Computer Assignment 2 - Exploratory Analysis

Machine Learning, Spring 2020

YOUR NAME

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.

```
YOUR CODE HERE
```

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?

```
YOUR CODE AND COMMENTS HERE
```

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

```
YOUR CODE HERE
```

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

```
YOUR CODE HERE
```

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.

```
YOUR CODE HERE
```

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`.

```
YOUR CODE HERE
```

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?

```
YOUR CODE AND COMMENTS HERE
```

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

```
YOUR CODE AND COMMENTS HERE
```

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.

```
YOUR CODE HERE
```

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.

```
YOUR CODE HERE
```

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.)

```
YOUR CODE HERE
```

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

YOUR COMMENTS HERE.

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")

# 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:

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

YOUR ANSWER HERE.

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?

YOUR COMMENT HERE.

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)
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.

```
YOUR CODE HERE
```

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.

```
YOUR CODE HERE
```

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?

YOUR COMMENT HERE.

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`?

YOU ANSWERS HERE.

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

```
YOUR CODE, AND RESPONSE, HERE
```

3. Repeat (2) for `drat` and `vs`.

```
YOUR CODE, AND RESPONSE, HERE
```

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)
```

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 setosa 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.

```
YOUR CODE HERE
```

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.

```
YOUR CODE, AND COMMENT, HERE
```

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.

```
YOUR CODE, AND COMMENTS, HERE
```

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?

```
YOUR CODE HERE
```

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?

```
YOUR CODE, AND COMMENTS, HERE
```

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?

```
YOUR CODE, AND COMMENTS, HERE
```