# Predicting Survival Among Individuals with Heart Failure

Final project in STAT-240 Multivariate Data Analysis

Nicole Frontero

Due: November 30, 2020

## Introduction

Heart failure is a cardiovascular disease (CVD) that happens when the heart cannot pump enough blood and oxygen throughout the body (CDC, 2020). In the United States alone, about 6.2 million adults have heart failure. When considered on a worldwide scale, heart failure is referred to as a "global pandemic" (Savarese & Lund, 2017). While scientists and doctors are aware of risk factors for developing heart failure (CDC, 2020), it is difficult at present for doctors to predict the survival of patients who already have heart failure (Chicco & Jurman, 2020). In their paper titled "Machine learning can predict survival of patients with heart failure from serum creatinine and ejection fraction alone," researchers Davide Chicco and Giuseppe Jurman set out to predict the survival of patients with heart failure by performing machine learning techniques on medical records data.

I plan to extend the work done by Chicco and Jurman by using **clustering** techniques to see if I can identify groups of individuals among those with heart failure. If I were able to identify groups of individuals, then I could make models for predicting survival that are specific to each group. In order to predict survival and learn what the most important variables are in this prediction process, I will use **classification** techniques. I will perform classification regardless of my ability to successfully identify groups among the patients.

Being able to answer the question "Does this individual belong to a particular subgroup of patients, and does that help us predict their survival?" has lasting implications for the field of cardiovascular health. Understanding individual patients' risk factors will allow physicians to provide personalized medicine that can hopefully increase the chances of a patient's survival.

# Preliminary Analysis

## Data

The data set that I will be performing my analysis on comes from a paper written by Ahmad et al. (2017). Ahmad and his colleagues obtained medical records of 299 patients who had class III or class IV heart failures (per the New York Heart Association, or NYHA, classification scheme) (Chicco & Jurman, 2020). All records were collected from the Faisalabad Institute of Cardiology and the Allied Hospital in Faisalabad (Punjab, Pakistan) during the months of April-December 2015 (Chicco & Jurman, 2020). Note that Chicco and Jurman analyzed Ahman et al.'s data via a website online where the data has been made publicly available (Ahman et al., 2017). I first identified the data, and downloaded it, from Kaggle (Kaggle, n.d.).

Since I am interested in predicting survival of patients with heart failure, I will utilize death as my outcome variable of interest throughout my analyses. All variables included in the data set are described below.

*Categorical variables*

- **Death:** categorical (0, 1); indicates death during the follow-up period.
  - This is the outcome variable
- **Anaemia:** categorical (0, 1); indicates presence of anaemia.
- **Diabetes:** categorical (0, 1); indicates presence of diabetes.
- **High blood pressure:** categorical (0, 1); indicates presence of high blood pressure.
- **Sex:** categorical (0, 1); indicates sex.
- **Smoking:** categorical (0, 1); indicates smoking status.

*Quantitative variables*

- **Time:** numerical; follow-up period (days), defined as days in between a patient's most recent visit and their next follow up visit.
- **Age:** numerical; age of subject.
- **Creatinine phosphokinase:** numerical; level of CPK enzyme in blood (mcg/L).
- **Ejection fraction:** numerical; percentage of blood leaving heart with each contraction.
- **Platelets:** numerical; concentration of platelets in the blood (kiloplatelets/mL).
- **Serum creatinine:** numerical; level of creatinine in the blood (mg/dL).
- **Serum sodium:** numerical; level of sodium in the blood (mEq/L).

I will use the 299 observations in this data set to try to learn more about the entire population of interest, which in the context of this report, is all adults in the world.

## Data wrangling

### Read the data in

```
# Reading in the data
hearts <- read_csv("FronteroNData.csv")
```

### Rename variables

I am choosing to rename the variables in the original data set so that I can adhere to standard variable naming conventions.

```r
# Renaming variables
hearts <- hearts %>% rename(time = TIME,
                            death = Event,
                            sex = Gender,
                            smoking = Smoking,
                            diabetes = Diabetes,
                            blood_pressure = BP,
                            anaemia = Anaemia,
                            age = Age,
                            ejection_fraction = Ejection.Fraction,
                            sodium = Sodium,
                            creatinine = Creatinine,
                            platelets = Pletelets
                            )
```

### Making categorical variables into factors

The original data has categorical variables stored as numerics, which is not the ideal format to have categorical variable stored as. I will make the categorical variables into factors with levels in order to fix this problem.

```r
# Making categorical variables into factors

# Sex (note that in the original data men are 1, women are 0)
hearts$sex <- factor(hearts$sex)
levels(hearts$sex) <- c("female", "male")

# Diabetes (note that in the original data, 0 is false 1 is true)
hearts$diabetes <- factor(hearts$diabetes)
levels(hearts$diabetes) <- c("false", "true")

# High blood pressure (note that in the original data, 0 is false 1 is true)
hearts$blood_pressure <- factor(hearts$blood_pressure)
levels(hearts$blood_pressure) <- c("false", "true")

# Smoking (note that in the original data, 0 is false 1 is true)
hearts$smoking <- factor(hearts$smoking)
levels(hearts$smoking) <- c("false", "true")

# Death (note that in the original data, 0 is false 1 is true)
hearts$death <- factor(hearts$death)
levels(hearts$death) <- c("false", "true")

# Anaemia (note that in the original data, 0 is false 1 is true)
hearts$anaemia <- factor(hearts$anaemia)
levels(hearts$anaemia) <- c("false", "true")
```

### Rounding the age variable:

The original data has three 0s after the decimal place for all of the age data, but these 0s are not necessary. Consequently, I am going to get rid of them by rounding.

```r
# Getting rid of the extra three 0s after the decimal place for the age data
hearts$age <- round(hearts$age, 0)
```

## Univariate analysis

Before performing any classification or clustering methods on the data set, it is important that I look into the univariate distribution of all of the variables.

### Categorical variables

All six of categorical variables each have two levels. I will create a table to summarize the counts and percentages for each level of each categorical variable. See Table 1 below.

```
# Calculating counts and percentages

# Sex
sex_count <- data.frame(tally(~ sex, data = hearts, format = "count"))
sex_percent <- data.frame(tally(~ sex, data = hearts, format = "percent"))
sex_df <- inner_join(sex_count, sex_percent, by = "sex")
colnames(sex_df) <- c("sex", "count", "percent")

# Anaemia
anaemia_count <- data.frame(tally(~ anaemia, data = hearts, format = "count"))
anaemia_percent <- data.frame(tally(~ anaemia, data = hearts, format = "percent"))
anaemia_df <- inner_join(anaemia_count, anaemia_percent, by = "anaemia")
colnames(anaemia_df) <- c("anaemia", "count", "percent")

# Diabetes
diabetes_count <- data.frame(tally(~ diabetes, data = hearts, format = "count"))
diabetes_percent <- data.frame(tally(~ diabetes, data = hearts, format = "percent"))
diabetes_df <- inner_join(diabetes_count, diabetes_percent, by = "diabetes")
colnames(diabetes_df) <- c("diabetes", "count", "percent")

# Blood pressure
blood_pressure_count <- data.frame(tally(~ blood_pressure, data = hearts,
                                         format = "count"))
blood_pressure_percent <- data.frame(tally(~ blood_pressure, data = hearts,
                                           format = "percent"))
blood_pressure_df <- inner_join(blood_pressure_count, blood_pressure_percent,
                                by = "blood_pressure")
colnames(blood_pressure_df) <- c("blood_pressure", "count", "percent")

# Smoking
smoking_count <- data.frame(tally(~ smoking, data = hearts, format = "count"))
smoking_percent <- data.frame(tally(~ smoking, data = hearts, format = "percent"))
smoking_df <- inner_join(smoking_count, smoking_percent, by = "smoking")
colnames(smoking_df) <- c("smoking", "count", "percent")

# Death
death_count <- data.frame(tally(~ death, data = hearts, format = "count"))
death_percent <- data.frame(tally(~ death, data = hearts, format = "percent"))
death_df <- inner_join(death_count, death_percent, by = "death")
colnames(death_df) <- c("death", "count", "percent")
```

```
# Adding a column to each df to note each respective variable

# Sex
sex_table <- cbind(c("sex"), sex_df)
colnames(sex_table) <- ""
```

```r
# Anaemia
anaemia_table <- cbind(c("anaemia"), anaemia_df)
colnames(anaemia_table) <- ""

# Diabetes
diabetes_table <- cbind(c("diabetes"), diabetes_df)
colnames(diabetes_table) <- ""

# Blood pressure
blood_pressure_table <- cbind(c("blood pressure"), blood_pressure_df)
colnames(blood_pressure_table) <- ""

# Smoking
smoking_table <- cbind(c("smoking"), smoking_df)
colnames(smoking_table) <- ""

# Death
death_table <- cbind(c("death"), death_df)
colnames(death_table) <- ""
```

```r
# Put all categorical univariate data into one df

# Row bind
categorical_univariate_df <- rbind(sex_table,
                                   anaemia_table,
                                   diabetes_table,
                                   blood_pressure_table,
                                   smoking_table,
                                   death_table)

# Set column names for table to display
colnames(categorical_univariate_df) <- c("Variable", "Level", "Count",
                                         "Percentage")

# Round the percentages to 2 decimal places
categorical_univariate_df$Percentage <- round(categorical_univariate_df$Percentage, 2)

# Convert to data frame
categorical_univariate_table <- data.frame(categorical_univariate_df)
```

```r
# Display using kable
kable(categorical_univariate_table, "latex", booktabs = TRUE) %>%
  kable_styling(position = "center") %>%
  add_header_above(
    c("Table 1: Univariate distributions of categorical variables" = 4)) %>%
  row_spec(2, hline_after = TRUE) %>%
  row_spec(4, hline_after = TRUE) %>%
  row_spec(6, hline_after = TRUE) %>%
  row_spec(8, hline_after = TRUE) %>%
  row_spec(10, hline_after = TRUE)
```

| Table 1: Univariate distributions of categorical variables | | | |
|---|---|---|---|
| Variable | Level | Count | Percentage |
| sex | female | 105 | 35.12 |
| sex | male | 194 | 64.88 |
| anaemia | false | 170 | 56.86 |
| anaemia | true | 129 | 43.14 |
| diabetes | false | 174 | 58.19 |
| diabetes | true | 125 | 41.81 |
| blood pressure | false | 194 | 64.88 |
| blood pressure | true | 105 | 35.12 |
| smoking | false | 203 | 67.89 |
| smoking | true | 96 | 32.11 |
| death | false | 203 | 67.89 |
| death | true | 96 | 32.11 |

The univariate distributions of the 6 categorical variables can be seen above. Of note, about $\frac{2}{3}$ of the patients were women, while $\frac{1}{3}$ were men. Approximately 43% of patients were anemic, while around 56% were not. About 41% of patients had diabetes, and approximately 58% of them did not. Around 35% had high blood pressure, while about 64% did not. Around 67% of the patients smoked, while around 33% did not.

**Distribution of outcome variable:**

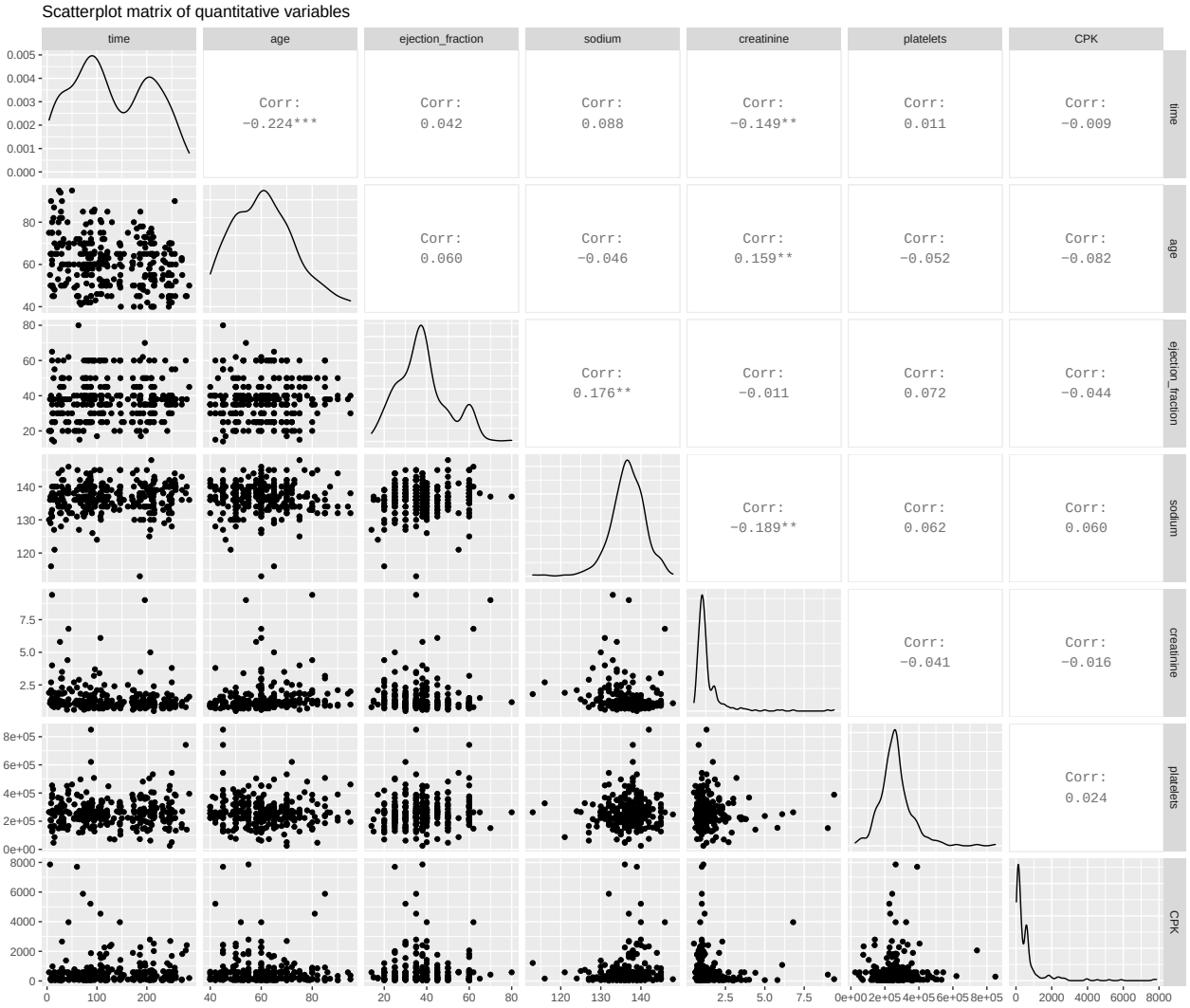
It is important that we examine the distribution of the outcome variable since I will be performing classification. In Table 1, we see that 68% of patients died (indicated by "true") and 32% survived (indicated by "false").

### Quantitative variables

To efficiently perform univariate analysis on the quantitative data, I will run `ggpairs` on the quantitative variables.

```
# Making a data set with just the quantitative variables
quantitative_df <- hearts %>%
  select(time, age, ejection_fraction, sodium, creatinine, platelets, CPK)
```

```
# Making the scatterplot matrix
GGally::ggpairs(quantitative_df,
                title = "Scatterplot matrix of quantitative variables")
```




The univariate distributions for each quantitative variable appear to be unimodal except for time, which exhibits a bimodal distribution. The distribution for time is bimodal. The distributions for age, ejection function, and sodium are all generally symmetric, although sodium is skewed somewhat left. The distribution of creatinine and CPK are both heavily skewed right, while the distribution of platelets is somewhat skewed right.

## Bivariate analysis

### Quantitative vs. quantitative

To examine the bivariate relationships between the quantitative variables, we can refer to the scatterplot matrix produced above. The pairwise correlations between the quantitative variables are all weak. The strongest pairwise correlation among all of the variables is the correlation between age and time ($r$ = -0.22).

We notice that all of the scatterplots featuring time as a plotted variable show roughly uniform vertical scatter. Similarly, we notice that all of the scatterplots featuring age as a plotted variable exhibit roughly uniform vertical scatter.

In the graphs plotting sodium and creatinine, sodium and platelets, and sodium and CPK, the observations seem to be tightly clustered together with some observations very far away for the cluster. The same is true for the scatterplots of creatinine and platelets, creatinine and CPK, and platelets and CPK. It may be worthwhile to look into outliers that may exist between the aforementioned bivariate variables.

After examining the bivariate relationships between each of the quantitative variables, there are a few key takeaways:

1) There is not much correlation between variables.
2) There may be some potential outliers, especially in the following bivariate relationships:
    - Sodium x Creatinine
    - Sodium x platelets
    - Sodium x CPK
    - Creatinine x platelets
    - Creatinine x CPK
    - Platelets x CPK

### Categorical vs. categorical

I am going to explore the bivariate relationships between each of the categorical variables and the outcome variable, death. To do so, I will make a table showing the bivariate distribution of each categorical variable with death.

```
# Bivariate distributions with percentages
sex_death_percent <- tally(death ~ sex, data = hearts, format = "percent")
smoking_death_percent <- tally(death ~ smoking, data = hearts, format = "percent")
diabetes_death_percent <- tally(death ~ diabetes, data = hearts, format = "percent")
blood_pressure_death_percent <- tally(death ~ blood_pressure, data = hearts,
                                      format = "percent")
anaemia_death_percent <- tally(death ~ anaemia, data = hearts, format = "percent")
```

```
# Column bind all of the dataframes together
bivariate_categorical_df <- cbind(sex_death_percent,
                                  smoking_death_percent,
                                  diabetes_death_percent,
                                  blood_pressure_death_percent,
                                  anaemia_death_percent)

bivariate_categorical_df <- round(bivariate_categorical_df, 2)
```

```
# Display using kable
kable(bivariate_categorical_df, "latex", booktabs = TRUE) %>%
  kable_styling(position = "center") %>%
  add_header_above(c("Death" = 1, "Sex" = 2, "Smoking" = 2, "Diabetes" = 2,
                     "Blood pressure" = 2, "Anaemia" = 2)) %>%
  add_header_above(
    c("Table 2: Bivariate distribution of categorical variables" = 11))
```

Table 2: Bivariate distribution of categorical variables

| Death | Sex | | Smoking | | Diabetes | | Blood pressure | | Anaemia | |
|---|---|---|---|---|---|---|---|---|---|---|
| | female | male | false | true | false | true | false | true | false | true |
| false | 67.62 | 68.04 | 67.49 | 68.75 | 67.82 | 68 | 70.62 | 62.86 | 70.59 | 64.34 |
| true | 32.38 | 31.96 | 32.51 | 31.25 | 32.18 | 32 | 29.38 | 37.14 | 29.41 | 35.66 |

Table 2 provides information on the bivariate distributions of all of the categorical variables with death. I notice that men and women exhibit similar death rates (around 32%), as do smokers and non-smokers (around 32%), and individuals with and without diabetes (around 32%). However, the death rate among individuals with high blood pressure (around 37%) is higher than the death rate among those without high

blood pressure (around 29%). Similarly, the death rate among those with anaemia (35%) is higher than the death rate among those without anaemia (29%).

**Categorical vs. quantitative:**

When I examine the bivariate relationship between the categorical variables and quantitative variables, there are a few key bivariate relationships I will look into:

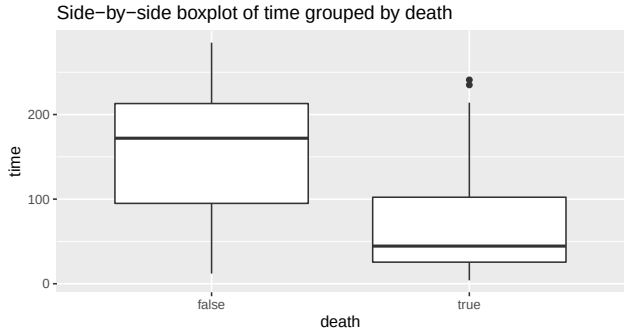
1) Death vs. every quantitative variable
    - This is important since death is the outcome variable
2) Anaemia vs. every quantitative variable
    - Anaemia vs. death exhibited an interesting bivariate relationship. Since we are ultimately trying to predict death, we may want to examine the relationship between anaemia and the quantitative variables.
3) Blood pressure vs. every quantitative variable
    - Blood pressure vs. death exhibited an interesting bivariate relationship. Since we are ultimately trying to predict death, we may want to examine the relationship between blood pressure and the quantitative variables.

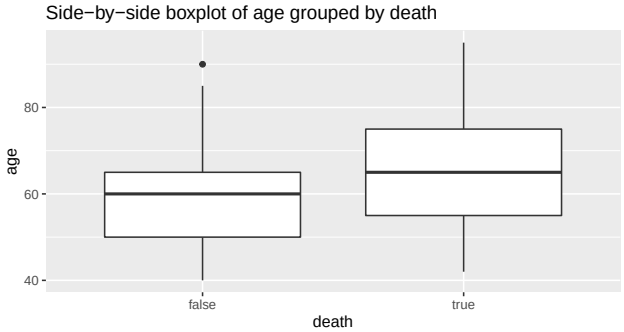
**Quantitative variables vs. death:**

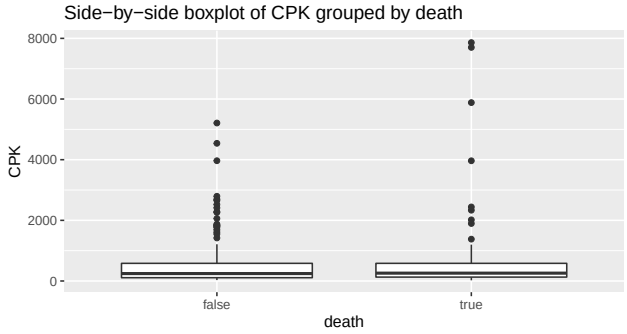
In an effort to explore the bivariate relationship between categorical and quantitative variables, I will first focus on looking at the bivariate distribution of quantitative variables grouped by death.

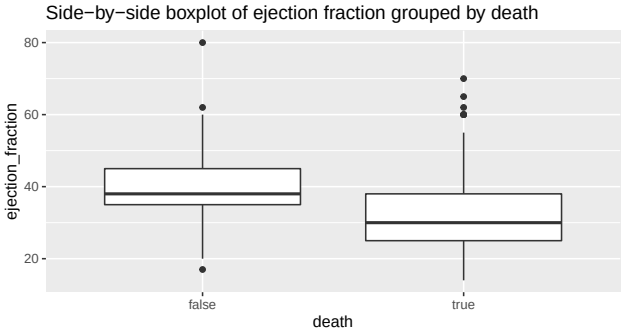
```
# Creating side-by-side boxplots for death x all quantitative variables

# time ~ death
time_death_boxplot <- gf_boxplot(time ~ death, data = hearts) +
  labs(title = "Side-by-side boxplot of time grouped by death")

# age ~ death
age_death_boxplot <- gf_boxplot(age ~ death, data = hearts) +
  labs(title = "Side-by-side boxplot of age grouped by death")

# CPK ~ death
CPK_death_boxplot <- gf_boxplot(CPK ~ death, data = hearts) +
  labs(title = "Side-by-side boxplot of CPK grouped by death")

# ejection_fraction ~ death
ejection_fraction_death_boxplot <- gf_boxplot(ejection_fraction ~ death,
                                              data = hearts) +
  labs(title = "Side-by-side boxplot of ejection fraction grouped by death")

# platelets ~ death
platelets_death_boxplot <- gf_boxplot(platelets ~ death, data = hearts) +
  labs(title = "Side-by-side boxplot of platelets grouped by death")

# creatinine ~ death
creatinine_death_boxplot <- gf_boxplot(creatinine ~ death, data = hearts) +
  labs(title = "Side-by-side boxplot of creatinine grouped by death")

# sodium ~ death
sodium_death_boxplot <- gf_boxplot(sodium ~ death, data = hearts) +
  labs(title = "Side-by-side boxplot of sodium grouped by death")
```

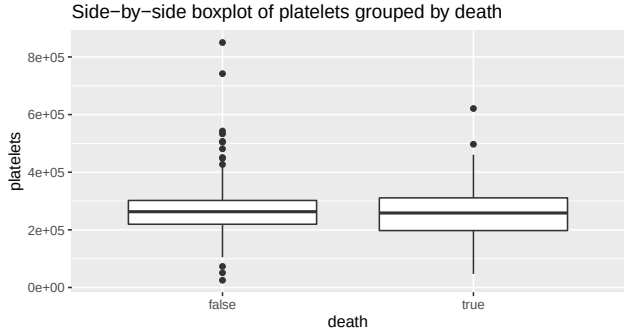
```
# Displaying the boxplots together using grid.arrange
grid.arrange(time_death_boxplot,
             age_death_boxplot,
             CPK_death_boxplot,
             ejection_fraction_death_boxplot,
             platelets_death_boxplot,
             creatinine_death_boxplot,
             sodium_death_boxplot,
             nrow = 4,
             ncol = 2)
```

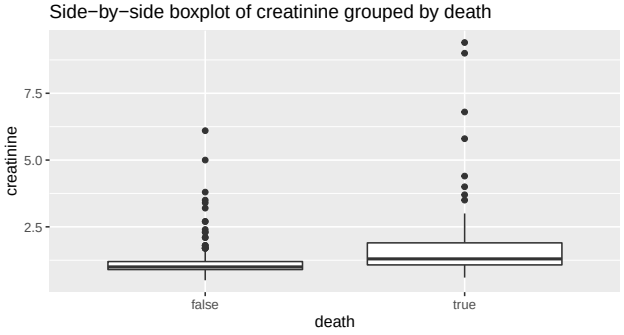


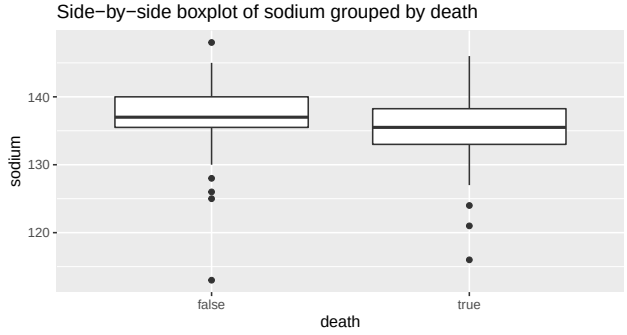



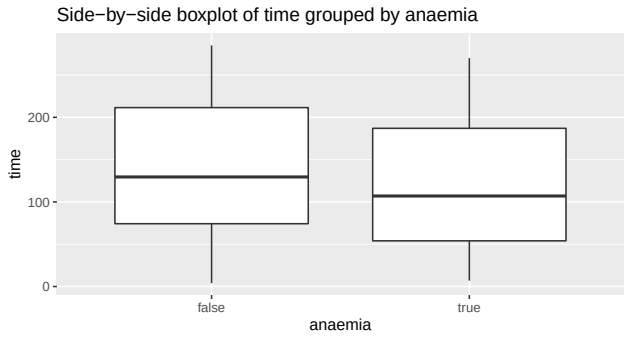
In the side-by-side boxplot of time grouped by death, we see that the median follow up time for those who died is a little less than 50 days, whereas the median follow up time among those who survived is approximately 160 days. Additionally, we notice that the side-by-side boxplot of age grouped by death shows that the median age of those who died is about 5 years greater than the median age of those who survived. We also see that the median ejection fraction of those who died is lower than the median ejection fraction of those who did not die. The side-by-side boxplots for CPK, platelets, creatinine, and sodium vs. death do not exhibit much difference between those who died and those who survived, at least upon visual inspection of the boxplots.

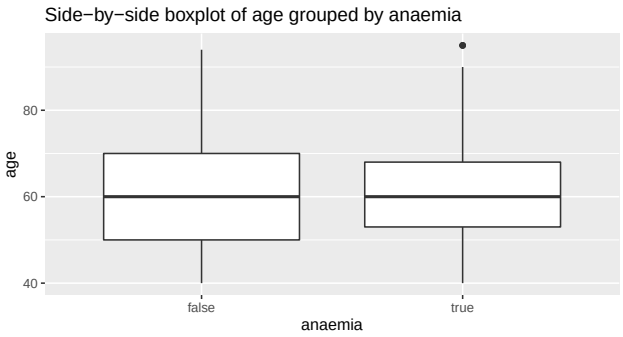
**Quantitative variables vs. anaemia:**

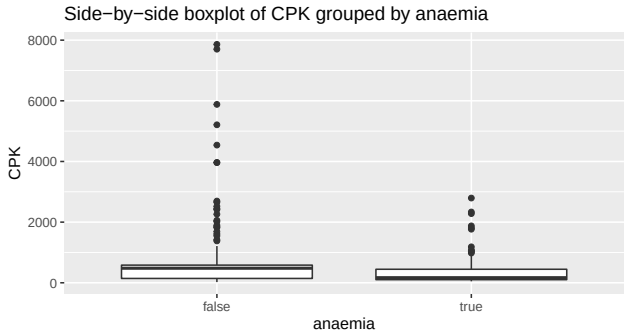
I also want to look at the bivariate relationships between anaemia and all of the quantitative variables, so I will make boxplots of all of those relationships as well.

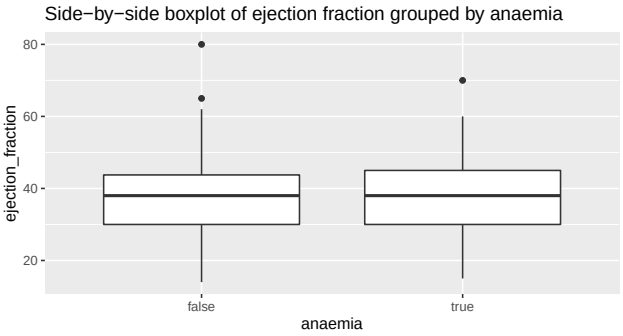
```
# Creating side-by-side boxplots for anaemia x all quantitative variables

# time ~ anaemia
time_anaemia_boxplot <- gf_boxplot(time ~ anaemia, data = hearts) +
  labs(title = "Side-by-side boxplot of time grouped by anaemia")

# age ~ anaemia
age_anaemia_boxplot <- gf_boxplot(age ~ anaemia, data = hearts) +
  labs(title = "Side-by-side boxplot of age grouped by anaemia")

# CPK ~ anaemia
CPK_anaemia_boxplot <- gf_boxplot(CPK ~ anaemia, data = hearts) +
  labs(title = "Side-by-side boxplot of CPK grouped by anaemia")

# ejection_fraction ~ anaemia
ejection_fraction_anaemia_boxplot <- gf_boxplot(ejection_fraction ~ anaemia,
                                                data = hearts) +
  labs(title = "Side-by-side boxplot of ejection fraction grouped by anaemia")

# platelets ~ anaemia
platelets_anaemia_boxplot <- gf_boxplot(platelets ~ anaemia, data = hearts) +
  labs(title = "Side-by-side boxplot of platelets grouped by anaemia")

# creatinine ~ anaemia
creatinine_anaemia_boxplot <- gf_boxplot(creatinine ~ anaemia, data = hearts) +
  labs(title = "Side-by-side boxplot of creatinine grouped by anaemia")

# sodium ~ anaemia
sodium_anaemia_boxplot <- gf_boxplot(sodium ~ anaemia, data = hearts) +
  labs(title = "Side-by-side boxplot of sodium grouped by anaemia")
```

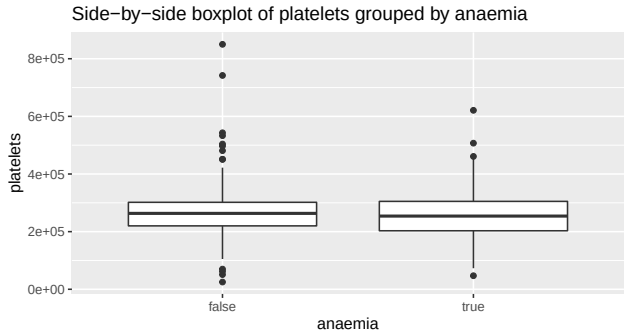
```
# Displaying the boxplots together using grid.arrange
grid.arrange(time_anaemia_boxplot,
             age_anaemia_boxplot,
             CPK_anaemia_boxplot,
             ejection_fraction_anaemia_boxplot,
             platelets_anaemia_boxplot,
             creatinine_anaemia_boxplot,
             sodium_anaemia_boxplot,
             nrow = 4,
             ncol = 2)
```

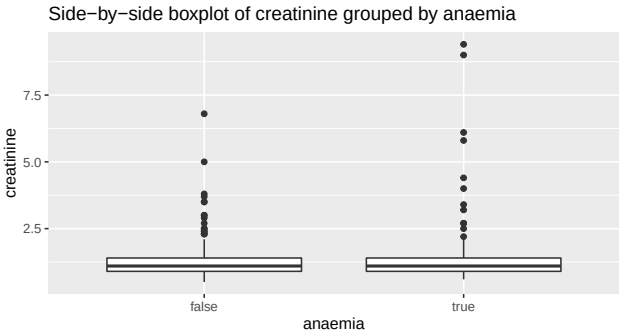


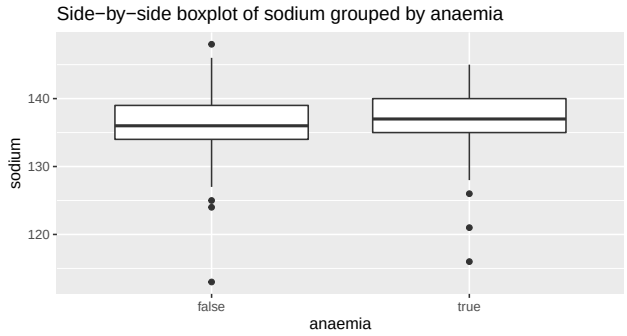



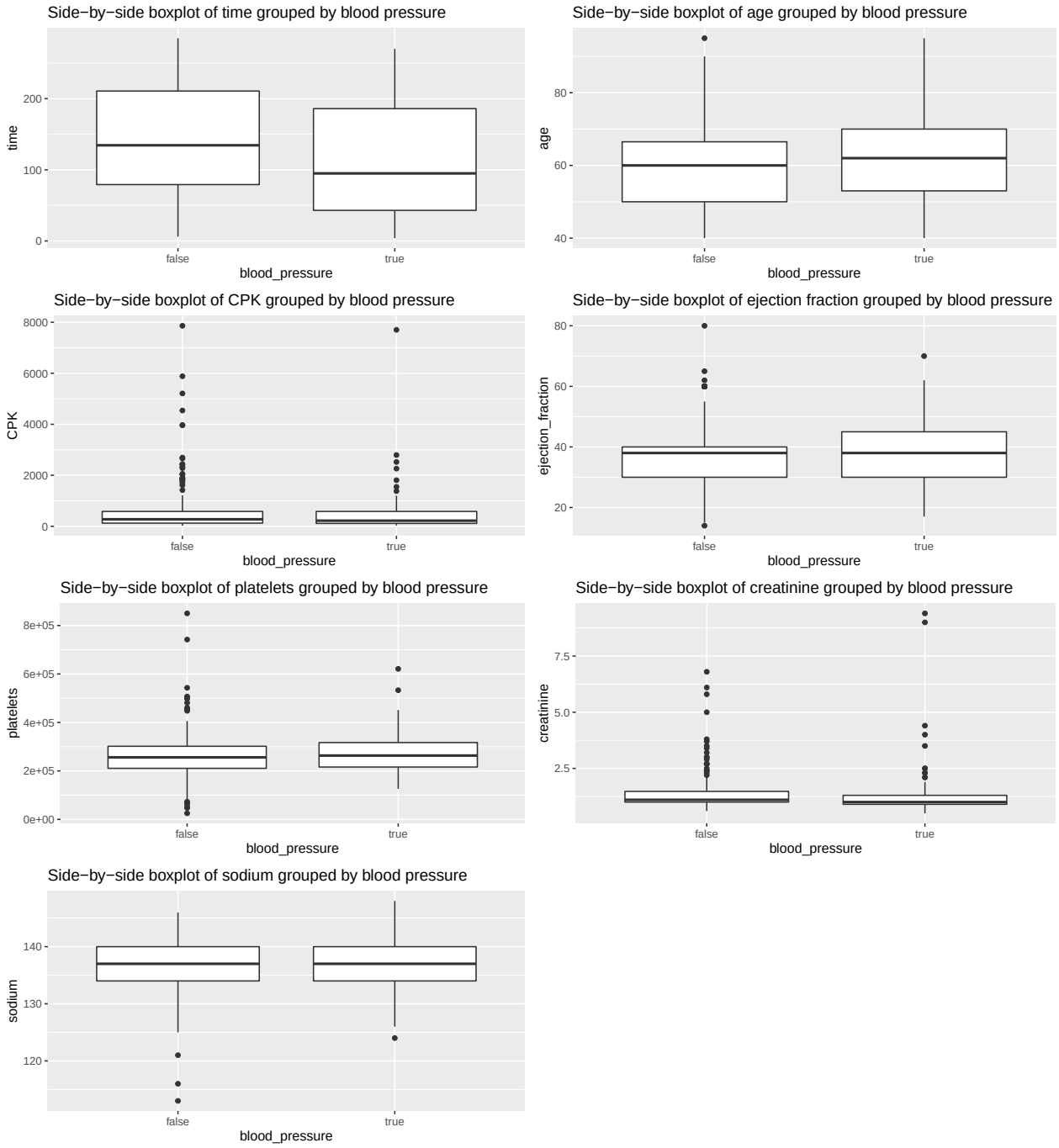

When we look the side-by-side boxplots for each quantitative variable above, we don't notice much difference at all between the distribution for those with anaemia vs. the distribution for those without anaemia. For example, the side-by-side boxplot for time grouped by anaemia indicates that those with anaemia and those without anaemia had rather similar median times until their next follow up visit. This same pattern - of negligible differences between the distributions of the various quantitative variables when grouped by anaemia - holds for all of the quantitative variables.

**Quantitative variables vs. blood pressure**

Finally, I want to examine the bivariate relationships between blood pressure and all of the quantitative variables, so I will make boxplots of all of those relationships as well.

```r
# Creating side-by-side boxplots for blood pressure x all quantitative variables

# time ~ blood_pressure
time_blood_pressure_boxplot <- gf_boxplot(time ~ blood_pressure, data = hearts) +
  labs(title = "Side-by-side boxplot of time grouped by blood pressure")

# age ~ blood_pressure
age_blood_pressure_boxplot <- gf_boxplot(age ~ blood_pressure, data = hearts) +
  labs(title = "Side-by-side boxplot of age grouped by blood pressure")

# CPK ~ blood_pressure
CPK_blood_pressure_boxplot <- gf_boxplot(CPK ~ blood_pressure, data = hearts) +
  labs(title = "Side-by-side boxplot of CPK grouped by blood pressure")

# ejection_fraction ~ blood_pressure
ejection_fraction_blood_pressure_boxplot <- gf_boxplot(ejection_fraction ~ blood_pressure,
                                                       data = hearts) +
  labs(title = "Side-by-side boxplot of ejection fraction grouped by blood pressure")

# platelets ~ blood_pressure
platelets_blood_pressure_boxplot <- gf_boxplot(platelets ~ blood_pressure,
                                               data = hearts) +
  labs(title = "Side-by-side boxplot of platelets grouped by blood pressure")

# creatinine ~ blood_pressure
creatinine_blood_pressure_boxplot <- gf_boxplot(creatinine ~ blood_pressure,
                                                data = hearts) +
  labs(title = "Side-by-side boxplot of creatinine grouped by blood pressure")

# sodium ~ blood_pressure
sodium_blood_pressure_boxplot <- gf_boxplot(sodium ~ blood_pressure, data = hearts) +
  labs(title = "Side-by-side boxplot of sodium grouped by blood pressure")
```

```r
# Displaying the boxplots together using grid.arrange
grid.arrange(time_blood_pressure_boxplot,
             age_blood_pressure_boxplot,
             CPK_blood_pressure_boxplot,
             ejection_fraction_blood_pressure_boxplot,
             platelets_blood_pressure_boxplot,
             creatinine_blood_pressure_boxplot,
             sodium_blood_pressure_boxplot,
             nrow = 4,
             ncol = 2)
```



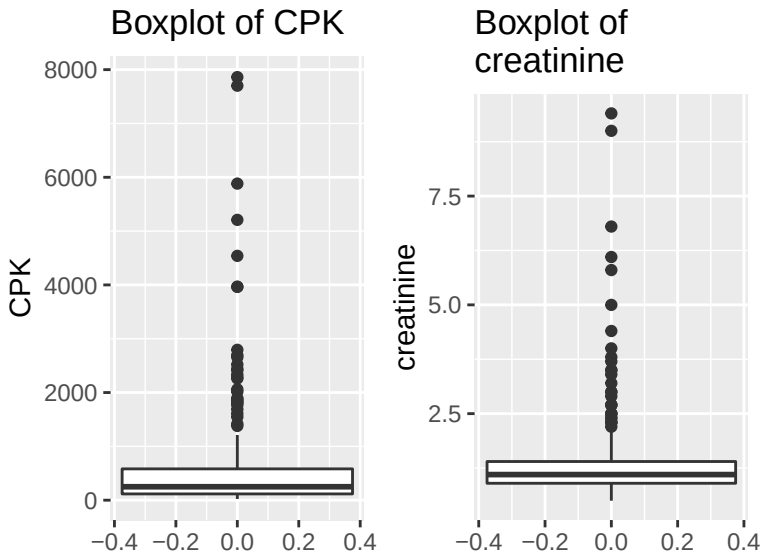

We see in the side-by-side boxplots above that for all of the quantitative variables, except time, the distribution of observations with high blood pressure and without high blood pressure appear relatively similar. However, when we look at the side-by-side boxplot for time, we notice that the median of the distribution for those with high blood pressure is lower than the median of the distribution for those without high blood pressure. Recall that the time variable indicates the number of days between a patient's most recent visit and their next follow up visit. Perhaps one explanation for the difference in follow-up time among those with high blood pressure versus those without high blood pressure is that physicians are making sure to see the patients with high blood pressure because they possibly consider them higher risk for dying from heart failure.

## Outliers

### Univariate outliers

We noticed in the scatterplot matrix that the univariate distribution of CPK and creatinine are both heavily skewed, which could potentially be due to outliers. In order to identify if there are outliers in these distributions, I will make boxplots for each of these univariate distributions.

```
# Creating boxplot for CPK
CPK_boxplot <- gf_boxplot(~ CPK, data = hearts) +
  labs(title = "Boxplot of CPK")

# Creating boxplot of creatinine
creatinine_boxplot <- gf_boxplot(~ creatinine, data = hearts) +
  labs(title = "Boxplot of\ncreatinine")

# Display using grid.arrange
grid.arrange(CPK_boxplot, creatinine_boxplot, nrow = 1, ncol = 2)
```




The boxplot for the univariate distribution of of CPK shows at least 15 observations greater than 1.5 times the interquartile range. We may want to consider omitting these points from analyses, but we that is a later decision.

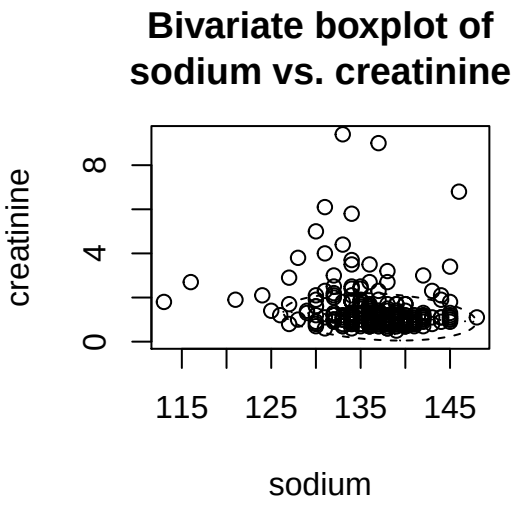
The boxplot for creatinine shows that there are at least 15 observations that extend beyond 1.5 times the interquartile range. We must consider what to do with these observations before beginning analyses.

### Bivariate outliers

Earlier, I looked at the scatterplots in the scatterplot matrix in order to look into the bivariate distribution of the quantitative variables. During this investigation, I flagged some bivariate relationships as potentially having outliers. I am going to look into these pairings of bivariate relationships by making multiple bivariate boxplots. The pairings I am going to look at are:

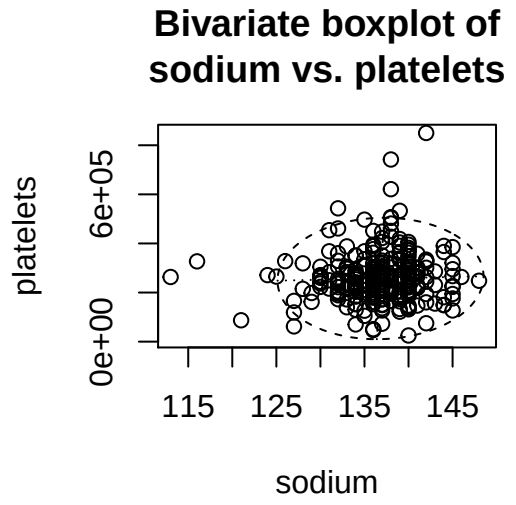
- sodium vs. creatinine
- sodium vs. platelets
- sodium vs. CPK
- creatinine vs. platelets
- creatinine vs. CPK
- platelets vs. CPK

```r
# Sodium x creatinine
sodium_creatinine <- hearts %>% select(sodium, creatinine)
bvbox(as.matrix(sodium_creatinine), xlab = "sodium", ylab = "creatinine")
title(main ="Bivariate boxplot of\nsodium vs. creatinine")
```



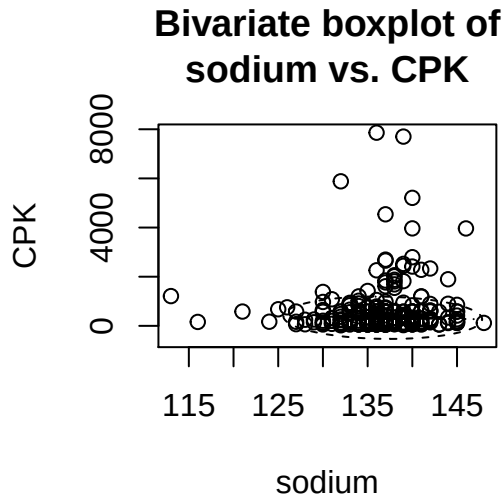

We see in the bivariate boxplot above that there are observations beyond the fence. We may consider only including points for which creatinine $< 2.25$ and for which sodium $> 125$.

```r
# Sodium x platelets
sodium_platelets <- hearts %>% select(sodium, platelets)
bvbox(as.matrix(sodium_platelets), xlab = "sodium", ylab = "platelets")
title("Bivariate boxplot of\nsodium vs. platelets")
```



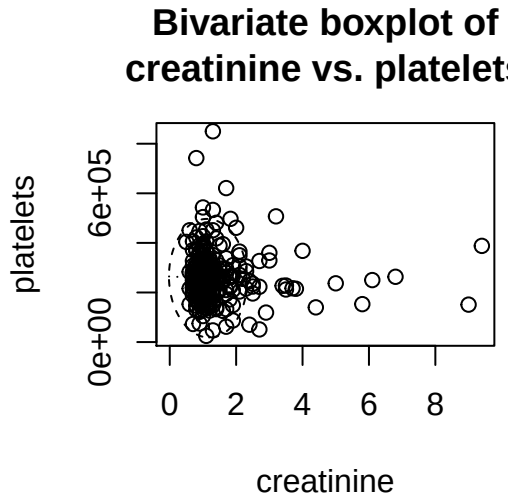

Based upon the bivariate boxplot above, we may consider only including points for which platelets $< 500{,}000$ and sodium $> 125$.

```r
# Sodium x CPK
sodium_CPK <- hearts %>% select(sodium, CPK)
bvbox(as.matrix(sodium_CPK), xlab = "sodium", ylab = "CPK")
title("Bivariate boxplot of\nsodium vs. CPK")
```



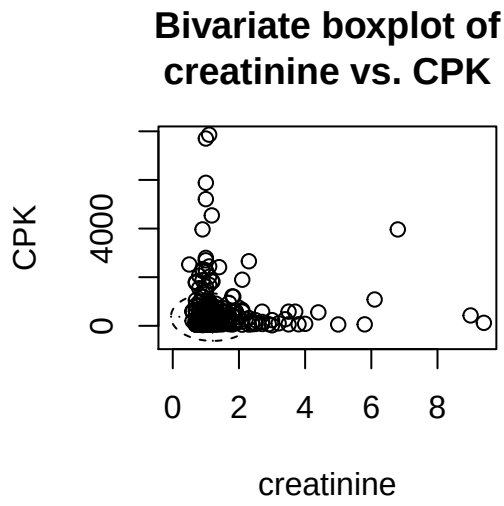

When we look at the bivariate boxplot of sodium vs. CPK, we see points beyond the fence that may be considered outliers, so we may want to only retain observations for which sodium $> 125$ and CPK $< 1600$.

```
# creatinine x platelets
creatinine_platelets <- hearts %>% select(creatinine, platelets)
bvbox(as.matrix(creatinine_platelets), xlab = "creatinine", ylab = "platelets")
title("Bivariate boxplot of\ncreatinine vs. platelets")
```



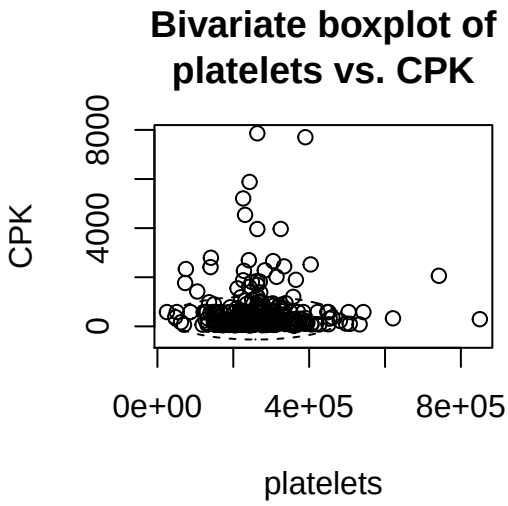

The bivariate boxplot of creatinine vs. platelets indicates that we may want to only include observations for which creatinine $< 2.25$ and platelets $< 500{,}000$ in order to remove outliers.

```
# creatinine x CPK
creatinine_CPK <- hearts %>% select(creatinine, CPK)
bvbox(as.matrix(creatinine_CPK), xlab = "creatinine", ylab = "CPK")
title("Bivariate boxplot of\ncreatinine vs. CPK")
```




When we look at the bivariate boxplot of creatinine vs. CPK, we see that we might want to consider only including observations for which creatinine $< 2.25$ and CPK $< 1600$.

```
# platelets x CPK
platelets_CPK <- hearts %>% select(platelets, CPK)
bvbox(as.matrix(platelets_CPK), xlab = "platelets", ylab = "CPK")
title("Bivariate boxplot of\nplatelets vs. CPK")
```




In the bivariate boxplot for platelets and CPK, we see outliers, and so to remove them, we may consider only including observations for which platelets $< 500{,}000$ and CPK $< 1600$.

**Decision with outliers**

When I examined the univariate distributions of CPK and creatinine, I noticed outliers but I wanted to examine CPK and creatinine in bivariate settings before deciding if I should go through with removing outliers.

When I looked at six bivariate relationships through bivariate boxplots, I noticed consistencies between graphs in the observations that were beyond the fence and were outliers. Specifically, the bivariate boxplots seemed to suggest that I should:

1) Remove observations for which creatinine $> 2.25$
2) Remove observations for which sodium $< 125$
3) Remove observations for which platelets $> 500{,}000$

4) Remove observations for which CPK $> 1600$

I plan to proceed with removing the observations that meet these criteria. Note that the bivariate boxplots confirmed that there are observations with extreme values for creatinine and CPK that ought to be removed, which is what the univariate distributions for creatinine and CPK suggested.

**Removing outliers**

```
# Filter the data set based on the inclusion/exclusion criteria
hearts_filtered <- hearts %>%
  filter(creatinine < 2.25,
         sodium > 125,
         platelets < 500000,
         CPK < 1600
         )

# Number of rows in the original data set
nrow(hearts)
```

```
[1] 299
```

```
# Number of rows in the filtered data set
nrow(hearts_filtered)
```

```
[1] 238
```

When we filter the data based on the inclusion/exclusion criteria, we see that the number of observations reduces from 299 to 238.

## Final thoughts

After performing this preliminary analysis, there are a few findings that I should remember as I move forward into performing my analysis:

1) The bivariate distribution of blood pressure and death indicates that among those who died, 37% had high blood pressure while only 29% did not.
2) The bivariate distribution of anaemia and death indicates that among those who died, 35% had anaemia while 29% did not.
3) The median follow up time for those who died is far less than the median follow up time for those who survived.
4) The median age of those who died is about 5 years greater than the median age of those who survived.
5) The median ejection fraction of those who died is lower than the median ejection fraction of those who survived.
6) The median follow up time for those with high blood pressure was less than the median follow up time for those without high blood pressure.

# Methods

I will be using two statistical analysis methods as I seek to address the research question of predicting survival of heart failure patients: clustering and classification. In the following subsections, I describe the specific types of clustering and classification methods that I will be using. I also provide a description for these methods.

## Clustering

I am performing clustering on the data set in an effort to identify groups of individuals among those with heart failure. If I am able to find clusters in the data, I could then use those clusters and make models predicting survival for each cluster, which would ultimately allow for a more tailored approach to identifying risk factors for dying.

I will be using agglomerative hierarchical clustering and k-means clustering. For both of these methods, I will use Euclidean distance and will scale the data. For k-means, I have to scale the data because the method is not scale invariant. I will also scale the data for our agglomerative hierarchical clustering because the data is not all on the same scale.

### Agglomerative hierarchical clustering

First, I will perform agglomerative hierarchical clustering. The methodology for performing agglomerative hierarchical clustering starts with having $n$ clusters, each that contain a single observation. Then, partitions are created by fusing the $n$ individuals together until there is just 1 cluster. At the end of this process, a decision needs to be made about what the final solution will be. In order to fuse clusters together, I will use the two aforementioned types of linkages: *Ward's method* and *complete linkage*.

When performing clustering using Ward's linkage, the computer identifies which fusion of two clusters will yield a new cluster with the smallest WGSS. It then merges those two clusters, and the process continues until there is just 1 cluster. When performing clustering using complete linkage, the computer identifies which fusion of two clusters would result in the maximum distance between an observation $i$ found in cluster $A$ and observation $j$ found in cluster $B$. The computer then performs the fusion.

When performing agglomerative hierarchical clustering, I will plot a dendrogram (I will do this for both a Ward's method and complete linkage solution). I will notice the height at which the merging of clusters seems to slow down in a dendrogram, and I will use the number of clusters found at this height to determine how many clusters I will ultimately use in a given solution.

### K-means clustering

Another clustering method that I will use is K-means clustering. K-means clustering involves obtaining an initial partition of individuals into the required $k$ number of groups. Then, the computer calculates the change in criterion (we will use WGSS) that is produced by moving points into different cluster. At each iteration stage, the computer makes the change that leads to the greatest improvement in WGSS. This process continues until the change in WGSS converges to 0, signifying that there is no significant improvement in WGSS made by moving points. I will look at a WGSS plot to help me determine the appropriate number of clusters to choose for the solution by looking for an elbow in the plot.

## Classification

The other statistical analysis method that I will be utilizing is classification. I will be employing two specific classification methods: classification trees, and random forests. I will be performing classification in an effort to help tackle a question whose answer has eluded researchers, which is "How can we predict if a patient with heart failure is going to die?". Even if the classification models that I create lack satisfactorily high accuracy, there is nevertheless potential to learn about important predictors by examining accuracy and Gini plots from the models.

**Classification trees:**

Classification trees are constructed by partitioning the variable space in a recursive manner into hypercubes. The hypercubes are used to separate the classes, and the goal is to maximize the purity of each hypercube (which means that all values in a hypercube are of the same class). When making a classification tree, we ideally want the end of a branch (a node) to be as pure as possible, and we can calculate impurity using an impurity measure (there are 3 standard ones). We also want to make sure that the tree does not overfit the data, so we can also utilize pruning, which specifies stopping conditions for the tree growth.

When we make a classification tree, we can use `rpart.control`. We can specify:

- `minsplit`: the minimum number of observations that must exist in a node in order for a split to be attempted
- `minbucket:` the minimal number of observations in any terminal node
- `xval`: number of cross-validations
- `cp`: complexity parameter (utilized when pruning trees)

**Random forests**

Random forests are an example of an ensemble method (which refers to the idea of combining methods to improve prediction). When performing random forest classification, we generate many predictions for each observation and use majority voting to determine the final class. We do this by looking at many classification trees, and we alter which observations are given as data to the algorithm by only giving the algorithm a few random variables to try at each node.

When we run random forests, we use `randomForest` in R. We can specify:

- `mtry`: the number of variables randomly selected as candidates at each split
- `ntree`: the number of trees to grow

We should also specify the `importance` and `proximity` arguments to be `TRUE`.

## Identifying best models:

There will be specific steps that I take and criteria that I look to in order to determine my best clustering model and my best classification model.

When evaluating my clustering models, I will use silhouette plots in order to identify cluster validation. I will consider average cluster silhouette values and silhouette coefficients, and I will regard models with high silhouette coefficients (and also high average cluster silhouette values) to be better than others.

When evaluating my classification models, I will first find the best model made using classification trees by looking for models with low values for AER (the error rate on the training set) and low values of TER (the error rate on the testing set). Note that I will prioritize models with a low TER, as that will be a better metric than AER for assessing the generalizability of a model to other data sets. I will then find the best model made using random forests, which will also involve looking for models with low AER and low TER. Finally, I will chose an overall classification model by comparing my best classification tree model and my best random forests model.

# Results

## Clustering

In order to perform clustering, I need to only use quantitative variables. I will start by making a data frame with all categorical variables removed.

```
# Data set with only quantitative variables in it
hearts_quantitative <- hearts_filtered %>% select(time,
                                                  age,
                                                  ejection_fraction,
                                                  sodium,
                                                  creatinine,
                                                  platelets,
                                                  CPK)
```

### Agglomerative hierarchical clustering

The first step I need to take before performing agglomerative hierarchical clustering is to compute a scaled distance matrix.

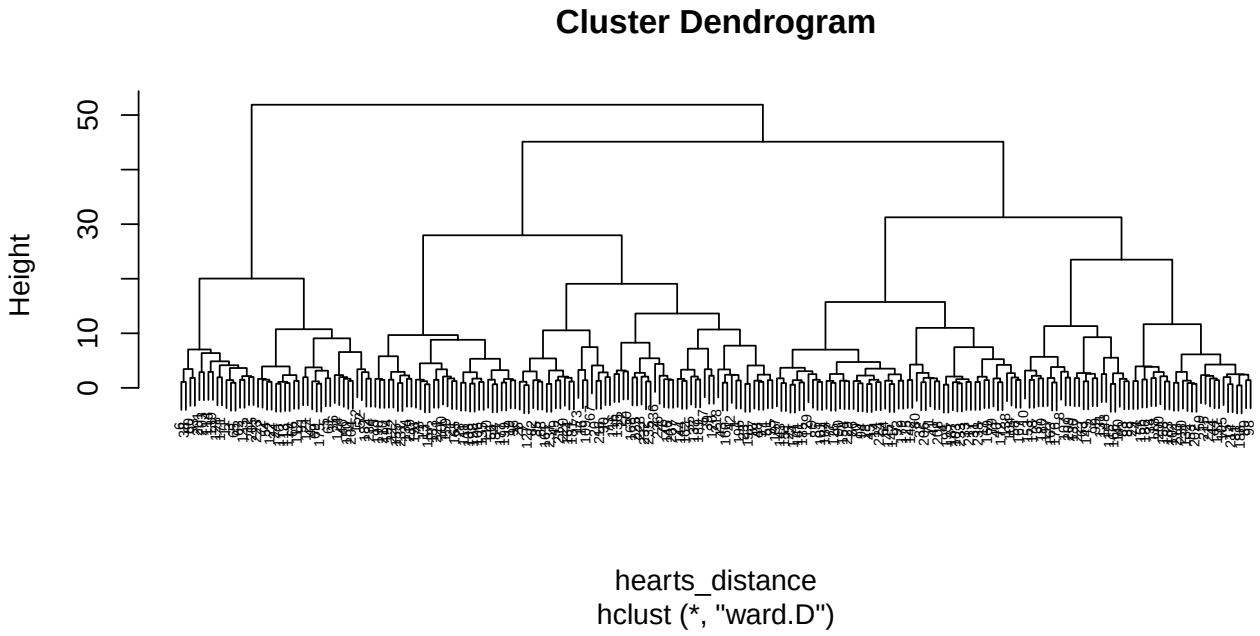
```
# Distance matrix
hearts_distance <- dist(scale(hearts_quantitative)) # Note that we are scaling here
```

Next, I can run hierarchical agglomerative clustering using **Ward's method** for linkage.

```
# Running hierarchical agglomerative clustering with Ward's method
ward <- hclust(hearts_distance, method = "ward.D")
```

I will make a dendrogram for this model so that I can try to see when the merging of clusters seems to slow down.

```
# Making the dendrogram
plot(ward, cex = 0.5)
```




It looks like the merging of clusters seems to slow down at about the point where there are 6 clusters. I will

cut the tree by specifying $k = 6$.

```
# Cut the tree specifying k = 6
ward_k6_solution <- cutree(ward, k = 6)

# Display number of observations in each cluster
summary(as.factor(ward_k6_solution))
```

```
 1  2  3  4  5  6
57 24 43 55 32 27
```

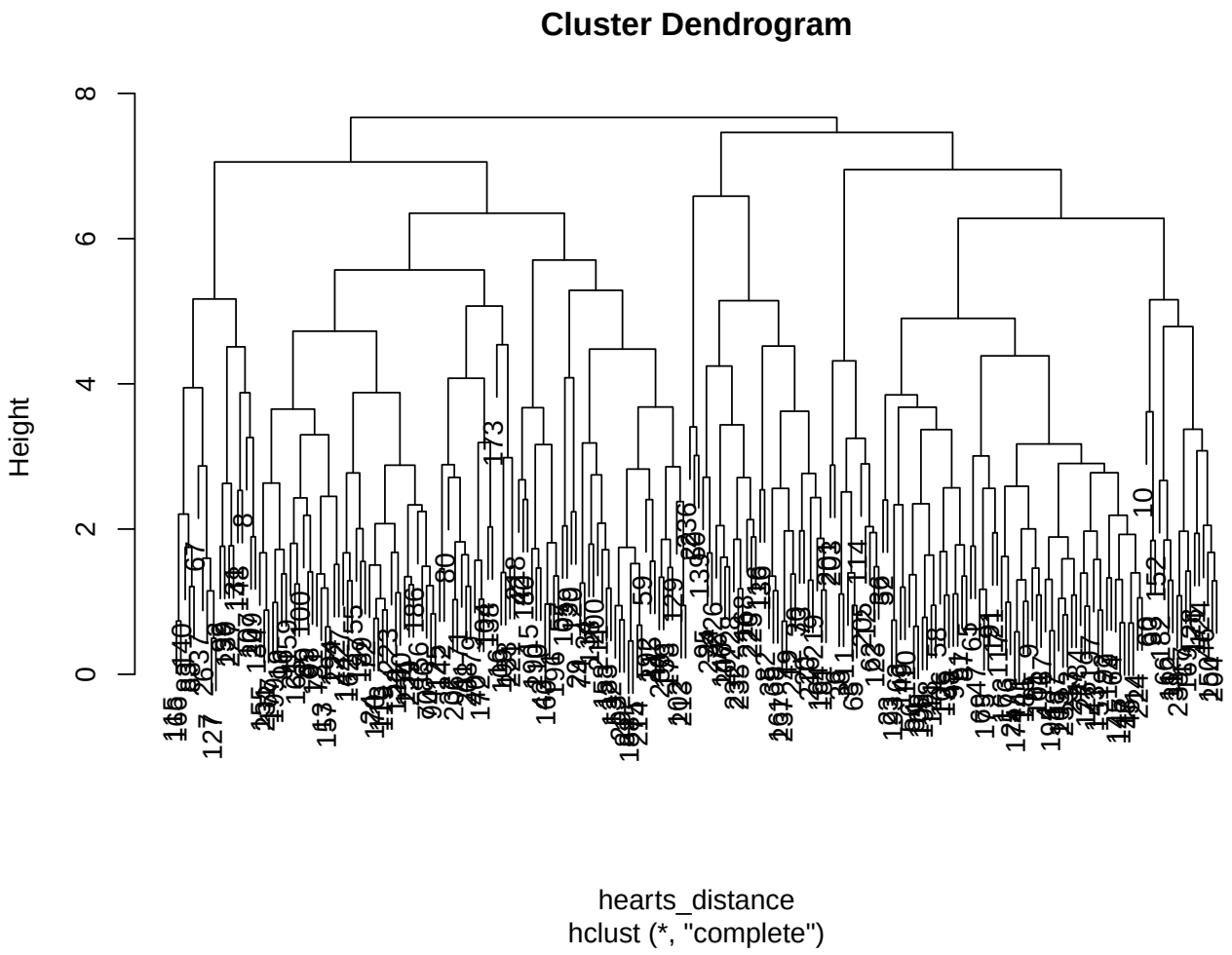
When we cut the tree, we see that all of the clusters have between 24 and 57 observations. Two clusters have particularly large amounts of observations in them (cluster 1 with 57 observations, and cluster 4 with 55 observations). Two clusters have moderate amounts of observations in them (cluster 3 has 43 observations and cluster 5 has 32 observations). Finally, two clusters have the fewest number of observations in them (cluster 2 with 24 observations and cluster 6 with 27 observations).

I also want to create a clustering model using **complete linkage**.

```
# Running hierarchical agglomerative clustering with complete linkage
complete_linkage <- hclust(hearts_distance, method = "complete")
```

I will make a dendrogram so that I can identify when the merging of clusters slows down.

```
# Making dendrogram
plot(complete_linkage)
```



It looks like the merging of clusters slows down at the point where there are approximately 5 clusters. We will cut the tree by specifying $k = 5$.

```
# Cut the tree specifying k = 5
complete_k5_solution <- cutree(complete_linkage, k = 5)

# Display the number of observations in each cluster
summary(as.factor(complete_k5_solution))
```

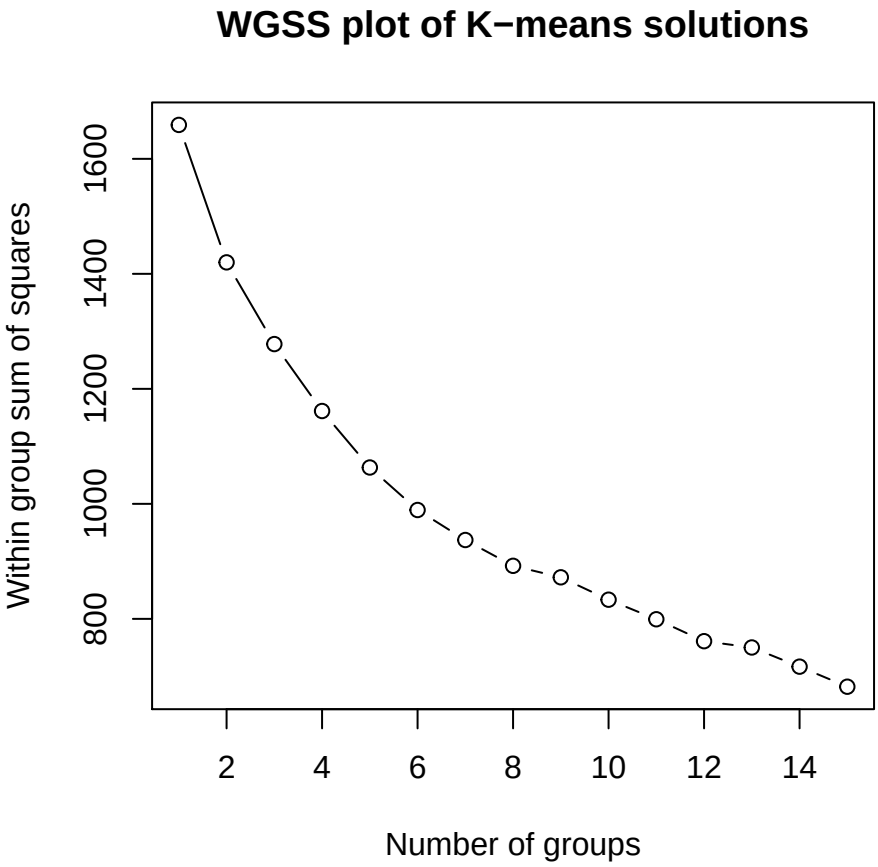
```
 1  2  3  4  5
98 19 12 77 32
```

We see that cluster 1 has the most observations in it out of any cluster with 98 observations. Cluster 4 also has a relatively high number of observations with 77 observations in it. The remaining three clusters have relatively low numbers of observations in each of them (19, 12, and 32 observations for clusters 2, 3, and 5, respectively).

### K-means clustering

The other clustering method that I am going to run is K-means clustering. The code below will allow me to generate a WGSS plot of K-means solutions. From that plot, I will be able to try and discern where I believe there is an "elbow" in the graph, meaning, at what value of $k$ it seems that the decrease in WGSS seems to tail off.

```
# Set seed
set.seed(75)

# Making an empty vector for within group sum of squares
wss <- rep(x = 0, times = 15)

# Generate within group sum of squares
for(i in 1:15) {
  wss[i] <- sum(kmeans(scale(hearts_quantitative), centers = i)$withinss)
}

# Plot the WGSS
plot(1:15, wss, type = "b", xlab = "Number of groups",
     ylab = "Within group sum of squares", main = "WGSS plot of K-means solutions")
```




Trying to identify an elbow in a WGSS plot is a notoriously difficult objective due to the subjectivity of the endeavor. However, I see a plateau starting at $k$=8, so I am going to choose a cluster size of 8 for a solution.

I am going to save an object for this solution so that I can use it later when performing cluster validation.

```
# Set seed
set.seed(75)

# K-means solution with k = 8
kmeans_k8_solution <- kmeans(scale(hearts_quantitative), centers = 8)
```

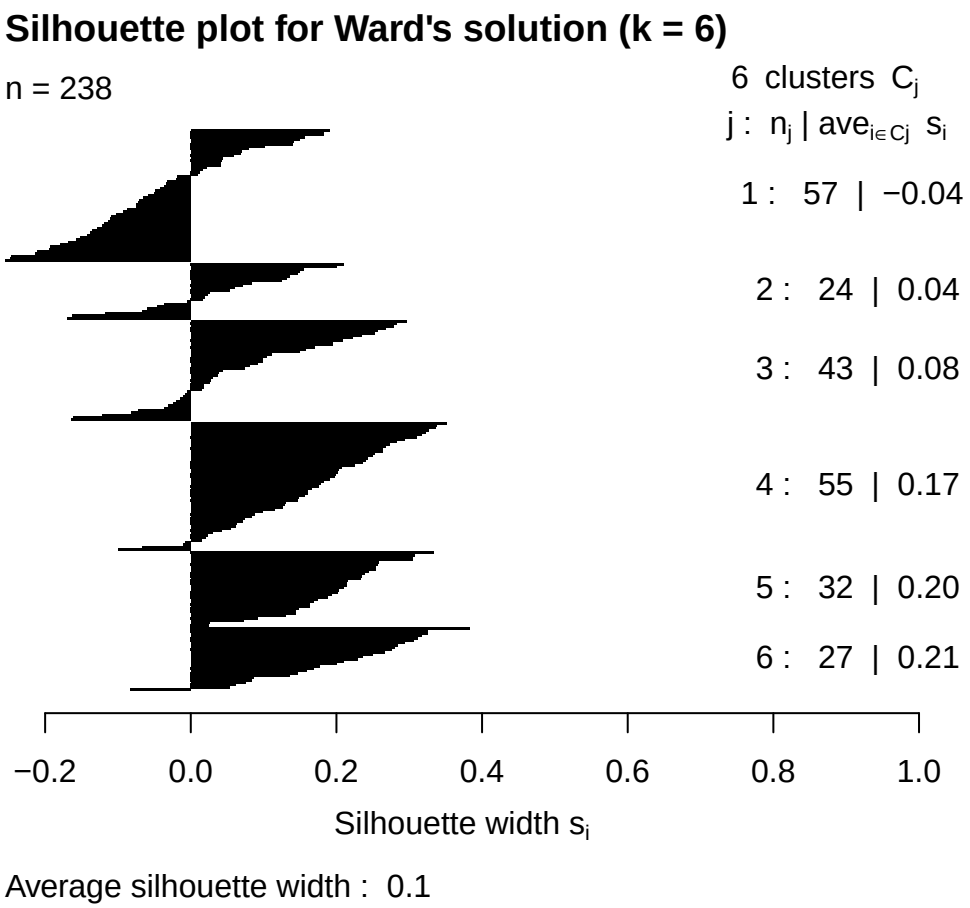
### Cluster validation

I am going to assess cluster validation by making a silhouette plot for each solution that I have come up with. To review, the models that I am considering are:

1) An agglomerative hierarchical solution with Ward's method, with $k = 6$
2) An agglomerative hierarchical solution made with complete linkage, where $k = 5$
3) A K-means solution, where $k = 8$

*Ward's method solution with $k = 6$*

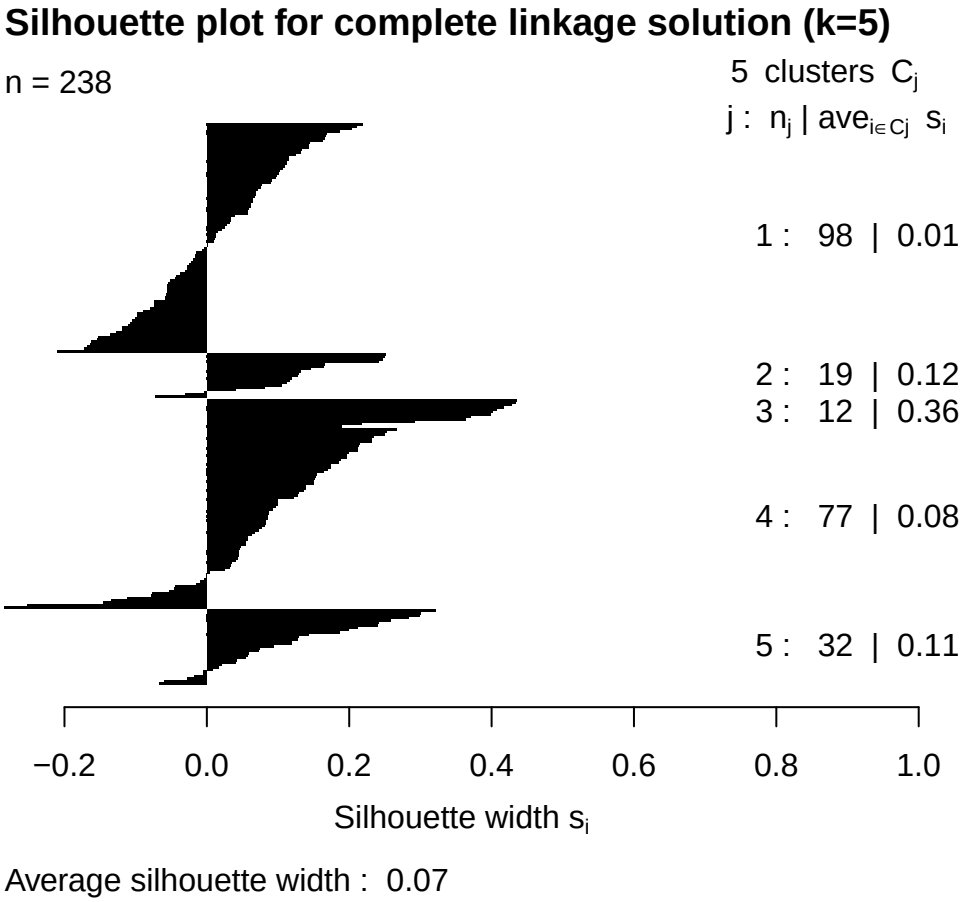
```
# Silhouette plot for Ward's method with k = 6
ward_silhouette <- silhouette(ward_k6_solution, hearts_distance)
plot(ward_silhouette, col = "black", main = "Silhouette plot for Ward's solution (k = 6)")
```




When we look at a silhouette plot for the $k$=6 solution made using Ward's method, we see that the model has no substantial structure, as the silhouette coefficient is 0.1. Additionally, all of the average cluster silhouette values indicate that none of the clusters have substantial structures, as the values are all less than 0.21. We notice that cluster 1 actually has a negative average cluster silhouette value, indicating that many points in that cluster would be better off in a different cluster than cluster 1. In fact, in all of the clusters except cluster 5, we see observations with negative average cluster silhouette values. In short, this cluster solution is most certainly not impressive.

*Complete linkage solution with $k = 5$*

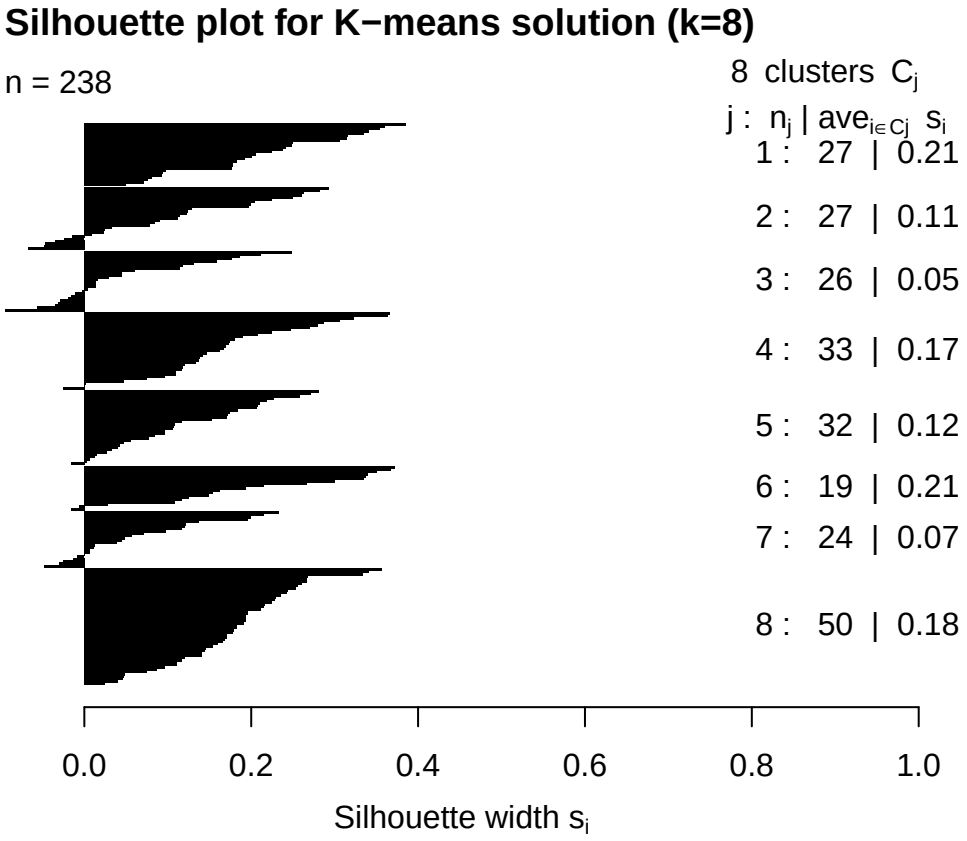
```
# Silhouette plot for complete linkage with k = 5
complete_silhouette <- silhouette(complete_k5_solution, hearts_distance)
plot(complete_silhouette, col = "black",
     main = "Silhouette plot for complete linkage solution (k=5)")
```




The silhouette plot for the complete linkage solution with 5 clusters tells us that this model has no substantial structure, given that the silhouette coefficient is 0.07. When we look at the individual clusters, we see that four of the clusters (clusters 1, 2, 4, and 5) have average cluster silhouette values that are less than 0.12, indicating that each of these clusters have no substantial structure. However, cluster 3 has a average cluster silhouette value of 0.36, which we can say indicates a weak, but potentially artificial structure, per the rule of thumb. We see that all clusters except cluster 3 have observations that have negative average cluster silhouette values, which helps further support that clusters 1, 2, 4, and 5 have no substantial structure. Given that most of the clusters have no substantial structure and that the overall solution has no substantial structure, this solution is not impressive.

*K-means solution with $k = 8$*

```
# Silhouette plot for K-means with k = 8
kmeans_silhouette <- silhouette(kmeans_k8_solution$cluster, hearts_distance)
plot(kmeans_silhouette, col = "black",
     main = "Silhouette plot for K-means solution (k=8)")
```




The silhouette plot for the k-means solution has an silhouette coefficient of 0.14, indicating that the model has no substantial structure. Additionally, all of the 8 clusters have average cluster silhouette values that are less than 0.21, indicating that all of the clusters have no substantial structure. Six of the clusters (clusters 2, 3, 4, 5, 6, and 7) have observations with negative average cluster silhouette values, indicating that those observations would be better off in a different cluster than the one they are in. Taking all of this into consideration, we conclude that this clustering solution is not very impressive at all.

**Identifying best clustering solution**

Cluster validation indicates that none of the solutions are particularly impressive. The best clustering solution was the k-means solution, which had a silhouette coefficient of 0.14. The next best was the Ward's method solution, which had an average cluster silhouette value of 0.10, and the worst was the complete linkage solution, with an average cluster silhouette value of 0.07. While the k-means solution may have outperformed the others, it still is not impressive whatsoever.

**Takeaways**

The objective with clustering is to be able to identify groups of observations. I chose to use clustering to see if, among those with heart failure, there were groups of individuals. My hope was that if I could identify groups of individuals, I could then try to see if there are specific risk factors for dying depending upon the cluster when performing classification. Unfortunately, given these unimpressive clustering solutions, I have not been able to identify groups of individuals, but future analysis could be conducted to try to run more clustering analyses. Despite the fact that even my best clustering solution failed to offer insight into groups of observations, I will proceed with performing classification on the data (the entire data set, not by cluster) so that I can identify certain risk factors for dying.

## Classification

In addition to using clustering to analyze the data, I also plan to use classification. I will perform two different types of classification - classification trees, and random forests.

### Classification trees

The first classification method that I will use is classification trees. When making classification trees, I will explore three different models, each with varying specifications for `minsplit`, `minbucket`, and `xval`.

First, I will have to make a data frame that has all variables stored as numerics (except death), even the categorical variables.

```
# Create data frame hearts_numerics
hearts_numerics <- hearts_filtered

# Change the categorical variables to be numerics

# Sex
hearts_numerics$sex <- as.numeric(hearts_numerics$sex)

# Diabetes
hearts_numerics$diabetes <- as.numeric(hearts_numerics$diabetes)

# High blood pressure
hearts_numerics$blood_pressure <- as.numeric(hearts_numerics$blood_pressure)

# Smoking
hearts_numerics$smoking <- as.numeric(hearts_numerics$smoking)

# Anaemia
hearts_numerics$anaemia <- as.numeric(hearts_numerics$anaemia)
```

#### Model 1:

For the first model I will make, I will specify `minsplit = 10`, `minbucket = 3`, and `xval = 0`. I will make the model but will not display the tree nor the model summary for the sake of brevity.

```
# Set seed
set.seed(75)

# Set parameters
tree_control1 <- rpart.control(minsplit = 10, minbucket = 3, xval = 0)

# Fit model
tree_model1 <- rpart(death ~ ., data = hearts_numerics,
                     method = "class", control = tree_control1)
```

```
# Tree
plot(tree_model1)
text(tree_model1, cex = 0.7)

# Model summary
printcp(tree_model1)
```

It is important that we take note of the AER of this model. Recall that AER = root node error $*$ final relative error. Note that we cannot calculate TER for this model because we did not use cross validation.

```r
# Root node error
classification_tree_RNE_model1 <- 0.2899

# Calculating AER
classification_tree_AER_model1 <- classification_tree_RNE_model1 * 0.2899

# Display AER
cat("AER =", round(classification_tree_AER_model1, 4))
```

```
AER = 0.084
```

We see that we get AER = 0.084.

**Model 2:**

We will go through the same process as outlined above for model 1 when making model 2. I will specify `minsplit = 10`, `minbucket = 3`, and `xval = 200`. As with model 1, I will not display the model summary nor the tree for the model for the sake of brevity.

```r
# Set seed
set.seed(75)

# Set parameters
tree_control2 <- rpart.control(minsplit = 10, minbucket = 3, xval = 200)

# Fit model
tree_model2 <- rpart(death ~ ., data = hearts_numerics,
                     method = "class", control = tree_control2)
```

```r
# Tree
plot(tree_model2)
text(tree_model2, cex = 0.7)

# Model summary
printcp(tree_model2)
```

We need to calculate the AER and TER of this model. Recall that AER = root node error$*$final relative error, and that TER = root node error $*$ final xerror.

```r
# Root node error
classification_tree_RNE_model2 <- 0.2899

# AER
classification_tree_AER_model2 <- classification_tree_RNE_model2 * 0.2899

# TER
classification_tree_TER_model2 <- classification_tree_RNE_model2 * 0.6957

# Display AER and TER
cat("AER = ", round(classification_tree_AER_model2, 4),
    "   TER = ", round(classification_tree_TER_model2, 3))
```

```
AER =  0.084    TER =  0.202
```

We see that the AER for model 2 is 0.084 and the TER is 0.202.

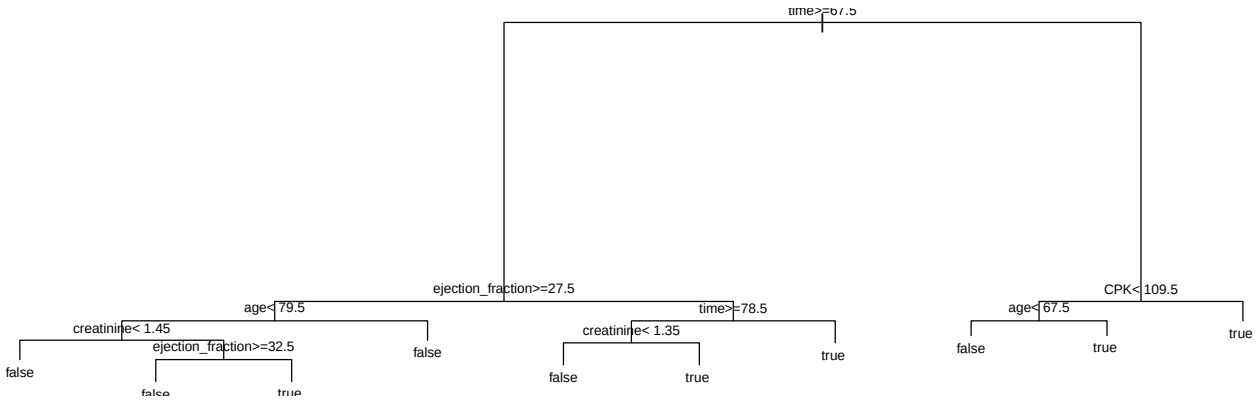
**Model 3:**

For the third and final classification tree model that I will make, I will specify `minsplit = 10`, `minbucket = 3`, and `xval = 10`.

```
# Set seed
set.seed(75)

# Set parameters
tree_control3 <- rpart.control(minsplit = 10, minbucket = 3, xval = 10)

# Fit model
tree_model3 <- rpart(death ~ ., data = hearts_numerics,
                     method = "class", control = tree_control3)
```

We can look at the tree created by this model, which is displayed below.

```
# Display tree
plot(tree_model3)
text(tree_model3, cex = 0.7)
```




We also will want to look at the model summary.

```
# Display model summary
printcp(tree_model3)
```

```
Classification tree:
rpart(formula = death ~ ., data = hearts_numerics, method = "class",
    control = tree_control3)

Variables actually used in tree construction:
[1] age               CPK               creatinine        ejection_fraction
[5] time

Root node error: 69/238 = 0.28992

n= 238

        CP nsplit rel error  xerror     xstd
1 0.550725      0   1.00000 1.00000 0.101445
2 0.028986      1   0.44928 0.50725 0.079185
3 0.014493      4   0.36232 0.60870 0.085234
4 0.010000      9   0.28986 0.65217 0.087548
```

We need to calculate the AER and TER of this model. Recall that AER = root node error$*$final relative error,

and that TER = root node error $*$ final xerror.

```
# Root node error
classification_tree_RNE_model3 <- 0.2899

# AER
classification_tree_AER_model3 <- classification_tree_RNE_model3 * 0.2899

# TER
classification_tree_TER_model3 <- classification_tree_RNE_model3 * 0.6522

# Display AER and TER
cat("AER = ", round(classification_tree_AER_model3, 4),
    "   TER = ", round(classification_tree_TER_model3, 3))
```

```
AER =  0.084    TER =  0.189
```

We see that the AER for model 2 is 0.084 and the TER is 0.189.

**Which classification tree model is the best?**

In order to choose the best classification tree model, I have to look at the AER's and TER's for all of the models. Table 3 summarizes the AER's and TER's for all of the models.

```
# AER's for all models
classification_tree_AER_all_models <- c(classification_tree_AER_model1,
                                        classification_tree_AER_model2,
                                        classification_tree_AER_model3)

# Round AER's
classification_tree_AER_all_models <- round(classification_tree_AER_all_models,
                                            4)

# TER's for all models
classification_tree_TER_all_models <- c(" ",
                  round(classification_tree_TER_model2, 3), # Round TER
                  round(classification_tree_TER_model3, 3)) # Round TER

# Dataframe of AER's and TER's for all models
classification_tree_AER_TER_all_models <- data.frame(
  cbind(classification_tree_AER_all_models, classification_tree_TER_all_models))

# Add row names
classification_tree_AER_TER_all_models <- cbind(
  c("Model 1", "Model 2", "Model 3"), classification_tree_AER_TER_all_models
  )

# Change column names
colnames(classification_tree_AER_TER_all_models) <- c("Model", "AER", "TER")
```

```
# Display AER's and TER's for classification trees
kable(classification_tree_AER_TER_all_models, "latex", booktabs = TRUE) %>%
  kable_styling(position = "center") %>%
  add_header_above(c("Table 3: AER's and TER's\nfor classification tree models" = 3))
```

Table 3: AER's and TER's for classification tree models

| Model | AER | TER |
|---|---|---|
| Model 1 | 0.084 | |
| Model 2 | 0.084 | 0.202 |
| Model 3 | 0.084 | 0.189 |

I can look at Table 3 to try to find the best classification tree model. Specifically, I am looking for a model that minimizes both AER and TER.

All of the models have the same AER (0.084), so I will look to TER to help me decide upon the best model. Recall that there is no TER for model 1 since we did not use cross validation. Model 3, with a TER of 0.189, has a lower TER than model 2 does (TER = 0.202). I am going to choose model 3 as the best classification tree model. It has the same AER as the other two models, and it has a lower TER than model 2 does. I wouldn't necessarily feel comfortable choosing model 1 because I have no TER due to not having used cross validation.

### Random forests

In addition to creating classification trees, I am also going to generate random forests models. I will use varying inputs for `mtry` and `ntree` when making my models.

**Model 1:**

The first random forests model I will make will be created by specifying `mtry = 3` and `ntree = 500`.

```
# Set seed
set.seed(75)

# Create model
random_forest_model1 <- randomForest(death ~ ., data = hearts_numerics,
                                     mtry = 3, ntree = 500,
                     importance = T, proximity = T)
```

```
# Model output
random_forest_model1

# Confusion matrix
table(hearts_numerics$death, predict(random_forest_model1,
                                              hearts_numerics))

# Distribution of tree size
gf_histogram(~ treesize(random_forest_model1))

# Accuracy and Gini plots
varImpPlot(random_forest_model1)
```

We can calculate AER and TER in order to assess model 1. Recall that for random forests models, we can get the TER from looking at the OOB estimate. We can get the AER by summing any non-zero value not on the diagonal of a confusion matrix, and then dividing that sum by the number of observations (and multiplying that quotient by 100).

Note that for the sake of brevity, neither the model output, confusion matrix, histogram of the distribution of tree size, nor accuracy and Gini plots are displayed. Regardless, the model output indicates that the TER for model 1 is 15.97% and the AER is 0%.

**Model 2:**

The next random forest model that I will make will have be specified to have `mtry = 4` and `ntree = 100`.

```
# Set seed
set.seed(75)

# Create model
random_forest_model2 <- randomForest(death ~ ., data = hearts_numerics,
                                     mtry = 4, ntree = 100,
                     importance = T, proximity = T)
```

```
# Model output
random_forest_model2

# Confusion matrix
table(hearts_numerics$death, predict(random_forest_model2,
                                           hearts_numerics))

# Distribution of tree size
gf_histogram(~ treesize(random_forest_model2))

# Accuracy and Gini plots
varImpPlot(random_forest_model2)
```

We will calculate AER and TER in order to assess model 2. For the sake of brevity, neither the model output, confusion matrix, histogram of the distribution of tree size, nor accuracy and Gini plots are displayed. Regardless, the OOB estimate is 15.95%, which means that TER = 16.39%. There are no non-zero values off the diagonals in the confusion matrix, so the AER is 0%.

**Model 3:**

For my third random forests model, I will specify `mtry = 5` and `ntree = 1000`.

```
# Set seed
set.seed(75)

# Create model
random_forest_model3 <- randomForest(death ~ ., data = hearts_numerics,
                                     mtry = 5, ntree = 1000,
                     importance = T, proximity = T)
```

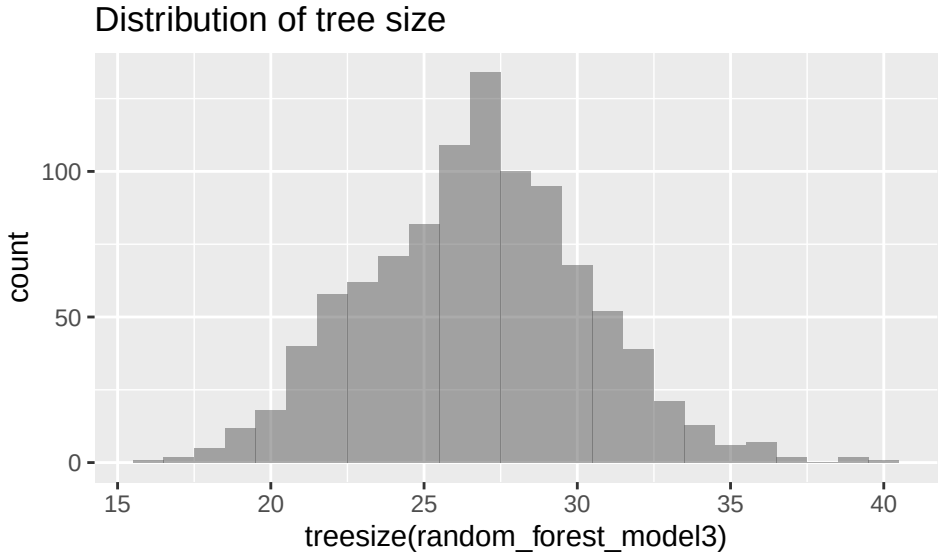
```
# Model output
random_forest_model3
```

```
Call:
 randomForest(formula = death ~ ., data = hearts_numerics, mtry = 5,      ntree = 1000, importance = T,
               Type of random forest: classification
                     Number of trees: 1000
No. of variables tried at each split: 5

        OOB estimate of  error rate: 15.55%
Confusion matrix:
      false true class.error
false   153   16  0.09467456
true     21   48  0.30434783
```

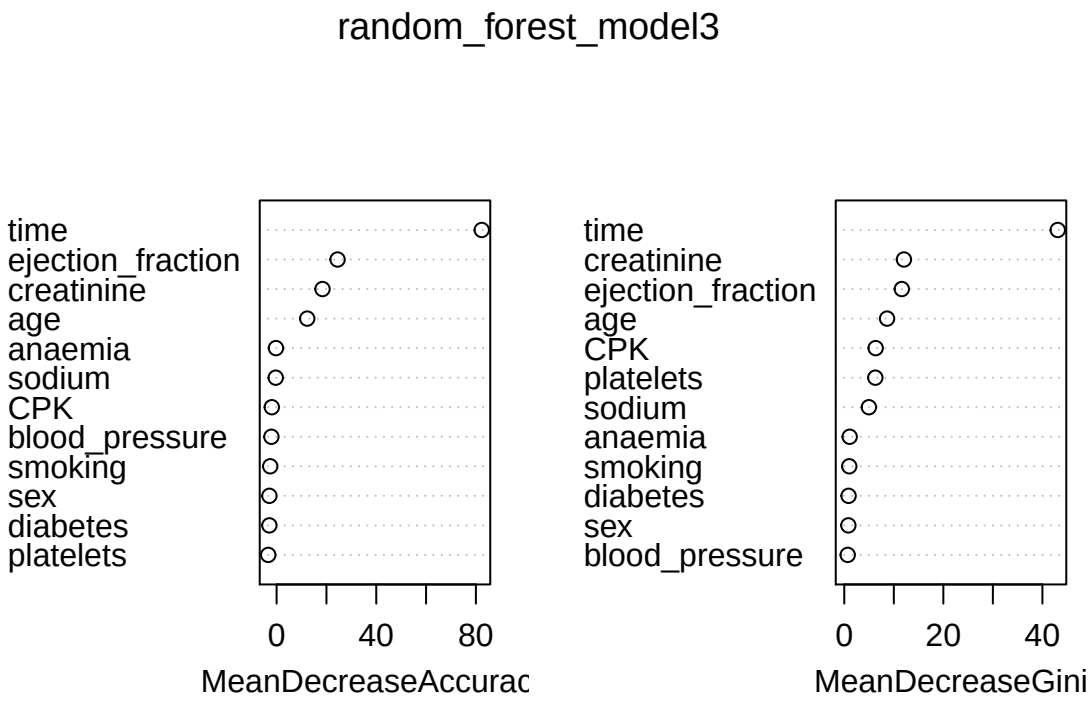
```
# Confusion matrix
table(hearts_numerics$death, predict(random_forest_model3,
                                     hearts_numerics))
```

```
       false true
  false   169    0
  true      0   69
```

```
# Distribution of tree size
gf_histogram(~ treesize(random_forest_model3)) +
  labs(title = "Distribution of tree size")
```




```
# Accuracy and Gini plots
varImpPlot(random_forest_model3)
```



For the third random forests model, the OOB estimate is 15.55%, indicating that TER = 15.55%. There are no misclassified observations in the training set, so AER = 0%.

An accuracy plot for model 3 shows that time is highest in importance, and that ejection fraction, creatinine, and age are also somewhat important. A Gini plot indicates that time is highest in importance, and that creatinine, ejection fraction, and age are somewhat important. The Gini plot also indicates that CPK, platelets, and sodium are of some importance as well, but notably, all of the categorical variables are of little to no importance within the model. Also note that the distribution of tree size is symmetric and appears to be roughly normal.

**Model 4:**

For my fourth and final random forests model, I will specify `mtry = 9` and `ntree = 500`.

```
# Set seed
set.seed(75)

# Create model
random_forest_model4 <- randomForest(death ~ ., data = hearts_numerics,
                                     mtry = 9, ntree = 500,
                       importance = T, proximity = T)
```

```
# Model output
random_forest_model4

# Confusion matrix
table(hearts_numerics$death, predict(random_forest_model4,
                                              hearts_numerics))

# Distribution of tree size
gf_histogram(~ treesize(random_forest_model4))

# Accuracy and Gini plots
varImpPlot(random_forest_model4)
```

In the fourth random forests model, the TER = 17.23%, and AER = 0% (for the sake of brevity, neither the model output, confusion matrix, histogram of the distribution of tree size, nor accuracy and Gini plots are displayed).

**Which random forests model is best?**

In this section, I fit four random forests models, changing the specifications for `mtry` (which refers to the number of variables randomly sampled as candidates at each split) and `ntree` (which specifies the number of trees generated) for each of the models. Model 1 randomly sampled 3 variables at each split (`mtry = 3`) and made 500 trees (`ntree = 500`). Model 2 randomly sampled 4 variables at each split (`mtry = 4`) and made 100 trees (`ntree = 100`). Model 3 randomly sampled 5 variables at each split (`mtry = 5`) and made 1000 trees (`ntree = 1000`). Model 4 randomly sampled 9 variables at each split (`mtry = 9`) and made 500 trees (`ntree = 500`).

In order to identify which random forests model is the best, I will look at the AER's and TER's for all of the models. All four of the models had AER's of 0%, indicating that each of the models performed perfectly on the training data. Since all of the models had the same AER, we will look at the TER's for the four models in order to decide upon the best model. The TER's for the four models were as follows: model 1: TER = 15.97%; model 2: TER = 16.38%; model 3: TER = 15.55%, model 4: TER = 17.23%. We see that model 3 had the lowest TER out of the four models. Because model 3 had the lowest TER, I will choose model 3 as the best random forests model, since it will be the best model to generalize to unseen data.

**Identifying best classification model**

I chose to create classification models using two classification methods - classification trees, and random forests. Of the three classification trees that I made, the best model was model 3, which was constructed with `minsplit = 10`, `minbucket = 3`, and `xval = 10`. This model yielded an AER of 8.4% and a TER of 18.9%. Of the four random forest models that I made, the best model was model 3, which had an AER of 0% and a TER of 15.55%. I see that the best random forest model outperforms the best classification model in both AER and TER:

- $\text{AER}_{randomforests} = 0\% < \text{AER}_{classification} = 8.4\%$
- $\text{TER}_{randomforests} = 15.55\% < \text{TER}_{classification} = 18.9\%$

Consequently, I will say that my best classification model is the random forests model that I made with `mtry = 5` and `ntree = 1000`.

**Takeaways**

The objective with classification is to use explanatory variables to correctly predict (classify) an outcome of observations. I chose to use classification because I wanted to identify, for those with heart failure, what important risk factors for dying were. The best classification model came from performing random forests. Unfortunately, random forests models are incredibly hard to interpret. However, there are some key takeaways that I can note from my model:

1) The model had a 15.55% misclassification rate on test data
    - This means that this model, if used on another data set, would make an incorrect prediction about an individual dying about 15.55% of the time.
    - We would like to have a more accurate model than this, especially since we are predicting the death of a patient.
2) The accuracy plot and the Gini plot both indicated that time is highest in importance in predicting survival
    - Recall from the preliminary analysis that the median follow up time for those who died is far less than the median follow up time for those who survived. However, learning that the follow up time (the time before a patient's next appointment) can help predict their death isn't ultimately that helpful, as it fails to point to a characteristic or condition of the patient that could be useful in predicting their death.
3) The accuracy plot and Gini plot indicate that besides time, the next most important variables are ejection fraction, creatinine (serum creatinine), and age
    - Note that Chicco and Jurman (2020) identified serum creatinine and ejection fraction as the two most relevant features through performing feature ranking.
    - Note that Chicco and Jurman (2020) also identified that machine learning can predict patient survival using just serum creatinine and ejection fraction (hence the title of their paper, "Machine learning can predict survival of patients with heart failure from serum creatinine and ejection fraction alone").
    - In short, the Chicco and Jurman (2020) team identified serum creatinine and ejection fraction as two important variables in the prediction process, which is a finding I also came across through my analyses. The fact that they and I arrived at similar results despite running different analyses suggests that these variables really are of great importance to predicting survival.

# Conclusion

### Review

In this report, I set out to identify:

1) Are there subgroups among individuals who have heart failure?

2) How we can predict survival of patients with heart failure?

To answer the first question, I performed cluster analysis. I used agglomerative hierarchical clustering, using both Ward's method and complete linkages, as well as K-means clustering. The best clustering model that I identified had a silhouette coefficient of 0.14, indicating that the model had no substantial structure.

Since my clustering solution was not particularly impressive, I did not perform classification on the clusters that arose in that solution. Instead, I performed classification on the entire data set, which helped me address the second question. I created 3 models using classification trees and 4 models using random forests, and ultimately, the best model that I came up with was a random forests model made by specifying `mtry = 5` and `ntree = 1000`. This model had an AER of 8.4% and a TER of 15.55%.

It is difficult to interpret this model, but it is worth noting that accuracy and Gini plots indicated that time (follow up time) was the most important predictor, followed by ejection fraction, serum creatinine, and age. I mentioned that Chicco and Jurman (2020) identified ejection fraction and serum creatinine as the two most relevant features in feature ranking, and that they showed how machine learning could predict survival using these two variables alone. The fact that they and I arrived at results that point to ejection fraction and sodium creatinine being important predictors of survival, despite running different methods, is important to note.

### Limitations and future directions

A limitation of this report is that I failed to arrive at a cluster solution that had a strong structure, so I was unable to run classification on specific clusters. A future study could focus on running more clustering models, which could include using additional clustering methods such as model-based methods, in order to generate a clustering solution with high cluster validity.

Another limitation of this report is that I only ran two classification methods - classification trees and random forests - and ended up choosing my best classification model out of only 7 models. Making more models and drawing upon additional classification methods would help with identifying the most robust model possible for predicting survival.

A final limitation is that I did not perform any analyses with the time variable removed. As mentioned in the classification takeaways subsection, including time as a predictor of survival isn't altogether that helpful since the time variable doesn't reflect the presence of conditions or a characteristic of an individual that could be useful in predicting their survival/death.

### A final note

Future investigation into research for predicting survival of patients with heart failure is certainly merited. Scientists have identified risk factors for developing heart failure, but lack of knowledge about predicting survival for those who already have heart failure. This lack of knowledge presents difficulties to physicians and patients alike, and advancements in insights about predicting survival could improve both medicine and the lives of patients.

# References

Ahmad, T., Munir, A., Bhatti, S. H., Aftab, M., & Raza, M. A. (2017). Survival analysis of heart failure patients: A case study. PLOS ONE, 12(7), e0181001. https://doi.org/10.1371/journal.pone.0181001

Ahman, T., Munir, A., Bhatti, S., Aftab, M., & Ali Raza, M. (2017). Survival analysis of heart failure patients: A case study. Dataset. [Data set]. PLOS ONE. https://doi.org/10.1371/journal.pone.0181001.s001

CDC. (2020, September 8). Heart Failure. Centers for Disease Control and Prevention. https://www.cdc.gov/heartdisease/heart_failure.htm

Chicco, D., & Jurman, G. (2020). Machine learning can predict survival of patients with heart failure from serum creatinine and ejection fraction alone. BMC Medical Informatics and Decision Making, 20(1), 16. https://doi.org/10.1186/s12911-020-1023-5

Kaggle. (n.d.). Heart Failure Prediction. Retrieved October 24, 2020, from https://kaggle.com/andrewmvd/heart-failure-clinical-data

Savarese, G., & Lund, L. H. (2017). Global Public Health Burden of Heart Failure. Cardiac Failure Review, 3(1), 7–11. https://doi.org/10.15420/cfr.2016:25:2