

Myocardial infarction complications

Akastia Christo

2022-08-29

Contents

1	Introduction	3
1.1	Data	3
1.2	Research question	3
2	EDA of myocardial infarction complications data	4
2.1	Missing NA	12
2.2	Visualizing the data set of myocardial infraction	14
3	Cleaning the data	21
4	Determine quality metrics	24
4.1	Weka	24
4.2	Investigating performance of Machine Learning algorithms	25
5	Exploring Meta-learners and optimize a selecton of algorithms	28
5.1	Attribute selection	28
5.2	Meta learners	30
6	ROC and learning curve analysis	32
6.1	ROC curve analysis	32
7	Learning curve	36

```
## Libraries
options(digits = 3)
library(dplyr)
library(ggplot2)
library(tidyr)
library(readr)
library(gridExtra)
library(GGally)
library(purrr)
```

```
library(psych)
library(kableExtra)
library(ggbiplot)
library(png)
library(ggpubr)
library(cowplot)
```

1 Introduction

Myocardial infarction (MI), is known as heart attacks happens when one or more areas of the heart muscles do not get enough oxygen which can happen when the blood flow of the heart muscle is blocked [1] MI is one of the most challenging problems of the modern medicine. Because the course of disease in patients with MI differs each patient. MI can happen with or without complications that do not affect the long-term prognosis. Approximately half of the patients in the acute and subacute period experience complication that results in worsening of the disease or death. Even an experienced specialist may not be able to predict the development of these complications. Acute MI is associated with high mortality in the first year after it. The incidence of MI remains high in all countries. This is especially true for the urban population of highly developed countries, which is exposed to chronic stress factors, irregular and not always balanced nutrition. [2]

1.1 Data

Myocardial infarction complications Database was collected in the Krasnoyarsk Interdistrict Clinical Hospital No20 named after I. S. Berzon (Russia) in 1992-1995. Database contains 1700 records (patients), 111 input features and 12 complications. In the database contains 7.6% of missing values.

The codebook for this dataset can be found as a pdf and is available in the repo. Because the data set is large it is difficult to make a codebook from it.

1.2 Research question

Can you predict the complications of the patients after the third day of the admission based on the admission period using the machine learning? And which complications did lead to worsening of the MI and does the age play a role?

2 EDA of myocardial infarction complications data

Before the data is being visualized, the data is loaded first. After the data is loaded, the data can be used.

```
# Defining the data file and reading the file
data_file <- "data/Myocardial_infarction_complications_Database.csv"
Myocardial <- read.table(data_file, sep=";", header = TRUE, na.strings = "?")

# Show the data
str(Myocardial)
```

```
## 'data.frame':    1700 obs. of  124 variables:
## $ ID             : int  1 2 3 4 5 6 7 8 9 10 ...
## $ AGE            : int  77 55 52 68 60 64 70 65 60 77 ...
## $ SEX            : int  1 1 1 0 1 1 1 1 1 0 ...
## $ INF_ANAM       : int  2 1 0 0 0 0 1 0 0 2 ...
## $ STENOK_AN      : int  1 0 0 0 0 1 1 1 0 0 ...
## $ FK_STENOK      : int  1 0 0 0 0 2 2 1 0 0 ...
## $ IBS_POST       : int  2 0 2 2 2 1 1 2 2 0 ...
## $ IBS_NASL       : int  NA 0 NA NA NA NA NA NA NA ...
## $ GB             : int  3 0 2 2 3 0 2 2 2 3 ...
## $ SIM_GIPERT     : int  0 0 0 0 0 0 0 0 0 0 ...
## $ DLIT_AG        : int  7 0 2 3 7 0 7 7 6 6 ...
## $ ZSN_A          : int  0 0 0 1 0 0 1 0 0 1 ...
## $ nr_11          : int  0 0 0 0 0 0 0 0 0 0 ...
## $ nr_01          : int  0 0 0 0 0 0 0 0 0 0 ...
## $ nr_02          : int  0 0 0 0 0 0 0 0 0 0 ...
## $ nr_03          : int  0 0 0 0 0 0 0 0 0 0 ...
## $ nr_04          : int  0 0 0 0 0 0 0 0 0 0 ...
## $ nr_07          : int  0 0 0 0 0 0 0 0 0 0 ...
## $ nr_08          : int  0 0 0 0 0 0 0 0 0 0 ...
## $ np_01          : int  0 0 0 0 0 0 0 0 0 0 ...
## $ np_04          : int  0 0 0 0 0 0 0 0 0 0 ...
## $ np_05          : int  0 0 0 0 0 0 0 0 0 0 ...
## $ np_07          : int  0 0 0 0 0 0 0 0 0 0 ...
## $ np_08          : int  0 0 0 0 0 0 0 0 0 0 ...
## $ np_09          : int  0 0 0 0 0 0 0 0 0 0 ...
## $ np_10          : int  0 0 0 0 0 0 0 0 0 0 ...
## $ endocr_01      : int  0 0 0 0 0 0 0 0 0 1 ...
## $ endocr_02      : int  0 0 0 0 0 0 0 0 0 0 ...
## $ endocr_03      : int  0 0 0 0 0 0 0 0 0 0 ...
## $ zab_leg_01     : int  0 0 0 1 0 0 1 0 0 0 ...
## $ zab_leg_02     : int  0 0 0 0 0 0 0 0 0 0 ...
## $ zab_leg_03     : int  0 0 0 0 0 0 0 0 0 0 ...
## $ zab_leg_04     : int  0 0 0 0 0 0 0 0 0 0 ...
## $ zab_leg_06     : int  0 0 0 0 0 0 0 0 0 0 ...
## $ S_AD_KBRIG     : int  NA NA 150 NA 190 NA 120 NA 200 NA ...
## $ D_AD_KBRIG     : int  NA NA 100 NA 100 NA 80 NA 120 NA ...
## $ S_AD_ORIT      : int  180 120 180 120 160 140 120 145 195 200 ...
## $ D_AD_ORIT      : int  100 90 100 70 90 90 80 95 120 100 ...
## $ O_L_POST       : int  0 0 0 0 0 0 0 0 0 0 ...
## $ K_SH_POST      : int  0 0 0 0 0 0 0 0 0 0 ...
```

```

## $ MP_TP_POST : int 0 0 0 0 0 0 0 0 0 0 ...
## $ SVT_POST : int 0 0 0 0 0 0 0 0 0 0 ...
## $ GT_POST : int 0 0 0 0 0 0 0 0 0 0 ...
## $ FIB_G_POST : int 0 0 0 0 0 0 0 0 0 0 ...
## $ ant_im : int 1 4 4 0 4 1 0 0 0 4 ...
## $ lat_im : int 0 1 1 1 1 1 0 0 0 1 ...
## $ inf_im : int 0 0 0 1 0 0 3 2 3 0 ...
## $ post_im : int 0 0 0 0 0 0 0 0 2 0 ...
## $ IM_PG_P : int 0 0 0 0 0 0 0 0 0 0 ...
## $ ritm_ecg_p_01: int 0 1 1 1 0 0 1 1 1 0 ...
## $ ritm_ecg_p_02: int 0 0 0 0 0 0 0 0 0 0 ...
## $ ritm_ecg_p_04: int 0 0 0 0 0 0 0 0 0 0 ...
## $ ritm_ecg_p_06: int 0 0 0 0 0 0 0 0 0 0 ...
## $ ritm_ecg_p_07: int 1 0 0 0 1 1 0 0 0 1 ...
## $ ritm_ecg_p_08: int 0 0 0 0 0 0 0 0 0 0 ...
## $ n_r_ecg_p_01 : int 0 0 0 0 0 0 0 0 1 0 ...
## $ n_r_ecg_p_02 : int 0 0 0 0 0 0 0 0 0 0 ...
## $ n_r_ecg_p_03 : int 0 0 1 0 0 0 0 0 0 1 ...
## $ n_r_ecg_p_04 : int 0 1 0 0 0 0 0 0 0 0 ...
## $ n_r_ecg_p_05 : int 1 0 0 0 0 0 0 0 0 0 ...
## $ n_r_ecg_p_06 : int 0 0 0 0 0 0 0 0 0 0 ...
## $ n_r_ecg_p_08 : int 0 0 0 0 0 0 0 0 0 0 ...
## $ n_r_ecg_p_09 : int 0 0 0 0 0 0 0 0 0 0 ...
## $ n_r_ecg_p_10 : int 0 0 0 0 0 0 0 0 0 0 ...
## $ n_p_ecg_p_01 : int 0 0 0 0 0 0 0 0 0 0 ...
## $ n_p_ecg_p_03 : int 0 0 0 0 0 0 0 0 0 0 ...
## $ n_p_ecg_p_04 : int 0 0 0 0 0 0 0 0 0 0 ...
## $ n_p_ecg_p_05 : int 0 0 0 0 0 0 0 0 0 0 ...
## $ n_p_ecg_p_06 : int 0 0 0 0 0 0 0 0 0 0 ...
## $ n_p_ecg_p_07 : int 0 0 0 0 0 0 0 0 0 0 ...
## $ n_p_ecg_p_08 : int 1 0 0 0 0 0 0 0 0 0 ...
## $ n_p_ecg_p_09 : int 0 0 0 0 0 0 0 0 0 0 ...
## $ n_p_ecg_p_10 : int 0 0 0 0 0 0 0 0 0 0 ...
## $ n_p_ecg_p_11 : int 0 0 0 0 0 0 0 0 0 0 ...
## $ n_p_ecg_p_12 : int 0 0 0 0 0 0 0 0 0 0 ...
## $ fibr_ter_01 : int 0 0 0 0 0 0 0 0 0 0 ...
## $ fibr_ter_02 : int 0 0 0 0 0 0 0 0 0 0 ...
## $ fibr_ter_03 : int 0 0 0 0 0 0 0 0 0 0 ...
## $ fibr_ter_05 : int 0 0 0 0 0 0 0 0 0 0 ...
## $ fibr_ter_06 : int 0 0 0 0 0 0 0 0 0 0 ...
## $ fibr_ter_07 : int 0 0 0 0 0 0 0 0 0 0 ...
## $ fibr_ter_08 : int 0 0 0 0 0 0 0 0 0 0 ...
## $ GIPO_K : int 0 1 0 1 1 NA NA 0 NA NA ...
## $ K_BLOOD : num 4.7 3.5 4 3.9 3.5 NA NA 4.5 NA NA ...
## $ GIPER_NA : int 0 0 0 0 0 NA NA 0 NA NA ...
## $ NA_BLOOD : int 138 132 132 146 132 NA NA 136 NA NA ...
## $ ALT_BLOOD : num NA 0.38 0.3 0.75 0.45 0.45 0.3 NA 0.3 0.38 ...
## $ AST_BLOOD : num NA 0.18 0.11 0.37 0.22 0.22 0.11 NA 0.37 0.11 ...
## $ KFK_BLOOD : num NA NA NA NA NA NA NA NA NA NA ...
## $ L_BLOOD : num 8 7.8 10.8 NA 8.3 7.2 11.1 6.2 6.2 6.9 ...
## $ ROE : int 16 3 NA NA NA 2 5 20 3 30 ...
## $ TIME_B_S : int 4 2 3 2 9 2 1 7 3 3 ...
## $ R_AB_1_n : int 0 0 3 0 0 0 0 3 0 0 ...
## $ R_AB_2_n : int 0 0 0 0 0 0 0 0 0 0 ...

```

```
## $ R_AB_3_n      : int  1 0 0 1 0 0 0 0 0 0 ...
## $ NA_KB         : int  NA 1 1 NA 0 0 0 0 0 NA ...
## $ NOT_NA_KB     : int  NA 0 1 NA 0 1 1 0 1 NA ...
## $ LID_KB        : int  NA 1 1 NA 0 0 0 0 0 NA ...
## $ NITR_S        : int  0 0 0 0 0 0 0 0 0 0 ...
## [list output truncated]
```

Shows the first 6 rows of the myocardial infraction data set
head(Myocardial)

```
## ID AGE SEX INF_ANAM STENOK_AN FK_STENOK IBS_POST IBS_NASL GB SIM_GIPERT
## 1 1 77 1 2 1 1 2 NA 3 0
## 2 2 55 1 1 0 0 0 0 0 0
## 3 3 52 1 0 0 0 2 NA 2 0
## 4 4 68 0 0 0 0 2 NA 2 0
## 5 5 60 1 0 0 0 2 NA 3 0
## 6 6 64 1 0 1 2 1 NA 0 0
## DLIT_AG ZSN_A nr_11 nr_01 nr_02 nr_03 nr_04 nr_07 nr_08 np_01 np_04 np_05
## 1 7 0 0 0 0 0 0 0 0 0 0 0
## 2 0 0 0 0 0 0 0 0 0 0 0 0
## 3 2 0 0 0 0 0 0 0 0 0 0 0
## 4 3 1 0 0 0 0 0 0 0 0 0 0
## 5 7 0 0 0 0 0 0 0 0 0 0 0
## 6 0 0 0 0 0 0 0 0 0 0 0 0
## np_07 np_08 np_09 np_10 endocr_01 endocr_02 endocr_03 zab_leg_01 zab_leg_02
## 1 0 0 0 0 0 0 0 0 0 0
## 2 0 0 0 0 0 0 0 0 0 0
## 3 0 0 0 0 0 0 0 0 0 0
## 4 0 0 0 0 0 0 0 1 0
## 5 0 0 0 0 0 0 0 0 0
## 6 0 0 0 0 0 0 0 0 0
## zab_leg_03 zab_leg_04 zab_leg_06 S_AD_KBRIG D_AD_KBRIG S_AD_ORIT D_AD_ORIT
## 1 0 0 0 NA NA 180 100
## 2 0 0 0 NA NA 120 90
## 3 0 0 0 150 100 180 100
## 4 0 0 0 NA NA 120 70
## 5 0 0 0 190 100 160 90
## 6 0 0 0 NA NA 140 90
## O_L_POST K_SH_POST MP_TP_POST SVT_POST GT_POST FIB_G_POST ant_im lat_im
## 1 0 0 0 0 0 0 1 0
## 2 0 0 0 0 0 0 4 1
## 3 0 0 0 0 0 0 4 1
## 4 0 0 0 0 0 0 0 1
## 5 0 0 0 0 0 0 4 1
## 6 0 0 0 0 0 0 1 1
## inf_im post_im IM_PG_P ritm_ecg_p_01 ritm_ecg_p_02 ritm_ecg_p_04
## 1 0 0 0 0 0 0
## 2 0 0 0 1 0 0
## 3 0 0 0 1 0 0
## 4 1 0 0 1 0 0
## 5 0 0 0 0 0 0
## 6 0 0 0 0 0 0
## ritm_ecg_p_06 ritm_ecg_p_07 ritm_ecg_p_08 n_r_ecg_p_01 n_r_ecg_p_02
## 1 0 1 0 0 0
```

## 2	0	0	0	0	0					
## 3	0	0	0	0	0					
## 4	0	0	0	0	0					
## 5	0	1	0	0	0					
## 6	0	1	0	0	0					
##	n_r_ecg_p_03	n_r_ecg_p_04	n_r_ecg_p_05	n_r_ecg_p_06	n_r_ecg_p_08	n_r_ecg_p_09				
## 1	0	0	1	0	0	0				
## 2	0	1	0	0	0	0				
## 3	1	0	0	0	0	0				
## 4	0	0	0	0	0	0				
## 5	0	0	0	0	0	0				
## 6	0	0	0	0	0	0				
##	n_r_ecg_p_10	n_p_ecg_p_01	n_p_ecg_p_03	n_p_ecg_p_04	n_p_ecg_p_05	n_p_ecg_p_06				
## 1	0	0	0	0	0	0				
## 2	0	0	0	0	0	0				
## 3	0	0	0	0	0	0				
## 4	0	0	0	0	0	0				
## 5	0	0	0	0	0	0				
## 6	0	0	0	0	0	0				
##	n_p_ecg_p_07	n_p_ecg_p_08	n_p_ecg_p_09	n_p_ecg_p_10	n_p_ecg_p_11	n_p_ecg_p_12				
## 1	0	1	0	0	0	0				
## 2	0	0	0	0	0	0				
## 3	0	0	0	0	0	0				
## 4	0	0	0	0	0	0				
## 5	0	0	0	0	0	0				
## 6	0	0	0	0	0	0				
##	fibr_ter_01	fibr_ter_02	fibr_ter_03	fibr_ter_05	fibr_ter_06	fibr_ter_07				
## 1	0	0	0	0	0	0				
## 2	0	0	0	0	0	0				
## 3	0	0	0	0	0	0				
## 4	0	0	0	0	0	0				
## 5	0	0	0	0	0	0				
## 6	0	0	0	0	0	0				
##	fibr_ter_08	GIPO_K	K_BLOOD	GIPER_NA	NA_BLOOD	ALT_BLOOD	AST_BLOOD	KFK_BLOOD		
## 1	0	0	4.7	0	138	NA	NA	NA		
## 2	0	1	3.5	0	132	0.38	0.18	NA		
## 3	0	0	4.0	0	132	0.30	0.11	NA		
## 4	0	1	3.9	0	146	0.75	0.37	NA		
## 5	0	1	3.5	0	132	0.45	0.22	NA		
## 6	0	NA	NA	NA	NA	0.45	0.22	NA		
##	L_BLOOD	ROE	TIME_B_S	R_AB_1_n	R_AB_2_n	R_AB_3_n	NA_KB	NOT_NA_KB	LID_KB	NITR_S
## 1	8.0	16	4	0	0	1	NA	NA	NA	0
## 2	7.8	3	2	0	0	0	1	0	1	0
## 3	10.8	NA	3	3	0	0	1	1	1	0
## 4	NA	NA	2	0	0	1	NA	NA	NA	0
## 5	8.3	NA	9	0	0	0	0	0	0	0
## 6	7.2	2	2	0	0	0	0	1	0	0
##	NA_R_1_n	NA_R_2_n	NA_R_3_n	NOT_NA_1_n	NOT_NA_2_n	NOT_NA_3_n	LID_S_n			
## 1	0	0	0	0	0	0	0	1		
## 2	0	0	0	1	0	0	0	1		
## 3	1	0	0	3	2	2	2	1		
## 4	0	0	0	0	0	0	0	0		
## 5	0	0	0	0	0	0	0	0		
## 6	0	0	0	0	0	0	0	0		

```

##   B_BLOK_S_n ANT_CA_S_n GEPAR_S_n ASP_S_n TIKL_S_n TRENT_S_n FIBR_PREDS
## 1           0           0           1           1           0           0           0
## 2           0           1           1           1           0           1           0
## 3           1           0           1           1           0           0           0
## 4           0           1           1           1           0           0           0
## 5           0           1           0           1           0           1           0
## 6           1           0           1           1           0           0           1
##   PRED_S_TAH JELUD_TAH FIBR_JELUD A_V_BLOK OTEK_LANC RAZRIV DRESSLER ZSN REC_IM
## 1           0           0           0           0           0           0           0 0 0
## 2           0           0           0           0           0           0           0 0 0
## 3           0           0           0           0           0           0           0 0 0
## 4           0           0           0           0           0           0           0 1 0
## 5           0           0           0           0           0           0           0 0 0
## 6           0           0           0           0           0           0           0 0 0
##   P_IM_STEN LET_IS
## 1           0           0
## 2           0           0
## 3           0           0
## 4           0           0
## 5           0           0
## 6           0           0

```

Loading the myocardial as a table to view all the data en giving the summary of the data set.

```

# Gives the summary of the data set of Myocardial infraction
summary(Myocardial)

```

```

##           ID           AGE           SEX           INF_ANAM           STENOK_AN
## Min.      : 1    Min.    :26.0    Min.    :0.000    Min.    :0.00    Min.    :0.0
## 1st Qu.: 426    1st Qu.:54.0    1st Qu.:0.000    1st Qu.:0.00    1st Qu.:0.0
## Median : 850    Median :63.0    Median :1.000    Median :0.00    Median :1.0
## Mean     : 850    Mean     :61.9    Mean     :0.626    Mean     :0.55    Mean     :2.3
## 3rd Qu.:1275    3rd Qu.:70.0    3rd Qu.:1.000    3rd Qu.:1.00    3rd Qu.:5.0
## Max.     :1700    Max.     :92.0    Max.     :1.000    Max.     :3.00    Max.     :6.0
##          NA's      :8          NA's      :4          NA's     :106
##   FK_STENOK   IBS_POST   IBS_NASL       GB       SIM_GIPERT
## Min.    :0.0    Min.    :0.0    Min.    :0      Min.    :0.00    Min.    :0.00
## 1st Qu.:0.0    1st Qu.:0.0    1st Qu.:0      1st Qu.:0.00    1st Qu.:0.00
## Median :2.0    Median :1.0    Median :0      Median :2.00    Median :0.00
## Mean     :1.2    Mean     :1.2    Mean     :0      Mean     :1.39    Mean     :0.03
## 3rd Qu.:2.0    3rd Qu.:2.0    3rd Qu.:1      3rd Qu.:2.00    3rd Qu.:0.00
## Max.     :4.0    Max.     :2.0    Max.     :1      Max.     :3.00    Max.     :1.00
## NA's     :73    NA's     :51    NA's     :1628  NA's     :9      NA's     :8
##   DLIT_AG   ZSN_A      nr_11      nr_01      nr_02
## Min.    :0.0    Min.    :0.0    Min.    :0.00    Min.    :0      Min.    :0.00
## 1st Qu.:0.0    1st Qu.:0.0    1st Qu.:0.00    1st Qu.:0      1st Qu.:0.00
## Median :3.0    Median :0.0    Median :0.00    Median :0      Median :0.00
## Mean     :3.3    Mean     :0.2    Mean     :0.03    Mean     :0      Mean     :0.01
## 3rd Qu.:7.0    3rd Qu.:0.0    3rd Qu.:0.00    3rd Qu.:0      3rd Qu.:0.00
## Max.     :7.0    Max.     :4.0    Max.     :1.00    Max.     :1      Max.     :1.00
## NA's     :248    NA's     :54    NA's     :21     NA's     :21     NA's     :21
##   nr_03      nr_04      nr_07      nr_08      np_01
## Min.    :0.00    Min.    :0.00    Min.    :0      Min.    :0      Min.    :0

```


##	1st Qu.:	0.00	1st Qu.:	0.00	1st Qu.:	0	1st Qu.:	0	1st Qu.:	0
##	Median :	0.00	Median :	0.00	Median :	0	Median :	0	Median :	0
##	Mean :	0.02	Mean :	0.02	Mean :	0	Mean :	0	Mean :	0
##	3rd Qu.:	0.00	3rd Qu.:	0.00	3rd Qu.:	0	3rd Qu.:	0	3rd Qu.:	0
##	Max. :	1.00	Max. :	1.00	Max. :	1	Max. :	1	Max. :	1
##	NA's :	21	NA's :	21	NA's :	21	NA's :	21	NA's :	18
##	np_04		np_05		np_07		np_08		np_09	
##	Min. :	0	Min. :	0.00	Min. :	0	Min. :	0	Min. :	0
##	1st Qu.:	0	1st Qu.:	0.00	1st Qu.:	0	1st Qu.:	0	1st Qu.:	0
##	Median :	0	Median :	0.00	Median :	0	Median :	0	Median :	0
##	Mean :	0	Mean :	0.01	Mean :	0	Mean :	0	Mean :	0
##	3rd Qu.:	0	3rd Qu.:	0.00	3rd Qu.:	0	3rd Qu.:	0	3rd Qu.:	0
##	Max. :	1	Max. :	1.00	Max. :	1	Max. :	1	Max. :	1
##	NA's :	18	NA's :	18	NA's :	18	NA's :	18	NA's :	18
##	np_10		endocr_01		endocr_02		endocr_03		zab_leg_01	
##	Min. :	0	Min. :	0.00	Min. :	0.00	Min. :	0.00	Min. :	0.00
##	1st Qu.:	0	1st Qu.:	0.00	1st Qu.:	0.00	1st Qu.:	0.00	1st Qu.:	0.00
##	Median :	0	Median :	0.00	Median :	0.00	Median :	0.00	Median :	0.00
##	Mean :	0	Mean :	0.13	Mean :	0.02	Mean :	0.01	Mean :	0.08
##	3rd Qu.:	0	3rd Qu.:	0.00	3rd Qu.:	0.00	3rd Qu.:	0.00	3rd Qu.:	0.00
##	Max. :	1	Max. :	1.00	Max. :	1.00	Max. :	1.00	Max. :	1.00
##	NA's :	18	NA's :	11	NA's :	10	NA's :	10	NA's :	7
##	zab_leg_02		zab_leg_03		zab_leg_04		zab_leg_06		S_AD_KBRIG	
##	Min. :	0.00	Min. :	0.00	Min. :	0.00	Min. :	0.00	Min. :	0
##	1st Qu.:	0.00	1st Qu.:	0.00	1st Qu.:	0.00	1st Qu.:	0.00	1st Qu.:	120
##	Median :	0.00	Median :	0.00	Median :	0.00	Median :	0.00	Median :	140
##	Mean :	0.07	Mean :	0.02	Mean :	0.01	Mean :	0.01	Mean :	137
##	3rd Qu.:	0.00	3rd Qu.:	0.00	3rd Qu.:	0.00	3rd Qu.:	0.00	3rd Qu.:	160
##	Max. :	1.00	Max. :	1.00	Max. :	1.00	Max. :	1.00	Max. :	260
##	NA's :	7	NA's :	7	NA's :	7	NA's :	7	NA's :	1076
##	D_AD_KBRIG		S_AD_ORIT		D_AD_ORIT		O_L_POST		K_SH_POST	
##	Min. :	0	Min. :	0	Min. :	0.0	Min. :	0.00	Min. :	0.00
##	1st Qu.:	70	1st Qu.:	120	1st Qu.:	80.0	1st Qu.:	0.00	1st Qu.:	0.00
##	Median :	80	Median :	130	Median :	80.0	Median :	0.00	Median :	0.00
##	Mean :	81	Mean :	135	Mean :	82.7	Mean :	0.07	Mean :	0.03
##	3rd Qu.:	90	3rd Qu.:	150	3rd Qu.:	90.0	3rd Qu.:	0.00	3rd Qu.:	0.00
##	Max. :	190	Max. :	260	Max. :	190.0	Max. :	1.00	Max. :	1.00
##	NA's :	1076	NA's :	267	NA's :	267	NA's :	12	NA's :	15
##	MP_TP_POST		SVT_POST		GT_POST		FIB_G_POST		ant_im	
##	Min. :	0.00	Min. :	0	Min. :	0	Min. :	0.00	Min. :	0.0
##	1st Qu.:	0.00	1st Qu.:	0	1st Qu.:	0	1st Qu.:	0.00	1st Qu.:	0.0
##	Median :	0.00	Median :	0	Median :	0	Median :	0.00	Median :	1.0
##	Mean :	0.07	Mean :	0	Mean :	0	Mean :	0.01	Mean :	1.6
##	3rd Qu.:	0.00	3rd Qu.:	0	3rd Qu.:	0	3rd Qu.:	0.00	3rd Qu.:	4.0
##	Max. :	1.00	Max. :	1	Max. :	1	Max. :	1.00	Max. :	4.0
##	NA's :	14	NA's :	12	NA's :	12	NA's :	12	NA's :	83
##	lat_im		inf_im		post_im		IM_PG_P		ritm_ecg_p_01	
##	Min. :	0.0	Min. :	0	Min. :	0.0	Min. :	0.000	Min. :	0.0
##	1st Qu.:	0.0	1st Qu.:	0	1st Qu.:	0.0	1st Qu.:	0.000	1st Qu.:	0.0
##	Median :	1.0	Median :	0	Median :	0.0	Median :	0.000	Median :	1.0
##	Mean :	0.9	Mean :	1	Mean :	0.3	Mean :	0.029	Mean :	0.7
##	3rd Qu.:	1.0	3rd Qu.:	2	3rd Qu.:	0.0	3rd Qu.:	0.000	3rd Qu.:	1.0
##	Max. :	4.0	Max. :	4	Max. :	4.0	Max. :	1.000	Max. :	1.0
##	NA's :	80	NA's :	80	NA's :	72	NA's :	1	NA's :	152

```

##      ritm_ecg_p_02  ritm_ecg_p_04  ritm_ecg_p_06  ritm_ecg_p_07  ritm_ecg_p_08
##  Min.      :0.0    Min.      :0    Min.      :0    Min.      :0.0    Min.      :0
##  1st Qu.:0.0    1st Qu.:0    1st Qu.:0    1st Qu.:0.0    1st Qu.:0
##  Median :0.0    Median :0    Median :0    Median :0.0    Median :0
##  Mean   :0.1    Mean   :0    Mean   :0    Mean   :0.2    Mean   :0
##  3rd Qu.:0.0    3rd Qu.:0    3rd Qu.:0    3rd Qu.:0.0    3rd Qu.:0
##  Max.    :1.0    Max.    :1    Max.    :1    Max.    :1.0    Max.    :1
##  NA's    :152    NA's    :152    NA's    :152    NA's    :152    NA's    :152
##      n_r_ecg_p_01  n_r_ecg_p_02  n_r_ecg_p_03  n_r_ecg_p_04  n_r_ecg_p_05
##  Min.      :0    Min.      :0    Min.      :0.0    Min.      :0    Min.      :0
##  1st Qu.:0    1st Qu.:0    1st Qu.:0.0    1st Qu.:0    1st Qu.:0
##  Median :0    Median :0    Median :0.0    Median :0    Median :0
##  Mean   :0    Mean   :0    Mean   :0.1    Mean   :0    Mean   :0
##  3rd Qu.:0    3rd Qu.:0    3rd Qu.:0.0    3rd Qu.:0    3rd Qu.:0
##  Max.    :1    Max.    :1    Max.    :1.0    Max.    :1    Max.    :1
##  NA's    :115    NA's    :115    NA's    :115    NA's    :115    NA's    :115
##      n_r_ecg_p_06  n_r_ecg_p_08  n_r_ecg_p_09  n_r_ecg_p_10  n_p_ecg_p_01
##  Min.      :0    Min.      :0    Min.      :0    Min.      :0    Min.      :0
##  1st Qu.:0    1st Qu.:0    1st Qu.:0    1st Qu.:0    1st Qu.:0
##  Median :0    Median :0    Median :0    Median :0    Median :0
##  Mean   :0    Mean   :0    Mean   :0    Mean   :0    Mean   :0
##  3rd Qu.:0    3rd Qu.:0    3rd Qu.:0    3rd Qu.:0    3rd Qu.:0
##  Max.    :1    Max.    :1    Max.    :1    Max.    :1    Max.    :1
##  NA's    :115    NA's    :115    NA's    :115    NA's    :115    NA's    :115
##      n_p_ecg_p_03  n_p_ecg_p_04  n_p_ecg_p_05  n_p_ecg_p_06  n_p_ecg_p_07
##  Min.      :0    Min.      :0    Min.      :0    Min.      :0    Min.      :0.0
##  1st Qu.:0    1st Qu.:0    1st Qu.:0    1st Qu.:0    1st Qu.:0.0
##  Median :0    Median :0    Median :0    Median :0    Median :0.0
##  Mean   :0    Mean   :0    Mean   :0    Mean   :0    Mean   :0.1
##  3rd Qu.:0    3rd Qu.:0    3rd Qu.:0    3rd Qu.:0    3rd Qu.:0.0
##  Max.    :1    Max.    :1    Max.    :1    Max.    :1    Max.    :1.0
##  NA's    :115    NA's    :115    NA's    :115    NA's    :115    NA's    :115
##      n_p_ecg_p_08  n_p_ecg_p_09  n_p_ecg_p_10  n_p_ecg_p_11  n_p_ecg_p_12
##  Min.      :0    Min.      :0    Min.      :0    Min.      :0    Min.      :0
##  1st Qu.:0    1st Qu.:0    1st Qu.:0    1st Qu.:0    1st Qu.:0
##  Median :0    Median :0    Median :0    Median :0    Median :0
##  Mean   :0    Mean   :0    Mean   :0    Mean   :0    Mean   :0
##  3rd Qu.:0    3rd Qu.:0    3rd Qu.:0    3rd Qu.:0    3rd Qu.:0
##  Max.    :1    Max.    :1    Max.    :1    Max.    :1    Max.    :1
##  NA's    :115    NA's    :115    NA's    :115    NA's    :115    NA's    :115
##      fibr_ter_01  fibr_ter_02  fibr_ter_03  fibr_ter_05  fibr_ter_06
##  Min.      :0.00  Min.      :0.00  Min.      :0.00  Min.      :0    Min.      :0.00
##  1st Qu.:0.00  1st Qu.:0.00  1st Qu.:0.00  1st Qu.:0    1st Qu.:0.00
##  Median :0.00  Median :0.00  Median :0.00  Median :0    Median :0.00
##  Mean   :0.01  Mean   :0.01  Mean   :0.04  Mean   :0    Mean   :0.01
##  3rd Qu.:0.00  3rd Qu.:0.00  3rd Qu.:0.00  3rd Qu.:0    3rd Qu.:0.00
##  Max.    :1.00  Max.    :1.00  Max.    :1.00  Max.    :1    Max.    :1.00
##  NA's    :10    NA's    :10    NA's    :10    NA's    :10    NA's    :10
##      fibr_ter_07  fibr_ter_08  GIPO_K  K_BLOOD  GIPER_NA
##  Min.      :0    Min.      :0    Min.      :0    Min.      :2    Min.      :0
##  1st Qu.:0    1st Qu.:0    1st Qu.:0    1st Qu.:4    1st Qu.:0
##  Median :0    Median :0    Median :0    Median :4    Median :0
##  Mean   :0    Mean   :0    Mean   :0    Mean   :4    Mean   :0
##  3rd Qu.:0    3rd Qu.:0    3rd Qu.:1    3rd Qu.:5    3rd Qu.:0

```

##	Max.	:1	Max.	:1	Max.	:1	Max.	:8	Max.	:1
##	NA's	:10	NA's	:10	NA's	:369	NA's	:371	NA's	:375
##	NA_BLOOD		ALT_BLOOD		AST_BLOOD		KFK_BLOOD		L_BLOOD	
##	Min.	:117	Min.	:0.0	Min.	:0.0	Min.	:1	Min.	: 2.0
##	1st Qu.:	:133	1st Qu.:	:0.2	1st Qu.:	:0.1	1st Qu.:	:1	1st Qu.:	: 6.4
##	Median	:136	Median	:0.4	Median	:0.2	Median	:2	Median	: 8.0
##	Mean	:137	Mean	:0.5	Mean	:0.3	Mean	:2	Mean	: 8.8
##	3rd Qu.:	:140	3rd Qu.:	:0.6	3rd Qu.:	:0.3	3rd Qu.:	:2	3rd Qu.:	:10.4
##	Max.	:169	Max.	:3.0	Max.	:2.1	Max.	:4	Max.	:27.9
##	NA's	:375	NA's	:284	NA's	:285	NA's	:1696	NA's	:125
##	ROE		TIME_B_S		R_AB_1_n		R_AB_2_n		R_AB_3_n	
##	Min.	: 1.0	Min.	:1.0	Min.	:0.00	Min.	:0.0	Min.	:0.0
##	1st Qu.:	: 5.0	1st Qu.:	:2.0	1st Qu.:	:0.00	1st Qu.:	:0.0	1st Qu.:	:0.0
##	Median	: 10.0	Median	:4.0	Median	:0.00	Median	:0.0	Median	:0.0
##	Mean	: 13.4	Mean	:4.7	Mean	:0.32	Mean	:0.1	Mean	:0.1
##	3rd Qu.:	: 18.0	3rd Qu.:	:7.0	3rd Qu.:	:0.00	3rd Qu.:	:0.0	3rd Qu.:	:0.0
##	Max.	:140.0	Max.	:9.0	Max.	:3.00	Max.	:3.0	Max.	:3.0
##	NA's	:203	NA's	:126	NA's	:16	NA's	:108	NA's	:128
##	NA_KB		NOT_NA_KB		LID_KB		NITR_S		NA_R_1_n	
##	Min.	:0	Min.	:0	Min.	:0	Min.	:0.00	Min.	:0.00
##	1st Qu.:	:0	1st Qu.:	:0	1st Qu.:	:0	1st Qu.:	:0.00	1st Qu.:	:0.00
##	Median	:1	Median	:1	Median	:0	Median	:0.00	Median	:0.00
##	Mean	:1	Mean	:1	Mean	:0	Mean	:0.12	Mean	:0.48
##	3rd Qu.:	:1	3rd Qu.:	:1	3rd Qu.:	:1	3rd Qu.:	:0.00	3rd Qu.:	:1.00
##	Max.	:1	Max.	:1	Max.	:1	Max.	:1.00	Max.	:4.00
##	NA's	:657	NA's	:686	NA's	:677	NA's	:9	NA's	:5
##	NA_R_2_n		NA_R_3_n		NOT_NA_1_n		NOT_NA_2_n		NOT_NA_3_n	
##	Min.	:0.0	Min.	:0.0	Min.	:0.00	Min.	:0.0	Min.	:0.0
##	1st Qu.:	:0.0	1st Qu.:	:0.0	1st Qu.:	:0.00	1st Qu.:	:0.0	1st Qu.:	:0.0
##	Median	:0.0	Median	:0.0	Median	:0.00	Median	:0.0	Median	:0.0
##	Mean	:0.1	Mean	:0.1	Mean	:0.33	Mean	:0.1	Mean	:0.1
##	3rd Qu.:	:0.0	3rd Qu.:	:0.0	3rd Qu.:	:1.00	3rd Qu.:	:0.0	3rd Qu.:	:0.0
##	Max.	:3.0	Max.	:2.0	Max.	:4.00	Max.	:3.0	Max.	:2.0
##	NA's	:108	NA's	:131	NA's	:10	NA's	:110	NA's	:131
##	LID_S_n		B_BLOK_S_n		ANT_CA_S_n		GEPAR_S_n		ASP_S_n	
##	Min.	:0.00	Min.	:0.00	Min.	:0.00	Min.	:0.00	Min.	:0.00
##	1st Qu.:	:0.00	1st Qu.:	:0.00	1st Qu.:	:0.00	1st Qu.:	:0.00	1st Qu.:	:0.00
##	Median	:0.00	Median	:0.00	Median	:1.00	Median	:1.00	Median	:1.00
##	Mean	:0.28	Mean	:0.13	Mean	:0.67	Mean	:0.71	Mean	:0.74
##	3rd Qu.:	:1.00	3rd Qu.:	:0.00	3rd Qu.:	:1.00	3rd Qu.:	:1.00	3rd Qu.:	:1.00
##	Max.	:1.00	Max.	:1.00	Max.	:1.00	Max.	:1.00	Max.	:1.00
##	NA's	:10	NA's	:11	NA's	:13	NA's	:17	NA's	:17
##	TIKL_S_n		TRENT_S_n		FIBR_PREDS		PREDS_TAH		JELUD_TAH	
##	Min.	:0.00	Min.	:0.0	Min.	:0.0	Min.	:0.000	Min.	:0.000
##	1st Qu.:	:0.00	1st Qu.:	:0.0	1st Qu.:	:0.0	1st Qu.:	:0.000	1st Qu.:	:0.000
##	Median	:0.00	Median	:0.0	Median	:0.0	Median	:0.000	Median	:0.000
##	Mean	:0.02	Mean	:0.2	Mean	:0.1	Mean	:0.012	Mean	:0.025
##	3rd Qu.:	:0.00	3rd Qu.:	:0.0	3rd Qu.:	:0.0	3rd Qu.:	:0.000	3rd Qu.:	:0.000
##	Max.	:1.00	Max.	:1.0	Max.	:1.0	Max.	:1.000	Max.	:1.000
##	NA's	:16	NA's	:16						
##	FIBR_JELUD		A_V_BLOK		OTEK_LANC		RAZRIV			
##	Min.	:0.000	Min.	:0.000	Min.	:0.000	Min.	:0.000		
##	1st Qu.:	:0.000	1st Qu.:	:0.000	1st Qu.:	:0.000	1st Qu.:	:0.000		
##	Median	:0.000	Median	:0.000	Median	:0.000	Median	:0.000		

```
## Mean      :0.042    Mean      :0.034    Mean      :0.094    Mean      :0.032
## 3rd Qu.   :0.000    3rd Qu.   :0.000    3rd Qu.   :0.000    3rd Qu.   :0.000
## Max.      :1.000    Max.      :1.000    Max.      :1.000    Max.      :1.000
##
## DRESSLER      ZSN      REC_IM      P_IM_STEN      LET_IS
## Min.         :0.000    Min.         :0.000    Min.         :0.000    Min.         :0.000    Min.         :0.00
## 1st Qu.      :0.000    1st Qu.      :0.000    1st Qu.      :0.000    1st Qu.      :0.000    1st Qu.      :0.00
## Median       :0.000    Median       :0.000    Median       :0.000    Median       :0.000    Median       :0.00
## Mean         :0.044    Mean         :0.232    Mean         :0.094    Mean         :0.087    Mean         :0.48
## 3rd Qu.      :0.000    3rd Qu.      :0.000    3rd Qu.      :0.000    3rd Qu.      :0.000    3rd Qu.      :0.00
## Max.         :1.000    Max.         :1.000    Max.         :1.000    Max.         :1.000    Max.         :7.00
##
```

The results of the summary shows that there are some columns that have a lot of missing values. And that the most of the columns consist of the numbers 0 and 1. Except for a few other columns, that gives information about the content of the blood and ecg content.

2.1 Missing NA

To see if there are missing values in this data and giving the percentage of the missing values in data set.

```
# All missing values of the data set
kable(colSums(is.na(Myocardial)), col.names = "Counts of NA's",
      caption = "The counts of NA's of each column")
```

Looking at the table 1 it shows that the missing values are in examinations which were needed for a patient to see their state at the time of admission. Which explains why there is many missing values, because it doesn't apply for every patient.

Seeing the results of the table 1, it can be concluded that the missing values could be removed from the data set. Because the missing values do not add more information to the analysis.

Table 1: The counts of NA's of each column

	Counts of NA's
ID	0
AGE	8
SEX	0
INF_ANAM	4
STENOK_AN	106
FK_STENOK	73
IBS_POST	51
IBS_NASL	1628
GB	9
SIM_GIPERT	8
DLIT_AG	248
ZSN_A	54
nr_11	21
nr_01	21
nr_02	21
nr_03	21
nr_04	21
nr_07	21
nr_08	21
np_01	18
np_04	18
np_05	18
np_07	18
np_08	18
np_09	18
np_10	18
endocr_01	11
endocr_02	10
endocr_03	10
zab_leg_01	7
zab_leg_02	7
zab_leg_03	7
zab_leg_04	7
zab_leg_06	7
S_AD_KBRIG	1076
D_AD_KBRIG	1076
S_AD_ORIT	267
D_AD_ORIT	267
O_L_POST	12
K_SH_POST	15
MP_TP_POST	14
SVT_POST	12
GT_POST	12
FIB_G_POST	12
ant_im	83
lat_im	80
inf_im	80
post_im	72
IM_PG_P	1
ritm_ecg_p_01	152
ritm_ecg_p_02	152
ritm_ecg_p_04 ¹³	152
ritm_ecg_p_06	152
ritm_ecg_p_07	152
ritm_ecg_p_08	152

2.2 Visualizing the data set of myocardial infraction

The selected data is the blood pressures of the patients that has arrived. This violin plot shows the ages which has a higher blood pressure whether it is systolic or diastolic.

```
tmp <- Myocardial %>% select(2, 35:38) %>% drop_na()

df1 <- data.frame(x=tmp$AGE, y=tmp$S_AD_KBRIG)
df2 <- data.frame(x=tmp$AGE, y=tmp$D_AD_KBRIG)
df3 <- data.frame(x=tmp$AGE, y=tmp$S_AD_ORIT)
df4 <- data.frame(x=tmp$AGE, y=tmp$D_AD_ORIT)

ggplot(df1, aes(x,y)) +
  geom_violin(aes(color="Systolic blood pressure of ECT")) +
  geom_jitter(aes(color="Systolic blood pressure of ECT"),
    alpha=0.4) +
  geom_violin(data=df2, aes(color="Diastolic blood pressure of ECT")) +
  geom_jitter(aes(color="Diastolic blood pressure of ECT"),
    alpha=0.4) +
  geom_violin(data=df3, aes(color="Systolic blood pressure of ICU")) +
  geom_jitter(aes(color="Systolic blood pressure of ICU"),
    alpha=0.4) +
  geom_violin(data=df4, aes(color="Diastolic blood pressure of ICU")) +
  geom_jitter(aes(color="Diastolic blood pressure of ICU"),
    alpha=0.4) +
  xlab("Ages of patients (in years)") +
  ylab("blood pressures (in mmHg)") +
  labs(color="legend")
```

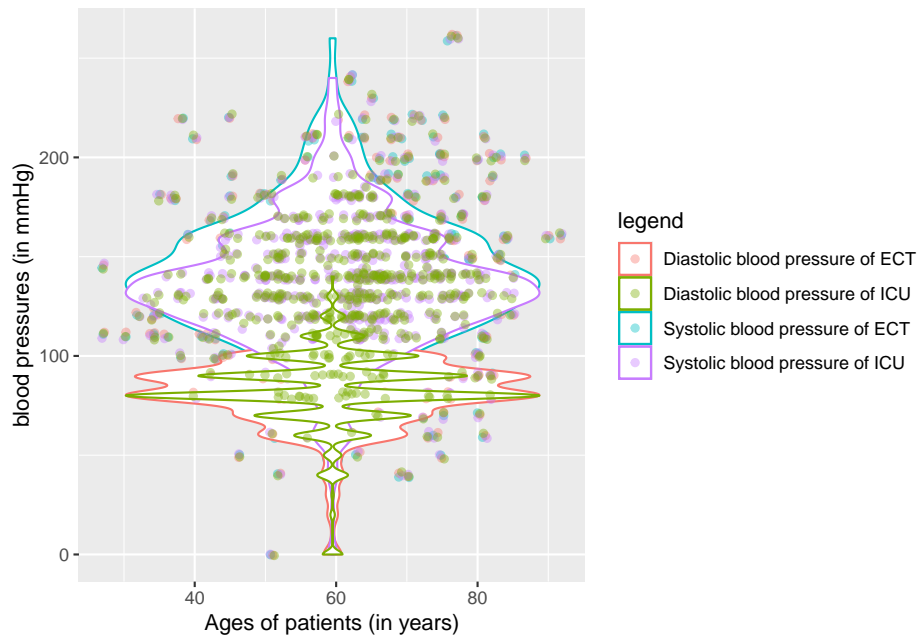


Figure 1: Blood pressures of patients according to the ECT (Emergency cardiology team) and ICU (Intensive care unit)

Figure 1 shows that there are a few outliers in this selected data set. The distribution is expected because most of the patients could have normal blood pressure. Most of the patients have normal diastolic blood pressure in this case, but the patients who measured the systolic blood pressure have a higher mmHg. This actively demonstrates that their heart pushed a lot of blood out. So when they have higher systolic blood pressure for an extended period, it can increase your risk of strokes and heart disease.

Making a bar chart to check if the time elapsed from the beginning of the attack of CHD to the hospital of all the patients relates to the risk of strokes or heart diseases.

```
ggplot(Myocardial, aes(x=factor(TIME_B_S))) +
  geom_bar(width = 0.7, fill = "orchid4") +
  xlab("Time elapsed from the start of the attack to the hospital") +
  ylab("Numbers of patients")
```

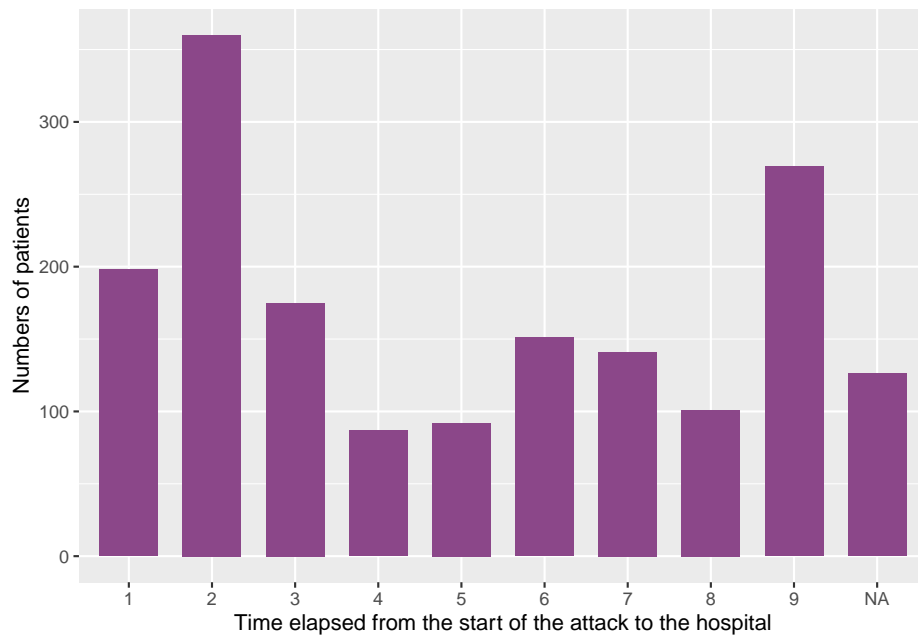


Figure 2: Time elapsed from the beginning of the attack of CHD to the hospital

This bar chart 2 shows that the the time elapsed is higher at the number 2. Number 2 is the time elapsed of 2-4 hours. Which could be concluded that the time elapsed is mostly around the time of 2 tot 4 hours for most of the patients that has been admitted to the hospital.

Mutating the data of the sex attribute to show a better figure.

```
# Mutating the sex attribute to make it readable.
Myocardial_SEX <- Myocardial %>%
  mutate(SEX = factor(SEX, labels = c("Female", "Male"), levels = c(0, 1)))
```

This figure shows the blood pressure with different sex and ages.

```
# Plots the blood pressures
SB_EC <- ggplot(Myocardial_SEX, aes(x= S_AD_KBRIG, y=AGE)) +
  geom_jitter(aes(color = SEX), alpha = 0.3) +
  xlab("Systolic bloodpressure\naccording to EC")
```

```

DB_EC <- ggplot(Myocardial_SEX, aes(x=D_AD_KBRIG, y=AGE)) +
  geom_jitter(aes(color = SEX), alpha = 0.3) +
  xlab("Diastolic bloodpressure\naccording to EC")
SB_IC <- ggplot(Myocardial_SEX, aes(x=S_AD_ORIT, y=AGE)) +
  geom_jitter(aes(color = SEX), alpha = 0.3) +
  xlab("Systolic bloodpressure\naccording to IC")
DB_IC <- ggplot(Myocardial_SEX, aes(x=D_AD_ORIT, y=AGE)) +
  geom_jitter(aes(color = SEX), alpha = 0.3) +
  xlab("Diastolic bloodpressure\naccording to IC")

# Combine the plots
plot_grid(SB_EC, DB_EC, SB_IC, DB_IC,
  labels = c("A", "B", "C", "D"),
  label_size = 12)

```

```

## Warning: Removed 1080 rows containing missing values (geom_point).
## Removed 1080 rows containing missing values (geom_point).

```

```

## Warning: Removed 273 rows containing missing values (geom_point).
## Removed 273 rows containing missing values (geom_point).

```

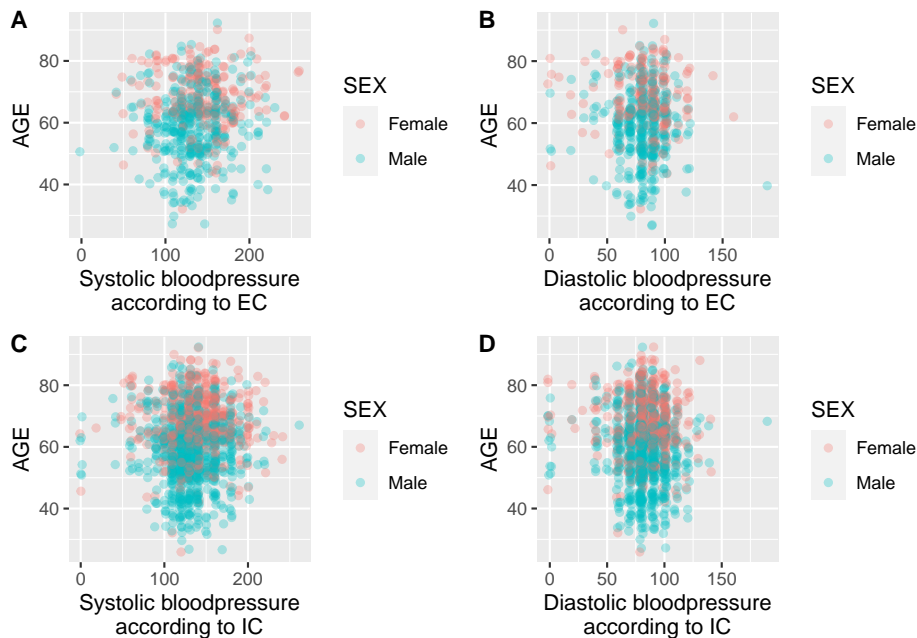


Figure 3: Blood pressure according to the Emergency Cardiology Team (EC) and Intensive Care Unit (IC) with the variables sex and ages

Figure 3 shows that most of the males in comparison to the females were younger than 60. And most of the females were older than 60. It could be concluded that the males have more risk to have a stroke or heart disease at a younger age.

Before making a figure to show the ages of the patients who had a ECG rhythm at the time of admission to the hospital of a sinus with a heart rate of below 60 to above 90. The data set has to be selected and mutated, to understand the distribution better.

Making a range of the heart rate with the ages of the patients to see if the patients fell in the category of the heart rate below 60, between 60 and 60 and above 90.

```
# Making a subset of the ECG to make a violin plot with it.
Myocardial.ECG <- Myocardial %>%
  select("AGE", "ritm_ecg_p_01", "ritm_ecg_p_07", "ritm_ecg_p_08") %>%
  pivot_longer(-AGE, names_to = "sinus_with_a_heart_rate",
               values_to = "heart_rate") %>% drop_na()

# Mutating the heart rate to yes and no, to understand it better
Myocardial.ECG <- Myocardial.ECG %>%
  mutate(heart_rate = factor(heart_rate, labels = c("No", "Yes"),
                           levels = c(0, 1)))

head(Myocardial.ECG)
```

```
## # A tibble: 6 x 3
##   AGE sinus_with_a_heart_rate heart_rate
##   <int> <chr>                  <fct>
## 1    77 ritm_ecg_p_01          No
## 2    77 ritm_ecg_p_07          Yes
## 3    77 ritm_ecg_p_08          No
## 4    55 ritm_ecg_p_01          Yes
## 5    55 ritm_ecg_p_07          No
## 6    55 ritm_ecg_p_08          No
```

This figure will show the distribution of the myocardial infarction heart rate data set. This violin plot will show the distribution of the myocardial infarction heart rate data set of the patients at the time of admission to the hospital.

```
# violin plot of the ECG rhythm
ggplot(Myocardial.ECG, aes(sinus_with_a_heart_rate, AGE, col = heart_rate))+
  geom_violin() +
  geom_jitter(alpha = 0.1) +
  scale_x_discrete(labels = c('heart rate 60-90', 'heart rate above 90',
                             'heart rate above below 60')) +
  ylab("Ages of the patients (in years)") +
  xlab("ECG rhythm (in sinus)")
```

Figure 4 shows that most of the patients didn't have a heart rate below 60 at the time of admission to the hospital. And it shows that most of the patients had a normal heart rhythm at the time of admission, so it could mean they don't have a problem with their heart rate. Patients with a heart rate above 90 should be observed because it could lead to heart diseases if they have an irregular heart rate.

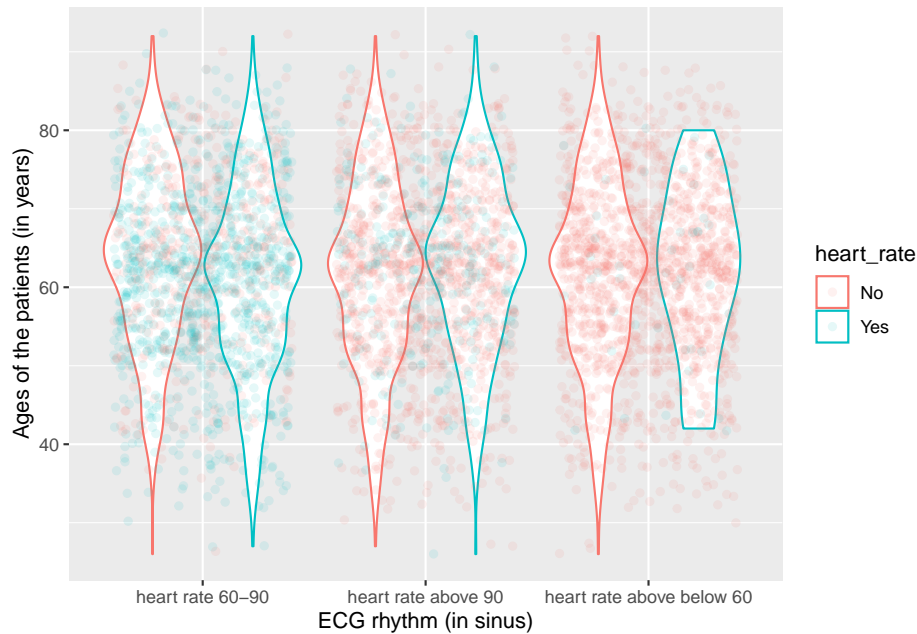


Figure 4: ECG rhythm at the time of admission to the hospital

2.2.1 Clustering of the myocardial infraction data set

Making a PCA to show the clustering of the data set.

```
# Selecting the data to make a PCA
rows <- nrow(Myocardial)

myo <- Myocardial %>%
  select(where(function(x){sum(is.na(x))/rows < 0.3})) %>%
  select(-ID) %>%
  select(where(function(x){sum(is.infinite(x)) == 0})) %>%
  select(where(function(x){(sum(x == 0, na.rm=T) / rows) < 0.8})) %>%
  drop_na()

myo_pca <- prcomp(as.matrix(myo),
  scale. = T,
  center = T)
# Plotting the pca
plot(myo_pca, type = "l")
```

Figure 5 shows that the variances of the PCA and that it decreases. With the 10-dimensional data, the PCA gives 10 principal components. This helps with organizing the information in Principal components to allow reducing the dimensional without losing information that is needed.

```
#shows the summary of PCA of the myocardial infraction
summary(myo_pca)
```

```
## Importance of components:
##          PC1      PC2      PC3      PC4      PC5      PC6      PC7      PC8
```

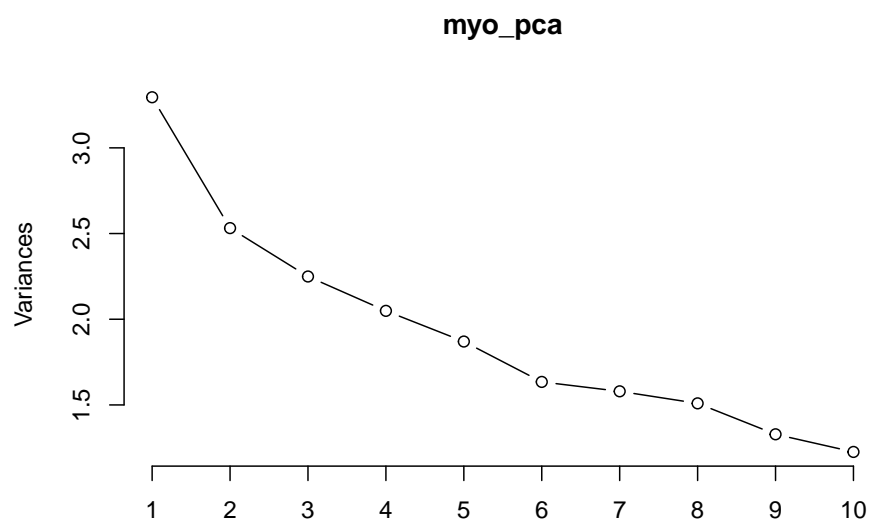


Figure 5: PCA of the classes that were important to see the clustering of the data were taken and set in a precomp we could see the variance of our PCA.

```
## Standard deviation      1.8152 1.5911 1.4998 1.4314 1.3675 1.2784 1.2568 1.2285
## Proportion of Variance 0.0998 0.0767 0.0682 0.0621 0.0567 0.0495 0.0479 0.0457
## Cumulative Proportion 0.0998 0.1766 0.2447 0.3068 0.3635 0.4130 0.4609 0.5066
##          PC9  PC10  PC11  PC12  PC13  PC14  PC15  PC16
## Standard deviation      1.1527 1.1072 1.0680 1.0322 0.9797 0.9511 0.9502 0.9230
## Proportion of Variance 0.0403 0.0372 0.0346 0.0323 0.0291 0.0274 0.0274 0.0258
## Cumulative Proportion 0.5469 0.5840 0.6186 0.6509 0.6800 0.7074 0.7347 0.7605
##          PC17  PC18  PC19  PC20  PC21  PC22  PC23  PC24
## Standard deviation      0.9158 0.9031 0.8733 0.8592 0.772 0.7559 0.7376 0.7197
## Proportion of Variance 0.0254 0.0247 0.0231 0.0224 0.018 0.0173 0.0165 0.0157
## Cumulative Proportion 0.7860 0.8107 0.8338 0.8562 0.874 0.8915 0.9080 0.9237
##          PC25  PC26  PC27  PC28  PC29  PC30  PC31
## Standard deviation      0.6980 0.6473 0.56594 0.55959 0.51528 0.46438 0.4298
## Proportion of Variance 0.0148 0.0127 0.00971 0.00949 0.00805 0.00653 0.0056
## Cumulative Proportion 0.9385 0.9512 0.96087 0.97036 0.97841 0.98494 0.9905
##          PC32  PC33
## Standard deviation      0.42410 0.36372
## Proportion of Variance 0.00545 0.00401
## Cumulative Proportion 0.99599 1.00000
```

This summary of PCA shows measures of the components. It shows the standard deviation, the proportion of the variance and the cumulative proportion. The proportion of the variance indicates that the data set has a low percentage of variability. This could be seen in the data set. This could be because most of the columns have only two variables, yes or no, in numeric levels, which results in low variability.

Making a PCA plot with the information of the PCA variances.

```
# Plot the PCA
g <- ggbiplot(myo_pca, obs.scale = 1, var.scale = 1, ellipse = FALSE, circle = TRUE)
```

```
g <- g + scale_color_discrete(name = '')
g <- g + theme(legend.direction = 'horizontal',
legend.position = 'top')
# Show plot
g
```

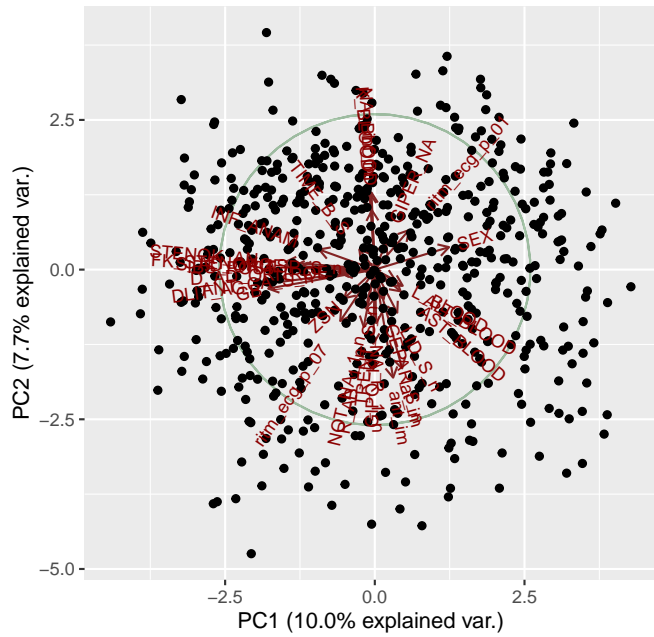


Figure 6: PCA plot of the myocardial of each complications with taking the group classes of the causes of the myocardial infraction.

Figure 6 shows a scatter plot of the variances of all the component. IT shows a variation of each principal component of the data where the x-axis shows the number of components and the y-axis shows the amount of variations. But 6 PC1 explains only 7.7% variations and PC2 explains only 10% variations. So this actively demonstrates that it doesn't show all of the variances of the myocardial infraction data.

3 Cleaning the data

To clean the data set, the column that contains NA's and columns that doesn't help with the further research are removed.

```
# Changing the variables names for the column LET_IS (Letset)
Myocardial_clean <- Myocardial %>%
  mutate(LET_IS = factor(LET_IS, labels = c("unknown", "cardiogenic shock", "pulmonary edema", "myocardial infarction")))

# Removing columns that contains NA's and columns that doesn't give more information to the data set.
Myocardial_clean <- Myocardial_clean[, -c(1, 8, 35, 36, 84, 93, 94, 97, 98, 100, 101, 103, 104, 113:123)]
```

Before the cleaned data is used for Weka, the labels of the gender column has to be changed to Female and Male to make the data set understandable.

```
Myocardial_clean <- Myocardial_clean %>%
  mutate(SEX = factor(SEX, labels = c("Female", "Male"), levels = c(0, 1)))
```

The head of the cleaned data to see the overview of it.

```
head(Myocardial_clean)
```

```
##   AGE    SEX INF_ANAM STENOK_AN FK_STENOK IBS_POST GB SIM_GIPERT DLIT_AG ZSN_A
## 1  77   Male      2      1      1      2  3      0      7      0
## 2  55   Male      1      0      0      0  0      0      0      0
## 3  52   Male      0      0      0      2  2      0      2      0
## 4  68 Female      0      0      0      2  2      0      3      1
## 5  60   Male      0      0      0      2  3      0      7      0
## 6  64   Male      0      1      2      1  0      0      0      0
##   nr_11 nr_01 nr_02 nr_03 nr_04 nr_07 nr_08 np_01 np_04 np_05 np_07 np_08 np_09
## 1     0     0     0     0     0     0     0     0     0     0     0     0
## 2     0     0     0     0     0     0     0     0     0     0     0     0
## 3     0     0     0     0     0     0     0     0     0     0     0     0
## 4     0     0     0     0     0     0     0     0     0     0     0     0
## 5     0     0     0     0     0     0     0     0     0     0     0     0
## 6     0     0     0     0     0     0     0     0     0     0     0     0
##   np_10 endocr_01 endocr_02 endocr_03 zab_leg_01 zab_leg_02 zab_leg_03
## 1     0         0         0         0         0         0         0
## 2     0         0         0         0         0         0         0
## 3     0         0         0         0         0         0         0
## 4     0         0         0         0         1         0         0
## 5     0         0         0         0         0         0         0
## 6     0         0         0         0         0         0         0
##   zab_leg_04 zab_leg_06 S_AD_ORIT D_AD_ORIT O_L_POST K_SH_POST MP_TP_POST
## 1           0         0      180      100      0         0         0
## 2           0         0      120       90      0         0         0
## 3           0         0      180      100      0         0         0
## 4           0         0      120       70      0         0         0
## 5           0         0      160       90      0         0         0
## 6           0         0      140       90      0         0         0
##   SVT_POST GT_POST FIB_G_POST ant_im lat_im inf_im post_im IM_PG_P
## 1         0     0         0      1     0     0         0         0
```

## 2	0	0	0	4	1	0	0	0	
## 3	0	0	0	4	1	0	0	0	
## 4	0	0	0	0	1	1	0	0	
## 5	0	0	0	4	1	0	0	0	
## 6	0	0	0	1	1	0	0	0	
##	ritm_ecg_p_01	ritm_ecg_p_02	ritm_ecg_p_04	ritm_ecg_p_06	ritm_ecg_p_07				
## 1	0	0	0	0	0	1			
## 2	1	0	0	0	0	0			
## 3	1	0	0	0	0	0			
## 4	1	0	0	0	0	0			
## 5	0	0	0	0	0	1			
## 6	0	0	0	0	0	1			
##	ritm_ecg_p_08	n_r_ecg_p_01	n_r_ecg_p_02	n_r_ecg_p_03	n_r_ecg_p_04				
## 1	0	0	0	0	0				
## 2	0	0	0	0	0	1			
## 3	0	0	0	1	0				
## 4	0	0	0	0	0	0			
## 5	0	0	0	0	0	0			
## 6	0	0	0	0	0	0			
##	n_r_ecg_p_05	n_r_ecg_p_06	n_r_ecg_p_08	n_r_ecg_p_09	n_r_ecg_p_10	n_p_ecg_p_01			
## 1	1	0	0	0	0	0			
## 2	0	0	0	0	0	0			
## 3	0	0	0	0	0	0			
## 4	0	0	0	0	0	0			
## 5	0	0	0	0	0	0			
## 6	0	0	0	0	0	0			
##	n_p_ecg_p_03	n_p_ecg_p_04	n_p_ecg_p_05	n_p_ecg_p_06	n_p_ecg_p_07	n_p_ecg_p_08			
## 1	0	0	0	0	0	1			
## 2	0	0	0	0	0	0			
## 3	0	0	0	0	0	0			
## 4	0	0	0	0	0	0			
## 5	0	0	0	0	0	0			
## 6	0	0	0	0	0	0			
##	n_p_ecg_p_09	n_p_ecg_p_10	n_p_ecg_p_11	n_p_ecg_p_12	fibr_ter_01	fibr_ter_02			
## 1	0	0	0	0	0	0			
## 2	0	0	0	0	0	0			
## 3	0	0	0	0	0	0			
## 4	0	0	0	0	0	0			
## 5	0	0	0	0	0	0			
## 6	0	0	0	0	0	0			
##	fibr_ter_03	fibr_ter_05	fibr_ter_06	fibr_ter_07	fibr_ter_08	GIPO_K	GIPER_NA		
## 1	0	0	0	0	0	0	0		
## 2	0	0	0	0	0	1	0		
## 3	0	0	0	0	0	0	0		
## 4	0	0	0	0	0	1	0		
## 5	0	0	0	0	0	1	0		
## 6	0	0	0	0	0	NA	NA		
##	NA_BLOOD	ALT_BLOOD	AST_BLOOD	KFK_BLOOD	L_BLOOD	ROE	TIME_B_S	R_AB_3_n	NA_KB
## 1	138	NA	NA	NA	8.0	16	4	1	NA
## 2	132	0.38	0.18	NA	7.8	3	2	0	1
## 3	132	0.30	0.11	NA	10.8	NA	3	0	1
## 4	146	0.75	0.37	NA	NA	NA	2	1	NA
## 5	132	0.45	0.22	NA	8.3	NA	9	0	0
## 6	NA	0.45	0.22	NA	7.2	2	2	0	0

##	NITR_S	NA_R_3_n	NOT_NA_3_n	LID_S_n	B_BLOK_S_n	ANT_CA_S_n	GEPAR_S_n	ASP_S_n
## 1	0	0	0	1	0	0	1	1
## 2	0	0	0	1	0	1	1	1
## 3	0	0	2	1	1	0	1	1
## 4	0	0	0	0	0	1	1	1
## 5	0	0	0	0	0	1	0	1
## 6	0	0	0	0	1	0	1	1

##	TIKL_S_n	TRENT_S_n	LET_IS
## 1	0	0	unknown
## 2	0	1	unknown
## 3	0	0	unknown
## 4	0	0	unknown
## 5	0	1	unknown
## 6	0	0	unknown

4 Determine quality metrics

In this section the quality metrics is going to be determined with the help of the program Weka. Following a machine learning performance, the accuracy metric, which is the default metric, is displayed. However, there are other metrics that can be used, which are more accurate, and are relevant to the data set.

After a machine learning performance shows the performance an accuracy which is the default metric. But other metrics can be applied and are more accuracy and relevant for the data set. For the dataset, there are 8 classes which has been categorized in unknown and all the lethal causes. It is important that these classes are correctly predicted and performed. Because if the test shows that is unknown than you don't want it to be a lethal cause. And when it is unknown you want to take others test to know what it could be other than a lethal cause

Thus, by raising the FP in this study. We don't want those who genuinely have cardiac diseases to be forecasted with the unknown, such as other diseases or doesn't have any conditions. In the confusion matrix, the FN is increased so that that FP decreases. The lethal causes, the TN, and the true positives, the unknown, are the outcomes from this data set that we want to maintain. This could be patients that didn't have any lethal reason in the aspect of heart diseases. And the results of the patients may not be related to heart diseases. However, it's also possible that those patients require further study.

As can be seen from table 2, where the first row represents the predicted class and the second and third rows represent the actual class of the myocardial infarction. Here stand FP by the predicted class unknown but the actual class is a lethal cause. It is crucial that the classification to be precise and quick because, if the test reveals that it is a lethal cause, it must be treated as soon as possible to prevent it from getting worse. Therefore, FP should be increased in a more accurate metric.

```
# Read confusion matrix csv
matrix <- read.csv("Weka_Data/matrix.csv", sep = ",", header = FALSE)

# Setting rownames and colnames to null
rownames(matrix) <- colnames(matrix) <- NULL
kable(matrix, caption = "Confusion matrix of the myocardial infraction data set") %>%
  kable_styling(latex_options = "HOLD_position")
```

Table 2: Confusion matrix of the myocardial infraction data set

Predicted		Unknown	Lethal cause
Real	Unknown	TP	FN
	Lethal cause	FP	TN

The confusion consist of the following layout in Weka: For this matrix is unknown positive and lethal cause negative.

=== Confusion Matrix ===

a b <- classified as

TP FN | a = Unknown

FP TN | b = Lethal cause ()

4.1 Weka

Creating a new csv file of the cleaned data set to use in Weka.

Table 3: Classification using cross validation 10-fold

Classifier	Speed	Accuracy	TP	FP	FN	TN
ZeroR	0.00	84.1	1429	271	0	0
OneR	0.11	88.6	1423	148	6	123
NaiveBayes	0.05	74.6	1195	85	234	189
SimpleLogistic	1.03	86.2	1424	235	5	49
SMO	1.18	85.5	1417	221	12	49
Ibk	0.00	80.6	1352	212	77	59
J48	0.59	80.6	1408	211	21	60
RandomForest	1.30	86.1	1428	236	1	35

```
# Creating a csv file for Weka
write.csv(Myocardial_clean, "../Data/Myocardial.csv", row.names = FALSE, na = "")
```

In Weka we are going to run a few classifiers and see which one does a good job on the data with the accuracy. Because the class has 8 variables it has to be grouped in two groups. Which I am going to make one which contains the unknown which could mean that there were no causes of the tests.

4.2 Investigating performance of Machine Learning algorithms

With the clean data set we are going to start investigating the performance of all the standard machine learning algorithms with the standard settings in Weka.

```
classifiers <- read.csv(file = 'Weka_Data/Classifiers.csv', sep = ",", header = TRUE, fileEncoding = "UTF-8")
kable(classifiers, caption = "Classification using cross validation 10-fold") %>%
  kable_styling(latex_options = 'scale_down', position = 'center')
```

The table 3 shows the results of the 8 classifiers from the machine learning algorithm using weka. The results are shows speed, accuracy, True Positive (TP), False Positive(FP), True Negative(TN), and False Negative(FN). The speed is shown in seconds, the accuracy is shown in percentages. The last four of the table 3 are part of the confusion matrix in Weka. Because the class that was chosen has 8 classes the classes are categorized in unknown and all the lethal causes.

Looking at the table 3 you could see that there are a few who has a higher accuracy compared to the others. OneR has 88.6% accuracy, RandomForest has 86.1% accuracy and SimpleLogistic has 86.2% accuracy. Because the data is about diseases the RandomForest and SimpleLogistic is a good algorithm to use even though the OneR scored better in accuracy.

With the information of the classifiers we are going to investigate the effect of different algorithms using the Weka experimenter. The ZeroR is the test base and the other algorithms are compared to ZeroR to show the accuracy.

```
knitr::include_graphics("images/experiment_output.png")
```

Dataset	(1) rules.ZeroR	(2) rules	(3) bayes	(4) funct	(5) funct	(6) lazy.	(7) trees	(8) trees
Myocardial	(100)	84.06	88.57 v	74.92 *	86.26 v	85.77 v	80.84 *	85.55 v
		(v/ /*)	(1/0/0)	(0/0/1)	(1/0/0)	(1/0/0)	(0/0/1)	(1/0/0)
Key:								
(1) rules.ZeroR '' 48055541465867954								
(2) rules.OneR '-B 6' -3459427003147861443								
(3) bayes.NaiveBayes '' 5995231201785697655								
(4) functions.SimpleLogistic '-I 0 -M 500 -H 50 -W 0.0' 7397710626304705059								
(5) functions.SMO '-C 1.0 -L 0.001 -P 1.0E-12 -N 0 -V -1 -W 1 -K \"functions.supportVector.PolyKernel -E 1.0 -C 250007\" -calibrator \"functions.Logistic								
(6) lazy.IBk '-K 1 -W 0 -A \"weka.core.neighboursearch.LinearNNSearch -A \"weka.core.EuclideanDistance -R first-last\"\"\" -3080186098777067172								
(7) trees.J48 '-C 0.25 -M 2' -217733168393644444								
(8) trees.RandomForest '-P 100 -I 100 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1' 1116839470751428698								

Figure 7: Accuracy of the algorithms for myocardial infraction. v shows that it is significant higher than the test base, ZeroR and * shows that it is significant lower than ZeroR

The image 7 shows the output of the experiment of the same algorithms that are shown in table 3, but with ten repetitions. And the same three algorithms offer the best accuracy compared to ZeroR. But if you compare to the table 3 it shows that the J48 does it better on the image 7 with five differences. And NaiveBayes does slightly better with 0.3 difference, SMO with 0.2 difference and IBk also with a 0.2 difference if compared to the table 3. Expect for RandomForest it does slightly worse with a 0.1 difference.

The two algorithm that are chosen are RandomForest and SimpleLogistic. The reason for this is that they have a good accuracy and for this data set a tree classifiers are a good algorithm to make a decision as if the patient is at risk. And SimpleLogistic is a good classifier to build a linear logistic regression model, so with this model it can help make a best fitting logistic model to use for deciding which lethal cause a patient could have.

The two algorithms, RandomForest and SimpleLogistic, the cost sensitive of the Myocardial Infraction is made with the settings that is to seen in the image under this.

```
knitr::include_graphics("images/costsensitive_setting.png")
```

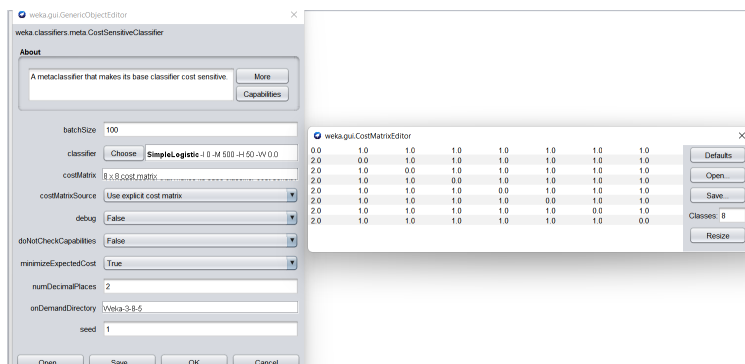


Figure 8: The settings of the CostSensitiveClassifier and of the CostMatrix.

The image 8 shows the settings of the CostSensitiveClassifier and the cost matrix, in which the FP is increased. The minimizeExpectedCost is set to true so that the cost is minimized of the expected misclassification. In the CostMatrixEditor, the FP is adapted to 2.0 to reduce the FP in the cost matrix.

With the settings of the given image 8 the results of the two algorithms is saved.

Table 4: CostSensitiveClassifier with the two algorithms

Classifier	Accuracy	costMatrix	TP	FP	FN	TN
SimpleLogistic	86.2	1	1424	223	5	48
SimpleLogistic	86.2	2	1420	216	9	55
SimpleLogistic	85.6	3	1407	211	19	60
SimpleLogistic	82.4	4	1349	179	80	92
RandomForest	86.1	1	1428	236	1	35
RandomForest	86.5	2	1421	203	8	68
RandomForest	84.6	3	1358	129	71	142
RandomForest	80.3	4	1270	88	159	183

```
costsensitive <- read.csv("Weka_Data/CostSensitive.csv", sep = ";",
                        header = TRUE, fileEncoding="UTF-8-BOM")
kable(costsensitive, caption = "CostSensitiveClassifier with the two algorithms" ) %>%
  kable_styling(latex_options = 'scale_down', position = 'center')
```

The table 4 shows the CostSensitiveClassifier for the two algorithms that work the best for the Myocardial Infraction data set, SimpleLogistic and RandomForest. The costMatrix is adapted from 1.0 to 4.0 for each build of the CostSensitiveClassifier. The RandomForest has for each row a different confusion matrix and accuracy. In contrast, SimpleLogistic has the same accuracy for the first two rows but, like the RandomForest different confusion matrix. The costMatrix cost of the FP is increased in this research. The reason for this is we don't want people who actually have heart diseases to be predicted with the unknown, as in other diseases or doesn't have any diseases. In the confusion matrix, the FN is increased so that that FP decreases. The results that we want to keep with this data set are the lethal causes, the TN, and the true positives, which is the unknown. This could be patients that didn't have any lethal reason in the aspect of heart diseases. And the results of the patients are not related to heart diseases. But it can also be that those patients need more research.

In table 4 it shows that the RandomForest with the cost adapted to 2.0 the accuracy increases by approximately 0.4, but when the cost is increased to 3.0 and 4.0, the accuracy decreases. But with the algorithm SimpleLogistic, the accuracy is the same for the cost with 1.0 and 2.0, but like the RandomForest, the accuracy decreases.

5 Exploring Meta-learners and optimize a selection of algorithms

Here is the effect of the attribute selection methods investigated and optimized and exploring the meta learners with the Myocardial infraction data set.

5.1 Attribute selection

For the attribute selection it has to be evaluated, each single attribute is evaluated with four evaluators. Which will delete the attribute that doesn't say much about the classes.

The settings for the AttributeSelectedClassifier is to see in the image under this.

```
knitr::include_graphics("images/attribute_settings.png")
```

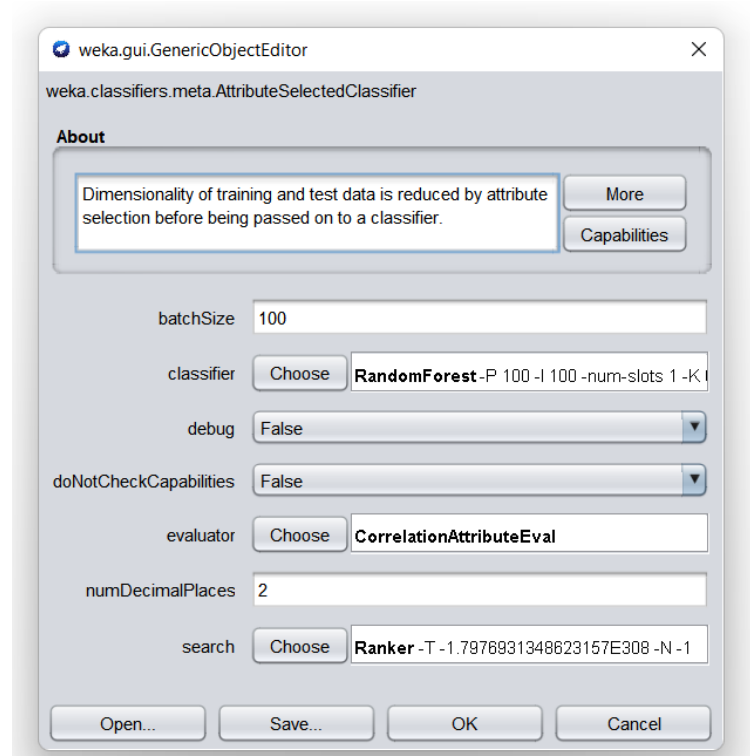


Figure 9: The settings of the AttributeSelectedClassifier

With the settings that shows in image 9 has been done with four evaluators. The evaluators are *CorrelationAttributeEval*, *GainRatioAttributeEval*, *InfoGainAttributeEval*, and *OneRAttributeEval*. The method search is set on Ranker for all four evaluators and the two algorithms.

```
# Reads the png files to create images
correlation <- readPNG("images/Correlation.png")
gain_ratio <- readPNG("images/GainRatio.png")
info_ratio <- readPNG("images/InfoGain.png")
oner <- readPNG("images/OneR.png")

# Plots each image and combines it into one
```


Table 5: Evaluator CfsSubsetEval with the searches RankSearch and BestFirst to evaluate the attributes

Classifier	Evaluator	Search	Accuracy	TP	FP	FN	TN
RandomForest	CfsSubsetEval	RankSearch	86.1	1425	230	4	41
RandomForest	CfsSubsetEval	BestFirst	86.4	1426	227	3	44
SimpleLogistic	CfsSubsetEval	RankSearch	86.1	1422	224	7	47
SimpleLogistic	CfsSubsetEval	BestFirst	86.3	1424	222	5	49

```
knitr::include_graphics("images/subset_settings.png")
```

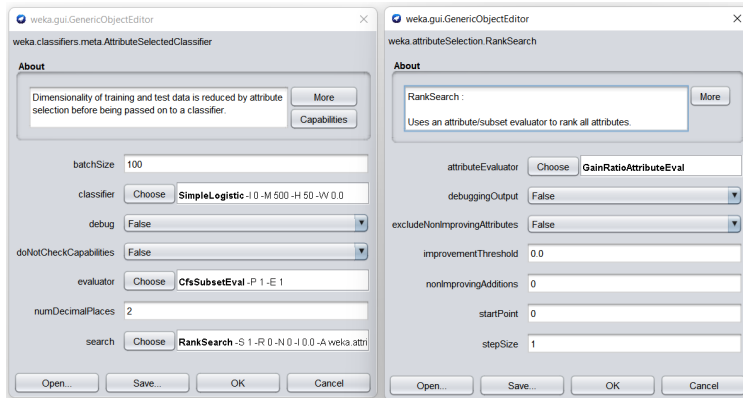


Figure 11: The settings of the AttributeSelection with the evaluator CfsSubsetEval with the settings of the RankSearch

Image 11 shows the settings of the evaluator and search. The other settings are the standard settings for the AttributeSelectedClassifier.

The GainRatio is chosen for the RankSearch is because the GainRatio evaluates the worth of the data set attribute by measuring the gain ratio.

```
subset <- read.csv(file = "Weka_Data/subseteval.csv", sep = ";", header = TRUE, row.names = NULL)
kable(subset, caption = "Evaluator CfsSubsetEval with the searches RankSearch and BestFirst to evaluate the attributes")
kable_styling(latex_options = 'scale_down', position = 'center')
```

The search BestFirst with both the classifiers shows an increase of the accuracy. But with the search RankSearch, shows for both Classifiers the same accuracy. The results of the CfsSubsetEval show that it selected the same attributes for the classifiers RandomForest and SimpleLogistic. After looking at attributes, it has been decided to choose the attributes to create a better algorithm. To verify the selected attributes, the Select attributes in Weka has been used with the evaluator CfsSubsetEval and with the search Method RankSearch with cross-validation 10-folds. The same chosen attributes have been selected, and the attribute SEX. The rest of the attributes have been deleted, and the file has been saved. With the new myocardial infarction data set the further investigation will take place.

5.2 Meta learners

The meta learners are stacking, boosting and bagging with these were are going to investigate the myocardial infarction data set.

Table 6: Meta Leaners with the cross validation 10-fold and with 10 repetitions

Algorithm	Classifiers	MetaClassifier	CombinationRule	Accuracy	TP	FP	FN	TN
Bagging	RandomForest	N.A	N.A	85.8235	1428	240	1	31
Bagging	SimpleLogistic	N.A	N.A	84.9412	1395	186	34	85
Boosting	RandomForest	N.A	N.A	86.3529	1425	218	4	53
Boosting	SimpleLogistic	N.A	N.A	86.2353	1424	223	5	48
Stacking	RandomForest & SimpleLogistic	RandomForest	N.A	86.2941	1418	210	11	61
Stacking	SimpleLogistic & RandomForest	SimpleLogistic	N.A	86.6471	1428	221	1	50
Vote	RandomForest & SimpleLogistic	N.A	Average of Probabilities	86.2353	1425	228	4	43
Vote	RandomForest & SimpleLogistic	N.A	Majority Voting	86.2353	1425	228	4	43

First the Paired T-Tester and the paired T-Tester (corrected) has been runned on the data set and both show the same effect in the experimenter for the algorithms RandomForest and SimpleLogistic. For both statistical tests has RandomForest an accuracy of 85,98%, whereas SimpleLogistic has an accuracy of 86.26%. The RandomForest has an lower accuracy than in table 3 while SimpleLogistic has like in the table 3 an accuracy of 86.26%.

```
meta_learners <- read.csv("Weka_Data/meta_learners.csv", sep = ";", header = TRUE,
                          fileEncoding="UTF-8-BOM")
```

```
kable(meta_learners, caption = "Meta Leaners with the cross validation 10-fold and with 10 repetitions")
```

The table 6 shows the outcome of the meta learners with the cross-validation 10-fold. The algorithm Bagging with the classifiers shows that the accuracy for both is lower than the accuracy in table 3, which could mean that the Bagging is not a good algorithm for this data set. For the rest of the algorithm, the accuracy is higher or the same as in the table 3. With the Bagging Algorithm, there is to see that for the classifier SimpleLogistic TN is higher than in the table 3, this means that there are more patients that have heart disease. The algorithm Boosting does it better with the classifier RandomForest, the accuracy of the RandomForest is higher than in the table 3, but for the classifiers SimpleLogistic, the algorithm Boosting does not have any effect on the accuracy, TP, FP, FN, and TN.

The Stacking algorithm does it better for both MetaClassifier RandomForest and SimpleLogistic. The accuracy is higher than in the table 3 and the TN has increased too. We want to know in this data set if a person has a probability of having heart disease. This means that the TN increase could be a good result and because the TP hasn't changed or not much, it could mean that the Stacking is a good algorithm for the data set. Table 3 shows that the accuracy for the algorithm Vote for both CombinationRule are the same. So the algorithm Vote does not show any improvement for the data set.

6 ROC and learning curve analysis

In this the the Receiver Operating Characteristics (ROC) curve is being visualized of the algorithm with the optimal settings. With the results the learning curve is made to get a a good performance estimate for the Myocardial Infraction data set.

6.1 ROC curve analysis

For both algorithms, RandomForest and SimpleLogistic, the ROC curves are visualized. The ROC curve is a graphical plot that illustrates the performance of the classification model as its threshold varies.

The following section visualizes Receiver Operating Characteristics (ROC) curves for both RandomForest and SimpleLogistic ROC curve is a graph showing the performance of a classification model at all threshold settings. The ROC curve is created by plotting the True Positive Rate (TPR) against the False Positive Rate (FPR), where TPR is on the y-axis and FDR is on the x-axis.

ROC curve for the classifier RandomForest and SimpleLogistic with the curve of all the threshold setting with the class attribute cardiogenic shock, because the cost matrix shows that the true negatives are mostly the cardiogenic shock.

In Weka the ROC curve of RandomForest and SimpleLogistic are visualized.

```
# Reads the png files to create images
roc_rf <- readPNG("images/roc_rf.png")
roc_sl <- readPNG("images/roc_sl.png")

# Plots each image and combines it into one
rf <- ggplot() +
  annotation_custom(rasterGrob(roc_rf))
sl <- ggplot() +
  annotation_custom(rasterGrob(roc_sl))

# Combines the images
plot_grid(rf, sl, labels = c("RandomForest", "SimpleLogistic"))
```

Figures 12 shows both visualizations of the threshold setting of the attribute cardiogenic shock.

The data is plotted in a figure with the ROC curve data of the classifiers RandomForst and SimpleLogistic.

```
# Read the roc curve data of RandomForest and SimpleLogistic
roc_data_randomforest <- read.table("Weka_Data/roccurve_randomforest.arff",
                                     sep = ",", comment.char = "@")
roc_data_simplelogistic <- read.table("Weka_Data/roccurve_simplelogistic.arff",
                                       sep = ",", comment.char = "@")

# Defining the names the roc data for RandomForest
names(roc_data_randomforest) <- c("Instance_number", "True_Positives", "False_Negatives",
                                   "False_Positives", "True_Negatives",
                                   "False_Positive_Rate", "True_Positive_Rate",
                                   "Precision", "Recall", "Fallout", "FMeasure",
                                   "Sample_Size", "Lift", "Threshold")

# Assign colors for the classifier and threshold
```

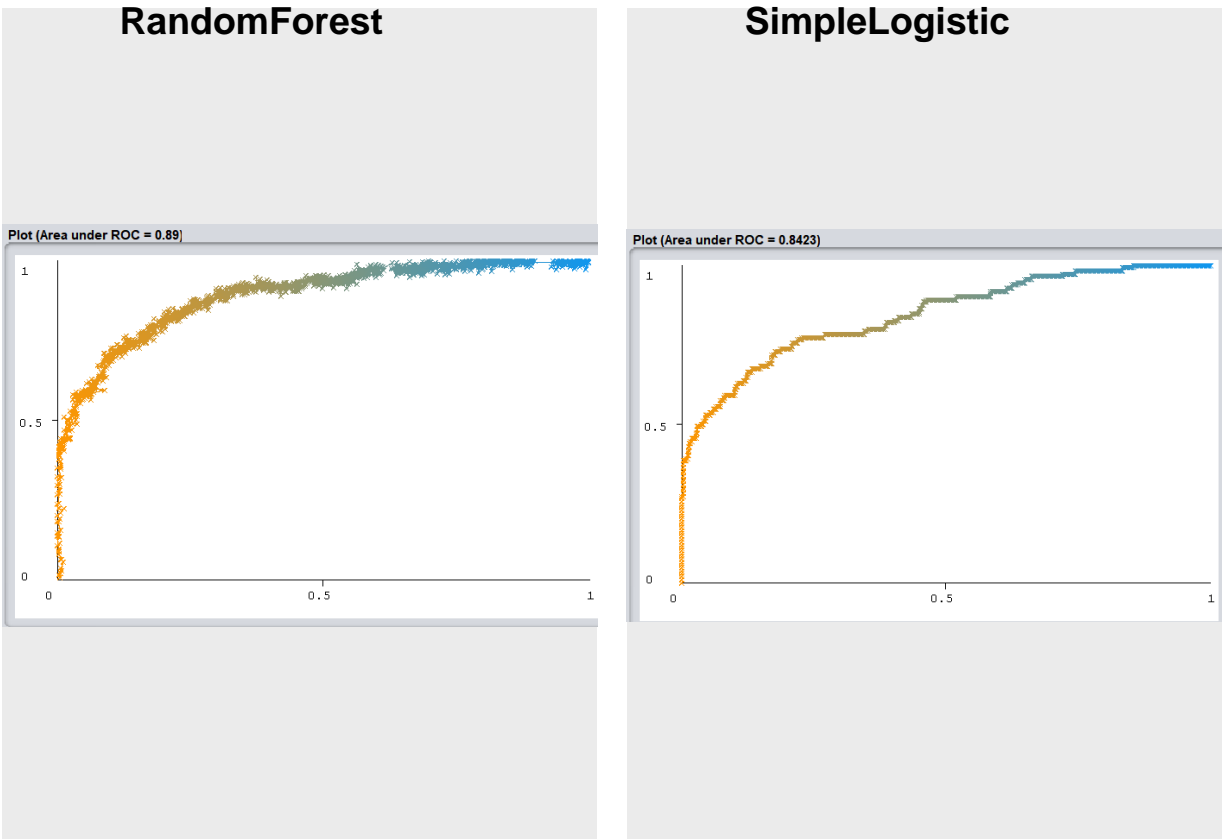



Figure 12: The outcome of the ROC curve threshold visualizer for both the classifiers RandomForest and SimpleLogistic

```

colors <- c(classifier = "orange", threshold = "blue")

# Plot the ROC Curve of RandomForest
roc_randomforest <- ggplot(data = roc_data_randomforest, mapping = aes(x = False_Positive_Rate,
                                                                    y = True_Positive_Rate)) +
  geom_point(mapping = aes(color = "classifier")) +
  geom_line(aes(color = "classifier")) +
  geom_abline(aes(color = "threshold", slope = 1, intercept = 0)) +
  scale_color_manual(values = colors) +
  ggtitle("RandomForest") +
  xlab("False Positive Rate") +
  ylab("True Positive Rate") +
  theme_pubr() +
  theme(legend.title = element_blank())

# Define the names for SimpleLogistic
names(roc_data_simplelogistic) <- names(roc_data_randomforest)

# Plot the ROC Curve of SimpleLogistic
roc_simplelogistic <- ggplot(data = roc_data_simplelogistic, mapping = aes(x = False_Positive_Rate,
                                                                    y = True_Positive_Rate)) +
  geom_point(mapping = aes(color = "classifier")) +
  geom_line(aes(color = "classifier")) +
  geom_abline(aes(color = "threshold", slope = 1, intercept = 0)) +
  scale_color_manual(values = colors) +
  ggtitle("SimpleLogistic") +
  xlab("False Positive Rate") +
  ylab("True Positive Rate") +
  theme_pubr() +
  theme(legend.title = element_blank())

# Combine ROC Curve of RandomForest and SimpleLogistic plots
combined_roccurve <- plot_grid(roc_randomforest + theme(legend.position = "none"),
                              roc_simplelogistic + theme(legend.position = "none"),
                              labels = c("A", "B"), label_size = 12)

# Create legend
legend <- get_legend(roc_simplelogistic + guides(color = guide_legend(nrow = 1)))

# Plot the combined plots
plot_grid(combined_roccurve, legend, ncol = 1)

```

In figure 13 there are two figures with a ROC curve. The first one figure 13A shows the ROC curve from the algorithm RandomForest and the second figure 13B shows the ROC curve from the algorithm SimpleLogistic. The TPR is plotted against the FPR at various thresholds, forming a line when the dots joins. The area under the ROC curve shown in orange is called the Area Under The Curve (AUC) curve. The blue threshold line shown in the figures is straight, the predicted observation probability. Figure 13A line classifier has dots that are connected, and between the dots at the end, the dots are connected. The line has almost a line of 90 degrees and a straight angle. This shows that the curve is ideal because the two curves of TP and TN almost do not overlap. The reason for it is that it is perfectly able to distinguish between positive class and negative class. If the two curves overlap, the type 1 and type 2 errors are introduced, and the model runs in a curve or a straight line like the threshold line. The ROC curve of SimpleLogistic is shown in figure 13B, and the TPR is plotted against the FPR. Both threshold lines of the ROC curve RandomForest and SimpleLogistic are the same. The most significant difference between the two figures is that the ROC curve of

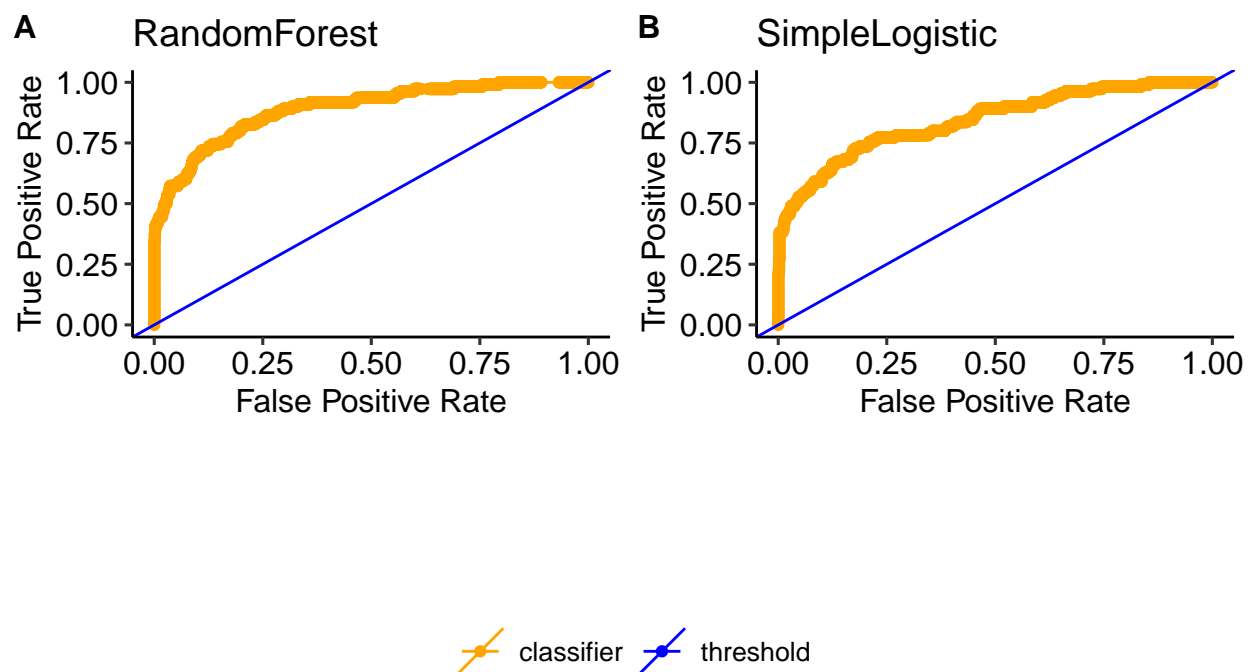


Figure 13: ROC curve for the classifiers RandomForest and SimpleLogistic at all threshold setting with the attribute cardiogenic shock

the RandomForest line compared to the ROC curve of SimpleLogistic is slightly straighter. This means that the ROC curve RandomForest figure 13A has almost an ideal situation because there is virtually no overlap. This the ROC curve of RandomForest is slightly better than SimpleLogistic. In Weka, the SimpleLogistic has the AUC value of 0.842, and the AUC value of RandomForest is 0.890. This will be indicated that the ROC curve of RandomForest is better than SimpleLogistic because the AUC of RandomForest is closer to one which categories as perfect.

7 Learning curve

The RandomForest came out as the best with the standard settings from all the algorithms, so for the next section, a learning curve is created with RandomForest as the classifier.

```
knitr::include_graphics("Images/learning_curve.png")
```

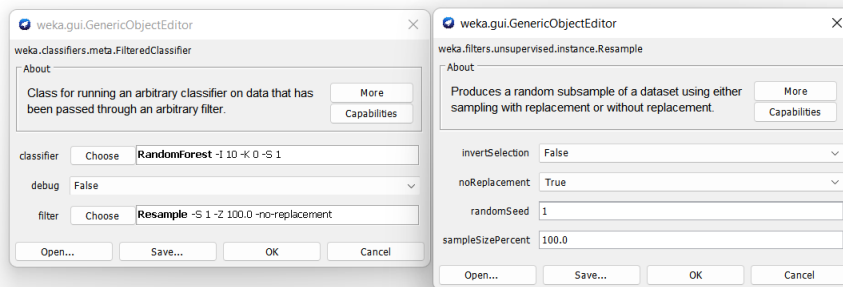


Figure 14: The settings of classifier FilteredClassifier to create a learning curve.

Image 14 shows the settings of the FilteredClassifier. The classifier is FilteredClassifier under meta. The chosen classifier in the FilteredClassifier is RandomForest and as filter Resample is chosen. The noReplacement is set to true so there are no replacements and the sampleSizePercent differs from 0% to 100% as seen in the right side of the image 14.

```
learning_data <- read.csv("Weka_Data/learning_curve.csv", header = TRUE,
                          sep = ",")
kable(learning_data, caption = "Accuracy of the FilteredClassifier with
different sampleSizePercent") %>%
  kable_styling(latex_options = "HOLD_position")
```

Table 7: Accuracy of the FilteredClassifier with different sampleSizePercent

sampleSizePercent	Accuracy
1	84.1
2	84.2
5	86.1
10	85.4
20	87.8
30	89.4
40	91.1
50	91.8
60	92.8
70	94.5
80	96.1
90	97.3
100	98.5

```
# Plotting the learning curve
ggplot(learning_data, aes(x = sampleSizePercent, y = Accuracy)) +
  geom_point(colour = "darkgreen") +
  geom_line(colour = "darkgreen") +
  geom_hline(yintercept = 65, linetype = "dashed", color = "darkblue") +
  theme_minimal() +
  labs(title="Learning curve", x="Sample Size Percent (%)", y="Accuracy (%)") +
  scale_x_continuous(breaks = seq(0, 100, 10)) +
  scale_y_continuous(breaks = seq(50, 100, 5))
```

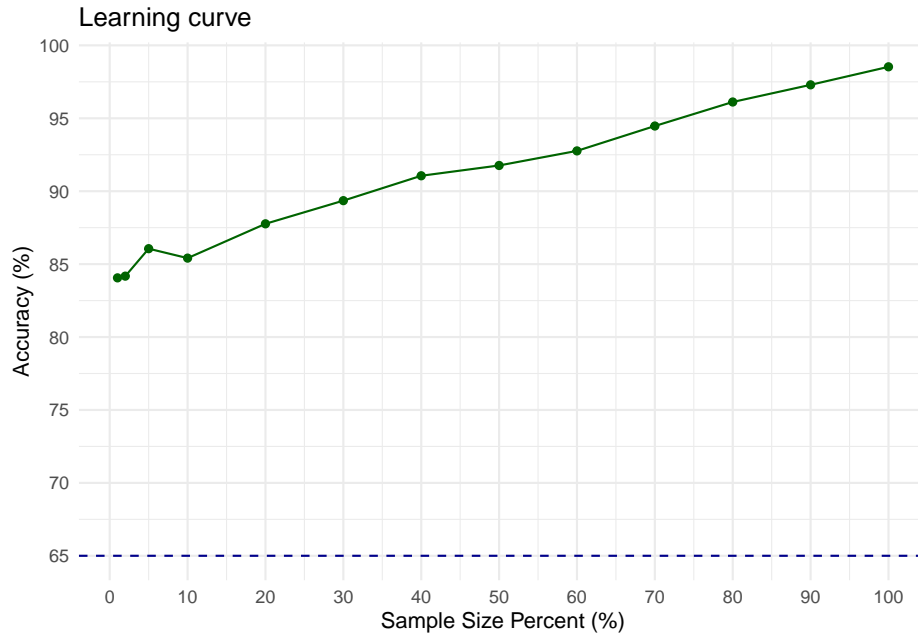


Figure 15: Learning curve of the accuracy for different sample size percentages compared to the baseline zeroR.

The table 7 contains two columns with the sample size percent and the second column gives the accuracy for

different sample size percentages from the classifier FilteredClassifier. Here can be seen that the accuracy of 1% to 100% is it increasing. However from 1% to 5% sample size percent the accuracy was increasing, but when it hits the 10% sample size percent the accuracy decreases with 1%. Afterwards it was increasing till it hits the mark of 100% sample size percent, so the highest is accuracy is for 100% of the sample size which is 98.5%. These values create a learning curve together which is shown in figure 15. Figure 15 has a green line with points which shows the accuracy for the different sample size percentages of RandomForest and the dashed blue line shows the baseline which is the accuracy of ZeroR for every percentage. The accuracy of ZeroR is the same because is the simplest classification method that relies only on predicting the class and ignores all attributes. And ZeroR has thus for all percentages the same accuracy. Compared to the baseline of ZeroR the RandomForest has a much higher accuracy. RandomForest shows a smooth linear increase. To get a reasonable performance there is at least 80% of the data needed, because from 80% it is above 95% accuracy and below 80% of the sample size it is lower than 95% accuracy. But for 100% of the sample size has RandomForest an accuracy of 98.5%. So there is at least 80% of the data needed to get a reasonable performance and to get a better performance 100% of the data is needed.

References

- [1] Johns Hopkins medicine: *Heart attack*, Conditions and Diseases, Retrieved from <https://www.hopkinsmedicine.org/health/conditions-and-diseases/heart-attack> on 4-10-2021
- [2] Machine learning repository: *Machine Learning*, Myocardial infraction complications data set, Retrieved from <https://archive.ics.uci.edu/ml/datasets/Myocardial+infarction+complications> on 09-12-2020