**FINAL REPORT**

****

**Mr.Adeel Khatri**

Introduction

The data is growing in enormous size through the various platforms in the world. Every day, the majority of the data generated in the areas of finance, banking, online shopping, education, social media and scientific research is gigantic in size. This data contains a lot of information such that, it has become necessary to understand the data to get some meaningful insights from this data, which can help us to take the better decision in the future. The statistical learning is known as the set of tools that can help us to understand the data with its insights.

In this report, we used the statistical tools that have been taught during the lectures of course Data Analysis and Statistical Learning in the University of Catania. The steps followed are as given below:

* Selection of a suitable data set.
* Describe the selected data set using summary statistics.
* Apply Univariate and Bivariate analysis.
* Apply Principle Component Analysis.
* Apply Clustering Techniques and Validations.
* Conclusion of the report.

Data set

This report uses the education dataset of performances of the students of the past batches of SMI University Karachi, Sindh and Pakistan. This data is collected from four different departments; Environmental Sciences, Computer Sciences, Media Sciences, and Business Administration comprise the data of students enrolled in the MS programs for the year of 2018 and 2019 for a random course. Initially, 952 records were selected from the database of the university. After refining and removing the anomalies in the data 939 instances were chosen for the mining process. Each instance has around 20 attributes out of which 12 attributes were selected for the processing. The table 1 shows the list of number of attributes, description, possible values and categorized values in detail.

Furthermore, to use this dataset I created a github repository “Student-dataset”; committed and pushed the Comma Separated Values (CSV) file at the repository. Using the below code I exported the data in the R-Studio project.

Table: 1 (List of Attributes and detail)

|  |  |  |  |
| --- | --- | --- | --- |
| ***Attribute*** | ***Description*** | ***Possible Values*** | ***Categorized Values*** |
| Gender | Gender of the students usually does not matter in our case | Male and Female | Male/Female |
| Batch Year | The data of two batches are used. | 2018 and 2019 | 2019/2018 |
| Program | The four programs are Computer Sciences, Media Sciences, Business Administration and Environmental Sciences. | MS(CS),MS(ES)  MS(MS) and MS(BA) | MS(CS)/ MS(ES)/ MS(MS)/ MS(BA |
| Sessional/ Mid Term Marks | It is the marks for the mid-term held in the course. | 1 to 30 |  |
| Class Participation Marks | This is the marks given by the teacher on the observation of the class participation i.e. asking questions, giving suggestions and presenting any research-oriented topic | 1 to 10 |  |
| Is Research Oriented | It describes whether a student is taking the program by thesis or course. | Yes and No | Yes/No |
| Student Type | It describes whether the student is part-time or full-time | Part-Time and Full-Time | Part-Time/ Full-Time |
| Financial Condition | It describes the students’ financial conditions. | Good, Bad and Average. | Good/Bad/Average |
| Attendance Marks | It is the attendance marks of the students in the class. | 1 to 10 |  |
| Final Exam Marks | It is the marks of the final exams taken by the students. | 1 to 50 |  |
| Total | It is the total marks taken during the course in the semester. | 1 to 100 | =SUM(Mid-term, Class-participation, attendance Marks, Final Marks) |
| Class Attribute | This class attribute is created according to the grading system of the university. | A+, A, B+, B, C, D and F Where F stands for the fail. | =IF(Total>=80,"A+",IF(Total>=70,"A",IF(Total>=65,"B+",IF(Total>=60,"B",IF(Total>=50,"C",IF(Total>=40,"D","F")))))) |
| CGPA | This is CGPA Continuous attribute created according to the scale of the university out of 5. | 0-5 | =IF(K2>=90,4.63+(K2-90)\*0.037,IF(K2>=80,4.25+(K280)\*0.037,IF(K2>=70,3.88+(K270)\*0.036,IF(K2>=60,3.5+(K260)\*0.037,IF(K2>=50,2.8+(K250)\*0.069,IF(K2>=40,2.8+(K250)\*0.069,IF(K2>0,1+(K2-0)\*0.0248,0))))))) |

R Notebook-Students Data Analysis

The below code, imports the RCurl library which is used to get the data from the online resources.

library(RCurl)

We uses the getURL method of the imported library in order to download the required dataset. And then assigned it into the variable studentsData.

# Collected data from github repository  
x <- getURL("https://raw.githubusercontent.com/adeelahmedcsku/student-dataset/main/20-01-2020-data%20developed%20records%20students-changes%20in%20grading%20system%20-%20data%20developed%20records%20students.csv")  
studentsData <- read.csv(text = x)

The preliminary analysis of the shows that there are 939 obervations and total of 12 variables using the **str()** command.

str(studentsData)

## 'data.frame': 939 obs. of 13 variables:  
## $ Gender : chr "M" "M" "M" "M" ...  
## $ Batch.year : int 2019 2019 2019 2019 2019 2019 2019 2019 2019 2019 ...  
## $ Program : chr "MS(CS)" "MS(CS)" "MS(CS)" "MS(CS)" ...  
## $ Sessional.Marks : int 29 25 18 28 22 28 16 16 15 14 ...  
## $ Class.Participation : int 7 3 6 6 7 3 5 8 8 6 ...  
## $ Is.Research.Oriented: chr "Yes" "Yes" "No" "No" ...  
## $ Student.Type : chr "Part-Time" "Full-Time" "Part-Time" "Part-Time" ...  
## $ Financial.Condition : chr "Good" "Bad" "Good" "Average" ...  
## $ Attendance.Marks : int 8 10 4 6 9 7 8 7 4 8 ...  
## $ Final.Exam.Marks : int 44 40 36 37 43 48 28 39 21 35 ...  
## $ Total : int 88 78 64 77 81 86 57 70 48 63 ...  
## $ CGPA : num 4.55 4.17 3.65 4.13 4.29 ...  
## $ Class : chr "A+" "A" "B" "A" ...

Each observation in the above dataset reflects the performance of a student for a random course in exam. It is composed of the various attributes related to the exams performances. For instances, sessional marks, class participation marks, attendance marks, final exam marks and finally the class attribute which reflects the student grades.

# Univariate Analysis

In Univariate analysis, we can see that the Gender, Program, Is.Research.Oriented, Student.Type, Class and financial condition are the categorical attributes and need to convert into factors. Some attributes are discrete and integer and the attribute CGPA is continuous. The CGPA is our main attribute to order to analysis the data because its the overall performance of the students.

Factoring the Gender attribute:

studentsData$Gender = factor(studentsData$Gender)  
str(studentsData$Gender)

## Factor w/ 2 levels "F","M": 2 2 2 2 2 2 2 1 1 1 ...

Factoring the program attribute:

studentsData$Program = factor(studentsData$Program)  
str(studentsData$Program)

## Factor w/ 4 levels "MS(BA)","MS(CS)",..: 2 2 2 2 2 2 2 2 2 2 ...

Factoring the Is.Research.Oriented attribute:

studentsData$Is.Research.Oriented = factor(studentsData$Is.Research.Oriented)  
str(studentsData$Is.Research.Oriented)

## Factor w/ 2 levels "No","Yes": 2 2 1 1 1 1 1 2 2 2 ...

Factoring the Student.Type attribute:

studentsData$Student.Type = factor(studentsData$Student.Type)  
str(studentsData$Student.Type)

## Factor w/ 2 levels "Full-Time","Part-Time": 2 1 2 2 1 1 2 2 2 1 ...

Factoring the Financial.Condition attribute:

studentsData$Financial.Condition = factor(studentsData$Financial.Condition)  
str(studentsData$Financial.Condition)

## Factor w/ 3 levels "Average","Bad",..: 3 2 3 1 3 3 2 1 2 1 ...

Factoring the Class attribute:

studentsData$Class = factor(studentsData$Class)  
str(studentsData$Class)

## Factor w/ 7 levels "A","A+","B","B+",..: 2 1 3 1 2 2 5 1 6 3 ...

Now, the structure of the dataset is ready to study:

str(studentsData)

## 'data.frame': 939 obs. of 13 variables:  
## $ Gender : Factor w/ 2 levels "F","M": 2 2 2 2 2 2 2 1 1 1 ...  
## $ Batch.year : int 2019 2019 2019 2019 2019 2019 2019 2019 2019 2019 ...  
## $ Program : Factor w/ 4 levels "MS(BA)","MS(CS)",..: 2 2 2 2 2 2 2 2 2 2 ...  
## $ Sessional.Marks : int 29 25 18 28 22 28 16 16 15 14 ...  
## $ Class.Participation : int 7 3 6 6 7 3 5 8 8 6 ...  
## $ Is.Research.Oriented: Factor w/ 2 levels "No","Yes": 2 2 1 1 1 1 1 2 2 2 ...  
## $ Student.Type : Factor w/ 2 levels "Full-Time","Part-Time": 2 1 2 2 1 1 2 2 2 1 ...  
## $ Financial.Condition : Factor w/ 3 levels "Average","Bad",..: 3 2 3 1 3 3 2 1 2 1 ...  
## $ Attendance.Marks : int 8 10 4 6 9 7 8 7 4 8 ...  
## $ Final.Exam.Marks : int 44 40 36 37 43 48 28 39 21 35 ...  
## $ Total : int 88 78 64 77 81 86 57 70 48 63 ...  
## $ CGPA : num 4.55 4.17 3.65 4.13 4.29 ...  
## $ Class : Factor w/ 7 levels "A","A+","B","B+",..: 2 1 3 1 2 2 5 1 6 3 ...

Below command is used to see the summary of the dataset:

summary(studentsData)

## Gender Batch.year Program Sessional.Marks Class.Participation  
## F:429 Min. :2018 MS(BA):302 Min. :10.00 Min. : 2.000   
## M:510 1st Qu.:2018 MS(CS):203 1st Qu.:16.00 1st Qu.: 4.000   
## Median :2019 MS(ES):201 Median :20.00 Median : 6.000   
## Mean :2019 MS(MS):233 Mean :20.24 Mean : 6.027   
## 3rd Qu.:2019 3rd Qu.:25.00 3rd Qu.: 8.000   
## Max. :2019 Max. :30.00 Max. :10.000   
##   
## Is.Research.Oriented Student.Type Financial.Condition Attendance.Marks  
## No :459 Full-Time:469 Average:273 Min. : 2.000   
## Yes:480 Part-Time:470 Bad :360 1st Qu.: 5.000   
## Good :306 Median : 7.000   
## Mean : 6.942   
## 3rd Qu.: 8.000   
## Max. :10.000   
##   
## Final.Exam.Marks Total CGPA Class   
## Min. : 6.00 Min. :30.00 Min. :1.744 A :350   
## 1st Qu.:30.00 1st Qu.:63.00 1st Qu.:3.611 A+:140   
## Median :35.00 Median :70.00 Median :3.880 B :133   
## Mean :35.12 Mean :68.33 Mean :3.753 B+:147   
## 3rd Qu.:40.00 3rd Qu.:77.00 3rd Qu.:4.132 C : 96   
## Max. :50.00 Max. :92.00 Max. :4.704 D : 30   
## F : 43

## Gender:

The gender is the categorical variable of type Nominal. It consists of two possible values ‘M’ for Male and ‘F’ for female.

table(studentsData$Gender)

##   
## F M   
## 429 510

frequencyGender <- table(studentsData$Gender)

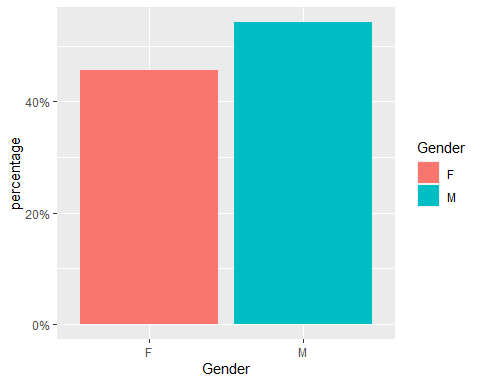
The percentage distribution of the dataset with respect to the gender can be calculated by multiply to 100.

prop.table(frequencyGender) \* 100

##   
## F M   
## 45.6869 54.3131

The above frequency clearly elaborate that the dataset is balanced and about 46% are female students and 54% are male students. The frequency in the percentages is shows in the below bar plot.

library(ggplot2)  
ggplot(studentsData, aes(x=Gender)) +  
geom\_bar(aes(y=(..count..)/sum(..count..),fill=Gender))+  
scale\_y\_continuous(labels=scales::percent) +  
ylab('percentage')



## Batch Year:

The Batch year is the numerical variable of type integer. It consists of two possible values ‘2018’ and ‘2019’.

table(studentsData$Batch.year)

##   
## 2018 2019   
## 401 538

frequencyBatchyear <- table(studentsData$Batch.year)

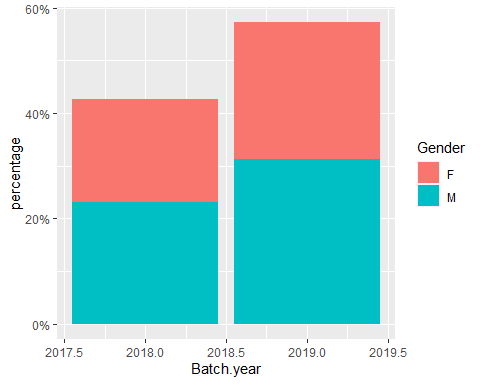
The percentage distribution of the dataset with respect to the batch year can be calculated by multiply to 100.

prop.table(frequencyBatchyear) \* 100

##   
## 2018 2019   
## 42.70501 57.29499

The above frequency clearly elaborate that the dataset is not much balanced and about 43% are 2018 students and 57% are 2019 students. The frequency in the percentages is shows in the below bar plot.

library(ggplot2)  
ggplot(studentsData, aes(x=Batch.year)) +  
geom\_bar(aes(y=(..count..)/sum(..count..),fill=Gender))+  
scale\_y\_continuous(labels=scales::percent) +  
ylab('percentage')



## Program:

The program is the Categorical variable of type nominal. It consists of four possible values ‘MS(CS)’ , ‘MS(MS)’, MS(ES) and MS(BA).

table(studentsData$Program)

##   
## MS(BA) MS(CS) MS(ES) MS(MS)   
## 302 203 201 233

frequencyProgram <- table(studentsData$Program)

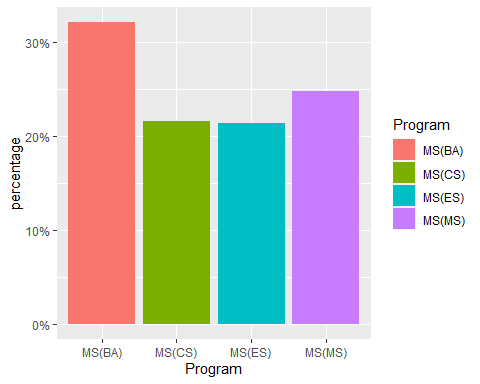
The percentage distribution of the dataset with respect to the Program can be calculated using the below formula.

prop.table(frequencyProgram) \* 100

##   
## MS(BA) MS(CS) MS(ES) MS(MS)   
## 32.16187 21.61874 21.40575 24.81363

The above frequency clearly elaborate that the dataset is consiss of 32% Business Administration students, 21% are Computer Science students, 21% are Envoirmental Science students and 24 percent are Media Science students. The frequency in the percentages is shows in the below bar plot.

library(ggplot2)  
ggplot(studentsData, aes(x=Program)) +  
geom\_bar(aes(y=(..count..)/sum(..count..),fill=Program))+  
scale\_y\_continuous(labels=scales::percent) +  
ylab('percentage')

 ## Is.Research.Oriented:

The Is.Research.Oriented is the categorical variable of type Nominal. It consists of two possible values ‘Yes’ and ‘No’.

table(studentsData$Is.Research.Oriented)

##   
## No Yes   
## 459 480

frequencyIsResearchOriented <- table(studentsData$Is.Research.Oriented)

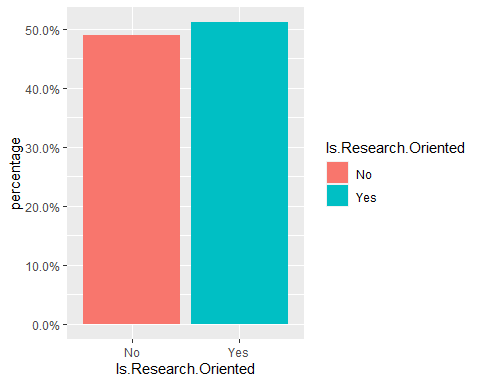
The percentage distribution of the dataset with respect to the Reserch Oriented can be calculated by multiply to 100.

prop.table(frequencyIsResearchOriented) \* 100

##   
## No Yes   
## 48.88179 51.11821

The above frequency clearly elaborate that the dataset is balanced and about 49% are Not the research oriented students and 51% are research oriented students. The frequency in the percentages is shows in the below bar plot.

library(ggplot2)  
ggplot(studentsData, aes(x=Is.Research.Oriented)) +  
geom\_bar(aes(y=(..count..)/sum(..count..),fill=Is.Research.Oriented))+  
scale\_y\_continuous(labels=scales::percent) +  
ylab('percentage')



## Student.Type:

The Student.Type is the categorical variable of type Nominal. It consists of two possible values ‘Full-Time’ and ‘Part-Time’.

table(studentsData$Student.Type)

##   
## Full-Time Part-Time   
## 469 470

frequencyStudentType <- table(studentsData$Student.Type)

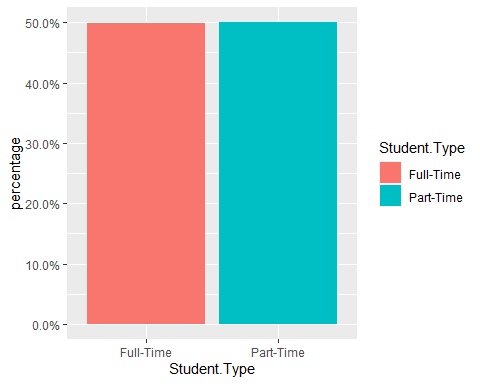
The percentage distribution of the dataset with respect to the student type can be calculated by multiply to 100.

prop.table(frequencyStudentType) \* 100

##   
## Full-Time Part-Time   
## 49.94675 50.05325

The above frequency clearly elaborate that the dataset is completly balanced and about 50 percent of both the students are present Full-Time and Part-Time. The frequency in the percentages is shows in the below bar plot.

library(ggplot2)  
ggplot(studentsData, aes(x=Student.Type)) +  
geom\_bar(aes(y=(..count..)/sum(..count..),fill=Student.Type))+  
scale\_y\_continuous(labels=scales::percent) +  
ylab('percentage')



## Financial.Condition:

The Financial.Condition is the categorical variable of type Nominal. It consists of three possible values ‘Average’, ‘Bad’ and ‘Good’.

table(studentsData$Financial.Condition)

##   
## Average Bad Good   
## 273 360 306

frequencyFinancialCondition <- table(studentsData$Financial.Condition)

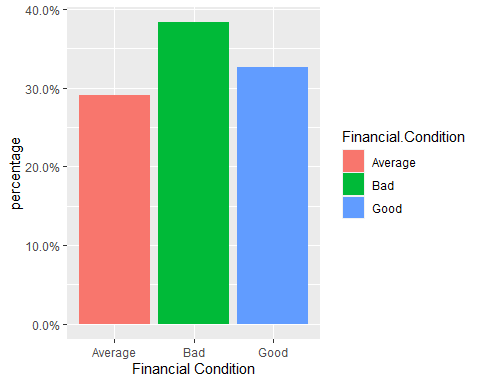
The percentage distribution of the dataset with respect to the financial condition can be calculated by multiply to 100.

prop.table(frequencyFinancialCondition) \* 100

##   
## Average Bad Good   
## 29.07348 38.33866 32.58786

The above frequency clearly elaborate that the dataset is partially balanced and about 29 percent of students have Average financial condition, 38 percent of the students have Bad financial condition and 32 percent of the students have Good financial condition. The frequency in the percentages is shows in the below bar plot.

library(ggplot2)  
ggplot(studentsData, aes(x=Financial.Condition)) +  
geom\_bar(aes(y=(..count..)/sum(..count..),fill=Financial.Condition))+  
scale\_y\_continuous(labels=scales::percent) +  
ylab('percentage')+  
xlab('Financial Condition')



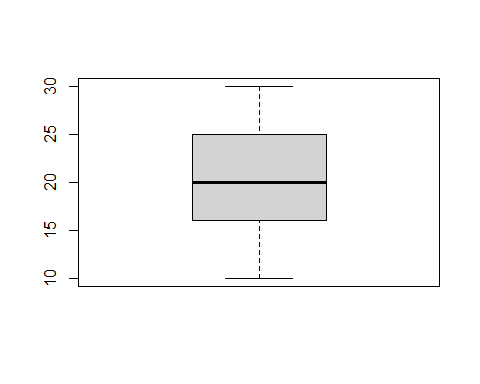
## Sessional Marks:

The Sessional marks is the Numerical variable of type Integer. It contains the mid-semester exams marks of the students. The summary of the sessional mark attribute is given below:

summary(studentsData$Sessional.Marks)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 10.00 16.00 20.00 20.24 25.00 30.00

boxplot(studentsData$Sessional.Marks)

 The variance provides an estimate of how much the observed data deviate from the mean. The variance is computed as the average squared distance of the values with respect to the mean.

var(studentsData$Sessional.Marks, na.rm=T)

## [1] 26.45429

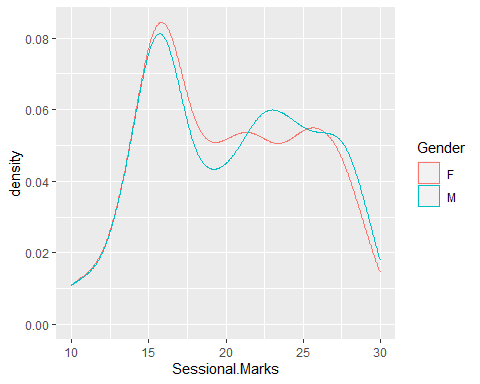
We can see that the mean is greater than variance which indicates binomial distribution.

sd(studentsData$Sessional.Marks, na.rm=T)

## [1] 5.143373

The standard deviation tells us something about the “shape” of the distribution. Let’s use aggregate to compute standard deviation of Age for passengers in the different classes.

library(ggplot2)  
ggplot(studentsData, aes(x=Sessional.Marks, color=Gender)) +  
geom\_density()



table(studentsData$Sessional.Marks)

##   
## 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29   
## 26 1 19 12 59 64 150 34 31 50 39 44 66 50 43 64 39 59 52 31   
## 30   
## 6

frequencySessionalMarks <- table(studentsData$Sessional.Marks)

The percentage distribution of the dataset with respect to the Program can be calculated using the below formula.

prop.table(frequencySessionalMarks) \* 100

##   
## 10 11 12 13 14 15 16   
## 2.7689031 0.1064963 2.0234292 1.2779553 6.2832801 6.8157614 15.9744409   
## 17 18 19 20 21 22 23   
## 3.6208733 3.3013845 5.3248136 4.1533546 4.6858360 7.0287540 5.3248136   
## 24 25 26 27 28 29 30   
## 4.5793397 6.8157614 4.1533546 6.2832801 5.5378062 3.3013845 0.6389776

The above frequency in represents the percentages of the sessional Marks in the dataset.

We can check the index of the most frequent marks as follows:

which.max(frequencySessionalMarks)

## 16   
## 7

This returns the the position of the max (7) and the name of the maximum value 16. We can obtain the maximum value as follows:

max(frequencySessionalMarks)

## [1] 150

Similarly with the least frequent item:

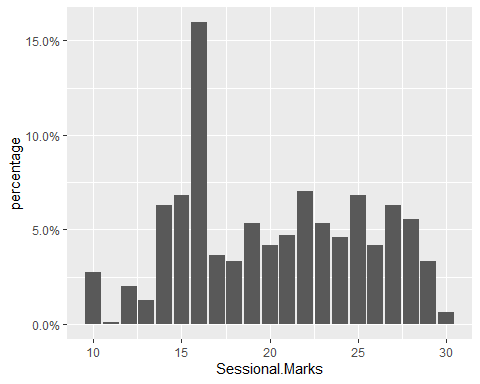
frequencySessionalMarks[which.min(frequencySessionalMarks)]

## 11   
## 1

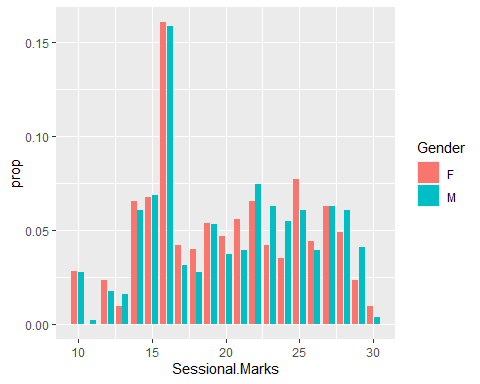
The frequency table is already telling us something about the sample: • The most frequent sessional mark is 16 ; • The least frequent sessional mark is 11 ;

The sessional marks barplot with the percentages of the marks is given that:

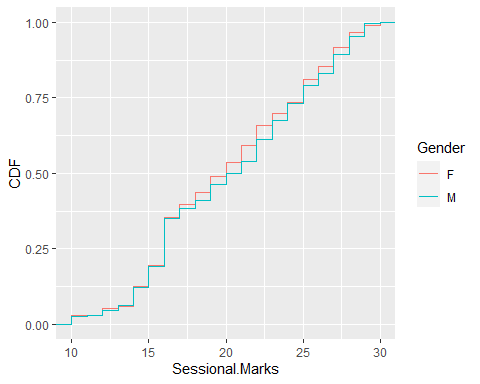
library(ggplot2)  
ggplot(studentsData, aes(x=Sessional.Marks)) +  
geom\_bar(aes(y=(..count..)/sum(..count..)))+  
scale\_y\_continuous(labels=scales::percent) +  
ylab('percentage')

 PMF of sessional marks is as:

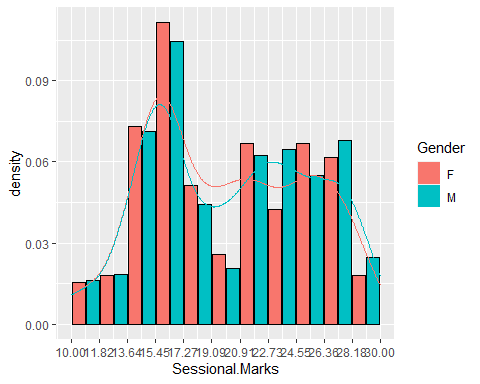
ggplot(studentsData, aes(x=Sessional.Marks, fill=Gender)) +  
geom\_bar(aes(y=..prop..), position=position\_dodge2(preserve="single"))

 For CDF, We can compare the male and female samples by specifying a color=sex aesthetic:

ggplot(studentsData, aes(x=Sessional.Marks,color=Gender)) +  
stat\_ecdf() +  
ylab('CDF')

 The density histogram is given below for the sessional marks with respect to the Gender.

ggplot(studentsData, aes(x=Sessional.Marks, fill=Gender)) +  
geom\_histogram(aes(y=..density.., ), bins=12, color='black', position = position\_dodge2(preserve='single')) +  
scale\_x\_continuous(breaks=seq(min(studentsData$Sessional.Marks),max(studentsData$Sessional.Marks),length=12),  
labels = scales::number\_format(accuracy = 0.01))+  
geom\_density(aes(color=Gender), fill=NA)

 Generalized Additive Models for Location, Scale and Shape (GAMLSS) were introduced by Rigby and Stasinopoulos (2001, 2005) and Akantziliotou et al. (2002) as a way of overcoming some of the limitations associated with Generalized Linear Models (GLM) and Generalized Additive Models (GAM) (Nelder and Wedderburn, 1972 and Hastie and Tibshirani, 1990, respectively) which are already part of the R CRAN packages. We install the packages of GAMLSS in R.

The GAMLSS is the package that is used to select the best model of the statiscal distribution fits to our data. In order to find the best fit model for our data i will keep including the knowlegde of this model library in this document.

The below are some libraries of these packages that we need to include in our code.

library(gamlss.dist)

## Loading required package: MASS

library(gamlss)

## Loading required package: splines

## Loading required package: gamlss.data

##   
## Attaching package: 'gamlss.data'

## The following object is masked from 'package:datasets':  
##   
## sleep

## Loading required package: nlme

## Loading required package: parallel

## \*\*\*\*\*\*\*\*\*\* GAMLSS Version 5.2-0 \*\*\*\*\*\*\*\*\*\*

## For more on GAMLSS look at https://www.gamlss.com/

## Type gamlssNews() to see new features/changes/bug fixes.

library(gamlss.data)  
library(gamlss.add)

## Loading required package: mgcv

## This is mgcv 1.8-33. For overview type 'help("mgcv-package")'.

## Loading required package: nnet

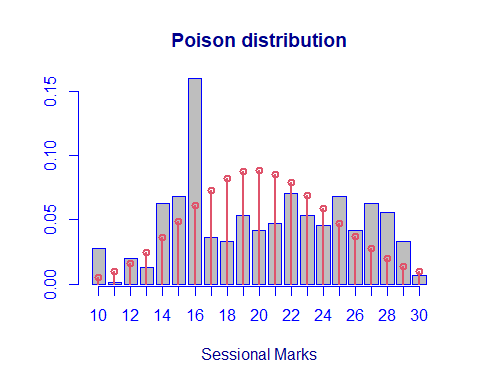
##   
## Attaching package: 'nnet'

## The following object is masked from 'package:mgcv':  
##   
## multinom

## Loading required package: rpart

library(gamlss.countKinf)  
library(gamlss.mx)

dist.poison <- histDist(studentsData$Sessional.Marks, family=PO, nbins = 30, main="Poison distribution",xlab = "Sessional Marks")



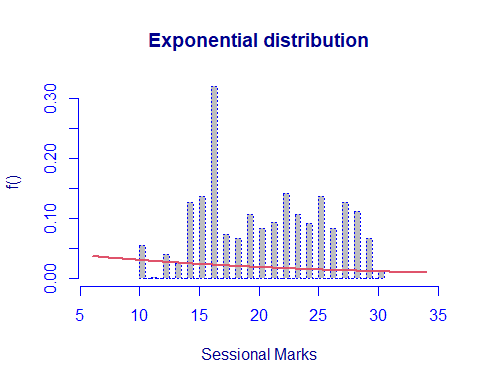
dist.poison$aic

## [1] 5775.793

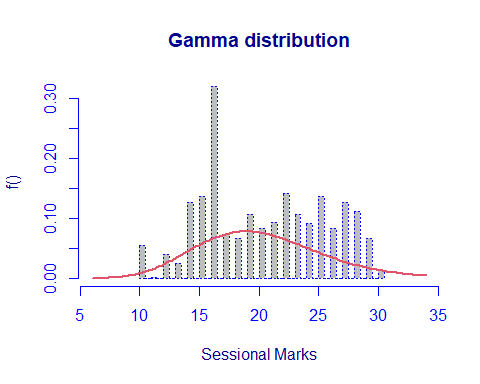
dist.poison$sbc

## [1] 5780.638

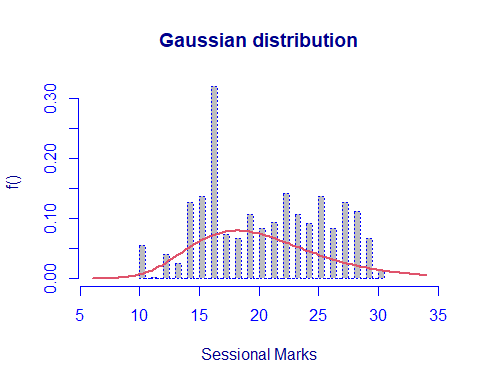
Sessional.Marks<- studentsData$Sessional.Marks  
 dist.expo <- histDist(studentsData$Sessional.Marks, family=EXP, nbins = 30, main="Exponential distribution",xlab = "Sessional Marks")



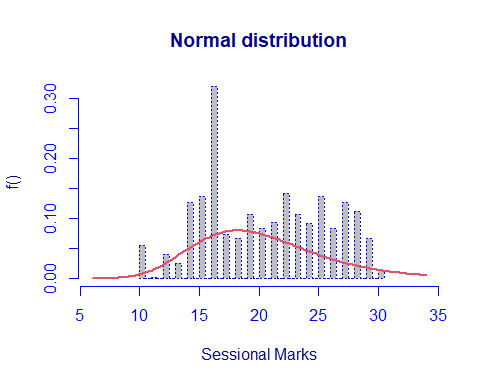
dist.gama <- histDist(studentsData$Sessional.Marks, family=GA, nbins = 30, main="Gamma distribution",xlab = "Sessional Marks")



dist.gaussian <- histDist(studentsData$Sessional.Marks, family=IG, nbins = 30, main="Gaussian distribution",xlab = "Sessional Marks")



dist.lognormal <- histDist(studentsData$Sessional.Marks, family=LOGNO, nbins = 30, main="Normal distribution",xlab = "Sessional Marks")



# {r} # dist.weibull <- histDist(studentsData$Sessional.Marks, family=WEI, nbins = 30, main="Weibull distribution",xlab = "Sessional Marks") # #

data.frame(row.names = c("Exponential", "Gamma", "Inverse Gaussian", "Log-Normal", "Poison"),  
 AIC=c(AIC(dist.expo), AIC(dist.gama), AIC(dist.gaussian), AIC(dist.lognormal), AIC(dist.poison)),  
 SBC=c(dist.expo$sbc, dist.gama$sbc, dist.gaussian$sbc, dist.lognormal$sbc, dist.poison$sbc)  
 )

## AIC SBC  
## Exponential 7528.549 7533.394  
## Gamma 5745.355 5755.045  
## Inverse Gaussian 5767.025 5776.715  
## Log-Normal 5769.279 5778.969  
## Poison 5775.793 5780.638

The Akaike information criterion (AIC) is used to estimate the prediction error in the statistical model of the given dataset. It estimates the quiality of each model comparing to the other models. AIC estimates the relative amount of information lost by a given model: the less information a model loses, the higher the quality of that model.

The Bayesian information criterion (BIC) or Schwarz information criterion (also SIC, SBC, SBIC) is a criterion for model selection among a finite set of models; the model with the lowest BIC is preferred. It is based, in part, on the likelihood function and it is closely related to the Akaike information criterion (AIC).

In the above table the gamma distribution model is the best fit for our data due to the lowest values of the AIC and SBC.

## Likelihood-ratio test

In order to assess the goodness-of-fit, Likelihood-ratio test between the Exponential model (under the null hypotesis) and the Weibull model (under the alternative hypotesis) is computed.

LR.test(dist.expo,dist.gama)

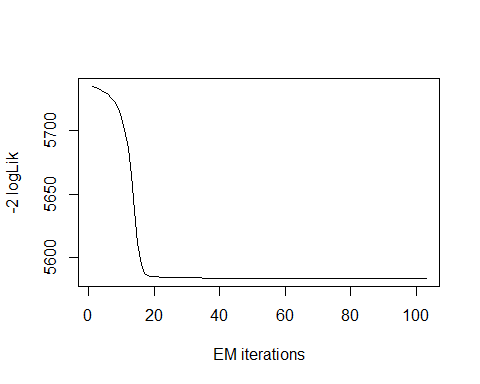
## Likelihood Ratio Test for nested GAMLSS models.   
## (No check whether the models are nested is performed).   
##   
## Null model: deviance= 7526.549 with 1 deg. of freedom   
## Altenative model: deviance= 5741.355 with 2 deg. of freedom   
##   
## LRT = 1785.194 with 1 deg. of freedom and p-value= 0

The p-values is 0: the null model, i.e. the Exponential distribution, is rejected.

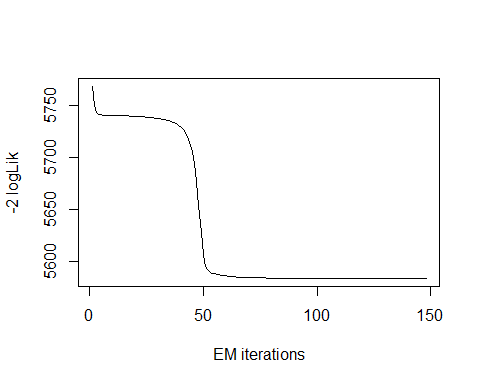
## Mixture Model of Gamma:

We can also fit mixture distributions, for example the Gamma ones:

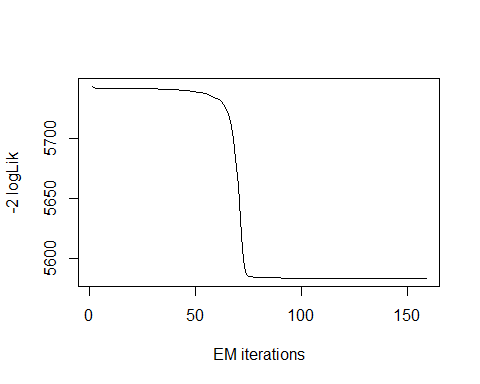
dist.gamma.mix <- gamlssMXfits(n = 5, Sessional.Marks~1, family = GA, K = 2,data = studentsData)



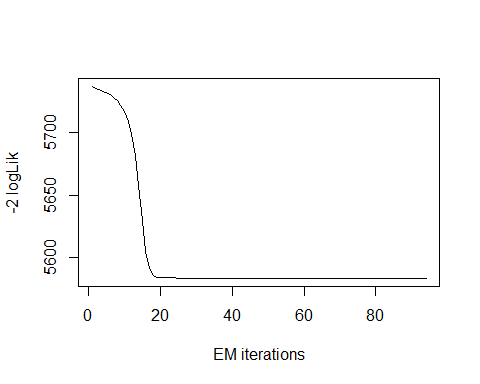
## model= 1



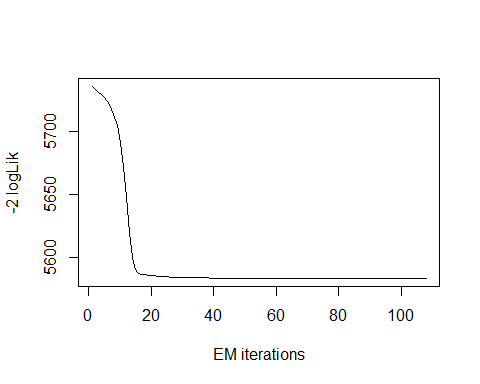
## model= 2



## model= 3



## model= 4



## model= 5

dist.gamma.mix$aic

## [1] 5593.424

dist.gamma.mix$sbc

## [1] 5617.648

We can see that with the mixture of two Gamma distributions the AIC and BIC values have improved, as they are lower than those obtained with the single Gamma distribution: AIC is now equal to 5429.286, while the previous value was 5449.719; BIC is now 5453.51, which is lower than 5454.564 , which was the previous value.

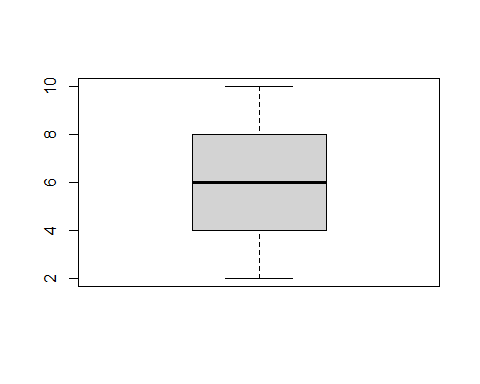
## Class.Participation

The summary of the Class Participation marks attribute is given below:

summary(studentsData$Class.Participation)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 2.000 4.000 6.000 6.027 8.000 10.000

boxplot(studentsData$Class.Participation)

 The variance provides an estimate of how much the observed data deviate from the mean. The variance is computed as the average squared distance of the values with respect to the mean.

var(studentsData$Class.Participation, na.rm=T)

## [1] 5.217841

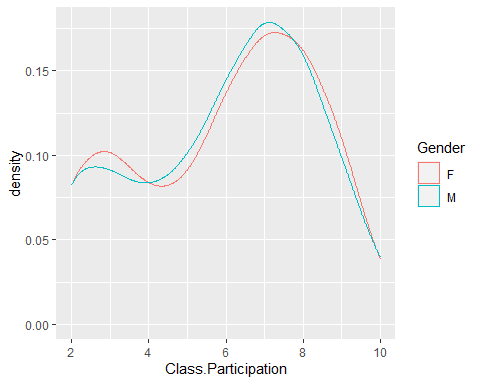
We can see that the mean is greater than variance which indicates binomial distribution.

sd(studentsData$Class.Participation, na.rm=T)

## [1] 2.284259

The standard deviation tells us something about the “shape” of the distribution. Let’s use aggregate to compute standard deviation of Age for passengers in the different classes.

library(ggplot2)  
ggplot(studentsData, aes(x=Class.Participation, color=Gender)) +  
geom\_density()



The class participation marks is the Numerical variable of type Integer. It contains the marks of the student who take part in the class presentation and answering the questions of the teacher.

table(studentsData$Class.Participation)

##   
## 2 3 4 5 6 7 8 9 10   
## 93 94 73 84 134 173 158 100 30

frequencyClassParticipation <- table(studentsData$Class.Participation)

The percentage distribution of the dataset with respect to the class participation marks can be calculated using the above formula.

prop.table(frequencyClassParticipation) \* 100

##   
## 2 3 4 5 6 7 8 9   
## 9.904153 10.010650 7.774228 8.945687 14.270501 18.423855 16.826411 10.649627   
## 10   
## 3.194888

The above frequency in represents the percentages of the Class participation Marks in the dataset.

We can check the index of the most frequent marks as follows:

which.max(frequencyClassParticipation)

## 7   
## 6

This returns the the position of the max (7) and the name of the maximum value 16. We can obtain the maximum value as follows:

max(frequencyClassParticipation)

## [1] 173

Similarly with the least frequent item:

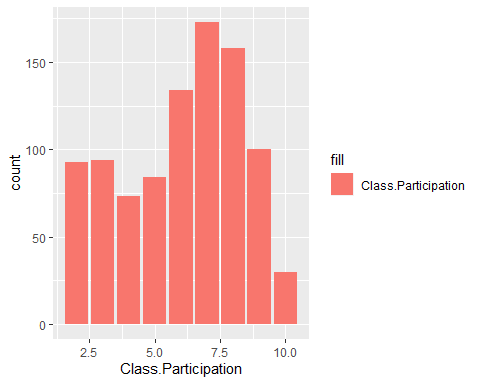
frequencyClassParticipation[which.min(frequencyClassParticipation)]

## 10   
## 30

The frequency table is already telling us something about the sample: • The most frequent class participation marks is 7 i.e. very few students were able to perform well in order to take all class participation marks; • The least frequent sessional mark is 10 ;

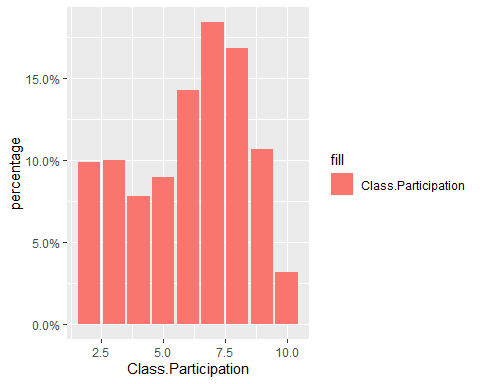
The bar plot for the frequencies is given that:

library(ggplot2)  
ggplot(studentsData, aes(x=Class.Participation,fill='Class.Participation')) +  
geom\_bar()

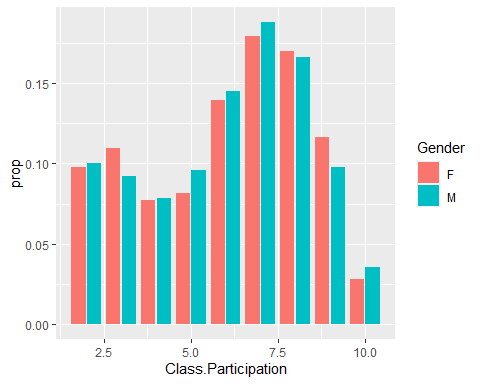


The Class participation marks barplot with the percentages of the marks is given that:

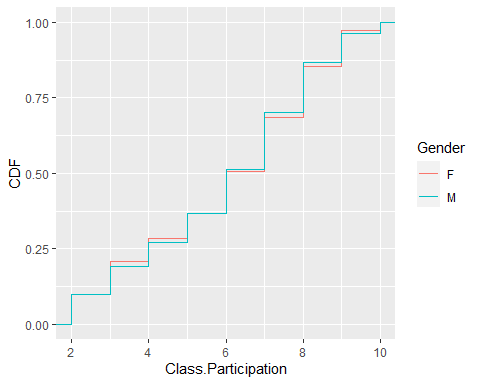
library(ggplot2)  
ggplot(studentsData, aes(x=Class.Participation)) +  
geom\_bar(aes(y=(..count..)/sum(..count..),fill='Class.Participation'))+  
scale\_y\_continuous(labels=scales::percent) +  
ylab('percentage')

 PMF of Class participation marks is as:

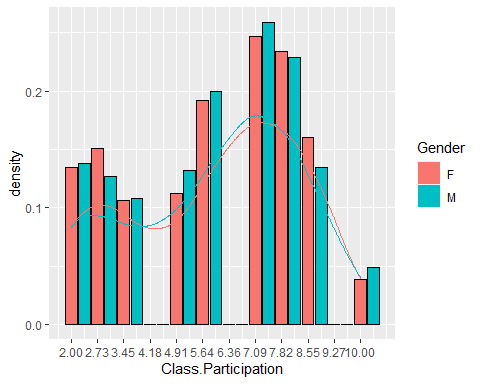
ggplot(studentsData, aes(x=Class.Participation, fill=Gender)) +  
geom\_bar(aes(y=..prop..), position=position\_dodge2(preserve="single"))

 For CDF, We can compare the male and female samples by specifying a color=sex aesthetic:

ggplot(studentsData, aes(x=Class.Participation,color=Gender)) +  
stat\_ecdf() +  
ylab('CDF')

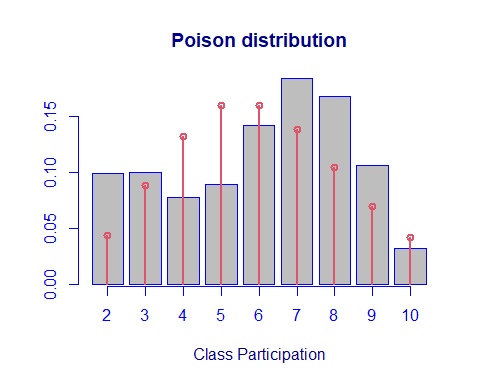
 The density histogram is given below for the Class Participation marks with respect to the Gender.

ggplot(studentsData, aes(x=Class.Participation, fill=Gender)) +  
geom\_histogram(aes(y=..density.., ), bins=12, color='black', position = position\_dodge2(preserve='single')) +  
scale\_x\_continuous(breaks=seq(min(studentsData$Class.Participation),max(studentsData$Class.Participation),length=12),  
labels = scales::number\_format(accuracy = 0.01))+  
geom\_density(aes(color=Gender), fill=NA)



Generalized Additive Models for Location, Scale and Shape (GAMLSS) analysis of different statistical models is given below:

dist.poison <- histDist(studentsData$Class.Participation, family=PO, nbins = 30, main="Poison distribution",xlab = "Class Participation")



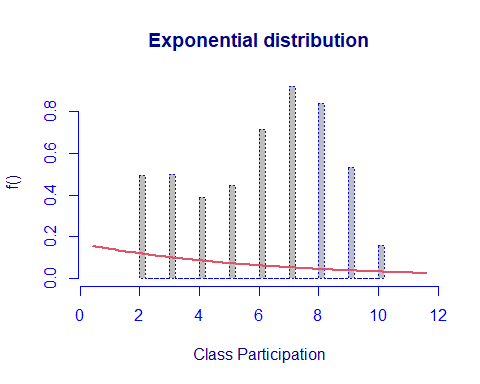
dist.poison$aic

## [1] 4255.456

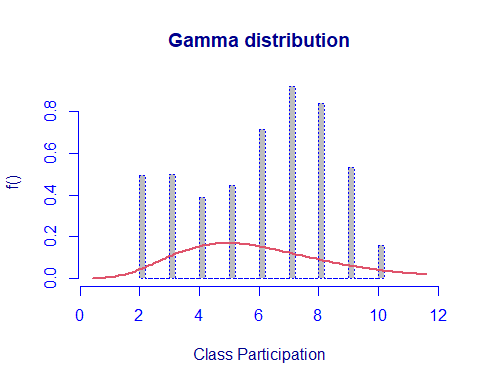
dist.poison$sbc

## [1] 4260.301

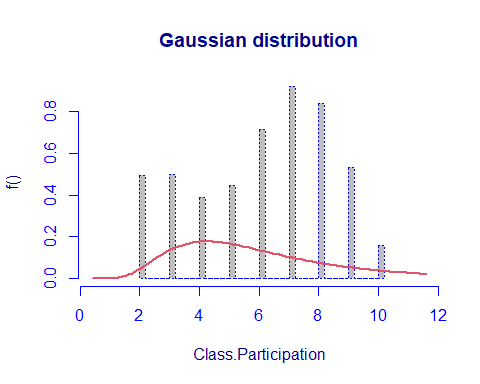
dist.expo <- histDist(studentsData$Class.Participation, family=EXP, nbins = 30, main="Exponential distribution",xlab = "Class Participation")



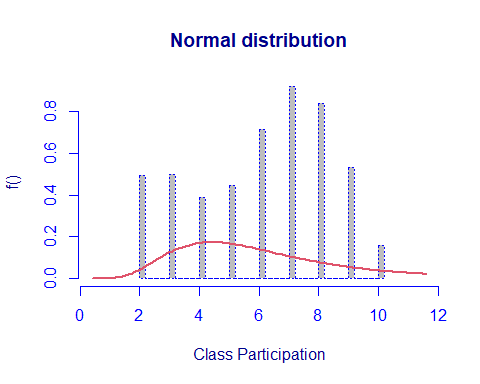
dist.gama <- histDist(studentsData$Class.Participation, family=GA, nbins = 30, main="Gamma distribution",xlab = "Class Participation")



dist.gaussian <- histDist(studentsData$Class.Participation, family=IG, nbins = 30, main="Gaussian distribution",xlab = "Class.Participation")



dist.lognormal <- histDist(studentsData$Class.Participation, family=LOGNO, nbins = 30, main="Normal distribution",xlab = "Class Participation")



# {r} # dist.weibull <- histDist(studentsData$Sessional.Marks, family=WEI, nbins = 30, main="Weibull distribution",xlab = "Sessional Marks") # #

data.frame(row.names = c("Exponential", "Gamma", "Inverse Gaussian", "Log-Normal", "Poison"),  
 AIC=c(AIC(dist.expo), AIC(dist.gama), AIC(dist.gaussian), AIC(dist.lognormal), AIC(dist.poison)),  
 SBC=c(dist.expo$sbc, dist.gama$sbc, dist.gaussian$sbc, dist.lognormal$sbc, dist.poison$sbc)  
 )

## AIC SBC  
## Exponential 5253.239 5258.084  
## Gamma 4326.303 4335.993  
## Inverse Gaussian 4451.412 4461.102  
## Log-Normal 4441.242 4450.932  
## Poison 4255.456 4260.301

In the above table the Poisson distribution model is the best fit for our data due to the lowest values of the AIC and SBC.

## Likelihood-ratio test

In order to assess the goodness-of-fit, Likelihood-ratio test between the Exponential model (under the null hypotesis) and the Poisson model (under the alternative hypothesis) is computed.

LR.test(dist.expo,dist.poison)

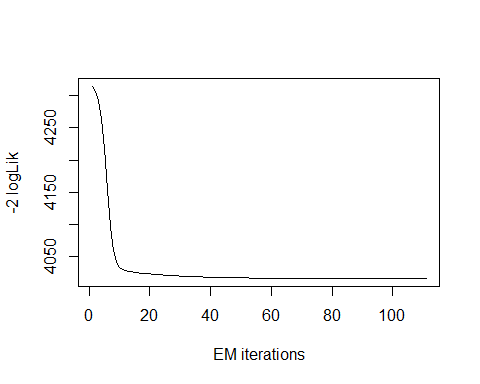
## Likelihood Ratio Test for nested GAMLSS models.   
## (No check whether the models are nested is performed).   
##   
## Null model: deviance= 5251.239 with 1 deg. of freedom   
## Altenative model: deviance= 4253.456 with 1 deg. of freedom   
##   
## LRT = 997.7831 with 0 deg. of freedom and p-value= 0

The p-values is 0: the null model, i.e. the Exponential distribution, is rejected.

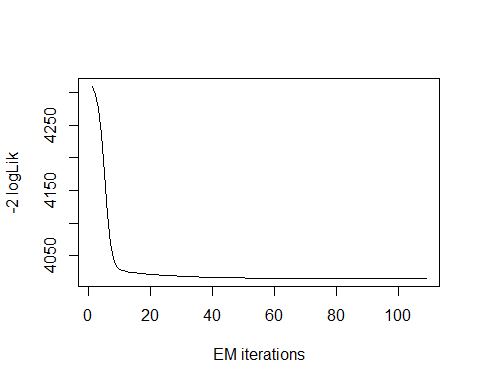
## Mixture Model of Gamma:

We can also fit mixture distributions, for example the Gamma ones:

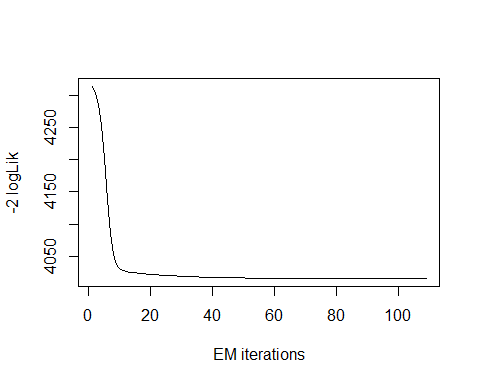
dist.gamma.mix <- gamlssMXfits(n = 5, Class.Participation~1, family = GA, K = 2,data = studentsData)



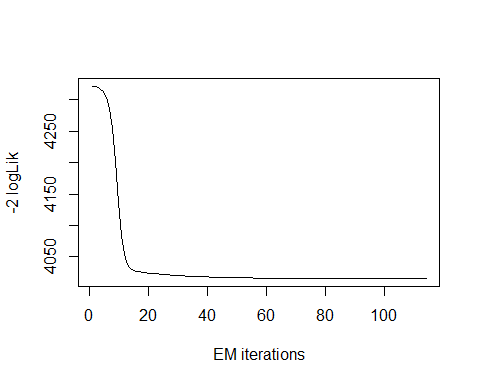
## model= 1



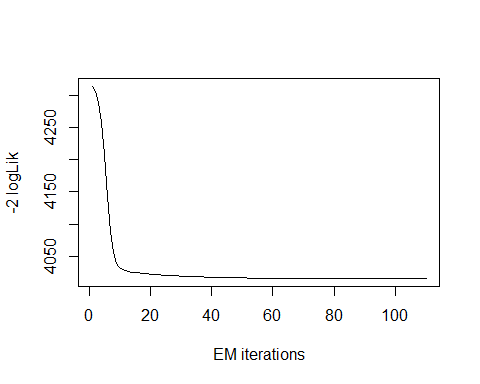
## model= 2



## model= 3



## model= 4



## model= 5

dist.gamma.mix$aic

## [1] 4025.565

dist.gamma.mix$sbc

## [1] 4049.789

We can see that with the mixture of two Gamma distributions the AIC and BIC values have improved, as they are lower than those obtained with the single Gamma distribution: AIC is now equal to 4025.565, while the previous value was 4326.303; SBC is now 4049.789, which is lower than 4335.993 , which was the previous value. Even the mixed gamma function model is much better than the chosen Poisson Model.

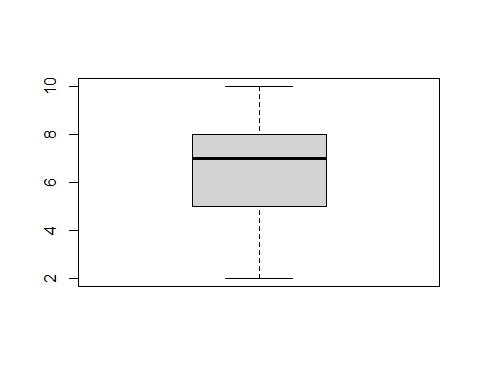
## Attendance.Marks

The Attendance marks is the Numerical variable of type Integer. It contains the marks of the students according to their attendance in the class.

summary(studentsData$Attendance.Marks)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 2.000 5.000 7.000 6.942 8.000 10.000

boxplot(studentsData$Attendance.Marks)



The variance provides an estimate of how much the observed data deviate from the mean. The variance is computed as the average squared distance of the values with respect to the mean.

var(studentsData$Attendance.Marks, na.rm=T)

## [1] 3.46364

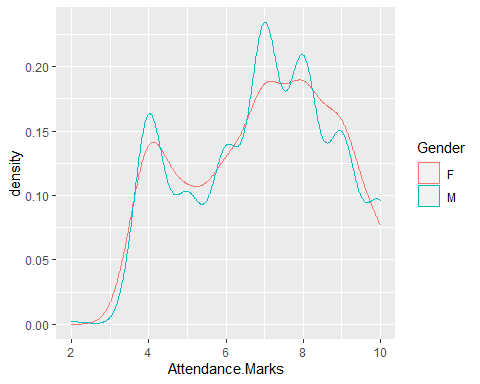
We can see that the mean is greater than variance which indicates binomial distribution.

sd(studentsData$Attendance.Marks, na.rm=T)

## [1] 1.861086

The standard deviation tells us something about the “shape” of the distribution. Let’s use aggregate to compute standard deviation of Age for passengers in the different classes.

library(ggplot2)  
ggplot(studentsData, aes(x=Attendance.Marks, color=Gender)) +  
geom\_density()



The summary of the Attendance marks attribute is given below:

table(studentsData$Attendance.Marks)

##   
## 2 4 5 6 7 8 9 10   
## 1 148 88 116 192 178 139 77

frequencyAttendanceMarks <- table(studentsData$Attendance.Marks)

The percentage distribution of the dataset with respect to the Attendance marks can be calculated using the above formula.

prop.table(frequencyAttendanceMarks) \* 100

##   
## 2 4 5 6 7 8 9   
## 0.1064963 15.7614483 9.3716720 12.3535676 20.4472843 18.9563365 14.8029819   
## 10   
## 8.2002130

The above frequency in represents the percentages of the attendance Marks in the dataset.

We can check the index of the most frequent marks as follows:

which.max(frequencyAttendanceMarks)

## 7   
## 5

This returns the the position of the max (5) and the name of the maximum value 7. We can obtain the maximum number of times a number repeats is as follows:

max(frequencyAttendanceMarks)

## [1] 192

Similarly with the least frequent item:

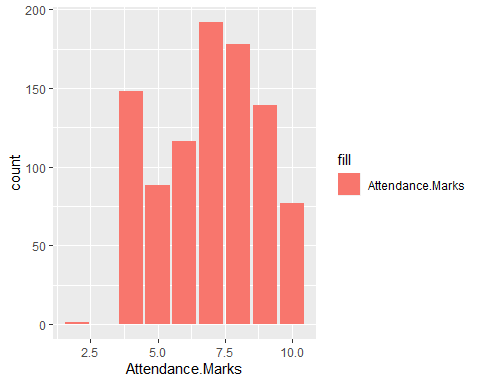
frequencyAttendanceMarks[which.min(frequencyAttendanceMarks)]

## 2   
## 1

The frequency table is already telling us something about the sample: • The most frequent attendacne marks is 7 it implies that most of the students took average marks in the attendance; • The least frequent attendance marks is 1 ;

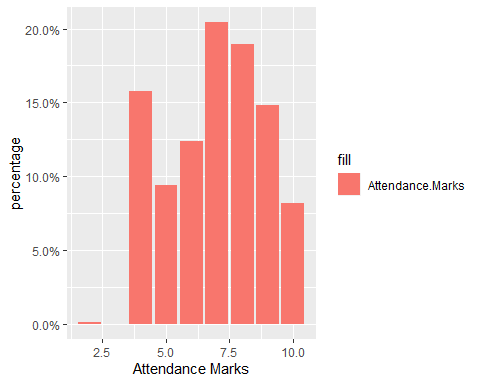
The bar plot for the frequencies is given that:

library(ggplot2)  
ggplot(studentsData, aes(x=Attendance.Marks,fill='Attendance.Marks')) +  
geom\_bar()



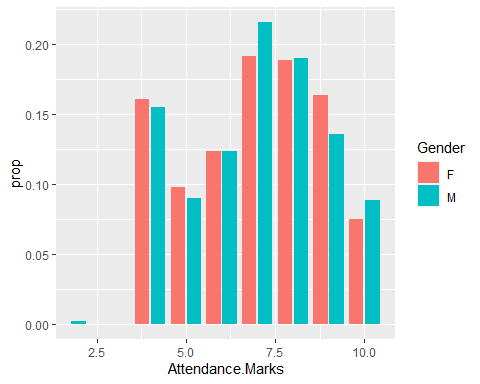
The Attendance marks barplot with the percentages of the marks is given that:

library(ggplot2)  
ggplot(studentsData, aes(x=Attendance.Marks)) +  
geom\_bar(aes(y=(..count..)/sum(..count..),fill='Attendance.Marks'))+  
scale\_y\_continuous(labels=scales::percent) +  
ylab('percentage')+  
xlab('Attendance Marks')



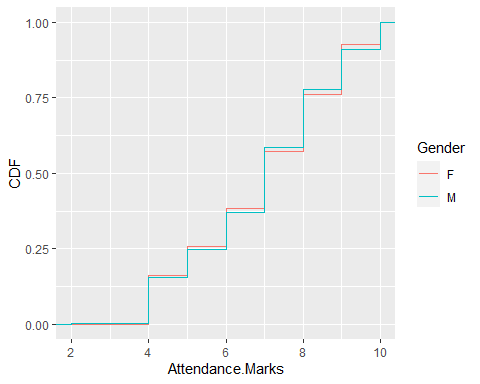
PMF of Attendance marks is as:

ggplot(studentsData, aes(x=Attendance.Marks, fill=Gender)) +  
geom\_bar(aes(y=..prop..), position=position\_dodge2(preserve="single"))



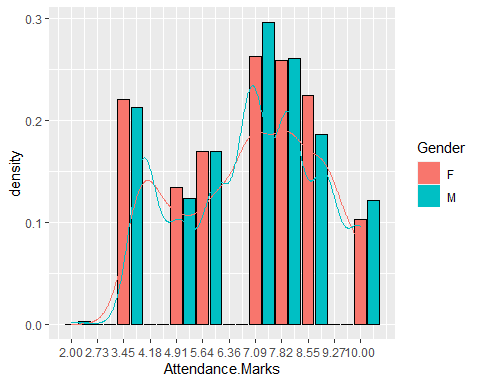
For CDF, We can compare the male and female samples by specifying a color=sex aesthetic:

ggplot(studentsData, aes(x=Attendance.Marks,color=Gender)) +  
stat\_ecdf() +  
ylab('CDF')



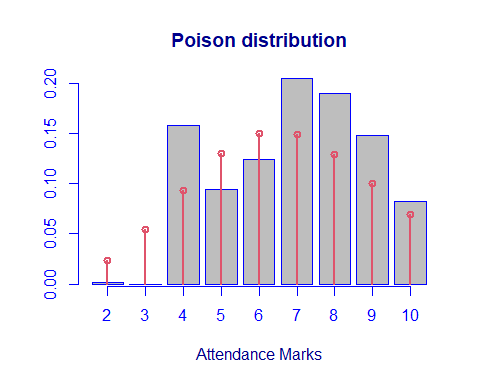
The density histogram is given below for the Attendance marks with respect to the Gender.

ggplot(studentsData, aes(x=Attendance.Marks, fill=Gender)) +  
geom\_histogram(aes(y=..density.., ), bins=12, color='black', position = position\_dodge2(preserve='single')) +  
scale\_x\_continuous(breaks=seq(min(studentsData$Attendance.Marks),max(studentsData$Attendance.Marks),length=12),  
labels = scales::number\_format(accuracy = 0.01))+  
geom\_density(aes(color=Gender), fill=NA)



Generalized Additive Models for Location, Scale and Shape (GAMLSS) analysis of different statistical models is given below:

dist.poison <- histDist(studentsData$Attendance.Marks, family=PO, nbins = 30, main="Poison distribution",xlab = "Attendance Marks")



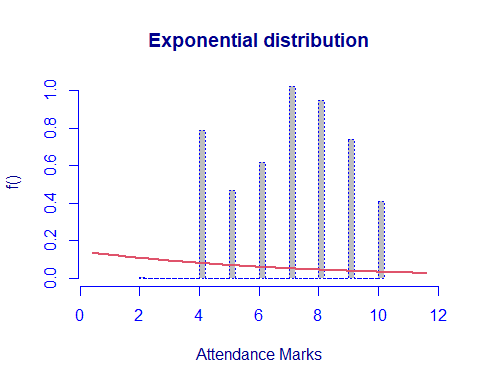
dist.poison$aic

## [1] 4021.742

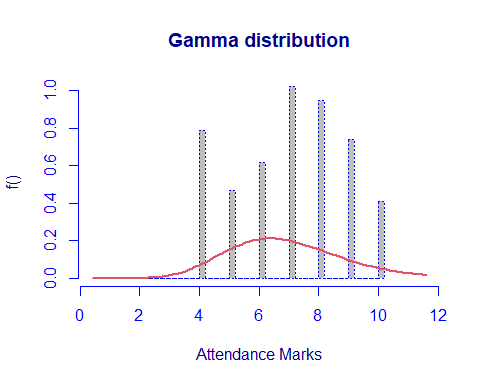
dist.poison$sbc

## [1] 4026.587

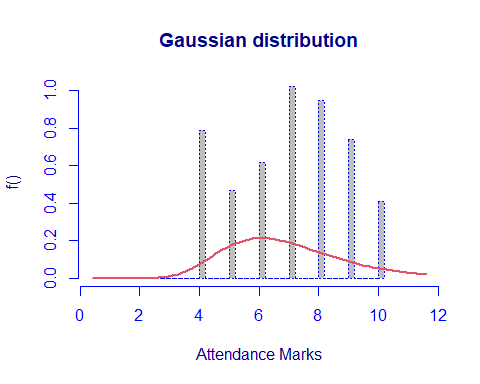
dist.expo <- histDist(studentsData$Attendance.Marks, family=EXP, nbins = 30, main="Exponential distribution",xlab = "Attendance Marks")



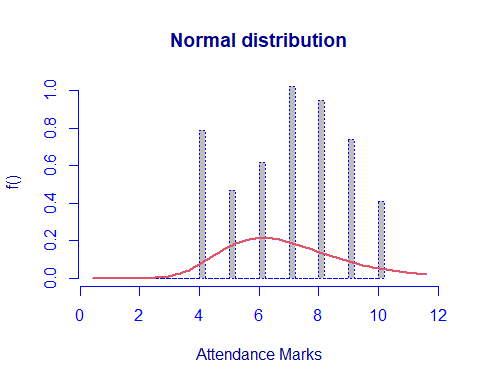
dist.gama <- histDist(studentsData$Attendance.Marks, family=GA, nbins = 30, main="Gamma distribution",xlab = "Attendance Marks")



dist.gaussian <- histDist(studentsData$Attendance.Marks, family=IG, nbins = 30, main="Gaussian distribution",xlab = "Attendance Marks")



dist.lognormal <- histDist(studentsData$Attendance.Marks, family=LOGNO, nbins = 30, main="Normal distribution",xlab = "Attendance Marks")



# {r} # dist.weibull <- histDist(studentsData$Sessional.Marks, family=WEI, nbins = 30, main="Weibull distribution",xlab = "Sessional Marks") # #

data.frame(row.names = c("Exponential", "Gamma", "Inverse Gaussian", "Log-Normal", "Poison"),  
 AIC=c(AIC(dist.expo), AIC(dist.gama), AIC(dist.gaussian), AIC(dist.lognormal), AIC(dist.poison)),  
 SBC=c(dist.expo$sbc, dist.gama$sbc, dist.gaussian$sbc, dist.lognormal$sbc, dist.poison$sbc)  
 )

## AIC SBC  
## Exponential 5518.927 5523.772  
## Gamma 3875.575 3885.265  
## Inverse Gaussian 3920.418 3930.107  
## Log-Normal 3922.066 3931.755  
## Poison 4021.742 4026.587

In the above table the Gamma distribution model is the best fit for our data due to the lowest values of the AIC and SBC.

## Likelihood-ratio test

In order to assess the goodness-of-fit, Likelihood-ratio test between the Exponential model (under the null hypotesis) and the gamma model (under the alternative hypothesis) is computed.

LR.test(dist.expo,dist.gama)

## Likelihood Ratio Test for nested GAMLSS models.   
## (No check whether the models are nested is performed).   
##   
## Null model: deviance= 5516.927 with 1 deg. of freedom   
## Altenative model: deviance= 3871.575 with 2 deg. of freedom   
##   
## LRT = 1645.352 with 1 deg. of freedom and p-value= 0

The p-values is 0: the null model, i.e. the Exponential distribution, is rejected.

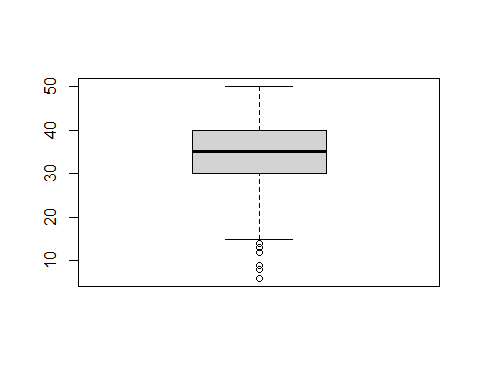
## Final.Exam.Marks

The Final Exam marks is the Numerical variable of type Integer. It contains the marks of the students according to their performance in the semester exam.

summary(studentsData$Final.Exam.Marks)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 6.00 30.00 35.00 35.12 40.00 50.00

boxplot(studentsData$Final.Exam.Marks)



table(studentsData$Final.Exam.Marks)

##   
## 6 8 9 12 13 14 15 16 17 18 19 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35   
## 1 1 2 1 1 2 2 4 19 1 3 24 23 4 6 3 9 11 77 25 30 75 20 31 45 50   
## 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50   
## 44 50 35 84 30 10 13 34 27 26 31 26 55 3 1

frequencyFinalExamMarks <- table(studentsData$Final.Exam.Marks)

The percentage distribution of the dataset with respect to the Final Exams marks can be calculated using the above formula.

prop.table(frequencyFinalExamMarks) \* 100

##   
## 6 8 9 12 13 14 15 16   
## 0.1064963 0.1064963 0.2129925 0.1064963 0.1064963 0.2129925 0.2129925 0.4259851   
## 17 18 19 21 22 23 24 25   
## 2.0234292 0.1064963 0.3194888 2.5559105 2.4494143 0.4259851 0.6389776 0.3194888   
## 26 27 28 29 30 31 32 33   
## 0.9584665 1.1714590 8.2002130 2.6624068 3.1948882 7.9872204 2.1299255 3.3013845   
## 34 35 36 37 38 39 40 41   
## 4.7923323 5.3248136 4.6858360 5.3248136 3.7273695 8.9456869 3.1948882 1.0649627   
## 42 43 44 45 46 47 48 49   
## 1.3844515 3.6208733 2.8753994 2.7689031 3.3013845 2.7689031 5.8572950 0.3194888   
## 50   
## 0.1064963

The above frequency in represents the percentages of the Final Exam Marks in the dataset.

We can check the index of the most frequent marks as follows:

which.max(frequencyFinalExamMarks)

## 39   
## 30

This returns the the position of the max (30) and the name of the maximum value 39. We can obtain the maximum number of times a number repeats is as follows:

max(frequencyFinalExamMarks)

## [1] 84

Similarly with the least frequent item:

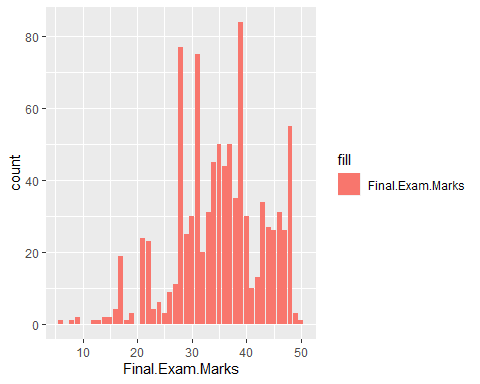
frequencyFinalExamMarks[which.min(frequencyFinalExamMarks)]

## 6   
## 1

The frequency table is already telling us something about the sample: • The most frequent Final Exams marks is 39 it implies that most of the students performs above average marks in the Final Exams Marks; • The least frequent Final Exam marks is 6 ;

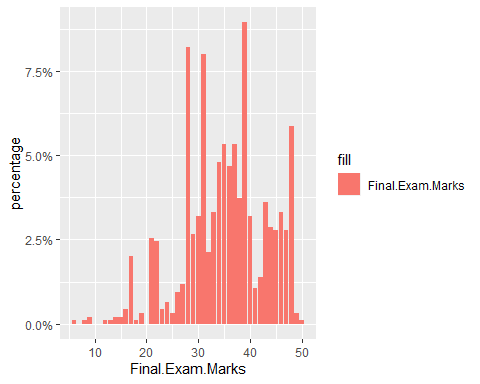
The bar plot for the frequencies is given that:

library(ggplot2)  
ggplot(studentsData, aes(x=Final.Exam.Marks,fill='Final.Exam.Marks')) +  
geom\_bar()

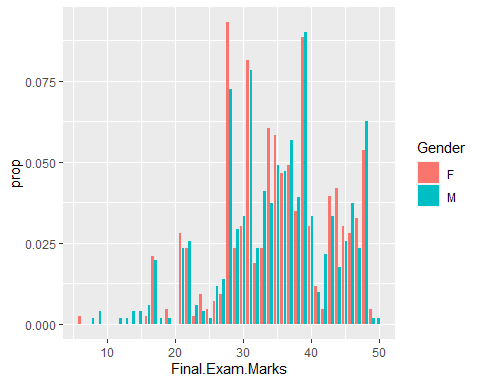


The Final Exam Marks barplot with the percentages of the marks is given that:

library(ggplot2)  
ggplot(studentsData, aes(x=Final.Exam.Marks)) +  
geom\_bar(aes(y=(..count..)/sum(..count..),fill='Final.Exam.Marks'))+  
scale\_y\_continuous(labels=scales::percent) +  
ylab('percentage')+  
xlab('Final.Exam.Marks')

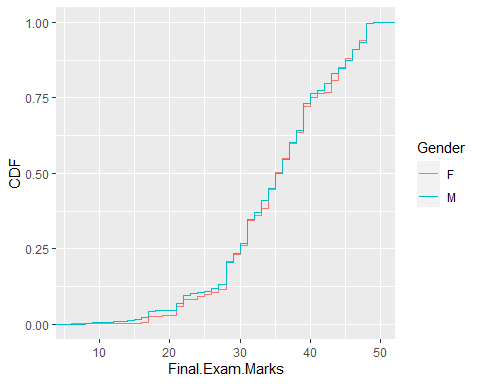
 PMF of Final Exam marks is as:

ggplot(studentsData, aes(x=Final.Exam.Marks, fill=Gender)) +  
geom\_bar(aes(y=..prop..), position=position\_dodge2(preserve="single"))



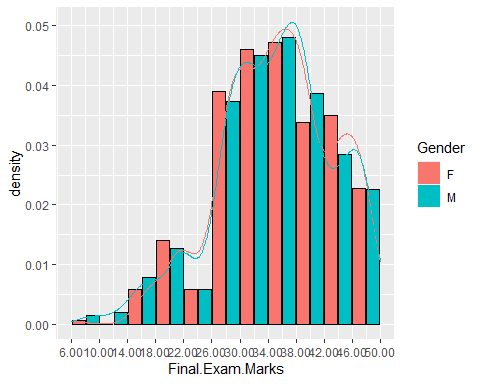
For CDF, We can compare the male and female samples by specifying a color=sex aesthetic:

ggplot(studentsData, aes(x=Final.Exam.Marks,color=Gender)) +  
stat\_ecdf() +  
ylab('CDF')



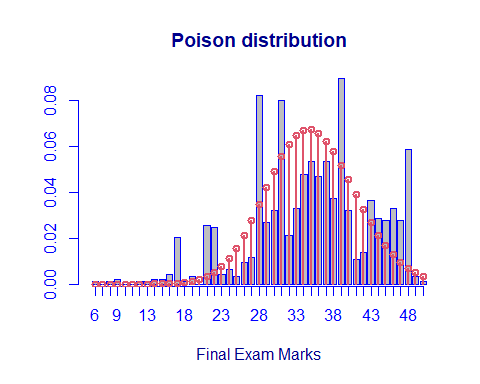
The density histogram is given below for the Final Exam marks with respect to the Gender.

ggplot(studentsData, aes(x=Final.Exam.Marks, fill=Gender)) +  
geom\_histogram(aes(y=..density.., ), bins=12, color='black', position = position\_dodge2(preserve='single')) +  
scale\_x\_continuous(breaks=seq(min(studentsData$Final.Exam.Marks),max(studentsData$Final.Exam.Marks),length=12),  
labels = scales::number\_format(accuracy = 0.01))+  
geom\_density(aes(color=Gender), fill=NA)



Generalized Additive Models for Location, Scale and Shape (GAMLSS) analysis of different statistical models is given below:

dist.poison <- histDist(studentsData$Final.Exam.Marks, family=PO, nbins = 30, main="Poison distribution",xlab = "Final Exam Marks")



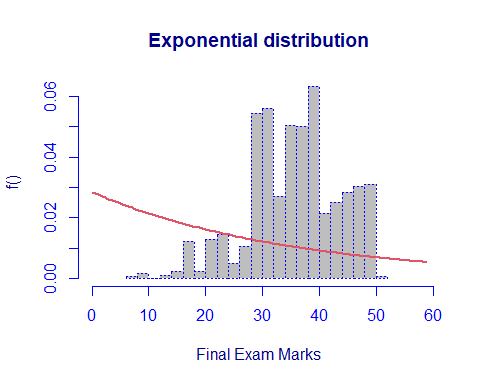
dist.poison$aic

## [1] 6901.753

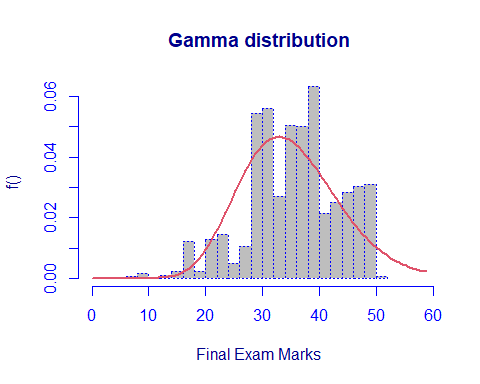
dist.poison$sbc

## [1] 6906.598

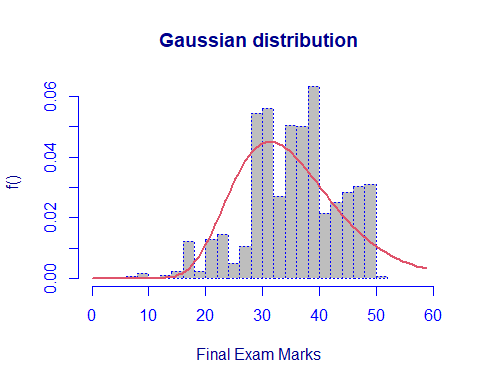
dist.expo <- histDist(studentsData$Final.Exam.Marks, family=EXP, nbins = 30, main="Exponential distribution",xlab = "Final Exam Marks")



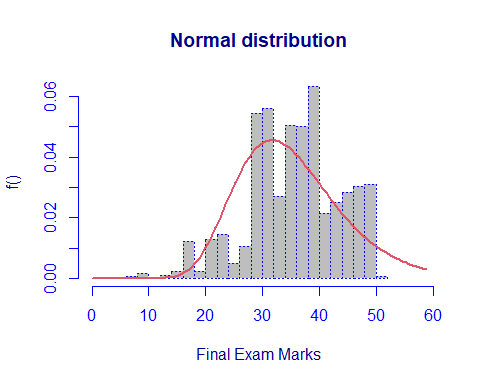
dist.gama <- histDist(studentsData$Final.Exam.Marks, family=GA, nbins = 30, main="Gamma distribution",xlab = "Final Exam Marks")



dist.gaussian <- histDist(studentsData$Final.Exam.Marks, family=IG, nbins = 30, main="Gaussian distribution",xlab = "Final Exam Marks")



dist.lognormal <- histDist(studentsData$Final.Exam.Marks, family=LOGNO, nbins = 30, main="Normal distribution",xlab = "Final Exam Marks")



# {r} # dist.weibull <- histDist(studentsData$Sessional.Marks, family=WEI, nbins = 30, main="Weibull distribution",xlab = "Sessional Marks") # #

data.frame(row.names = c("Exponential", "Gamma", "Inverse Gaussian", "Log-Normal", "Poison"),  
 AIC=c(AIC(dist.expo), AIC(dist.gama), AIC(dist.gaussian), AIC(dist.lognormal), AIC(dist.poison)),  
 SBC=c(dist.expo$sbc, dist.gama$sbc, dist.gaussian$sbc, dist.lognormal$sbc, dist.poison$sbc)  
 )

## AIC SBC  
## Exponential 8563.447 8568.291  
## Gamma 6706.484 6716.174  
## Inverse Gaussian 6836.995 6846.685  
## Log-Normal 6807.716 6817.406  
## Poison 6901.753 6906.598

In the above table the Gamma distribution model is the best fit for our data due to the lowest values of the AIC and SBC.

## Likelihood-ratio test

In order to assess the goodness-of-fit, Likelihood-ratio test between the Exponential model (under the null hypotesis) and the gamma model (under the alternative hypothesis) is computed.

LR.test(dist.expo,dist.gama)

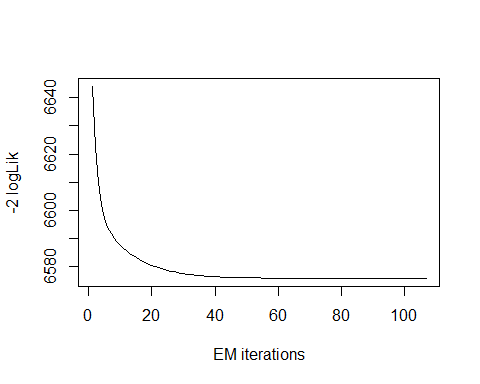
## Likelihood Ratio Test for nested GAMLSS models.   
## (No check whether the models are nested is performed).   
##   
## Null model: deviance= 8561.447 with 1 deg. of freedom   
## Altenative model: deviance= 6702.484 with 2 deg. of freedom   
##   
## LRT = 1858.963 with 1 deg. of freedom and p-value= 0

The p-values is 0: the null model, i.e. the Exponential distribution, is rejected.

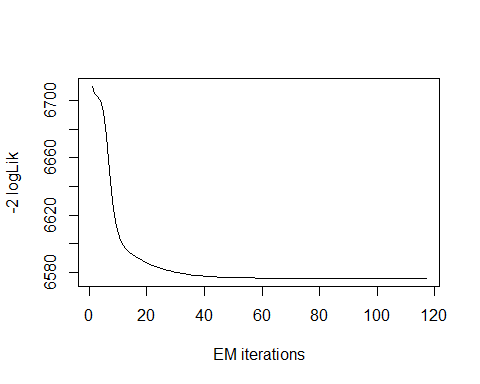
## Mixture Model of Gamma:

We can also fit mixture distributions, for example the Gamma ones:

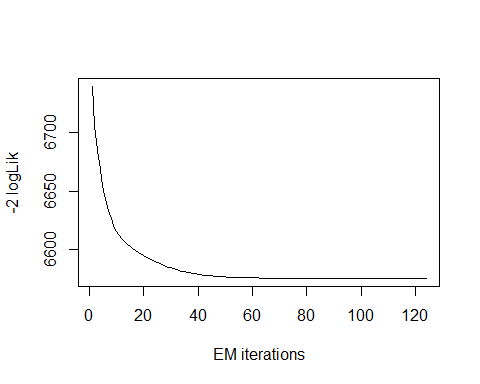
dist.gamma.mix <- gamlssMXfits(n = 5, Final.Exam.Marks~1, family = GA, K = 2,data = studentsData)



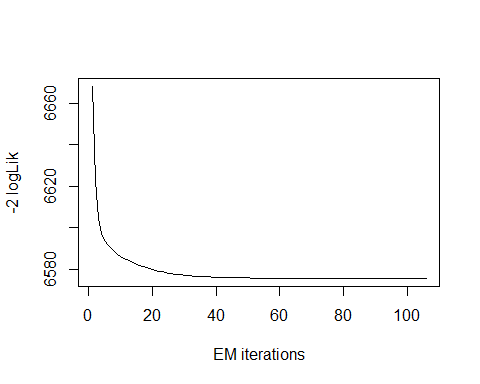
## model= 1



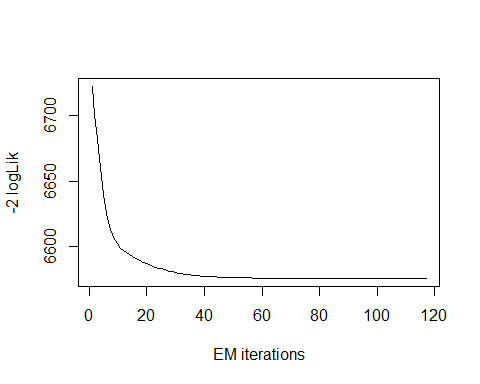
## model= 2



## model= 3



## model= 4



## model= 5

dist.gamma.mix$aic

## [1] 6585.734

dist.gamma.mix$sbc

## [1] 6609.958

We can see that with the mixture of two Gamma distributions the AIC and BIC values have improved, as they are lower than those obtained with the single Gamma distribution: AIC is now equal to 6585.734, while the previous value was 6706.484; SBC is now 6609.958, which is lower than 6716.174 , which was the previous value. Even the mixed gamma function model is much better than the chosen Poisson Model.

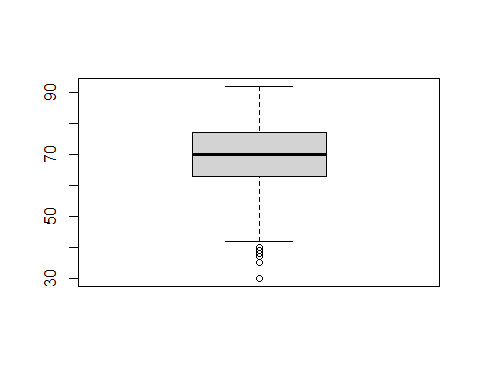
## Total

The Total marks is the Numerical variable of type Integer. It is the sum of the total marks obtained on the basis of sessional, class participation and attendance marks.

summary(studentsData$Total)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 30.00 63.00 70.00 68.33 77.00 92.00

boxplot(studentsData$Total)

 The frequencies of the Total marks are given below:

table(studentsData$Total)

##   
## 30 35 37 38 39 40 42 43 44 47 48 50 51 52 53 54 55 56 57 58   
## 1 2 17 1 22 1 1 1 1 2 24 1 2 2 1 4 3 3 63 7   
## 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78   
## 10 12 13 15 75 18 23 25 49 27 23 113 19 25 23 24 17 22 64 28   
## 79 80 81 82 83 84 85 86 87 88 89 90 91 92   
## 15 12 31 14 8 9 7 43 5 4 3 1 2 1

frequencyTotal <- table(studentsData$Total)

The percentage distribution of the dataset with respect to the Total marks can be calculated using the above formula.

prop.table(frequencyTotal) \* 100

##   
## 30 35 37 38 39 40 42   
## 0.1064963 0.2129925 1.8104366 0.1064963 2.3429180 0.1064963 0.1064963   
## 43 44 47 48 50 51 52   
## 0.1064963 0.1064963 0.2129925 2.5559105 0.1064963 0.2129925 0.2129925   
## 53 54 55 56 57 58 59   
## 0.1064963 0.4259851 0.3194888 0.3194888 6.7092652 0.7454739 1.0649627   
## 60 61 62 63 64 65 66   
## 1.2779553 1.3844515 1.5974441 7.9872204 1.9169329 2.4494143 2.6624068   
## 67 68 69 70 71 72 73   
## 5.2183174 2.8753994 2.4494143 12.0340788 2.0234292 2.6624068 2.4494143   
## 74 75 76 77 78 79 80   
## 2.5559105 1.8104366 2.3429180 6.8157614 2.9818956 1.5974441 1.2779553   
## 81 82 83 84 85 86 87   
## 3.3013845 1.4909478 0.8519702 0.9584665 0.7454739 4.5793397 0.5324814   
## 88 89 90 91 92   
## 0.4259851 0.3194888 0.1064963 0.2129925 0.1064963

The above frequency in represents the percentages of the Total Exam Marks in the dataset.

We can check the index of the most frequent marks as follows:

which.max(frequencyTotal)

## 70   
## 32

This returns the the position of the max (32) and the name of the maximum value 70. We can obtain the maximum number of times a number repeats is as follows:

max(frequencyTotal)

## [1] 113

Similarly with the least frequent item:

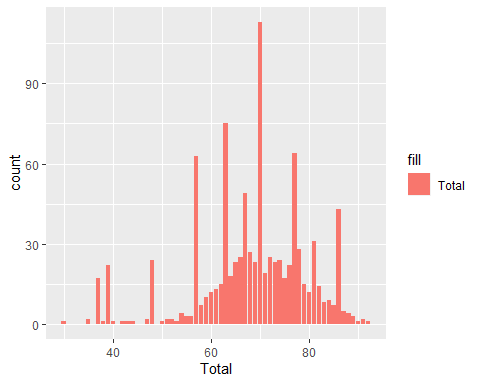
frequencyTotal[which.min(frequencyTotal)]

## 30   
## 1

The frequency table is already telling us something about the sample: • The most frequent Total marks is 70 it implies that most of the students performs above average marks as total; • The least frequent Final Exam marks is 30 ;

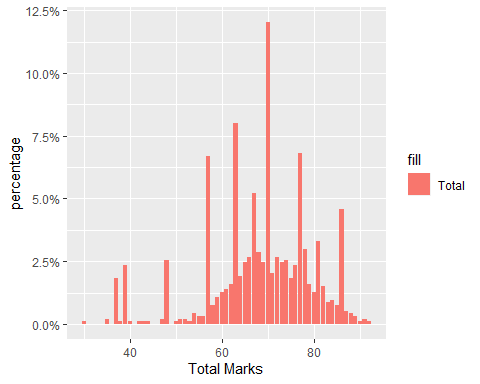
The bar plot for the frequencies is given that:

library(ggplot2)  
ggplot(studentsData, aes(x=Total,fill='Total')) +  
geom\_bar()

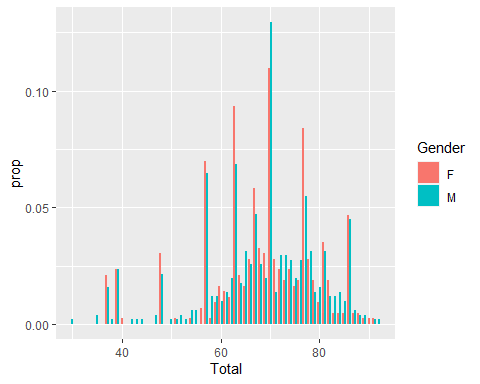


The Total marks barplot with the percentages of the marks is given that:

library(ggplot2)  
ggplot(studentsData, aes(x=Total)) +  
geom\_bar(aes(y=(..count..)/sum(..count..),fill='Total'))+  
scale\_y\_continuous(labels=scales::percent) +  
ylab('percentage')+  
xlab('Total Marks')

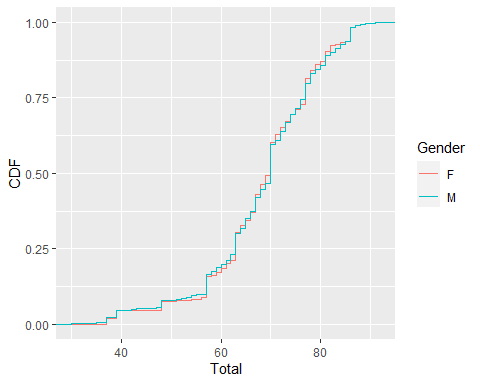
 PMF of Total marks is as:

ggplot(studentsData, aes(x=Total, fill=Gender)) +  
geom\_bar(aes(y=..prop..), position=position\_dodge2(preserve="single"))



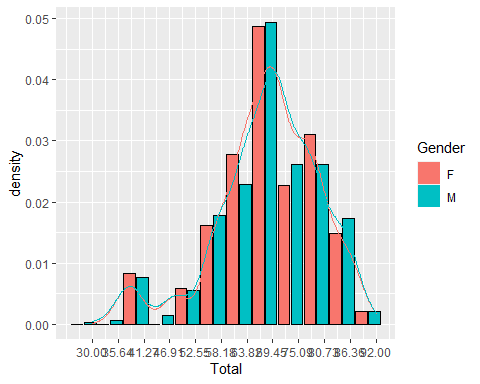
For CDF, We can compare the male and female samples by specifying a color=sex aesthetic:

ggplot(studentsData, aes(x=Total,color=Gender)) +  
stat\_ecdf() +  
ylab('CDF')



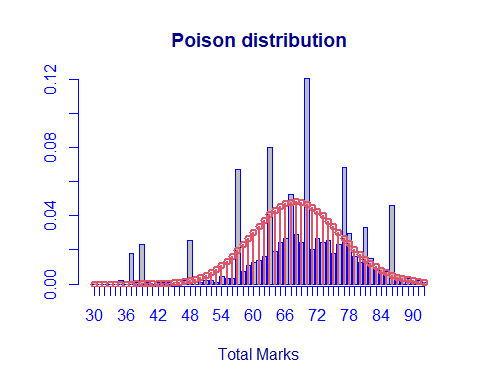
The density histogram is given below for the Total marks with respect to the Gender.

ggplot(studentsData, aes(x=Total, fill=Gender)) +  
geom\_histogram(aes(y=..density.., ), bins=12, color='black', position = position\_dodge2(preserve='single')) +  
scale\_x\_continuous(breaks=seq(min(studentsData$Total),max(studentsData$Total),length=12),  
labels = scales::number\_format(accuracy = 0.01))+  
geom\_density(aes(color=Gender), fill=NA)



Generalized Additive Models for Location, Scale and Shape (GAMLSS) analysis of different statistical models is given below:

dist.poison <- histDist(studentsData$Total, family=PO, nbins = 30, main="Poison distribution",xlab = "Total Marks")



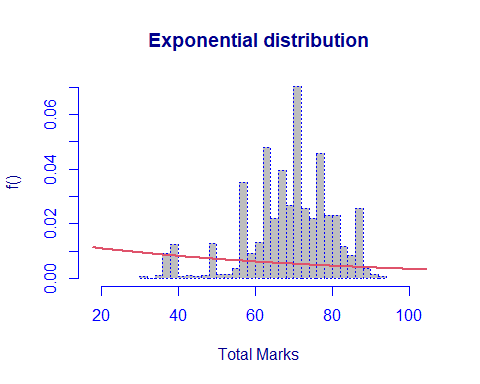
dist.poison$aic

## [1] 7557.037

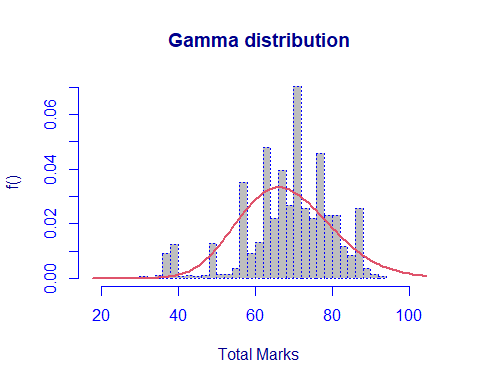
dist.poison$sbc

## [1] 7561.881

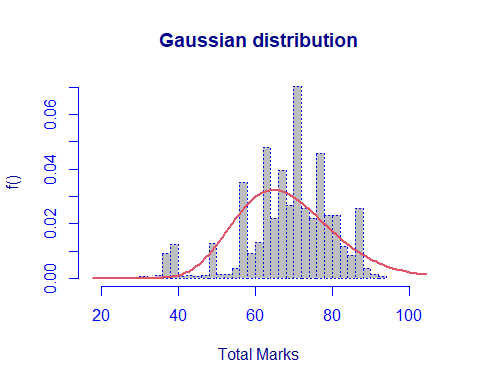
dist.expo <- histDist(studentsData$Total, family=EXP, nbins = 30, main="Exponential distribution",xlab = "Total Marks")



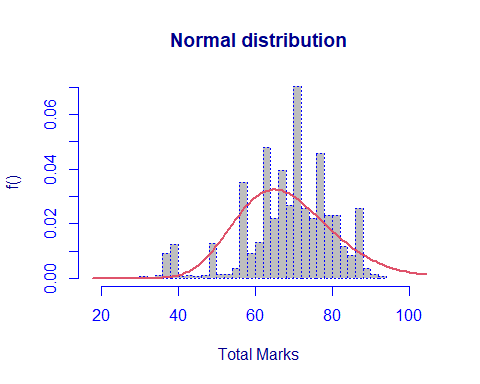
dist.gama <- histDist(studentsData$Total, family=GA, nbins = 30, main="Gamma distribution",xlab = "Total Marks")



dist.gaussian <- histDist(studentsData$Total, family=IG, nbins = 30, main="Gaussian distribution",xlab = "Total Marks")



dist.lognormal <- histDist(studentsData$Total, family=LOGNO, nbins = 30, main="Normal distribution",xlab = "Total Marks")



# {r} # dist.weibull <- histDist(studentsData$Sessional.Marks, family=WEI, nbins = 30, main="Weibull distribution",xlab = "Sessional Marks") # #

data.frame(row.names = c("Exponential", "Gamma", "Inverse Gaussian", "Log-Normal", "Poison"),  
 AIC=c(AIC(dist.expo), AIC(dist.gama), AIC(dist.gaussian), AIC(dist.lognormal), AIC(dist.poison)),  
 SBC=c(dist.expo$sbc, dist.gama$sbc, dist.gaussian$sbc, dist.lognormal$sbc, dist.poison$sbc)  
 )

## AIC SBC  
## Exponential 9813.390 9818.234  
## Gamma 7337.966 7347.656  
## Inverse Gaussian 7419.184 7428.874  
## Log-Normal 7411.225 7420.915  
## Poison 7557.037 7561.881

In the above table the Gamma distribution model is the best fit for our data due to the lowest values of the AIC and SBC.

## Likelihood-ratio test

In order to assess the goodness-of-fit, Likelihood-ratio test between the Gaussian model (under the null hypothesis) and the gamma model (under the alternative hypothesis) is computed.

LR.test(dist.gaussian,dist.gama)

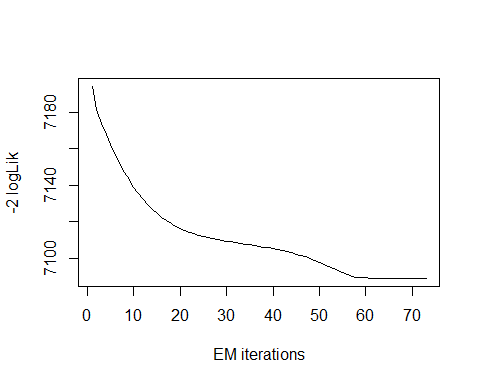
## Likelihood Ratio Test for nested GAMLSS models.   
## (No check whether the models are nested is performed).   
##   
## Null model: deviance= 7415.184 with 2 deg. of freedom   
## Altenative model: deviance= 7333.966 with 2 deg. of freedom   
##   
## LRT = 81.21811 with 0 deg. of freedom and p-value= 0

The p-values is 0: the null model, i.e. the Gaussian distribution, is rejected.

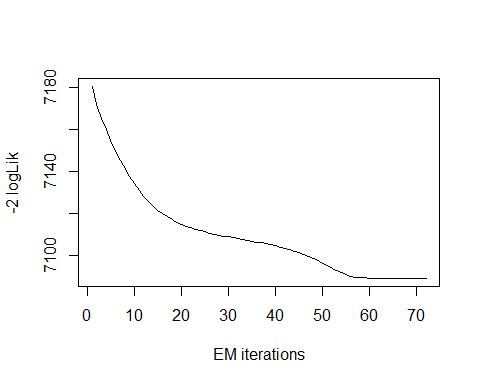
## Mixture Model of Gamma:

We can also fit mixture distributions, for example the Gamma ones:

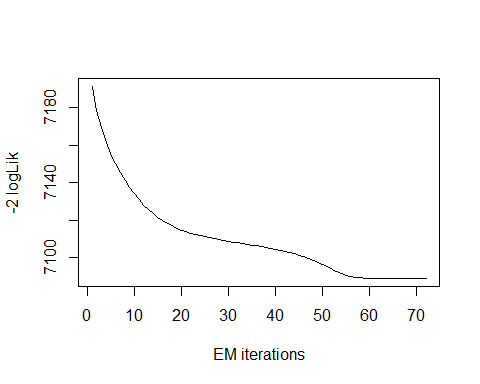
dist.gamma.mix <- gamlssMXfits(n = 5, Total~1, family = GA, K = 2,data = studentsData)



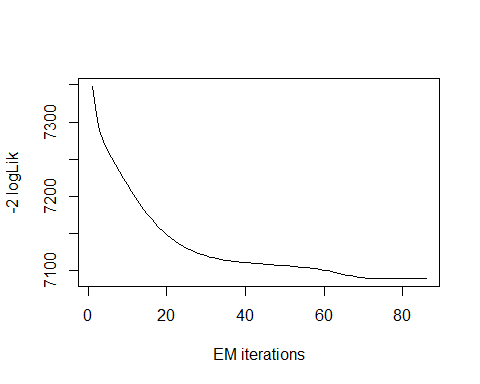
## model= 1



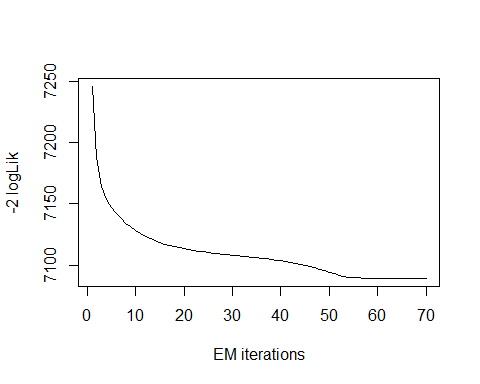
## model= 2



## model= 3



## model= 4



## model= 5

dist.gamma.mix$aic

## [1] 7099.079

dist.gamma.mix$sbc

## [1] 7123.303

We can see that with the mixture of two Gamma distributions the AIC and BIC values have improved, as they are lower than those obtained with the single Gamma distribution: AIC is now equal to 7099.079, while the previous value was 7337.966; SBC is now 7123.303, which is lower than 7347.656 , which was the previous value. Even the mixed gamma function model is much better than the chosen Poisson Model.

## Class:

The Class is the categorical variable of type Nominal. It consists of seven possible values ‘A+’,‘A’,‘B+’,‘B’,‘C’,‘D’ and ‘F’ .

summary(studentsData$Class)

## A A+ B B+ C D F   
## 350 140 133 147 96 30 43

The frequencies are given below:

table(studentsData$Class)

##   
## A A+ B B+ C D F   
## 350 140 133 147 96 30 43

frequencyClass <- table(studentsData$Class)

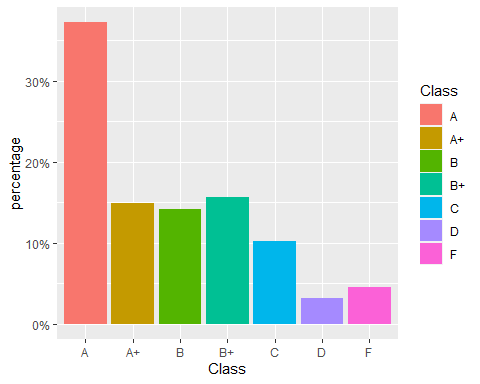
The percentage distribution of the dataset with respect to the Class can be calculated by multiply to 100.

prop.table(frequencyClass) \* 100

##   
## A A+ B B+ C D F   
## 37.273695 14.909478 14.164004 15.654952 10.223642 3.194888 4.579340

The above frequency clearly elaborate that the dataset is skewed towards the obove average performances and about 37 percent of students performances is A. The frequency in the percentages is shows in the below bar plot.

library(ggplot2)  
ggplot(studentsData, aes(x=Class)) +  
geom\_bar(aes(y=(..count..)/sum(..count..),fill=Class))+  
scale\_y\_continuous(labels=scales::percent) +  
ylab('percentage')



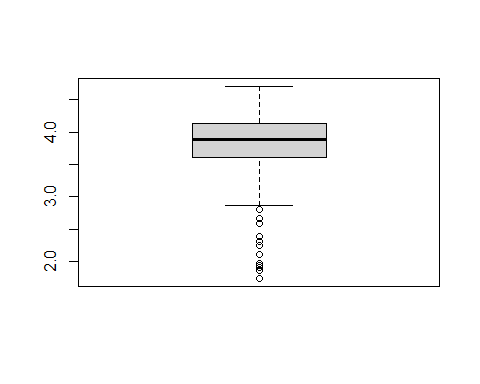
## CGPA

The CGPA is the Numerical variable of type decimal.It is calculated with the formula mentioned in the table 1 of this report.

summary(studentsData$CGPA)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 1.744 3.611 3.880 3.753 4.132 4.704

boxplot(studentsData$CGPA)



The frequencies of the CGPA are given below:

table(studentsData$CGPA)

##   
## 1.744 1.868 1.9176 1.9424 1.9672 2.11 2.248 2.317 2.386 2.593 2.662   
## 1 2 17 1 22 1 1 1 1 2 24   
## 2.8 2.869 2.938 3.007 3.076 3.145 3.214 3.283 3.352 3.421 3.5   
## 1 2 2 1 4 3 3 63 7 10 12   
## 3.537 3.574 3.611 3.648 3.685 3.722 3.759 3.796 3.833 3.88 3.916   
## 13 15 75 18 23 25 49 27 23 113 19   
## 3.952 3.988 4.024 4.06 4.096 4.132 4.168 4.204 4.25 4.287 4.324   
## 25 23 24 17 22 64 28 15 12 31 14   
## 4.361 4.398 4.435 4.472 4.509 4.546 4.583 4.63 4.667 4.704   
## 8 9 7 43 5 4 3 1 2 1

frequencyTotal <- table(studentsData$CGPA)

The percentage distribution of the dataset with respect to the Total marks can be calculated using the above formula.

prop.table(frequencyTotal) \* 100

##   
## 1.744 1.868 1.9176 1.9424 1.9672 2.11 2.248   
## 0.1064963 0.2129925 1.8104366 0.1064963 2.3429180 0.1064963 0.1064963   
## 2.317 2.386 2.593 2.662 2.8 2.869 2.938   
## 0.1064963 0.1064963 0.2129925 2.5559105 0.1064963 0.2129925 0.2129925   
## 3.007 3.076 3.145 3.214 3.283 3.352 3.421   
## 0.1064963 0.4259851 0.3194888 0.3194888 6.7092652 0.7454739 1.0649627   
## 3.5 3.537 3.574 3.611 3.648 3.685 3.722   
## 1.2779553 1.3844515 1.5974441 7.9872204 1.9169329 2.4494143 2.6624068   
## 3.759 3.796 3.833 3.88 3.916 3.952 3.988   
## 5.2183174 2.8753994 2.4494143 12.0340788 2.0234292 2.6624068 2.4494143   
## 4.024 4.06 4.096 4.132 4.168 4.204 4.25   
## 2.5559105 1.8104366 2.3429180 6.8157614 2.9818956 1.5974441 1.2779553   
## 4.287 4.324 4.361 4.398 4.435 4.472 4.509   
## 3.3013845 1.4909478 0.8519702 0.9584665 0.7454739 4.5793397 0.5324814   
## 4.546 4.583 4.63 4.667 4.704   
## 0.4259851 0.3194888 0.1064963 0.2129925 0.1064963

The above frequency in represents the percentages of the CGPA in the dataset.

We can check the index of the most frequent marks as follows:

which.max(frequencyTotal)

## 3.88   
## 32

This returns the the position of the max (32) and the name of the maximum value 70. We can obtain the maximum number of times a number repeats is as follows:

max(frequencyTotal)

## [1] 113

Similarly with the least frequent item:

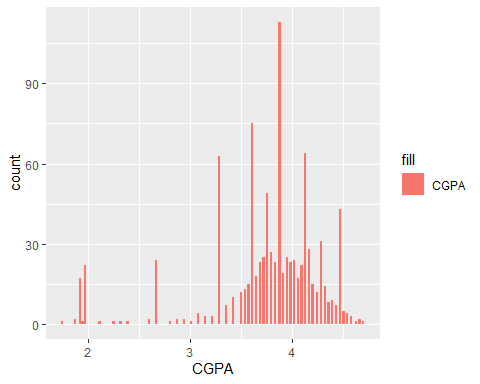
frequencyTotal[which.min(frequencyTotal)]

## 1.744   
## 1

The frequency table is already telling us something about the sample: • The most frequent CGPA is 3.88 it implies that most of the students performs above average marks as total; • The least frequent Final Exam marks is 1.744 ;

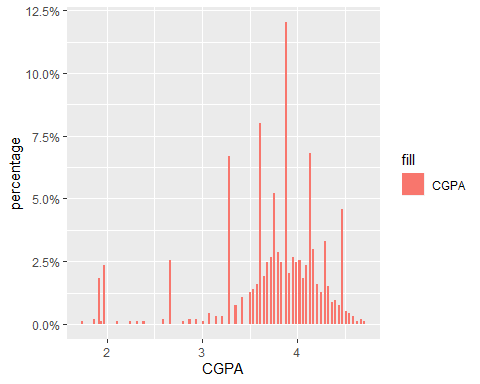
The bar plot for the frequencies is given that:

library(ggplot2)  
ggplot(studentsData, aes(x=CGPA,fill='CGPA')) +  
geom\_bar()

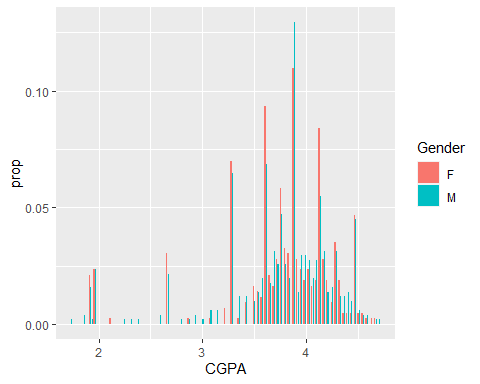


The CGPA barplot with the percentages of the marks is given that:

library(ggplot2)  
ggplot(studentsData, aes(x=CGPA)) +  
geom\_bar(aes(y=(..count..)/sum(..count..),fill='CGPA'))+  
scale\_y\_continuous(labels=scales::percent) +  
ylab('percentage')+  
xlab('CGPA')

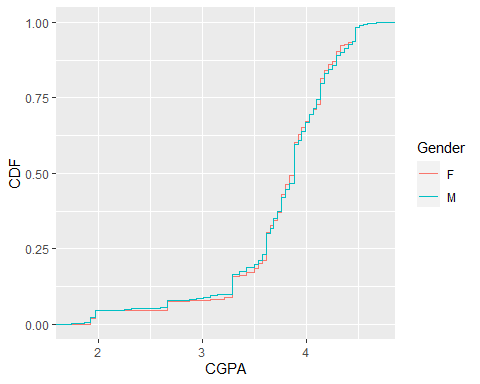
 PMF of CGPA is as:

ggplot(studentsData, aes(x=CGPA, fill=Gender)) +  
geom\_bar(aes(y=..prop..), position=position\_dodge2(preserve="single"))



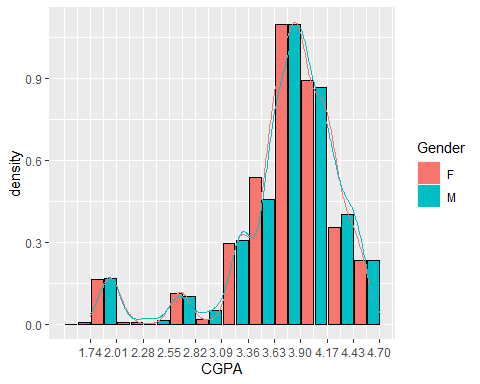
For CDF, We can compare the male and female samples by specifying a color=sex aesthetic:

ggplot(studentsData, aes(x=CGPA,color=Gender)) +  
stat\_ecdf() +  
ylab('CDF')



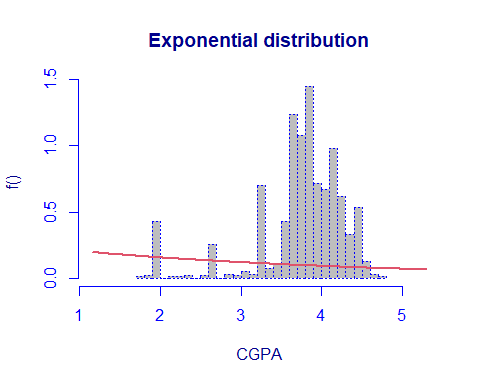
The density histogram is given below for the CGPA with respect to the Gender.

ggplot(studentsData, aes(x=CGPA, fill=Gender)) +  
geom\_histogram(aes(y=..density.., ), bins=12, color='black', position = position\_dodge2(preserve='single')) +  
scale\_x\_continuous(breaks=seq(min(studentsData$CGPA),max(studentsData$CGPA),length=12),  
labels = scales::number\_format(accuracy = 0.01))+  
geom\_density(aes(color=Gender), fill=NA)

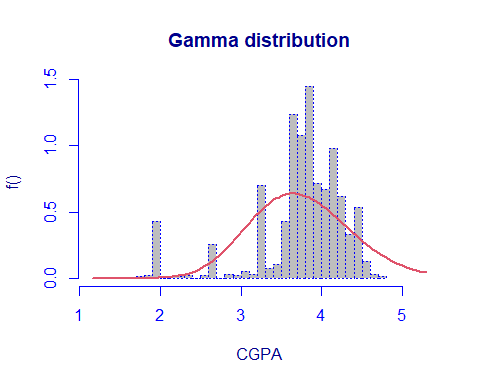


Generalized Additive Models for Location, Scale and Shape (GAMLSS) analysis of different statistical models is given below:

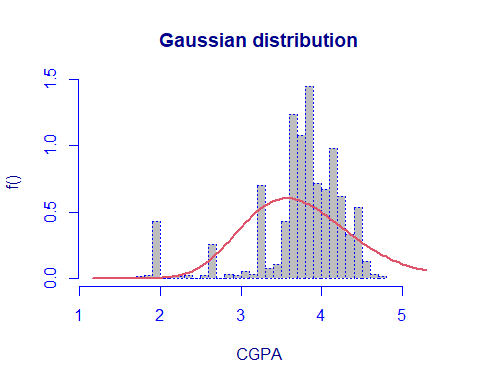
dist.expo <- histDist(studentsData$CGPA, family=EXP, nbins = 30, main="Exponential distribution",xlab = "CGPA")



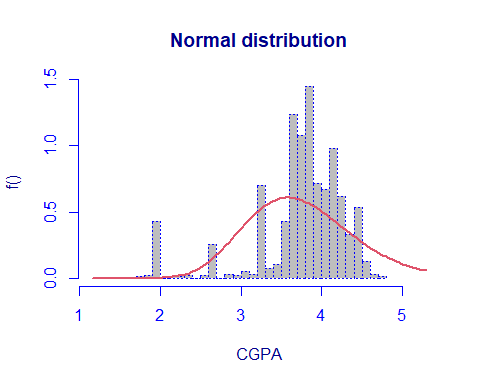
dist.gama <- histDist(studentsData$CGPA, family=GA, nbins = 30, main="Gamma distribution",xlab = "CGPA")



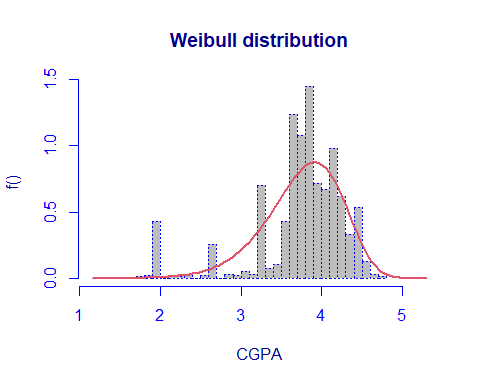
dist.gaussian <- histDist(studentsData$CGPA, family=IG, nbins = 30, main="Gaussian distribution",xlab = "CGPA")



dist.lognormal <- histDist(studentsData$CGPA, family=LOGNO, nbins = 30, main="Normal distribution",xlab = "CGPA")



dist.weibull <- histDist(studentsData$CGPA, family=WEI, nbins = 30, main="Weibull distribution",xlab = "CGPA")



data.frame(row.names = c("Exponential", "Gamma", "Inverse Gaussian", "Log-Normal", "Weibull"),  
 AIC=c(AIC(dist.expo), AIC(dist.gama), AIC(dist.gaussian), AIC(dist.lognormal), AIC(dist.weibull)),  
 SBC=c(dist.expo$sbc, dist.gama$sbc, dist.gaussian$sbc, dist.lognormal$sbc, dist.weibull$sbc)  
 )

## AIC SBC  
## Exponential 4363.801 4368.646  
## Gamma 1789.262 1798.951  
## Inverse Gaussian 1922.308 1931.998  
## Log-Normal 1906.523 1916.212  
## Weibull 1351.009 1360.698

In the above table the Weibull distribution model is the best fit for our data due to the lowest values of the AIC and SBC.

## Likelihood-ratio test

In order to assess the goodness-of-fit, Likelihood-ratio test between the gamma model (under the null hypothesis) and the weibull model (under the alternative hypothesis) is computed.

LR.test(dist.gama,dist.weibull)

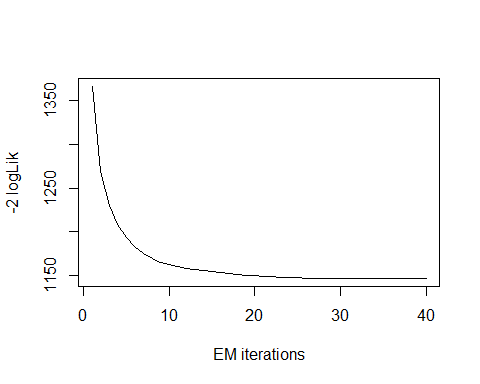
## Likelihood Ratio Test for nested GAMLSS models.   
## (No check whether the models are nested is performed).   
##   
## Null model: deviance= 1785.262 with 2 deg. of freedom   
## Altenative model: deviance= 1347.009 with 2 deg. of freedom   
##   
## LRT = 438.2529 with 0 deg. of freedom and p-value= 0

The p-values is 0: the null model, i.e. the Gamma distribution, is rejected.

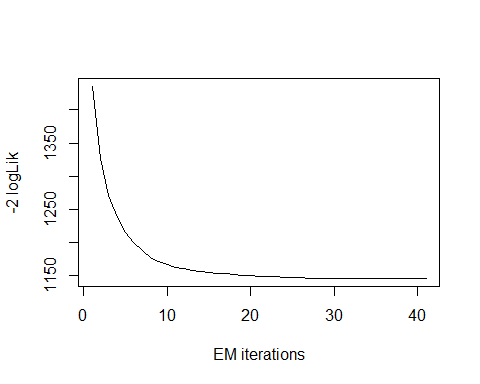
## Mixture Model of Gamma:

We can also fit mixture distributions, for example the Gamma ones:

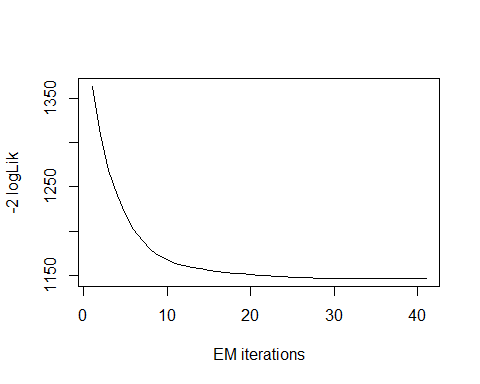
dist.gamma.mix <- gamlssMXfits(n = 5, CGPA~1, family = GA, K = 2,data = studentsData)



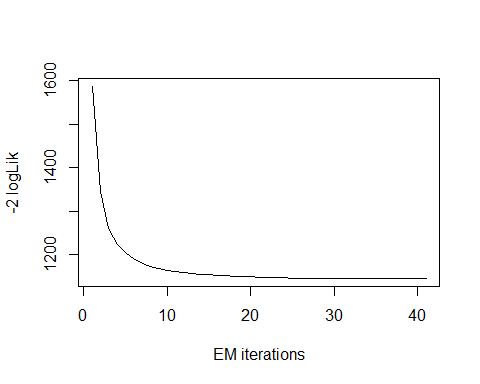
## model= 1



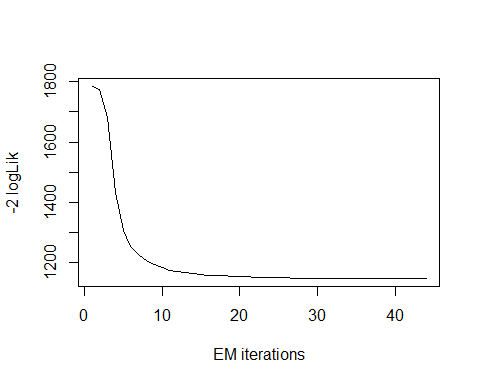
## model= 2



## model= 3



## model= 4



## model= 5

dist.gamma.mix$aic

## [1] 1156.329

dist.gamma.mix$sbc

## [1] 1180.553

We can see that with the mixture of two Gamma distributions the AIC and BIC values have improved, as they are lower than those obtained with the single Gamma distribution: AIC is now equal to 1156.329, while the previous value was 1789.262; SBC is now 1180.553, which is lower than 1798.951, which was the previous value. Even the mixed gamma function model is much better than the chosen Weibull Model.

Principle Component Analysis:

Principal component analysis (PCA) allows us to summarize and to visualize the information in a data set containing statistical units described by multiple correlated quantitative variables. Each variable could be considered as a different dimension.

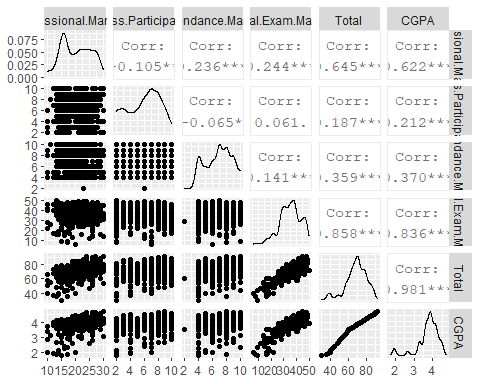
PCA is used to extract the important information from a multivariate data set and to express this information as a set of few new variables called principal components (PCs). These new variables correspond to a linear combination of the originals. The number of PCs is less than or equal to the number of original variables.

Before extracting the PCAs first we need to analysis the correlation between the numerical attributes.

library(GGally)

## Registered S3 method overwritten by 'GGally':  
## method from   
## +.gg ggplot2

studentsData\_PCA<-data.frame(studentsData[,c(-1,-2,-3,-6,-7,-8,-13)])  
  
ggpairs(studentsData\_PCA)

 The plot above shows:

All scatterplots in the bottom part; Density estimates of the univariate samples on the diagonal; Pearson Correlation coefficients in the upper part;

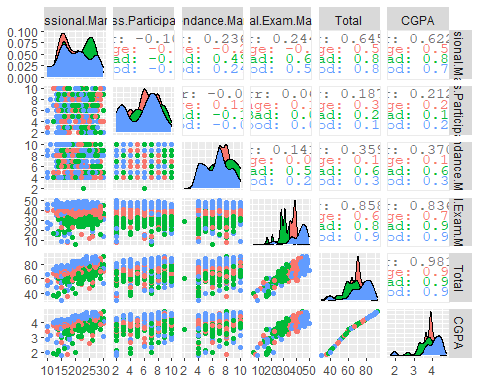
The stars next to the correlation values indicate whether the correlation is significant. In particular: One star means that the p-value is in the range [0.01,0.05]; Two stars mean that the p-value is in the range [0.001,0.01]; Three stars mean that the p-value is in the range[0,0.001];

Hence, we need at least one star to say that the observed correlation is significant.

From the analysis of the plot above, we can see that the strongest correlation is between Sessional Marks and Total Marks; and also with the final exam marks and total marks while there is a moderate correlation between others variables. While the corelation between class participation marks and final exam marks have no significance corelation.

If we consider the financial condition of the students then the below plot shows that:

ggpairs(studentsData, #dataset  
 columns = c(4,5,9,10,11,12), #select only the first four columns (exclude Species)   
 aes(color=Financial.Condition)) #color points by species

 Lets throw the updated, numeric data only dataset into the pca function, prcomp. Here, we used the scaling which means to standardize the variance across variables.

pca <- prcomp(studentsData\_PCA, center = TRUE,scale. = TRUE)

As, the PCA goal is to reduce the dimension and remove those PCs which have less meaningful variance.

summary(pca)

## Importance of components:  
## PC1 PC2 PC3 PC4 PC5 PC6  
## Standard deviation 1.8180 1.0638 0.9166 0.8350 0.15991 2.346e-15  
## Proportion of Variance 0.5509 0.1886 0.1400 0.1162 0.00426 0.000e+00  
## Cumulative Proportion 0.5509 0.7395 0.8795 0.9957 1.00000 1.000e+00

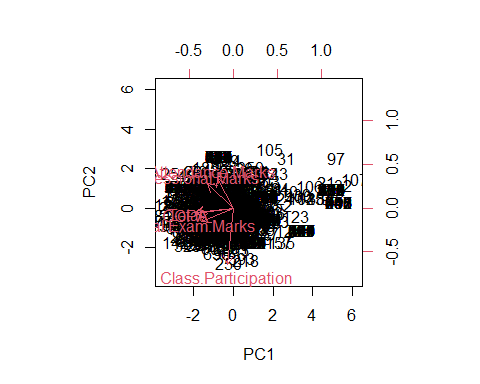
pca$rotation[, 1:6]

## PC1 PC2 PC3 PC4  
## Sessional.Marks -0.36809906 0.35632669 -0.12374264 0.752258747  
## Class.Participation -0.08280623 -0.80287056 0.46051681 0.315167132  
## Attendance.Marks -0.23561157 0.42682277 0.81740942 -0.264293179  
## Final.Exam.Marks -0.45608373 -0.18844187 -0.31788496 -0.514649376  
## Total -0.54711344 -0.06418191 -0.05543416 -0.004903710  
## CGPA -0.54294374 -0.08138075 -0.01816818 -0.006127549  
## PC5 PC6  
## Sessional.Marks 0.20164949 3.400976e-01  
## Class.Participation 0.11967423 1.510431e-01  
## Attendance.Marks 0.09568888 1.230614e-01  
## Final.Exam.Marks 0.32538702 5.335595e-01  
## Total 0.36298877 -7.494596e-01  
## CGPA -0.83559658 1.665335e-16

We have to see the proportion of variance in order to reduced the dimension of the PCs. It is clearly from the above table that we can consider the first four principle components to work with because they have some significant variance proportions.

We can use the biplot to represent at the same time original variables, principal components and the sample units (PC scores):

biplot(pca, scale=0)



From the above plot we can see that there is the high correlation between Total, Final exam marks, attendance marks and Sessional Marks.

But, There are several criteria to do so. These are all heuristic rules rather than rigorous methodologies. The reason is that PCA is completely distribution free. We can choose PCs based on:

Kaiser’s rule: omit the PCs containing less information than the mean information per PC;

pca.var = (pca$sdev)^2  
round(pca.var,3)

## [1] 3.305 1.132 0.840 0.697 0.026 0.000

The rule suggests retaining the principal components with variance greater than 1, reason why we can retain the first 2 PCs.

cumulative proportion of variance explained: extract the first PCs which are able to explain a substantial proportion of variance explained;

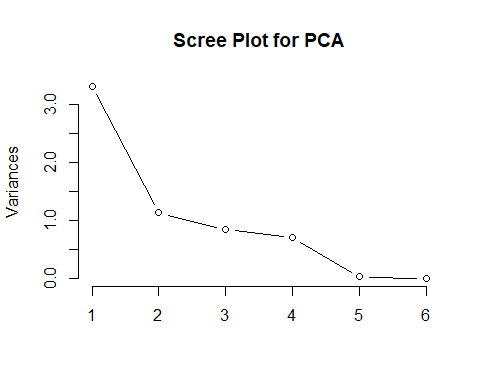
pve = pca.var/sum(pca.var)  
 round(pve,3) \*100

## [1] 55.1 18.9 14.0 11.6 0.4 0.0

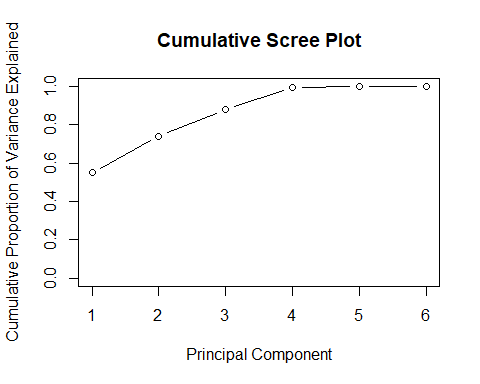
According to the CPVE we have to retain as many PCs as needed to explain at least the 80% of the total variance, hence we have to retain the first 4 PCs.

scree plot: retain as many components as until a significant jump on the scree-plot appears;

screeplot(pca, npcs = length(pca$sdev), type = "lines", main = "Scree Plot for PCA")



plot(cumsum(pve), xlab="Principal Component", ylab="Cumulative Proportion of Variance Explained",ylim=c(0,1),type='b', main= "Cumulative Scree Plot")



## PCA result

The cumulative PVE rule suggests retaining the first Four principal components, the Kaiser’s rule suggest retaining the first two principal components while the Scree plot provide a much clear result. The reduction from d=6 original variables to 4 PCs.

Cluster Analysis

library(RCurl)

# Collected data from github repository  
x <- getURL("https://raw.githubusercontent.com/adeelahmedcsku/student-dataset/main/20-01-2020-data%20developed%20records%20students-changes%20in%20grading%20system%20-%20data%20developed%20records%20students.csv")  
studentsData <- read.csv(text = x)

studentsData$Gender =as.numeric(factor(studentsData$Gender))  
str(studentsData$Gender)

## num [1:939] 2 2 2 2 2 2 2 1 1 1 ...

Factoring the program attribute:

studentsData$Program =as.numeric(factor(studentsData$Program))  
str(studentsData$Program)

## num [1:939] 2 2 2 2 2 2 2 2 2 2 ...

Factoring the Is.Research.Oriented attribute:

studentsData$Is.Research.Oriented = as.numeric(factor(studentsData$Is.Research.Oriented))  
str(studentsData$Is.Research.Oriented)

## num [1:939] 2 2 1 1 1 1 1 2 2 2 ...

Factoring the Student.Type attribute:

studentsData$Student.Type = as.numeric(factor(studentsData$Student.Type))  
str(studentsData$Student.Type)

## num [1:939] 2 1 2 2 1 1 2 2 2 1 ...

Factoring the Financial.Condition attribute:

studentsData$Financial.Condition = as.numeric(factor(studentsData$Financial.Condition))  
str(studentsData$Financial.Condition)

## num [1:939] 3 2 3 1 3 3 2 1 2 1 ...

Factoring the Class attribute:

studentsData$Class = as.numeric(factor(studentsData$Class))  
str(studentsData$Class)

## num [1:939] 2 1 3 1 2 2 5 1 6 3 ...

Cluster Analysis

Clustering is one of the important data mining methods for discovering knowledge in multidimensional data. The goal of clustering is to identify pattern or groups of similar objects within a data set of interest. In the literature, it is referred as “pattern recognition” or “unsupervised machine learning” - “unsupervised” because we are not guided by a priori ideas of which variables or samples belong in which clusters. “Learning” because the machine algorithm “learns” how to cluster.

The Clustering involves several steps. The first step is to calculate the dissimilarity matrix amoung the cluster units.

The value of distance measures is intimately related to the scale on which measurements are made. Therefore, variables are often scaled (i.e. standardized) before measuring the inter-observation dissimilarities.

studentsData.scaled <- scale(studentsData)

The Euclidean distance of the dataset is show below.

dist.eucl <- dist(studentsData.scaled, method = "euclidean")  
dist.eucl.matrix<-round(as.matrix(dist.eucl)[1:13, 1:13], 2)  
rownames(dist.eucl.matrix) <- c("G","B.Y","Pro","S.M","C.P","R.O","S.T","F.C","A.M","F.E.M","Total","CGPA","Class")  
colnames(dist.eucl.matrix) <- c("G","B.Y","Pro","S.M","C.P","R.O","S.T","F.C","A.M","F.E.M","Total","CGPA","Class")  
dist.eucl.matrix

## G B.Y Pro S.M C.P R.O S.T F.C A.M F.E.M Total CGPA Class  
## G 0.00 3.50 4.66 3.80 3.28 3.42 5.66 4.69 7.40 5.69 4.75 2.57 5.01  
## B.Y 3.50 0.00 5.24 4.06 3.14 3.26 5.11 4.57 7.39 4.18 2.52 3.43 4.98  
## Pro 4.66 5.24 0.00 3.84 4.08 4.53 3.10 4.46 4.70 4.87 5.68 4.45 3.02  
## S.M 3.80 4.06 3.84 0.00 3.96 3.96 4.48 3.89 6.57 4.93 4.74 3.23 5.14  
## C.P 3.28 3.14 4.08 3.96 0.00 2.50 4.69 4.82 7.23 4.76 4.43 3.53 4.28  
## R.O 3.42 3.26 4.53 3.96 2.50 0.00 5.67 5.78 8.08 5.77 4.51 4.73 5.17  
## S.T 5.66 5.11 3.10 4.48 4.69 5.67 0.00 4.58 4.16 4.08 5.21 4.63 4.34  
## F.C 4.69 4.57 4.46 3.89 4.82 5.78 4.58 0.00 5.06 2.71 4.55 3.32 4.38  
## A.M 7.40 7.39 4.70 6.57 7.23 8.08 4.16 5.06 0.00 4.64 6.68 6.25 4.53  
## F.E.M 5.69 4.18 4.87 4.93 4.76 5.77 4.08 2.71 4.64 0.00 3.79 4.46 3.99  
## Total 4.75 2.52 5.68 4.74 4.43 4.51 5.21 4.55 6.68 3.79 0.00 4.31 5.54  
## CGPA 2.57 3.43 4.45 3.23 3.53 4.73 4.63 3.32 6.25 4.46 4.31 0.00 4.53  
## Class 5.01 4.98 3.02 5.14 4.28 5.17 4.34 4.38 4.53 3.99 5.54 4.53 0.00

The most dissimilarity we can see in the table is between Attendance Marks(A.M) and Research Oriented (R.O) which is 8.08. while the most similar is class participation marks and research oriented attribute which 2.50.

We can also try to compute the Manhattan distance:

dist.man <- dist(studentsData.scaled, method = "manhattan")  
dist.man.matrix<-round(as.matrix(dist.man)[1:13, 1:13], 2)  
rownames(dist.man.matrix) <-c("G","B.Y","Pro","S.M","C.P","R.O","S.T","F.C","A.M","F.E.M","Total","CGPA","Class")  
colnames(dist.man.matrix)<-c("G","B.Y","Pro","S.M","C.P","R.O","S.T","F.C","A.M","F.E.M","Total","CGPA","Class")  
dist.man.matrix

## G B.Y Pro S.M C.P R.O S.T F.C A.M F.E.M Total CGPA  
## G 0.00 9.49 11.99 9.39 7.10 7.28 15.32 12.01 20.56 15.45 13.88 6.97  
## B.Y 9.49 0.00 14.95 9.84 7.56 8.27 14.86 12.17 21.85 11.86 5.17 6.06  
## Pro 11.99 14.95 0.00 8.82 9.97 12.32 7.63 11.87 13.45 11.76 16.69 12.01  
## S.M 9.39 9.84 8.82 0.00 9.70 9.72 11.76 9.07 19.72 13.33 13.18 7.15  
## C.P 7.10 7.56 9.97 9.70 0.00 5.38 13.30 13.98 22.54 13.43 12.72 9.04  
## R.O 7.28 8.27 12.32 9.72 5.38 0.00 15.86 17.22 25.77 16.86 12.66 13.25  
## S.T 15.32 14.86 7.63 11.76 13.30 15.86 0.00 12.95 10.99 11.21 15.36 12.30  
## F.C 12.01 12.17 11.87 9.07 13.98 17.22 12.95 0.00 12.23 6.52 10.64 8.23  
## A.M 20.56 21.85 13.45 19.72 22.54 25.77 10.99 12.23 0.00 12.92 18.34 16.67  
## F.E.M 15.45 11.86 11.76 13.33 13.43 16.86 11.21 6.52 12.92 0.00 9.34 12.68  
## Total 13.88 5.17 16.69 13.18 12.72 12.66 15.36 10.64 18.34 9.34 0.00 9.43  
## CGPA 6.97 6.06 12.01 7.15 9.04 13.25 12.30 8.23 16.67 12.68 9.43 0.00  
## Class 12.73 12.56 5.61 14.43 10.70 13.94 12.16 10.95 11.83 7.90 14.30 11.87  
## Class  
## G 12.73  
## B.Y 12.56  
## Pro 5.61  
## S.M 14.43  
## C.P 10.70  
## R.O 13.94  
## S.T 12.16  
## F.C 10.95  
## A.M 11.83  
## F.E.M 7.90  
## Total 14.30  
## CGPA 11.87  
## Class 0.00

The maximum distance is between Research Oriented and Attendance marks is maximum and minimum distance is between class participation and research oriented which is same as euclidean distance. So there is no difference in these two distance matrices we can say that there is no outlier in our dataset.

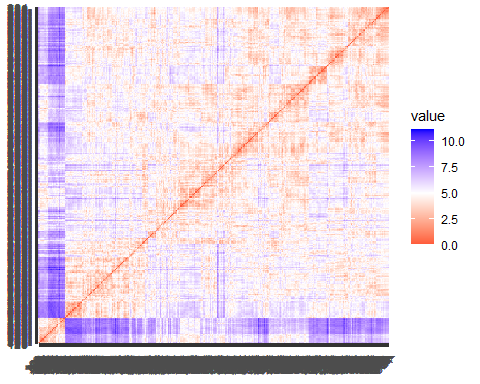
We can do the same observations by looking at the graphical visualization of the distance matrix:

library(factoextra)

## Loading required package: ggplot2

## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa

fviz\_dist(dist.eucl)

 The red represents high similarity and blue represents low dissimilarity.

# Hierarchical clustering method

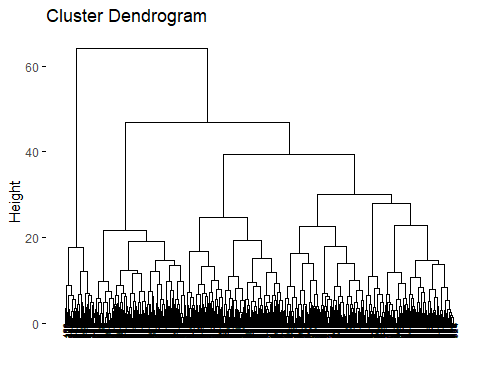
Hierarchical clustering is a method of CA which seeks to build a hierarchy of clusters. Hierarchical clustering methods do not require the number of clusters K as an input (i.e., to be fixed in advance).

Strategies for hierarchical clustering generally fall into two types: 1. Agglomerative hierarchical clustering 2. Divisive hierarchical clustering

#Agglomerative hierarchical clustering

## Using Euclidean Distance and Ward’s Linkage Methods:

res.hc <- hclust(d = dist.eucl, method = "ward.D2")  
fviz\_dend(res.hc, cex = 0.5)

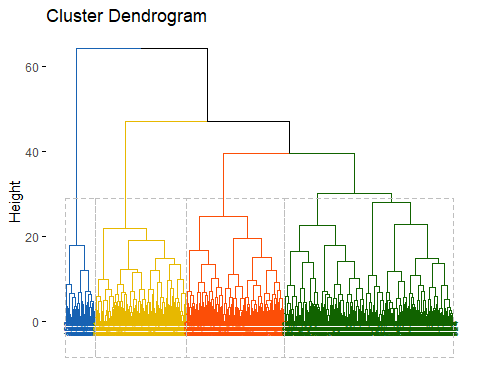
 The height of the fusion, provided on the vertical axis, indicates the (dis)similarity/distance between two objects/clusters. The higher the height of the fusion, the less similar the objects are. This height is known as the cophenetic distance between the two objects.

# Compute cophentic distance  
res.coph <- cophenetic(res.hc)

# Correlation between cophenetic distance and  
# the original distance  
cor(dist.eucl, res.coph)

## [1] 0.6551251

grp <- cutree(res.hc, k = 4)  
 fviz\_dend(res.hc, k = 4, cex = 0.5, k\_colors = c("#1762b3", "#E7B800", "#FC4E07","#116300"), color\_labels\_by\_k = TRUE,rect = TRUE)

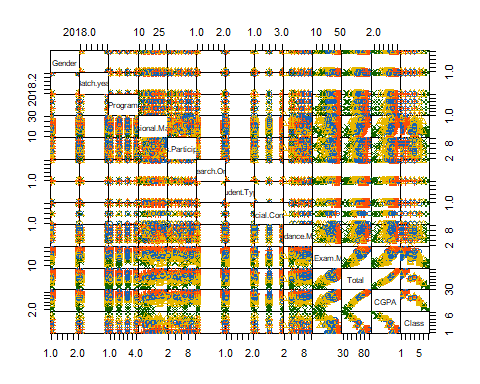
 The last cluster is bigger than the first two clusters

table(grp)

## grp  
## 1 2 3 4   
## 410 236 221 72

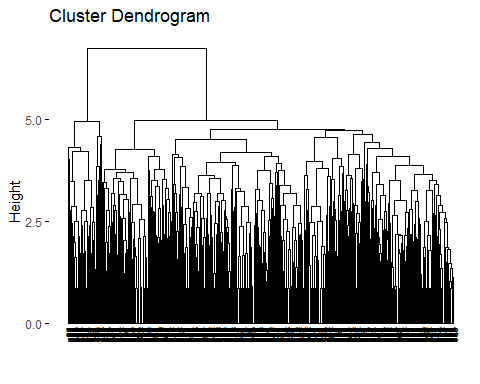
We can visualize the clustering results in the original space via the matrix of pairwise scatterplots:

pairs(studentsData, gap=0, pch=grp, col=c("#1762b3", "#E7B800", "#FC4E07","#116300")[grp])



## Using Euclidean Distance and Average Linkage Methods:

res.hc <- hclust(d = dist.eucl, method = "average")  
fviz\_dend(res.hc, cex = 0.5)



The height of the fusion, provided on the vertical axis, indicates the (dis)similarity/distance between two objects/clusters. The higher the height of the fusion, the less similar the objects are. This height is known as the cophenetic distance between the two objects.

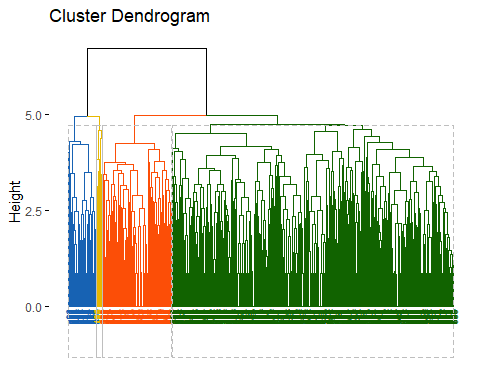
# Compute cophentic distance  
res.coph <- cophenetic(res.hc)

# Correlation between cophenetic distance and  
# the original distance  
cor(dist.eucl, res.coph)

## [1] 0.7236952

The correlation value is 0.723 which is very good with compare to the previous WARD.D2 method.

grp <- cutree(res.hc, k = 4)  
 fviz\_dend(res.hc, k = 4, cex = 0.5, k\_colors = c("#1762b3", "#E7B800", "#FC4E07","#116300"), color\_labels\_by\_k = TRUE,rect = TRUE)

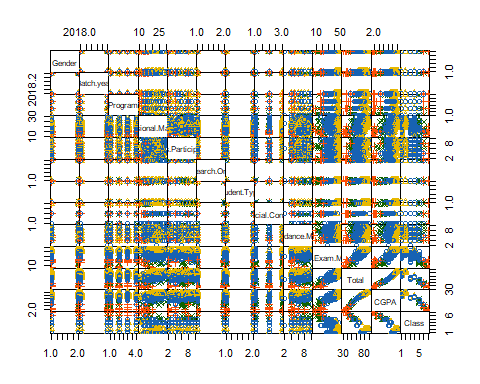
 The last cluster is bigger than the first two clusters

table(grp)

## grp  
## 1 2 3 4   
## 687 169 67 16

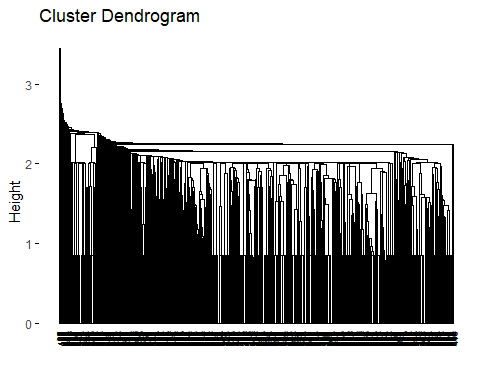
We can visualize the clustering results in the original space via the matrix of pairwise scatterplots:

pairs(studentsData, gap=0, pch=grp, col=c("#1762b3", "#E7B800", "#FC4E07","#116300")[grp])



## Using Euclidean Distance and Single Linkage Methods:

res.hc <- hclust(d = dist.eucl, method = "single")  
fviz\_dend(res.hc, cex = 0.5)



The height of the fusion, provided on the vertical axis, indicates the (dis)similarity/distance between two objects/clusters. The higher the height of the fusion, the less similar the objects are. This height is known as the cophenetic distance between the two objects.

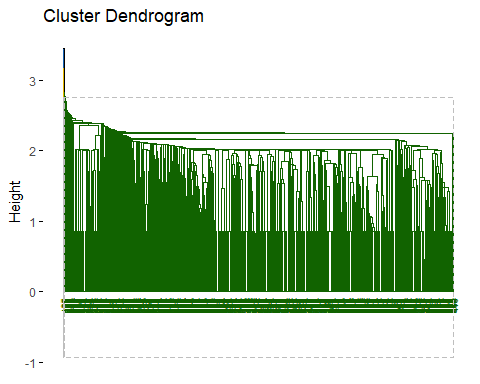
# Compute cophentic distance  
res.coph <- cophenetic(res.hc)

# Correlation between cophenetic distance and  
# the original distance  
cor(dist.eucl, res.coph)

## [1] 0.540343

The correlation value is 0.54 which is very bad with compare to the previous linkage method method.This means that this clustering method does not preserve the true original distances between units.

grp <- cutree(res.hc, k = 4)  
 fviz\_dend(res.hc, k = 4, cex = 0.5, k\_colors = c("#1762b3", "#E7B800", "#FC4E07","#116300"), color\_labels\_by\_k = TRUE,rect = TRUE)

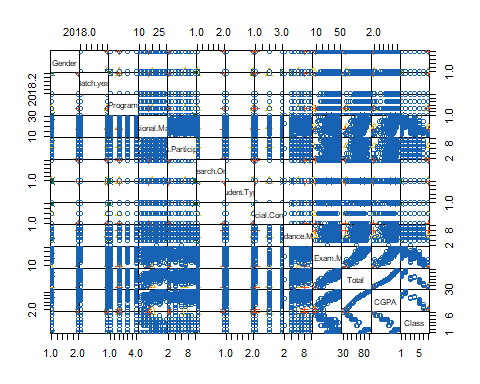
 As we can see, almost all units are grouped in a single cluster and we can check this also through the R code which gives us the clusters size:

table(grp)

## grp  
## 1 2 3 4   
## 936 1 1 1

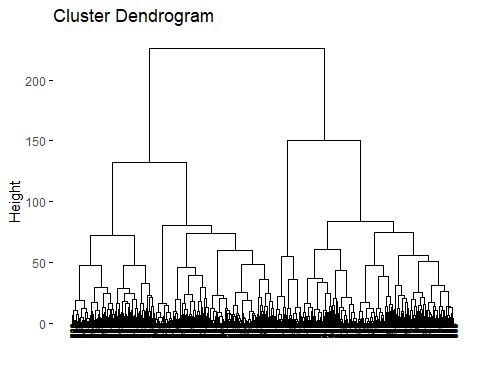
We can visualize the clustering results in the original space via the matrix of pairwise scatterplots:

pairs(studentsData, gap=0, pch=grp, col=c("#1762b3", "#E7B800", "#FC4E07","#116300")[grp])



## Using Manhattan Distance and WARD’S Linkage Methods:

res.hc <- hclust(d = dist.man, method = "ward.D2")  
fviz\_dend(res.hc, cex = 0.5)



The height of the fusion, provided on the vertical axis, indicates the (dis)similarity/distance between two objects/clusters. The higher the height of the fusion, the less similar the objects are. This height is known as the cophenetic distance between the two objects.

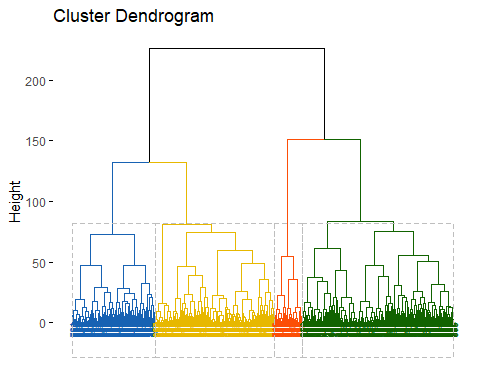
# Compute cophentic distance  
res.coph <- cophenetic(res.hc)

# Correlation between cophenetic distance and  
# the original distance  
cor(dist.eucl, res.coph)

## [1] 0.4771551

The correlation value is 0.47 which is very low it means this clustering method doesnot preserve the orginal distances.

grp <- cutree(res.hc, k = 4)  
 fviz\_dend(res.hc, k = 4, cex = 0.5, k\_colors = c("#1762b3", "#E7B800", "#FC4E07","#116300"), color\_labels\_by\_k = TRUE,rect = TRUE)

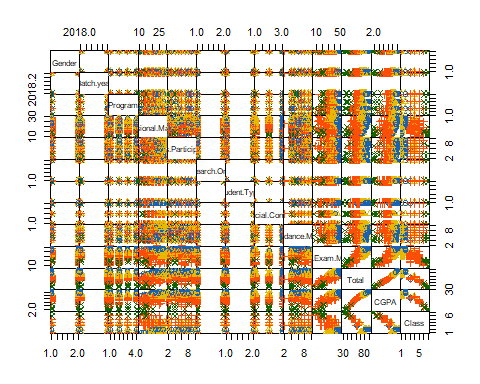
 As we can see, almost all units are grouped in a single cluster and we can check this also through the R code which gives us the clusters size:

table(grp)

## grp  
## 1 2 3 4   
## 204 294 373 68

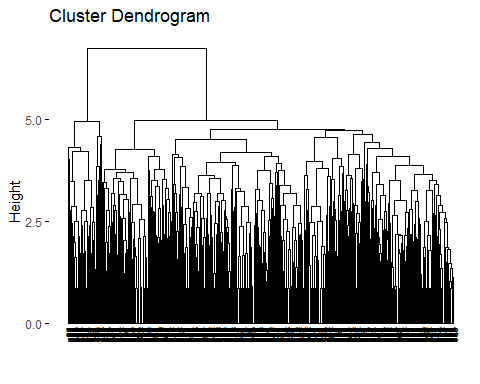
We can visualize the clustering results in the original space via the matrix of pairwise scatterplots:

pairs(studentsData, gap=0, pch=grp, col=c("#1762b3", "#E7B800", "#FC4E07","#116300")[grp])



## Using Manhattan Distance and Average Linkage Methods:

res.hc <- hclust(d = dist.eucl, method = "average")  
fviz\_dend(res.hc, cex = 0.5)



The height of the fusion, provided on the vertical axis, indicates the (dis)similarity/distance between two objects/clusters. The higher the height of the fusion, the less similar the objects are. This height is known as the cophenetic distance between the two objects.

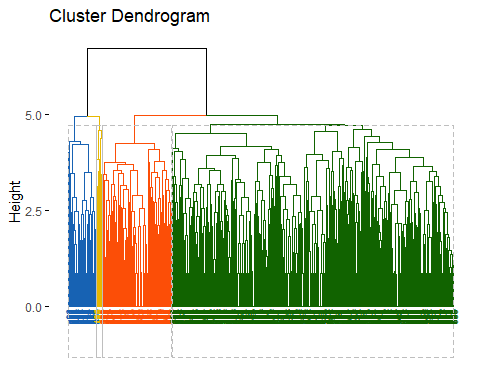
# Compute cophentic distance  
res.coph <- cophenetic(res.hc)

# Correlation between cophenetic distance and  
# the original distance  
cor(dist.eucl, res.coph)

## [1] 0.7236952

The correlation value is 0.723 which is same as the Euclidean distance using the average linkage method and its the best

grp <- cutree(res.hc, k = 4)  
 fviz\_dend(res.hc, k = 4, cex = 0.5, k\_colors = c("#1762b3", "#E7B800", "#FC4E07","#116300"), color\_labels\_by\_k = TRUE,rect = TRUE)

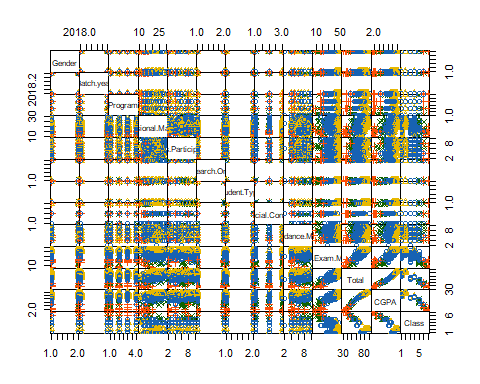
 As we can see, almost all units are grouped in a single cluster and we can check this also through the R code which gives us the clusters size:

table(grp)

## grp  
## 1 2 3 4   
## 687 169 67 16

We can visualize the clustering results in the original space via the matrix of pairwise scatterplots:

pairs(studentsData, gap=0, pch=grp, col=c("#1762b3", "#E7B800", "#FC4E07","#116300")[grp])



The above heirarchical agglomerative clustering analysis of the dataset suggest that the best combination is with the Euclidean distance with linkage method and same with Manhattan distance as well.

# Partitioning clustering

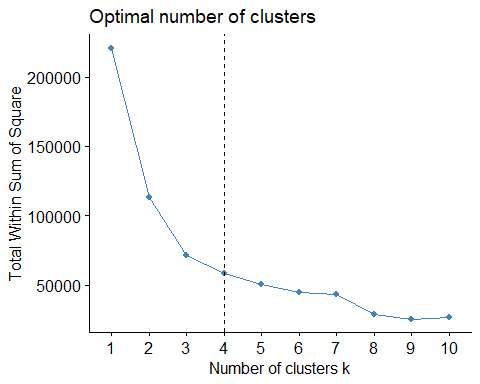
## K-Means Clustering:

K-means clustering is the most commonly used partitioning clustering algorithm. It classifies the n units to the K clusters such that units within the same cluster are as similar as possible (high cluster cohesion), whereas units from different clusters are as dissimilar as possible (high cluster separation).

The basic idea behind k-means clustering consists of defining clusters so that the total intra-cluster variation (known as total within-cluster variation) is minimized.

The K-means clustering requires the users to specify the number of clusters K to be generated.

library(factoextra)  
fviz\_nbclust(studentsData, kmeans, method = "wss") +  
geom\_vline(xintercept = 4, linetype = 2)



The plot above represents the variance within the clusters. It decreases as k increases, but it can be seen a bend (or “elbow”) at k = 4. This bend indicates that additional clusters beyond the fourth have little value. In the next section, we’ll classify the observations into 4 clusters.

The K-means clustering using the cluster size 4 is given below:

kmean.res <- kmeans(studentsData, 4, nstart = 25)

This time we have 4 clusters, the largest being the third, as we can see by the R code:

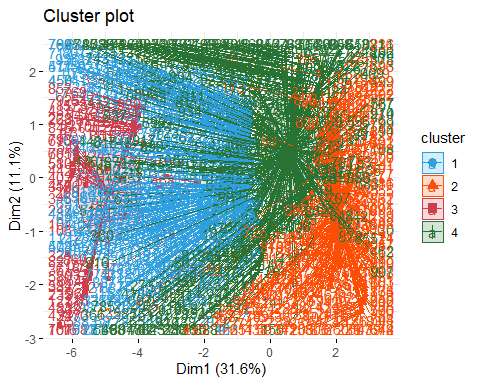
kmean.res$size

## [1] 254 233 80 372

# Print the results  
print(kmean.res)

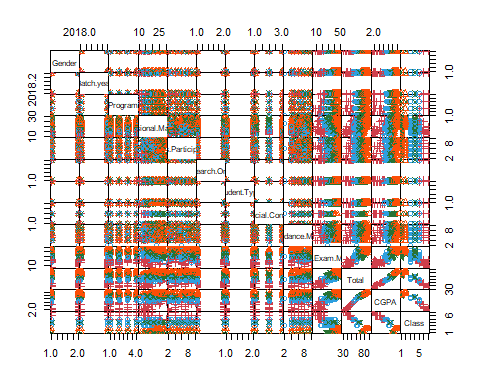
## K-means clustering with 4 clusters of sizes 254, 233, 80, 372  
##   
## Cluster means:  
## Gender Batch.year Program Sessional.Marks Class.Participation  
## 1 1.527559 2018.602 2.397638 17.31102 6.118110  
## 2 1.540773 2018.532 2.377682 23.08584 5.948498  
## 3 1.587500 2018.700 2.300000 13.90000 4.887500  
## 4 1.545699 2018.551 2.408602 21.82527 6.258065  
## Is.Research.Oriented Student.Type Financial.Condition Attendance.Marks  
## 1 1.500000 1.393701 1.909449 6.677165  
## 2 1.347639 1.351931 2.669528 7.180258  
## 3 1.600000 1.900000 2.275000 4.925000  
## 4 1.602151 1.580645 1.672043 7.408602  
## Final.Exam.Marks Total CGPA Class  
## 1 30.88976 60.99606 3.500343 3.826772  
## 2 44.80258 81.01717 4.285116 1.596567  
## 3 18.87500 42.58750 2.277600 6.450000  
## 4 35.44086 70.93280 3.909726 1.930108  
##   
## Clustering vector:  
## [1] 2 2 1 4 2 2 1 4 3 1 4 4 1 4 1 3 1 4 1 1 3 4 4 4 1 4 4 2 2 4 3 3 3 1 2 3 4  
## [38] 1 2 1 4 4 1 4 2 1 4 3 1 4 4 1 4 1 3 1 2 1 2 2 1 4 2 4 1 2 1 4 3 1 4 4 1 4  
## [75] 1 3 2 2 2 1 4 2 2 1 4 3 1 4 4 1 4 1 2 3 1 3 3 1 2 3 4 4 2 3 3 3 3 4 2 1 2  
## [112] 1 4 3 1 4 4 1 4 1 3 2 3 3 3 1 3 4 1 4 1 2 2 2 1 1 1 1 2 4 4 2 1 2 4 4 4 4  
## [149] 4 4 2 1 4 1 4 4 1 2 1 4 3 1 4 4 1 4 1 3 4 2 2 2 2 2 2 4 4 2 2 4 2 1 2 1 4  
## [186] 2 1 4 3 1 4 4 1 4 1 3 2 2 4 2 2 1 4 2 2 1 4 3 1 4 4 1 4 4 4 4 2 1 4 4 2 2  
## [223] 2 1 2 1 4 3 1 4 4 1 4 1 3 1 4 2 2 1 4 3 1 4 4 1 4 4 4 4 2 4 4 2 1 2 4 2 2  
## [260] 2 1 4 3 1 4 4 1 4 1 3 4 1 4 4 1 1 1 1 1 4 4 4 1 2 2 2 4 4 2 2 1 4 4 4 2 1  
## [297] 1 2 2 2 1 4 4 4 2 2 1 4 3 1 4 4 1 4 1 3 4 3 1 4 4 4 4 4 2 4 2 2 1 4 3 1 4  
## [334] 4 1 4 1 3 4 4 2 2 1 4 3 1 4 4 2 1 4 4 1 1 2 2 4 1 1 4 1 1 1 4 1 4 1 4 2 4  
## [371] 1 2 1 4 3 1 4 4 1 4 1 3 1 3 2 4 4 2 4 4 2 2 4 4 1 1 2 4 2 2 1 4 3 1 4 4 4  
## [408] 4 1 2 1 4 2 4 2 2 2 4 2 2 4 2 1 2 1 4 3 1 4 4 1 4 1 3 2 4 4 2 2 1 4 3 1 4  
## [445] 4 2 4 4 4 4 2 4 2 2 1 4 3 1 4 4 2 4 1 2 4 4 2 3 1 4 2 4 2 1 4 3 1 4 4 1 4  
## [482] 1 3 4 4 4 4 1 4 2 2 1 1 2 4 1 1 4 4 4 4 2 4 1 4 2 2 1 4 3 1 4 4 1 4 4 2 1  
## [519] 4 4 2 1 2 2 1 4 3 1 4 4 1 4 1 3 2 1 1 2 4 4 1 4 2 2 1 4 3 1 4 4 1 1 1 2 2  
## [556] 4 4 4 4 2 4 4 4 4 4 4 4 4 1 2 1 4 3 1 4 4 1 4 1 3 1 4 3 1 4 4 1 4 4 4 2 2  
## [593] 2 1 1 1 4 2 2 4 2 2 1 2 1 1 1 2 4 4 2 2 1 4 3 1 4 4 4 1 2 2 1 2 2 4 4 4 2  
## [630] 4 2 4 4 2 2 2 4 1 2 4 2 2 1 2 1 4 3 1 4 4 1 4 1 3 4 2 2 2 2 4 2 2 4 1 2 2  
## [667] 1 4 1 4 2 4 4 4 4 4 2 2 1 4 3 1 4 4 4 2 4 1 4 4 2 2 4 4 4 2 4 1 4 4 2 2 1  
## [704] 4 3 1 4 2 1 4 3 1 4 4 1 4 1 3 2 4 2 1 4 2 2 4 4 4 1 2 4 2 2 4 2 4 4 4 1 2  
## [741] 1 4 4 4 4 4 2 2 1 4 3 1 4 4 2 1 1 1 1 1 4 4 4 4 1 2 1 4 3 1 4 4 1 4 1 3 1  
## [778] 1 1 1 4 2 2 2 4 1 4 4 2 2 1 4 3 1 4 4 4 4 4 2 1 4 2 2 1 4 4 4 4 2 4 4 2 2  
## [815] 2 1 2 1 4 3 1 4 4 1 4 1 3 2 4 4 4 2 4 1 2 2 2 4 1 1 4 2 2 1 4 3 1 4 4 2 1  
## [852] 4 3 1 4 4 1 4 1 3 2 4 4 2 4 2 2 4 2 2 4 2 2 2 4 2 2 2 1 4 2 2 1 1 1 4 1 4  
## [889] 1 1 1 2 4 4 2 2 2 2 4 4 2 2 1 4 3 1 4 4 4 4 4 1 1 4 1 4 4 2 2 1 4 2 1 4 3  
## [926] 1 4 4 1 4 1 3 4 4 2 2 4 2 3  
##   
## Within cluster sum of squares by cluster:  
## [1] 12910.103 13193.377 5826.272 23245.670  
## (between\_SS / total\_SS = 75.0 %)  
##   
## Available components:  
##   
## [1] "cluster" "centers" "totss" "withinss" "tot.withinss"  
## [6] "betweenss" "size" "iter" "ifault"

fviz\_cluster(kmean.res, data = studentsData,palette = c("#2E9FDF", "#FC4E07", "#CF3E4B", "#287233"),ellipse.type = "euclid", star.plot = TRUE,repel = TRUE, ggtheme = theme\_minimal() )



We can see that separation between clusters is not so clear, because some units are closer to a different cluster than the one they belong to.

cl2 <- kmean.res$cluster  
pairs(studentsData, gap=0, pch=cl2, col=c("#2E9FDF", "#FC4E07","#CF3E4B", "#287233")[cl2])



From this plot we can understand what variables are useful to find clusters. The variable sessional marks and final exams marks have good seperation between the clusters in the scatered plot.So if we remove some initial variables from the data it will be good to find the more accurate clusters.But, most of the variables in this case are overlapped and it is difficult to find the clusters in it.

## K-Medoids Clustering:

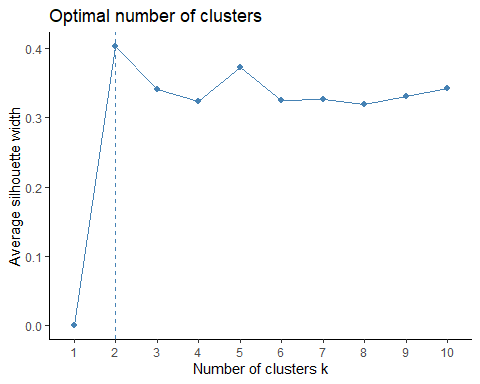
Differently from K-means, in K-medoids clustering each cluster is represented by one of the data points in the cluster. These points are named cluster medoids. The term medoid refers to a unit within a cluster for which average dissimilarity between it and all the other members of the cluster is minimal. It corresponds to the most centrally located point in the cluster. These units (one per cluster) can be considered as a representative example of the members of that cluster.

The most common K-medoids clustering method is the PAM (partitioning around medoids) algorithm.

The K-medoids algorithm requires the user to specify K, the number of clusters to be generated (like in K-means clustering). A useful approach to determine the optimal number of clusters is the silhouette method. The silhouette value, for each unit, is a measure of how similar a unit is to its own cluster (cohesion) compared to other clusters (separation).

The R function fviz\_nbclust() [factoextra package] provides a convenient solution to estimate the optimal number of clusters.

library(cluster)  
library(factoextra)  
fviz\_nbclust(studentsData, pam, method = "silhouette")+  
theme\_classic()

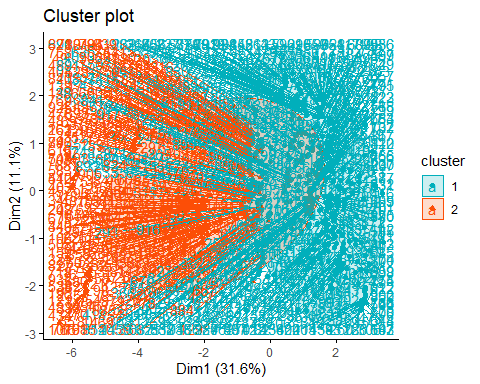
 We can see that the plot suggest us the number of clusters should be 2 from our dataset.

pam.res <- pam(studentsData, 2)  
pam.res$medoids

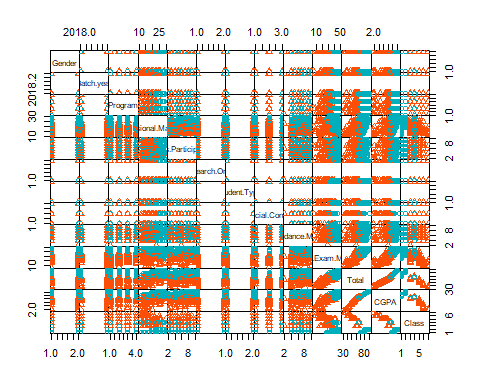
## Gender Batch.year Program Sessional.Marks Class.Participation  
## [1,] 1 2018 3 22 6  
## [2,] 2 2019 2 16 7  
## Is.Research.Oriented Student.Type Financial.Condition Attendance.Marks  
## [1,] 1 1 1 7  
## [2,] 1 1 2 6  
## Final.Exam.Marks Total CGPA Class  
## [1,] 39 74 4.024 1  
## [2,] 28 57 3.283 5

we can visualize the obtained cluster using the below plots.

fviz\_cluster(pam.res,  
palette = c("#00AFBB", "#FC4E07"), # color palette  
ellipse.type = "t", # Concentration ellipse  
repel = TRUE, # Avoid label overplotting (slow)  
ggtheme = theme\_classic()  
)



pam.clus<- pam.res$clustering  
pairs(studentsData, gap=0, pch=pam.clus, col= c("#00AFBB", "#FC4E07")[pam.clus] )

 The obove representation shows that there is very low tendency in the dataset for the clustering and we need to remove many attributes in order to find a suitable clustering in the dataset for the analysis or predictions.

# Cluster Validation:

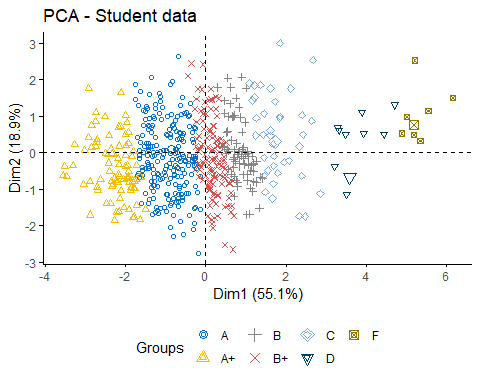
In the cluster analysis with the dataset we applied the clustering techniques on our data set without checking the validation of the clustering. Cluster validation gives us insights about the dataset that is it valid for applying clustering or not? So, in other words it is the measering of the goodness of the cluster results.

Before applying any clustering method on our data, it’s important to evaluate whether the data set contains meaningful clusters (i.e. non-random structures) or not. This process is defined as the assessing of clustering tendency or the feasibility of the cluster analysis.

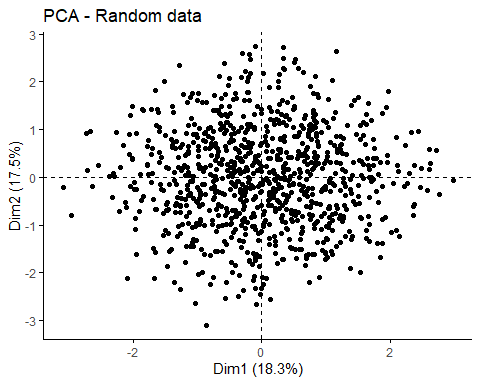
# Collected data from github repository  
x <- getURL("https://raw.githubusercontent.com/adeelahmedcsku/student-dataset/main/20-01-2020-data%20developed%20records%20students-changes%20in%20grading%20system%20-%20data%20developed%20records%20students.csv")  
studentsData <- read.csv(text = x)  
  
studentsData.alter<-data.frame(studentsData[,c(-1,-2,-3,-6,-7,-8,-13)])  
  
# Random data generated from the our data set  
student.alter.random\_df <- apply(studentsData.alter, 2,  
function(x){runif(length(x), min(x), (max(x)))})  
student.alter.random\_df <- as.data.frame(student.alter.random\_df)  
# Standardize the data sets  
studentsData.alter.scaled <- scale(studentsData.alter)  
student.alter.random\_df <- scale(student.alter.random\_df)

As the data contain more than two variables, we need to reduce the dimensionality in order to plot a scatter plot. This can be done using principal component analysis (PCA) algorithm (R function: prcomp()). After performing PCA, we use the function fviz\_pca\_ind() [factoextra R package] to visualize the output.

library("factoextra")  
# Plot faithful data set  
fviz\_pca\_ind(prcomp(studentsData.alter.scaled), title = "PCA - Student data",  
habillage = studentsData$Class, palette = "jco",  
geom = "point", ggtheme = theme\_classic(),  
legend = "bottom")



# Plot the random df  
fviz\_pca\_ind(prcomp(student.alter.random\_df), title = "PCA - Random data",  
geom = "point", ggtheme = theme\_classic())

 It can be seen from the graph that student data contains three real clusters while the random data plot has no meaningful clusters.

# Cluster Tendency:

In the previous sections where we computed the k-means algorithm and the hierarchical clustering impose a classification on the random uniformly distributed data set even if there are no meaningful clusters present in it. This is why, clustering tendency assessment methods should be used to evaluate the validity of clustering analysis. That is, whether a given data set contains meaningful clusters.

So, lets check the cluster tendency in our selected dataset.

We can assess if there are clusters from both a statistical point of view and from a graphical one, by means of, respectively, the Hopkins statistic and the VAT algorithm.

## Hopkins statistic:

Hopkins statistic on the dataset altered by removing the nominal variables.

library(clustertend)  
hopkins(studentsData.alter.scaled, n = nrow(studentsData.alter.scaled)-1)

## $H  
## [1] 0.2001505

Hopkins statistics on the randomly generated data set.

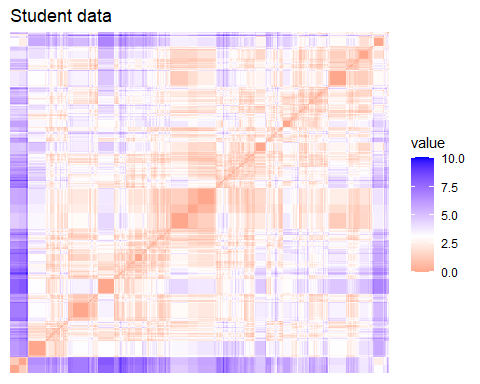
hopkins(student.alter.random\_df, n = nrow(student.alter.random\_df)-1)

## $H  
## [1] 0.5032931

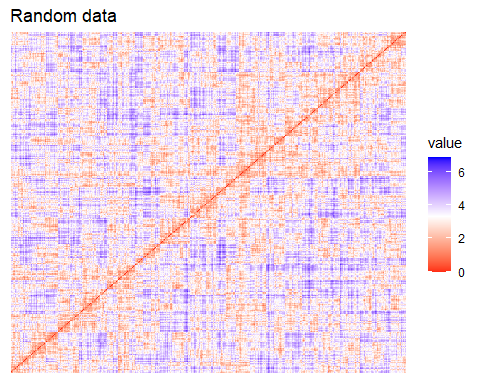
It can be seen that the student data set which is altered and removed the categorical attrbutes is clusterable because the value of H = 0.19 which is closed to zero. While the random sample of the same data set is not clusterable because the value of the H=0.5.

## VAT algorithm:

fviz\_dist(dist(studentsData.alter.scaled), show\_labels = FALSE)+ labs(title = "Student data")



fviz\_dist(dist(student.alter.random\_df), show\_labels = FALSE)+ labs(title = "Random data")

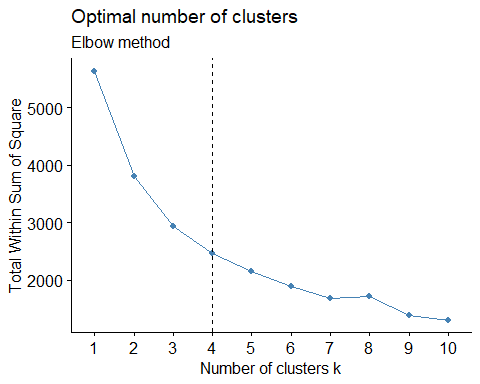
 The dissimilarity matrix image confirms that there is a cluster structure in the standardized Student data set but not in the random one.

# Determining the optimal number of clusters:

There are many ways to calculate the optimal numbers of clusters, like Elbow method,Silhouette method and gap statistics method.

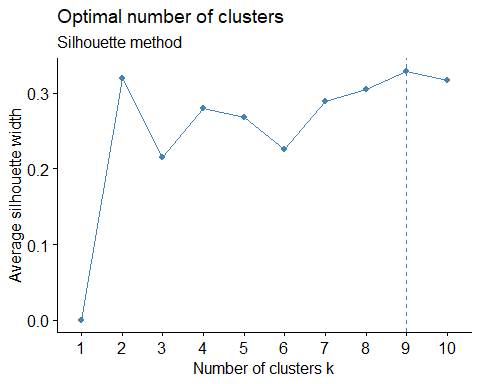
## Elbow method

fviz\_nbclust(studentsData.alter.scaled, kmeans, method = "wss") +  
geom\_vline(xintercept = 4, linetype = 2)+  
labs(subtitle = "Elbow method")



## Silhouette method

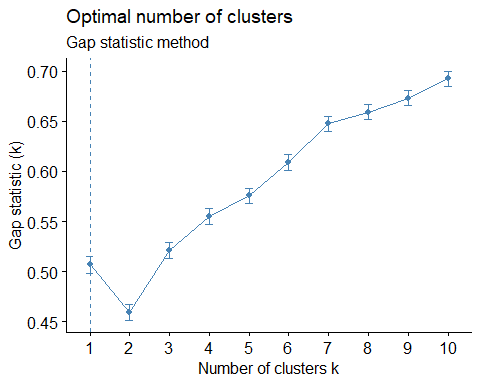
fviz\_nbclust(studentsData.alter.scaled, kmeans, method = "silhouette")+  
labs(subtitle = "Silhouette method")



## Gap statistic

# nboot = 50 to keep the function speedy.  
# recommended value: nboot= 500 for your analysis.  
# Use verbose = FALSE to hide computing progression.  
set.seed(123)  
fviz\_nbclust(studentsData.alter.scaled, kmeans, nstart = 25, method = "gap\_stat", nboot = 500)+  
labs(subtitle = "Gap statistic method")

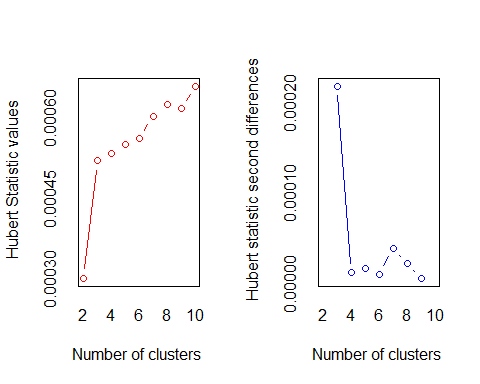
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations  
  
## Warning: did not converge in 10 iterations



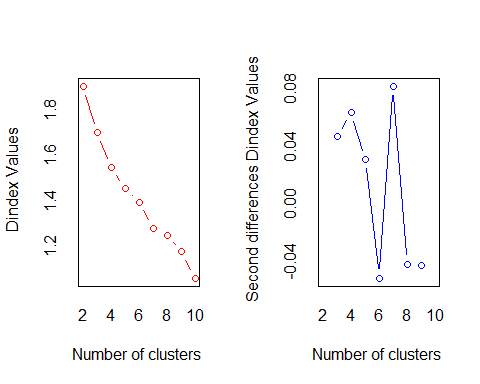
1. Elbow method: 4 clusters solution suggested
2. Silhouette method: 9 clusters solution suggested
3. Gap statistic method: 1 clusters solution suggested

## NbClust Method:

library("NbClust")  
  
 nb <- NbClust(studentsData.alter.scaled, distance = "euclidean", min.nc = 2, max.nc = 10, method = "kmeans")



## \*\*\* : The Hubert index is a graphical method of determining the number of clusters.  
## In the plot of Hubert index, we seek a significant knee that corresponds to a   
## significant increase of the value of the measure i.e the significant peak in Hubert  
## index second differences plot.   
##



## \*\*\* : The D index is a graphical method of determining the number of clusters.   
## In the plot of D index, we seek a significant knee (the significant peak in Dindex  
## second differences plot) that corresponds to a significant increase of the value of  
## the measure.   
##   
## \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*   
## \* Among all indices:   
## \* 6 proposed 2 as the best number of clusters   
## \* 4 proposed 3 as the best number of clusters   
## \* 5 proposed 4 as the best number of clusters   
## \* 2 proposed 7 as the best number of clusters   
## \* 1 proposed 9 as the best number of clusters   
## \* 5 proposed 10 as the best number of clusters   
##   
## \*\*\*\*\* Conclusion \*\*\*\*\*   
##   
## \* According to the majority rule, the best number of clusters is 2   
##   
##   
## \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

The results of NbClust() can be graphically visualized using fviz nbclust() (in the factoextra package) as follows:

library("factoextra")  
fviz\_nbclust(nb)

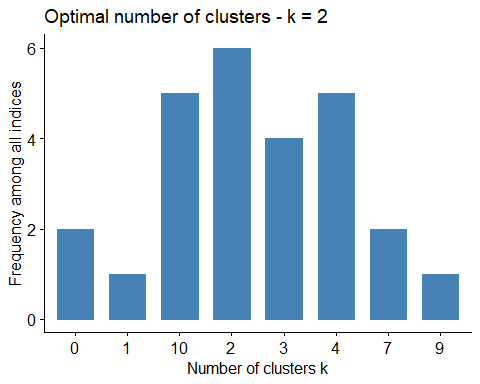
## Warning in if (class(best\_nc) == "numeric") print(best\_nc) else if  
## (class(best\_nc) == : the condition has length > 1 and only the first element  
## will be used

## Warning in if (class(best\_nc) == "matrix") .viz\_NbClust(x, print.summary, : the  
## condition has length > 1 and only the first element will be used

## Warning in if (class(best\_nc) == "numeric") print(best\_nc) else if  
## (class(best\_nc) == : the condition has length > 1 and only the first element  
## will be used

## Warning in if (class(best\_nc) == "matrix") {: the condition has length > 1 and  
## only the first element will be used

## Among all indices:   
## ===================  
## \* 2 proposed 0 as the best number of clusters  
## \* 1 proposed 1 as the best number of clusters  
## \* 6 proposed 2 as the best number of clusters  
## \* 4 proposed 3 as the best number of clusters  
## \* 5 proposed 4 as the best number of clusters  
## \* 2 proposed 7 as the best number of clusters  
## \* 1 proposed 9 as the best number of clusters  
## \* 5 proposed 10 as the best number of clusters  
##   
## Conclusion  
## =========================  
## \* According to the majority rule, the best number of clusters is 2 .



# Clustering Validation Statistics:

The term cluster validation is used to design the procedure of evaluating the goodness of clustering algorithm results. Generally, clustering validation statistics can be categorized into 3 classes:

1. Internal cluster validation: Consider the insights of the cluster.
2. External Cluster Validation: Consider the external provided class labels etc.
3. Relative Cluster Validation: Consider the variation in the parameter values like no of clusters etc.

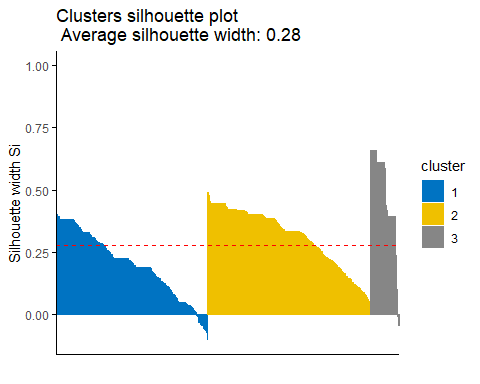
The internal cluster validation allows us to estimate the optimal number of clusters and to select the appropriate clustering algorithm, by means of two indices: the Silhouette Width and the Dunn index.

## Silhouette plot

The silhouette coecient (Si) measures how similar an object i is to the other objects in its own cluster versus those in the neighbor cluster.

km.res <- eclust(studentsData.alter.scaled, "kmeans", k = 3, nstart = 25, graph = FALSE)  
 fviz\_silhouette(km.res, palette = "jco", ggtheme = theme\_classic())

## cluster size ave.sil.width  
## 1 1 414 0.20  
## 2 2 445 0.31  
## 3 3 80 0.48



# Silhouette information  
silinfo <- km.res$silinfo  
 silinfo$avg.width

## [1] 0.2758341

The average silhouette width is 0.27. No cluster participate overall widths. Looking at the plot, we can see that in cluster 1 some of the units have a silhouette width higher than the average one and while some have negative silhouette width. While in cluster 2 there most of the units have a silhouette width lower than the average one; in cluster 3 almost half units have a high silhouette width,and some outlier as well with the negative silhouette width others a low one.

It can be seen that several samples, in cluster 1 and 3, have a negative silhouette coeffcient. This means that they are not in the right cluster. We can find the name of these samples and determine the clusters they are closer (neighbor cluster), as follow:

# Silhouette width of observation  
sil <- km.res$silinfo$widths[, 1:3]  
# Objects with negative silhouette  
neg\_sil\_index <- which(sil[, 'sil\_width'] < 0)  
sil[neg\_sil\_index, , drop = FALSE]

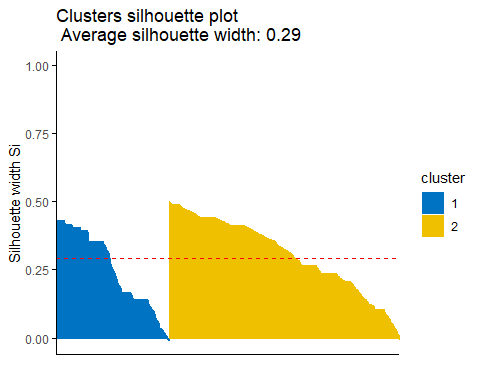
## cluster neighbor sil\_width  
## 564 1 2 -0.003351573  
## 813 1 2 -0.006538247  
## 447 1 2 -0.007013895  
## 914 1 2 -0.007182948  
## 828 1 2 -0.014321073  
## 744 1 2 -0.030442362  
## 214 1 2 -0.031525939  
## 356 1 2 -0.031660254  
## 655 1 2 -0.031739095  
## 180 1 2 -0.032353877  
## 147 1 2 -0.038553938  
## 364 1 2 -0.039353907  
## 763 1 2 -0.039688584  
## 675 1 2 -0.040991680  
## 155 1 2 -0.044752551  
## 755 1 2 -0.044893601  
## 567 1 2 -0.048711872  
## 808 1 2 -0.050543076  
## 148 1 2 -0.051183202  
## 865 1 2 -0.057294588  
## 736 1 2 -0.057977860  
## 598 1 2 -0.059410181  
## 719 1 2 -0.060014101  
## 809 1 2 -0.060734057  
## 293 1 2 -0.064824652  
## 630 1 2 -0.064824652  
## 724 1 2 -0.066606607  
## 829 1 2 -0.074974505  
## 672 1 2 -0.079391650  
## 690 1 2 -0.104082860  
## 152 3 2 -0.009015080  
## 105 3 2 -0.018833522  
## 143 3 2 -0.042865558  
## 110 3 2 -0.044909600

There are some elements of cluster 1 which we should include in the cluster 2 and there are also some elements of the cluster 3 which we need to inculde in the cluster 2.

This is a further confirmation that the optimal number of clusters is 2, in fact, if we run the kmeans with K=2, the plot does not present any unit with a negative silhouette width:

km.res <- eclust(studentsData.alter.scaled, "kmeans", k = 2, nstart = 25, graph = FALSE)  
fviz\_silhouette(km.res, palette = "jco", ggtheme = theme\_classic())

## cluster size ave.sil.width  
## 1 1 308 0.25  
## 2 2 631 0.31



## Dunn Index:

library(fpc)  
# Statistics for k-means clustering  
km\_stats <- cluster.stats(dist(studentsData.alter.scaled), km.res$cluster)  
# Dun index  
km\_stats$dunn

## [1] 0.02317697

The aim of the Dunn index is to identify sets of clusters that are compact, with a small variance between members of the cluster, and well separated. A higher Dunn index indicates better clustering, so it should be maximized.

## Choosing the Best Clustering Algorithms

## Internal Measures:

We can compare the multiple clustering methods using the R library clValid in order to find out that which clustering algorithm suited best for our data.

library(clValid)

# Compute clValid

clmethods <- c("hierarchical","kmeans","pam")

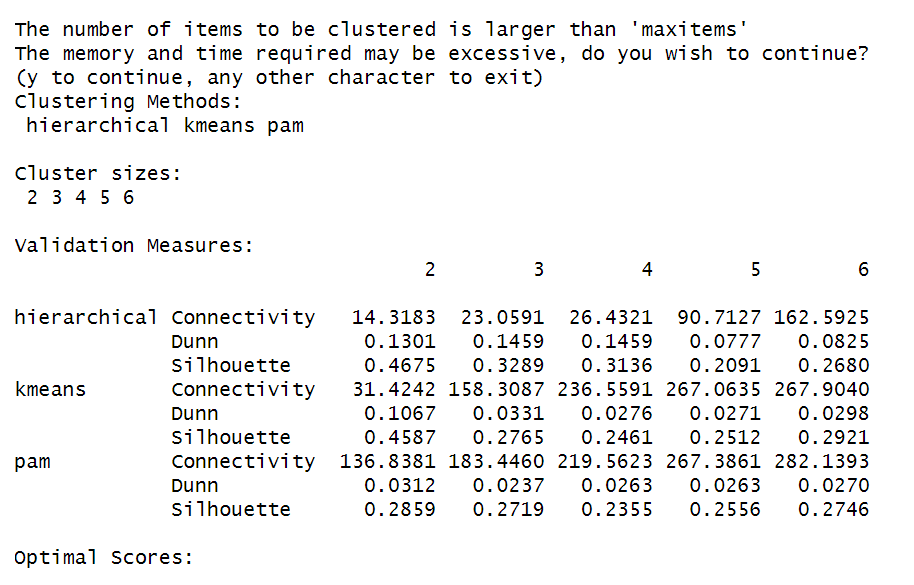
intern <- clValid(studentsData.alter.scaled, nClust = 2:6,

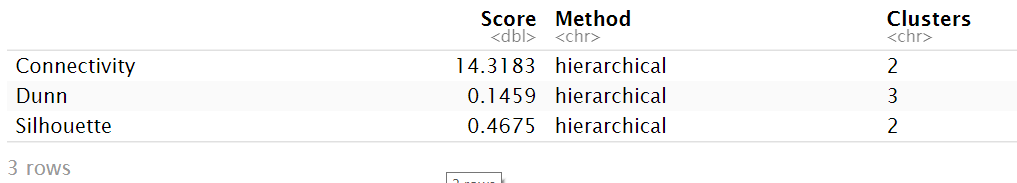
clMethods = clmethods, validation = "internal")

# Summary

summary(intern)

Results:





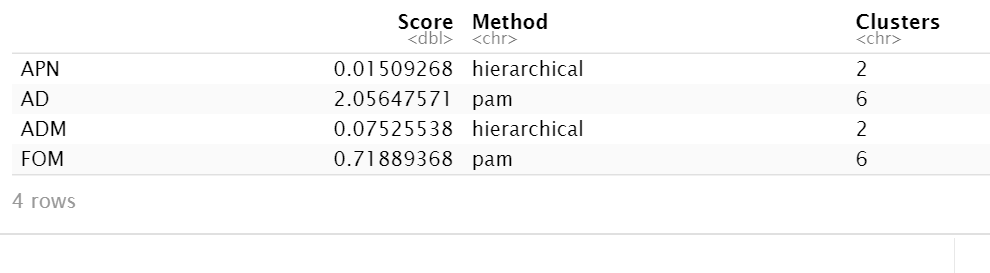
In comparing the three clustering methods hierarchical, k-means and pam connectivity must be minimized hence, the best solution is 14.31 which belongs to hierarchical method. Dunn and Silhouette must be maximized; hence the best solutions are also hierarchical 0.13 and 0.46 respectively.

## Stability Measures:

clmethods <- c("hierarchical","kmeans","pam")

stab <- clValid(studentsData.alter.scaled, nClust = 2:6, clMethods = clmethods,validation = "stability")

optimalScores(stab)



According to the stability measure there is an ex aqua between APN and ADM and the best clustering algorithm is Hierarchical clustering with the number of clusters 2.

Model Based Clustering

library(RCurl)  
  
# Collected data from github repository  
x <- getURL("https://raw.githubusercontent.com/adeelahmedcsku/student-dataset/main/20-01-2020-data%20developed%20records%20students-changes%20in%20grading%20system%20-%20data%20developed%20records%20students.csv")  
studentsData <- read.csv(text = x)  
# Standardize the data sets

The model based clustering consider the data as coming from a distribution that is mixture of two or more clusters. In model-based clustering, the data is considered as coming from a mixture of density. Each component (i.e. cluster) k is modeled by the normal or Gaussian distribution.

## Choosing the best model:

The Mclust package uses maximum likelihood to fit all these models, with dierent covariance matrix parameterizations, for a range of k components. The best model is selected using the Bayesian Information Criterion or BIC. A large BIC score indicates strong evidence for the corresponding model.

We start by installing the mclust package as follow: install.packages(“mclust”)

#install.packages("mclust")

library(mclust)

## Package 'mclust' version 5.4.7  
## Type 'citation("mclust")' for citing this R package in publications.

studentsData.alter<-data.frame(studentsData[,c(-1,-2,-3,-6,-7,-8,-13)])  
studentsData.alter.scaled <- scale(studentsData.alter)  
mc <- Mclust(studentsData.alter.scaled) # Model-based-clustering  
summary(mc) # Print a summary

## ----------------------------------------------------   
## Gaussian finite mixture model fitted by EM algorithm   
## ----------------------------------------------------   
##   
## Mclust XXX (ellipsoidal multivariate normal) model with 1 component:   
##   
## log-likelihood n df BIC ICL  
## 25405.53 939 27 50626.24 50626.24  
##   
## Clustering table:  
## 1   
## 939

For our Student dataset, it can be seen that model-based clustering selected a model with one component (i.e. clusters). The optimal selected model name is XXX model. As the data set is not fit for this type of clustering i.e., model based clustering. May be we need to improve our dataset or change the techniques and use the hierarchical clustering which fitted best for our data.