Computer Assignment 2 - Exploratory Analysis

Machine Learning, Spring 2020

Rui Li

The purpose of this assignment is to walk you through a typical starting analysis of real-world data in `R`. We will also introduce how to draw from various probability distributions using built-in `R` functions.

Input/Output

`scan` and `read.table` are the two main functions in `R` for reading data. The main difference between them lies in that `scan` reads one component (also called "field") at a time while `read.table` reads one line at a time. Thus, `read.table` requires the data to have a table structure and will create a data.frame in **R** automatically, while `scan` can be flexible but might require effort in manipulating data after reading. Overall, their usages are quite similar. One should pay attention to the frequently used options `file`, `header`, `sep`, `dec`, `skip`, `nmax`, `nlines` and `nrows` in their **R** documents.

`write.table` is the converse of `read.table`. While the latter reads data into `R` from a file, the former writes data to a file from `R`.

Note: If, in the future, you have data stored in another format, *e.g.* EXCEL or SAS dataset, then you can output it as a CSV file and read it into **R** via the `read.csv` function (which is almost identical to `read.table`).

Questions

In the folder for HW 1, you can find data on the 1974 Motor Trend US Magazine on a series of road tests they did.

1. Read the data using read.csv into a data frame called `mtcars_dat`. This dataset is actually available in base `R`, but for practice use the read.csv() function.

Important: When reading files into R that are in local directories, you'll need to set your working directory to the place where the file is located. Do this by going `Session->Set Working Directory->Choose Directory...` and choosing the file in which the data is located.

Important as well: You have just been given the specific functions you will need to solve this exercise. Don't know how to use them? Google them! Or check the manual page for them by inputting `?read.table()`! Coding in general requires this sort of independence and tenacity.

```
mtcars_dat = read.csv("mtcars.csv")
```

Use `str(mtcars_dat)` and `head(mtcars_dat)` observe the dimensions and structure of the dataset. Comment on what you see. How many different variables are in this dataset? What are the row names?

```r
str(mtcars_dat)
```

```
## 'data.frame':    32 obs. of  12 variables:
##  $ X   : Factor w/ 32 levels "AMC Javelin",..: 18 19 5 13 14 31 7 21 20 22
 ...
##  $ mpg : num  21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...
##  $ cyl : int  6 6 4 6 8 6 8 4 4 6 ...
##  $ disp: num  160 160 108 258 360 ...
##  $ hp  : int  110 110 93 110 175 105 245 62 95 123 ...
##  $ drat: num  3.9 3.9 3.85 3.08 3.15 2.76 3.21 3.69 3.92 3.92 ...
##  $ wt  : num  2.62 2.88 2.32 3.21 3.44 ...
##  $ qsec: num  16.5 17 18.6 19.4 17 ...
##  $ vs  : int  0 0 1 1 0 1 0 1 1 1 ...
##  $ am  : int  1 1 1 0 0 0 0 0 0 0 ...
##  $ gear: int  4 4 4 3 3 3 3 4 4 4 ...
##  $ carb: int  4 4 1 1 2 1 4 2 2 4 ...
```

```r
head(mtcars_dat)
```

```
##                   X  mpg cyl disp  hp drat    wt  qsec vs am gear carb
## 1         Mazda RX4 21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
## 2     Mazda RX4 Wag 21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
## 3        Datsun 710 22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
## 4    Hornet 4 Drive 21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
## 5 Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
## 6           Valiant 18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
```

```
The `str(mtcars_dat)` compactly displays the internal structure of the datase
t `mtcars.csv`, and the `head(mtcars_dat)` displays the first six rows of the
 dataset.

There are 12 variables in this dataset, including the row names, and the name
s of the row are different type of tested motors.
```

2. Make a new data frame called `relevant` consisting only of the columns: X, mpg, cyl, disp, hp (Hint: consider the `subset` function).

```r
relevant = subset(mtcars_dat, select = c(X,mpg,cyl,disp,hp))
```

3. Make a new data frame called `top_hp` consisting of cars in `relevant` that had greater than or equal to 150 horsepower.

```r
top_hp = subset(relevant,hp>=150)
```

4. Split the data into two groups called `top_hp` and `bottom_hp`, where the former has all observations with greater than or equal to 150 horsepower, and the latter has all observations with less than 150 horsepower. Now compute the average mpg in each group.

```r
#Split the data into two groups
top_hp = subset(relevant,hp>=150)
bottom_hp = subset(relevant,hp<150)
```

```
#Caculate the average mpg in each group
average_top_mpg = mean(top_hp$mpg)
average_bottom_mpg = mean(bottom_hp$mpg)
```

Data Exploration and Manipulation

Data analysis in **R** starts with reading data into a data.frame object via `scan` and `read.table` as discussed before. The following step is to explore the profiles of data via various descriptive statistics whose usages are also introduced in the previous sections. Calling the `summary` function with a data.frame input also provides appropriate summaries, *e.g.* means and quantiles for numeric variables and frequencies for factor variables.

What is often the next step is to visualize the data.

Plots

Compared to other statistical softwares in data analysis, **R** is very good at graphic generation and manipulation. The plotting functions in **R** can be classified into the high-level ones and low-level ones.

`plot` is the most generic high-level plotting function in **R**. It will be compatible with most classes of objects that the user uses and will produce appropriate graphics. For example, if one had a numeric vector `x` and imputed it into the plot function (`plot(x)`) this would result in a scatter plot. Advanced classes of objects like `lm` (fitted result by a linear model) can also be called in `plot`. Sometimes the best preliminary thing to try when you have any class of data object `x` is `plot(x)`.

Other plotting features include

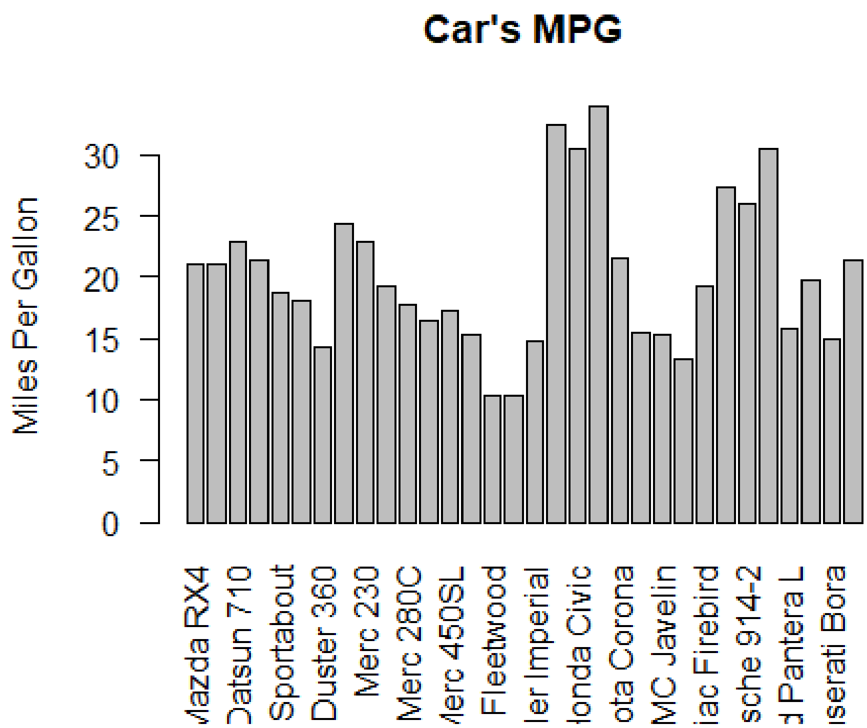
- High-level plotting options: `type`, `main`, `sub`, `xlab`, `ylab`, `xlim`, `ylim`
- Low-level plotting functions
  - **Symbols:** `points`, `lines`, `text`, `abline`, `segments`, `arrows`, `rect`, `polygon`
  - **Decorations:** `title`, `legend`, `axis`
- Environmental graphic options (`?par`)
  - **Symbols and texts:** `pch`, `cex`, `col`, `font`
  - **Lines:** `lty`, `lwd`
  - **Axes:** `tck`, `tcl`, `xaxt`, `yaxt`
  - **Windows:** `mfcol`, `mfrow`, `mar`, `new`
- User interaction: `location`

Beginners should learn from examples and grab whatever necessary plot when needed instead of going over such an overwhelming brochure. The following sections illustrate two basic scenarios in data analysis. More high-level plotting functions will also be introduced.

Questions

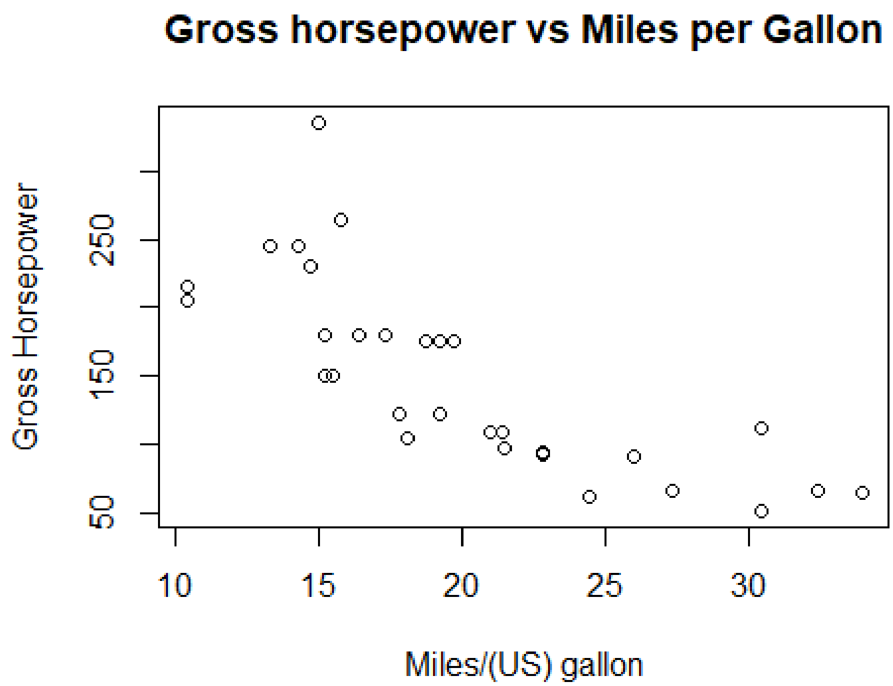
1. Recall the `mtcars` data set. From the `mtcars_dat` data frame plot a bar chart of each car's mpg. The x-axis here should be the name of each car `X` and the y-axis should `mpg`.

```
barplot(mtcars_dat$mpg, names.arg = mtcars_dat$X, angle = 90, las=2, main = "
Car's MPG", ylab = "Miles Per Gallon")
```



2. Now plot a scatterplot of `mpg` vs. `hp`. Find the correlation coefficient between these two variables. Does it reflect what you see on the plot?

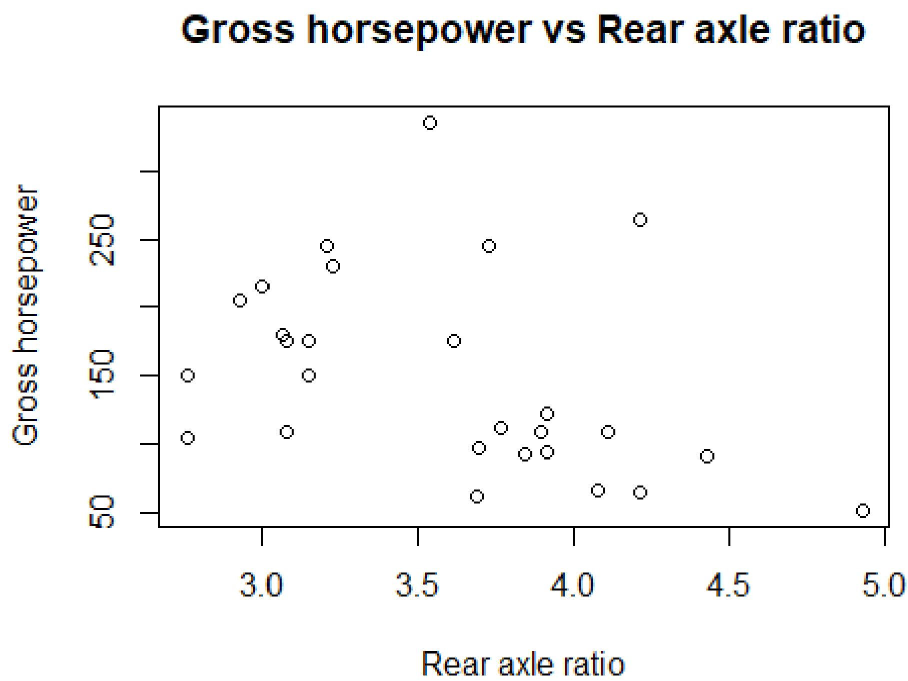
```
plot(x = mtcars_dat$mpg, y = mtcars_dat$hp, xlab = "Miles/(US) gallon", ylab
= "Gross Horsepower", main = "Gross horsepower vs Miles per Gallon")
```




```
cor(mtcars_dat$mpg,mtcars_dat$hp)

## [1] -0.7761684

The correlation coefficient between these two variables is r=-0.78, showing t
hat they are generally negatively correlated, which can be seen on the plot a
s well.
```

3. Plot a scatterplot of drat and hp. Do you observe any linear trend?

```
plot(x = mtcars_dat$drat, y = mtcars_dat$hp, xlab = "Rear axle ratio", ylab =
 "Gross horsepower", main = "Gross horsepower vs Rear axle ratio")
```




```
cor(mtcars_dat$drat,mtcars_dat$hp)

## [1] -0.4487591

From the plot, there is no clear linear trend between the two variables. This
 result can be proved by the correlation calculation as well, where r=-0.45 s
howing that they have a loosely relation.
```

Create advanced plots

If you are a JAVA programmer, then you might anticipate a plotting toolbox to establish graphs layer-by-layer interactively. The `ggplot2` package endows **R** with more advanced and powerful visualization techniques like this. Explore more in the online package manual.

```
mu <- mean(iris$Sepal.Length)
sigma <- sd(iris$Sepal.Length)
ggplot(iris, aes(Sepal.Length)) +
  geom_histogram( aes(y = ..density..),
                  bins = 8, color = "black", fill = "white") +
  geom_density(aes(color = "blue")) +
  stat_function( aes(color = "red"),
                 fun = dnorm, args = list(mean = mu, sd = sigma)) +
  labs(title = "Histogram of Sepal_Length") +
  scale_color_identity( name   = "Density Estimate",
                        guide  = "legend",
                        labels = c("Kernel", "Normal")) +
  theme_bw()
```

Further Analysis and Practice

Let's again load the `mtcars` dataset into R. Again, this dataset is actually available in base R, but to practice the input/output features in R we'll load it from a CSV (Comma Separated File). Make sure the file "mtcars.csv" is in the same directory as this Markdown file, set the working directory, and run the following command.

```
# We will first read in the data using the read.table() command,
my.dat = read.table("mtcars.csv", sep = ",", header = TRUE)
# then do just a bit of cleaning up:
row.names(my.dat) = my.dat$X
my.dat = my.dat[,-1]
# It may have been easier here to use the read.csv() function.  Check out it'
s manual page ?read.csv()
# to figure out why.
```

The *read.table()* function can be used to import, or read, data from pre-saved files including .txt, .csv, .xlsx or files available online. Similarly, the *write.table()* function can be used to write a saved **R** variable to a .txt, .csv, or .xlsx file.

Questions

1. Remember that once we read in a dataset with **R**, the value will be a data.frame type. Try to change our data.frame into a matrix type by using, for example, the *dat.1 = as.matrix(dat.1)* command.

```
as.matrix(my.dat)

##                      mpg cyl  disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4           21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag       21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710          22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive      21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout   18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
## Valiant             18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1
## Duster 360          14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4
## Merc 240D           24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2
## Merc 230            22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2
## Merc 280            19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4
## Merc 280C           17.8   6 167.6 123 3.92 3.440 18.90  1  0    4    4
## Merc 450SE          16.4   8 275.8 180 3.07 4.070 17.40  0  0    3    3
## Merc 450SL          17.3   8 275.8 180 3.07 3.730 17.60  0  0    3    3
## Merc 450SLC         15.2   8 275.8 180 3.07 3.780 18.00  0  0    3    3
## Cadillac Fleetwood  10.4   8 472.0 205 2.93 5.250 17.98  0  0    3    4
## Lincoln Continental 10.4   8 460.0 215 3.00 5.424 17.82  0  0    3    4
## Chrysler Imperial   14.7   8 440.0 230 3.23 5.345 17.42  0  0    3    4
## Fiat 128            32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1
## Honda Civic         30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2
## Toyota Corolla      33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1
## Toyota Corona       21.5   4 120.1  97 3.70 2.465 20.01  1  0    3    1
## Dodge Challenger    15.5   8 318.0 150 2.76 3.520 16.87  0  0    3    2
```

```
## AMC Javelin          15.2   8 304.0 150 3.15 3.435 17.30  0  0    3    2
## Camaro Z28           13.3   8 350.0 245 3.73 3.840 15.41  0  0    3    4
## Pontiac Firebird     19.2   8 400.0 175 3.08 3.845 17.05  0  0    3    2
## Fiat X1-9            27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1
## Porsche 914-2        26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
## Lotus Europa         30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2
## Ford Pantera L       15.8   8 351.0 264 4.22 3.170 14.50  0  1    5    4
## Ferrari Dino         19.7   6 145.0 175 3.62 2.770 15.50  0  1    5    6
## Maserati Bora        15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8
## Volvo 142E           21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2
```

2. Calculate the standard deviation and the 5-number summary for each of the columns using the *summary()* and *sd()*. Also, estimate the coefficient of variation ($c_v = \sigma/\mu$) for each of the columns of our dataset.

```
#Standard deviation of each column
for (i in 1:ncol(my.dat)) {
  print(paste("The standard deviation of column ",i, " is ",sd(my.dat[,i])))
}

## [1] "The standard deviation of column  1  is  6.0269480520891"
## [1] "The standard deviation of column  2  is  1.78592164694654"
## [1] "The standard deviation of column  3  is  123.938693831382"
## [1] "The standard deviation of column  4  is  68.5628684893206"
## [1] "The standard deviation of column  5  is  0.534678736070971"
## [1] "The standard deviation of column  6  is  0.978457442989697"
## [1] "The standard deviation of column  7  is  1.78694323609684"
## [1] "The standard deviation of column  8  is  0.504016128774185"
## [1] "The standard deviation of column  9  is  0.498990917235846"
## [1] "The standard deviation of column  10  is  0.737804065256947"
## [1] "The standard deviation of column  11  is  1.61519997763185"

#5-number Summary for each column
summary(my.dat)

##       mpg             cyl             disp             hp       
##  Min.   :10.40   Min.   :4.000   Min.   : 71.1   Min.   : 52.0  
##  1st Qu.:15.43   1st Qu.:4.000   1st Qu.:120.8   1st Qu.: 96.5  
##  Median :19.20   Median :6.000   Median :196.3   Median :123.0  
##  Mean   :20.09   Mean   :6.188   Mean   :230.7   Mean   :146.7  
##  3rd Qu.:22.80   3rd Qu.:8.000   3rd Qu.:326.0   3rd Qu.:180.0  
##  Max.   :33.90   Max.   :8.000   Max.   :472.0   Max.   :335.0  
##       drat             wt             qsec             vs        
##  Min.   :2.760   Min.   :1.513   Min.   :14.50   Min.   :0.0000  
##  1st Qu.:3.080   1st Qu.:2.581   1st Qu.:16.89   1st Qu.:0.0000  
##  Median :3.695   Median :3.325   Median :17.71   Median :0.0000  
##  Mean   :3.597   Mean   :3.217   Mean   :17.85   Mean   :0.4375  
##  3rd Qu.:3.920   3rd Qu.:3.610   3rd Qu.:18.90   3rd Qu.:1.0000  
##  Max.   :4.930   Max.   :5.424   Max.   :22.90   Max.   :1.0000  
##        am              gear            carb      
##  Min.   :0.0000   Min.   :3.000   Min.   :1.000  
```

```
##  1st Qu.:0.0000   1st Qu.:3.000   1st Qu.:2.000  
##  Median :0.0000   Median :4.000   Median :2.000  
##  Mean   :0.4062   Mean   :3.688   Mean   :2.812  
##  3rd Qu.:1.0000   3rd Qu.:4.000   3rd Qu.:4.000  
##  Max.   :1.0000   Max.   :5.000   Max.   :8.000
```

```
#Coefficient of variance
for (i in 1:ncol(my.dat)) {
  print(paste("The coefficient of variance of column ",i, " is ", sd(my.dat[,
i])/mean(my.dat[,i])))
}
```

```
## [1] "The coefficient of variance of column  1  is  0.299988081609661"
## [1] "The coefficient of variance of column  2  is  0.288633801526714"
## [1] "The coefficient of variance of column  3  is  0.537177906652249"
## [1] "The coefficient of variance of column  4  is  0.467407710195624"
## [1] "The coefficient of variance of column  5  is  0.148663824435408"
## [1] "The coefficient of variance of column  6  is  0.304128508194793"
## [1] "The coefficient of variance of column  7  is  0.100115875682994"
## [1] "The coefficient of variance of column  8  is  1.15203686576957"
## [1] "The coefficient of variance of column  9  is  1.22828533473439"
## [1] "The coefficient of variance of column  10  is  0.200082458374765"
## [1] "The coefficient of variance of column  11  is  0.574293325380214"
```

3. For columns 1, 3, and 6, find the following information:

- Whether the data is integer or non-integer valued
- Mean
- Median
- Standard deviation
- Coefficient of variation (if applicable) (Use the *summary()* and *sd()* commands.)

```
for (i in c(1,3,6)) {
  print(paste("For column ",i,": "))
  print("Non-integer")
  print(paste("Mean: ",as.numeric(summary(my.dat[,i])[4])))
  print(paste("Median: ",as.numeric(summary(my.dat[,i])[3])))
  print(paste("Standard deviation: ",sd(my.dat[,i])))
  print(paste("Coefficient of variation: ",sd(my.dat[,i])/as.numeric(summary
(my.dat[,i])[4])))
}
```

```
## [1] "For column  1 : "
## [1] "Non-integer"
## [1] "Mean:  20.090625"
## [1] "Median:  19.2"
## [1] "Standard deviation:  6.0269480520891"
## [1] "Coefficient of variation:  0.299988081609661"
## [1] "For column  3 : "
## [1] "Non-integer"
## [1] "Mean:  230.721875"
```

```
## [1] "Median:  196.3"
## [1] "Standard deviation:  123.938693831382"
## [1] "Coefficient of variation:  0.537177906652249"
## [1] "For column  6 : "
## [1] "Non-integer"
## [1] "Mean:  3.21725"
## [1] "Median:  3.325"
## [1] "Standard deviation:  0.978457442989697"
## [1] "Coefficient of variation:  0.304128508194793"
```

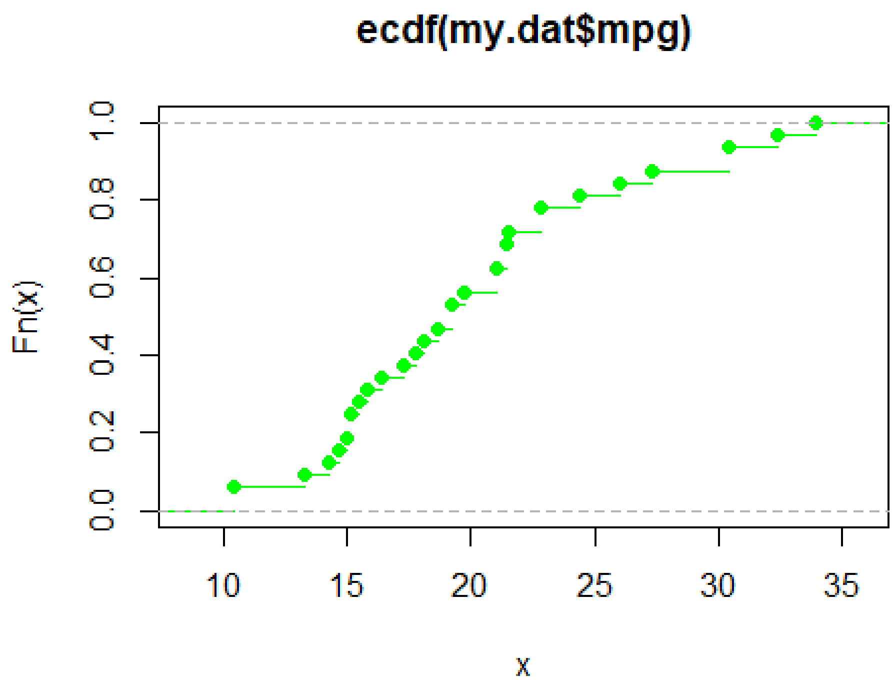
4. Comment on the similarities and differences between each of these samples.

```
All have non-integer values. Three columns have very different mean and media
n value: column 1 has the largest values, column 3 the second, and the column
 6 the last.

However, their mean and median are very closed, and they have similar coeffic
ient of variance: column 1 is 0.3, column 3 is 0.5, and column 6 is 0.3.
```

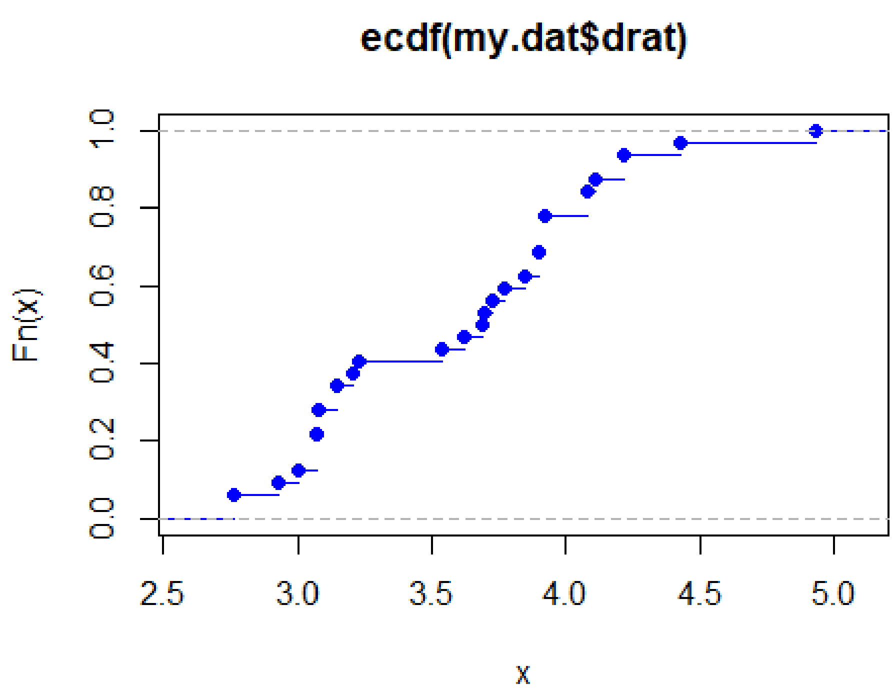
Empirical cumulative distribution functions

The empirical cumulative distribution function (ECDF) of a random sample provides a summary of the sample based on the order (smallest to largest) of the sample. When the sample is perceived to come from a probability distribution, the ECDF can be used to estimate the true cumulative distribution function of the sample. We can calculate the ECDF of a sample by using the *ecdf()* command in **R**. Plot the ECDF of the first, fourth, and fifth columns.

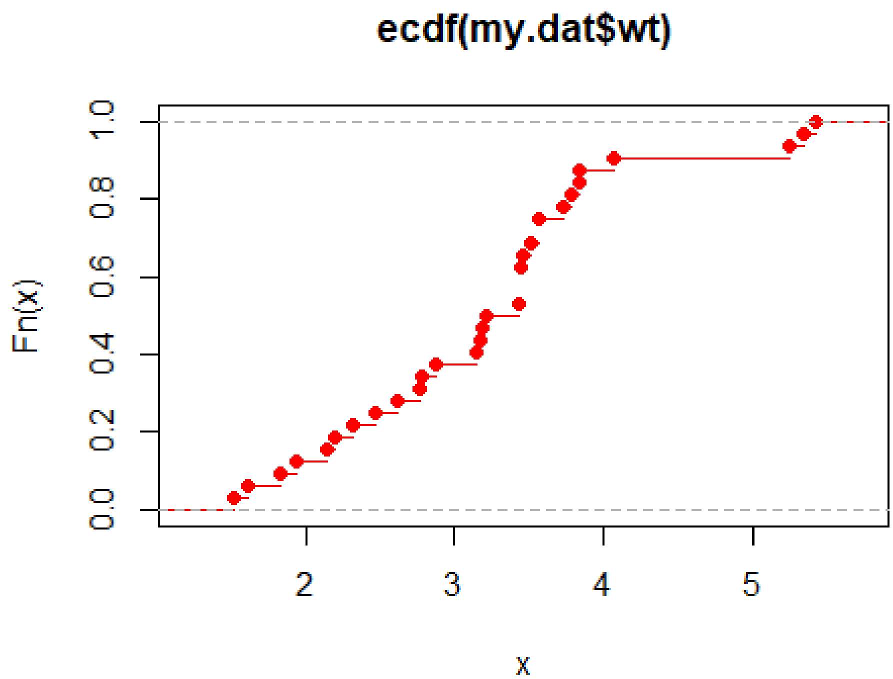
```
# plot the ecdf of car miles per gallon and color it green
plot(ecdf(my.dat$mpg), col = "green")
```




```
# plot the ecdf of rear axle ratio, color it blue
plot(ecdf(my.dat$drat), col = "blue")
```

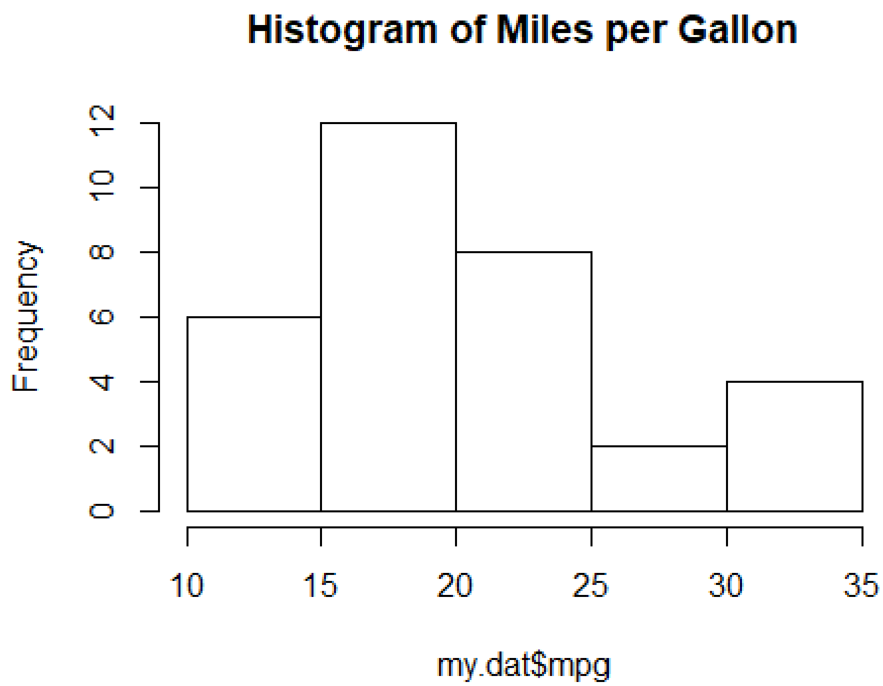


```r
# plot the ecdf of car weight, color it red
plot(ecdf(my.dat$wt), col = "red")
```

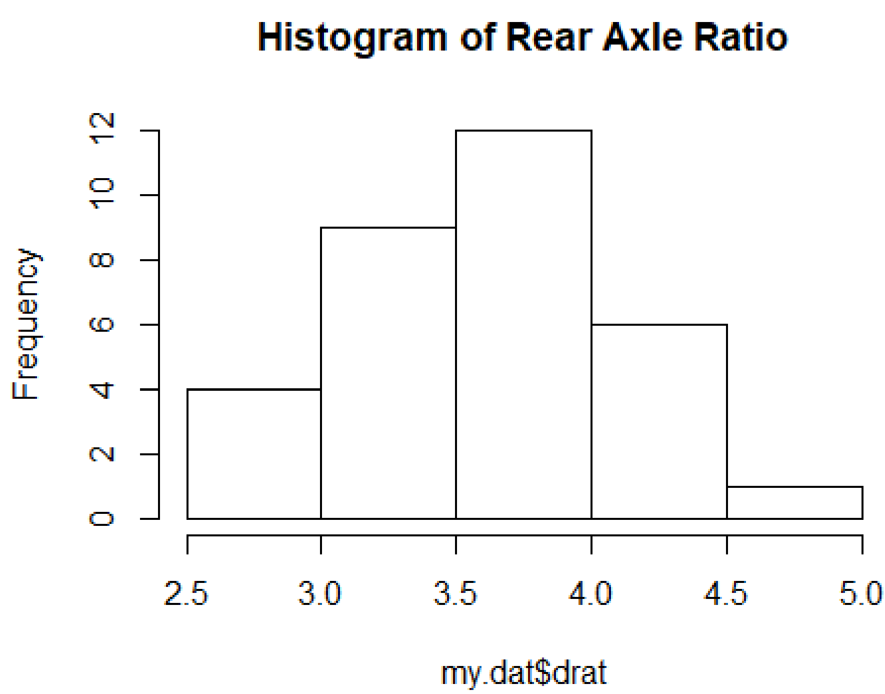


Next, plot the histograms of each of the same columns using the following code:

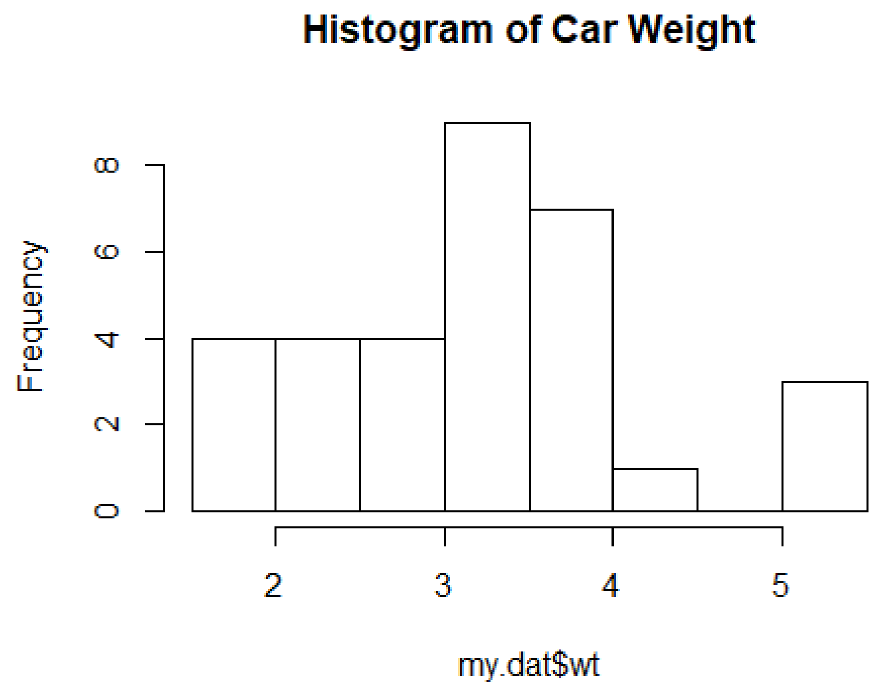
```r
# plot the histogram of x1
hist(my.dat$mpg, main = "Histogram of Miles per Gallon")
```




```r
hist(my.dat$drat, main = "Histogram of Rear Axle Ratio")
```




```r
hist(my.dat$wt, main = "Histogram of Car Weight")
```



Questions

1. Note that the means of `drat` and `wt` are approximately equal. It may be tempting to think that two samples are similar (or even that they are samples from the same population) when they share the same mean. Do the ECDFs of these variables support this claim? Examine closely the beginning of each ECDF.

```
Clearly, the ECDFs of these variables support this claim. First, two ECDFs ha
ve similar shape, and value distribution. Then, when Fn(x)=0.5 we can get aro
und x less than 3.5 on both ECDFs. Also, the value range of both ECDFs are fr
om 2 to 5.
```

2. Comment on what the histograms of each of these samples provides. Do these histograms support the claim that the samples are realizations of the same random variable?

```
The histograms display the frequency distribution of each variable. while 'dr
at' and 'wt' have the similar value range from 2 to 5, the have different dis
tribution, showing that they are less likely to be the samples of same random
 variable.
```

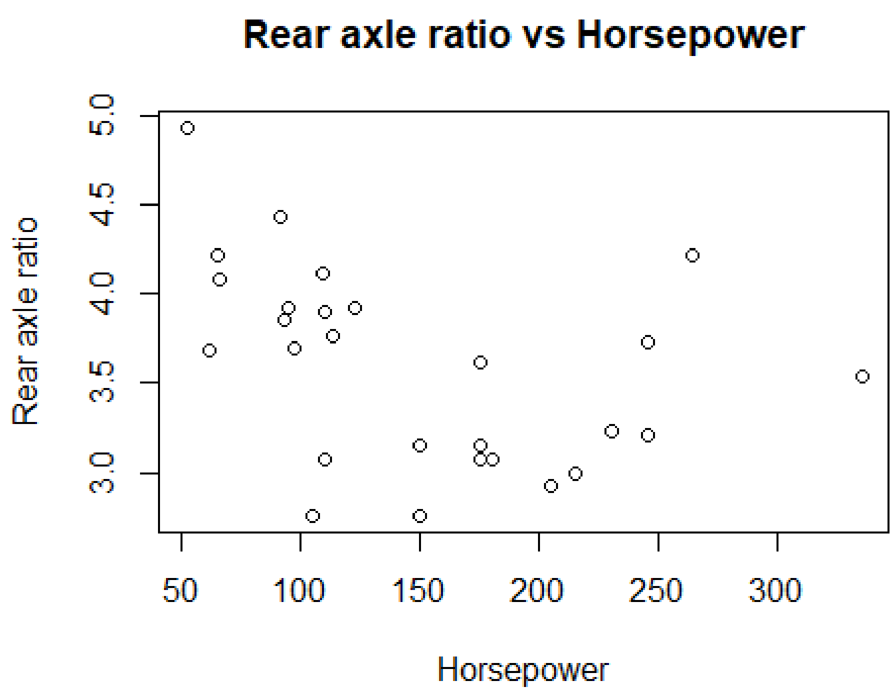
Bivariate relationships and Correlation

Correlation and covariance are two descriptive statistics that quantify the association between two variables. In **R**, we can calculate the sample correlation between two quantitative variables x and y using the *cor(x,y)* command. Similarly, we can use the *cov(x,y)* command to calculate the covariance between x and y. Calculate the pairwise correlations and covariances between `hp`, `drat`, and `weight` using the code below:

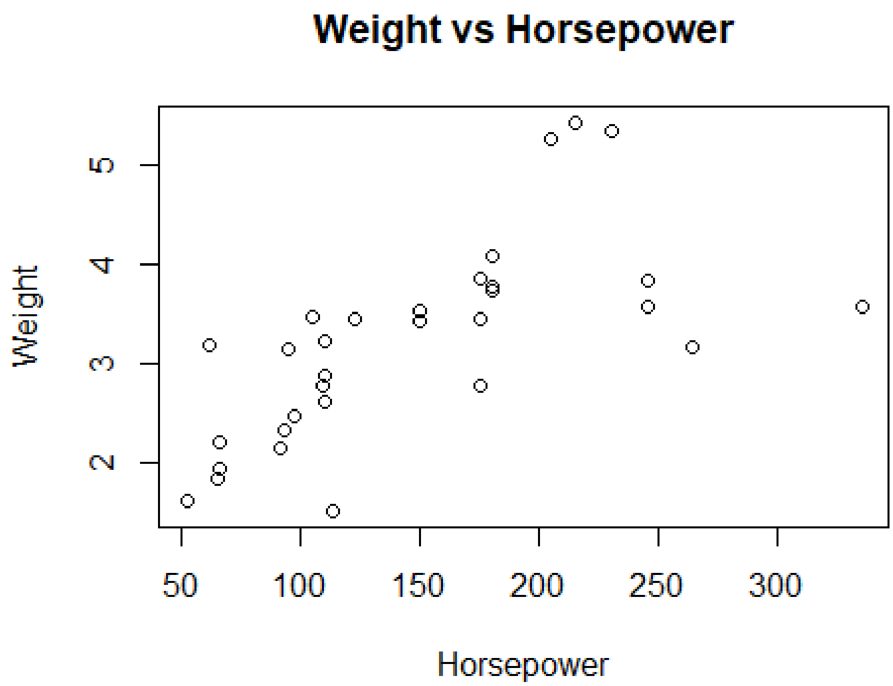
```
# calculate pairwise covariances
attach(my.dat)

## The following object is masked from package:ggplot2:
##
##     mpg

cov.45 = cov(hp,drat)
cov.46 = cov(hp,wt)
cov.56 = cov(drat,wt)

# calculate pairwise correlations
cor.45 = cor(hp,drat)
cor.46 = cor(hp,wt)
cor.56 = cor(drat,wt)
detach(my.dat)
```

Using the *plot()* command, plot a scatterplot between each pair of the above three variables. Be sure to appropriately label each of these plots.

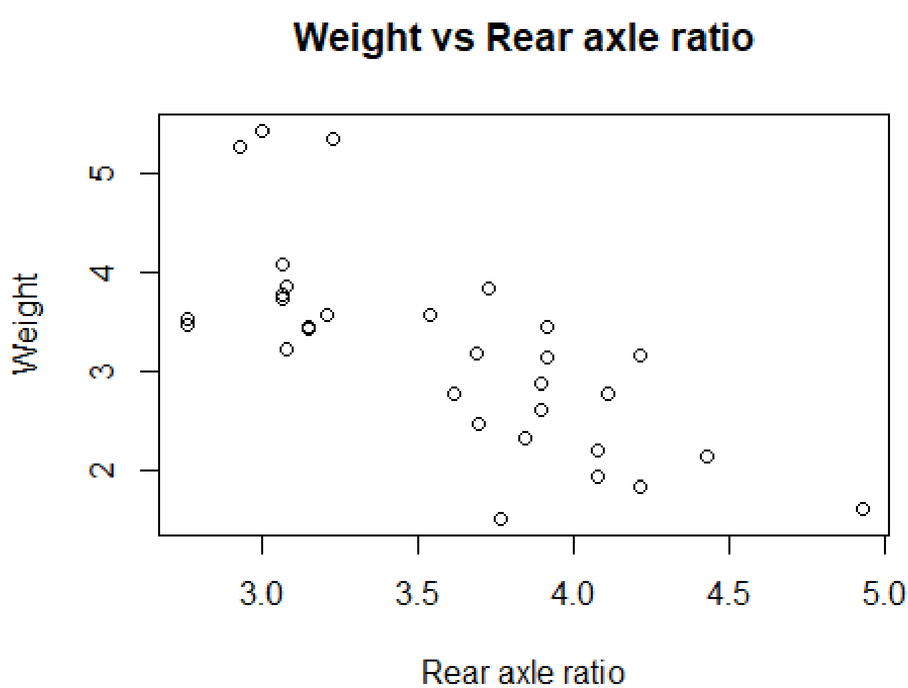
```
plot(x=my.dat$hp,y=my.dat$drat,xlab ="Horsepower",ylab="Rear axle ratio", mai
n = "Rear axle ratio vs Horsepower" )
```




```
plot(x=my.dat$hp,y=my.dat$wt,xlab ="Horsepower",ylab="Weight", main = "Weight
 vs Horsepower" )
```




```r
plot(x=my.dat$drat,y=my.dat$wt,xlab ="Rear axle ratio",ylab="Weight", main =
"Weight vs Rear axle ratio" )
```



Questions

1. Verify that for each of the pairs (`hp`, `drat`), (`hp`, `wt`), and (`drat`, `wt`), the correlation is the quotient of the covariance and the product of the standard deviations.

```
print(cor.45 - cov.45/(sd(my.dat$hp)*sd(my.dat$drat)))

## [1] 0

print(cor.46 - cov.46/(sd(my.dat$hp)*sd(my.dat$wt)))

## [1] 0

print(cor.56 - cov.56/(sd(my.dat$drat)*sd(my.dat$wt)))

## [1] 0
```

2. Comment on each of the generated scatterplots. What does the correlation tell us about the relationships shown in the scatterplots? Does the covariance provide similar information as the correlation?

```
For the first plot "hp vs drat", the correlation is -0.45 showing that they h
ave negative relation but the relation is very loose; for the plot "hp vs wt",
 the correlation is 0.7 showing that they have a positive relation, and the r
elation is quite strong; for the plot "drat vs wt", the correltion is -0.7 sh
owing that they have a negative relation, and the relation is quite strong.

While the covariance does measure the directional relationship between two va
riables, it does not show the strength of the relationship between the them.

Covariance provides different information as the correlation. When covariance
 is positive, the mean of two variables moving together, when they move inver
sely, the covariance is negative.
```

t-tests

One way to test for statistically significant differences between two samples is to use a formal hypothesis test known as the t-test. There are two types of t-statistics that we will consider: the *Student's* t-statistic and the *Welsh* t-statistic. These statistics are used in different situations depending on the variance of the two samples being compared. You can use the typical Student's t-test whenever variances are the same, but should use Welsh's whenever the variances are not equal.

Consider comparing two samples x and y. We can calculate either of these t-test statistics using the function *t.test()*. In particular, if the variance of the two samples are **not** equal, then we use the command *t.test(x, y, var.equal = FALSE)*. If the variances **are** equal, then we use the command *t.test(x, y, var.equal = TRUE)*.

Questions

1. Which t-statistic is appropriate to compare the samples `hp` and `wt`? How about `drat` and `vs`?

```
var.hp = sd(my.dat$hp)^2
var.wt = sd(my.dat$wt)^2
var.drat = sd(my.dat$drat)^2
var.vs = sd(my.dat$vs)^2
```

```
The Welch's t-statistic is appropriate to compare the samples 'hp' and 'wt'.
Because they have different variance, (var.hp=4700, and var.wt=0.95).

The Student's t-statistic is appropriate to compare the samples 'drat' and 'v
s'. Because they have similar variance, (var.drat=0.29, and var.vs=0.25).
```

2. Calculate the t-statistic to compare hp and wt. Are the two samples statistically significantly different at a 0.05 level?

```
t.test(my.dat$hp,my.dat$wt,var.equal = FALSE)

##
##  Welch Two Sample t-test
##
## data:  my.dat$hp and my.dat$wt
## t = 11.836, df = 31.013, p-value = 4.919e-13
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  118.7486 168.1919
## sample estimates:
## mean of x mean of y
## 146.68750   3.21725
```

```
Since the p-value=4.9e-13 which is smaller than 0.05, the two samples are sta
tistically significantly different at a 0.05 level.
```

3. Repeat (2) for drat and vs.

```
t.test(my.dat$drat,my.dat$vs,var.equal = TRUE)

##
##  Two Sample t-test
##
## data:  my.dat$drat and my.dat$vs
## t = 24.32, df = 62, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  2.899409 3.418716
## sample estimates:
## mean of x mean of y
##  3.596563  0.437500
```

```
Since the p-value<2.2e-16 which is smaller than 0.05, the two samples are sta
tistically significantly different at a 0.05 level.
```

Fisher's iris data

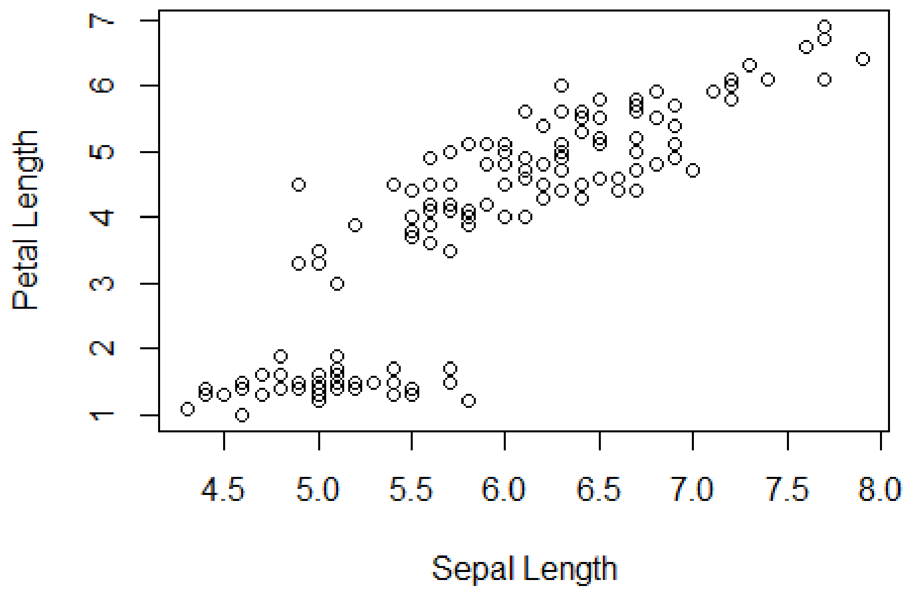
Now we will apply the above techniques to further explore the *iris* data set in **R**. Suppose we want to study the data in the *iris* dataset by flower species. It is easy in **R** to subset datasets based on there variables. For example:

```
# Load the iris data
data(iris)

# Make the setosa species subset
iris.setosa = iris[iris$Species == "setosa", ]

# Look at the five-number summary for only the setosa species
summary(iris.setosa)
```

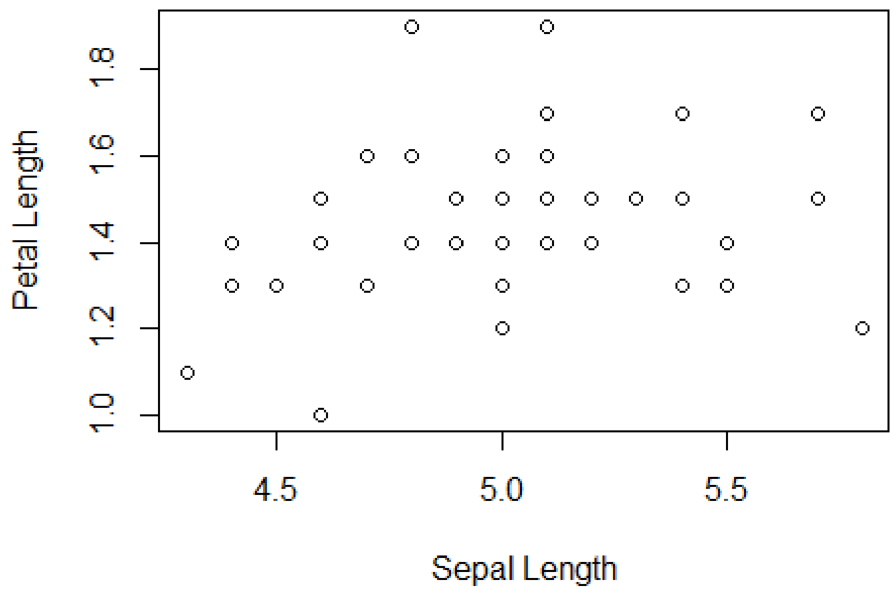
```
##   Sepal.Length    Sepal.Width     Petal.Length    Petal.Width
##  Min.   :4.300   Min.   :2.300   Min.   :1.000   Min.   :0.100
##  1st Qu.:4.800   1st Qu.:3.200   1st Qu.:1.400   1st Qu.:0.200
##  Median :5.000   Median :3.400   Median :1.500   Median :0.200
##  Mean   :5.006   Mean   :3.428   Mean   :1.462   Mean   :0.246
##  3rd Qu.:5.200   3rd Qu.:3.675   3rd Qu.:1.575   3rd Qu.:0.300
##  Max.   :5.800   Max.   :4.400   Max.   :1.900   Max.   :0.600
##        Species
##  setosa    :50
##  versicolor: 0
##  virginica : 0
##
##
##
```

As you work with more datasets/subsets and more variables, it can become repetitive to call variables via $. A useful function in **R** is *with()*, which wraps around your code and specifies which dataset to work with. Try the following examples:

```
#Plot a scatterplot between the sepal length and the petal length
with(iris,
plot(Sepal.Length, Petal.Length, xlab = "Sepal Length", ylab = "Petal Length
"))
```




```
#Plot a scatterplot between the sepal length and the petal length for only se
tosa species
with(iris.setosa,
plot(Sepal.Length, Petal.Length, xlab = "Sepal Length", ylab = "Petal Length
"))
```



Answer each of the following questions.

Questions

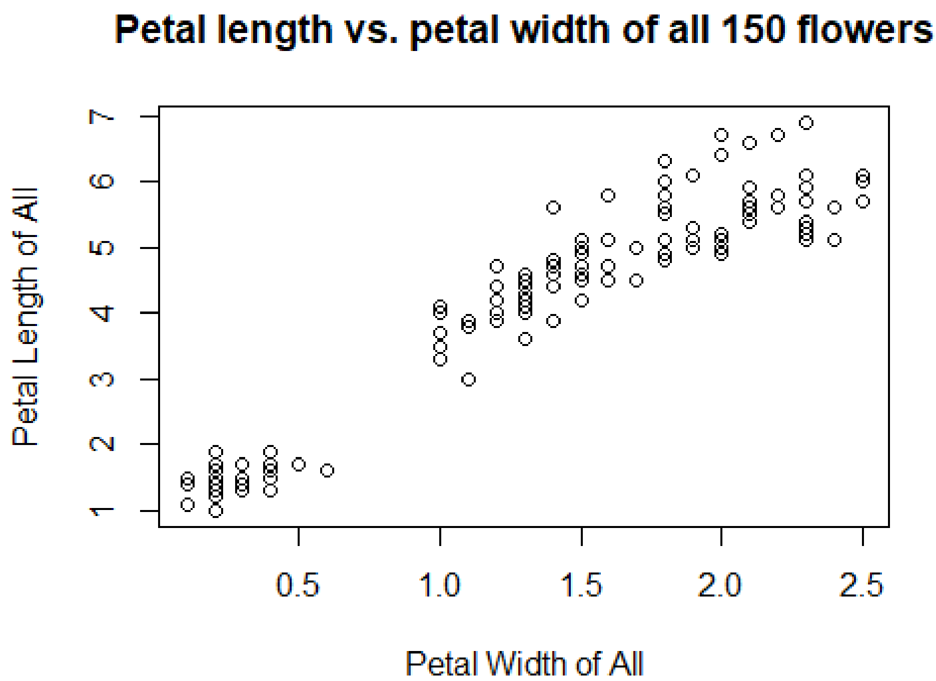
1. Make a table that includes the five-number summary as well as standard deviation of the petal length and petal width of a) all 150 flowers, b) setosa species, and c) virginica species.

```
#Prepare for the table
iris.petal=as.data.frame(matrix(NA, ncol = 7, nrow = 6),row.names = c("all_pe
tal length","all_petal width","setosa_petal length","setosa_petal width","vir
ginica_petal length","virginica_petal width"))

names(iris.petal)=c("Minimum","1st Quartile","Median","Mean","3rd Quartile","
Maximum","Standard Deviation")

#Assign corresponding value
for (i in 1:ncol(iris.petal)) {
  if(i==ncol(iris.petal)){
    iris.petal[1,i]=sd(iris$Petal.Length)
    iris.petal[2,i]=sd(iris$Petal.Width)
    iris.petal[3,i]=sd(iris$Petal.Length[iris$Species=="setosa"])
    iris.petal[4,i]=sd(iris$Petal.Width[iris$Species=="setosa"])
    iris.petal[5,i]=sd(iris$Petal.Length[iris$Species=="virginica"])
    iris.petal[6,i]=sd(iris$Petal.Width[iris$Species=="virginica"])
  }else{
    iris.petal[1,i]=as.numeric(summary(iris$Petal.Length))[i]
    iris.petal[2,i]=as.numeric(summary(iris$Petal.Width))[i]
```

```
    iris.petal[3,i]=as.numeric(summary(iris$Petal.Length[iris$Species=="setos
a"]))[i]
    iris.petal[4,i]=as.numeric(summary(iris$Petal.Width[iris$Species=="setosa
"]))[i]
    iris.petal[5,i]=as.numeric(summary(iris$Petal.Length[iris$Species=="virgi
nica"]))[i]
    iris.petal[6,i]=as.numeric(summary(iris$Petal.Width[iris$Species=="virgin
ica"]))[i]
  }

}

iris.petal

##                         Minimum 1st Quartile Median     Mean 3rd Quartile
## all_petal length            1.0          1.6   4.35 3.758000        5.100
## all_petal width             0.1          0.3   1.30 1.199333        1.800
## setosa_petal length         1.0          1.4   1.50 1.462000        1.575
## setosa_petal width          0.1          0.2   0.20 0.246000        0.300
## virginica_petal length      4.5          5.1   5.55 5.552000        5.875
## virginica_petal width       1.4          1.8   2.00 2.026000        2.300
##                         Maximum Standard Deviation
## all_petal length            6.9          1.7652982
## all_petal width             2.5          0.7622377
## setosa_petal length         1.9          0.1736640
## setosa_petal width          0.6          0.1053856
## virginica_petal length      6.9          0.5518947
## virginica_petal width       2.5          0.2746501
```

2. Generate an appropriately labeled scatterplot showing the relationship between the petal length and petal width of all 150 flowers. Calculate the correlation and covariance between these two variables. Based on what you see, comment on the relationship between these two variables.

```
plot(x=iris$Petal.Width, y=iris$Petal.Length, main="Petal length vs. petal wi
dth of all 150 flowers", xlab = "Petal Width of All", ylab = "Petal Length of
 All")
```




```
cor(iris$Petal.Width,iris$Petal.Length)

## [1] 0.9628654

cov(iris$Petal.Width,iris$Petal.Length)

## [1] 1.295609

From the plot, the correlation is 0.96, and the covariance is 1.30. This indi
cates that the mean value of the two sample moving together, and they have a
strong postive relationship.
```

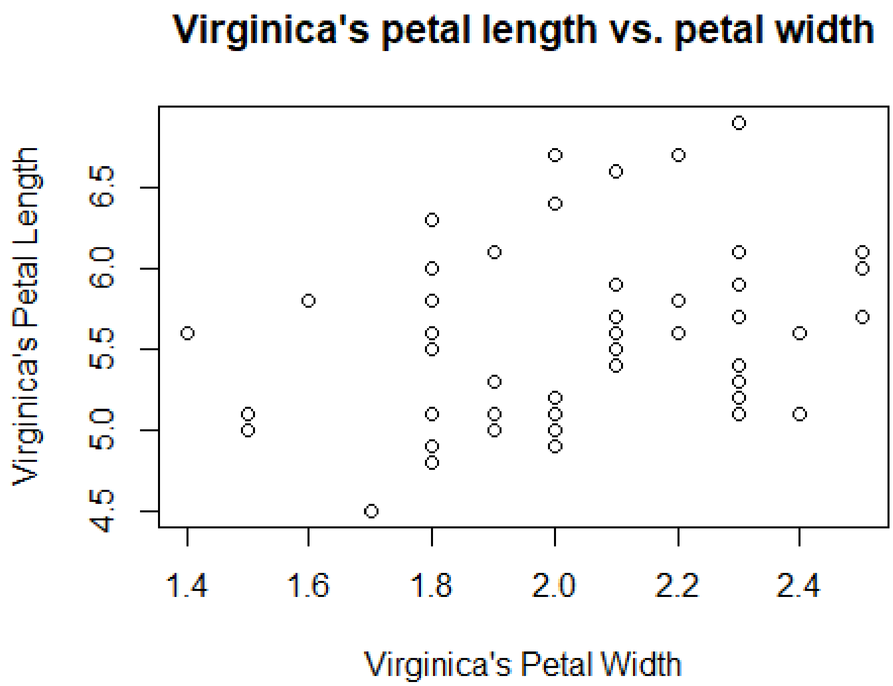
3. Repeat part (b) for the petal length and petal width of a) just the *setosa* species, and b) just the *virginica* species. Comment on what the differences between these two scatterplots and any observations that may be useful in distinguishing the two species.

```
plot(x=iris$Petal.Width[iris$Species=="setosa"], y=iris$Petal.Length[iris$Spe
cies=="setosa"], main="Setosa's petal length vs. petal width", xlab = "Setosa
's Petal Width", ylab = "Setosa's Petal Length")
```


```
cor(iris$Petal.Width[iris$Species=="setosa"],iris$Petal.Length[iris$Species==
"setosa"])

## [1] 0.33163

cov(iris$Petal.Width[iris$Species=="setosa"],iris$Petal.Length[iris$Species==
"setosa"])

## [1] 0.006069388
```

```
From the setosa specials, we can see both from the plot and the correlation,
covariance values that there is no linear relation between petal length and p
etal width.
```

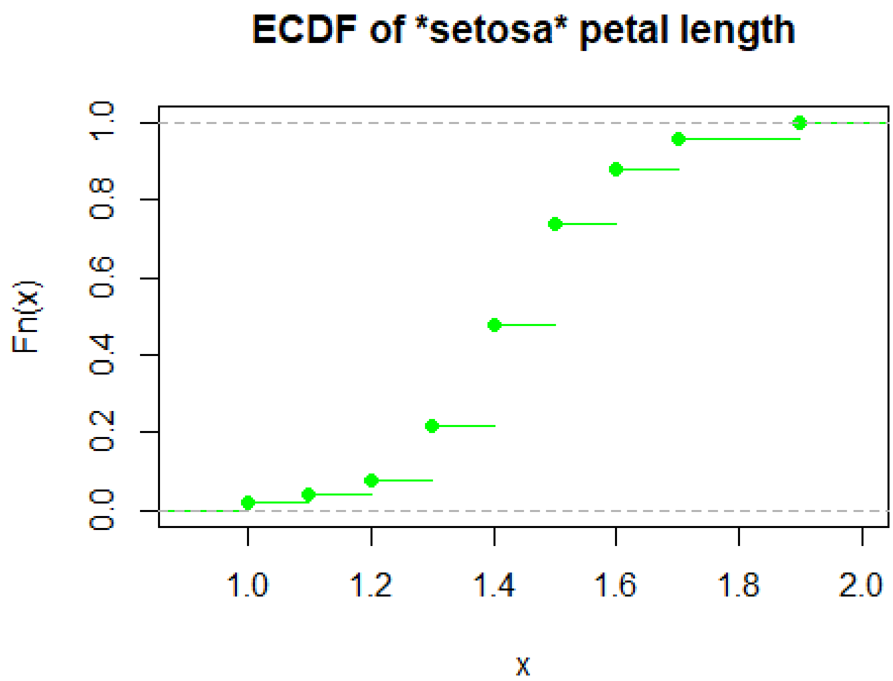
```
plot(x=iris$Petal.Width[iris$Species=="virginica"], y=iris$Petal.Length[iris
$Species=="virginica"], main="Virginica's petal length vs. petal width", xlab
 = "Virginica's Petal Width", ylab = "Virginica's Petal Length")
```



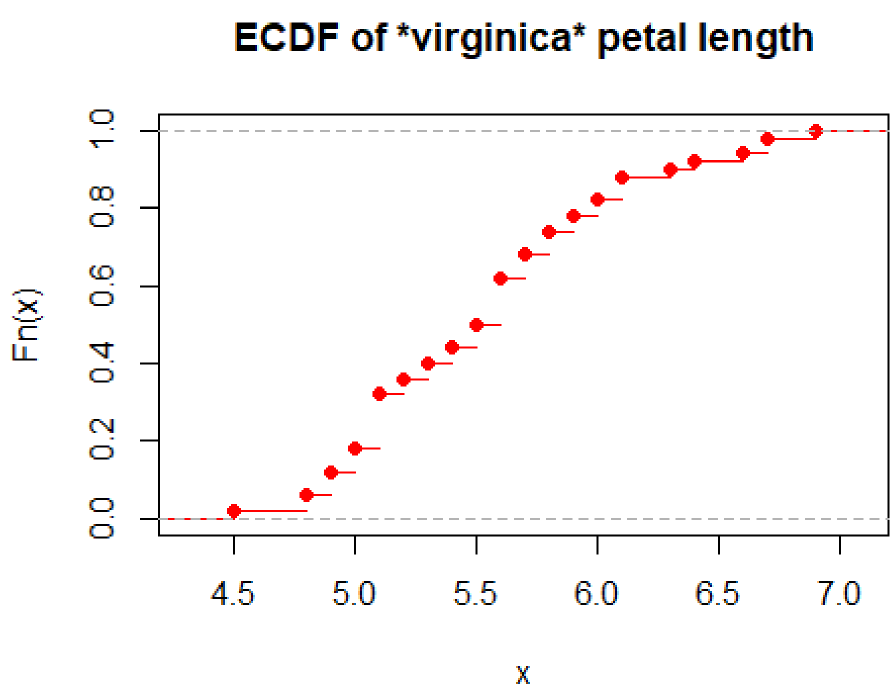

```
cor(iris$Petal.Width[iris$Species=="virginica"],iris$Petal.Length[iris$Specie
s=="virginica"])

## [1] 0.3221082

cov(iris$Petal.Width[iris$Species=="virginica"],iris$Petal.Length[iris$Specie
s=="virginica"])

## [1] 0.04882449

From the virginica specials, we can see both from the plot and the correlatio
n, covariance values that there is no linear relation between petal length an
d petal width.

Even though there is no clear relation on both scatterplot, we can be easily
told from the value range of the graph that Virginica has longer petal width
and longer petal length than Setosa.
```

4. Plot the ECDF of the petal length of the *setosa* and the petal length of the *virginica* species in two different plots. Do the same for the petal width of these two species. Be sure to appropriately label each of the four plots. What do these plots reveal about the relationship between these two species?

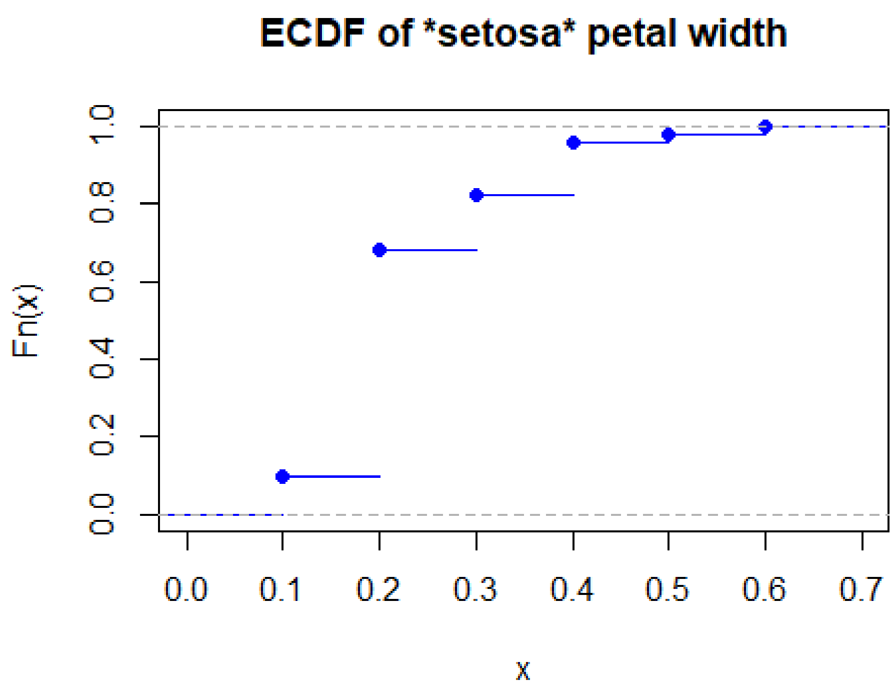
```
plot(ecdf(iris$Petal.Length[iris$Species=="setosa"]), col = "green", main="EC
DF of *setosa* petal length")
```



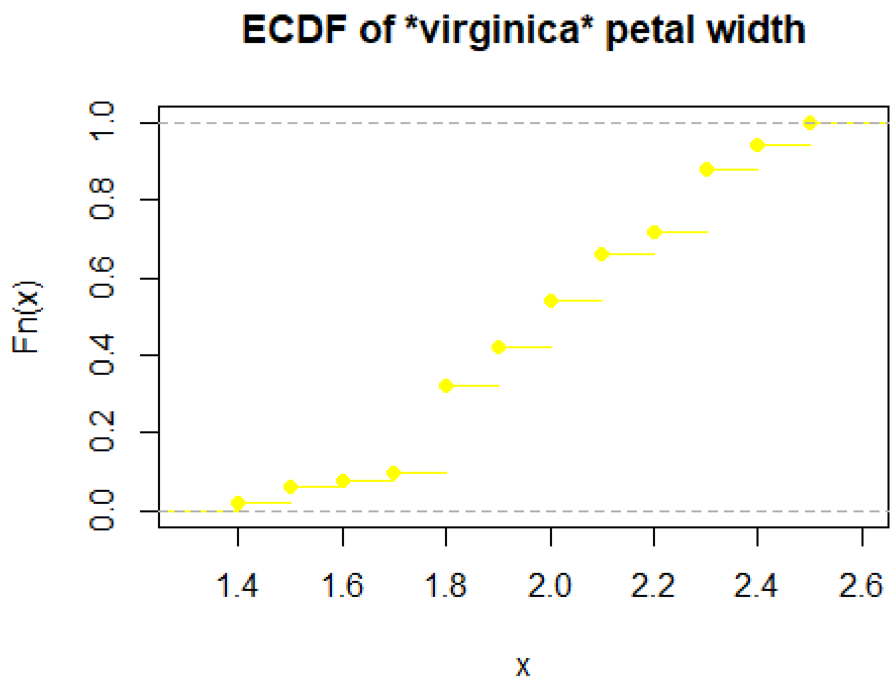

```r
plot(ecdf(iris$Petal.Length[iris$Species=="virginica"]), col = "red", main="ECDF of *virginica* petal length")
```



```r
plot(ecdf(iris$Petal.Width[iris$Species=="setosa"]), col = "blue", main="ECDF
 of *setosa* petal width")
```




```r
plot(ecdf(iris$Petal.Width[iris$Species=="virginica"]), col = "yellow", main=
"ECDF of *virginica* petal width")
```




```
As for the petal length, the ECDFs of both species have similar shape, but vi
rginica have more value and a longer length.
As for the petal width, the ECDFs of both species have different shape, and s
etosa has a popular width of 0.2. Also, virginica have more value and a longe
r width.
```

5. Which t-statistic is appropriate for testing the difference between the petal length of the *setosa* and *virginica* species? Why? Calculate the t-statistic. Are the petal lengths between these two species statistically different?

```r
var.set.len = sd(iris$Petal.Length[iris$Species=="setosa"])^2
var.vir.len = sd(iris$Petal.Length[iris$Species=="virginica"])^2

t.test(iris$Petal.Length[iris$Species=="setosa"], iris$Petal.Length[iris$Spec
ies=="virginica"], var.equal=FALSE)

##
##  Welch Two Sample t-test
##
## data:  iris$Petal.Length[iris$Species == "setosa"] and iris$Petal.Length[i
ris$Species == "virginica"]
## t = -49.986, df = 58.609, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -4.253749 -3.926251
## sample estimates:
```

```
## mean of x mean of y
##     1.462     5.552
```

```
The welch t-statistic is appropriate for testing the difference between the p
eral length of the 'setosa' and 'virginica' species, because they have differ
ent variance(setosa is 0.03, and the virginica is 0.30)

Since the p-value of the t-test is less than 2.2e-16, the petal lengths betwe
en these two species are statistically different.
```

6. Which t-statistic is appropriate for testing the difference between the petal width of the *setosa* and *virginica* species? Why? Calculate the t-statistic. Are the petal widths between these two species statistically different?

```
var.set.wid = sd(iris$Petal.Width[iris$Species=="setosa"])^2
var.vir.wid = sd(iris$Petal.Width[iris$Species=="virginica"])^2

t.test(iris$Petal.Width[iris$Species=="setosa"], iris$Petal.Width[iris$Specie
s=="virginica"], var.equal=TRUE)
```

```
##
##  Two Sample t-test
##
## data:  iris$Petal.Width[iris$Species == "setosa"] and iris$Petal.Width[iri
s$Species == "virginica"]
## t = -42.786, df = 98, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -1.862559 -1.697441
## sample estimates:
## mean of x mean of y
##     0.246     2.026
```

```
The student's t-statistic is appropriate for testing the difference between t
he peral length of the 'setosa' and 'virginica' species, because they have si
milar variance(setosa is 0.01, and the virginica is 0.07)

Since the p-value of the t-test is less than 2.2e-16, the petal lengths betwe
en these two species are statistically different.
```