Descriptive Statistics

Felipe José Bravo Márquez

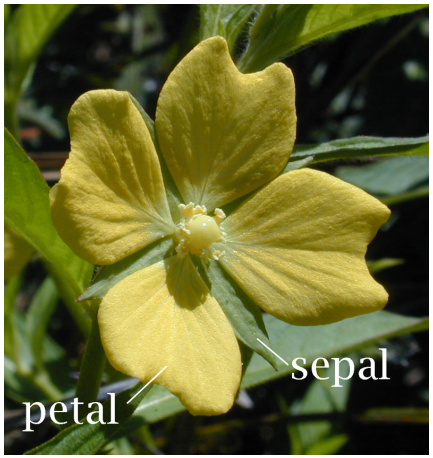
August 12, 2021

Exploratory Data Analysis

- Descriptive Statistics or Exploratory Data Analysis or (EDA) encompasses a set of techniques to quickly understand the nature of a data collection or **dataset**.
- The main goal of descriptive statistics is to explore the data to find some patterns that can be exploited to generate hypotheses.
- It was proposed by the statistician John Tukey.
- It is based mainly on two types of techniques: **summary statistics** and **data visualization**.
- In this class you will see both types of techniques, in addition to their application in R for some toy datasets.
- This class in partially based on chapter 3 of [Tan et al., 2016].

The Iris dataset

- We will work with a well-known dataset called **Iris**.
- The dataset consists of 150 observations of iris plant flowers.
- There are three types of iris flower classes: **virginica**, **setosa** and **versicolor**.
- There are 50 observations of each.
- The variables or attributes measured for each flower are:
  1. The type of flower as a categorical variable.
  2. The length and width of the petal in cm as numerical variables.
  3. The length and width of the sepal in cm as numeric variables.



The Iris dataset

Figure: Virginica - Setosa - Versicolor

- The dataset is available in R:

```
>data(iris) #  to load the dataset into the workspace
> names(iris)
[1] "Sepal.Length" "Sepal.Width"  "Petal.Length"
    "Petal.Width"  "Species"
```

- In order to access the variables directly in our workspace we use the command `attach(iris)`.

Summary Statistics

- Summary statistics are values that explain properties of the data.
- Some of these properties include: frequencies, measures of central tendency and dispersion.
- Example:
  - Central tendency: mean, median, mode.
  - Variation: measure the variability of the data, such as standard deviation, range, etc..
- Most summary statistics can be calculated by making a single pass through the data.

Frequency and Mode

- The absolute frequency of an attribute value is the number of times it is observed.
- The relative frequency is the absolute frequency divided by the total number of examples.
- In R we can count the frequencies of occurrence of each distinct value of a vector using the command `table`:

```
> table(iris$Species)
    setosa versicolor  virginica
        50         50         50
> vec<-c(1,1,1,0,0,3,3,3,3,2)
> table(vec)
vec
0 1 2 3
2 3 1 4
```

- Exercise: Calculate the relative frequencies of the above vector.

```
> table(vec)/length(vec)  # Relative frequencies
vec
  0   1   2   3
0.2 0.3 0.1 0.4
```

Frequency and Mode (2)

- The mode of an attribute is the most frequent value observed.
- The mode function is not implemented natively in R, but it is easy to calculate using `table` and `max`:

```
my_mode<-function(var){
  frec.var<-table(var)
  value<-which(frec.var==max(frec.var))
  as.numeric(names(value))
}
> my_mode(vec)
[1] 3
> my_mode(iris$Sepal.Length)
[1] 5
```

- We generally use frequencies and mode to study categorical variables.

Central Tendency Measures

- These measures attempt to summarize the observed values into a single value associated with the centrally located value.
- The mean is the most common measure of central tendency for a numeric variable.
- If we have n observations it is calculated as the arithmetic mean or average.

$$\text{mean}(x) = \overline{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

- The major problem with the mean is that it is very sensitive to **outliers**.
- We take a random vector of mean 20 and then add a random element that comes from a distribution of much larger mean. We see that the mean is strongly affected by noise:

```
> vec<-rnorm(10,20,10)
> mean(vec)
[1] 16.80036
> vec.noise<-c(vec,rnorm(1,300,100))
> mean(vec.noise)
[1] 35.36422
```

Central Tendency Measures (2)

- We can robust the mean by removing a fraction of the extreme values using the **trimmed mean**.
- In R we can give a second parameter to the function `mean` called `trim` that defines the fraction of extreme elements to discard.
- Example: We discard 10% of the extreme values in the previous example:

```
> mean(vec,trim=0.1)
[1] 17.78799
> mean(vec.noise,trim=0.1)
[1] 19.51609  # much more robust
```

Central Tendency Measures (3)

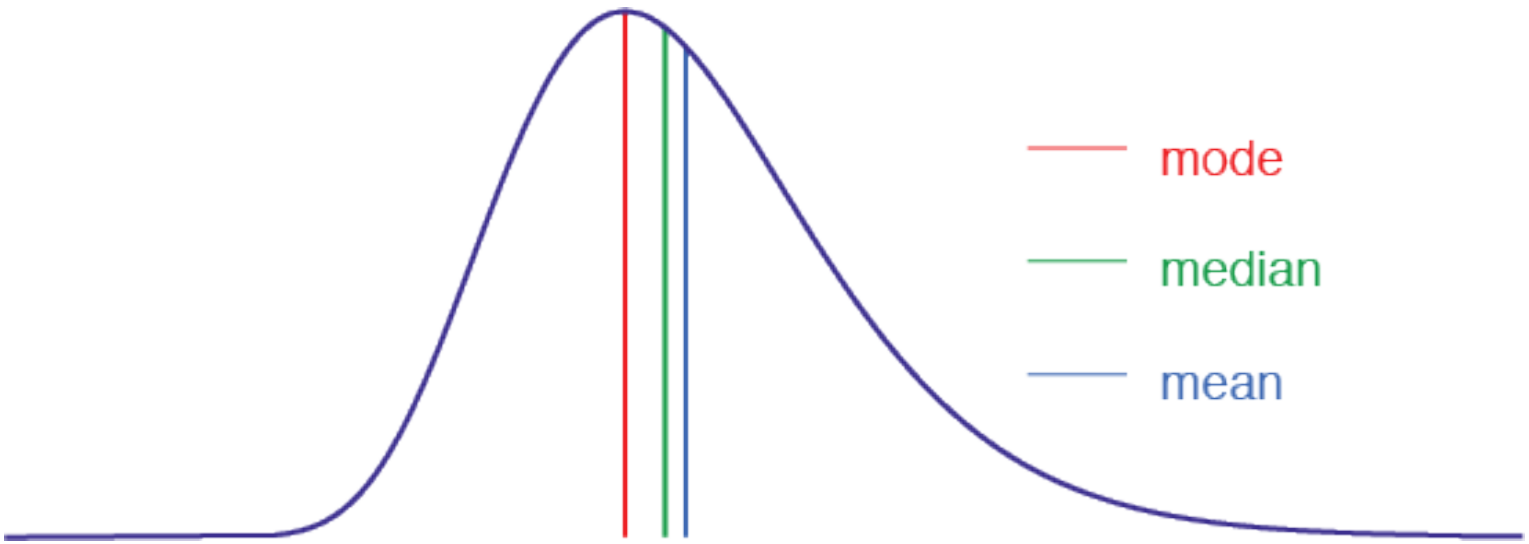
- The median represents the central ranking position of the variable that separates the lower half and the upper half of the observations.
- Intuitively, it consists of the value where for one half of the observations all values are greater than it, and for the other half all are less.

$$\text{median}(x) = \begin{cases} x_{r+1} & \text{If } |x| \text{ (vector length) is odd, } |x| = 2r+1 \\ \frac{1}{2}(x_r + x_{r+1}) & \text{If } |x| \text{ is even, } |x| = 2r \end{cases}$$

- For the above example, we see that the median is more robust to noise than the mean:

```
> median(vec)
[1] 17.64805
> median(vec.noise)
[1] 17.64839
```

Comparison between mode, median and mean



Percentiles or Quantiles

- The k-th percentile of a numerical variable is a value such that k% of the observations are below the percentile and $(100 - k)$% are above this value.
- Quantiles are equivalent to percentiles but expressed in fractions instead of percentages.
- In R they are calculated with the command `quantile`:

```
# All percentiles
quantile(Sepal.Length,seq(0,1,0.01))
```

- In addition it is very common to talk about the **quartiles** which are three specific percentiles:
  - The first quartile Q_1 (lower quartile) is the percentile with $k = 25$.
  - The second quartile Q_2 is with $k = 50$ which is equivalent to the median.
  - The third quartile Q_3 (upper quartile) is with $k = 75$.

```
# The minimum, the three quartiles and the maximum.
> quantile(Sepal.Length,seq(0,1,0.25))
  0%  25%  50%  75% 100%
 4.3  5.1  5.8  6.4  7.9
```

Summarizing a Data Frame

- In R we can summarize various summary statistics of a variable or of a data.frame using the command `summary`.
- For numerical variables it gives us the minimum, the quartiles, the mean and the maximum.
- For categorical variables it gives us the frequency table.

```
> summary(iris)
 Sepal.Length    Sepal.Width     Petal.Length    Petal.Width
 Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100
 1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300
 Median :5.800   Median :3.000   Median :4.350   Median :1.300
 Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199
 3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800
 Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500

 Species
 setosa    :50
 versicolor:50
 virginica :50
```

Exercise

- Using the command `tapply` analyze the mean, median and quartiles for the three species of **Iris** for the four variables.
- Do you notice any differences in the different species?

```
tapply(iris$Petal.Length,iris$Species,summary)
tapply(iris$Petal.Width,iris$Species,summary)
tapply(iris$Sepal.Length,iris$Species,summary)
tapply(iris$Sepal.Width,iris$Species,summary)
```

Variability Measures

- Variability mesaures or dispersion measures tell us how different or similar the observations tend to be with respect to a particular value. Usually this value refers to some measure of central tendency.
- The range is the difference between the maximum and minimum value:

```
> max(Sepal.Length)-min(Sepal.Length)
[1] 3.6
```

- The standard deviation is the square root of the variance that measures the mean squared differences of the observations from the mean.

$$\text{var}(x) = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \overline{x})^2$$

$$\text{sd}(x) = \sqrt{\text{var}(x)}$$

```
> var(Sepal.Length)
[1] 0.6856935
> sd(Sepal.Length)
[1] 0.8280661
```

Variability Measures (2)

- Like the mean, the standard deviation is sensitive to outliers.
- The most robust measures are usually based on the median.
- Let $m(x)$ be a measure of central tendency of x (usually the median), we define the **average absolute deviation** (AAD) as:

$$\text{AAD}(x) = \frac{1}{n}\sum_{i=1}^{n}|x_i - m(x)|$$

- Exercise: Program the function `add` in R, as a function that receives a vector `x` and a central mean function `fun`. The absolute value is calculated with the command `abs`:

```
aad<-function(x,fun=median){
  mean(abs(x-fun(x)))
}
> aad(Sepal.Length)
[1] 0.6846667
> aad(Sepal.Length,mean)
[1] 0.6875556
```

Variability Measures (3)

- Let b be a constant we define the **median absolute deviation** as:

$$\text{MAD}(x) = b \times \text{median}(|x_i - \text{m}(x)|)$$

- In R is calculated with the command `mad` with the parameters `center` as a function measuring the central tendency of the variable and `constant` as the constant b. By default the median and the value 1.482 is used.

```
 > mad(Sepal.Length)
[1] 0.7
```

- Finally, the interquartile range (IQR) is defined as the difference between the third and the first quartile ($Q_3 - Q_1$).

```
IQR(Sepal.Length)
[1] 1.3
```

Multivariate Summary Statistics

- To compare how one variable varies with respect to another, we use multivariate measures.
- The covariance $cov(x, y)$ measures the degree of joint linear variation of a pair of variables x, y:

$$cov(x, y) = \frac{1}{n-1} \sum_{i=1}^{n} (x - \overline{x})(y - \overline{y})$$

- Where $cov(x, x) = var(x)$
- In R it is computed with the command `cov`:

```
> cov(Sepal.Length,Sepal.Width)
[1] -0.042434
```

- If we give it a matrix or a data.frame of numeric variables, it computes a covariance matrix:

```
> cov(iris[,1:4])
              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
```

Multivariate Summary Statistics (2)

- If two variables are independent of each other, their covariance is zero.
- A measure of relationship that does not depend on the scale of each variable is the **linear correlation**.
- The linear correlation or **Pearson's** correlation coefficient $r(x, y)$ is defined as:

$$r(x, y) = \frac{cov(x, y)}{sd(x)sd(y)}$$

- The linear correlation varies between -1 to 1.

Multivariate Summary Statistics (3)

- A value close to 1 indicates that as one variable grows the other also grows in a linear proportion.
- A value close to -1 indicates an inverse relationship (one is growing and the other is decreasing).
- If the correlation is close to zero we have linear independence.
- Note that a correlation of zero does not imply that there cannot be a non-linear relationship between the variables.
- In R, the correlation is calculated with the command `cor`.

```
> cor(iris[,1:4])
             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
```

Contingency Tables

- To analyze the relationship between categorical variables we use **contingency tables**.
- The table is filled with the marginal frequencies of all pairs of values between two categorical variables.
- In R they are created using the command `table` that we used before for computing frequencies, but now giving two vectors as input:

```
gender<-c("Male", "Female", "Male", "Female", "Female", "Male")
studies<-c("college","postgraduate","high school",
           "postgraduate","high school","college")

>table(gender,studies)
        studies
gender   college high school postgraduate
  Female       0           1            2
  Male         2           1            0
```

Skweness and Kurtosis

There are other two summary statistics that focus on more complex properties of the data distribution called skeweness and kurtosis [Pipis, 2020].

Skeweness

- Skeweness is a measure of the **asymmetry** of the probability distribution.

$$\text{skewness}(x) = \frac{\sum_{i=1}^{n}(x_i - \overline{x})^3}{(n-1)sd(x)^3}$$

- It indicates how much our underlying distribution deviates from the normal distribution since the normal distribution has skewness 0.
- Generally, we have three types of skewness:
  1. **Symmetrical**: the skewness is close to 0 and the mean is almost the same as the median.
  2. **Negative skew**: the majority of the observations are concentrated on the right tail (the median is greater than the mean).
  3. **Positive skew**: the majority of the observations are concentrated on the left tail (the median is less than the mean).

Skweness

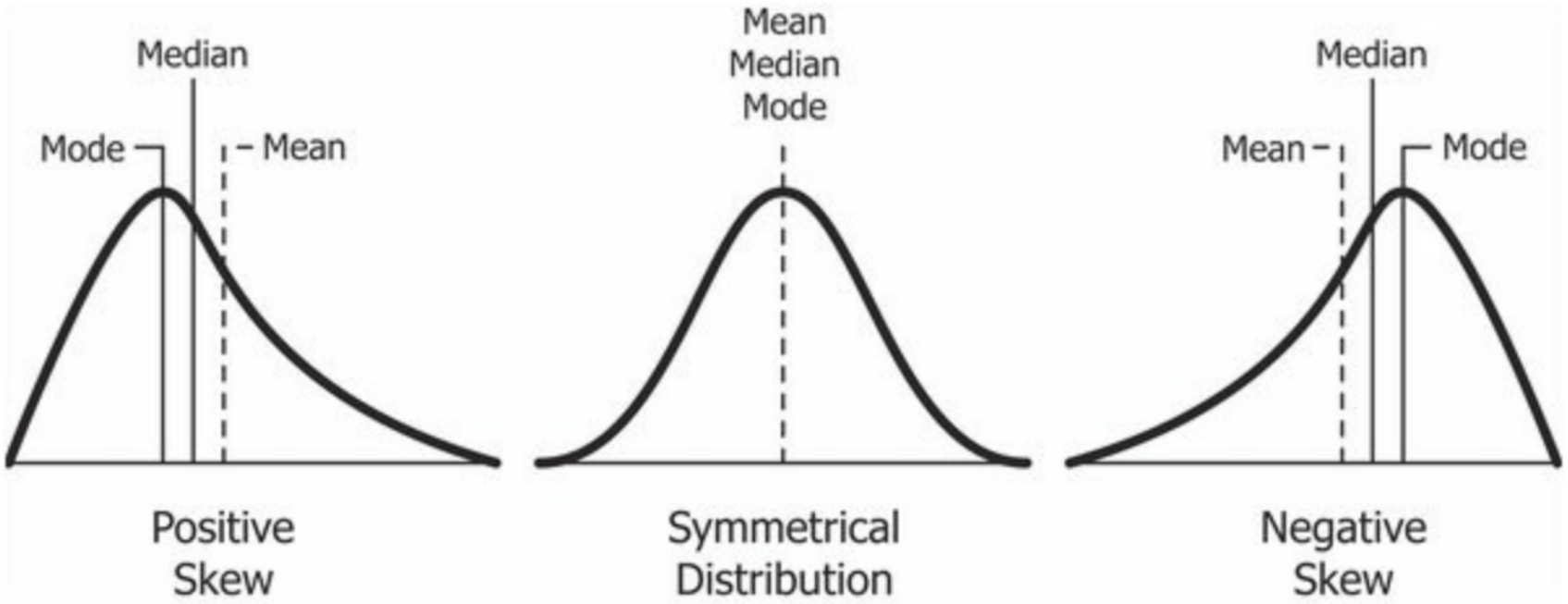


Figure: Source: Wikipedia

Skweness

- In R, we can calculate skweness with the formula **skewness** from the library **moments**:

```
> library(moments)
#positive skew
> skewness(c(1,1,2,3))
[1] 0.4933822
#symmetrical
> skewness(c(1,2,3))
[1] 0
#negative skew
> skewness(c(1,2,3,3))
[1] -0.4933822
```

Kurtosis

Kurtosis

- The Kurtosis describes the "tailedness" of a distribution [Westfall, 2014].

$$\text{kurtosis}(x) = \frac{\sum_{i=1}^{n}(x_i - \overline{x})^4}{(n-1)sd(x)^4}$$

- Let's see the main three types of kurtosis.

1. **Mesokurtic**: This is the normal distribution (kurtosis ≈ 3)
2. **Leptokurtic**: This distribution has fatter tails than a normal distribution (kurtsosis > 3). Consequently, outliers are more likely to occur than in a normal distribution.
3. **Platykurtic**: The distribution has thinner tails than a normal distribution (kurtosis < 3). Consequently, outliers are less likely to occur than in a normal distribution.

Kurtosis

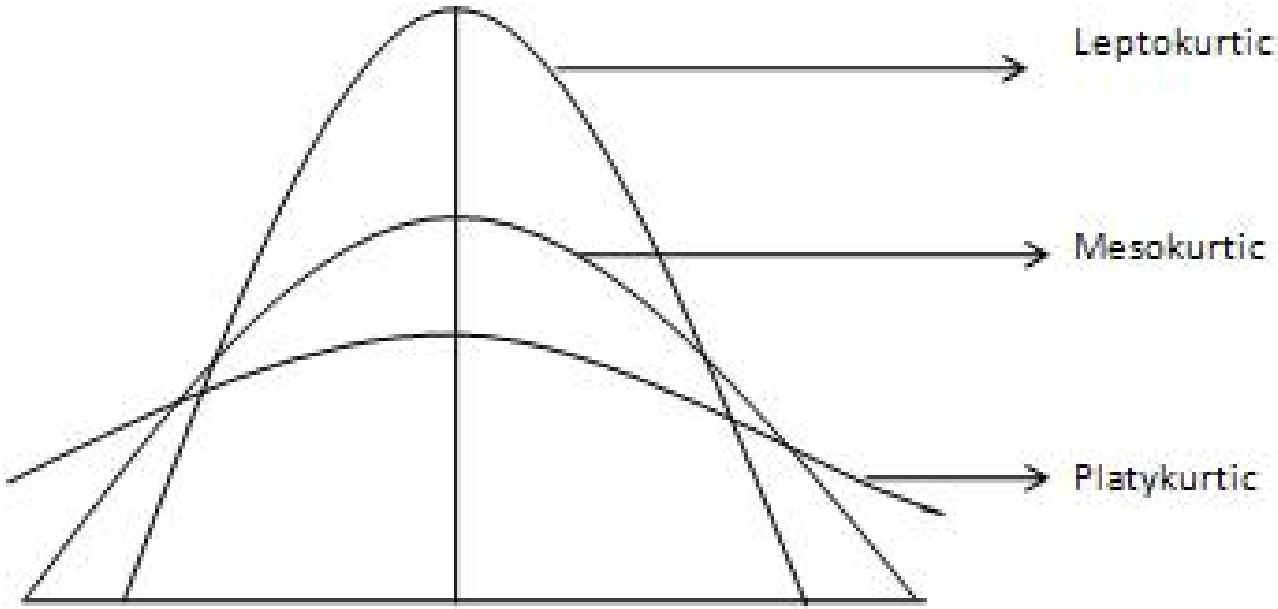


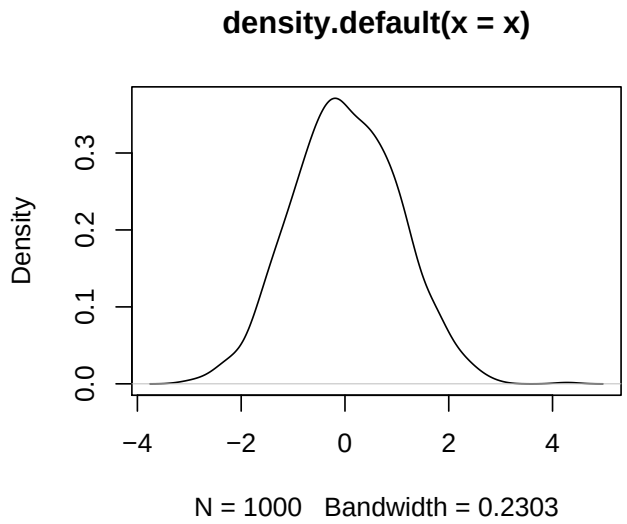
Figure: Source: tutorialspoints

Kurtosis

- In R, we can calculate kurtosis with the formula **kurtosis** from the library **moments**.
- Let's calculate kurtosis for normal data:

```
> x <- rnorm(1000, 0,1)
> plot(density(x))
> kurtosis(x)
[1] 2.946869
```

- As expected we got a value close to 3.

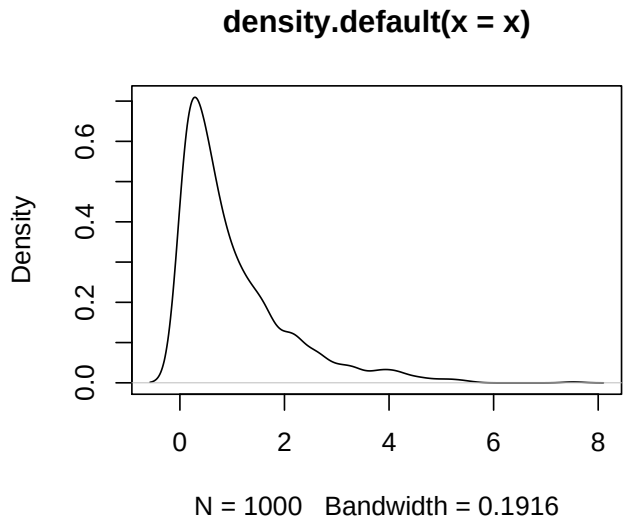


Kurtosis

- Now using random data generated with an exponential long-tailed distribution:

```
> x<-rexp(1000)
> plot(density(x))
> kurtosis(x)
[1] 6.839485
```

- As expected we get a positive excess kurtosis (i.e. greater than 3) since the distribution has fatter tails (Leptokurtic).

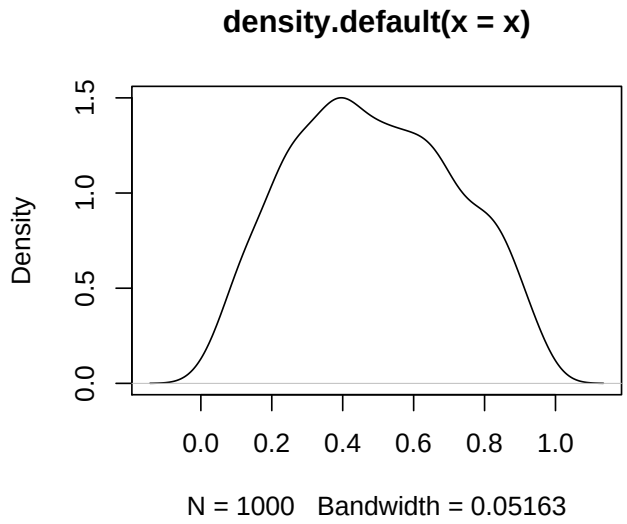


Kurtosis

- Now using data generated with a thinner-tailed Beta distribution with hyperparameters $2, 2$:

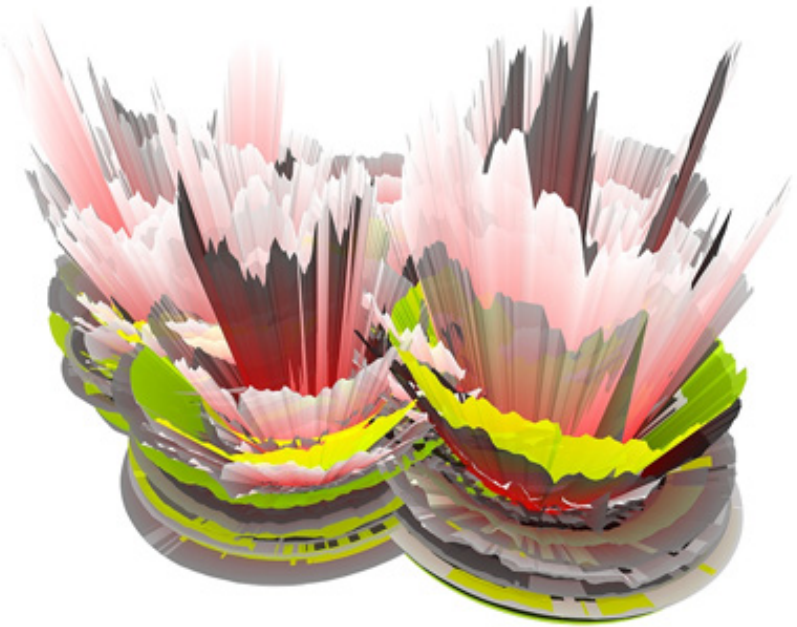
```
> x <-rbeta(1000,2,2)
> plot(density(x))
> kurtosis(x)
[1] 2.1046
```

- As expected we get a negative excess kurtosis (i.e. less than 3) since the distribution has thinner tails (Platykurtic).



Data Visualization

- Data visualization is the transformation of a dataset into a visual format that allows people to identify the characteristics and relationships between examples in the dataset.
- Visualization allows people to recognize patterns or trends based on their judgment or expertise in the particular domain.



Representation

- Representation is understood as the mapping of data into a visual format
- Examples, their attributes and relationships are translated into graphical elements such as points, lines, shapes and colors.
- Examples are usually represented as points.
- Attribute values are represented as the position of the points or the characteristics of the points, e.g. color, size and shape.
- When using position to represent values it is simple to detect if groups of objects are formed or the presence of outliers.

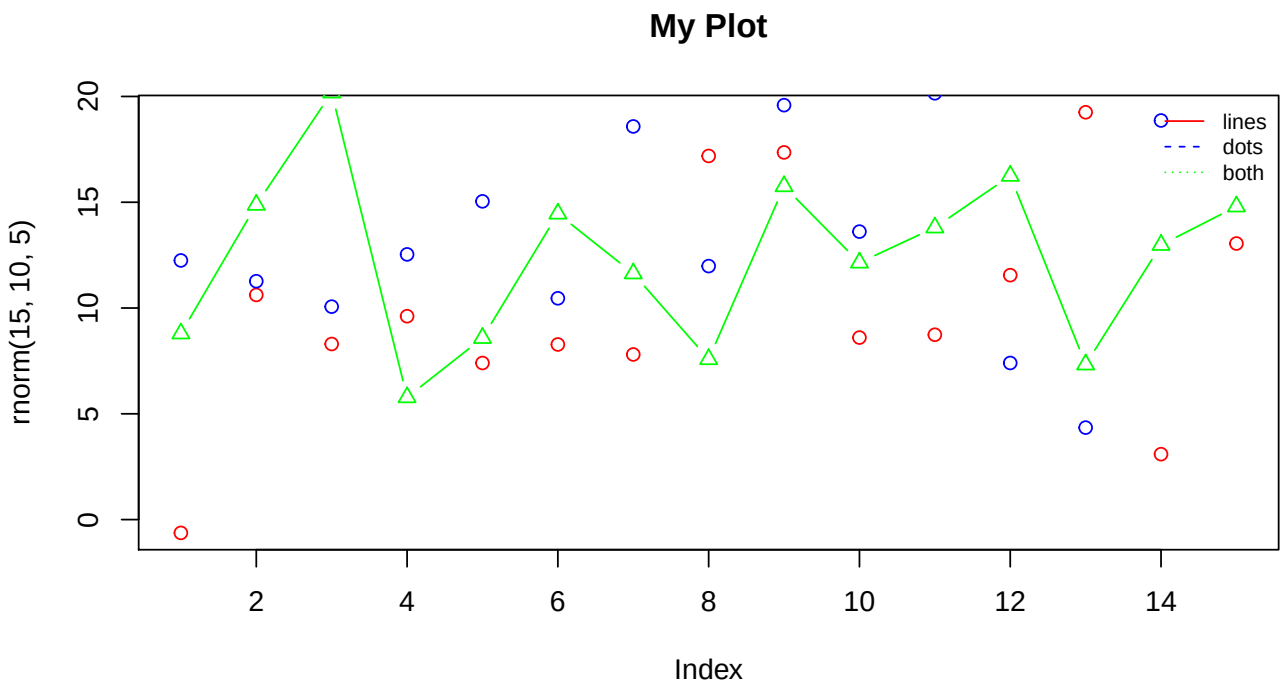
Plotting in R

- In R the most frequent display function is `plot`.
- t is a generic function whose result depends on the nature of the variables given as input.
- To all plots we can add additional parameters such as: `main` for the title, `xlab` and `ylab` for the name of the x-axis and y-axis.
- Other properties are: `col` to define the color, `type` to define the chart type: (p) for points or (l) for lines.
- We can also add new layers to a plot with the command `lines`.
- To save an image to a file we can use Rstudio's **export** button.
- To do it from the R command line:

```
png("image.png")
plot(1:10)
dev.off()
```

Example

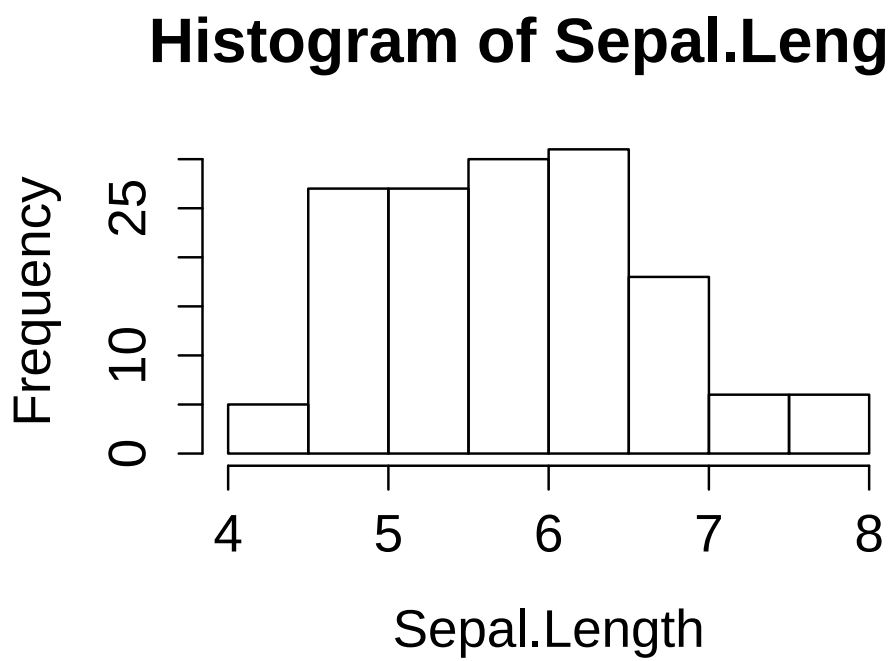
```
plot(rnorm(15,10,5),col="red",type="p",pch=1)
lines(rnorm(15,10,5),col="blue",type="p",pch=1)
lines(rnorm(15,10,5),col="green",type="b",pch=2)
title(main="My Plot")
legend('topright', c("lines","dots","both") ,
       lty=1:3, col=c("red", "blue","green"), bty='n', cex=.75)
```



Histograms

- They show the distribution of the values of a variable.
- The values of the elements are divided into bins and bar charts are created for each of them.
- The height of each bar indicates the number of examples in the corresponding bin.
- In R they are created with the command `hist`.

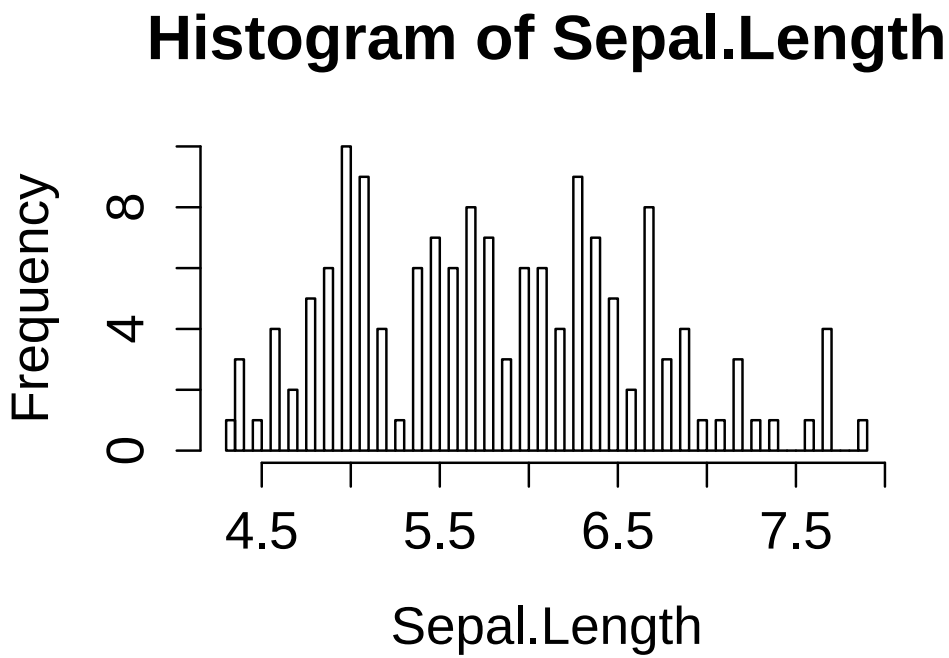
```
> hist(Sepal.Length)
```



Histograms (2)

- The shape of the histogram depends on the number of bins.
- In R this number can be defined with the parameter `nclass`.

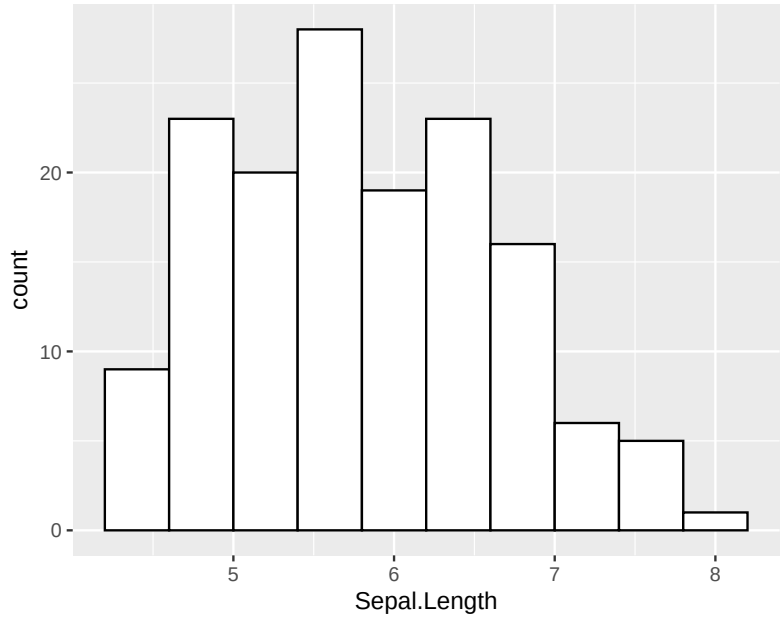
```
> hist(Sepal.Length,nclass=100)
```



Histograms (3)

- A very popular library for making visualizations in R, which is part of tidyverse, is *ggplot2*.
- It is based on the idea of decomposing the plot into semantic components such as scales and layers.

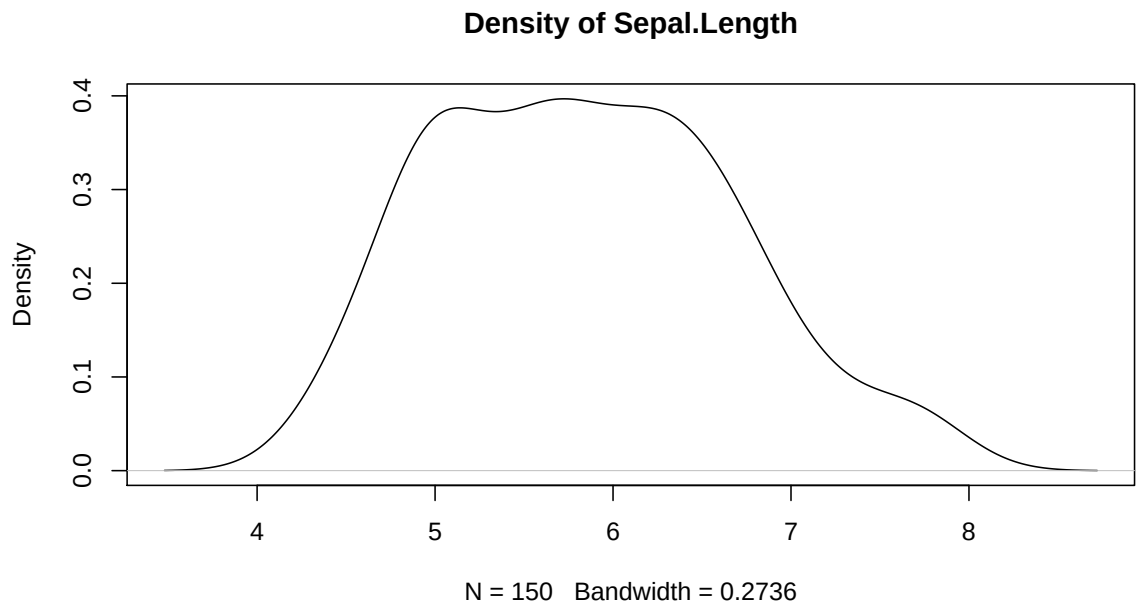
```
>install.packages("ggplot2")
>library(ggplot2)
>ggplot(iris, aes(x=Sepal.Length))
+ geom_histogram(bins = 10, color="black", fill="white")
```



Density

- Another way to visualize how the data are distributed is to estimate a density.
- These are calculated using nonparametric statistical techniques called **kernel** density estimation.
- The density is a smoothed version of the histogram and allows us to determine more clearly if the observed data behaves like a known density e.g., Gaussian.
- In R they are created with the command `density`, and then visualized with the command `plot`.

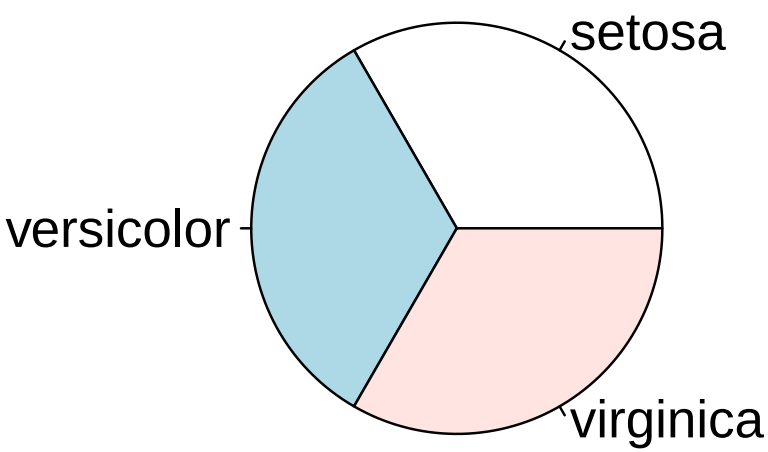
```
plot(density(iris$Sepal.Length),main=""Density of Sepal.Length")
```



Pie Charts

- The pie charts represent the frequency of the elements in a circle.
- Each category has a share proportional to its relative frequency.
- They are generally used for categorical variables:

```
pie(table(iris$Species))
```



- Many people consider pie charts to be misleading and recommend using histograms as a better alternative [Poldrack, 2019].

Boxplots

- Boxplots are built from the percentiles.
- A rectangle is constructed between the first and third quartiles (Q_1 and Q_3).
- The height of the rectangle is the interquartile range IQR ($Q_3 - Q_1$).
- The median is a line that divides the rectangle.
- Each end of the rectangle is extended with a line or arms of length $Q1 - 1.5 * \text{IQR}$ for the lower line and $Q_3 + 1.5 * \text{IQR}$ for the upper line.
- Values more extreme than the length of the arms are considered outliers.
- The boxplot gives us information about the symmetry of the data distribution.
- If the median is not in the center of the rectangle, the distribution is not symmetrical.
- They are useful to detect the presence of outliers.

Boxplots (2)

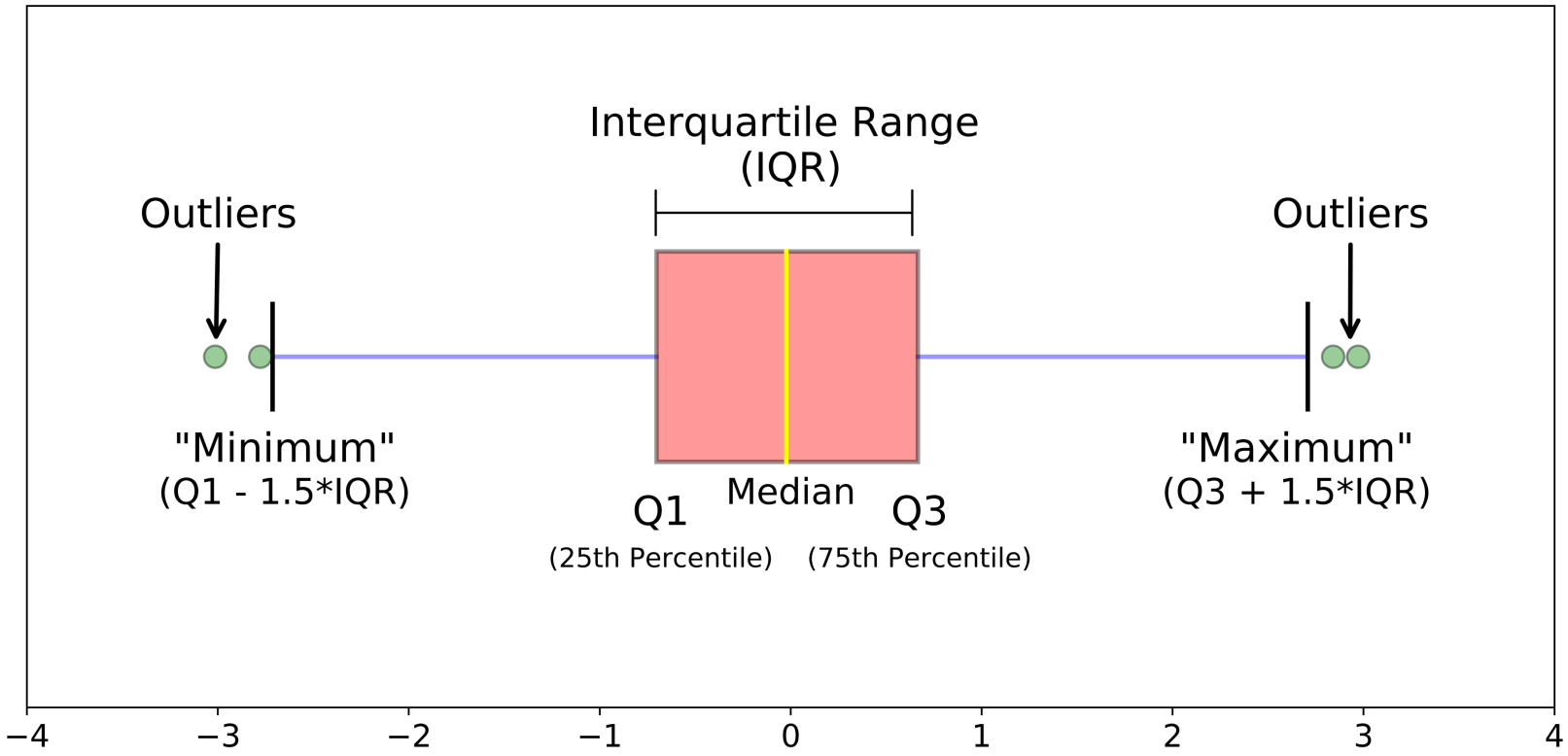
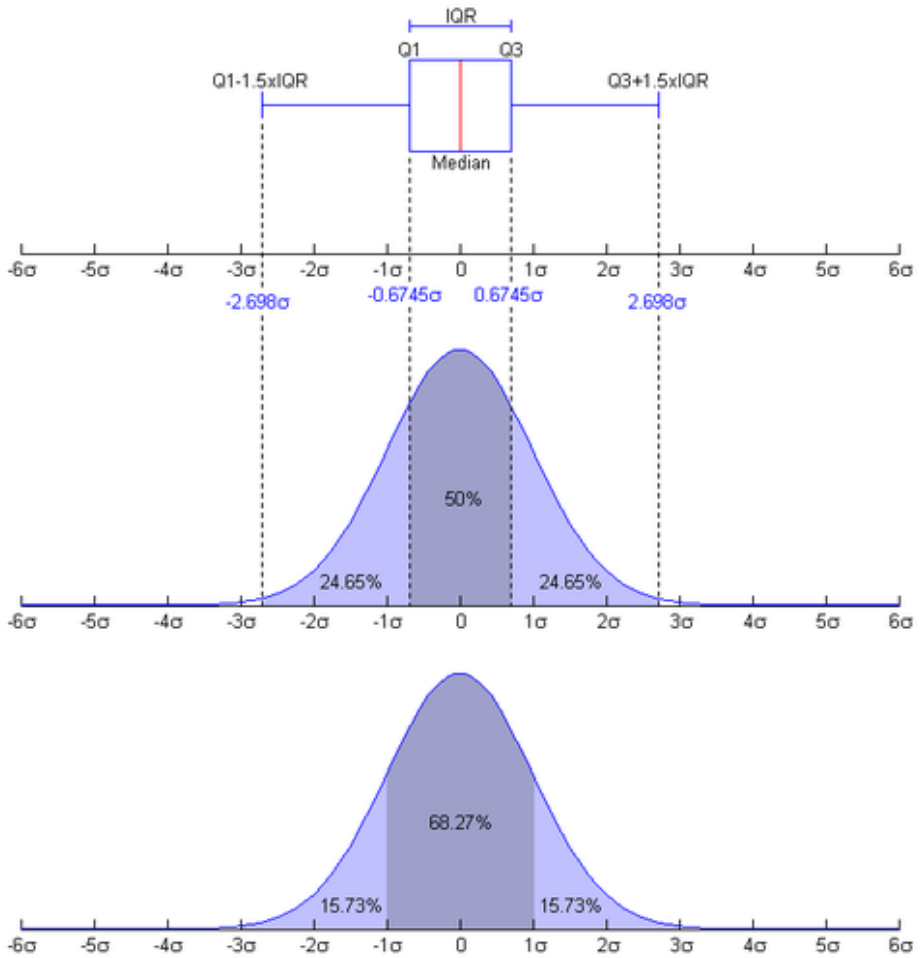


Figure: https://www.laboneconsultoria.com.br/wp-content/uploads/2018/07/Boxplot-04.png

Boxplots (3)

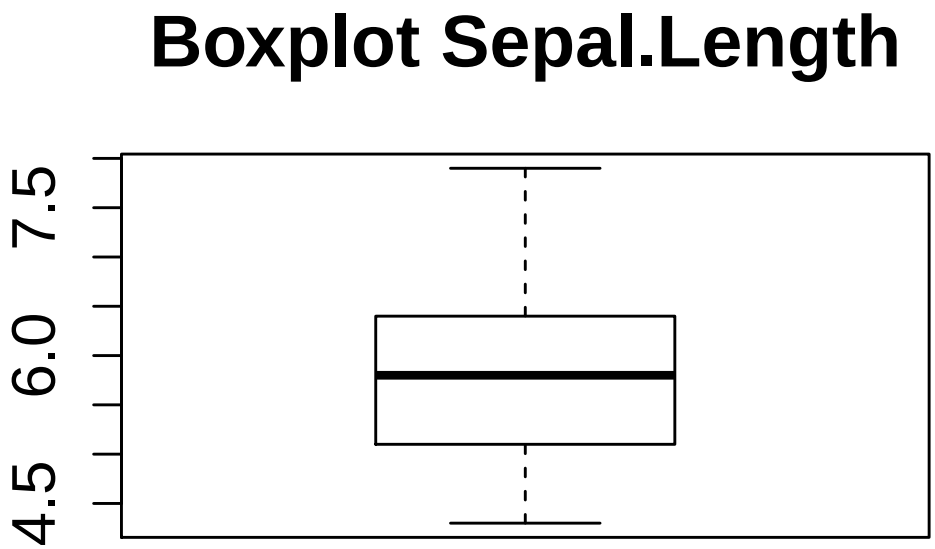
- The length of the arms as well as the criteria for identifying outliers is based on the behavior of a normal distribution.



Boxplots (4)

- In R boxplots are plotted with the command `boxplot`:

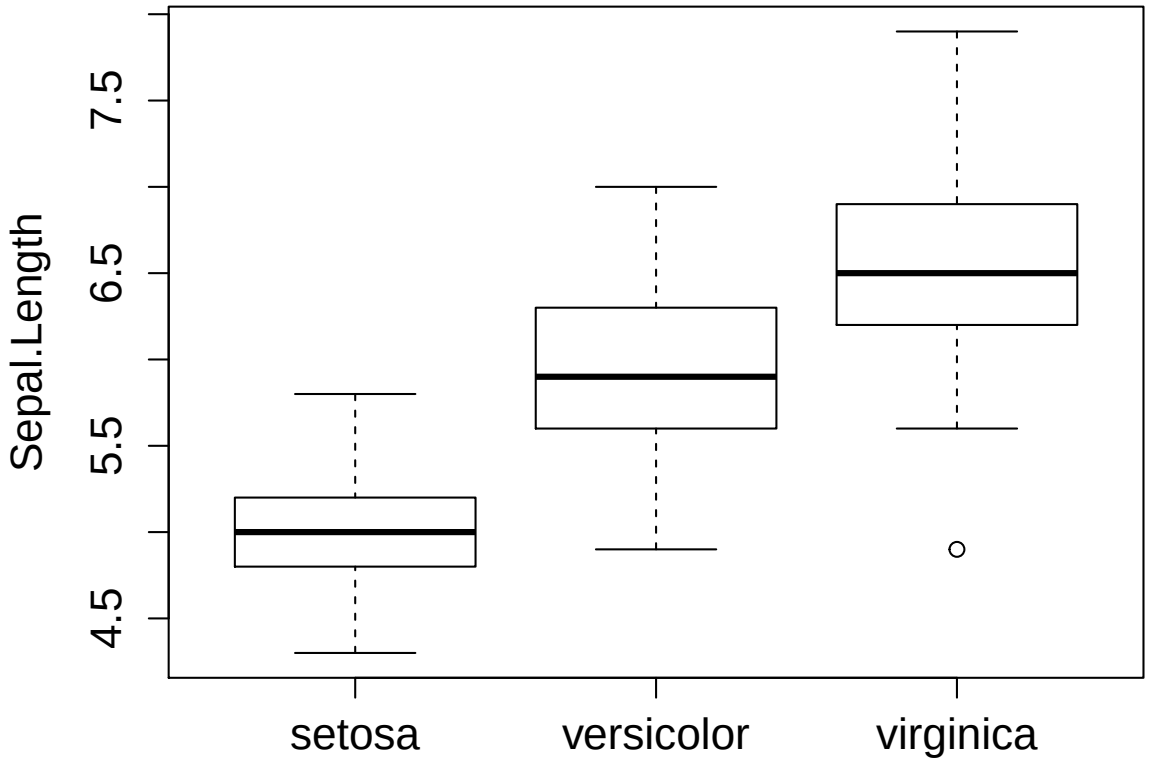
```
> boxplot(Sepal.Length,main="Boxplot Sepal.Length")
```



Boxplots (4)

- If we have a factor variable we can create a boxplot for each category as follows:

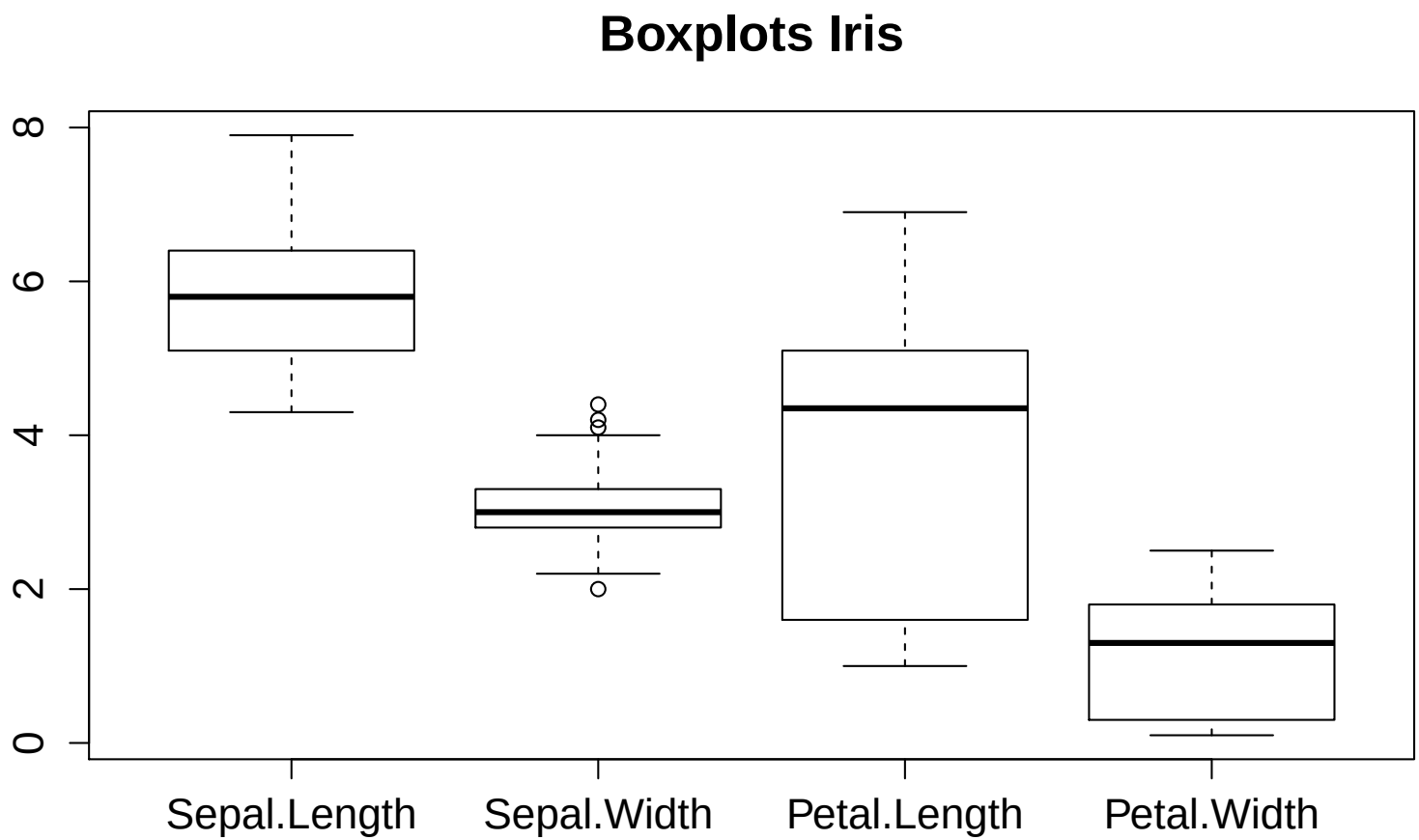
```
> boxplot(Sepal.Length~Species,ylab="Sepal.Length")
```



Boxplots (5)

- We can also compare several boxplots in the same plot:

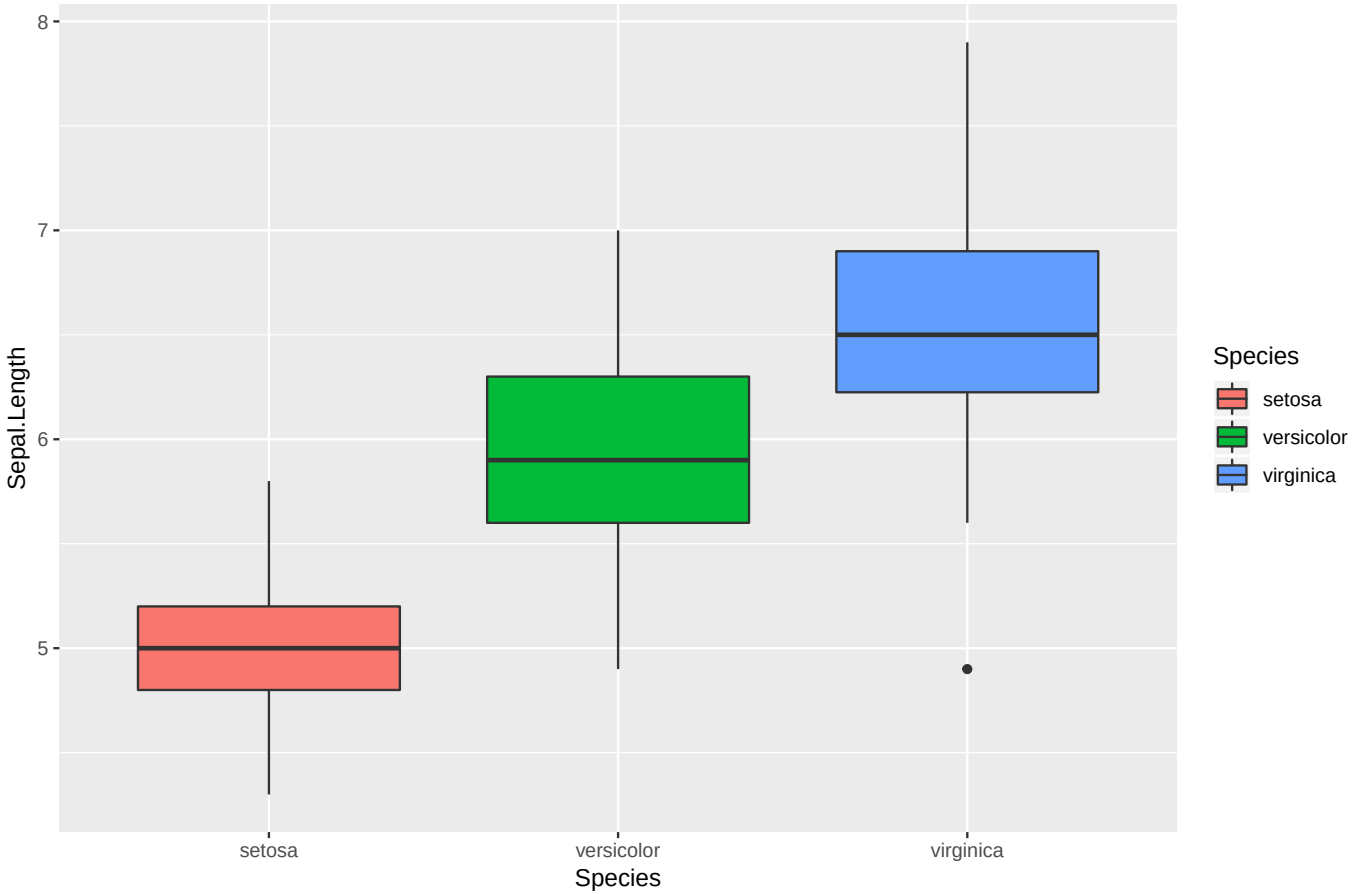
```
> boxplot(x=iris[,1:4],main="Boxplots Iris")
```



Boxplots (6)

- Now using *ggplot2*:

```
> ggplot(iris, aes(x = Species, y = Sepal.Length,
  fill = Species)) +  geom_boxplot()
```



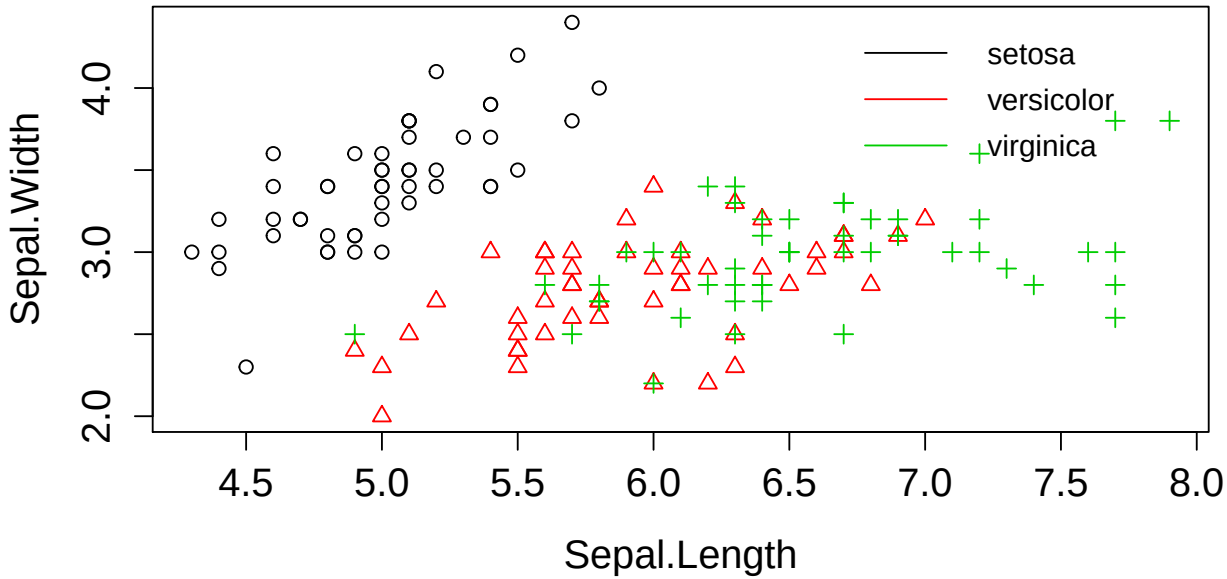
Scatter Plots

- Scatter plots use Cartesian coordinates to display the values of two numerical variables of the same length.
- The values of the attributes determine the position of the elements.
- Other attributes can be used to define the size, shape or color of objects.
- In R we can plot a scatterplot of two numeric variables using the command `plot(x,y)`, which would be y vs x.
- We can also define formulas $f(x) = y$ using the notation `y~x`.
- Thus, the command `plot(y~x)` is equivalent to `plot(x,y)`.
- If we have a data.frame or a numerical matrix, we can see the scatterplots of all pairs using the command `pairs(x)`.

Scatter Plots (2)

- Examples:

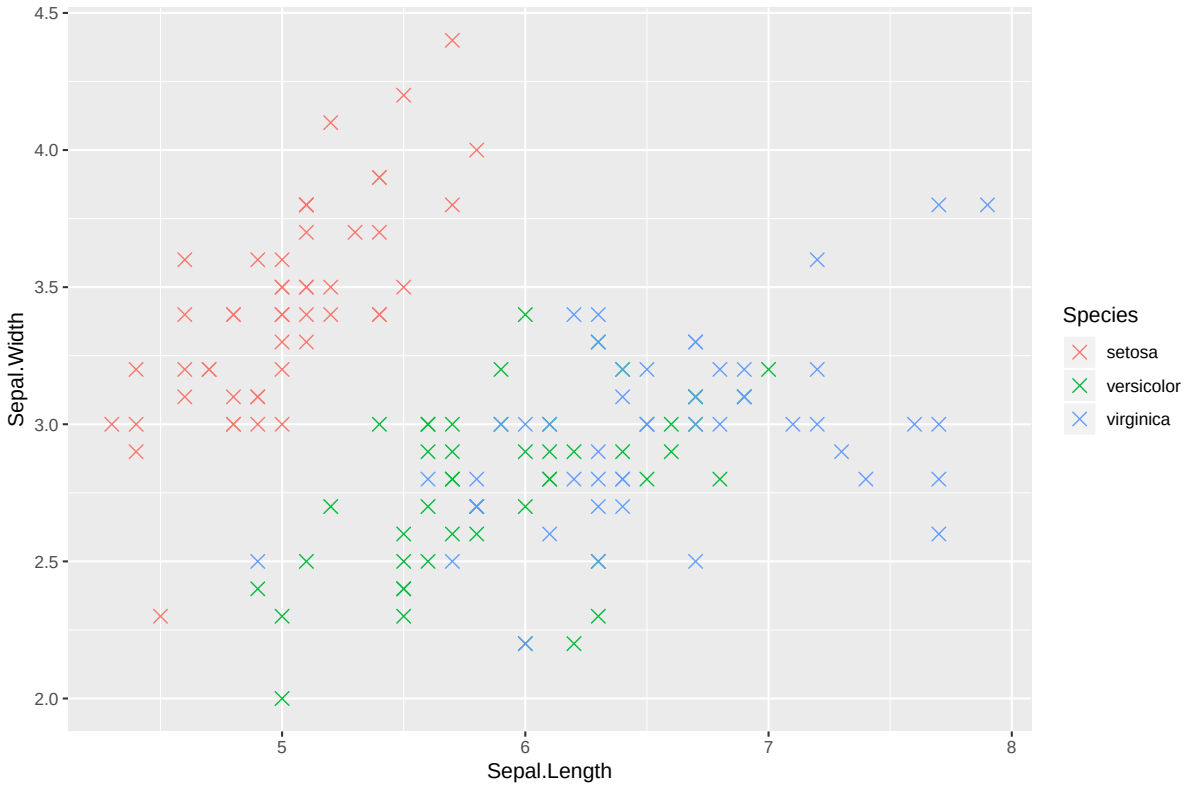
```
# Sepal width vs. sepal length
plot(Sepal.Width~Sepal.Length, col=Species)
# Equivalent
plot(Sepal.Length, Sepal.Width,col=Species,
     pch=as.numeric(Species))
# We add a legend
legend('topright', levels(Species) ,
       lty=1, col=1:3, bty='n', cex=.75)
```



Scatter Plots (3)

- The same using *ggplot2*:

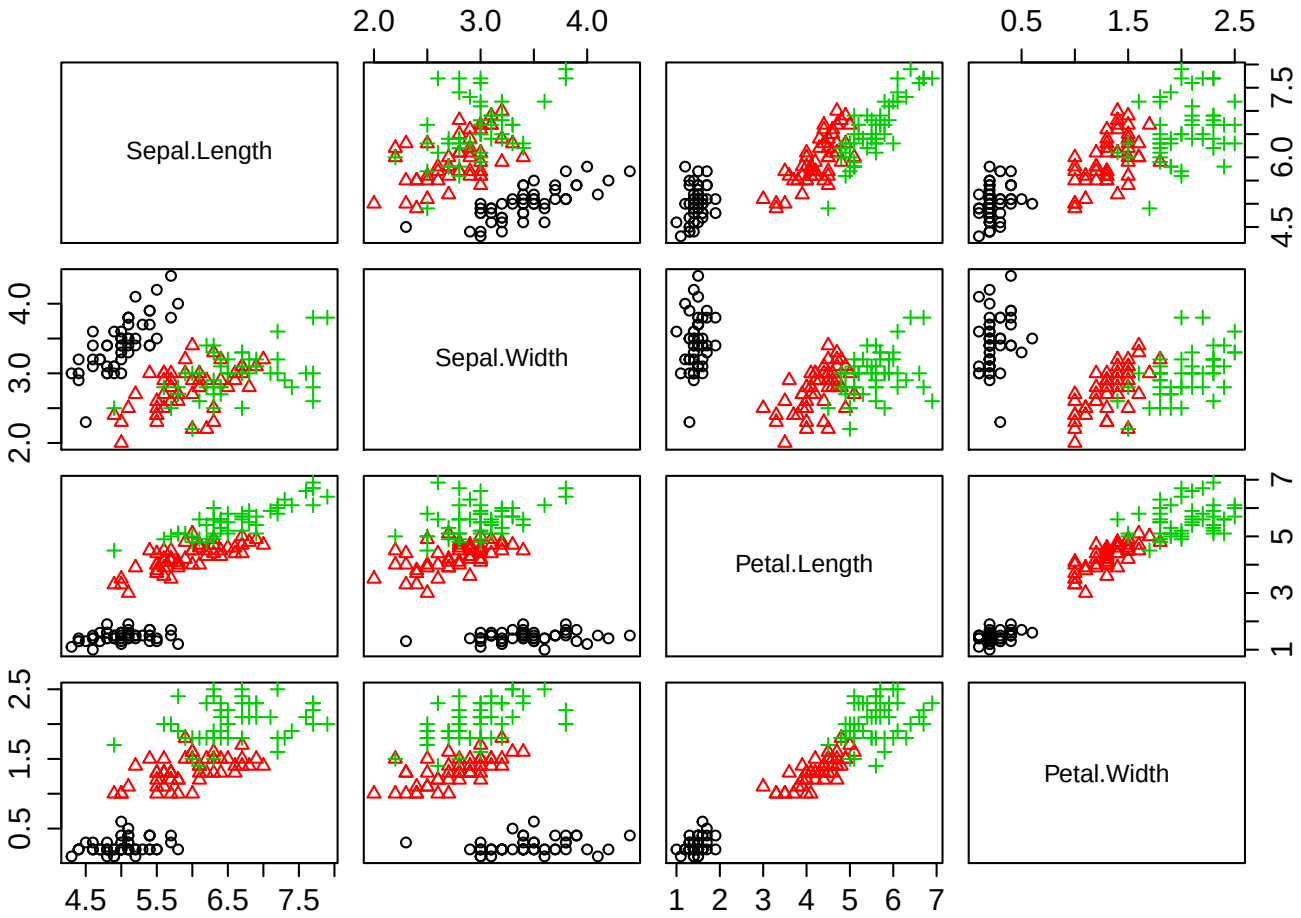
```
ggplot(iris, aes(x=Sepal.Length,
y=Sepal.Width, color=Species)) +
geom_point(size=3,shape=4)
```



Scatter Plots (4)

- All pairs of the 4 variables of the iris dataset using a different color and character for each species:

```
pairs(iris[,1:4],pch=as.numeric(iris$Species),col=iris$Species)
```



Scatter Plots (5)

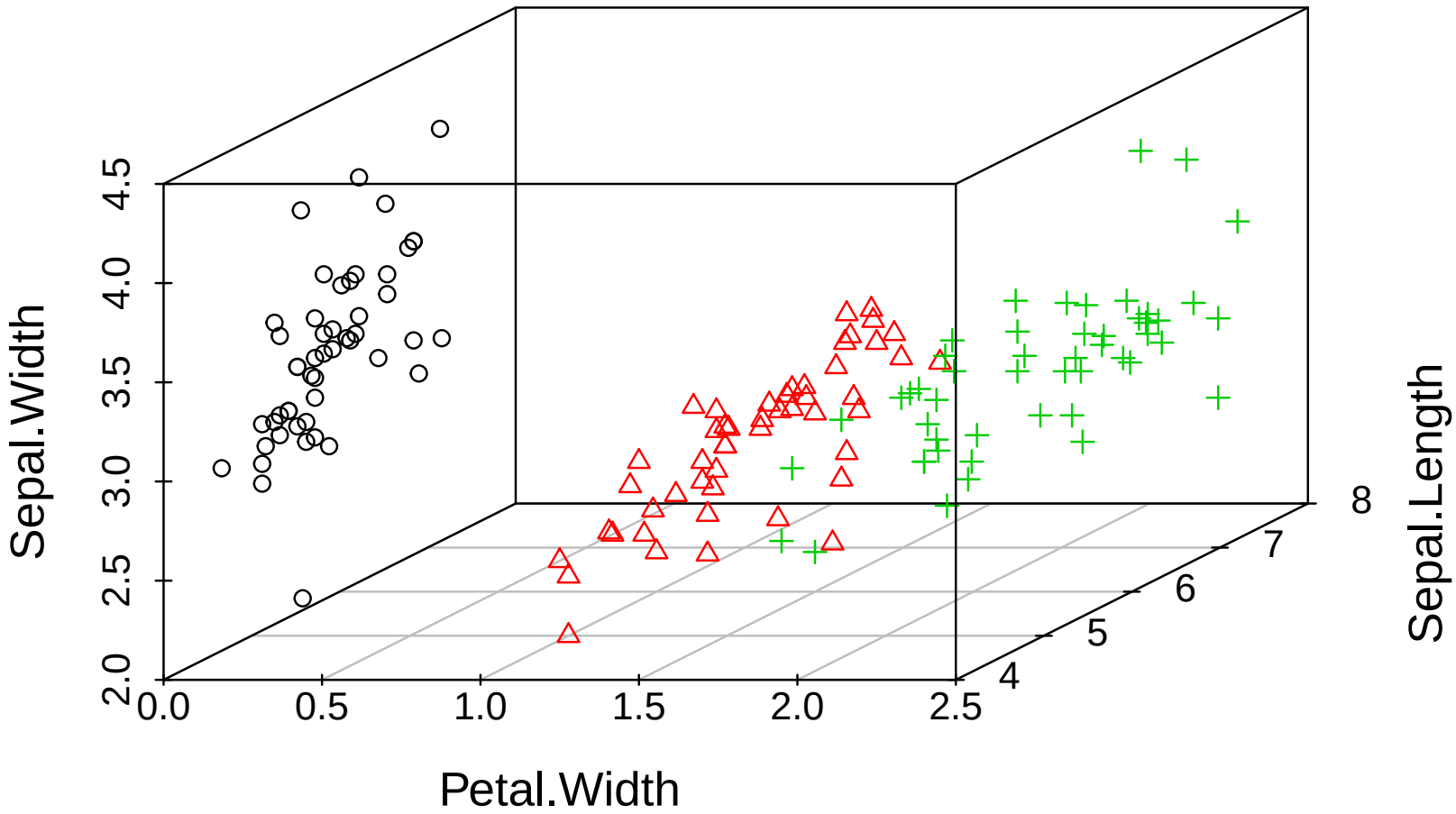
- Three-dimensional scatterplots can also be created.
- The `scatterplot3d` library must be installed using the following command:

```
install.packages("scatterplot3d",dependencies=T)
```

- Then load the library by typing `library(scatterplot3d)`.
- A 3d scatterplot for petal width, sepal length and sepal width:

```
scatterplot3d(iris$Petal.Width, iris$Sepal.Length,
              iris$Sepal.Width, color=as.numeric(iris$Species),
              pch=as.numeric(iris$Species))
```

Scatter Plots (6)



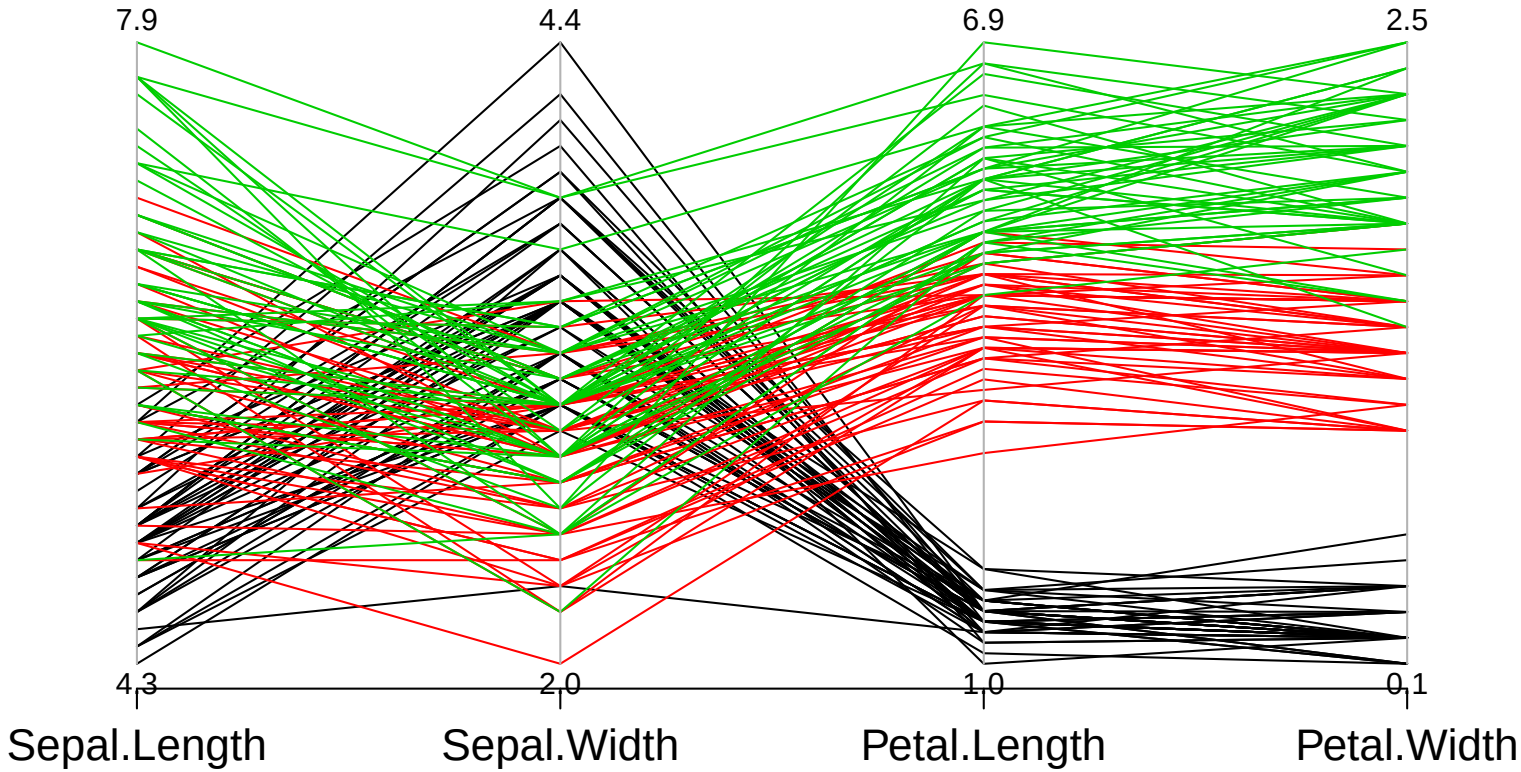
Parallel Coordinate Plots

- Parallel coordinate plots are another way to visualize multi-dimensional data.
- Instead of using perpendicular axes (x-y-z) we use several axes parallel to each other.
- Each attribute is represented by one of the parallel axes with its respective values.
- The values of the different attributes are scaled so that each axis has the same height.
- Each observation represents a line that joins the different axes according to their values.
- In this way, examples similar to each other tend to be grouped in lines with similar trajectory.
- In many occasions it is necessary to re-order the axes in order to visualize a pattern.

Parallel Coordinate Plots (2)

- In R we can create parallel coordinate graphs with the command `parcoord` from the `MASS` library.
- Example:

```
library(MASS)
parcoord(iris[1:4], col=iris$Species,var.label=T)
```



Conclusions

- We have learned how to describe a dataset using various metrics and visualizations.
- These techniques give us a general understanding of the data.

References I


Pipis, G. (2020).
Skewness and kurtosis in statistics: R-bloggers.


Poldrack, R. A. (2019).
Statistical thinking for the 21st century.
https://statsthinking21.org/.


Tan, P.-N., Steinbach, M., and Kumar, V. (2016).
Introduction to data mining.
Pearson Education India.


Westfall, P. H. (2014).
Kurtosis as peakedness, 1905–2014. rip.
The American Statistician, 68(3):191–195.