**DESCRIPTIVE STATISTICS USING R PROGRAMMING ALONG WITH THE DEMONSTRATION**

**IRIS FLOWER FEATURES:**

****

**IRIS-DATA-SET:**

****

**Descriptive statistics** consist of describing simply the data using some **summary statistics** and graphics.

# **Import your data into R**

1. **Prepare your data**
2. **Save your data** in an external .txt tab or .csv files
3. **Import your data into R** as follow:

**# If .txt tab file, use this**

**my\_data <- read.delim(file.choose())**

**# Or, if .csv file, use this**

**my\_data <- read.csv(file.choose())**

**Here, we’ll use the built-in R data set named *iris*.**

**# Store the data in the variable my\_data**

**my\_data <- iris**

# 

# 

# 

# 

# **Check your data**

You can inspect your data using the functions **head**() and **tails**(), which will display the first and the last part of the data, respectively.

**# Print the first 6 rows**

**head(my\_data, 6)**

**----------------------------------------------------------------------------------**

**Output:**

Sepal.Length Sepal.Width Petal.Length Petal.Width Species

1 5.1 3.5 1.4 0.2 setosa

2 4.9 3.0 1.4 0.2 setosa

3 4.7 3.2 1.3 0.2 setosa

4 4.6 3.1 1.5 0.2 setosa

5 5.0 3.6 1.4 0.2 setosa

6 5.4 3.9 1.7 0.4 setosa

# 

# 

# 

# 

# 

# **R functions for computing descriptive statistics**

Some R functions for computing descriptive statistics:

|  |  |
| --- | --- |
| Description | R function |
| **Mean** | **mean**() |
| **Standard deviation** | **sd**() |
| **Variance** | **var**() |
| **Minimum** | **min**() |
| **Maximum** | **maximum**() |
| **Median** | **median**() |
| **Range of values** (minimum and maximum) | **range**() |
| **Sample quantiles** | **quantile**() |
| **Generic function** | **summary**() |
| **Interquartile range** | **IQR**() |

**The function mfv(), for most frequent value, [in modeest package] can be used to find the statistical mode of a numeric vector. (For Finding the MODE)**

# 

# 

# 

# 

# 

# **Descriptive statistics for a single group**

## **Measure of central tendency: mean, median, mode**

The central tendency **measures the “average” or the “middle” of your data**. The most commonly used measures include:

* **the mean:** the average value. It’s sensitive to outliers.
* **the median:** the middle value. It’s a robust alternative to mean.
* and **the mode**: the most frequent value

In R,

* The function **mean**() and **median**() can be used to compute the mean and the median, respectively;
* The function **mfv**() [in the **modeest** R package] can be used to compute the mode of a variable.

The R code below computes the mean, median and the mode of the variable *Sepal.Length* [in *my\_data* data set]:

**# Compute the mean value**

**mean(my\_data$Sepal.Length)**

**[1] 5.843333**

**# Compute the median value**

**median(my\_data$Sepal.Length)**

**[1] 5.8**

**# Compute the mode**

**# install.packages("modeest")**

**require(modeest)**

**mfv(my\_data$Sepal.Length)**

**[1] 5**

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## **Measure of variability**

Measures of variability gives **how “spread out” the data** are.

### **Range: minimum & maximum**

* **Range** corresponds to the biggest value minus the smallest value. It gives you the full spread of the data.

**# Compute the minimum value**

**min(my\_data$Sepal.Length)**

**[1] 4.3**

**# Compute the maximum value**

**max(my\_data$Sepal.Length)**

**[1] 7.9**

**# Range**

**range(my\_data$Sepal.Length)**

**[1] 4.3 7.9**

### 

### 

### 

### 

### **(IQR) Interquartile range**

Recall that, quartiles divide the data into 4 parts. Note that the **interquartile range** (IQR) - corresponding to the difference between the first and third quartiles - is sometimes used as a robust alternative to the standard deviation.

* R function:

**quantile(x, probs = seq(0, 1, 0.25))**

* **x**: numeric vector whose sample quantiles are wanted.
* **probs**: numeric vector of probabilities with values in [0,1].
* Example:

**quantile(my\_data$Sepal.Length)**

0% 25% 50% 75% 100%

4.3 5.1 5.8 6.4 7.9

**By default, the function returns the minimum, the maximum and three quartiles (the 0.25, 0.50 and 0.75 quartiles).**

**##CUSTOMISE THE INTERQUARTILE RANGE**

**To compute deciles (0.1, 0.2, 0.3, …., 0.9), use this:**

**quantile(my\_data$Sepal.Length, seq(0, 1, 0.1))**

**To compute the interquartile range, type this:**

**IQR(my\_data$Sepal.Length)**

**[1] 1.3**

### 

### **Variance and standard deviation**

The variance represents the average squared deviation from the mean. The standard deviation is the square root of the variance. It measures the average deviation of the values, in the data, from the mean value.

**# Compute the variance**

**var(my\_data$Sepal.Length)**

**# Compute the standard deviation =**

**# square root of the variance**

**sd(my\_data$Sepal.Length)**

### **Median absolute deviation**

The median absolute deviation (MAD) measures the deviation of the values, in the data, from the median value.

**# Compute the median**

**median(my\_data$Sepal.Length)**

**# Compute the median absolute deviation**

**mad(my\_data$Sepal.Length)**

### 

### **Which measure to use?**

* **Range**. It’s not often used because it’s very sensitive to outliers.
* **Interquartile range**. It’s pretty robust to outliers. It’s used a lot in combination with the median.
* **Variance**. It’s completely uninterpretable because it doesn’t use the same units as the data. It’s almost never used except as a mathematical tool
* **Standard deviation**. This is the square root of the variance. It’s expressed in the same units as the data. The standard deviation is often used in the situation where the mean is the measure of central tendency.
* **Median absolute deviation**. It’s a robust way to estimate the standard deviation, for data with outliers. It’s not used very often.

In summary, the IQR and the standard deviation are the two most common measures used to report the variability of the data.

## **Computing an overall summary of a variable and an entire data frame**

### **summary() function**

The function **summary**() can be used to display several statistic summaries of either one variable or an entire data frame.

* **Summary of a single variable**. Five values are returned: the mean, median, 25th and 75th quartiles, min and max in one single line call:

**summary(my\_data$Sepal.Length)**

**Min. 1st Qu. Median Mean 3rd Qu. Max.**

**4.300 5.100 5.800 5.843 6.400 7.900**

* **Summary of a data frame**. In this case, the function **summary**() is automatically applied to each column. The format of the result depends on the type of the data contained in the column. For example:
  + If the column is a numeric variable, mean, median, min, max and quartiles are returned.
  + If the column is a factor variable, the number of observations in each group is returned.

**summary(my\_data, digits = 1)**

Sepal.Length Sepal.Width Petal.Length Petal.Width Species

Min. :4 Min. :2 Min. :1 Min. :0.1 setosa :50

1st Qu.:5 1st Qu.:3 1st Qu.:2 1st Qu.:0.3 versicolor:50

Median :6 Median :3 Median :4 Median :1.3 virginica :50

Mean :6 Mean :3 Mean :4 Mean :1.2

3rd Qu.:6 3rd Qu.:3 3rd Qu.:5 3rd Qu.:1.8

Max. :8 Max. :4 Max. :7 Max. :2.5

### **sapply() function**

It’s also possible to use the function **sapply**() to **apply a particular function over a list or vector**.

For instance, we can use it to compute for each column in a data frame, the mean, sd, var, min, quantile, …

**# Compute the mean of each column**

**sapply(my\_data[, -5], mean)**

**Sepal.Length Sepal.Width Petal.Length Petal.Width**

**5.843333 3.057333 3.758000 1.199333**

**# Compute quartiles**

**sapply(my\_data[, -5], quantile)**

**Sepal.Length Sepal.Width Petal.Length Petal.Width**

**0% 4.3 2.0 1.00 0.1**

**25% 5.1 2.8 1.60 0.3**

**50% 5.8 3.0 4.35 1.3**

**75% 6.4 3.3 5.10 1.8**

**100% 7.9 4.4 6.90 2.5**

### 

### 

### 

### 

### 

### 

## **Case of missing values**

Note that, when the data contains missing values, some R functions will return errors or NA even if just a single value is missing.

For example, the **mean()** function will return NA if even only one value is missing in a vector. This can be avoided using the argument **na.rm = TRUE**, which tells to the function to remove any NAs before calculations. An example using the **mean** function is as follow:

**mean(my\_data$Sepal.Length, na.rm = TRUE)**

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## **Graphical display of distributions**

The R package [**ggpubr**](http://www.sthda.com/english/wiki/ggpubr-r-package-ggplot2-based-publication-ready-plots) will be used to create graphs.

### **Installation and loading ggpubr**

* Install the latest version from GitHub as follow:

# Install

**install.packages("ggpubr")**

* Load ggpubr as follow:

**library(ggpubr)**

### 

### 

### 

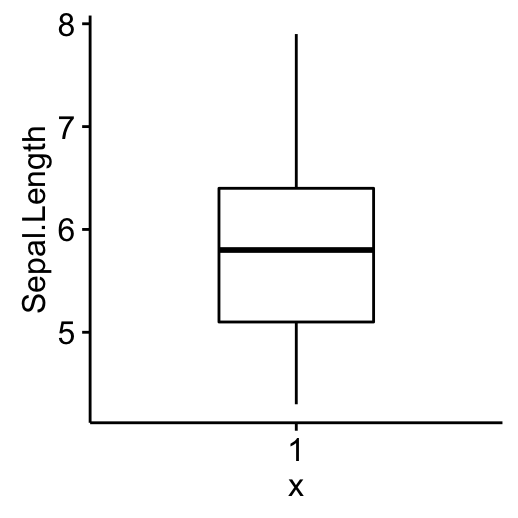
### 

### 

### 

### **Box plots**

**ggboxplot(my\_data, y = "Sepal.Length", width = 0.5)**



### 

### 

### 

### 

### 

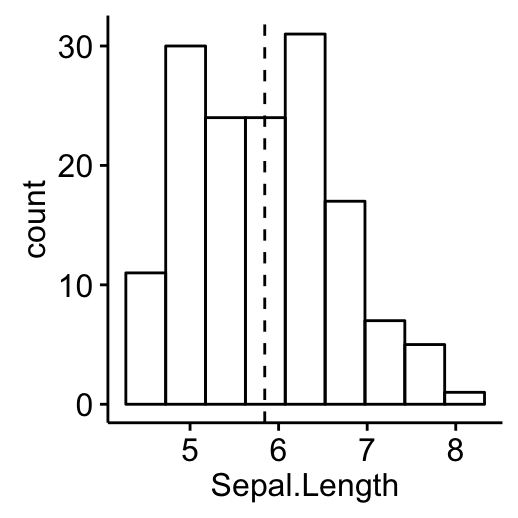
### **Histogram**

Histograms show the number of observations that fall within specified divisions (i.e., bins).

Histogram plot of Sepal.Length with mean line (dashed line).

**gghistogram(my\_data, x = "Sepal.Length", bins = 9,**

**add = "mean")**

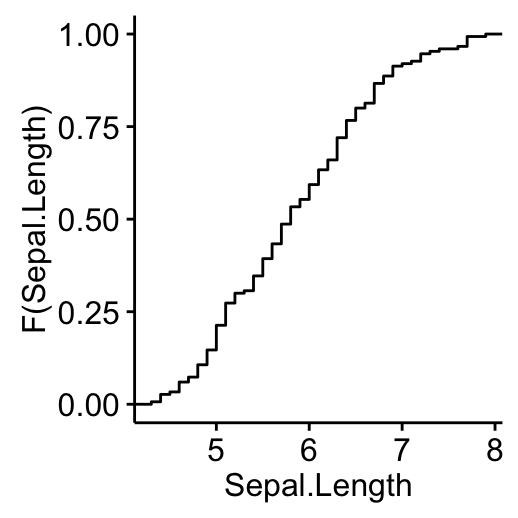


### 

### **Empirical cumulative distribution function (ECDF)**

ECDF is the **fraction of data smaller than or equal to x.**

**ggecdf(my\_data, x = "Sepal.Length")**



### 

### 

### 

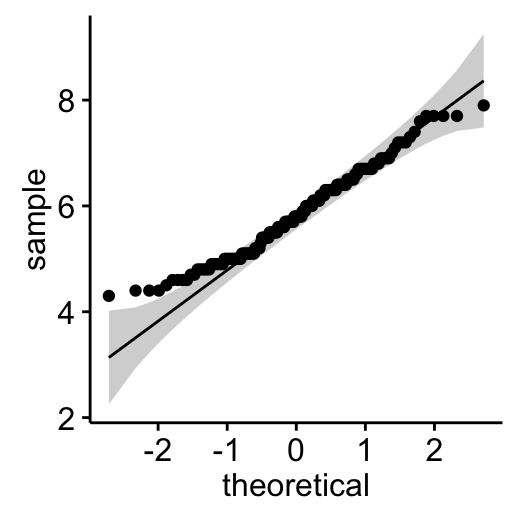
### 

### 

### **Q-Q plots**

QQ plots are used to **check whether the data is normally distributed.**

**ggqqplot(my\_data, x = "Sepal.Length")**



# 

# 

# **Descriptive statistics by groups**

To compute summary statistics by groups, the functions **group\_by**() and **summarise**() [in **dplyr** package] can be used.

* We want to group the data by *Species* and then:
  + compute the number of element in each group. R function: **n**()
  + compute the mean. R function **mean**()
  + and the standard deviation. R function **sd**()

The function **%>%** is used to chain operations.

* Install **ddplyr** as follow:

**install.packages("dplyr")**

* Descriptive statistics by groups:

**library(dplyr)**

**group\_by(my\_data, Species) %>%**

**summarise(**

**count = n(),**

**mean = mean(Sepal.Length, na.rm = TRUE),**

**sd = sd(Sepal.Length, na.rm = TRUE)**

**)**

Source: local data frame [3 x 4]

Species count mean sd

(fctr) (int) (dbl) (dbl)

1 setosa 50 5.006 0.3524897

2 versicolor 50 5.936 0.5161711

3 virginica 50 6.588 0.6358796

* Graphics for grouped data:

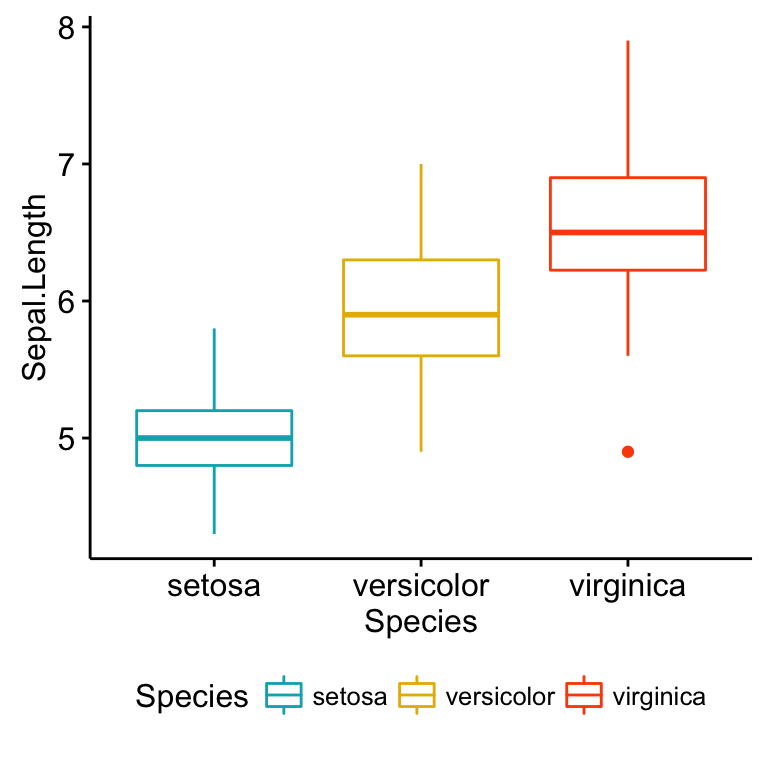
**library("ggpubr")**

**# Box plot colored by groups: Species**

**ggboxplot(my\_data, x = "Species", y = "Sepal.Length",**

**color = "Species",**

palette = c("#00AFBB", "#E7B800", "#FC4E07"))



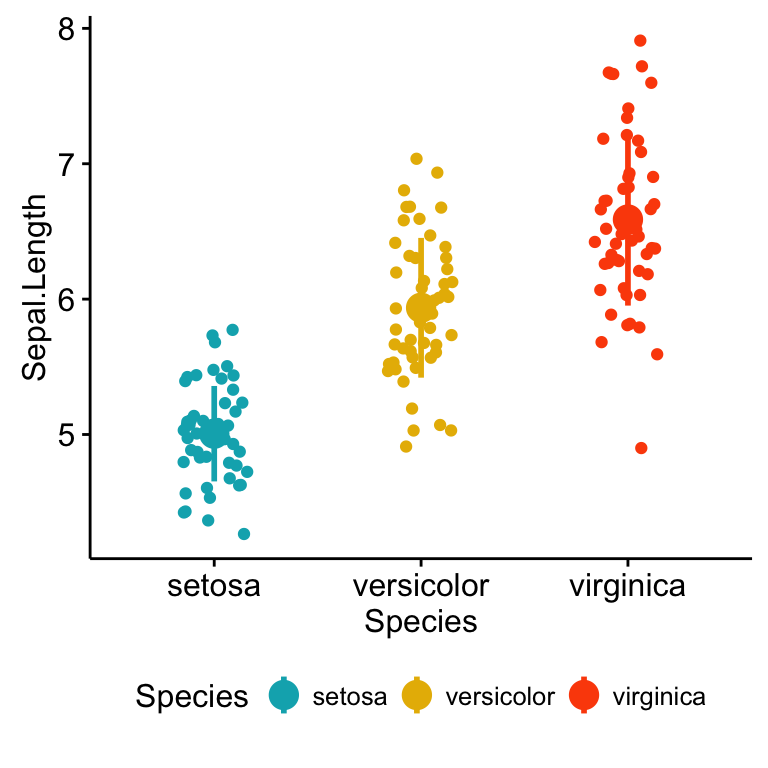
**# Stripchart colored by groups: Species**

**ggstripchart(my\_data, x = "Species", y = "Sepal.Length",**

**color = "Species",**

**palette = c("#00AFBB", "#E7B800", "#FC4E07"),**

**add = "mean\_sd")**



Note that, when the number of observations per groups is small, it’s recommended to use **strip chart** compared to box plots.