Descriptive Statistics Part II

Md Mominul Islam

11/7/2021

Exploratory Data Analysis

Data Set Information This is perhaps the best known database to be found in the pattern recognition literature. Fisher's paper is a classic in the field and is referenced frequently to this day. (See Duda & Hart, for example.) The data set contains 3 classes of 50 instances each, where each class refers to a type of iris plant. One class is linearly separable from the other 2; the latter are NOT linearly separable from each other.

Predicted attribute: class of iris plant.

This is an exceedingly simple domain.

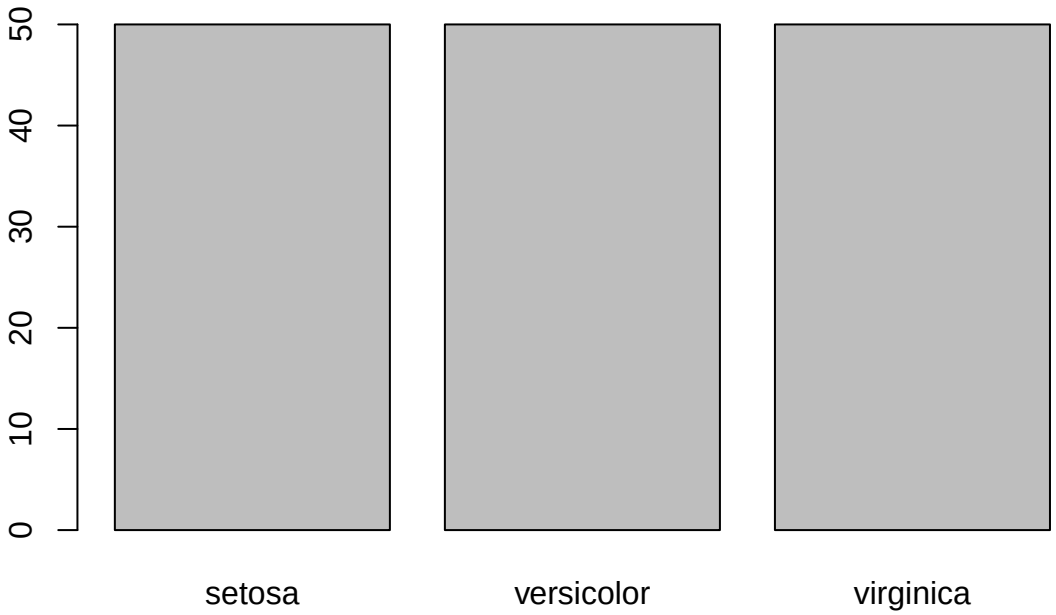
This data differs from the data presented in Fishers article (identified by Steve Chadwick, spchadwick '@' espeedaz.net). The 35th sample should be: 4.9,3.1,1.5,0.2,"Iris-setosa" where the error is in the fourth feature. The 38th sample: 4.9,3.6,1.4,0.1,"Iris-setosa" where the errors are in the second and third features.

Attribute Information:

1. sepal length in cm
2. sepal width in cm
3. petal length in cm
4. petal width in cm
5. class: – Iris Setosa – Iris Versicolour – Iris Virginica

Link: https://archive.ics.uci.edu/ml/datasets/iris

```
attach(iris)
#Barplot as table
barplot(table(iris$Species))
```



Explore Multiple Variables

We can investigate relationship between two variables. Both covariance and correlation measure the relationship and the dependency between two variables. Covariance indicates the direction of the linear relationship between variables while correlation measures both the strength and direction of the linear relationship between two variables. Correlation is a function of the covariance.

If an increase in one variable results in an increase in the other variable, both variables are said to have a positive covariance. Decreases in one variable also cause a decrease in the other. Both variables move together in the same direction when they change. Decreases in one variable resulting in the opposite change in the other variable are referred to as negative covariance. These variables are inversely related and always move in different directions. When a positive number is used to indicate the magnitude of covariance, the covariance is positive. A negative number represents an inverse relationship.

The correlation is scaled by the product of the two standard devations of the variables. This measure is called the Pearson correlation which holds true only when the relationship between two variables is linear in nature. When the relationship is non-linear in nature Spearman correlation or rank correlation is used in order to account for the deviation from linearity.

Pearson: The correlation number would always be in the range of -1 to +1. A value of 1 means that the variables always move in the same direction and a value of -1 means the two always move in the opposite direction. In the case where the variables are independent the covariance is zero which means the correlation is also zero. In other words the two variables do not exhibit any movement relative to each other. Any number in between indicates that the one number moves less positively or negatively in relation to changes in another number.

Spearman: This measure is useful when there might be errors in the data and is less sensitive to outliers and more robust.

```
#covariance between sepal length and petal length
cov(iris$Sepal.Length, iris$Petal.Length)
```

```
## [1] 1.274315
```

```
cov(iris[,1:4]) # cov among all attributes
```

```
##              Sepal.Length Sepal.Width Petal.Length Petal.Width
## Sepal.Length    0.6856935  -0.0424340    1.2743154   0.5162707
## Sepal.Width    -0.0424340   0.1899794   -0.3296564  -0.1216394
## Petal.Length    1.2743154  -0.3296564    3.1162779   1.2956094
## Petal.Width     0.5162707  -0.1216394    1.2956094   0.5810063
```

```
#correlation between sepal length and petal length
cor(iris$Sepal.Length, iris$Petal.Length)
```

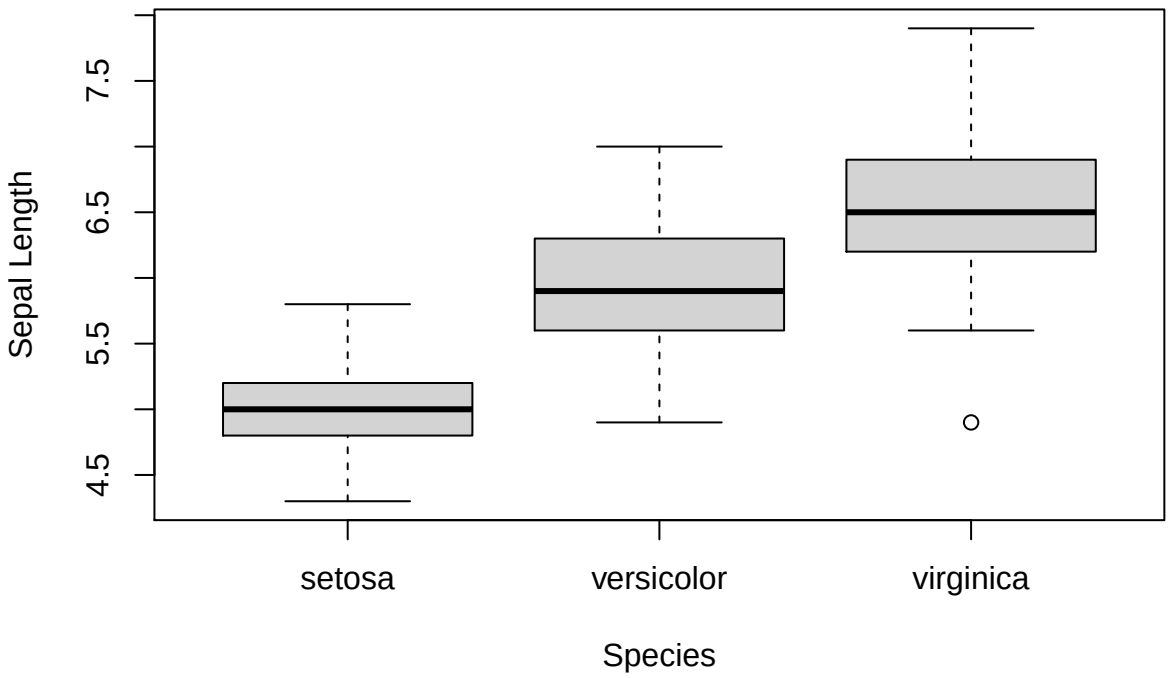
```
## [1] 0.8717538
```

```
cor(iris[,1:4]) # correlation among all attributes
```

```
##              Sepal.Length Sepal.Width Petal.Length Petal.Width
## Sepal.Length    1.0000000  -0.1175698    0.8717538   0.8179411
## Sepal.Width    -0.1175698   1.0000000   -0.4284401  -0.3661259
## Petal.Length    0.8717538  -0.4284401    1.0000000   0.9628654
## Petal.Width     0.8179411  -0.3661259    0.9628654   1.0000000
```

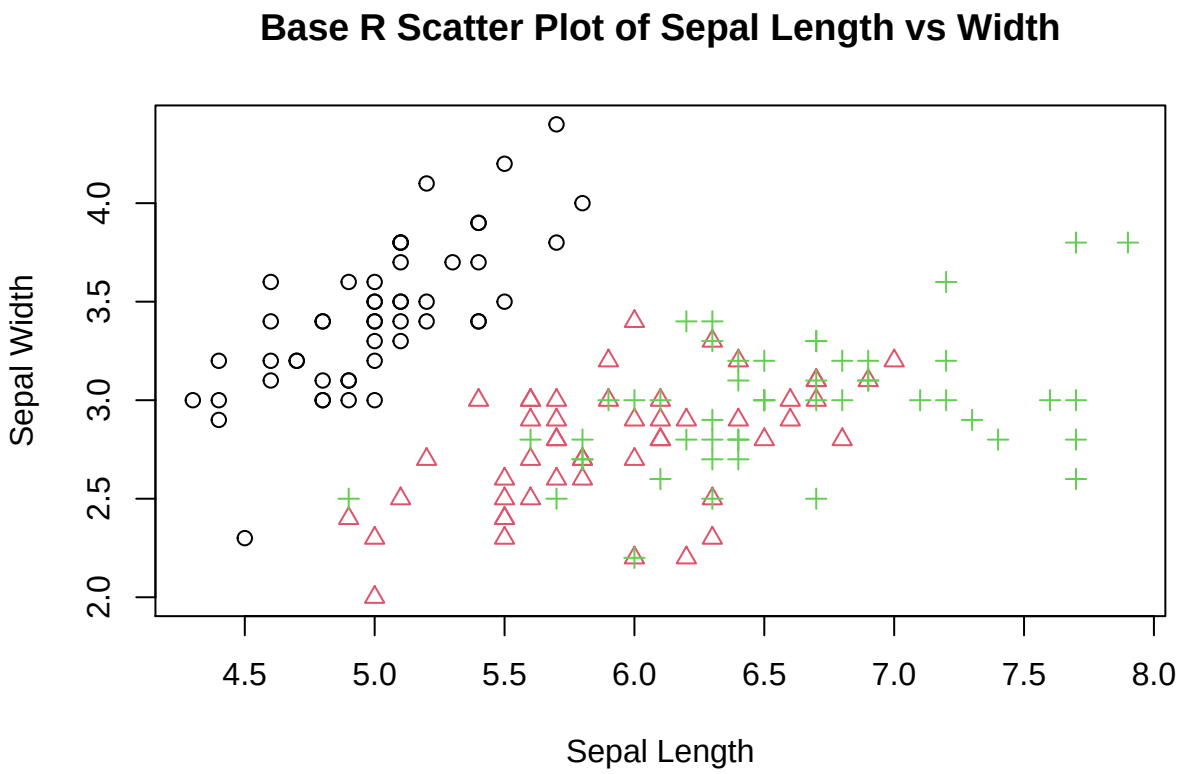
Boxplot

```
boxplot(Sepal.Length~Species,
        ylab = "Sepal Length",
        data=iris)
```



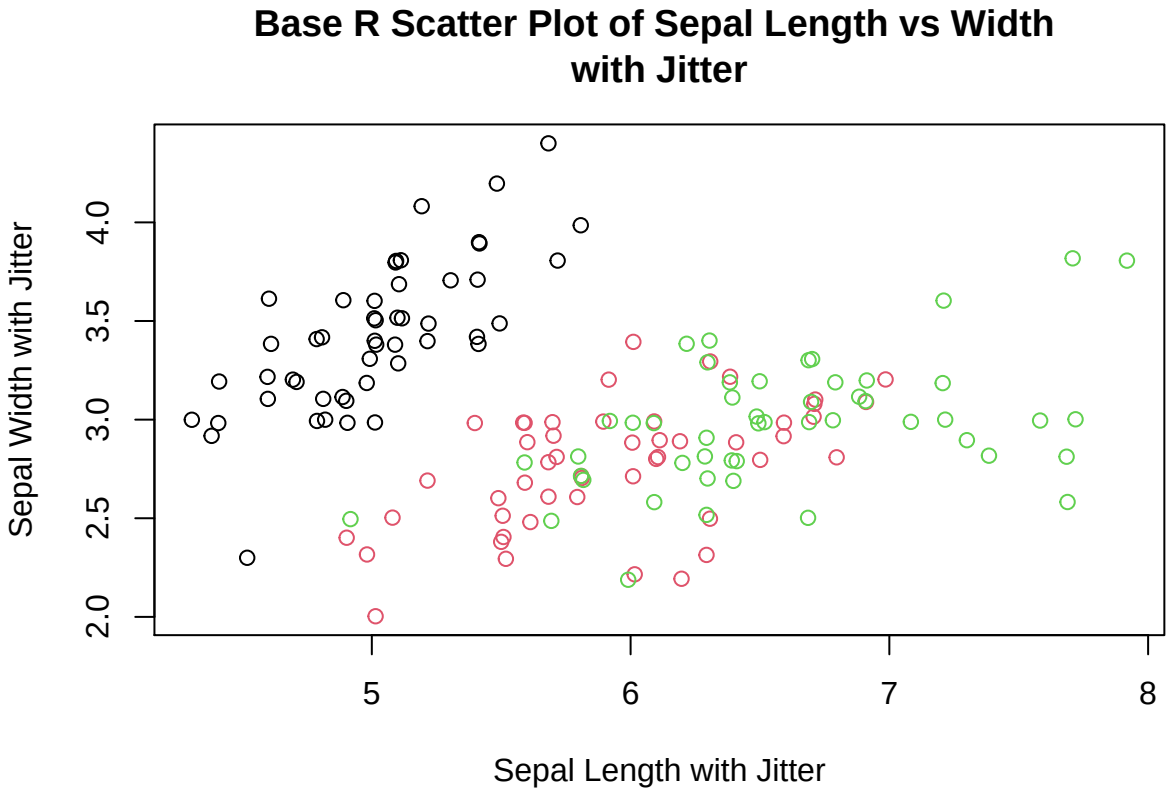
A scatter plot can be drawn for two numeric variables with plot() as below. Using function with(), we don't need to add "iris$" before variable names. In the code below, the colors (col) and symbols (pch) of points are set to Species.

```
with(iris, plot(Sepal.Length,
                Sepal.Width,col=Species,pch=as.numeric(Species),
                xlab = "Sepal Length",
                ylab = "Sepal Width",
                main = "Base R Scatter Plot of Sepal Length vs Width"))
```



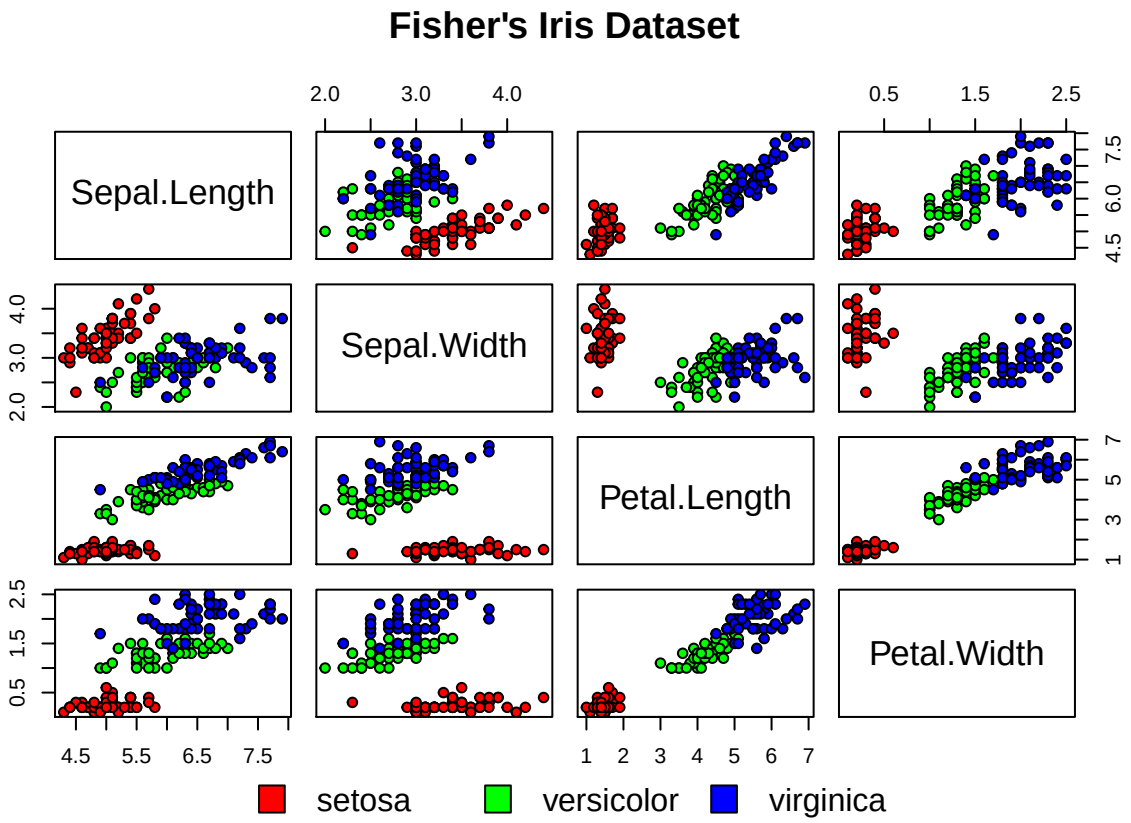
When there are many points, some of them may overlap. We can use jitter() to add a small amount of noise to the data before plotting.

```
plot(jitter(iris$Sepal.Length),
     jitter(iris$Sepal.Width),
     col=Species,
     xlab = "Sepal Length with Jitter",
     ylab = "Sepal Width with Jitter",
     main = "Base R Scatter Plot of Sepal Length vs Width \nwith Jitter")
```



Pairs Plot

```
# define the colors
colors <- c("red", "green", "blue")
# draw the plot matrix
pairs(iris[1:4], main = "Fisher's Iris Dataset",
pch = 21, bg = colors[unclass(iris$Species)] )
# set graphical parameter to clip plotting to the figure region
par(xpd = TRUE)
# add legend
legend(0.2, 0.02, horiz = TRUE, as.vector(unique(iris$Species)),
fill = colors, bty = "n")
```

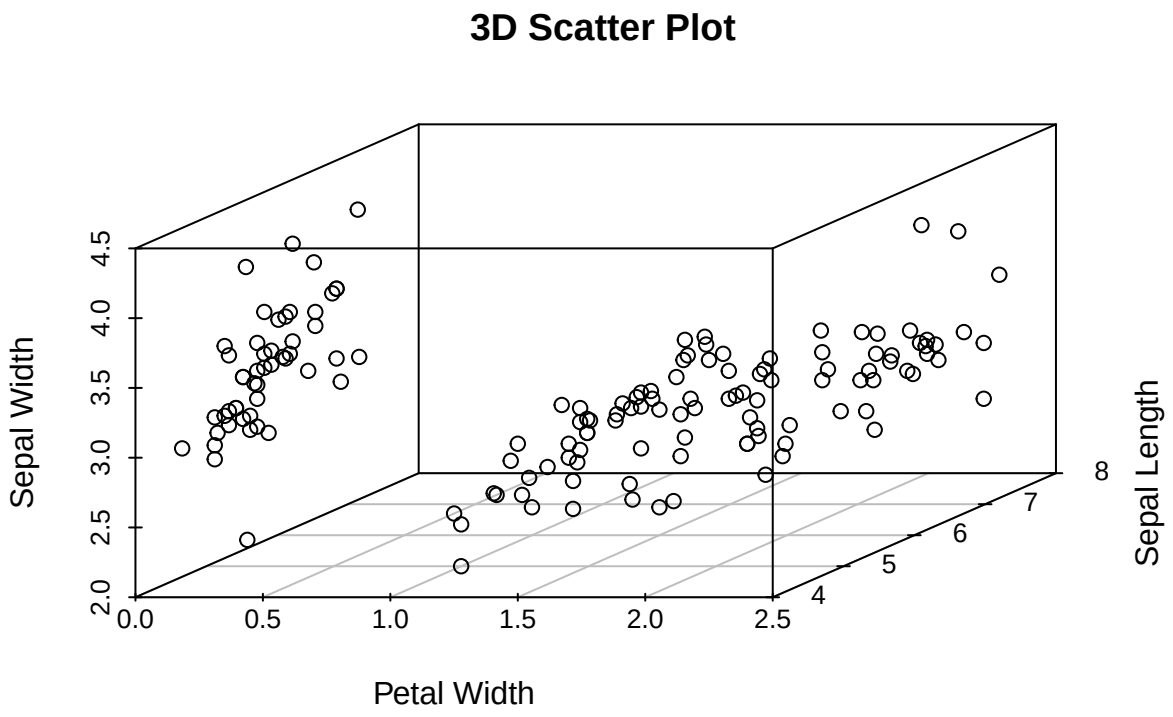


The way to interpret the matrix is as follows:

- The variable names are shown along the diagonals boxes.
- All other boxes display a scatterplot of the relationship between each pairwise combination of variables.
- For example, the box in the top right corner of the matrix displays a scatterplot of values for Sepal Length and Species.
- The box in the top left displays a scatterplot of values for Sepal Length and Sepal Width, and so on.
- From the figure we can say that sepal length and petal length are positively related.

3D Scatter Plot

```
library(scatterplot3d)
scatterplot3d(iris$Petal.Width,
              iris$Sepal.Length,
              iris$Sepal.Width,
              xlab = "Petal Width",
              ylab = "Sepal Length",
              zlab = "Sepal Width",
              main = "3D Scatter Plot")
```



R Function Exercise 4

[Book Link: https://link.springer.com/book/10.1007/978-0-387-93837-0]

Exercise 1. The use of the tapply, sapply, and lapply functions to calculate mean temperature per month.

The file temperature.xls contains temperature observations made at 31 locations along the Dutch coastline. The data were collected and provided by the Dutch institute RIKZ (under the monitoring program MWTL; Monitoring Waterstaatkundige Toestand des Lands). Sampling began in 1990, and the final measurements in the spreadsheet were taken in December 2005, a period of 16 years. Sampling frequency was 0–4 times per month, depending on the season.

```
library(readxl)
setwd("/Users/mominul/OneDrive - South Dakota State University - SDSU/DATA SCIENCE/Books/R Books/A Begi
temp.data <- read_excel("Temperature.xls")
head(temp.data)
```

```
## # A tibble: 6 x 16
##   Sample          Date DateNr dDay1 dDay2 dDay3 Station Area  `31UE_ED50` `31UN_ED50`
##   <chr>          <dbl> <chr>  <dbl> <dbl> <dbl> <chr>   <chr>       <dbl>       <dbl>
## 1 DANT.19900110 1.99e7 33147      7     9     9 DANT    WZ        681380.    5920571.
## 2 DANT.19900206 1.99e7 33026     34    36    36 DANT    WZ        681380.    5920571.
## 3 DANT.19900308 1.99e7 33088     64    66    66 DANT    WZ        681380.    5920571.
## 4 DANT.19900404 1.99e7 32967     91    93    93 DANT    WZ        681380.    5920571.
```

```
## 5 DANT.19900509 1.99e7 33121    126   128   128 DANT    WZ        681380.    5920571.
## 6 DANT.19900620 1.99e7 6/20/~   168   170   170 DANT    WZ        681380.    5920571.
## # ... with 6 more variables: Year <dbl>, Month <dbl>, Season <chr>,
## #   Salinity <chr>, Temperature <chr>, CHLFa <chr>
```

Checking Missing Values

```
library(knitr)
#Missing Value Function
NAPerVariable <- function(X1) {
  D1 <- is.na(X1)  #creates a Boolean matrix of the same dimension as X1
  colSums(D1)
}
kable(NAPerVariable(temp.data),
      caption = "Mising Values") # checking missing values
```

Table 1: Mising Values

	x
Sample	0
Date	0
DateNr	0
dDay1	0
dDay2	0
dDay3	0
Station	0
Area	0
31UE_ED50	0
31UN_ED50	0
Year	0
Month	0
Season	0
Salinity	0
Temperature	0
CHLFa	0

There is no missing values in our data set. We have checked applying a function.

variable Type

Converting character columns to numeric

```
char_columns <- sapply(temp.data, is.character)             # Identify character columns
data_chars_as_num <- temp.data                              # Replicate data
data_chars_as_num[ , char_columns] <- as.data.frame(   # Recode characters as numeric
  apply(data_chars_as_num[ , char_columns], 2, as.numeric))
sapply(data_chars_as_num, class)                          # Print classes of all colums
```

Using tapply to Calculate Mean Temparature per month

```
#Mean of temparature per month
mean.temp <- tapply(X = (data_chars_as_num$Temperature), INDEX = data_chars_as_num$Month, FUN = mean,na

mean.temp2 <- tapply(X = as.numeric(temp.data$Temperature), INDEX = temp.data$Month, FUN = mean,na.rm=T
data.frame(mean.temp)
```

```
##    mean.temp
## 1   5.174210
## 2   4.737400
## 3   6.125961
## 4   8.702035
## 5  12.293479
## 6  15.659933
## 7  18.077343
## 8  19.388355
## 9  16.995974
## 10 13.619670
## 11  9.848891
## 12  6.746339
```