SMDE FIRST ASSIGNEMENT (20% OF THE FINAL MARK, INDIVIDUAL)

First question: generate a random sample.

On this exercise we are going to start working with probability distributions. The first work to do is to **generate more than 200 observation**s of your selected probability distribution. We are going to generate this using a Spreadsheet.

Now you must import this distribution to R and **test the fitting** of the values with a new sample now generated by R. Use for the fitting a Chi-square test.

Is the sample generated by the Spreadsheet correct?

Test other 5 different distributions and analyze the results (i.e. modify the distribution or the parameters used).

Justify your answers.

Second question: compare three samples.

Generate three populations that follow your specific distribution, but now change one of the parameters (change it following your criteria). As an example, the first is a population that follows an exponential distribution with a parameter λ=10, the second with λ=20, and the third with λ=30.

We want to **analyze using an ANOVA** if these three populations are different (or not) depending on the parameter selected.

Remember to test the ANOVA assumptions. What do you expect on the assumptions?

Analyze and explain the results obtained. Justify your answers.

Third question: define a linear model for an athlete in the 1500 m.

To start: load the package RCmdrPlugin.FactoMinerR.

Load the data “decathlon” located in the package.

The data represents a data frame with observations for different athletes.

What is the linear expression that better predicts the behavior of an athlete for 1500m?

Explore different expressions describing the power and the features of each one of them.

Justify your answers.

Remember to test the assumptions of the linear model.

Fourth question: use the model to predict the behavior of an athlete.

Now use the expression to predict the behavior for a specific athlete. Use the data contained in the table.

Example, if you have a model that only uses X400m as a variable you can construct a new dataframe:

new <- data.frame(X400m=48)

And then use it to predict:

predict(LinearModel.3, newdata=new, interval="prediction")

or

predict(LinearModel.3, newdata=new, interval="confidence")

Remember to test the assumptions of the PCA.

See Annex: The distinction between confidence intervals, prediction intervals and tolerance intervals.

Analyze and explain the results obtained.

Is the model accurate? What do you expect?

Justify your answers.

Five question: PCA

Assuming that the data contained on “decathlon” is huge, use PCA to describe the main variables that exists on the dataset.

What you can conclude analyzing the data? is this **coherent with the previous analysis done?**

Remember to test the assumptions of the PCA.

Justify your answers.

ANNEX: THE DISTINCTION BETWEEN CONFIDENCE INTERVALS, PREDICTION INTERVALS AND TOLERANCE INTERVALS.

When you fit a parameter of a model, the accuracy or precision can be expressed as (i) confidence interval, (ii) prediction interval or (iii) tolerance interval. Assume that the data really are randomly sampled from a Gaussian distribution.

Confidence intervals tell you about how well you have determined the mean. If you do this many times, and calculate a confidence interval of the mean from each sample, you'd expect about 95 % of those intervals to include the true value of the population mean. The key point is that the confidence interval tells you about the likely location of the true population parameter.

Prediction intervals tell you where you can expect to see the next data point sampled. Collect a sample of data and calculate a prediction interval. Then sample one more value from the population. If you do this many times, you'd expect that next value to lie within that prediction interval in 95% of the samples. The key point is that the prediction interval tells you about the distribution of values, not the uncertainty in determining the population mean.

Prediction intervals must account for both the uncertainty in knowing the value of the population mean, plus data scatter. So a prediction interval is always wider than a confidence interval.

The word 'expect' used in defining of a prediction interval means there is a 50% chance that you'd see the value within the interval in more than 95% of the samples, and a 50% chance that you'd see the value within the interval in less than 95% of the samples. As an example doing lots of simulations, you know the true value and thus know if it is in the prediction interval or not. Hence you can then tabulate what fraction of the time the value is enclosed by the interval. On average the obtained value will be 95%, but it might be 92% or 98%. That means that half the time it will be less than 95% and half the time it will be more than 95%.

If you want to be 95% sure that the interval contains 95% of the values, or 91% sure that the interval contains 99% of the values, you need the **tolerance interval**. To compute, or understand, a tolerance interval you have to specify two different percentages: (i) one expresses how sure you want to be, and (ii) other to expresses what fraction of the values the interval will contain. If you set the first value (how sure) to 50%, the tolerance interval is the prediction interval. If you set it to a higher value (say 80% or 99%) then the tolerance interval is wider.

Source: http://cran.r-project.org/web/packages/tolerance/tolerance.pdf

ANNEX: SOME USEFUL R FUNCTIONS

```
#import the data from a csv.

taula.de.dades <-  read.table("MyPath/Random samples.csv",
header=TRUE, sep=";")


#generation of a normal distribution.

v1=rnorm(200, mean=0, sd=1)

summary(v1)


#Work with the data as a dataframe.

taula_v1=data.frame(x1=v1)

#Definition of the intervals, categories to be used.

taula_v1_cat=transform(taula_v1, cat = ifelse(x1 < -1,"-1",

                                    ifelse(x1 < -0.5,"-0.5",

                                    ifelse(x1 < 0,"0",

                                    ifelse(x1 < 0.5,"0.5",

                                    ifelse(x1 <1,"1","Inf"))))))


#Counting the amount of elements in each category “table” function.

taula_freq_v1=as.data.frame(with(taula_v1_cat, table(cat)))


Test=chisq.test(taula_freq, correct=FALSE)

 Test

            Pearson's Chi-squared test

data:  taula_freq
X-squared = 9.7665, df = 5, p-value = 0.08213
```