SMDE FIRST ASSIGMENT

Balbina Virgili Rocosa

27th October 2017

FIRST QUESTION: GENERATE A RANDOM SAMPLE

To start working with probability distributions, first of all, 220 observations have been generated using Spreadsheet.

To do it, next steps have been followed:

1-. 220 random number have been generated

`=RAND()`

2-. The normal distribution of each number generated have been calculated, with mean 0 and standard deviation 1.

`=NORM.INV(AX;0;1)`

3-. The resultant values of the last calculation have been saved on a new document (values_generated.csv).

4-. The observations calculated have been loaded to RStudio

```r
#Load the values generated on the spreadsheet file
genDatasetImported <- read.table("./generated_values.csv", header=FALSE)
#The name of the column has been changed just to make it equal as the other subset
colnames(genDatasetImported) <- c("x1")
```

Afterwards, a new collection of 220 observations has been created now with R (with same mean and standard distribution as before).

```r
#Generation of a normal distribution
genValuesR <- rnorm(220, mean=0, sd=1)
#Work with the data as a dataframe
genDatasetR <- data.frame(x1=genValuesR)
```

Once both samples are ready to be used, the fitting test of the values is performed in order to determine if exists a statistical dependence between the generated distributions or not. The Chi-squared test of independence will be used for it.

To be able to execute the test, the different values need to be classified into defined intervals.

```r
#Definition of the intervals, categories to be used.
tableDatasetImported = transform( genDatasetImported, 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"))))))
#Definition of the intervals, categories to be used.
tableDatasetR = transform( genDatasetR, 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"))))))
```

The results obtained from the previous classification will be checked in order to see the amount of elements on each frequency.

```
#Counting the amount of elements in each category "table" function.
freqTableImp = as.data.frame(with(tableDatasetImported, table(cat)))
```

```
##    cat Freq
## 1 -0.5   46
## 2   -1   32
## 3    0   42
## 4  0.5   46
## 5    1   33
## 6  Inf   21
```

```
freqTableR = as.data.frame(with(tableDatasetR, table(cat)))
```

```
##    cat Freq
## 1 -0.5   37
## 2   -1   30
## 3    0   45
## 4  0.5   44
## 5    1   29
## 6  Inf   35
```

Both generated frequencies are joined in a new table where the data is classified appropriately to perform the Chi-squared test.

```
#Join both calculated frequencies tables into a final one
freqTableFinal = data.frame(x1 = freqTableImp[2], x2 = freqTableR[2])
```

```
##   Freq Freq.1
## 1   46     37
## 2   32     30
## 3   42     45
## 4   46     44
## 5   33     29
## 6   21     35
```

Finally, the Chi-square test has been performed.

```
#Chi-sqaure test
chiSquareTest = chisq.test(freqTableFinal, correct=FALSE)
```

```
##
##  Pearson's Chi-squared test
##
## data:  freqTableFinal
## X-squared = 4.9464, df = 5, p-value = 0.4225
```

With the results obtained, we can appreciate that the p-value calculated is greater than 0.05. Then, the null hypothesis can be accepted and it can be concluded that there is relationship between the two distributions. So, they are not independent and have a similarity. Unfortunately, the level of similarity cannot be calculated with this test.

Now, two other distributions will be analyzed. In this case, two distributions with different mean will be generated and analyzed.

```
#Generate other distributions
genValuesR2 <- rnorm(220, mean=3, sd=3)
genValuesR3 <- rnorm(220, mean=10, sd=4)
```

```r
genDatasetR2 <- data.frame(x1=genValuesR2)
genDatasetR3 <- data.frame(x1=genValuesR3)

tableDatasetR2 = transform( genDatasetR2, 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"))))))
tableDatasetR3 = transform( genDatasetR3, 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"))))))

freqTableR2 = as.data.frame(with(tableDatasetR2, table(cat)))
freqTableR3 = as.data.frame(with(tableDatasetR3, table(cat)))

freqTableFinal2 = data.frame(x1 = freqTableR2[2], x2 = freqTableR3[2])
```

```r
chiSquareTest2 = chisq.test(freqTableFinal2)
```

```
##
##  Pearson's Chi-squared test
##
## data:  freqTableFinal2
## X-squared = 203.18, df = 5, p-value < 2.2e-16
```

With the results obtained, we can appreciate that the p-value calculated is lower than 0.05. Then, the null hypothesis cannot be accepted and it can be concluded that there is no relationship between the two distributions. So, they are independent and have a no similarity.

SECOND QUESTION: ANOVA

In this exercise, the ANOVA test will be used to test whether three populations have similarity or not.

To start with it, three different distributions have been generated. All of them have different mean values.

```r
#Generation of Three different populations with different mean values
v1 = rnorm(200, mean=2, sd=1)
v2 = rnorm(200, mean=5, sd=1)
v3 = rnorm(200, mean=7, sd=1)

#Work with the data as a dataframe
population1 = data.frame(x1=v1, x2="v1")
population2 = data.frame(x1=v2, x2="v2")
population3 = data.frame(x1=v3, x2="v3")
```

ANOVA test is useful to compare the amount of variation among more than two groups in order to determine if exists a statistical dependence between all of them or not.

The three generated frequencies are joined in a new table where the data is classified appropriately to perform the ANOVA test. In the generated table the first column is representing the calculated value and the second one the category.

```
#RcmdrMisc needs to be loaded to execute the following steps
library("RcmdrMisc")
```

```
## Loading required package: car
```

```
## Loading required package: sandwich
```

```
#Merge all generated distributions into a common table
data = mergeRows(population1, population2, common.only=FALSE)
data = mergeRows(as.data.frame(data), population3, common.only=FALSE)
```

Finally, the ANOVA test has been performed.

```
#ANOVA test
anovaModel <- aov(x1 ~ x2, data=data)
```

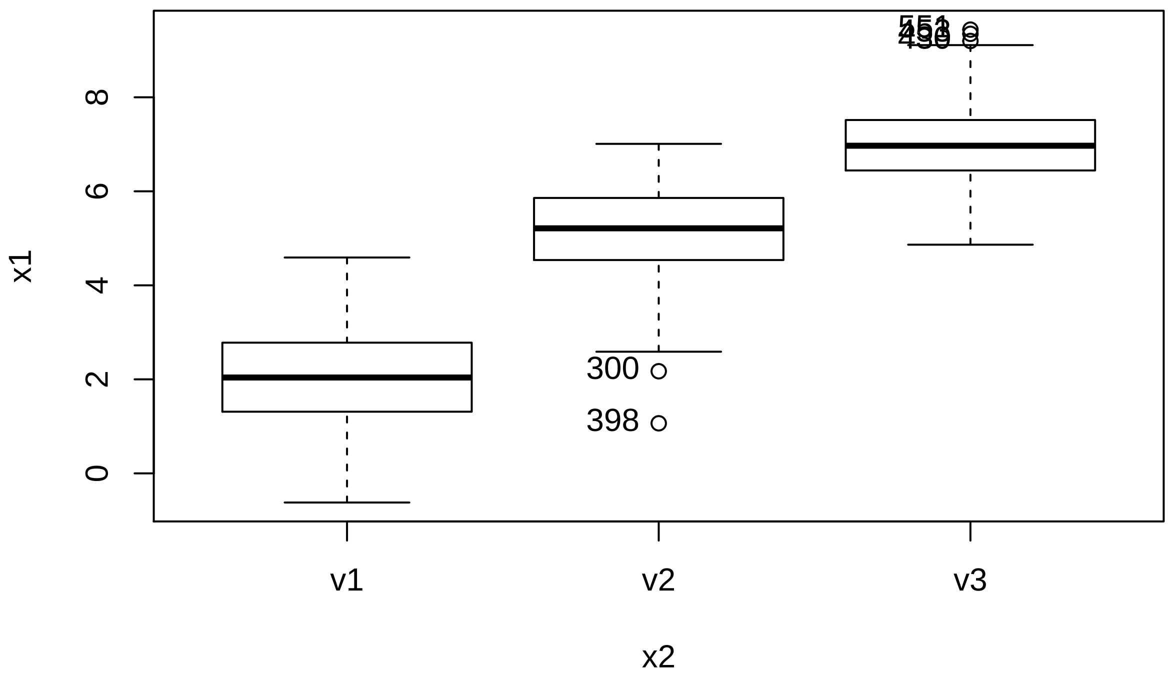
```
##              Df Sum Sq Mean Sq F value Pr(>F)
## x2            2 2544.5    1272    1307 <2e-16 ***
## Residuals   597  580.9       1
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

With the results obtained, we can appreciate that the p-value calculated is lower than 0.05. As the p-value is less than the significance level 0.05, we can conclude that there are significant differences between the three distributed models generated.

Unfortunately, ANOVA test does not allow to check if just some of the distributions are similar to others or not. To see it, different pair test would be performed separately.

Below this lines, a visual plot is showed to easily appreciate the results just calculated. With the graphic visualization can be proved that all three distributions are not similar to each other because their mean values are much different.

```
Boxplot(x1~x2, data=data, id.method="y")
```



```
## [1] "300" "398" "430" "453" "551"
```

To prove the test validity, ANOVA test assumptions must be tested.

1-. Normality test

It is necessary to test that the response is normally distributed with Shapiro-Wilk normality test.

```
#Shapiro-Wilk normality test
shapiro.test(residuals(anovaModel))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  residuals(anovaModel)
## W = 0.99458, p-value = 0.03191
```

The calculated p-value is higher than the significance level 0.05, so there is no indication that normality distribution is violated.

2-. Homogeneity of variance

It is necessary to test that the variance is similar within different groups with Breusch Pagan test.

```
#Breusch Pagan test
lmtest::bptest(anovaModel)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  anovaModel
## BP = 0.43991, df = 2, p-value = 0.8026
```

The calculated p-value is higher than the significance level 0.05, so there is no indication of heteroscedasticity on the calculated model. So we can conclude that there is homogeneity of variance. Otherwise, some transformations to eliminate heteroscedasticity would have been performed.

3-. Independence of variables

It is necessary to test that the data values are independent with Durbin Watson test.

```
#Durbin Watson test
lmtest::dwtest(anovaModel)
```

```
##
##  Durbin-Watson test
##
## data:  anovaModel
## DW = 2.0387, p-value = 0.6529
## alternative hypothesis: true autocorrelation is greater than 0
```

The calculated p-value is higher than the significance level 0, so the alternative hypothesis can be accepted. It means that the variables are not independent and it exits a correlation between them.

Now, a new ANOVA test with a different data will be performed. The data that will be used is geomorphology from FactoMinerR package. In fact, the main objective of this part of the assignment is to analyze if drift is a factor that defines different block sizes or wind effects.

First, the data must be loaded.

```
#FactoMineR needs to be loaded to be able to load the desired data
library("FactoMineR")
```

```r
#Load geomorphology dataset
data(geomorphology)
```

Once the data is loaded, the ANOVA test can be executed. To determine if drift is a factor that defines block sizes or wind effects two different ANOVA tests are performed to compare each desired factor with drift one.

```r
#ANOVA test to compare block sizes to drifts
anovaModelBlockDrift <- aov(Block.size.median ~ Drift, data=geomorphology)
summary(anovaModelBlockDrift)
```

```
##             Df Sum Sq Mean Sq F value Pr(>F)
## Drift        5    644   128.9   0.726  0.606
## Residuals   69  12246   177.5
```

With the results obtained, we can appreciate that the p-value calculated is higher than 0.05. We can conclude that there are not significant differences between both distributions so we can confirm that drift is a factor that defines different block of sizes.

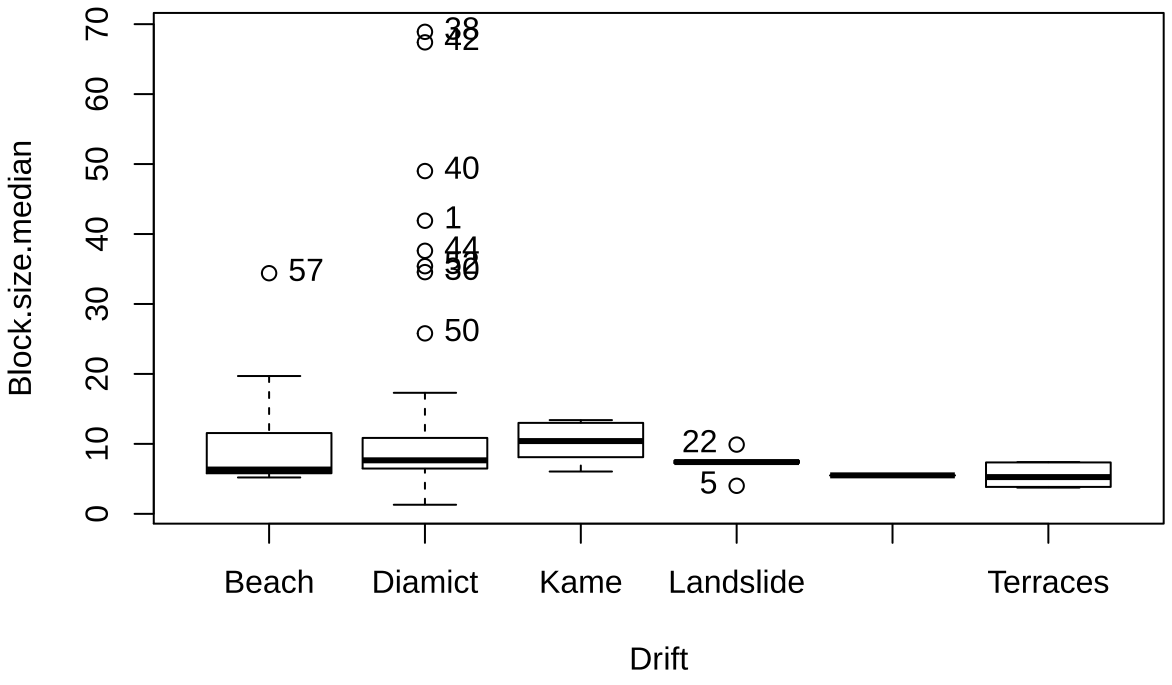
```r
#ANOVA test to compare wind effects to drifts
anovaModelWindDrift <- aov(Wind.effect ~ Drift, data=geomorphology)
summary(anovaModelWindDrift)
```

```
##             Df Sum Sq Mean Sq F value   Pr(>F)
## Drift        5 0.2395 0.04790   10.07 2.86e-07 ***
## Residuals   69 0.3280 0.00475
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

With the results obtained, we can appreciate that the p-value calculated is lower than 0.05. We can conclude that there are significant differences between both distributions so we can confirm that drift is not a factor that defines wind effects.

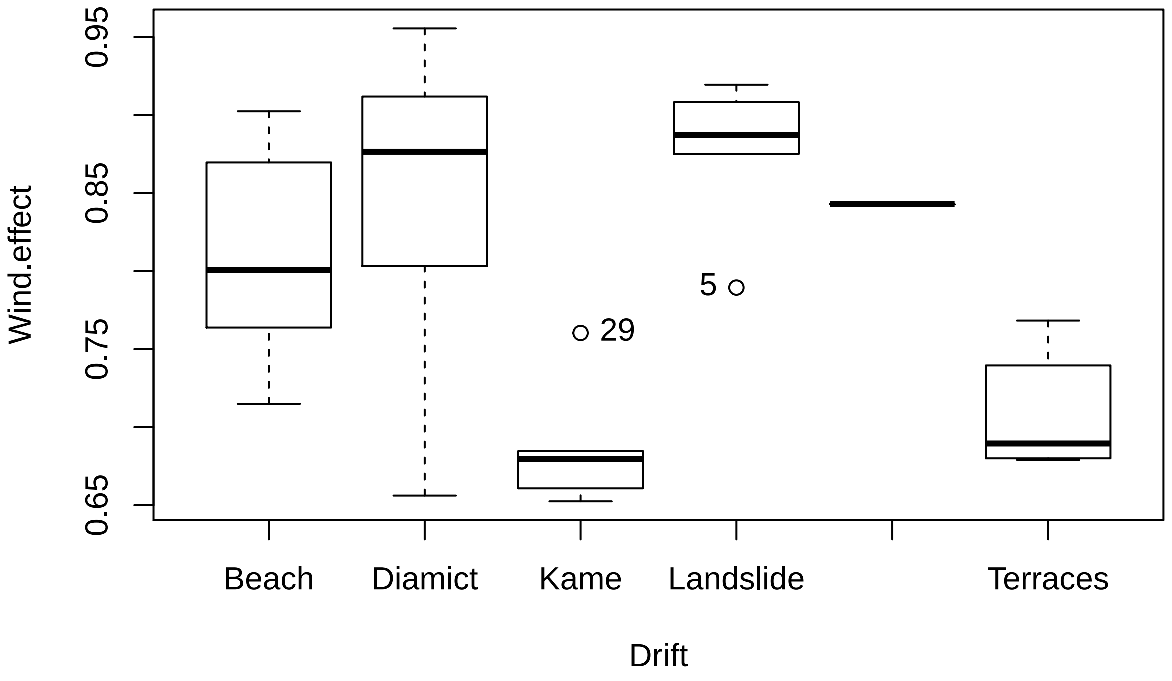
Finally, the plot of each ANOVA test is shown to check that our conclusions are the correct ones from a visual source.

```r
#Boxplot that compare block sizes with drifts
Boxplot(Block.size.median ~ Drift, data=geomorphology, id.method="y")
```



```
##  [1] "57" "1"  "30" "38" "40" "42" "44" "50" "52" "5"  "22"
```

```
#Boxplot that compare wind effects with drifts
Boxplot(Wind.effect ~ Drift, data=geomorphology, id.method="y")
```



```
## [1] "29" "5"
```

The first boxplot shows that all different categories have almost the same mean value with a small value of standard variance. While the other boxplot shows the opposite than before, different categories have different means with significant standard variance. So, we may also confirm this way the similarity between the first two factors. Otherwise, the calculation of the ANOVA test is more precise to determine our assumptions.

THIRD QUESTION: DEFINE A LINEAR MODEL FOR AN ATHLETE IN THE 1500 M

In this exercise, the linear expression that better predicts the behavior of an athlete in the 1500m must be determined. The data that will be used is decathlon from FactoMinerR package.

First of all, the data must be loaded.

```r
#Load geomorphology dataset
data(decathlon)
```

Once the data has been loaded, it is time to start playing with it. For it, first test is performed with the all categories that loaded data contains.

```r
#Generate the linear model with all data loaded
lRegression <- lm(`1500m` ~ ., data=decathlon)
```

```
##
## Call:
## lm(formula = `1500m` ~ ., data = decathlon)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.11082 -0.40320 -0.02978  0.37054  0.99005
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)          1.411e+03  2.281e+01  61.851   <2e-16 ***
## `100m`              -3.711e+01  6.090e-01 -60.937   <2e-16 ***
## Long.jump            3.947e+01  6.635e-01  59.489   <2e-16 ***
## Shot.put             9.984e+00  2.172e-01  45.972   <2e-16 ***
## High.jump            1.470e+02  2.538e+00  57.931   <2e-16 ***
## `400m`              -7.680e+00  2.351e-01 -32.670   <2e-16 ***
## `110m.hurdle`       -1.951e+01  3.730e-01 -52.320   <2e-16 ***
## Discus               3.461e+00  5.077e-02  68.184   <2e-16 ***
## Pole.vault           4.855e+01  6.462e-01  75.127   <2e-16 ***
## Javeline             2.442e+00  4.648e-02  52.533   <2e-16 ***
## Rank                 3.099e-04  4.278e-02   0.007    0.994
## Points              -1.632e-01  2.566e-03 -63.589   <2e-16 ***
## CompetitionOlympicG  1.387e-01  5.532e-01   0.251    0.804
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5796 on 28 degrees of freedom
## Multiple R-squared:  0.9983, Adjusted R-squared:  0.9975
## F-statistic:  1350 on 12 and 28 DF,  p-value: < 2.2e-16
```

With the results retrieved, we can see that almost all values, except Rank and Competition, have a p-value lower than 0.05 and the t-value much higher or much lower than 0. So we can determine that all categories, except Rank and Competition, have a relation with the 1500m distribution.

Otherwise, the results show that Rank and Competition are not much related with 1500m distribution so both of them can be discarded from the linear model.

Now, a new linear regression model is calculated discarding Rank and Competition distributions.

```r
#Generate the linear model without discarted distributions
lRegression2 <- update(lRegression, . ~ . - Rank - Competition)
```

```
##
## Call:
## lm(formula = `1500m` ~ `100m` + Long.jump + Shot.put + High.jump +
##     `400m` + `110m.hurdle` + Discus + Pole.vault + Javeline +
##     Points, data = decathlon)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.04641 -0.38236 -0.07186  0.39878  0.94814
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)    1.409e+03  1.788e+01   78.81   <2e-16 ***
## `100m`        -3.721e+01  5.541e-01  -67.15   <2e-16 ***
## Long.jump      3.934e+01  5.699e-01   69.03   <2e-16 ***
## Shot.put       9.983e+00  2.074e-01   48.13   <2e-16 ***
## High.jump      1.464e+02  2.090e+00   70.08   <2e-16 ***
## `400m`        -7.627e+00  1.926e-01  -39.60   <2e-16 ***
## `110m.hurdle` -1.947e+01  3.382e-01  -57.57   <2e-16 ***
## Discus         3.451e+00  4.311e-02   80.05   <2e-16 ***
## Pole.vault     4.839e+01  5.317e-01   91.00   <2e-16 ***
## Javeline       2.434e+00  3.966e-02   61.38   <2e-16 ***
## Points        -1.627e-01  1.789e-03  -90.99   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5623 on 30 degrees of freedom
## Multiple R-squared:  0.9983, Adjusted R-squared:  0.9977
## F-statistic:  1721 on 10 and 30 DF,  p-value: < 2.2e-16
```

With the results retrieved, we can see that all values have a p-value lower than 0.05 and the t-value much higher or much lower than 0. So we could determine that all categories have a relation with the 1500m distribution.

In this case, the R-squared value is 0.9975 so the probability to predict the 1500m distribution in this calculated linear model can be considered as very good one.

Unfortunately, analyzing all different values used, we can determine that Points must not be used to create the linear model because it is a value calculated from the undetermined value. So, it does not make sense to use this category to predict the 1500m value.

So, a new linear regression model is calculated discarding Points distribution.

```r
#Generate the linear model without discarted distribution
lRegression3 <- update(lRegression2, . ~ . - Points)
```

```
##
## Call:
## lm(formula = `1500m` ~ `100m` + Long.jump + Shot.put + High.jump +
##     `400m` + `110m.hurdle` + Discus + Pole.vault + Javeline,
##     data = decathlon)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -20.015  -4.692  -1.217   4.821  20.968
##
## Coefficients:
```

```
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)    -5.089196 144.658704  -0.035 0.972161
## `100m`        -14.026804   8.055565  -1.741 0.091556 .
## Long.jump       4.363645   6.888604   0.633 0.531080
## Shot.put       -0.712440   2.797866  -0.255 0.800685
## High.jump      -6.663147  20.281067  -0.329 0.744712
## `400m`          7.056379   1.721604   4.099 0.000277 ***
## `110m.hurdle`   0.002113   4.287265   0.000 0.999610
## Discus          1.328956   0.593554   2.239 0.032474 *
## Pole.vault     12.480774   5.834140   2.139 0.040398 *
## Javeline       -0.656775   0.335114  -1.960 0.059056 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.206 on 31 degrees of freedom
## Multiple R-squared:  0.518,  Adjusted R-squared:  0.378
## F-statistic: 3.701 on 9 and 31 DF,  p-value: 0.003022
```

With the new results retrieved, we can see that dependencies have changed. Now, we can determine that 400m, Discus, Pole.vault, Discus, Javeline and 400m have a relation with 1500m distribution. The one that seems to be more related to the 1500m distribution is 400m.

In this case, the R-squared value is 0.378 so the probability to predict the 1500m distribution in this calculated linear model has decreased significantly from the previous calculated model.

To determine that our assumptions are correct, a new linear regression model is calculated discarding all distributions that have been considered as not relationated.

```
#Generate the linear model without discarted distributions
lRegression4 <- lm(formula = `1500m` ~ `100m` + `400m` + Pole.vault + Discus +
                     Javeline, data = decathlon)
```

```
##
## Call:
## lm(formula = `1500m` ~ `100m` + `400m` + Pole.vault + Discus +
##     Javeline, data = decathlon)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -20.831  -3.978  -1.043   4.421  20.784
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  40.5049    85.1779   0.476  0.63736
## `100m`      -15.1377     6.3870  -2.370  0.02343 *
## `400m`        6.6108     1.4152   4.671 4.31e-05 ***
## Pole.vault   13.2420     5.0826   2.605  0.01339 *
## Discus        1.2150     0.4304   2.823  0.00779 **
## Javeline     -0.6844     0.2947  -2.323  0.02613 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.761 on 35 degrees of freedom
## Multiple R-squared:  0.5071, Adjusted R-squared:  0.4367
## F-statistic: 7.201 on 5 and 35 DF,  p-value: 0.0001001
```

With the new results retrieved, we can see that dependencies are still the same as the ones calculated on the previous model. But now, the R-squared value is 0.4367. So the probability to predict the 1500m distribution in this calculated linear model has increased from the value calculated during the previous model, where much more information was used.

Finally, just to take into account all useful possibilities, a new linear regression model is calculated using just the 400m distribution, which seems to be the most related one from the results retrieved on the previous linear models.

```
#Generate the linear model without discarted distributions
lRegression5 <- lm(formula = `1500m` ~ `400m`, data = decathlon)
```

```
##
## Call:
## lm(formula = `1500m` ~ `400m`, data = decathlon)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -19.0877  -6.9098  -0.7062   4.7360  31.5996
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   74.102     73.424   1.009  0.31909
## `400m`         4.130      1.479   2.792  0.00808 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10.79 on 39 degrees of freedom
## Multiple R-squared:  0.1666, Adjusted R-squared:  0.1452
## F-statistic: 7.793 on 1 and 39 DF,  p-value: 0.008078
```

With the new results retrieved, we can see that dependencies have changed a little bit and the R-squared value is 0.1452. So the probability to predict the 1500m distribution in this calculated linear model has decreased almost a 20% from the value calculated during the previous model, where much more information was used.

To conclude, it can be determined that the best linear model to predict the behavior of an athlete for 1500m is the one that takes into account all the following variables:

- 400m
- Pole.vault
- Javeline
- Discus
- 100m

Finally, Linear model test assumptions must be calculated to prove the test validity.

1-. Normality test

It is necessary to test that the response is normally distributed with Shapiro-Wilk normality test.

```
#Shapiro-Wilk normality test
shapiro.test(residuals(lRegression4))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  residuals(lRegression4)
## W = 0.98581, p-value = 0.8808
```

The calculated p-value is higher than the significance level 0.05, so there is no indication that normality distribution is violated.

2-. Homogeneity of variance

It is necessary to test that the variance is similar within different groups with Breusch Pagan test.

```
#Breusch Pagan test
lmtest::bptest(lRegression4)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  lRegression4
## BP = 8.8268, df = 5, p-value = 0.1162
```

The calculated p-value is higher than the significance level 0.05, so there is no indication of heteroscedasticity on the calculated model. So we can conclude that there is homogeneity of variance. Otherwise, some transformations to eliminate heteroscedasticity would have been performed.

3-. Independence of variables

It is necessary to test that the data values are independent with Durbin Watson test.

```
#Durbin Watson test
lmtest::dwtest(lRegression4)
```

```
##
##  Durbin-Watson test
##
## data:  lRegression4
## DW = 1.4115, p-value = 0.01718
## alternative hypothesis: true autocorrelation is greater than 0
```

The calculated p-value is higher than the significance level 0, so the alternative hypothesis can be accepted. It means that the variables are not independent and it exits a correlation between them.

FOURTH QUESTION: USE THE MODEL TO PREDICT THE BEHAVIOR OF AN ATHLETE

In this exercise, the behavior of an athlete wants to be predicted. It can be done thanks to the linear model calculated on the previous exercise. To do it, decathlon data will be tested.

Firstly, the decathlon data is separated by competition into different variables. This way, the initial data will be divided into training and test data. In this prediction the data obtained from Decastar competition is going to be used as training data and the one obtained from OlympicG is the one that will be used to test our prediction results.

```
#Two new subsets are created with the decathlon data separated by competition
dataDecastar <- subset(decathlon, Competition=='Decastar')
dataOlympicG <- subset(decathlon, Competition=='OlympicG')
```

So the linear model calculated on the previous exercise is again executed but just using the training data.

```
#Generate the linear model with training data
lRegression6 <- lm(formula = `1500m` ~ `100m` + `400m` + Pole.vault, data = dataDecastar)
```

```
##
## Call:
## lm(formula = `1500m` ~ `100m` + `400m` + Pole.vault, data = dataDecastar)
```

```
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -12.421  -3.943  -1.780   2.795  17.172
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  381.959    154.656   2.470  0.03559 *
## `100m`       -40.186     11.492  -3.497  0.00676 **
## `400m`         5.476      3.138   1.745  0.11495
## Pole.vault    16.067     10.022   1.603  0.14335
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.667 on 9 degrees of freedom
## Multiple R-squared:  0.6239, Adjusted R-squared:  0.4986
## F-statistic: 4.977 on 3 and 9 DF,  p-value: 0.02638
```

With the new results retrieved of this new linear model, we can see that R-squared value is 0.5464 so the probability of prediction is above 50% which is considered the minimum percentage accepted on athleticism.

Two different predictions can be performed.

1-. The first one is to predict the behavior of the athletes during the next race. 2-. The second one is to predict the behavior of the athletes during the following races.

As the dataset we are using is not very complete because it does not have lots of data, just the first prediction test will be performed. So, the prediction interval has been calculated using the data sample collected.

```
#Predict the behavior of the athletes with prediction interval
predictedData <- predict(lRegression4, newdata=dataOlympicG, interval="prediction")
```

```
##                  fit      lwr      upr
## Sebrle      273.0930 253.0748 293.1111
## Clay        285.7053 265.5114 305.8992
## Karpov      276.6603 256.4152 296.9055
## Macey       276.3674 257.7128 295.0220
## Warners     276.9651 258.2059 295.7244
## Zsivoczky   276.1599 257.8867 294.4330
## Hernu       274.9476 256.7853 293.1099
## Nool        280.3077 260.9544 299.6611
## Bernard     278.2743 259.3566 297.1921
## Schwarzl    283.7812 265.3860 302.1765
## Pogorelov   294.3210 275.4616 313.1805
## Schoenbeck  286.5156 268.0967 304.9345
## Barras      269.7074 251.0662 288.3486
## Smith       275.0395 255.8271 294.2520
## Averyanov   284.1941 264.4627 303.9255
## Ojaniemi    272.9313 253.9383 291.9243
## Smirnov     272.4774 254.2211 290.7337
## Qi          274.1188 255.8486 292.3889
## Drews       276.3185 257.3087 295.3283
## Parkhomenko 278.7023 259.7072 297.6974
## Terek       293.7927 274.2821 313.3034
## Gomez       260.5929 241.4373 279.7484
## Turi        285.6955 266.5427 304.8483
## Lorenzo     267.1617 248.4769 285.8465
## Karlivans   279.1269 260.4148 297.8390
## Korkizoglou 296.2165 276.9115 315.5216
## Uldal       278.1043 259.5713 296.6374
## Casarsa     297.4954 276.8127 318.1782
```

With the predict function we obtain the ranges within the data of the following race must be determined according to our designed model.

So, we are going to check if the predict function is accurated or not. To do it, we are going to determine if the data from OlympicsG competition actually lies between the ranges calculated.

```
#Determine if the data of OlympicsG competition lies between the rangs calculated
#TRUE if fits into the rang, FALSE OTHERWISE
compData <- c()
predictedDataFrame <- data.frame(x1=predictedData)
dataOlympicGFrame <- data.frame(x1=dataOlympicG)
for(x in 1:nrow(predictedDataFrame)) {
  if( dataOlympicGFrame$x1.1500m[x] >= predictedDataFrame$x1.lwr[x] &&
      dataOlympicGFrame$x1.1500m[x] <= predictedDataFrame$x1.upr[x]){
    compData[x] <- 'TRUE'
  }else{
    compData[x] <- 'FALSE'
  }
}

predictedDataFrame["prediction"] <- compData
```

```
##               x1.fit   x1.lwr   x1.upr prediction
## Sebrle      273.0930 253.0748 293.1111       TRUE
## Clay        285.7053 265.5114 305.8992       TRUE
## Karpov      276.6603 256.4152 296.9055       TRUE
## Macey       276.3674 257.7128 295.0220       TRUE
## Warners     276.9651 258.2059 295.7244       TRUE
## Zsivoczky   276.1599 257.8867 294.4330       TRUE
## Hernu       274.9476 256.7853 293.1099       TRUE
## Nool        280.3077 260.9544 299.6611       TRUE
## Bernard     278.2743 259.3566 297.1921       TRUE
## Schwarzl    283.7812 265.3860 302.1765       TRUE
## Pogorelov   294.3210 275.4616 313.1805       TRUE
## Schoenbeck  286.5156 268.0967 304.9345       TRUE
## Barras      269.7074 251.0662 288.3486       TRUE
## Smith       275.0395 255.8271 294.2520       TRUE
## Averyanov   284.1941 264.4627 303.9255       TRUE
## Ojaniemi    272.9313 253.9383 291.9243       TRUE
## Smirnov     272.4774 254.2211 290.7337       TRUE
## Qi          274.1188 255.8486 292.3889       TRUE
## Drews       276.3185 257.3087 295.3283       TRUE
## Parkhomenko 278.7023 259.7072 297.6974       TRUE
## Terek       293.7927 274.2821 313.3034       TRUE
## Gomez       260.5929 241.4373 279.7484       TRUE
## Turi        285.6955 266.5427 304.8483       TRUE
## Lorenzo     267.1617 248.4769 285.8465       TRUE
## Karlivans   279.1269 260.4148 297.8390       TRUE
## Korkizoglou 296.2165 276.9115 315.5216      FALSE
## Uldal       278.1043 259.5713 296.6374       TRUE
## Casarsa     297.4954 276.8127 318.1782       TRUE
```

As it was expected, we have obtained 27 well-predicted values and 1 wrong-predicted one. Then, the prediction interval obtained is 96’4% and can be confirmed that the values obtained lie within the prediction interval of 95% of the samples so it can be determined that the model designed during the previous exercise is an accurate one.

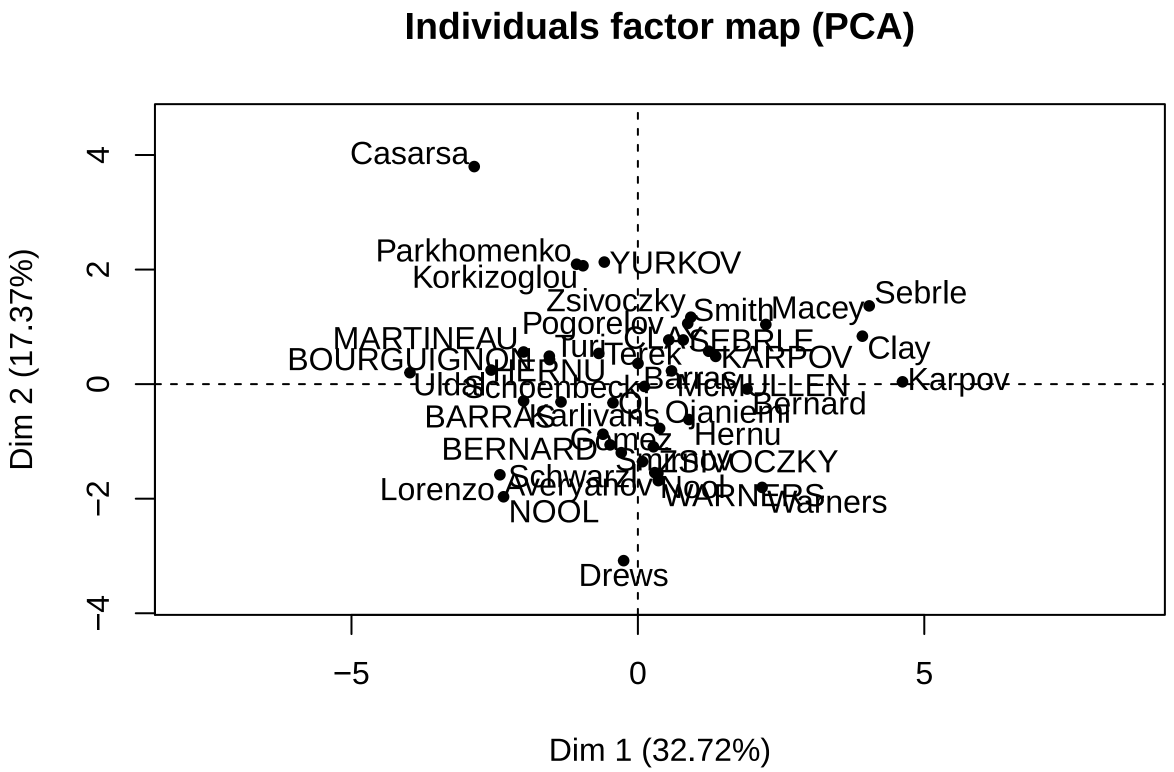
FIVE QUESTION: PCA

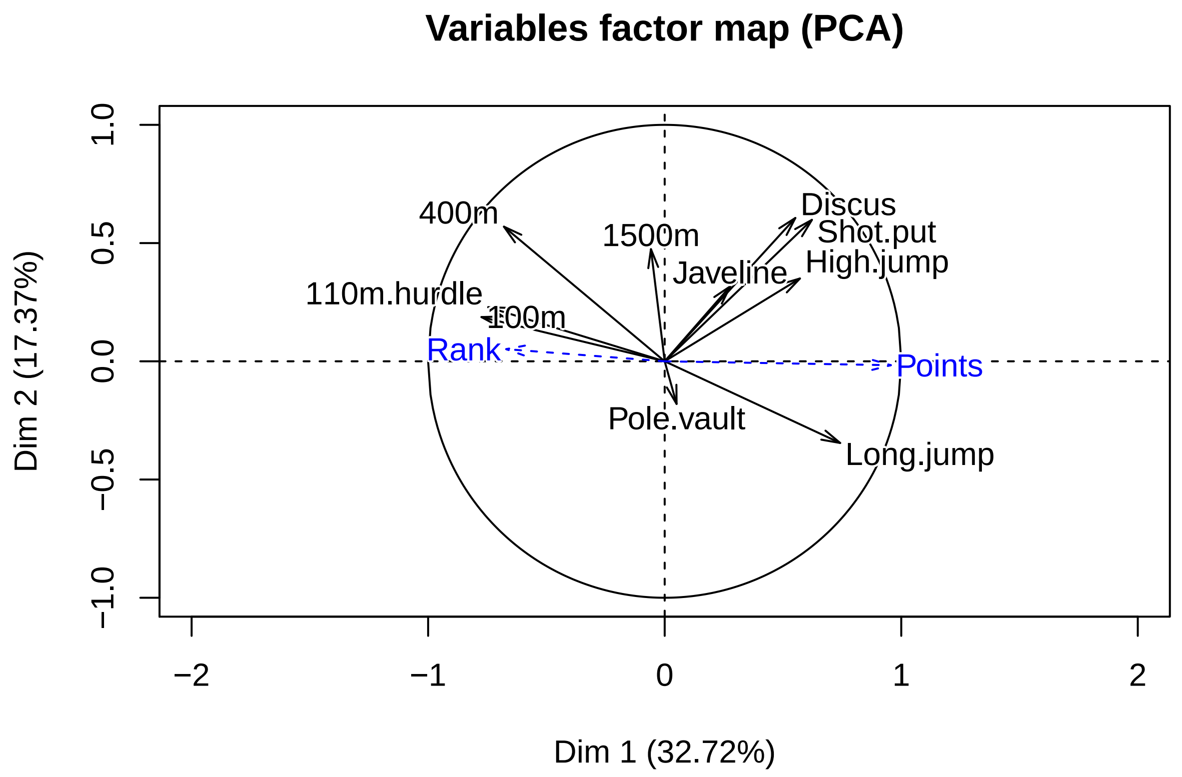
In this exercise, the PCA analysis will be used to describe the main variables that exists on the dataset decathlon, which has been already used during the previous exercises.

The PCA analysis is useful if to see if there are linear relationships between variables. So, for the analysis, all variables will be used as active variables except Points and Rank which will be used as supplementary variables.

Looking to the loaded data decathlon, Points and Rank variables can be determined to derived data because it can be calculated from the combination of other variables. So, we will use it as supplementary just to see them on the result of the analysis but they are not influencing the active variables initial relationship.

```r
#PCA analysis
pcaAnalysis = PCA(decathlon[,1:12], scale.unit=TRUE, quanti.sup=c(11: 12))
```





With the results obtained, it can be determined all the relationships between the different characteristics that an athlete is likely to have.

Analyzing the data obtained, it can be assumed that the most linked analysis to the number of pints are the variables 100m, 110m.hurdle, 400m and long jump. While not for 1500m. In fact, 1500m will depend on other values, which were already calculated on the thirtieth exercise, such as Discuss, 400m, Pole.vault…

Additionally, analyzing the map, it can be easily seen that as much faster as an athlete can run 100m, less it will long jump or, viceversa. So, opposite arrows means opposite result of the behavior.

We can also assume that the first two dimensions resume the 50% of the total variance of the dataset.