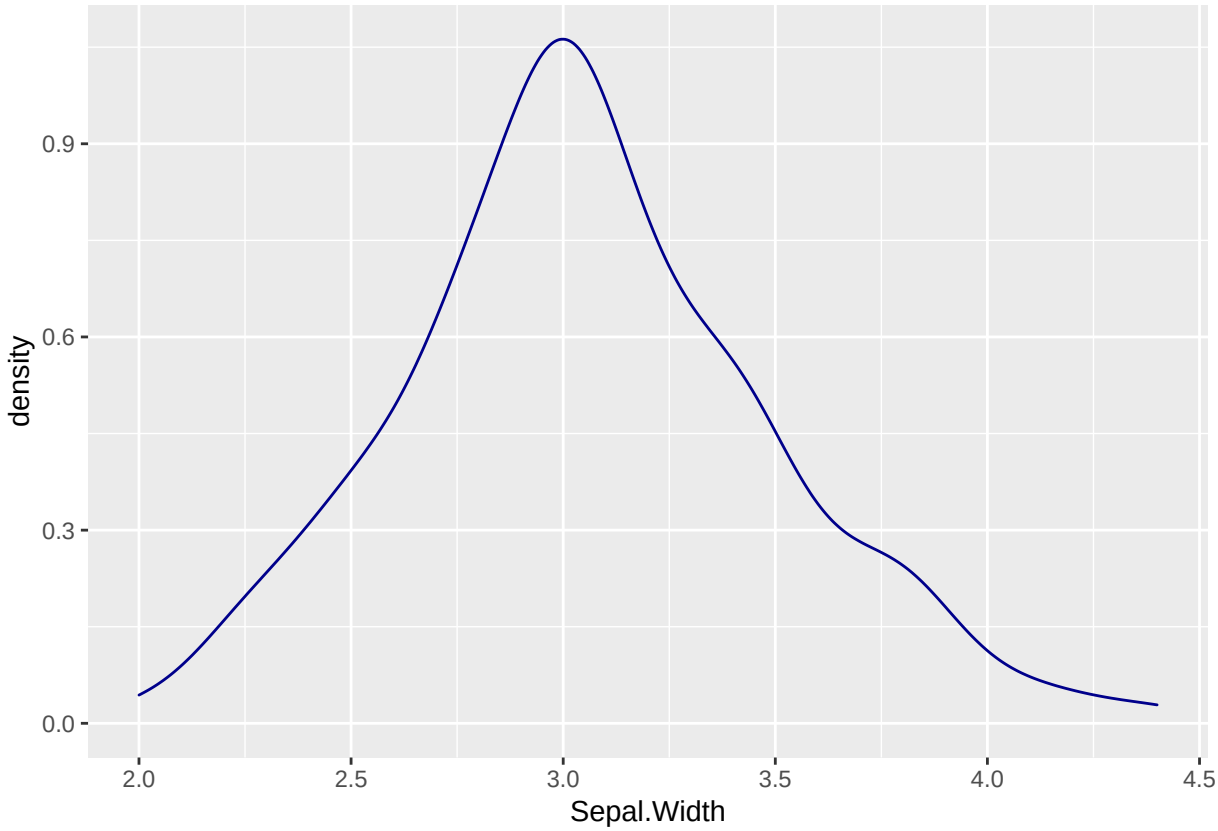
Load iris data set from ISLR package.

```
library(ISLR)
data(iris)
```

Part B:

Obtain a density plot for Sepal Width variable.

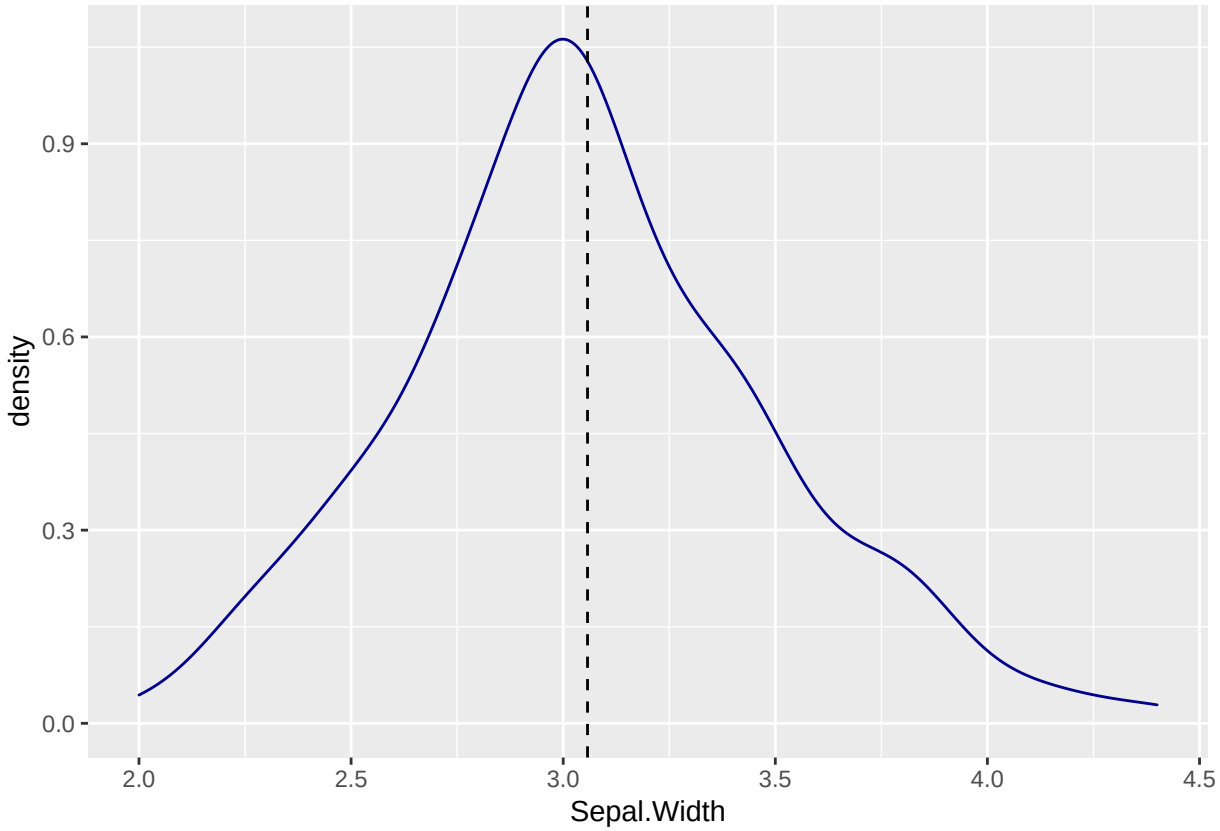
```
ggplot(iris, aes(Sepal.Width)) +
  geom_density(col = "Dark Blue")
```



Part C:

Now, add a mean line (a vertical line passes through mean sepal width) to the previous density plot.

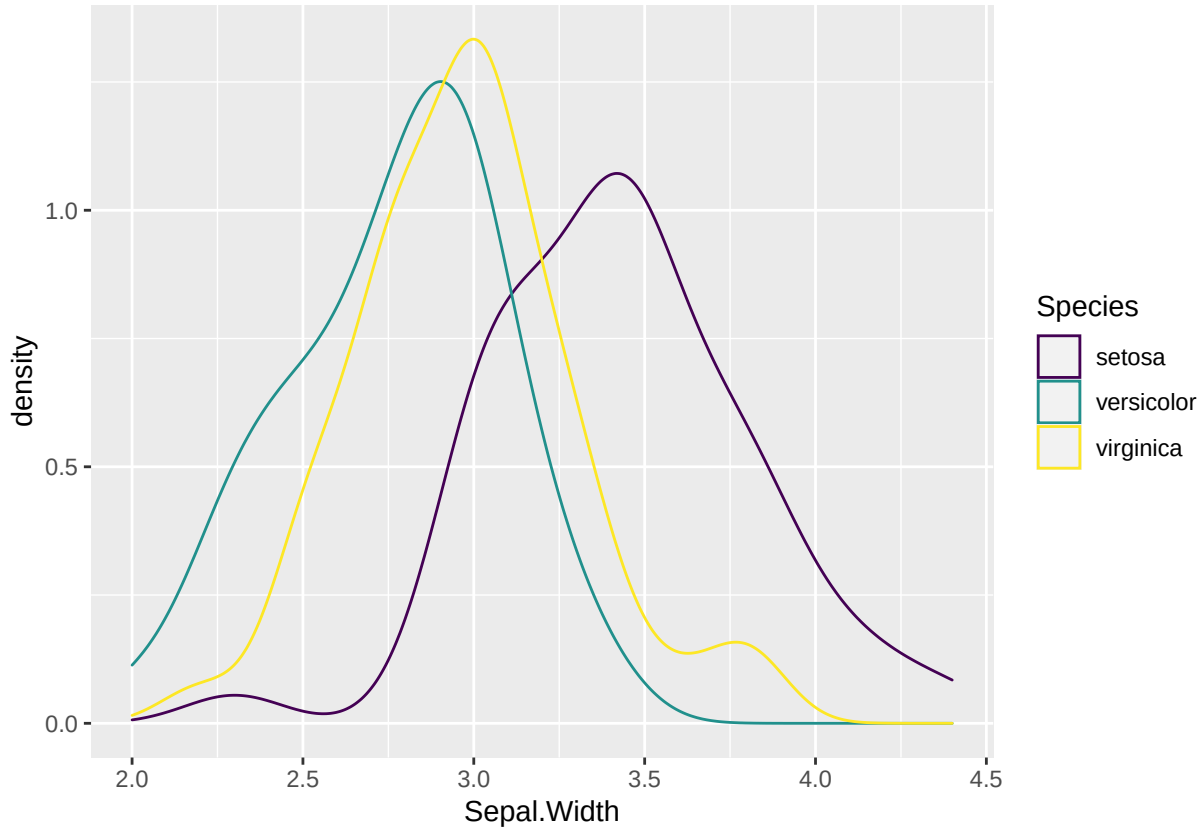
```
ggplot(iris, aes(Sepal.Width)) +
  geom_density(col = "Dark Blue") +
  geom_vline(aes(xintercept = mean(Sepal.Width)), linetype = 2)
```



Part D:

Draw density lines for Sepal Width for each Species in one single plot.

```
ggplot(iris, aes(Sepal.Width, color = Species)) +
  geom_density() +
  scale_color_viridis_d()
```

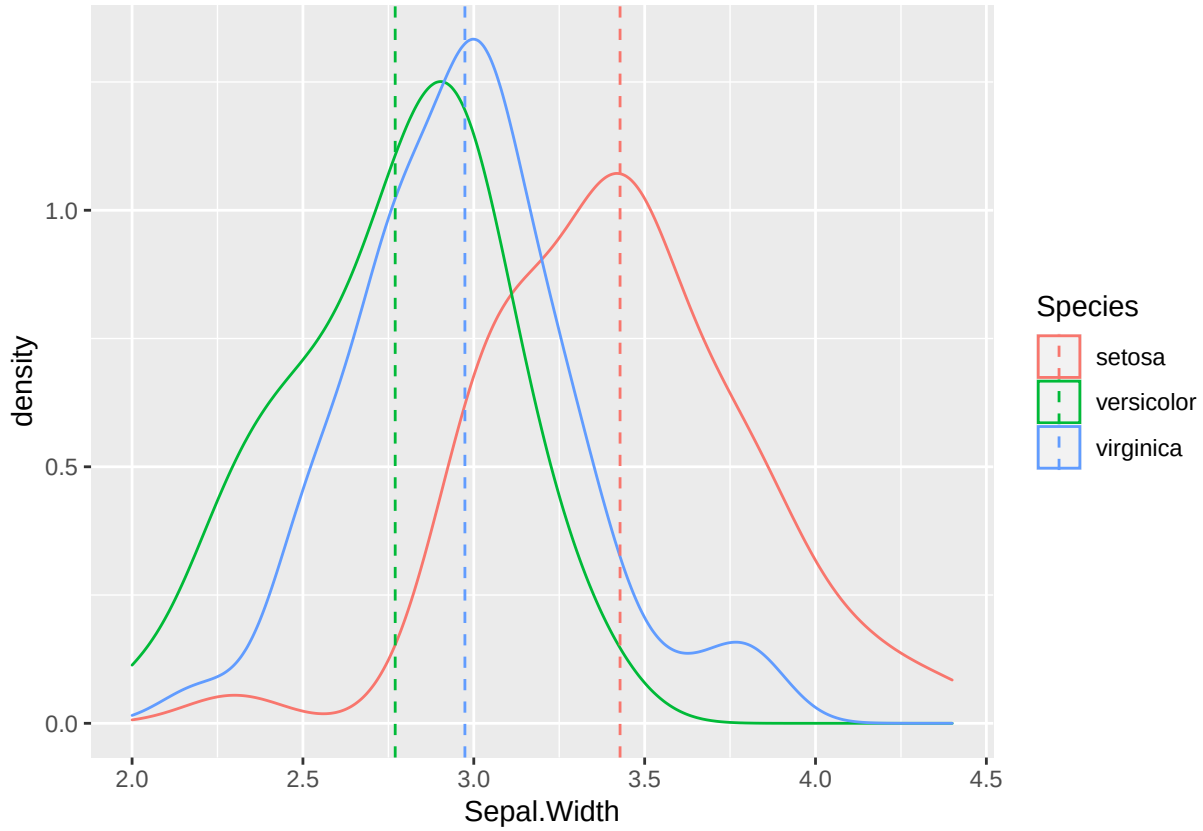


Part E:

Now, add mean line in the same plot for each group.

```
means <- tapply(iris$Sepal.Width, iris$Species, mean)
sw_means <- data.frame(Species = names(means), means = means)

ggplot(iris, aes(Sepal.Width, color = Species)) +
  geom_density() +
  geom_vline(data = sw_means,
             aes(xintercept = means, color = Species),
             linetype = 2)
```



Exercise 2:

Part A:

Import EconomistData.csv data set into R. Check the structure of the data set.

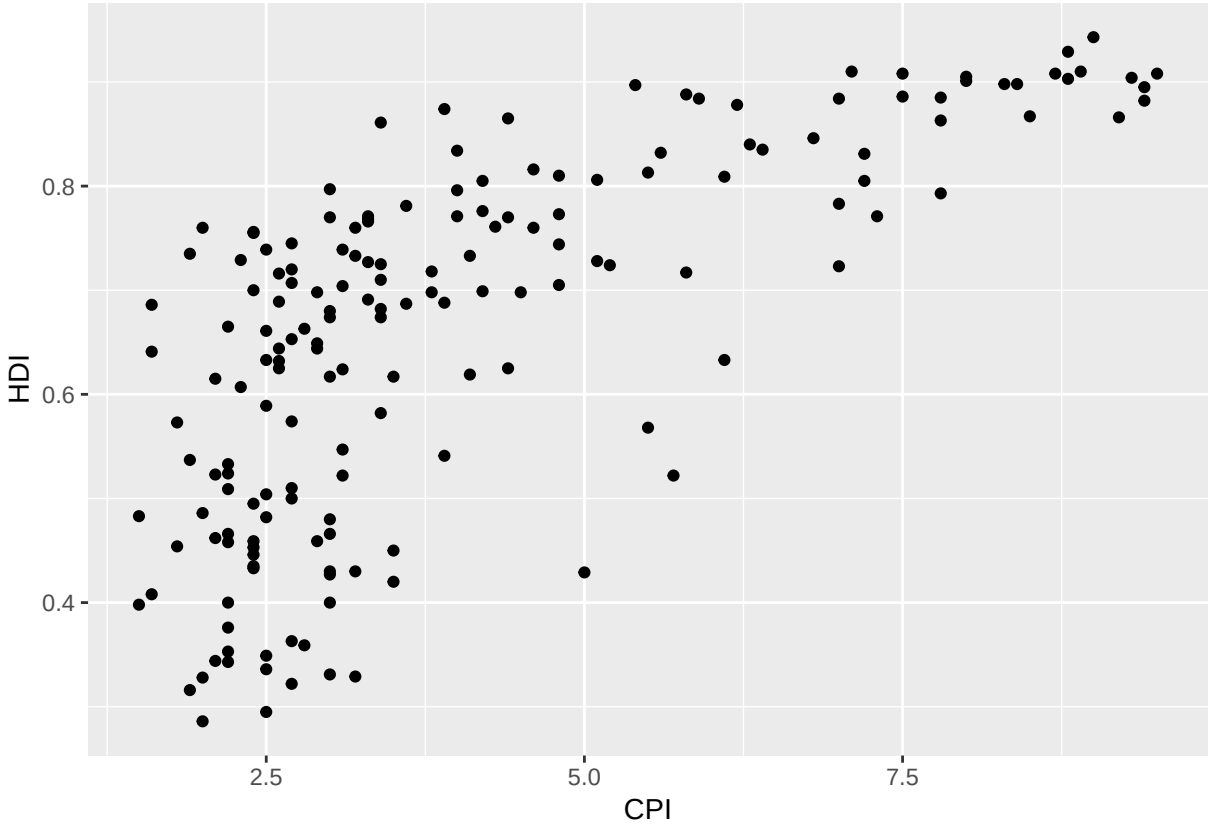
```
econ_data <- read.csv("EconomistData.csv")
str(econ_data)
```

```
## 'data.frame':    173 obs. of  5 variables:
##  $ Country : chr  "Afghanistan" "Albania" "Algeria" "Angola" ...
##  $ HDI.Rank: int  172 70 96 148 45 86 2 19 91 53 ...
##  $ HDI     : num  0.398 0.739 0.698 0.486 0.797 0.716 0.929 0.885 0.7 0.771 ...
##  $ CPI     : num  1.5 3.1 2.9 2 3 2.6 8.8 7.8 2.4 7.3 ...
##  $ Region  : chr  "Asia Pacific" "East EU Cemt Asia" "MENA" "SSA" ...
```

Part B:

Create a scatter plot with CPI on the x axis and HDI on the y axis.

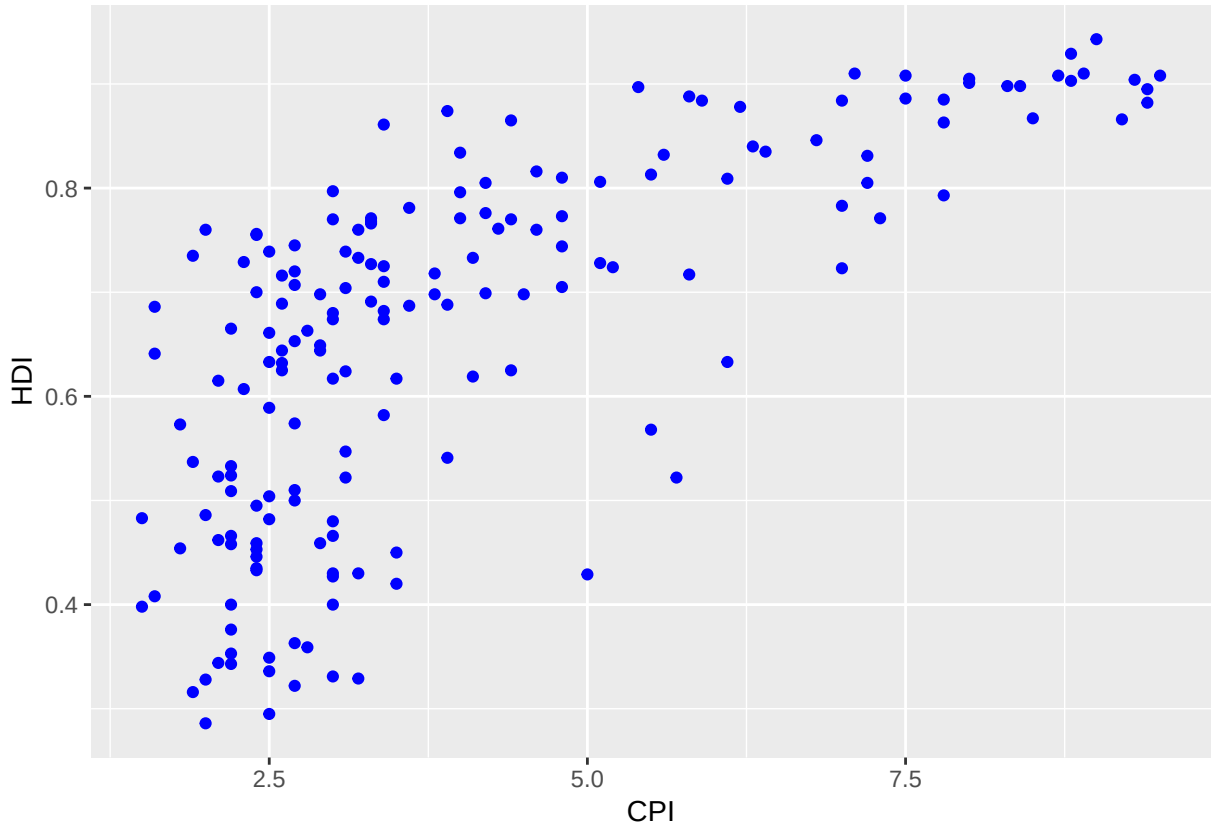
```
ggplot(econ_data, aes(x = CPI, y = HDI)) +
 geom_point()
```



Part C:

Color the points in the CPI vs HDI plot blue.

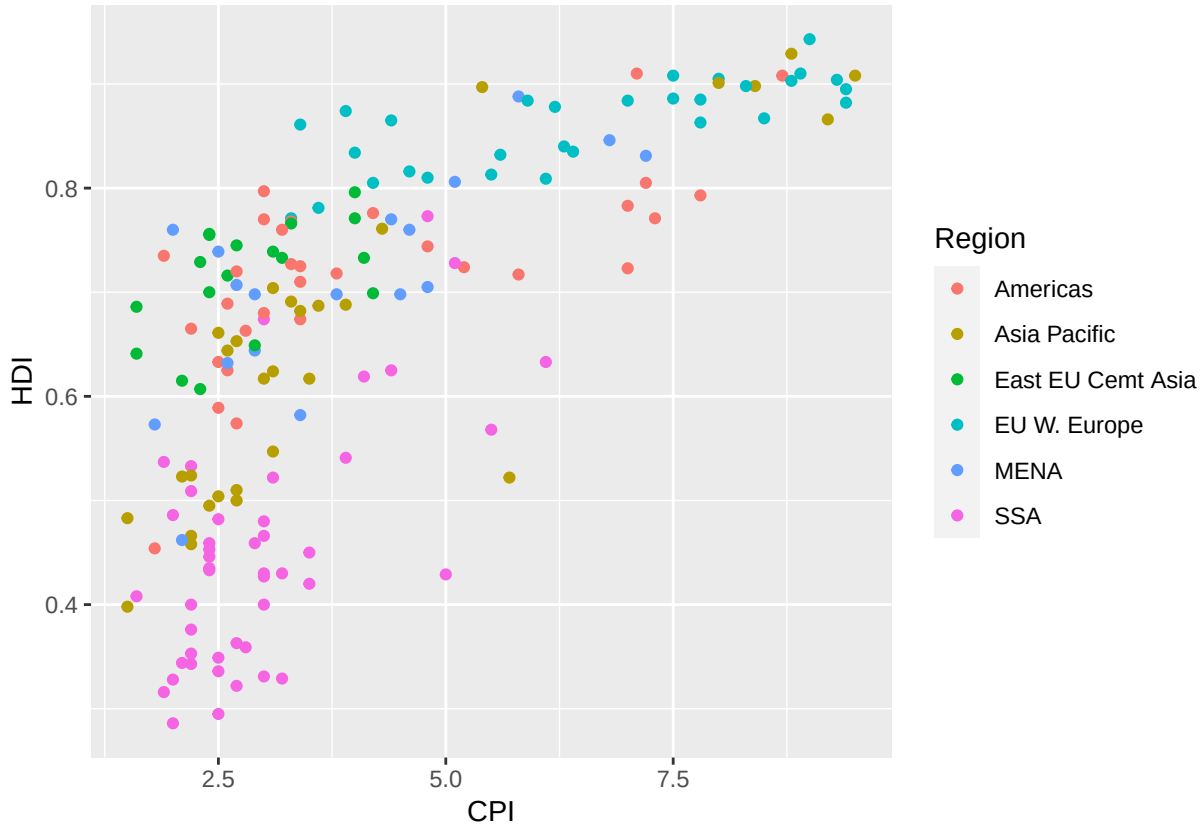
```
ggplot(econ_data, aes(x = CPI, y = HDI)) +
 geom_point(color = "blue")
```



Part D:

Color the points in the CPI vs HDI plot according to Region.

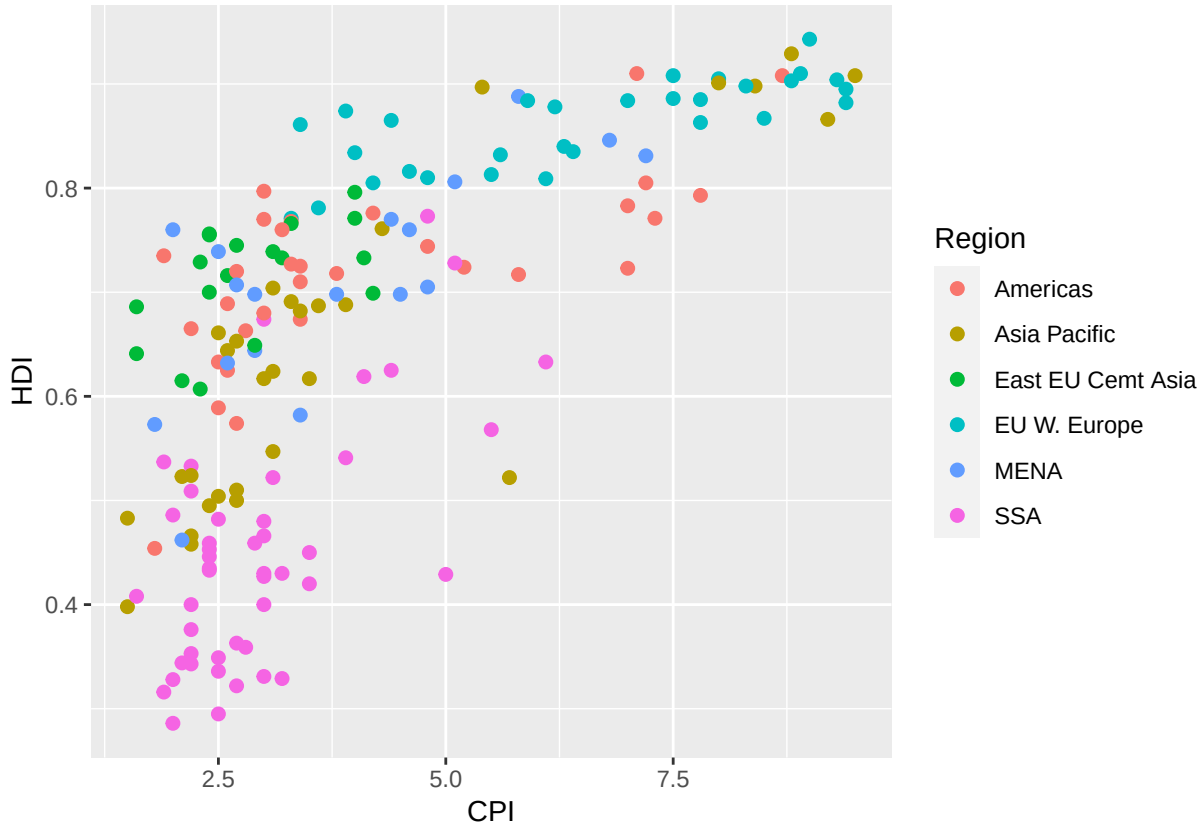
```
ggplot(econ_data, aes(x = CPI, y = HDI)) +
 geom_point(aes(color = Region))
```



Part E:

Make the points bigger by changing the 'size' option.

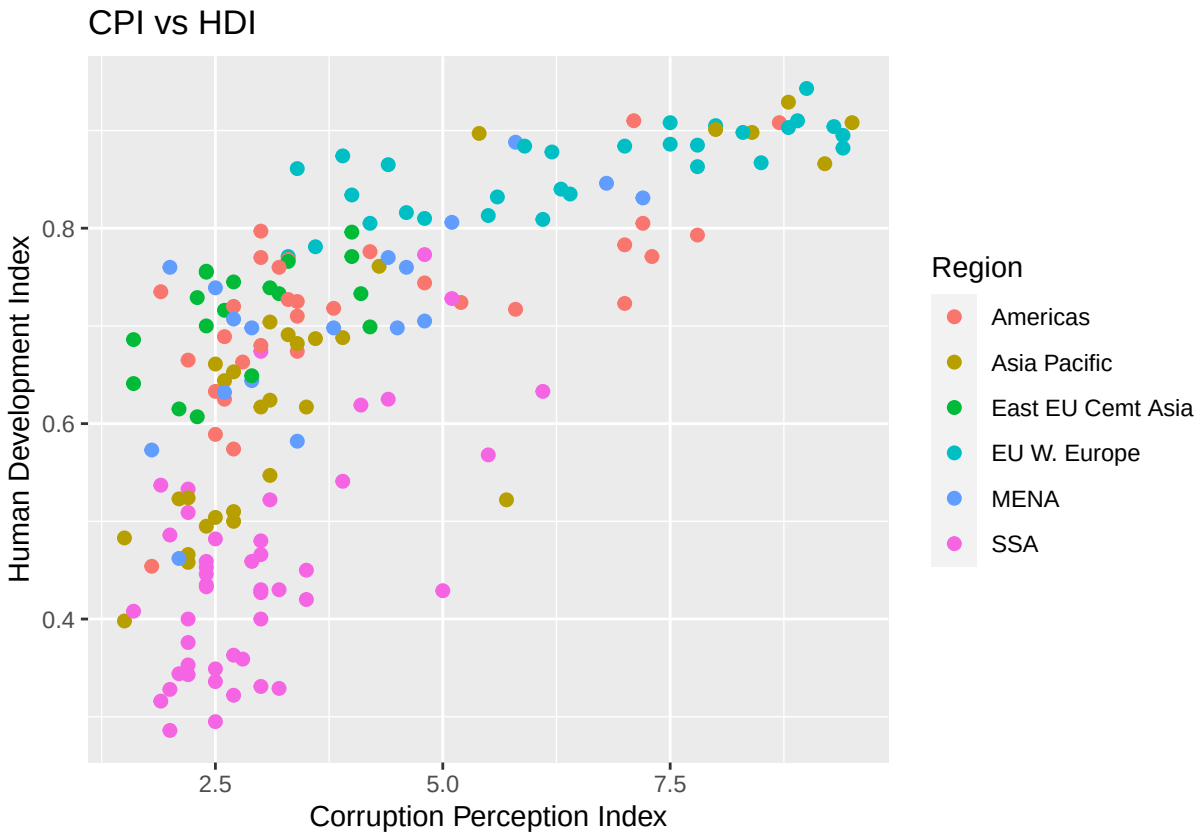
```
ggplot(econ_data, aes(x = CPI, y = HDI)) +
 geom_point(aes(color = Region), size = 2)
```



Part F:

Change axis labels as Corruption Perception Index, and Human Development Index. Also, add a proper title to previous scatter plot.

```
ggplot(econ_data, aes(x = CPI, y = HDI)) +
  geom_point(aes(color = Region), size = 2) +
  labs(x = "Corruption Perception Index",
       y = "Human Development Index",
       title = "CPI vs HDI")
```

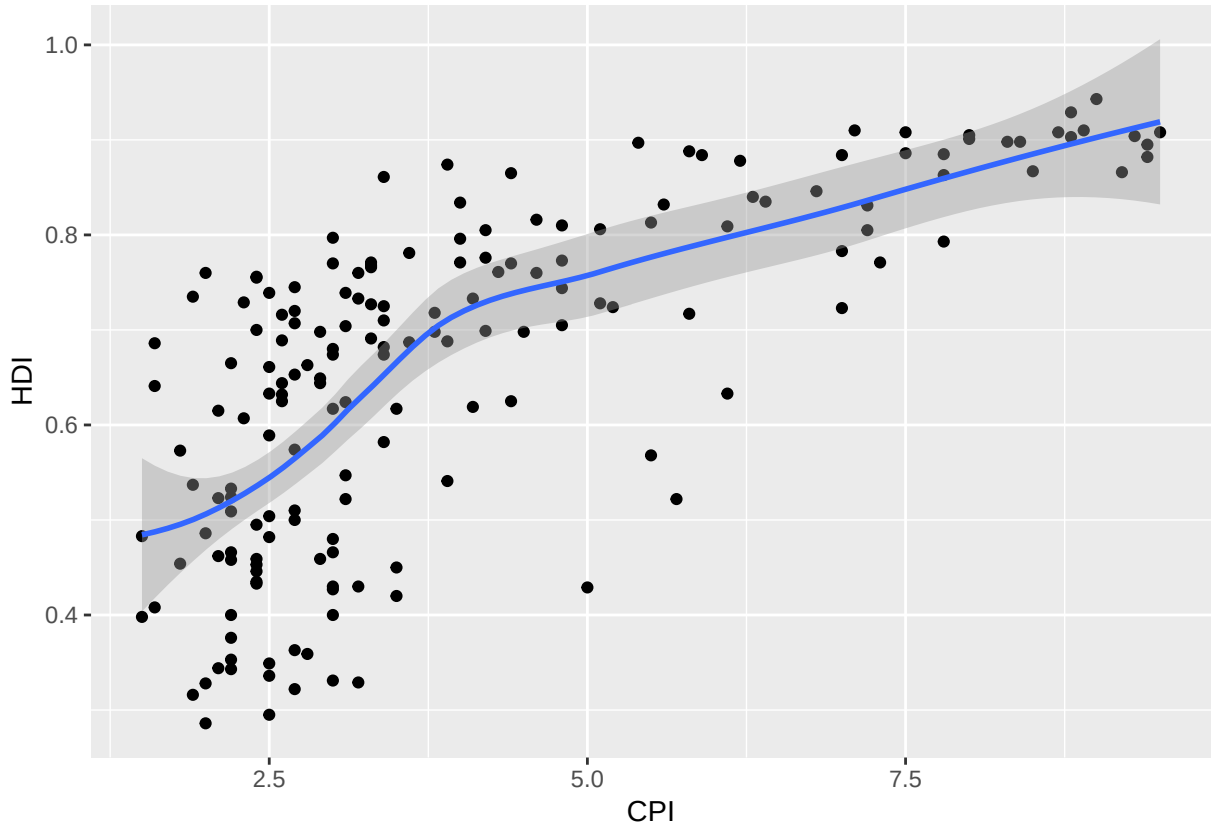


Part G:

Re-create a scatter plot with CPI on the x axis and HDI on the y axis and then overlay a smoothing line on top of the scatter plot using geom_smooth.

```
ggplot(econ_data, aes(x = CPI, y = HDI)) +
 geom_point() +
 geom_smooth()
```

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

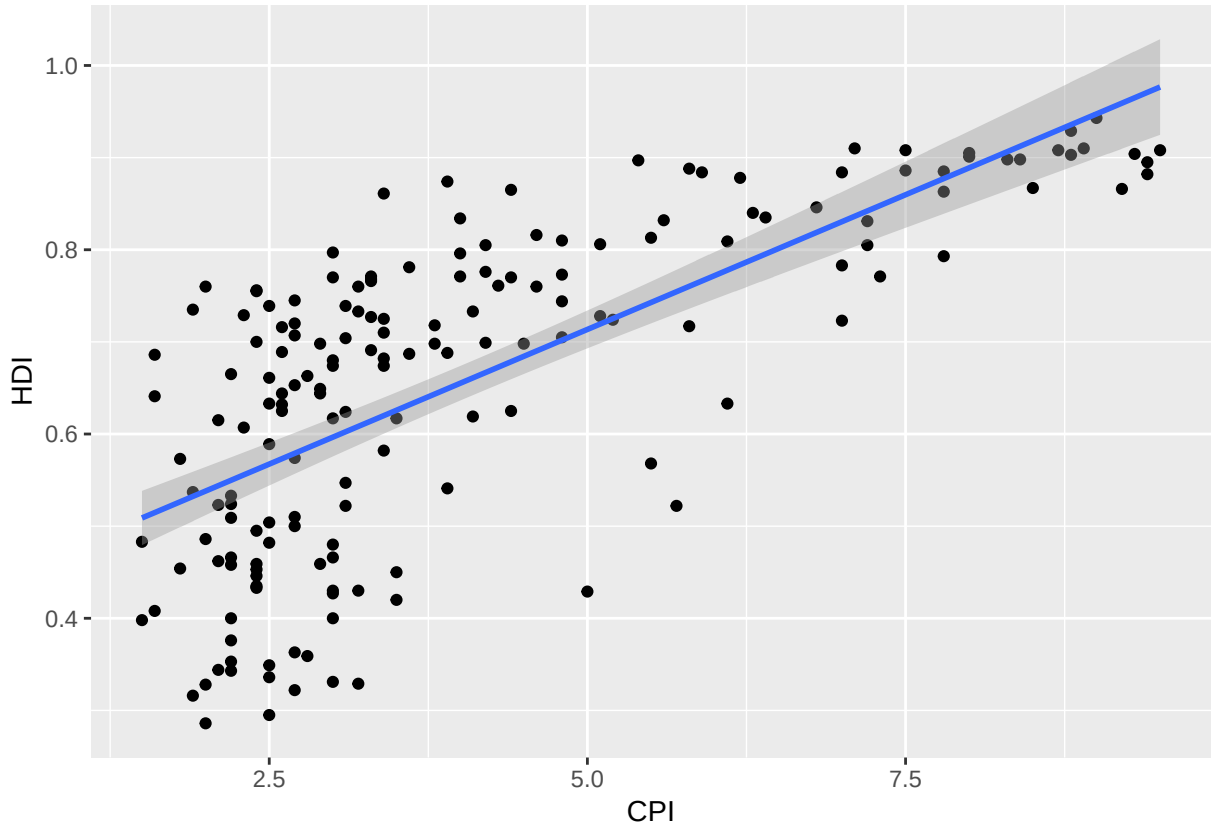


Part H:

Overlay a smoothing line on top of the scatter plot using geom_smooth, but use a linear model for the predictions. (**Hint: see ?stat_smooth**)

```
ggplot(econ_data, aes(x = CPI, y = HDI)) +
 geom_point() +
 geom_smooth(method = "lm")
```

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



Example 3:

Part A:

Load Auto data set in ISLR package, then drop 'name' variable. Convert cylinders and origin to factors.

```
library(ISLR)
data(Auto)
Auto$name <- NULL
Auto$origin <- factor(Auto$origin,
                      levels = 1:3,
                      labels=c("American","European","Japanese"))
Auto$cylinders <- factor(Auto$cylinders)
str(Auto)
```

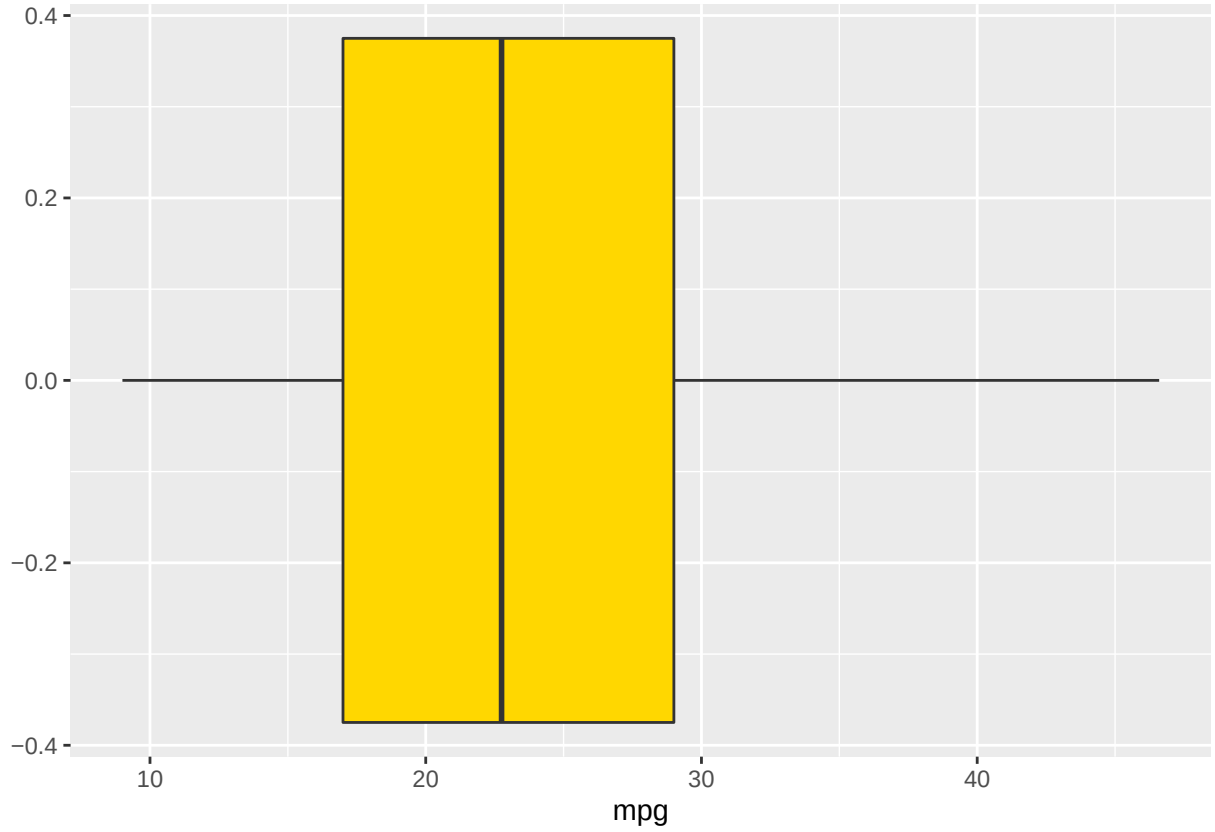
```
## 'data.frame':    392 obs. of  8 variables:
##  $ mpg         : num  18 15 18 16 17 15 14 14 14 15 ...
##  $ cylinders   : Factor w/ 5 levels "3","4","5","6",..: 5 5 5 5 5 5 5 5 5 5 ...
##  $ displacement: num  307 350 318 304 302 429 454 440 455 390 ...
##  $ horsepower  : num  130 165 150 150 140 198 220 215 225 190 ...
##  $ weight      : num  3504 3693 3436 3433 3449 ...
```

```
##  $ acceleration: num  12 11.5 11 12 10.5 10 9 8.5 10 8.5 ...
##  $ year        : num  70 70 70 70 70 70 70 70 70 70 ...
##  $ origin      : Factor w/ 3 levels "American","European",..: 1 1 1 1 1 1 1 1 1 1 ..
```

Part B:

Obtain a boxplot for MPG.

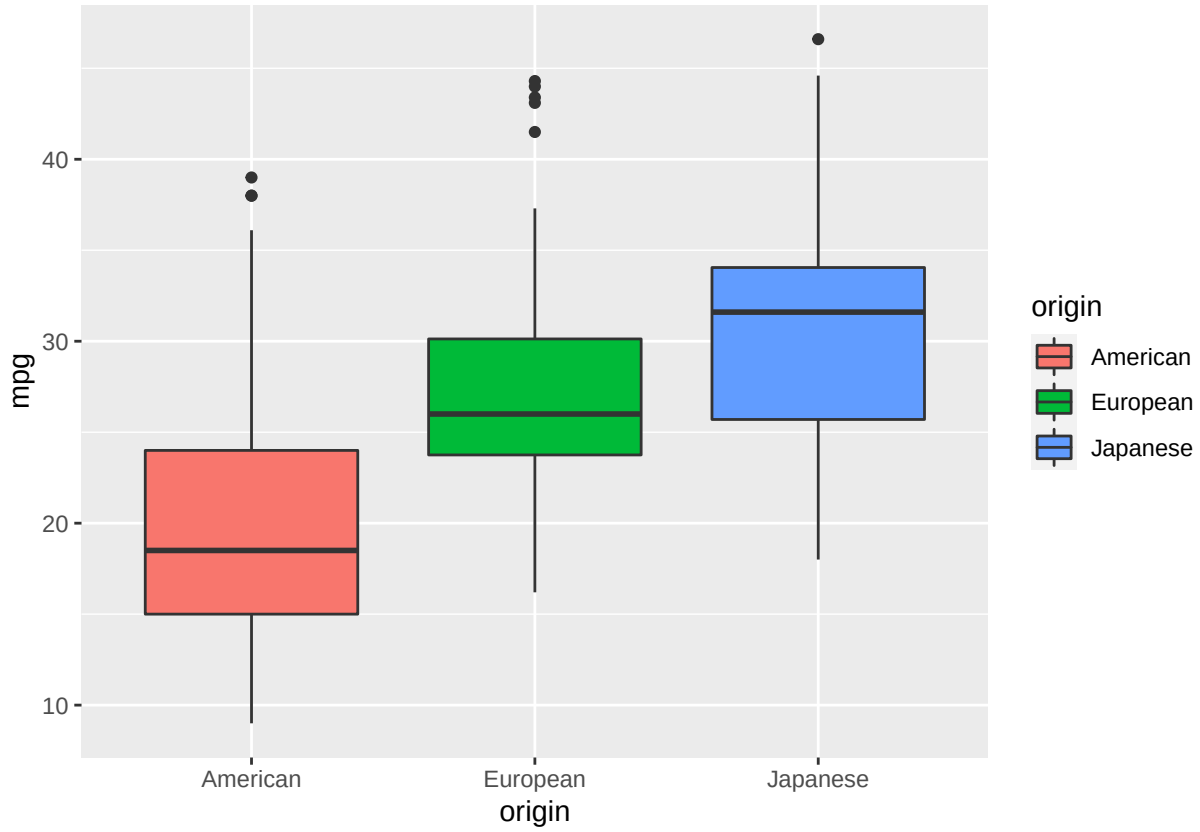
```
ggplot(Auto, aes(x = mpg)) +
  geom_boxplot(fill = "Gold")
```



Part C:

Now, obtain boxplots for MPG for each origin. First change labels of origin as follows; 1. American, 2. European, 3. Japanese.

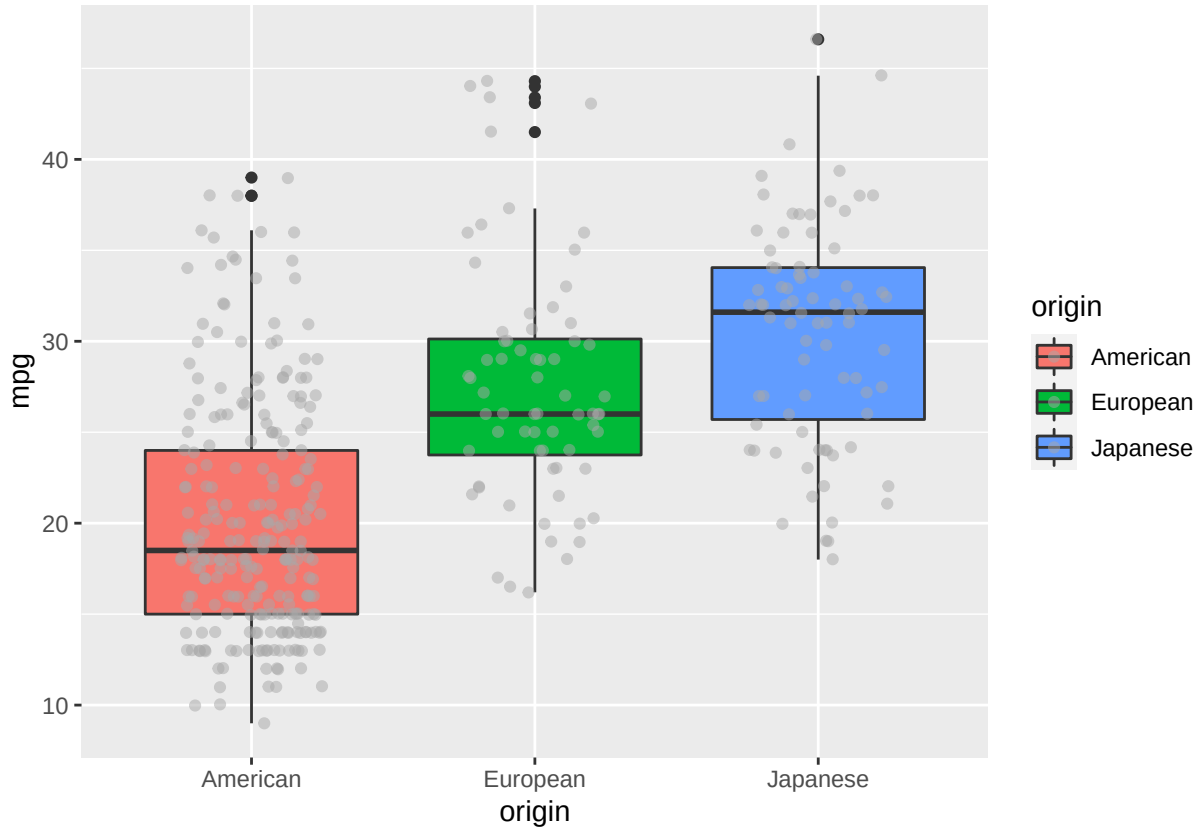
```
ggplot(Auto, aes(x = origin, y = mpg, fill = origin)) +
  geom_boxplot()
```



Part D:

Using geom_jitter, add jittered points to previous boxplot.

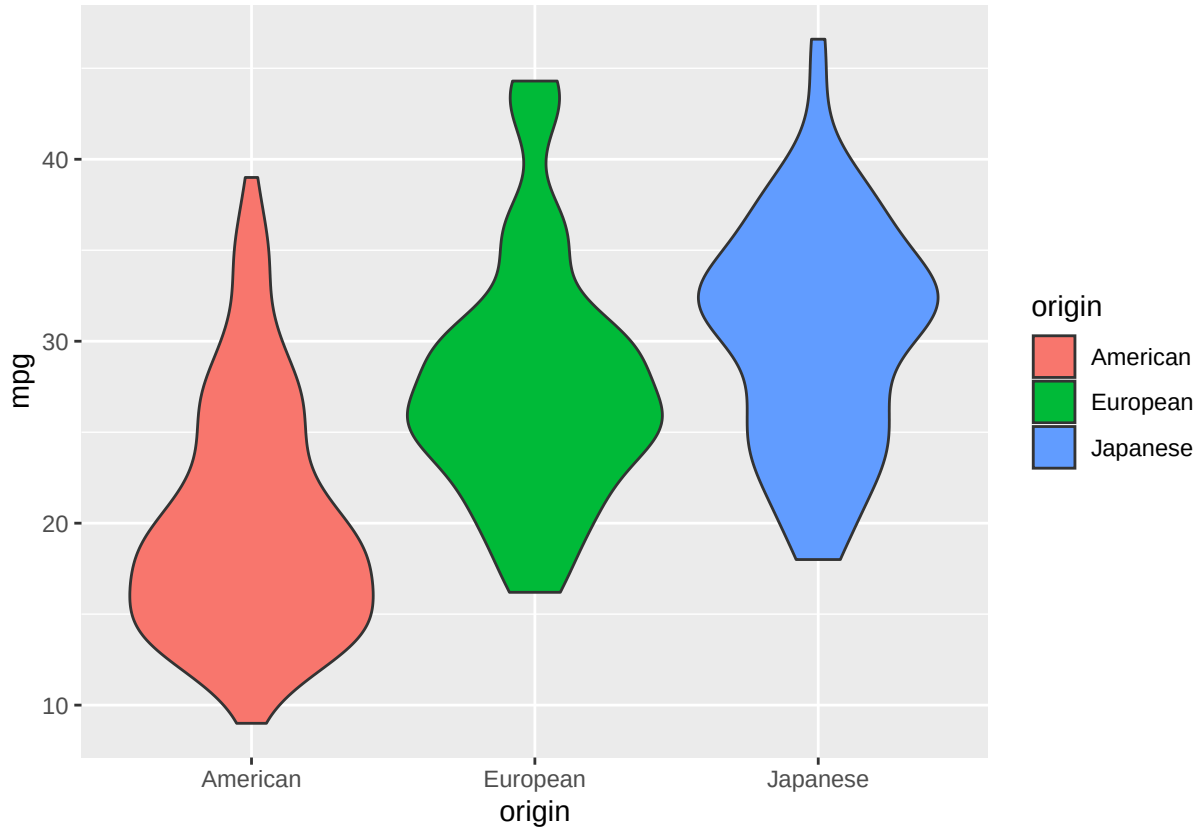
```
ggplot(Auto, aes(x = origin, y = mpg, fill = origin)) +
  geom_boxplot() +
  geom_jitter(alpha = 0.5, color = "Dark Grey", width = 0.25)
```



Part E:

For the same scenario, MPG vs Origin, obtain violin plot instead of boxplot.

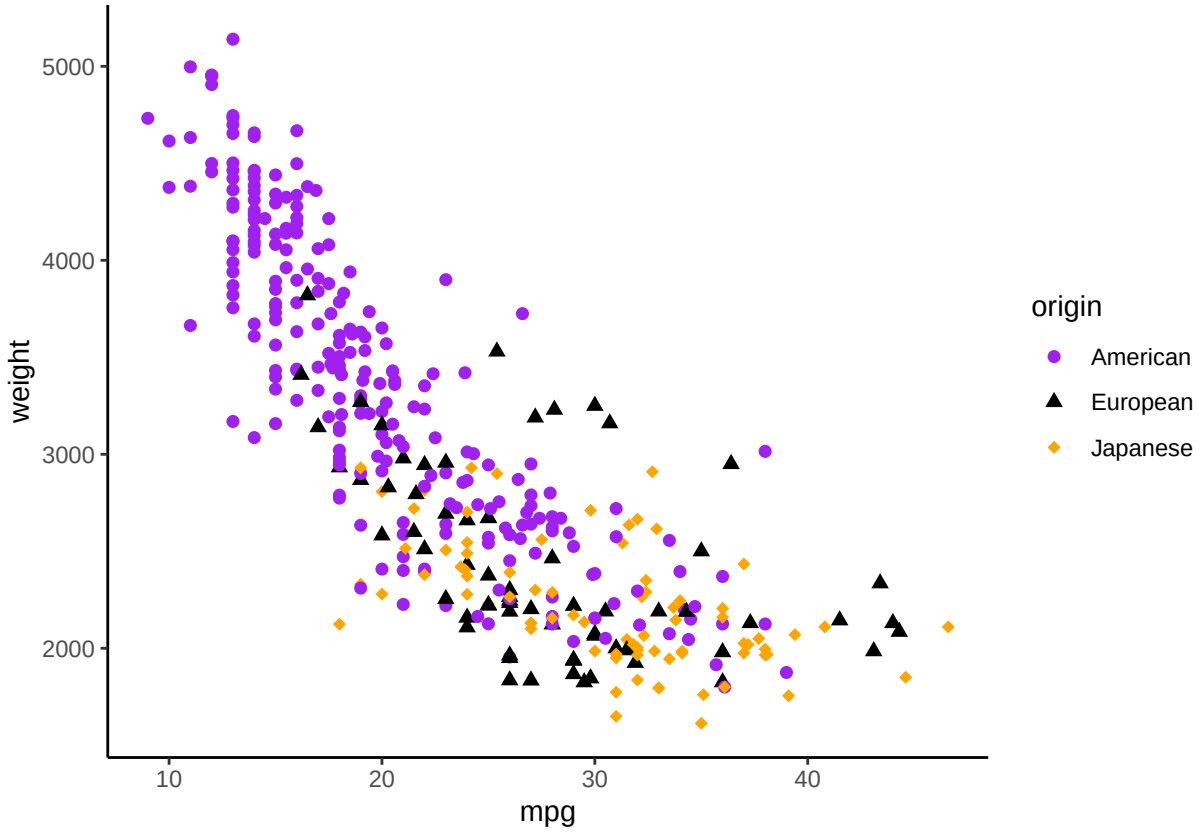
```
ggplot(Auto, aes(x = origin, y = mpg, fill = origin)) +
  geom_violin()
```



Part F:

Plot MPG vs. weight, also add another dimension by coloring points them by origin. Use different shapes for each level of origin.

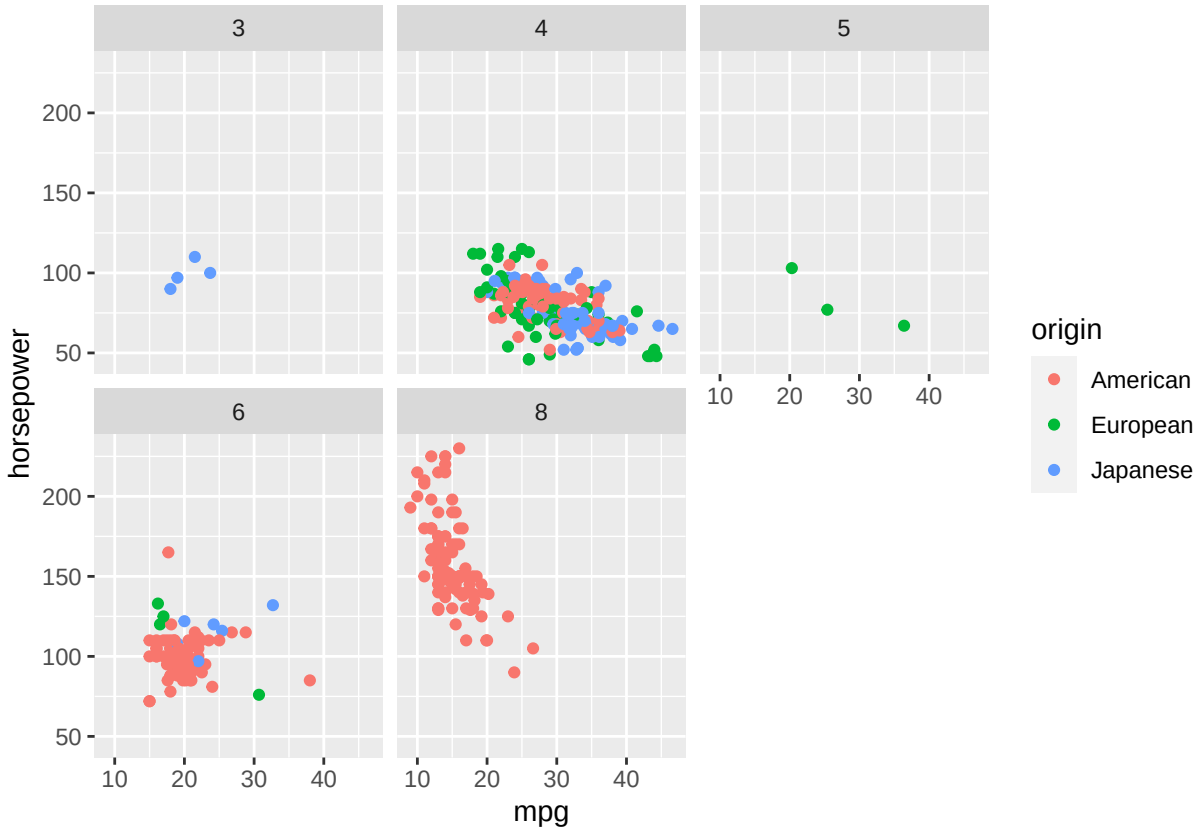
```
ggplot(data = Auto, aes(x = mpg, y = weight, color = origin)) +
  geom_point(aes(shape = origin), size = 2) +
  scale_shape_manual(values = c(16, 17, 18)) +
  scale_color_manual(values = c("purple",
                                "black",
                                "orange")) +
  theme_classic()
```



Part G:

Now, facet_wrap() function, obtain scatter plots mpg vs. horsepower for each subset of 'cylinders'. Also, add 'origin' factor in your scatter plots as third dimension. (Simply color the points by their origins.)

```
ggplot(data = Auto) +
  geom_point(mapping = aes(x = mpg, y = horsepower,
                           color = origin)) +
  facet_wrap(~ cylinders, nrow = 2)
```



Exercise 4:

Part A:

Load diamonds data set.

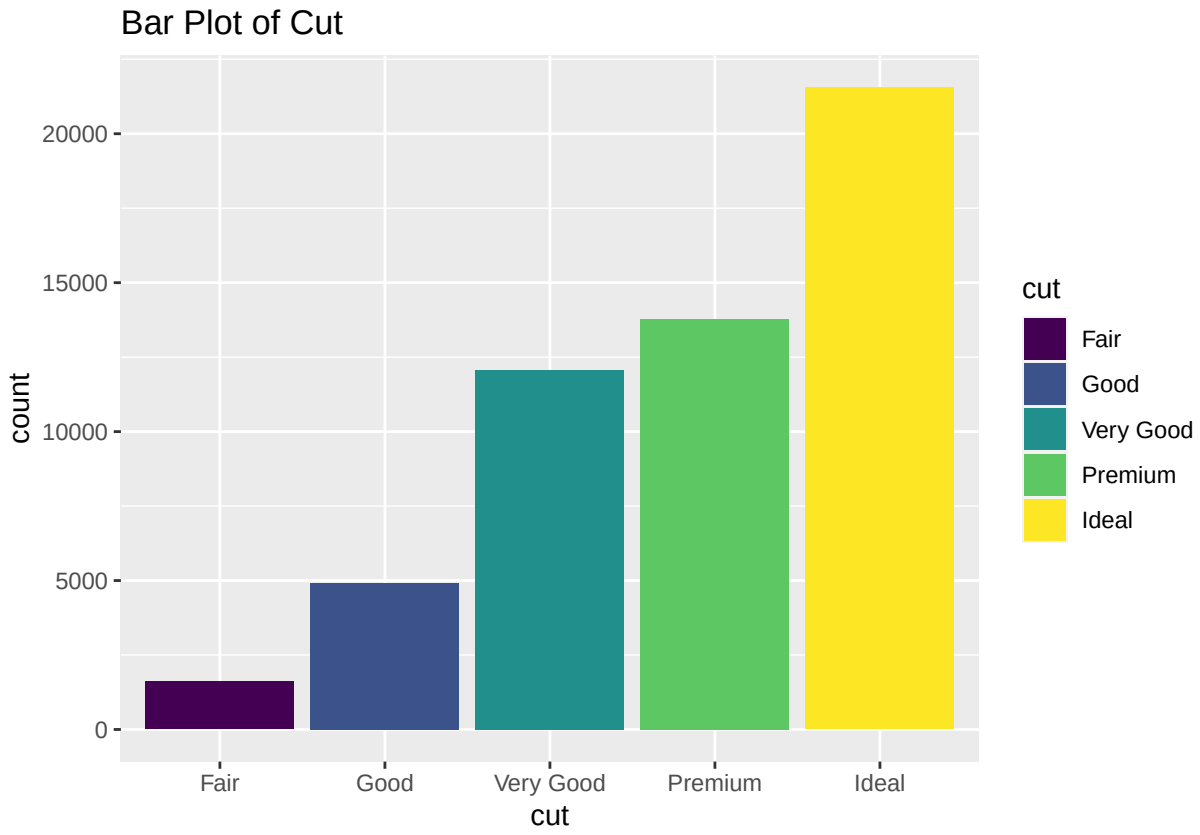
```
data(diamonds)
str(diamonds)
```

```
## tibble [53,940 x 10] (S3: tbl_df/tbl/data.frame)
##  $ carat  : num [1:53940] 0.23 0.21 0.23 0.29 0.31 0.24 0.24 0.26 0.22 0.23 ...
##  $ cut    : Ord.factor w/ 5 levels "Fair"<"Good"<..: 5 4 2 4 2 3 3 3 1 3 ...
##  $ color  : Ord.factor w/ 7 levels "D"<"E"<"F"<"G"<..: 2 2 2 6 7 7 6 5 2 5 ...
##  $ clarity: Ord.factor w/ 8 levels "I1"<"SI2"<"SI1"<..: 2 3 5 4 2 6 7 3 4 5 ...
##  $ depth  : num [1:53940] 61.5 59.8 56.9 62.4 63.3 62.8 62.3 61.9 65.1 59.4 ...
##  $ table  : num [1:53940] 55 61 65 58 58 57 57 55 61 61 ...
##  $ price  : int [1:53940] 326 326 327 334 335 336 336 337 337 338 ...
##  $ x      : num [1:53940] 3.95 3.89 4.05 4.2 4.34 3.94 3.95 4.07 3.87 4 ...
##  $ y      : num [1:53940] 3.98 3.84 4.07 4.23 4.35 3.96 3.98 4.11 3.78 4.05 ...
##  $ z      : num [1:53940] 2.43 2.31 2.31 2.63 2.75 2.48 2.47 2.53 2.49 2.39 ...
```

Part B:

Obtain a bar plot for 'cut'. Add a title, color each bar for each level of cut.

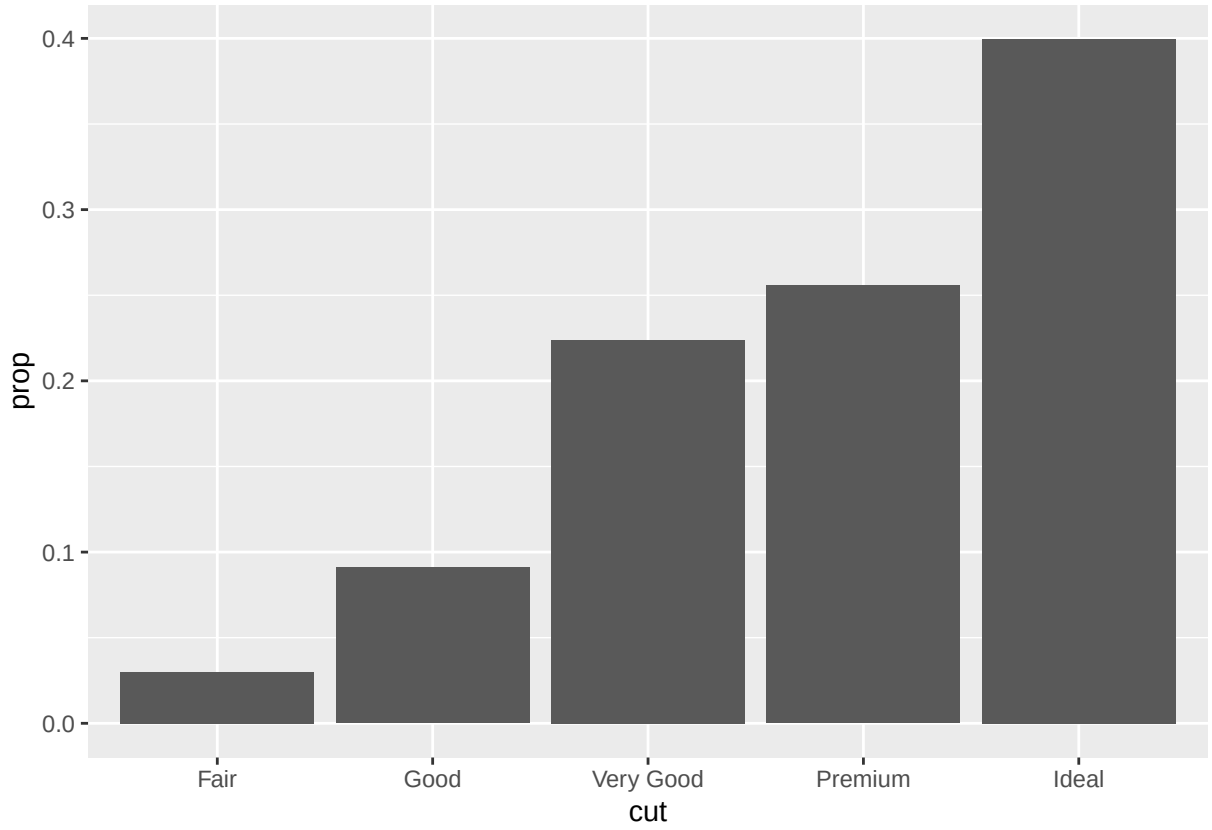
```
ggplot(data = diamonds) +
  geom_bar(mapping = aes(x = cut, fill = cut)) +
  labs(title = "Bar Plot of Cut")
```



Part B:

Obtain bar plot for proportions for each cut type.

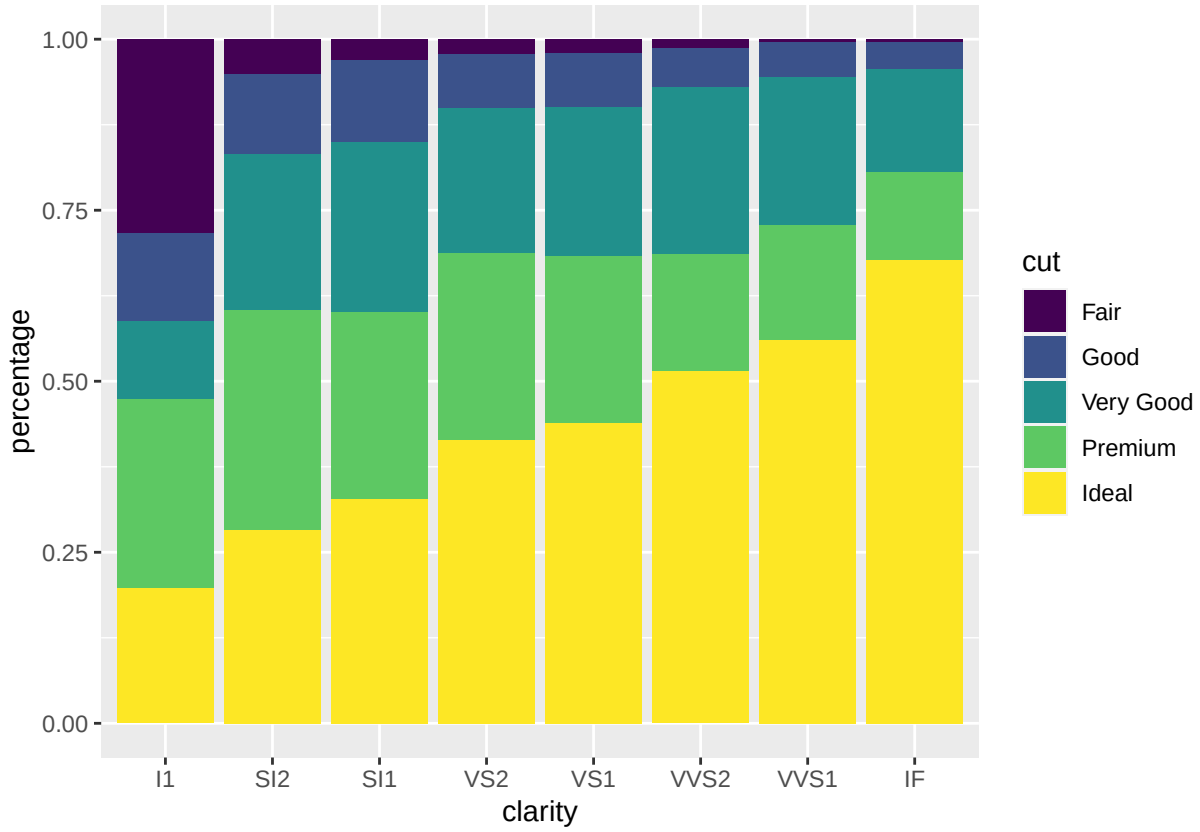
```
ggplot(data = diamonds) +
  geom_bar(mapping = aes(x = cut, y = stat(prop), group = 1))
```



Part C:

Obrtain a stacked bar plot for the percentages of clarity vs. cut.

```
ggplot(diamonds, aes(x = clarity)) +
  geom_bar(aes(fill = cut), position = 'fill') +
  labs(y="percentage")
```



Part D:

Save the previous plot in part c.

```
ggsave("MyCoolPlot.png")
```

```
## Saving 6.5 x 4.5 in image
```

Part E:

Obtain a histogram for price. Store this plot in a variable, partE.

```
partE <- ggplot(data = diamonds, aes(x = price)) +
  geom_histogram() +
  labs(title = "Histogram of Price, bins=30", x = "Price", y = "Count")
```

Part F:

Now, obtain a histogram for price with number of bins = 200, change the fill color, also store this plot in a variable, partF.

```
partF <- ggplot(data = diamonds, aes(x = price)) +
  geom_histogram(fill = "Gold", bins = 200) +
```

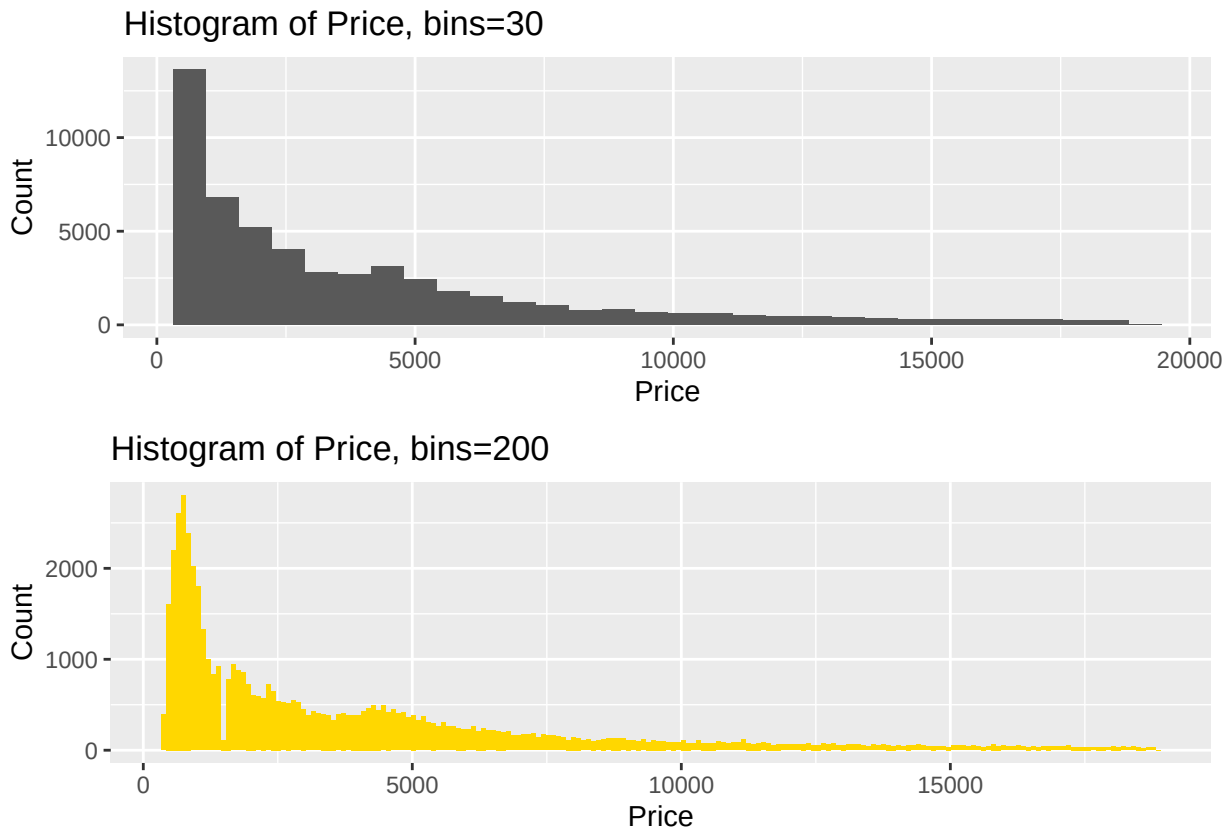
```
  labs(title = "Histogram of Price, bins=200", x = "Price", y = "Count")
```

Part G:

Plot partE and partF in a same plot, using grid.arange function in 'gridExtra' package.

```
gridExtra::grid.arrange(partE, partF, ncol = 1)
```

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



Exercise 5:

Part A:

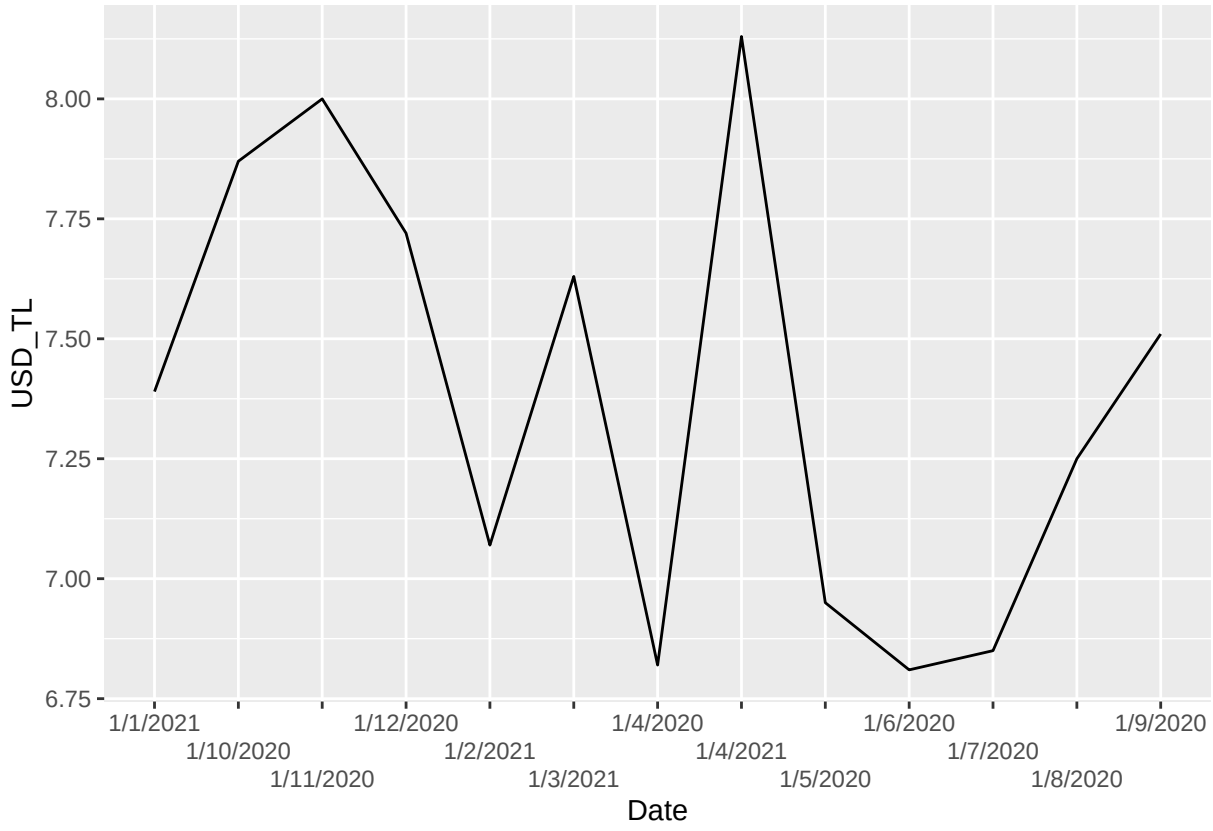
Load currency data set into R.

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

Part B:

Obtain a Line graph for USD - TL.

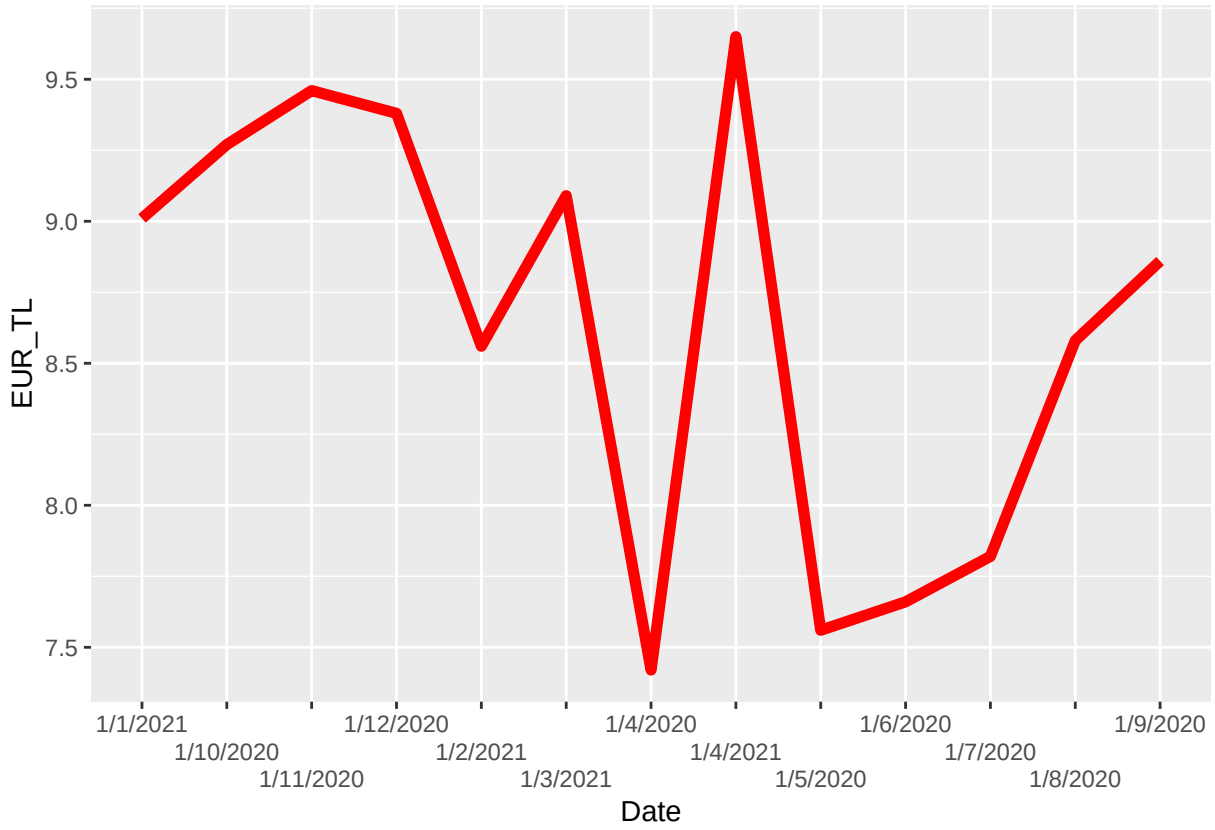
```
ggplot(data = currency, aes(x = Date, y = USD_TL, group = 1)) +
  geom_line() +
  scale_x_discrete(guide = guide_axis(n.dodge=3))
```



Part C:

Obtain a Line graph for EUR - TL. Color it Red and make it thicker.

```
ggplot(data = currency, aes(x = Date, y = EUR_TL, group = 1)) +
  geom_line(color = "Red", size = 2) +
  scale_x_discrete(guide = guide_axis(n.dodge=3))
```



Part D:

Draw 2 line plots for USD_TL and EUR_TL in a same plot. Store this plot in an object.

```
library(dplyr)
```

```
##
## 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
```

```
library(tidyr)
currency_new <- currency %>%
  select(Date, USD_TL, EUR_TL) %>%
  gather(key = "variable", value = "value", -Date)
```

```
# Hint: Create a dataframe looks like this
head(currency_new)
```

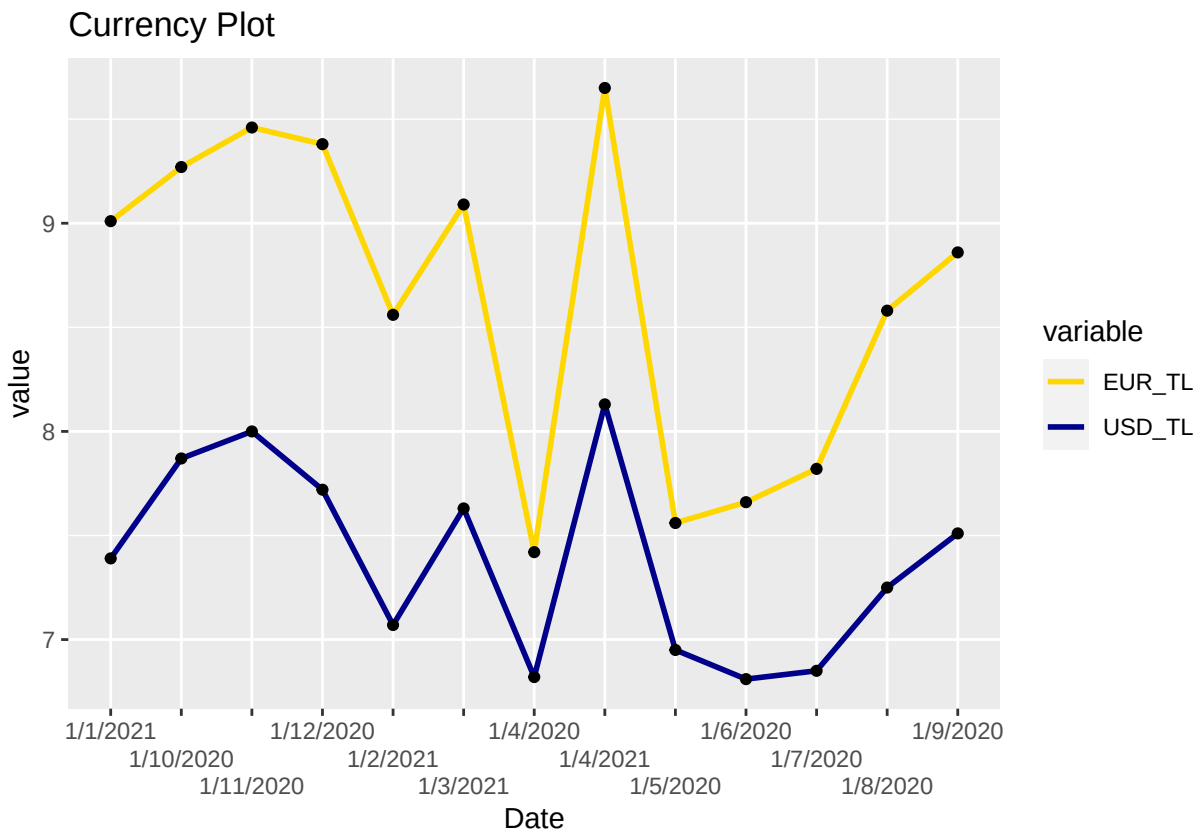
```
##       Date variable value
## 1 1/4/2020   USD_TL  6.82
## 2 1/5/2020   USD_TL  6.95
## 3 1/6/2020   USD_TL  6.81
## 4 1/7/2020   USD_TL  6.85
## 5 1/8/2020   USD_TL  7.25
## 6 1/9/2020   USD_TL  7.51
```

```
plot_currency <- ggplot(currency_new,
                        aes(x = Date, y = value, group=variable)) +
  geom_line(aes(color = variable), size = 1) +
  labs(title = "Currency Plot") +
  scale_color_manual(values = c("Gold", "Dark Blue")) +
  scale_x_discrete(guide = guide_axis(n.dodge=3))
```

Part E:

Add points on both lines for each date.

```
plot_currency + geom_point()
```

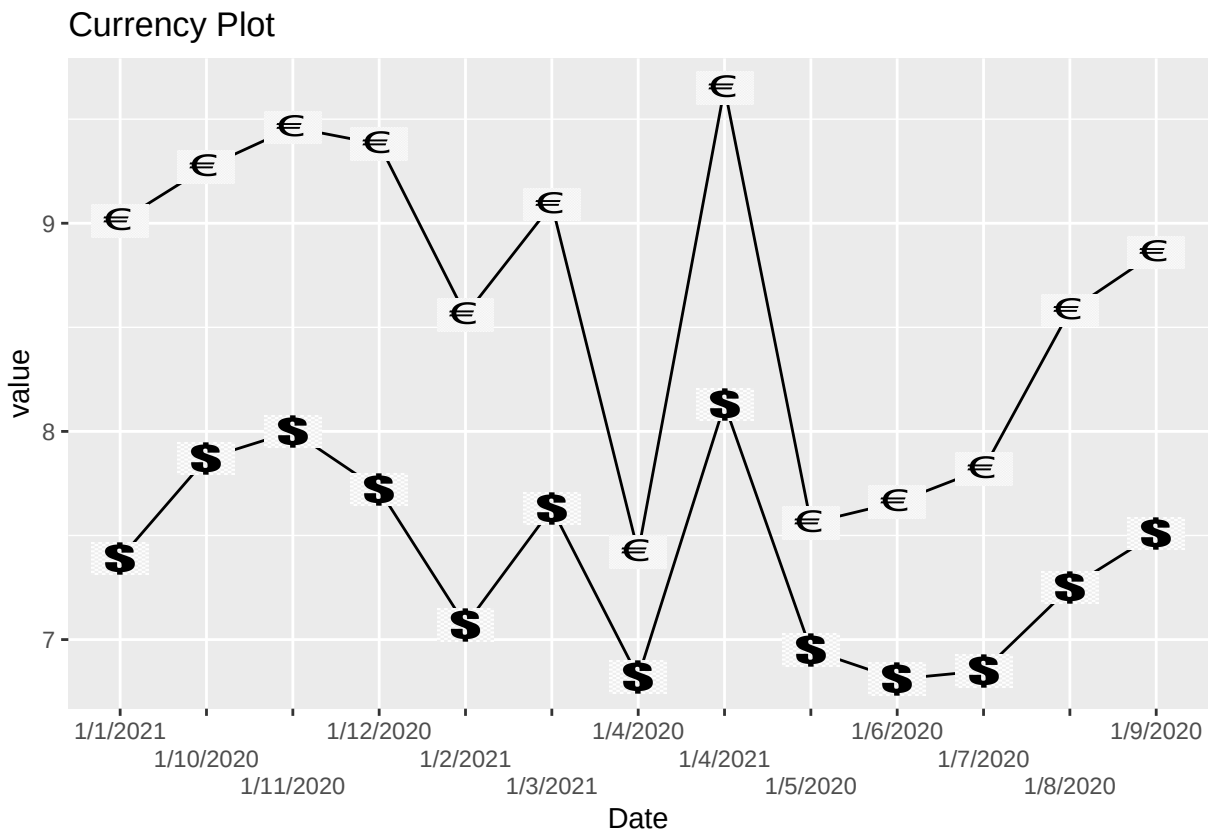


Part F:

Add dollar and euro signs on the plot. I used 'ggimage' package to achieve the following plot.

```
library("ggimage")
currency_new2 <- data.frame(currency_new,
                            image = rep(c("dollar.png","euro.png"),
                                        each = nrow(currency_new)/2))

ggplot(currency_new2,
       aes(Date, value, group = variable)) +
  geom_line() +
  geom_image(aes(image=image), size=.05) +
  labs(title = "Currency Plot") +
  scale_x_discrete(guide = guide_axis(n.dodge=3))
```



Bonus:

Use plot3d function in 'rgl' package, to create a 3D plot for mpg, horsepower and weight. Color them by origin.

```
library(rgl)
```

```
library(magrittr)
Auto %$% plot3d(mpg, horsepower, weight,
                type = "s", col = as.numeric(origin),
                size = 0.75)
legend3d("right",
         legend = c('American', 'European', 'Japanese'),
         pch = 16, col = 1:3)
```