Name-Surname : Rayene BECH **Email : l1118115@std.yildiz.edu.tr**

No :18011115 **Signature :**

HOME WORK 1 (Return by 20.12.2021)

BLM3590 – Statistical Data Analysis

T1(10)	T2(15)	T3(15)	T4(15)	T5(15)	T6(15)	T7(15)				Total(100)

The attached Excel file (**SdA-HW**) consists of two clases of data (embolic signals (**class 1**), and Doppler speckle (**class 2**)) recorded from stroke patients and some relevant numerical variables (**tpthrt**, **pkthrt**, **dfdrrt**, **time**, **rrt**, **frt**). Using this data file, implement the following tasks in **R**. You must include the **R** scripts in your answers.

==========

T1: Show how to read this Excel datafile into **R** environment.

#First the package that allow us to read ".xls" files should be installed using the following instruction:

```
> install.packages("readxl")
```

#The following code is executed:

```
library("readxl")
df <- read_excel("2k21-SdA-HW1.xls")
```

#Note: The working directory should be selected prior to the execution of the program using the following instruction:

```
setwd("working_directory_path")
```

==========

T2: This data file requires some preprocessing as it inludes a column with no value, some cells with no numerical value (divide by 0 error, etc.), and some cells with zero. Write required script in **R** to remove the empty column and correct the cells with no numerical value and zero by using simple interpolation.

#The empty column can be identified using the function sapply() :

```
empty_column <- sapply(df, function(x) all(is.na(x) | x == ""))
```

#This function returns True for the empty columns and False for non-empty columns as it can be seen bellow:

```
> empty_column
  class    tpthrt   pkthrt   dfdrrt     time    rrt      frt
 FALSE    FALSE    FALSE    FALSE     TRUE    FALSE    FALSE
```

#Now the empty column can be deleted:

```
df= df[, !empty_column]
```

#To get rid of zero valuesw, we turn them to NA and then To replace non numerical values the package "imputeTS" is installed and used:

```
df[df == 0] <- NA
install.packages("imputeTS")
library(imputeTS)
df = na_interpolation(df)
```

This package "imputeTS" contains many other functions that we could use to replace the NA values. For example, we could use "na_mean" to replace the missing values with the mean.
All the functions of this package are listed in this table:

Simple Imputation	Imputation	Plots & Statistics	Datasets
na_locf	na_interpolation	ggplot_na_distribution	tsAirgap
na_mean	na_kalman	ggplot_na_intervals	tsAirgapComplete
na_random	na_ma	ggplot_na_gapsize	tsHeating
na_replace	na_seadec	ggplot_na_imputations	tsHeatingComplete
na_remove	na_seasplit	statsNA	tsNH4
			tsNH4Complete

Table 1: General Overview imputeTS package

==========

T3: Find **Five-number data summary** of the **variables** for each **data class** in this dataset.

Requirements: to use the pipeline operator %>% we need to install and import the "dplyr" package
install.packages("dplyr")
library("dplyr")
Then to access each class we use the function filter()
We finally call the function summary which will give us the five number summary
For class 1:

```
> summary(subset(df %>% filter(df$class==1), select = -c(class)))
```

tpthrt	pkthrt	dfdrrt	rrt	frt
Min. : 0.3364	Min. :-8.359	Min. : 0.2601	Min. :-59.765	Min. : 0.9525
1st Qu.:12.0774	1st Qu.: 3.320	1st Qu.:12.2683	1st Qu.: 2.925	1st Qu.: 4.5475
Median :14.8535	Median : 5.949	Median :17.8565	Median : 5.655	Median : 6.5960
Mean :15.6507	Mean : 5.793	Mean :17.5927	Mean : 4.236	Mean : 7.5667
3rd Qu.:19.4746	3rd Qu.: 8.319	3rd Qu.:22.4063	3rd Qu.: 9.469	3rd Qu.:10.1026
Max. :28.8640	Max. :19.907	Max. :45.4140	Max. : 20.610	Max. :24.7323

For class 2:

```
> summary(subset(df %>% filter(df$class==2), select = -c(class)))
```

tpthrt	pkthrt	dfdrrt	rrt	frt
Min. : 0.008013	Min. :-9.623	Min. :-3.476	Min. :-68.801	Min. : 0.05729
1st Qu.: 8.038797	1st Qu.:-1.003	1st Qu.: 7.022	1st Qu.: -7.168	1st Qu.: 2.16383
Median :11.451826	Median : 1.277	Median :13.799	Median : 2.211	Median : 3.90122
Mean :11.093465	Mean : 1.301	Mean :14.419	Mean : -3.131	Mean : 4.93257
3rd Qu.:14.265218	3rd Qu.: 4.161	3rd Qu.:21.877	3rd Qu.: 4.616	3rd Qu.: 6.30550
Max. :20.963047	Max. : 7.324	Max. :40.545	Max. : 26.127	Max. :32.28238

==========

T4: Plot **boxplots** of the **variables** for each **data class** and determine if there is any outlier in these variables.

For class 1:

boxplot(subset(df %>% filter(df$class==1), select = -c(class)))

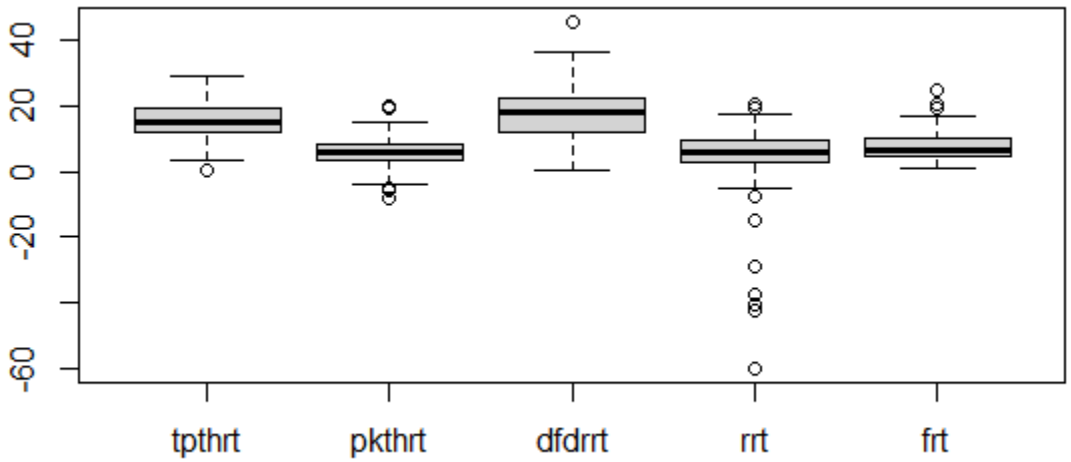


Figure 1: boxplot of class 1

Yes there are outliers for all variables shown by the circle

For class 2:

boxplot(subset(df %>% filter(df$class==2), select = -c(class)))

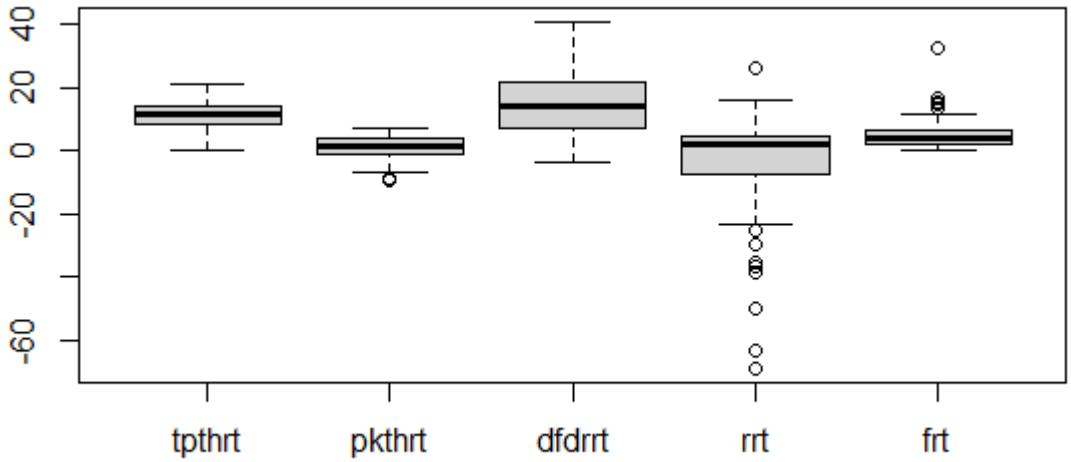


Figure 2: boxplot of class 2

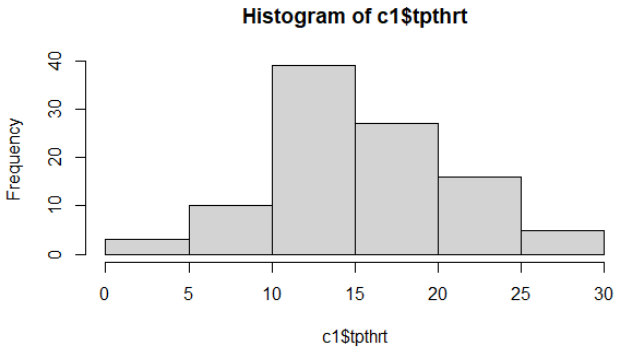
Yes there are outliers for the following variables: “pkthrt”, “rrt”, “frt”

==========

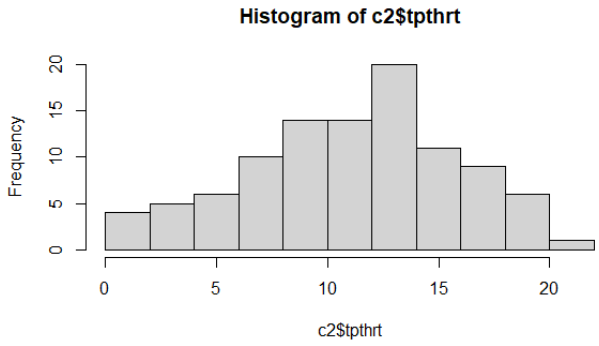
T5: Plot histograms of the **variables** for each **data class**, compare the histograms, and comment on the distributions.

Plotting "tpthrt"

```
c1 <- subset(df %>% filter(df$class==1), select = c(tpthrt))
hist(c1$tpthrt)
c2 <- subset(df %>% filter(df$class==2), select = c(tpthrt))
hist(c2$tpthrt)
```



Histogram of "tpthrt" for class 1

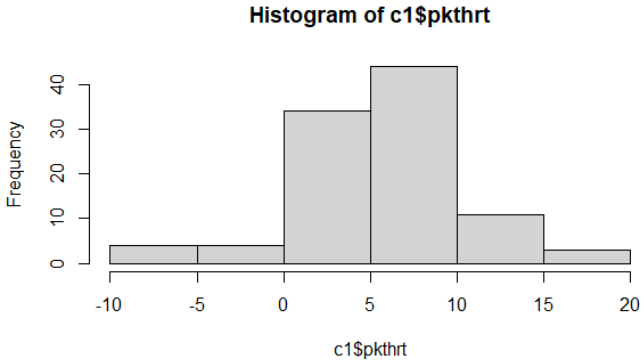


Histogram of "tpthrt" for class 2

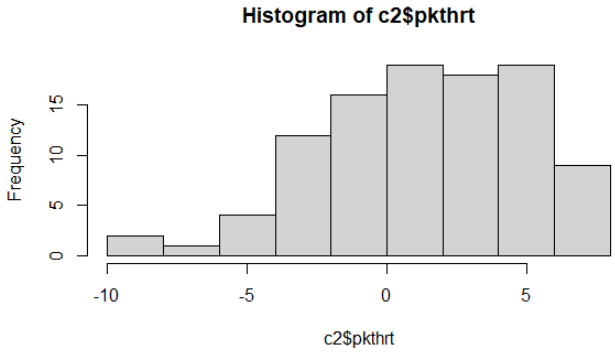
Both distributions can be considered as normal distributions. Note that class 1 has greater range (from 0 to 30) comparing to class 2 (from 0 to 25) While class 2 has greater number of cells. Class 1 has greater mean and median (around 15) than class 2 has (around 12)

Plotting "pkthrt"

```
c1 <- subset(df %>% filter(df$class==1), select = c(pkthrt))
hist(c1$pkthrt)
c2 <- subset(df %>% filter(df$class==2), select = c(pkthrt))
hist(c2$pkthrt)
```



Histogram of "pkthrt" for class 1

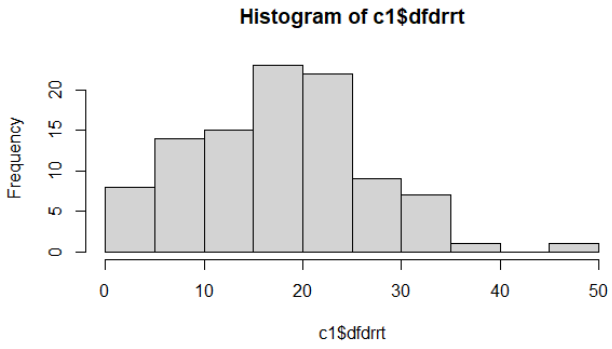


Histogram of "pkthrt" for class 2

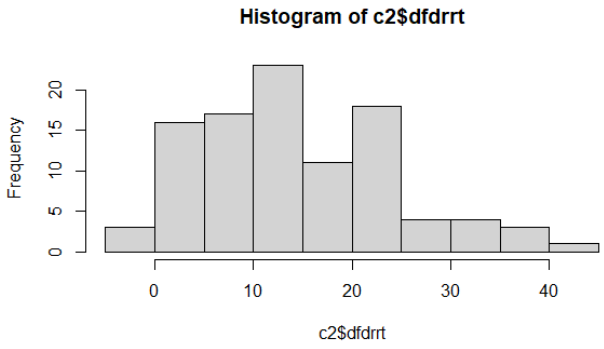
The class 1 has normal distribution while class 2 has left-skewed distribution. Also class 1 has greater range (from -10 to 20) comparing to class 2 (from -10 to 10). We can also notice that the histogram of class 2 has greater number of cells. Class 1 has greater mean and median (around 5) than class 2 has (around 1)

Plotting "dfdrrt"

```
c1 <- subset(df %>% filter(df$class==1), select = c(dfdrrt))
hist(c1$dfdrrt)
c2 <- subset(df %>% filter(df$class==2), select = c(dfdrrt))
hist(c2$dfdrrt)
```



Histogram of "dfdrrt" for class 1

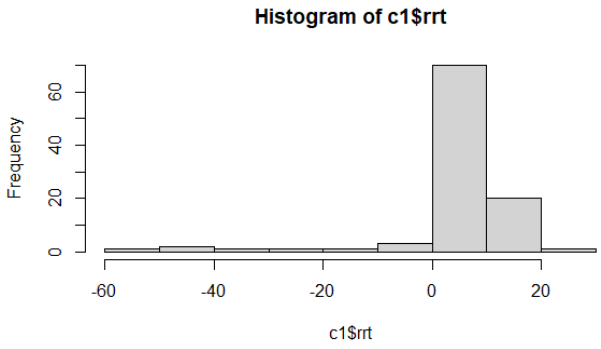


Histogram of "dfdrrt" for class 1

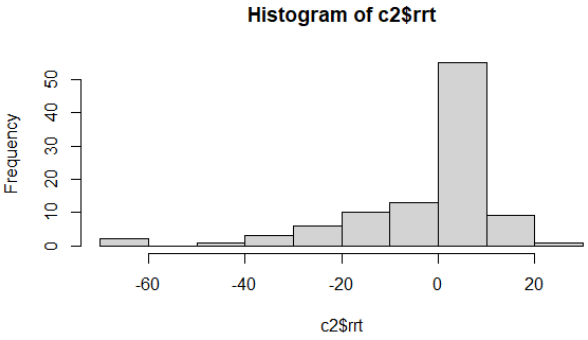
The class 1 has right-skewed distribution while class 2 seems to have bimodal distribution. Also class 1 has greater range (from 0 to 50) comparing to class 2 (from 0 to 40). That's because class 1 has an outlier at 50.

Plotting "rrt"

```
c1 <- subset(df %>% filter(df$class==1), select = c(rrt))
hist(c1$rrt)
c2 <- subset(df %>% filter(df$class==2), select = c(rrt))
hist(c2$rrt)
```



Histogram of "rrt" for class 1

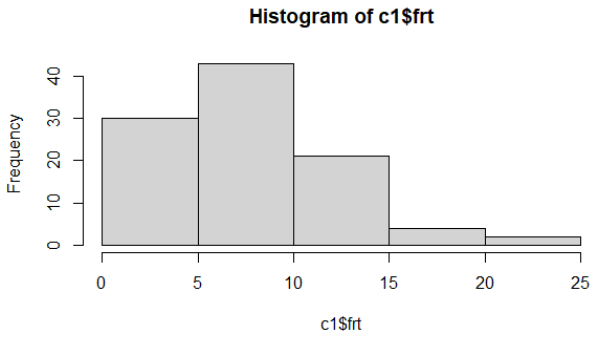


Histogram of "rrt" for class 2

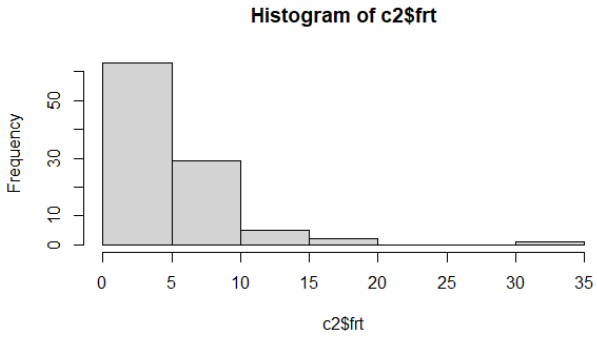
Both classes have left-skewed distributions and equal ranges. Class 2 has more standard deviation than class 1 .

Plotting "frt"

```
c1 <- subset(df %>% filter(df$class==1), select = c(frt))
hist(c1$frt)
c2 <- subset(df %>% filter(df$class==2), select = c(frt))
hist(c2$frt)
```



Histogram of "frt" for class 1



Histogram of "frt" for class 2

Both classes have right-skewed distributions. Class 2 has more standard deviation than class 1 . Also class 2 has greater range (from 0 to 35) comparing to class 2 (from 0 to 25). That's because class 2 has an outlier at 35. Finally, Class 1 has greater mean and median (around 7) than class 2 has (around 4)

==========

T6: First, normalize the **variables** for each **data class** so that the values of these variables range between **0** and **1**, and then line-plot (using different colors) each variables for both data classes in one figure (total 5 figures). Comment on the similarities of the variables for each plot.

To normalize we define the following function:

```
normalize <- function(x) {
  return ((x - min(x)) / (max(x) - min(x)))
}
```

To normalize the first class:

```
c1 <- subset(df %>% filter(df$class==1), select = -c(class))
for(i in 1:ncol(c1)) {       # for-loop over columns
  c1[ , i] <- normalize(c1[ , i])
}
```

To normalize the second class:

```
c2 <- subset(df %>% filter(df$class==2), select = -c(class))
for(i in 1:ncol(c2)) {       # for-loop over columns
  c2[ , i] <- normalize(c2[ , i])
}
```

We could also use the "data.Normalization" function which provides mmany methods to do the normalization:

Usage

data.Normalization (x,type="n0",normalization="column",...)

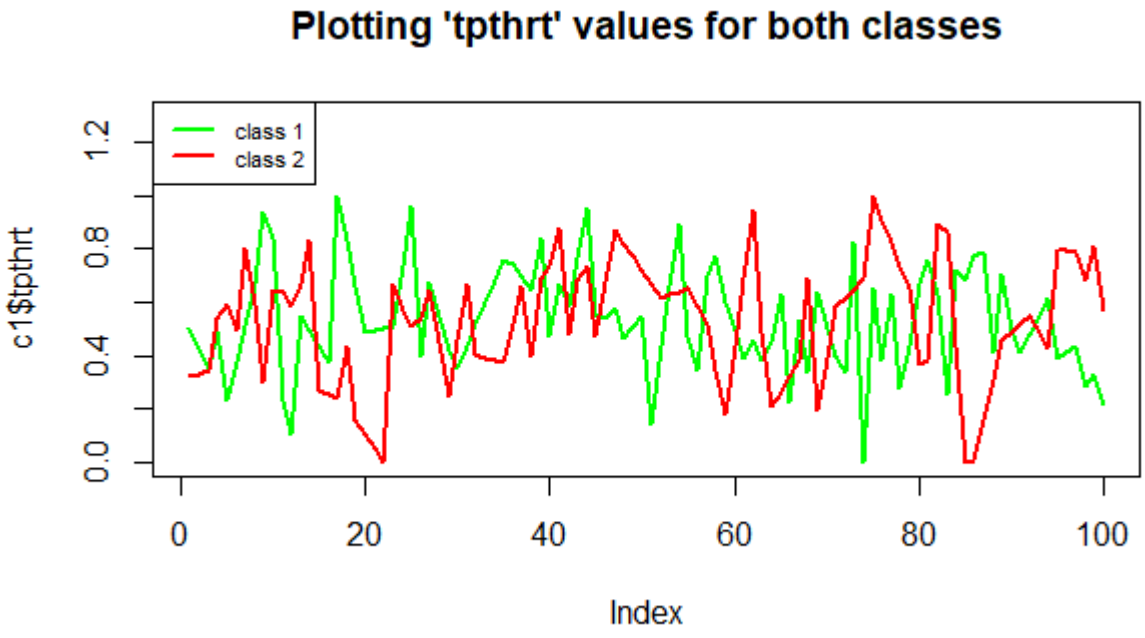
type of normalization:

n0 - without normalization, n1 - standardization ((x-mean)/sd), n2 - positional standardization ((x-median)/mad), n3 - unitization ((x-mean)/range), n3a - positional unitization ((x-median)/range), n4 - unitization with zero minimum ((x-min)/range), n5 - normalization in range <-1,1> ((x-mean)/max(abs(x-mean))), n5a - positional normalization in range <-1,1> ((x-median)/max(abs(x-median))), n6 - quotient transformation (x/sd), n6a - positional quotient transformation (x/mad), n7 - quotient transformation (x/range), n8 - quotient transformation (x/max), n9 - quotient transformation (x/mean), n9a - positional quotient transformation (x/median), n10 - quotient transformation (x/sum), n11 - quotient transformation (x/sqrt(SSQ)), n12 - normalization ((x-mean)/sqrt(sum((x-mean)^2))), n12a - positional normalization ((x-median)/sqrt(sum((x-median)^2))), n13 - normalization with zero being the central point ((x-midrange)/(range/2))

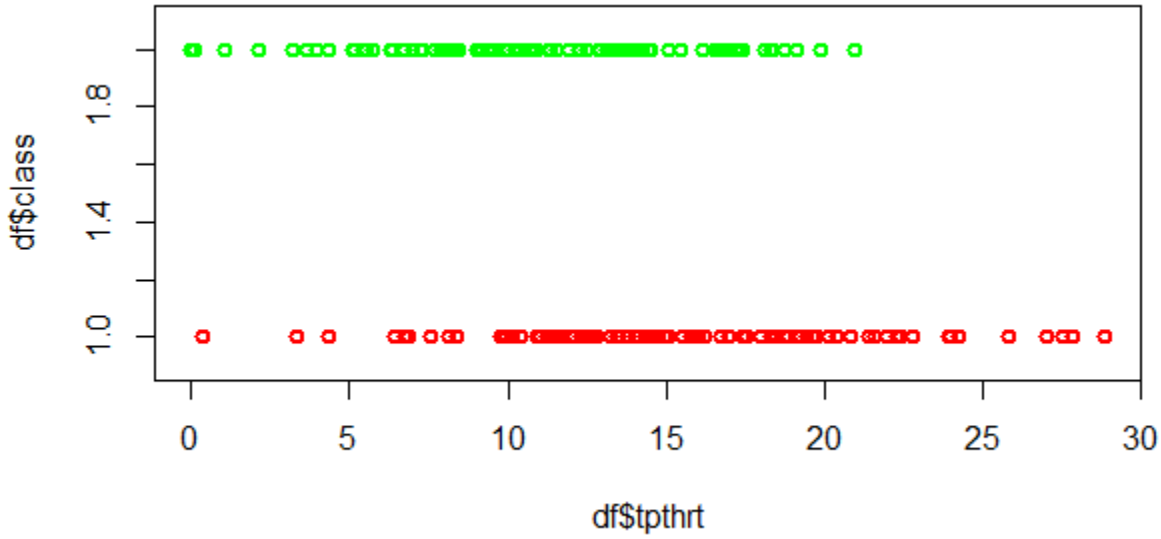
line-plot for “tpthrt”

```
plot(c1$tpthrt, type="l", col="green",lwd=2, ylim=c(0,1.3))
lines(c2$tpthrt, col="red", lwd=2)
title("Plotting 'tpthrt' values for booth classes")
legend("topleft", legend=c("class 1", "class 2"),col=c("green", "red"), lwd=c(2,2), cex=0.7)
```

We can notice that the graphs are almost negatively corelated. Class 1 has values that are almost the opposite of Class 2:



Now let's compare the distribution of "tpthrt" for both classes using this command:
plot(df$tpthrt,df$class, type="p", col=c("red","green")[df$class],lwd=2, ylim=c(0.9,2.1))

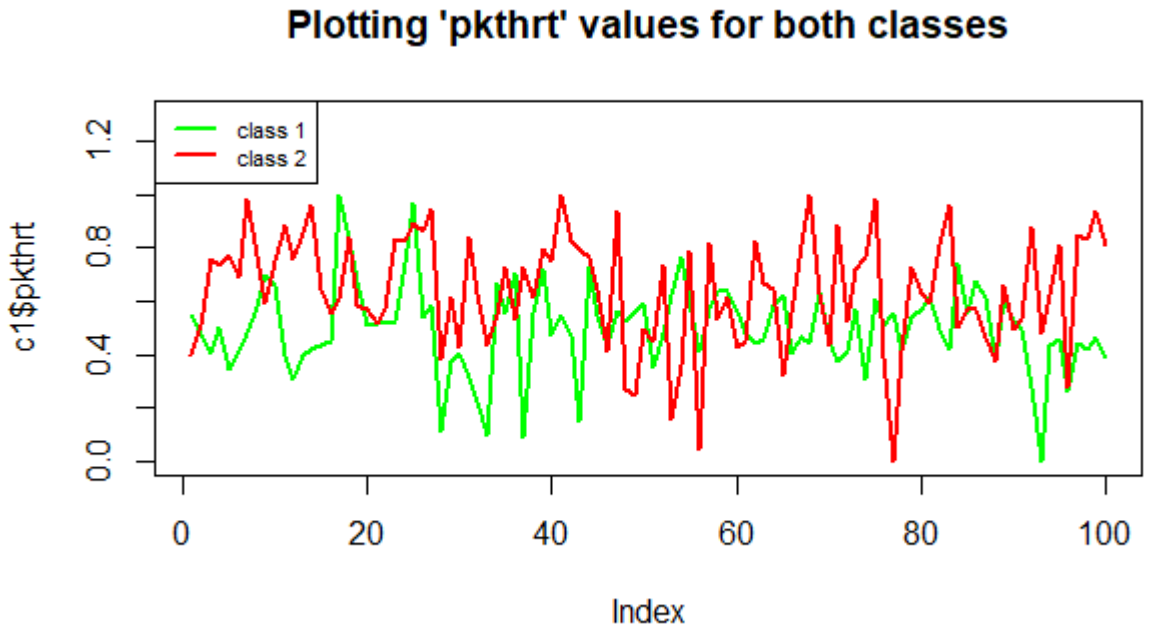


We notice that the variable tpthrt has different values and density function for each class.

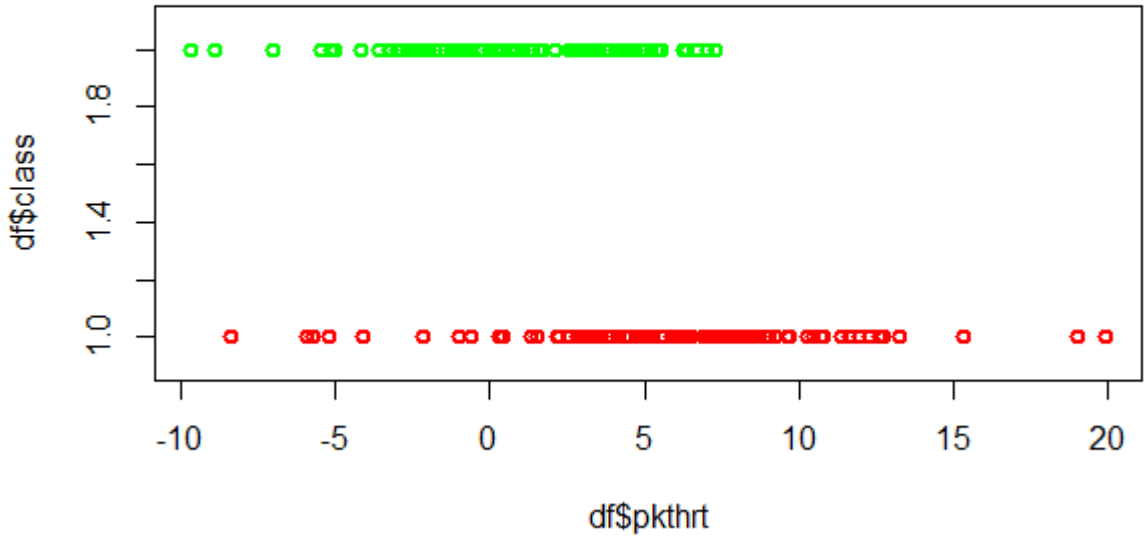
line-plot for "pkthrt"

plot(c1$pkthrt, type="l", col="green",lwd=2, ylim=c(0,1.3))
lines(c2$pkthrt, col="red", lwd=2)
title("Plotting ' pkthrt ' values for booth classes")
legend("topleft", legend=c("class 1", "class 2"),col=c("green", "red"), lwd=c(2,2), cex=0.7)

We can notice that the graphs are almost negatively corelated. Class 1 has values that are almost the opposite of Class 2. But the are a bit similar in the interval [50-80]:



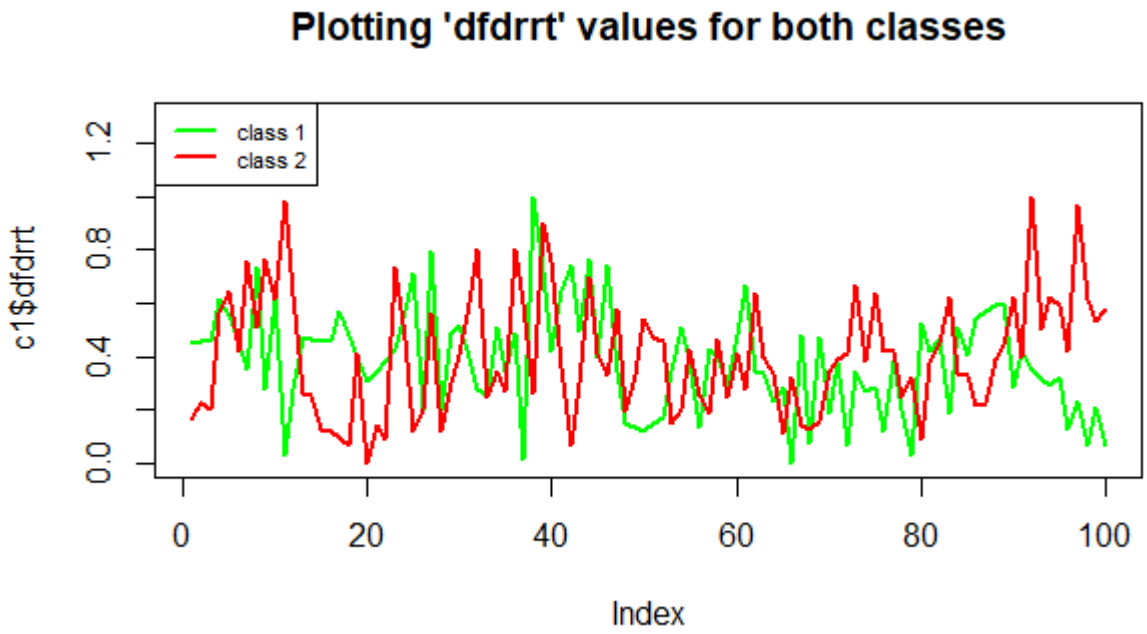
Now let's compare the distribution of "pkthrt" for both classes using this command:
plot(df$pkthrt,df$class, type="p", col=c("red","green")[df$class],lwd=2, ylim=c(0.9,2.1))



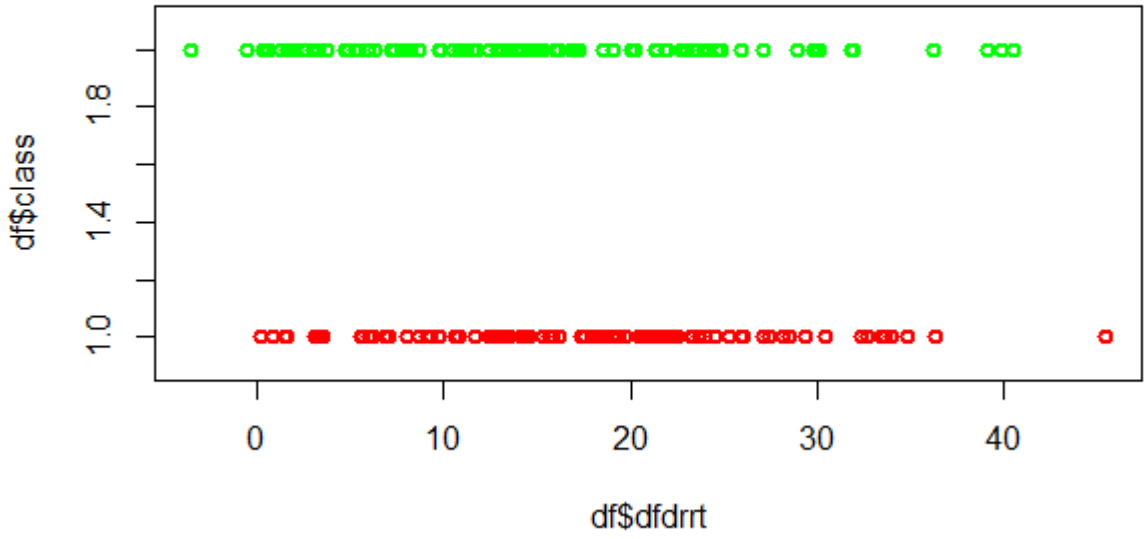
Again we notice that the variable pkthrt has different values and density function for each class.

line-plot for “dfdrrt”

plot(c1$dfdrrt, type="l", col="green",lwd=2, ylim=c(0,1.3))
lines(c2$dfdrrt, col="red", lwd=2)
title("Plotting 'dfdrrt' values for both classes")
legend("topleft", legend=c("class 1", "class 2"),col=c("green", "red"), lwd=c(2,2), cex=0.7)
We can notice that the graphs are very similar for many intervals.



Now let’s compare the distribution of “dfdrrt” for both classes using this command:
plot(df$dfdrrt,df$class, type="p", col=c("red","green")[df$class],lwd=2, ylim=c(0.9,2.1))

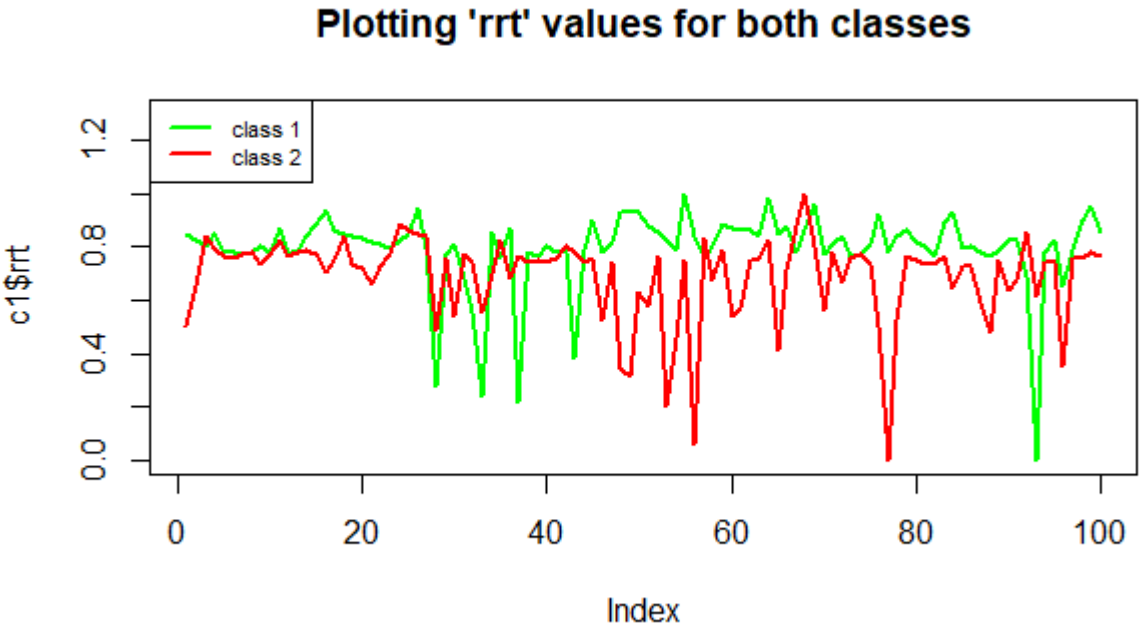


We notice that the variable dfdrrt has almost same values and density function for both class. THUS WE CONCLUDE THAT IT IS A REDUNDANT DATA and can be deleted when comparing the classes or performing a classification task.

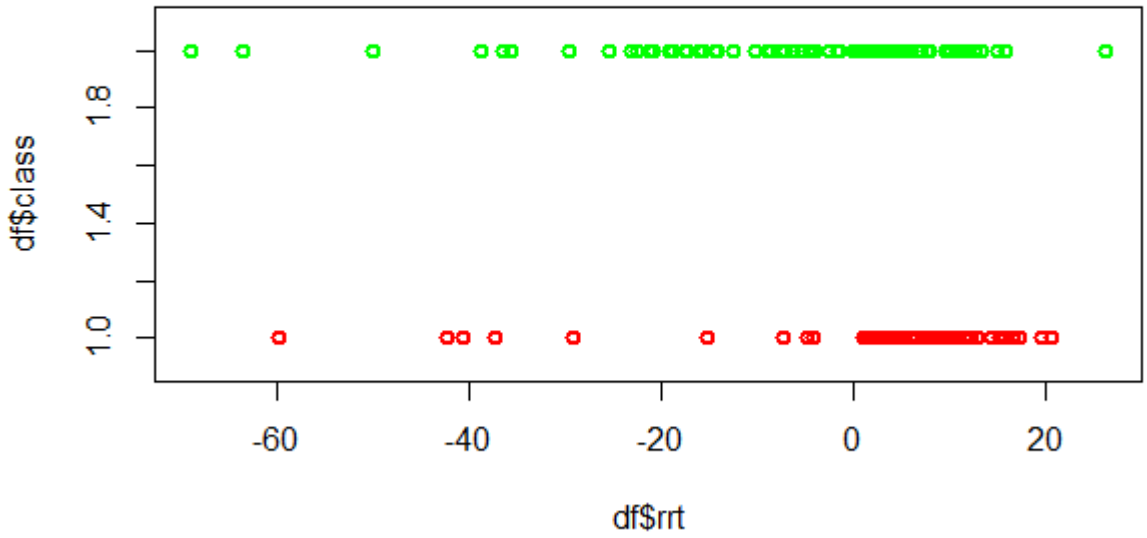
line-plot for "rrt"

```
plot(c1$rrt, type="l", col="green",lwd=2, ylim=c(0,1.3))
lines(c2$rrt, col="red", lwd=2)
title("Plotting 'rrt' values for both classes")
legend("topleft", legend=c("class 1", "class 2"),col=c("green", "red"), lwd=c(2,2), cex=0.7)
```

We can notice that the graphs are a little bit similar in the range [5-20]. But values of class 2 has opposite values of class 1 in the range [30-100]



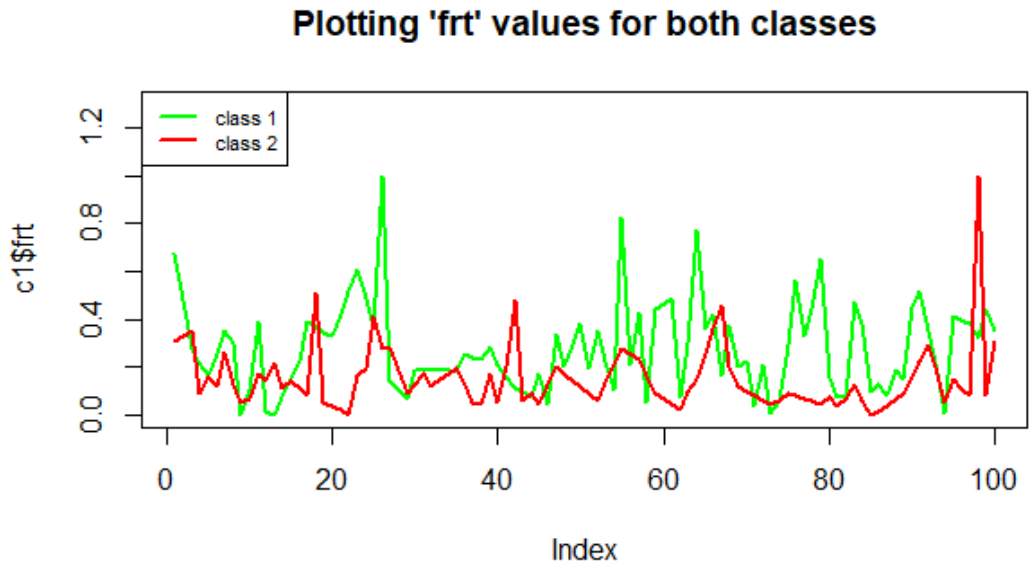
Now let's compare the distribution of "rrt" for both classes using this command:
plot(dfrrt,dfclass, type="p", col=c("red","green")[df$class],lwd=2, ylim=c(0.9,2.1))



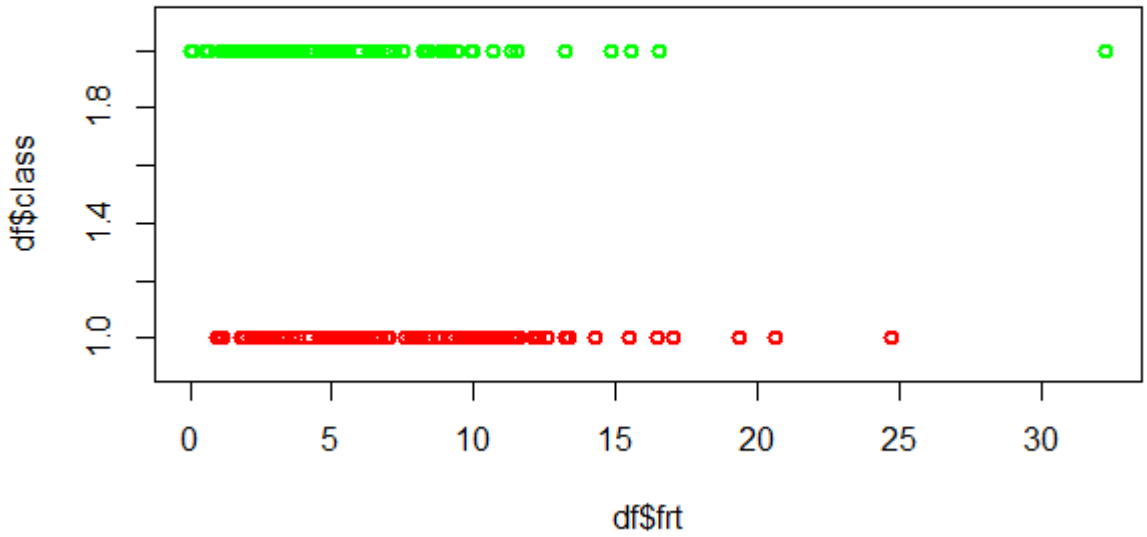
We notice that the variable rrt has different values and density function for each class.

line-plot for "frt"

plot(c1$frt, type="l", col="green",lwd=2, ylim=c(0,1.3))
lines(c2$frt, col="red", lwd=2)
title("Plotting 'frt' values for both classes")
legend("topleft", legend=c("class 1", "class 2"),col=c("green", "red"), lwd=c(2,2), cex=0.7)
Again we can notice that the graphs are very similar for some intervals . But the values of class 1 are much bigger.



Now let's compare the distribution of "frt" for both classes using this command:
plot(dffrt,dfclass, type="p", col=c("red","green")[df$class],lwd=2, ylim=c(0.9,2.1))



We notice that the variable frt has almost same values and density function for both class. THUS WE CONCLUDE THAT IT IS A REDUNDANT DATA and can be deleted when comparing the classes or performing a classification task.

==========

T7: First, determine how similar the variables **tpthrt** and **pkthrt** are for each **data class**, and then determine how similar **tpthrt of data class 1** and **tpthrt of data class 2** by using similarity metric (correlation).

For class 1:

c1 <- subset(df %>% filter(df$class==1), select = -c(class))
cor(subset(c1,select= c(tpthrt,pkthrt)))
output:

	tpthrt	pkthrt

tpthrt	**1.0000000**	**0.6007039**
pkthrt	**0.6007039**	**1.0000000**

This means the variables "tpthrt" and "pkthrt" of class 1 are 60.07039% similar

For class 2:

c2 <- subset(df %>% filter(df$class==2), select = -c(class))
cor(subset(c2,select= c(tpthrt,pkthrt)))
output:

	tpthrt	**pkthrt**
tpthrt	**1.0000000**	**0.2726601**
pkthrt	**0.2726601**	**1.0000000**

This means the variables "tpthrt" and "pkthrt" of class 2 are 27.26601% similar

Comparing "tpthr" of both classes:

cor(subset(c1,select= c(tpthrt)),subset(c2,select= c(tpthrt)))
output:

	tpthrt
tpthrt	**-0.1982791**

This means the variable "tpthrt" has similarity value of -0.1982791 between both classes.