Norwegian University of Life Sciences

Master's Thesis 2018 60 ECTS

Norwegian University of Life Sciences
Faculty of Chemistry, Biotechnology and Food Science

On the Application of Machine Learning Techniques for Phenotypic Classification and Prediction of Heart Failure

Samir Adrik
Master of Science in Bioinformatics and Applied Statistics

On the Application of Machine Learning Techniques for Phenotypic Classification and Prediction of Heart Failure

Samir Adrik

Thesis submitted for the degree of
Master of Science in Bioinformatics and Applied Statistics

Norwegian University of Life Sciences

May 18, 2016

Abstract

Acknowledgments

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Heart failure (HF) is a clinical syndrome typically associated with high prevalence, high mortality, frequent hospitalization and overall reduced quality of life (QoL). Approximately 65 million people are effected by HF globally (Hay et al., 2017). With an aging population, it is expected that the prevalence of HF is to increase. In developed countries, about 3-5% of hospital admissions are linked with HF, accounting for about 2% of the total health cost (Tripoliti et al., 2017). It is not unusual for HF to be characterized as a global pandemic with prognosis being worse than that of most cancers, see e.g. Braunwald (2015) and Savarese and Lund (2017).

In terms of clinical classification, there is no single ”universally agreed upon” system for classifying the causes of HF. Typically HF manifests it self as at least two major subtypes (Alonso-Betanzos et al., 2015). All being commonly distinguished based on measures of the left ventricle ejection fraction (LVEF)[1]. The first subtype encompasses patients with LVEF values larger or equal to 50%. These patients are characterized as patients with HE with preserved ejection fraction (HEpEF). The second subtype includes patients with LVEF values less than 40%, and are characterized as patients with HE with reduced ejection fraction (HErEF). However, the European Society of Cardiology (ESC) recently defined a third subtype with patients belong to the ”gray zone” or the ”the middle child”, namely when the LVEF

[1]Fraction of blood ejected from the left ventricle of the heart with each contraction. Calculated as the left ventricle stroke volume (LVSV) divided by the left ventricle end-diastolic volume (LVEDV), i.e. $LVEF = LVSV/LVEDS$ (Cikes and Solomon, 2015)

values lies between 40% and 49%[2]. These patients are defined as having HF with mid-range ejection fraction (HFmrEF), see e.g. Lam and Solomon (2014) and Ponikowski et al. (2016). Clinically clustering patients according to HF subtypes and identifying HF patients most at risk of mortality and readmission is something that remains challenging. Especially considering that the 1-year mortality rates for acute HF across different regions in Europa ranges from 21.6% to 36.5% (35.1% - 37.5% in the US), see e.g. Cheng et al. (2014), Inamdar and Inamdar (2016) and Crespo-Leiro et al. (2016). Patients with HFmrEF have also a clinical profile and prognosis that is close to those of HFpEF (which have LVEF values considered to be normal). Current therapies have also shown to be unable to reduce *both* morbidity and mortality in patients with HFmrEF and HFpEF, see e.g. Ponikowski et al. (2016) and Hsu et al. (2017). All of which makes the overall job of identifying and distinguishing these patients challenging. It is also unknown if improving phenotypic classification is clinically useful or even possible (Shah et al., 2014).

Nonetheless, the rapid increase in available medical data on patients has lead to machine learning (ML) techniques gaining widespread attention by researchers. The application of such techniques is one that *may* offer an opportunity to build better management strategies, as well as early detection and better prediction of adverse effects associated with HF. Of the ML techniques gaining most attention, one typically finds *clustering* and *classification* methods being intensely studied. Accordingly, the use of ML techniques to identify distinct patient groups with *post-diagnosed* HFmrEF and HFpEF most at risk of mortality and readmission, is one we will try to examine to it full potential.

1.1 Problem statement

In this thesis, we investigate how well various clustering methods (hierarchical clustering, K-Means and expectation–maximization) perform in producing phenotypically distinct clinical patient groups (i.e. phenomapping) with HFpEF and HFmrEF. Furthermore, we also evaluate the performance of various classification algorithms (K-nearest neighbours, linear

[2]The American College of Cardiology Foundation/American Heart Association (ACCF/AHA) were the first to define HF with borderline ejection fraction as being patients with LVEF values between **41%** to 49% (Yancy et al., 2013).

discriminant analysis, support vector machines and random forest) in predicting the clinical outcomes (mortality and re-admission) of patients with HFpEF and HFmrEF. When evaluating the results, we compare the clustered according to their level Homogeneity, i.e. the number of statistically significant baseline characteristics within each group and rank methods accordingly. For the classification we evaluate the forecasting estimations based on various statistical properties (sensitivity, specificity and accuracy) and validate with k-fold cross-validation in order to rank methods accordingly. All the models and techniques are applied on a data set consisting of 375 patients with symptomatic HF identified at a tertiary hospital in the United Kingdom.

1.2 Thesis Structure

The thesis is divided into five chapters and proceeds as follows: The next chapter (2) reviews the literature on the application of ML techniques for the assessment of heart failure. This is done to put the proposed research in a relevant context. Chapter (3) details the methodology, including presenting the data and the quality of the data. Preliminary analysis on the data will also be done in this chapter. This includes evaluating and selecting a data set based on methods of imputation and dimensional reduction. Next, chapter (4) presents the results of the clustering comparisons with the prediction accuracy of the clinical outcomes, with conclusive remarks and discussion found in chapter (5). The source code and relevant statistical output can be found in the appendix.

Chapter 2

Background

The following chapter presents a thorough treatment on the literature of the application of ML techniques for the assessment of heart failure[1]. Important topics such as HF detection, subtype estimation and prediction of clinical outcomes in the context of ML will be presented and explained.

2.1 HF detection

The ESC defines HF as a clinical syndrome caused by structural and/or functional cardiac abnormality, resulting in a reduced cardiac output (CO) and/or elevated intracardiac pressures at rest or during stress. It is typically characterized by symptoms, such as breathlessness, ankle swelling and fatigue that may be accompanied by signs, such as elevated jugular venous pressure (JVP), pulmonary crackles and peripheral oedema (swelling in lower limbs) (Ponikowski et al., 2016). HF prevents the heart from fulfilling the circulatory demands from the body, due to its impairing abilities on the ventricles to maintain the bodies hemodynamics (blood flow). As there is no broad definitive industry accepted diagnostic test for HF, one finds in clinical practice that medical diagnosis is done with a combination of careful examinations (physical and historical) with assisting tests, such as blood tests, chest radiography (chest X-ray, CXR), electrocardiography (EKG) and echocardiography (cardiac echo), see e.g Henein (2010) and Son et al. (2012). As a result of this, several criteria for determining the presence of HF have

[1]We highly recommend reading Tripoliti et al. (2017) for a broader overview of the literature on the state-of-the-art ML techniques applied for the assessment of heart failure.

been proposed, including the Framingham criteria (McKee et al., 1971), the Boston criteria (Carlson et al., 1985), the Gothenburg criteria (Eriksson et al., 1987) and the ESC criteria (Swedberg et al., 2005) (Roger, 2010). All of which are much used.

In a non-acute onset, the ESC has also defined an algorithm for diagnosing HF (Ponikowski et al., 2016). The algorithm is structured in the following way: The probability of HF ($\hat{p}_{HF}$) is evaluated along three dimensions:

(i) **Prior clinical history**: History of coronary artery disease (CAD) or arterial hypertension, exposition to cardiotoxic drugs/ radiation, diuretic use (any substance that promotes the production of urine) or orthopnea (shortness of breath when lying down)

(ii) **Physical examination**: Crackles/rales, bilateral ankle oedema (swelling in both ankles), abnormal heart sounds/murmur, jugular venous dilatation, laterally displaced/broadened apical beat (pulse felt at the point of maximum impulse (PMI))

(iii) **Abnormalities in electrocardiography (EKG)**

If all elements along the three dimensions are normal/absent, $\hat{p}_{HF}$ is estimated to be highly unlikely. Should, however, at least one element be abnormal, then plasma Natriuretic Peptides (NP)[2] should be measured in order to identify patients who need echocardiography. Specifically, should the NP values be above the exclusion threshold[3] or should the assessment of NPs not be routinely done in clinical practice then patients need to be forwarded for an echocardiography. With the help of the cardiac echo, specialist can detect abnormalities in the heart rhythm. Should the results of the plasma NP or the echocardiography be normal[4], then HF is also considered unlikely. Should the results of the echo yield any abnormal results, appropriate HF treatment should be initiated. The structure of the

[2]A hormone, mainly secreted from the heart, that has important natriuretic and kaliuretic properties (excretion of sodium and potassium in the urine) (Pandit et al., 2011). In clinical practice it is found that brain NP (also called BNP) levels can be used to predict the risk of death and cardiovascular events (Wang et al., 2004).

[3]The recommended threshold levels are BNP levels $\geq 35pg/mL$ or NTproBNP levels $\geq 125pg/mL$, see e.g. Cowie et al. (1997), Yamamoto et al. (2000), Krishnaswamy et al. (2001), Zaphiriou et al. (2005), Fuat et al. (2006) and Maisel et al. (2008).

[4]Normal ventricular and atrial volumes and function (Aune et al., 2009).

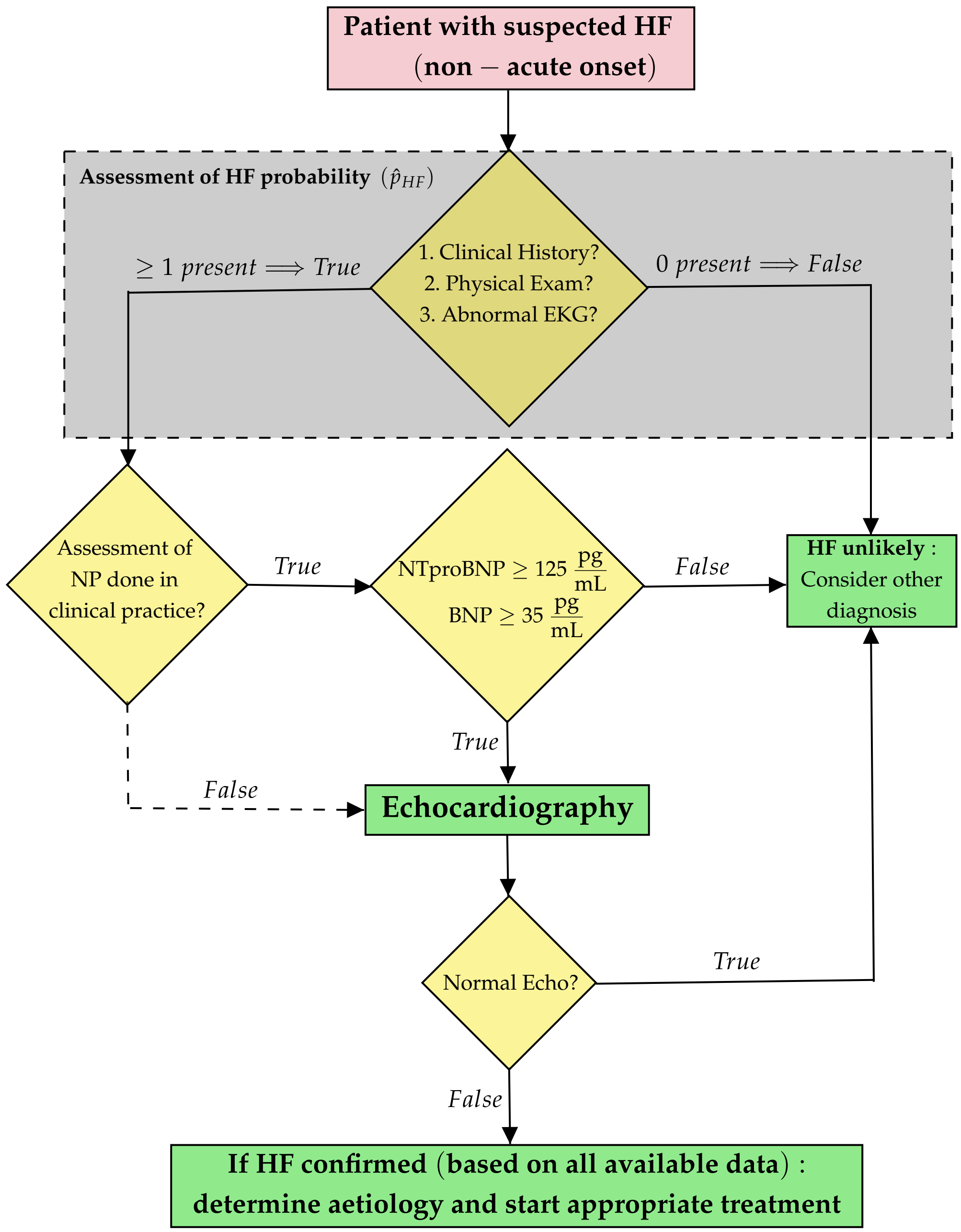

Figure 2.1: *ESC diagnostic algorithm for the diagnosis of heart failure of non-acute onset (Ponikowski et al., 2016, page. 2141).*

ESC algorithm is shown as a flow chart in figure (2.1). Being that the ESC algorithm is much used in clinical practice throughout the world, there is research that suggest that the medical and economic benefits of applying ML in the detection of HF should not be ignored. In the context of diagnosing patients with HF, the benefits typically include (i) less time consumption, (ii) more support (large global community of ML practitioners in business and academia) and (iii) same level of accuracy as conventional tools when applied on available data. Many ML methods used to detect HF as a statistical learning problem, fall in the category of *supervised* statistical learning (see section 2.2.1). The relevant ones include expressing the detection of HF as a two class classification problem, where the presence of HF is the output of the classifiers. Methods including logistic regression, linear discriminant analysis (LDA), Bayesian classifier, k-nearest neighbours (k-NN), random forests (RF), boosting, support vector machines (SVM) and neural networks (NN) are all very popular. As the response variable of the classification problem is categorical, most ML studies tend to use measures of heart rate variability (HRV)[5] as the main predictors for distinguishing patients as normal or with HF (Tripoliti et al., 2017). Other predictors include parameters from clinical tests (i.e. blood test, echo, EKG, chest radiography), clinical variables (e.g. gender, age, blood pressure, smoking habit) and other laboratory findings. Relevant

Table 2.1: Literature review of HF detection

Author	HRV?	Method	Data	Features	Evaluation
Masetic and Subasi (2016)	False	SVM, k-NN, NN, RF	$N = 28$ (13 normal and 15 HF)	Response: Normal & HF. Predictor: Features extracted by EKG.	SVM: Accuracy: 99.53% k-NN: Accuracy: 99.93% NN: Accuracy: 99.20% RF: Accuracy: 100.00% Validation: 10-fold cross validation

[5]HRV is the amount of heart rate fluctuations around the mean heart rate (van Ravenswaaij-Arts et al., 1993). The HRV can be assessed using R-waves produced by an EKG and reduced HRV is typically an established sign of HF (Ernst, 2016).

Table 2.1: Literature review of HF detection (*continued*)

Author	HRV?	Method	Data	Features	Evaluation
Liu et al. (2014)	True	SVM, k-NN	$N = 47$ (30 normal and 17 HF)	Response: Normal & HF. Predictor: Short term HRV measure (ST-HRV)	SVM: Accuracy: 100.00% Validation: Cross-validation
Narin et al. (2014)	True	SVM, k-NN, LDA, NN	$N = 83$ (54 normal and 29 HF)	Response: Normal & HF. Predictor: ST-HRV	SVM: Accuracy: 91.56% k-NN: Accuracy: 85.54% LDA: Accuracy: 85.54% NN: Accuracy: 89.15% Validation: Leave-one-ut cross validation.
Gharehchopogh and Khalifelu (2011)	False	NN	$N = 40$ (26 normal and 14 HF)	Response: Normal & HF. Predictor: Gender, age, blood pressure, smoking habits.	NN: Accuracy: 95.00% Validation: Testing set.
Yang et al. (2010)	False	Naive-Bayes, SVM, NNC	$N = 153$ (58 Normal, 30 HF-prone, 65 HF)	Response: Non-HF group (Health or HF-prone) & HF. Predictor: clinical test results	SVM: Accuracy: 74.40% Validation: Test set of $N = 90$ subjects

articles where one applies ML techniques to address the statistical learning problem of detecting patients with HF is shown in table (2.1). Some common evaluation measures used in such research include: sensitivity (true positive rate), specificity (true negative rate) and accuracy[6]. The accuracy is the only evaluation measure reported in table (2.1). We also need to emphasize that as this particular statistical learning problem (i.e. detection of HF) is one that falls outside of the scope of the problem statement mentioned in chapter (1). We will not be pursuing a further literature review of this problem. However, we highly recommend reading the likes of Tripoliti et al. (2017), Acharya et al. (2017) or Awan et al. (2018), for a more up-to-date overview of the literature on ML used for HF detection.

2.2 Subtype Estimation

According to the ESC algorithm (figure 2.1), once HF is confirmed and the probability of HF is assessed and estimated to be likely, the next step is to estimate the causes (aetiology) or subtypes of HF. The main definition of HF subtypes is based on historical research. Most of the research done after the 1990s emphasizes estimating the subtype of HF patients based on the measure of the left ventricle ejection fraction (LVEF). The two usual ways of obtaining the LVEF values are through an echocardiography or cardiac magnetic resonance imaging (CMR or cardiac MR) (Ponikowski et al., 2016). In prior guidelines presented by the ESC, HFrEV and HFpEF were the two main subtypes of HF (McMurray et al., 2012). The ESC did however acknowledge that a gray zone existed between the two. As a result of this a new subtype was introduced, namely HFmrEF. The ESC did so in hopes of stimulating research into the underlying characteristics, pathophysiology and treatment

Table 2.2: HF subtypes based on LVEF (Ponikowski et al., 2016, page. 2137)

Criteria	HFrEF	HFmrEF	HFpEF
1	Symptoms $\pm$ Signs	Symptoms $\pm$ Signs	Symptoms $\pm$ Signs
2	LVEF $< 40\%$	$40 \leq$ LVEF < 50	$50 \leq$ LVEF

[6]The fraction/proportion of true positives (*sensitivity*) or true negatives (*specificity*) correctly identified (James et al., 2013).

Table 2.2: HF subtypes based on LVEF (*continued*)

Criteria	HFrEF	HFmrEF	HFpEF
3	–	1. Elevated NP levels (fig 2.1) 2. At least one additional crit- eria: a) Relevant structural heart disease[7] b) Diastolic dysfunction[8]	1. Elevated NP levels (fig 2.1) 2. At least one additional crit- eria: a) Relevant structural heart disease b) Diastolic dysfunction

of this group of patients (Ponikowski et al., 2016). Details about the criteria for the various HF subtypes are shown in table (2.2). The differences between HFmrEF and HFpEF are difficult to distinguish. As mentioned earlier, these two groups were previously only classified as HFpEF. Diagnosing HFpEF is a very complex process with the diagnosis of chronic HEpEF being especially cumbersome in elderly patients with one or more additional diseases (comorbidity). With the exception of the LVEF values, signs and symptoms between HFmrEF and HFpEF are often non-specific and do not discriminate between other clinical conditions. LVEF $\geq$ 50% is also considered to be normal. The ECS has also underlined the difficulties with an emphasis on the LVEF as the main discriminant between HFmrEF and HFpEF. The cut-off at 50% is set arbitrary and in clinical trials patients with LVEF between 40% and 49% are often classified as HFpEF, see e.g. Kelly et al. (2015) and Ponikowski et al. (2016). The ESC places an emphasis on additional objective measures of cardiac dysfunction in order to sufficiently discriminate the two subtypes, but currently no gold standard exists. The hope of stimulating more research into the characteristics of the patient group HFmrEF has fuelled much research into the application of ML, to further advance the literature. The appeal from the ESC into further re-

[7] *Left ventricular hypertrophy* (LVH): Thickening of the heart muscle of the left ventricle of the heart and/or *Left atrial enlargement* (LAE): Enlargement of the left atrium (LA) of the heart (Nagueh et al., 2009)

[8] Increased resistance to diastolic filling of one or both cardiac ventricles. In addition to structural abnormalities, physiological derangement of myocardial inactivation and relaxation (Grossman, 1990).

search has also served as a motivation for much of the research done. There has been many approaches to researching this question. Accordingly, we have organized the literature review into two parts and have structured the literature based on the statistical learning problem category, i.e. supervised or unsupervised learning.

2.2.1 Supervised Learning

In this thesis we use the terms *machine learning* (ML) and *statistical learning* (SL) interchangeably. Even though the two are very closely linked, they do differ in terms of emphasis and terminology. ML is defined as *"a set of methods that can automatically detect patterns in data, and then use the uncovered patterns to predict future data, or to perform other kinds of decision making under uncertainty"* (Murphy, 2012). SL on the other hand is often considered to be the statistical framework of ML, and emphasize the importance of building *probabilistic* models for the analysis and prediction of data in order to draw inference, see e.g. Friedman et al. (2009), Murphy (2012), James et al. (2013) and Wasserman (2013). Individuals of both camps (i.e. computer scientists and statisticians) often use different language for the same thing. In this thesis we refer to the underlying learning problem to be solved by a given algorithm as a statistical learning problem. The actual algorithms used to solve the SL problem are referred to as ML methods/algorithms[9]. This is done to hopefully reduce confusion for the readers.

Most SL problems fall into one of two main categories, i.e. *supervised* and *unsupervised* learning, see e.g. Friedman et al. (2009) and James et al. (2013)[10]. The example of detecting HF we discussed in section (2.1) is typically a learning problem that falls into the supervised learning domain. For each predictor(s) (input(s) or independent variable(s)) x_i, $i = 1, \dots, n$ there is an associated response (output or dependent variable), y_i. The objective of supervised learning is to fit a model that relates the response (y_i) to the predictors (x_i) (James et al., 2013). Supervised learning is the most common category of SL problem in practice. Of the ML methods most used to solve supervised SL problems, one typically mentions *classification*. The goal of classification is to learn a mapping from the predictors (x_i) to the response

[9]We need to emphasize that the methods can also be called statistical learning methods/algorithms as they are often done so in the literature.

[10]The categories are also referred to as the two main types of ML, see e.g. Murphy (2012)

(y_i), where $y \in \{1, \ldots, C\}$, with C being the number of classes. We can formalize classification as a SL problem by referring it to as a functional approximation problem. We assume a functional form $y = f(\mathbf{x})$ exists for some unknown function f, and the goal of the learning process is to estimate f given a training set with labeled and known values. We can then use the estimated function $\hat{y} = \hat{f}(\mathbf{x})$ to make predictions on a testing / validation set (Murphy, 2012).

The application of classification to estimate HF subtypes is a relatively new approach. HF subtype estimation using ML in earlier research have similarities with HF detection. Both subjects reduce the classification problem to a two class classification problem with the assumption that the predicted responses are mutually exclusive. As $C = 2$, one often call this a *binary classification* problem. In which case one often assumes that $y \in \{0, 1\}$ (Murphy, 2012). Prior to the ESC introduction of HFmrEF as a third subtype of HF, most ML research focused on classifying HF patients according to the two common subtypes, i.e. HFrEF and HFpEF. A list of some relevant literature can be found in table (2.3). Most predictors have features including measures of demographic characteristics, HRV, signs and symptoms, vital signs, results of laboratory investigations and previous medical history. Methods include bagging, boosting, random forest, support vector

Table 2.3: Literature review of HF subtype classification

Author	Method	Data	Features	Evaluation
Austin et al. (2013)	Bagging, Boosting, RF, SVM	$N = 8212$ (3697 for training, 4515 for testing)	Response: HFrEF & HFpEF. Predictor: Demographics, vital signs, symptoms, lab investigation and prev. history.	Bagging: Sensitivity: 45.1% Specificity: 84.9% Boosting: Sensitivity: 87.6% Specificity: 45.3% Random Forest: Sensitivity: 37.8% Specificity: 89.7% SVM: Sensitivity: 40.1% Specificity: 88.7% Validation: Testing set of 8339 subjects

Table 2.3: Literature review of HF subtype classification (*continued*)

Author	Method	Data	Features	Evaluation
Alonso-Betanzos et al. (2015)	Naive-Bayes, SVM, NNC	$N = 111$ (48 for training, 63 Monte Carlo simulated instances for testing)	Response: HFrEF & HFpEF. Predictor: End-systolic Volume Index.	Naive-Bayes: Train error: 4.14% Test error: 9.52% SVM: Train error: 2.08% Test error: 4.76% NNC (ib1, see Aha et al. (1991)): Train error: 2.08% Test error: 4.76% Validation: Testing set of 63 instances. 10-fold cross validation.
Isler (2016)	k-NN, NN	$N = 30$ (18 with HFrEF & 12 with HFpEF)	Response: HFrEF & HFpEF. Predictor: Short term HRV measures	k-NN: Sensitivity: 87.5% Specificity: 91.07% Accuracy: 89.29% NN: Sensitivity: 93.75% Specificity: 100.00% Accuracy: 96.43% Validation: Leave-one-out cross-validation.

machines (SVM), Naive-Bayes, nearest neighbour classifiers (NNC), k-nearest neighbours (k-NN) and neural networks (NN). As classification methods are much used in the literature of HF subtype estimation, we reserve the use of these methods in a later section dealing with the prediction of clinical outcomes (see section 2.3). Supervised learning methods also assume a priori that there exists a response y_i with a predefined number of classes (C). Because of this we feel that such an application to the problem of HF subtype estimation would fall outside the scope of the problem statement mentioned in chapter (1). One of the main motivations of this thesis is to investigate how well it is possible to produce phenotypically distinct clinical patient groups using dense phentoypic data (also known as phenomapping). Given the motivation, we seek to better understand a possible relationship between patient groups by placing an assumption of no response variable to supervise our analysis. To answer this question, we turn to the second main category of SL problems, namely unsupervised learning.

2.2.2 Unsupervised Learning

The main goal of unsupervised learning is to discover hidden structures in the data that are not predefined. Sometimes its also refereed to as *knowledge discovery* and is widely used, as it is arguably more typical for animal and human learning. The formalization of unsupervised learning is often done in the setting of *unconditional density estimation*, i.e. we want to build models of the form $p(\mathbf{x}_i|\theta)$. Instead of a conditional setting as done with supervised learning, i.e. $p(y_i|\mathbf{x}_i, \theta)$, the use of unsupervised learning is often considered to be more ”convenient” than supervised learning, as it does not require an expert to manually label all the data (Murphy, 2012). This convenience is often stated as a major reason for the relevance of unsupervised learning done for distinguishing phenotypical characteristics between HF patient groups. Not to mention that there is no agree-upon measure of what distinguishes HF subtypes (see section 2.2). Furthermore, because of the complex nature and high degree of heterogeneity of HF subtypes such as HFpEF, the sole use of genetic information for helping to *precisely* classify HF subtypes has often been seen as unlikely. Uncertain behavior by weak genetic factors is very probable in eliciting disease phenotypes (Deo, 2015). This additional complexity is avoided by framing the SL problem in the setting of unsupervised learning.

A lot of research has been conducted using unsupervised learning to group HF patients into subtypes with phenotypically distinct characteristics. Of the ML methods most used here, one typically finds *clustering* methods. These methods are designed to find subgroups or *clusters* within a data set. The goal of clustering is to partition the data set into distinct groups with high degree of homogeneity and arranging the clusters into a natural hierarchy (Friedman et al., 2009). A list of the newest literature on the application of clustering methods for phenomapping of HF patients is shown in table (2.4). Of the clustering methods found here, one can men-

Table 2.4: Literature review of HF subtype clustering

Author	Method	Data	Features	Results
Shah et al. (2014)	Hierarchical, model-based clustering	$N = 397$ with HFpEF	67 continuous clinical variables	The analysis revealed 3 distinct pheno-groups.
Ahmad et al. (2014)	Hierarchical clustering (Ward's minimum variance method)	$N = 2331$ (1619 incl., 712 excl.)	45 baseline clinical variables	Four clusters were identified whose patients varied considerably along measures of age, sex, race, symptoms, comorbidities, HF etiology, socio-economic status, quality of life, cardiopulmonary exercise testing parameters, and biomarker levels.
Alonso-Betanzos et al. (2015)	k-Means clustering, EM, SIBA.	3 Data sets: D1: $N = 48$ (13 HFrEF, 35 HFpEF) D2: $n = 63$ (29 HFrEF, 34 HFpEF) D3: $N = 403$ (137 HFrEF, 150 HFpEF)	End-systolic Volume Index, End-diastolic volume index	Algorithms generated dividing patterns
Kao et al. (2015)	Latent class analysis (LCA)	$N = 4113$ with HFpEF	11 prospectively selected clinical features	Identified 6 subgroups of HFpEF patients with significant differences in event-free survival.

Table 2.4: Literature review of HF subtype classification (*continued*)

Author	Method	Data	Features	Results
Ahmad et al. (2016)	Hierarchical clustering (Ward's minimum variance method)	$N = 433$ (172 incl.)	29 baseline clinical variables	Four advanced HF clusters were identified. The analysis was done on patients diagnosed with acute decompensated heart failure (ADHF).
Katz et al. (2017)	Hierarchical clustering, model-based clustering	$N = 1273$	47 continuous clinical variables	Identified 2 distinct groups that differed markedly in clinical characteristics, cardiac structure /function, and indices of cardiac mechanics.

tion hierarchical, K-Means, model-based clustering, such as expectation maximization (EM), sequential information bottleneck algorithm (SIBA) and latent class analysis (LCA). Addressing phenomapping within an unsupervised setting was started with Ahmad et al. (2014) and Shah et al. (2014). The latter employed the use of hierarchical and penalizing model-based clustering to distinguish HFpEF patients. The analysis was done on 67 continuous variables including clinical, laboratory, electrocardiography and echocardiography features. The results suggest that HFpEF patients can be clustered into three distinct pheno-groups with meaningful, clinically relevant categories.

Ahmad et al. (2014) did a similar analysis using 45 baseline clinical variables on a much larger data set consisting of 1619 patients with chronic HF (i.e. both HFrEF and HFpEF). The study identified four clusters of patients which varied considerably along measures of demographics, symptoms and comorbidities. The study underscored the high degree of disease heterogeneity that exists within chronic HF patients and the need for improved phenotyping of the syndrome. Alonso-Betanzos et al. (2015) used a somewhat different approach to phenomapping HF patient groups. Their objective was to use ML techniques to discriminate between patients with preserved EF and those with reduced EF using the concept of the Volume Regulation Graph (VRG)[11]. The authors evaluated three clustering methods (i.e. K-Means, EM and SIBA) and found that the algorithms generated dividing patterns. Kao et al. (2015) used latent class analysis (LCA) on a data set of 4113 HFpEF patients along 11 prospectively selected clinical features. The use of LCA is in many ways different than other clustering algorithms as it does not require continuous variables. It is optimized for analyzing categorical variables and identifies clusters based several traits rather than a single trait. With the use of LCA the authors identified 6 subgroups of HFpEF patients with significant differences in event-free survival. Other authors like Katz et al. (2017) and Ahmad et al. (2016) have organized their research along different phenomapping objectives. The latter addressed phenomapping on patients diagnosed with acute decompensated heart failure (ADHF), and Katz et al. (2017) on the systemic hypertensive patients with myocardial substrate (i.e. abnormal cardiac mechanics). As the two studies have a different phenomapping objective than the ones mentioned earlier, they still managed to identify four and two respective patient groups with acute ADHF and systemic hypertension with myocardial substrate.

The number of studies done on phenomapping HF patients is eminence and as evident from table (2.4), the results vary considerably with respect to the optimal number of clusters. This is something that this thesis will try to address by re-evaluating a number of the clustering methods used in the literature, but along a single phenomapping objective. Before that time, we move on to reviewing the literature associated with the second objective of the problem statement, namely predicting clinical outcomes due to HF.

2.3 Prediction of Clinical Outcomes

As we mentioned in chapter (1), HF is a syndrome that globally effects approximately 65 million people (Hay et al., 2017). In addition to the high prevalence and overall reduced quality of life (QoL), one cannot but mention the many serious clinical outcomes. This includes, but is not limited

[11] A graph of ESV versus EDV, which has the clear advantage of yielding (nearly perfect) linear relationships (Beringer and Kerkhof, 1998).

to mortality, morbidity, destabilization and re-admission. These outcomes effect not only the patients and their families, but also the society. The patients and their families are effected by the many constraints that HF places on family life and an overall reduction in QoL. With the emotional dimensions of health being more important than the physical dimensions (Dunderdale et al., 2005), the society is effected by the many economic constraints, such as an increase in the burden and cost of national health care expenditures. With the main driver of costs related to HF being that of hospitalization, where about 60-70% of HF costs are related to inpatient case and almost 20% to primary care (Braunwald, 2015). The use of prognostics can assist in the monitoring and treatment of HF patients, with the goal of improving the quality of care and the outcomes of patients hospitalized with HF (Tripoliti et al., 2017).

Conducting good prognostics is often conditional on estimating the severity of HF of a given patient. Accordingly, the two most used classification systems for the severity estimation, is the New York Heart Association (NYHA) Functional Classification (NYHA, 1994) and the American College of Cardiology/American Heart Association (ACC/AHA) stages of HF (Hunt et al., 2001). The NYHA system places the patients in one of four categories based on how much they are limited during physical activity and is based on symptoms as well as physical activity. The ACC/AHA system on the other hand structures HF stages based on structural changes to the heart and symptoms. Both systems provide complementary information about the presence and severity of HF. The various stages and classes of the two systems are shown in figure (2.2). Being that the NYHA classification system is based on subjective evaluation, it has been criticized because of a lack of taking into account the variability that can occur within patient groups. Furthermore, with the ACC/AHA system there is no moving backwards to prior stages, i.e. ones a patient is assigned a HF stage. The patient can never again achieve a different prior stage. With the NYHA it's different as patients can move between classes relatively quickly, as these are all based on symptoms alone, see Fleg et al. (2000) and Yancy et al. (2013). Most studies address HF severity estimation by expressing the statistical learning problem as a two or three class classification problem. The use of ML to address this particular SL problem will not be pursued, as the focus will be on the second objective of the problem statement, namely the prediction of clinical outcomes. The use of severity estimation is very important as it serves as complementary information for medical practi-

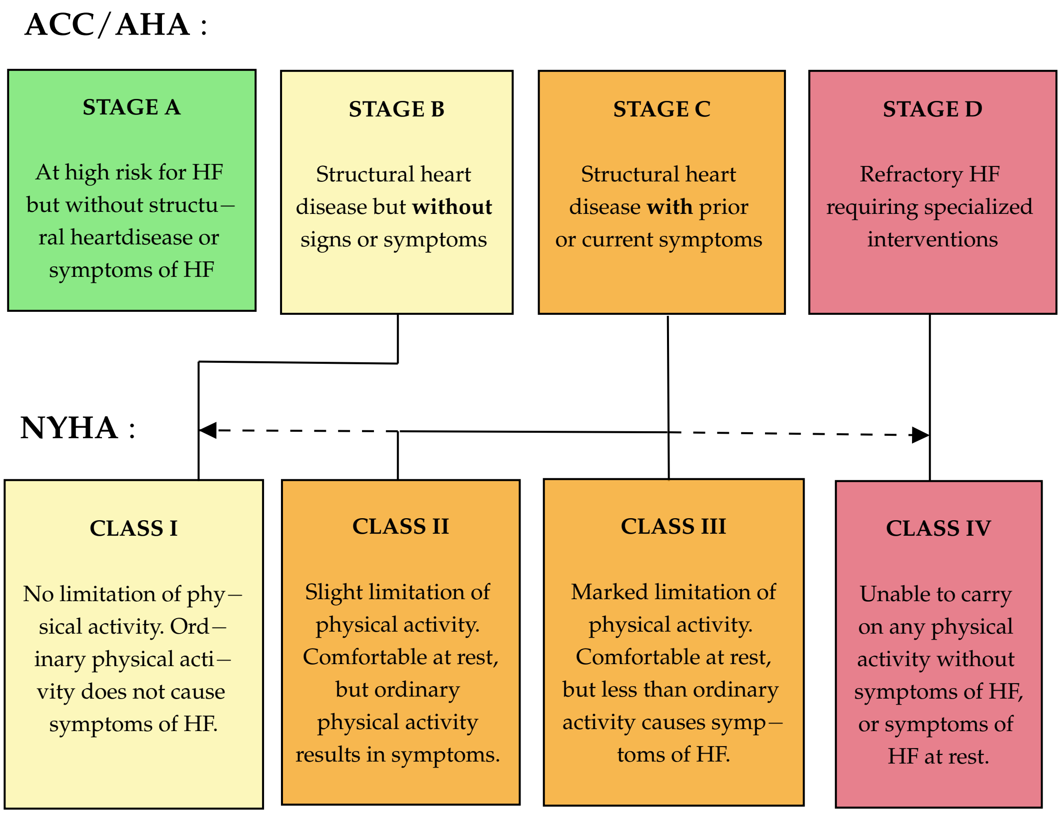

Figure 2.2: *Comparison of ACCF/AHA Stages of HF and NYHA Functional Classifications (Yancy et al., 2013, page. 1502).*

tioners to give objective prognostics about HF patients. A lot of studies have been conducted on the use ML to estimate HF severity, and again we recommend reading Tripoliti et al. (2017) for a further overview of the literature. As for the prediction of clinical outcomes it's especially re-admission and mortality that has gained a lot of interest by researchers. Re-admission is important because of the negative impact on healtcare systems' budgets. Mortality is obviously important as HF is one of the leading causes of death worldwide. The use of prediction models for mortality can benefit both physicians and patients. The literature is full of models taking into account various factors in producing statistical models that have the objective of predicting mortality. Some of the most used statistical methods include the Kaplan-Meier estimator (Kaplan and Meier, 1958) and multiple variable

Cox proportional hazard models (Cox, 1972). All of which have lead to the formation of multiple scores that estimate the risk of mortality that are much used in clinical practice. Examples include: the Enhanced Feedback for Effective Cardiac Treatment (EFFECT) Score (Lee et al., 2003), Seattle Heart Failure Model (Levy et al., 2006), Get With the Guidelines (GWTG) Score (Peterson et al., 2010) and Heart Failure Survival Score (Ketchum and Levy, 2011). A small list of the relevant literature related to the application

Table 2.5: Literature review of prediction of HF outcomes

Author	Outcome	Method	Data	Features	Evaluation
Austin et al. (2012)	Mortality	Logistic regression Logistic, Bagged and Boosted trees. Random Forrest	Baseline: $N = 9945$ (8240 incl.) Followup: $N = 8339$ (7608 incl.)	Response: Whether 30-day death in hospital Predictors: 34 clinical variables	Logistic regression: (Splines) AUC: 0.786 R^2: 0.203 Brier's score: 0.119 Boosted regression: (depth four) AUC: 0.777 R^2: 0.180 Brier's score: 0.107 Validation: Follow-up sample used as validation.
Zolfaghar et al. (2013)	Re-hosp-italization	Logistic regression Random Forrest	No. of data: 1681562.	Response: 30-day risk of re-admission. Yes or No Predictor: more than 100 featur-es	Logistic regression: Accuracy: 78.03% Random Forest: Accuracy: 87.12% Validation: 70% training 30% testing
Shah et al. (2014)	Mortality & Re-hos-pitaliza-tion	SVM	$N = 397$ with HFpEF	Response: mortality and re-admission: Yes or No. Predictor: 67 features	Mortality: Precision: 60.90% Re-hospitalization: Precision: 63.60%

Table 2.5: Literature review of prediction of HF outcomes (*continued*)

Author	Outcome	Method	Data	Features	Evaluation
Panahiazar et al. (2015)	Mortality	Logistic Regression Random Forest	$N = 5044$	Response: 1, 2 and 5 yr survival Predictor: 45 clinical variables	1-year: Log Regression: AUC: 81.00% Random Forest: AUC: 80.00% 2-year: Log Regression: AUC: 74.00% Random Forrest: AUC: 72.00% 5-year: Log Regression: AUC: 73.00% Random Forrest: AUC:72.00% Validation: Testing set of 3484 patients.
Koulaouzidis et al. (2016)	Re-hospitalization	Naive Bayes classifier	$N = 308$	Response: High or Low Risk of HF hospitalization Predictor: 25 clinical variables	Naive Bayes classifier: AUC: 82.00% Validation:10-fold-cross-validation

of ML for predicting re-admission and mortality is shown in table (2.5). One of the first to use ML methods for this particular SL problem was Austin et al. (2012). They investigated predicting the 30-day mortality using a binary variable to denote whether a patient died within 30 days of hospital admission. Methods used include: Logistic regression and trees, Bagged, Boosted trees and Random forest. The researchers used the methods on a total of 8240 baseline patients and 7608 follow-ups. The results seemed to suggest that Logistic Regression and Boosted Regression trees are the most accurate with area under the curve (AUC) of 0.786 and 0.777 respectively. Zolfaghar et al. (2013) applied logistic regression and random forest to predict 30 day risk of re-admission. This was done on a data set consisting of 1.681.562 patients. The predictors of the analysis contained more than 100 features. The accuracy was 78.03% and 87.12%, with 70% of the data set being reserved for training and 30% for testing. Shah et al. (2014) analyzed the prediction of both re-admission and mortality on 397 patients and 67 clinical variables using support vector machines (SVM). The precision of mortality and re-admission were 60.90% and 63.60%. As is evident from table (2.5), the accuracy and precision of the prediction models using ML methods varies throughout the various studies. Along with the variability in the number of optimal clusters mentioned in section (2.2.2), we'll also try to address this point by again re-evaluating the performance of a number of classification algorithm related to the SL problem of predicting clinical outcomes.

Chapter 3

Methodology

In this chapter, we present the methodology and research structure used in this thesis. Some pre-processing of data, including imputation and dimensional reduction, will also be presented and explained. The implementation of the ML algorithms that produce the results are also presented in this chapter.

3.1 Overview

As stated in chapter (1), the aim of the thesis is split into two parts. The first part is seeing how well various clustering methods perform in producing phenotypically distinct clinical patient groups with HFpEF and HFmrEF. We frame the SL problem in the setting of unsupervised learning and accordingly use the following clustering methods: hierarchical clustering, K-Means and expectation-maximization to evaluate which produce the most clinically useful patient groups. The use of these clustering methods are common in the literature (see section 2.2.2) and serves as the main motivation for including them in our analysis. The second part of the problem statement looks at evaluating the accuracy of various classification algorithms in predicting the mortality and re-admission of patients with HFpEF and HFmrEF. In accordance with the literature as presented in section (2.3), we reduce the SL problem of predicting the mortality and re-admission into a two class classification problem where both classes of outcomes are whether or not mortality/re-admission occurred. The classification methods that will be evaluated are k-nearest neighbours (k-

NN), support vector machines (SVM), linear discriminant analysis (LDA) and random forest. All the algorithms are much used in the literature. The motivation behind the use of the chosen algorithms, has always been to confirm the practices done in the literature. We do, however, need to emphasize that many algorithms exist that can be used to further broaden the analysis done in this thesis. We have not done this due to limitations.

The machine learning procedure adopted in this thesis is illustrated in figure (3.1). The procedure starts by pre-processing the data. This pre-processing step consists of three sub processes: consolidation, imputation and dimension reduction. The consolidation process merges the HFpEF and HFmrEF data set into one data set with the same types of variables. In addition to having one data set with all the observations, the process also leaves the data separate (but with equal variables), so that an analysis on each separate data set can be done. Furthermore, the clinical outcomes of the patients in the data set are extracted by this process and stored for later use in the classification part of the thesis. The imputation process does imputation of missing data to ensure that the data is balanced, and the dimensional reduction process addresses eventual problems with higher dimensional multi-correlated variables. The pre-processing steps are explained in further detail later in this chapter (see section 3.2). After the pre-processing is done, the procedure continues by first addressing the cluster analysis. Being that the dimension reduction is relevant for both the cluster analysis and classification, we use the components derived from the dimension reduction process as input into the clustering algorithms evaluated. We also compute new components when training the classification algorithms. The cluster analysis runs the produced components through the three cluster algorithms (hierarchical clustering, K-Means and expectation maximization). After the procedure is done, three sets of clusters are produced. The next step is to evaluate the clusters to assess their medical usefulness. On the other hand the supervised classification track is structured in a somewhat different way. The imputed data is run through the dimension reduction iteratively when training the four classification algorithms (k-NN, LDA, SVM and RF). The data is trained and validated to produce approximately unbiased estimates of the test errors/accuracy. After the data is run thought the classification process and the accuracy are produced, the algorithms are ranked and evaluated accordingly. The outputs of the whole ML procedure are i) clinical clusters that *may* have distinct phenotypical properties and ii) the accuracy of the various classification

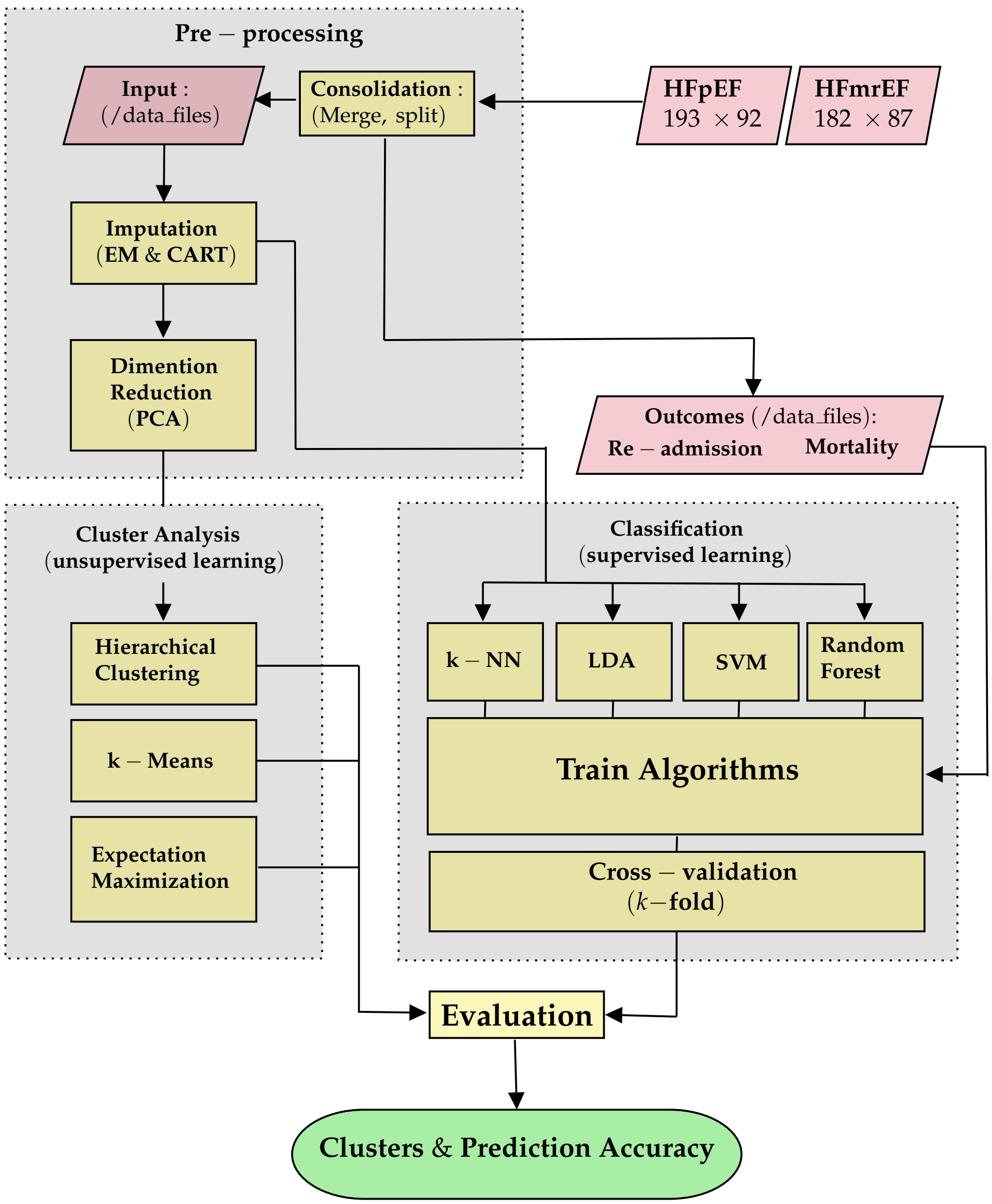

Figure 3.1: *Machine learning procedure adopted in the thesis*

algorithms in predicting re-admission and mortality in the data sets. All the processes mentioned in the ML procedure in figure (3.1) are developed using the `R` statistical programming language (version 3.4.4 - *Someone to Lean On*) (R Core Team, 2018a) with RStudio as the integrated development environment (IDE), version 1.1.423 (RStudio Team, 2018). We use a number of external libraries and self-made algorithms in order to make the process more efficient. Data description with variable explanations, descriptive statistics and some relevant plots can be found in appendix (A). The source code used to produce all the results in this thesis, can be found in appendix (B). As we now have given an overview of the ML procedure used in this thesis, we move on to presenting the data.

3.2 Data

The data used is comprised of two data sets (`data_use_HFpEF.mat`, dim: 193×92 and `data_use_HFmrEF.mat`, dim: 182×87). Being that both data sets has different types of clinical variables, we consolidated the data into three main data set with the same number and types of variables:

(i) Full sample (`HFfullDataSet.Rdat`, dim: 374×55)

(ii) HFpEF sample (`HFpEFdataSet .Rdat`, dim: 193×55)

(iii) HFmrEF sample (`HFmrEFdataSet.Rdat`, dim: 182×55)

The data was collected by the medical staff at a tertiary hospital in the United Kingdom. At this particular hospital NT-proBNP led heart failure service were run on all patients with suspected heart failure. All patients with suspected HF based on an assessment of the HF probability and raised NT-proBNP/BNP levels (see figure 2.1) were included and forwarded for an echocardiography. An expert HF physician reviewed all the patients after the echocardiography was performed. The patients were diagnosed with HF according to the 2016 ESC guidelines (Ponikowski et al., 2016). Accordingly, signs and symptoms of HF, raised NP values, echocardiographic results including left ventricular ejection fraction (LVEF) and evidence of structural or functional heart abnormalities were the primary basis for the assessment done by the hospitals cardiac physicians. After the diagnosis patients were categorized based on LVEF following the ESC guidelines, i.e. patients with LVEF $> 50\%$ were classified as HFpEF and those with $40 \leq$ LVEF < 50 as HFmrEF. The patients with LVEF $< 40\%$, greater than moderate valvular heart disease and prior cardiac transplantation were excluded. The data was collected over a one-year period from October 10th 2014 to October 9th 2015. In total 375 patients were analyzed over this one-year period with data from 126 clinical features being recorded. The outcomes were evaluated through the hospital databases and mortality was confirmed with the Office for National Statistics. All the data was collected as part of the hospitals approved Clinical Audit. As mentioned in the previous section, we reduced the SL problem in the supervised learning part of the ML procedure to a two-class classification problem. The way this was done was with respect to the various `patient_groups` in the data. The patients were grouped based on various outcomes. In total six outcome categories were defined in the data sets. The outcome categories are as follows: `IN` - inhospital mortality, `Z` mortality within 30 days, `Y` - mortality within 1 year, `X` - mortality by Fluorouracil (medication), `V` - cardiac readmission within 30 days, `U` - readmission and `R` - the rest. The various combinations of the outcome classes found in the data sets, and the way in which they were classified, are listed in table (3.1).

Table 3.1: Clinical outcome classes

PANEL I: Full Sample (`HFfullDataSet.Rdat`)				
Group	Mort?	Readm?	n	% Tot
R	no	no	186	0.496
U	no	yes	59	0.157
X, R	yes	no	29	0.077
Y	yes	no	16	0.043
IN	yes	no	15	0.040
V	no	yes	15	0.040
Y, U	yes	yes	13	0.035
X, U	yes	yes	11	0.029
Y, V	yes	yes	11	0.029
X	yes	no	9	0.024
Z	yes	no	7	0.019
X, V	yes	yes	3	0.008
Z, V	yes	yes	1	0.003

PANEL II: Outcome Classes by Clinical Syndrome

HFpEF (`HFpEFdataSet.Rdat`)					HFmrEF (`HFmrEFdataSet.Rdat`)				
Group	Mort?	Readm?	n	% Tot	**Group**	Dead?	Readm?	n	% Tot
R	no	no	85	0.440	R	no	no	101	0.555
U	no	yes	40	0.207	U	no	yes	19	0.104
X, R	yes	no	29	0.150	Y	yes	no	15	0.082
V	no	yes	10	0.052	IN	yes	no	8	0.044
IN	yes	no	7	0.036	X	yes	no	8	0.044
Y, U	yes	yes	7	0.036	Z	yes	no	7	0.038
Y, V	yes	yes	7	0.036	Y, U	yes	yes	6	0.033
X, U	yes	yes	6	0.031	V	no	yes	5	0.027
X	yes	no	1	0.005	X, U	yes	yes	5	0.027
Y	yes	no	1	0.005	Y, V	yes	yes	4	0.022
					X, V	yes	yes	3	0.016
					Z, V	yes	yes	1	0.005

From this table, we can see that approximately 36.8% of all the patients in the HFpEF data set were readmitted in some form, i.e either within 30 days or more. In the HFmrEF data set, this number was somewhat smaller being approximately 23.4%. In the full sample, approximately 29.1% of the patients were readmitted. The number also differed with respect to whether the patients were confirmed deceased or not. In the HFpEF data set, approximately 29.9% of the patients had confirmed mortality and in the HFmrEF data set this number was 31.1%. For the full sample, this number is approximately 30.7%. Further descriptive statistics on the data can be found in appendix (A.3). The source code for the two-class outcome classification shown in table (3.1), can be found in appendix (B.3). As the data used in this thesis is cross-sectional, we need to emphasize that it is not ideal. Limitations to the data sets are many and one of the most relevant one is that of missing data.

3.2.1 Missing Data

Missing values in data is a very important concept in data management and a highly prevalent problem in any data analysis. If one does not handle missing values properly, this may lead to inaccurate or invalid inference being drawn from the data. Results where improper treatment of missing data is present may differ significantly from those where missing data is not present. In medical research, it is not uncommon for patient data to be missing. Missing data from patients clinical variables are typically defined as the values that are not directly observed (Ibrahim et al., 2012).

Table 3.2: Summary of missing values

PANEL I: Full Sample (`HFfullDataSet.Rdat`)				
Variable (V)	#Na	%n	%Na	%V
grand.tot	3081	0.149	1.000	
irondef	254	0.012	0.082	0.677
ferritin	250	0.012	0.081	0.667
bmiadmission	223	0.011	0.072	0.595
ironlevels	210	0.010	0.068	0.560
tsat	210	0.010	0.068	0.560
timetohfadm	184	0.009	0.060	0.491
pasp	181	0.009	0.059	0.483
admissionwgt	164	0.008	0.053	0.437
ecgqrsduration	141	0.007	0.046	0.376
obesity	137	0.007	0.044	0.365

HFpEF (`HFpEFdataSet.Rdat`)					HFmrEF (`HFmrEFdataSet.Rdat`)				
Variable (V)	#Na	%n	%Na	%V	**Variable (V)**	#Na	%n	%Na	%V
grand.tot	973	0.092	1		grand.tot	2108	0.211	1	
irondef	124	0.012	0.127	0.642	bmiadmission	178	0.018	0.084	0.978
timetohfadm	124	0.012	0.127	0.642	admissionwgt	131	0.013	0.062	0.720
ferritin	122	0.011	0.125	0.632	irondef	130	0.013	0.062	0.714
tsat	99	0.009	0.102	0.513	obesity	129	0.013	0.061	0.709
ironlevels	98	0.009	0.101	0.508	ferritin	128	0.013	0.061	0.703
pasp	71	0.007	0.073	0.368	breathless	127	0.013	0.060	0.698
bmiadmission	45	0.004	0.046	0.233	ironlevels	112	0.011	0.053	0.615
ee	41	0.004	0.042	0.212	tsat	111	0.011	0.053	0.610
ecgqrsduration	36	0.003	0.037	0.187	pasp	110	0.011	0.052	0.604
ecgrate	34	0.003	0.035	0.176	ecgqrsduration	105	0.010	0.050	0.577

Data can be missing due to a number of reasons. In clinical research some reasons may include: poor communication with study subject, difficulties assessing the clinical outcomes, lack of consolidation from test, duration of trial etc. The latter is often a reason for missing data, as longer trials tend to produce more risk of missing data. Especially considering that patients often run the risk of being dropped from the studies before completion

(Myers, 2000). In our data sets, the problem with missing values is very much present. In the full data set, a total of 3081 observations are missing accounting for about 14.9% of the total data set. The main non-indicator variables accounting for the highest amount of this number is the lack of registering ferritin levels (`ferritin`, 8.1% of missing), BMI at admission (`bmiadmission`, 7.2%), ironlevels (`ironlevels`, 6.8%), transferrin saturation (`tsat`, 6.8%), time of HF admission (`timetohfadm`, 6%), pulmonary artery systolic pressure (`pasp`, 5.9%), weight at admission (`admissionwgt`, 5.3%) and ECQ QRS duration (`ecgqrsduration`, 4.6%). We can also look at the missing values in both the sub data sets. In the HFpEF data set a total of 973 observations, i.e. approximately 9.2% of the data set is missing. Of the non-indicator variables, the largest contributors can be attributed to the failure of registering time to HF admission (`timetohfadm`, 12.7% of missing), ferritin levels (`ferritin`, 12.5%), transferrin saturation (`tsat`, 10.2%), iron levels (`ironlevels`, 10.1%), pulmonary artery systolic pressure (`pasp`, 7.3%), registering body-mass-index (BMI) at admission (`bmiadmission`, 4.6%), E/e' ratio (`ee`, 4.2%), ECQ QRS duration (`ecgqrsduration`, `3.7%`) and ECG rate (`ecgrate`, 3.5%). These variables contribute to approximately 68.8% of the missing values in the HFpEF data. In the HFmrEF data set, the picture is very much different. In general we can say that this data set has a much larger presence of missing values. Even though the clinical variables used in both sets are the same. In total 2108 observations, i.e. approximately 21.1% of the data is missing. The largest non-indicator contributors are: inability to record the body mass index (BMI) at admission (`bmiadmission`, 8.4%), the weight of patients at admission (`admissionwgt`, 6.2%), ferritin levels (`ferritin`, 6.1%), iron levels (`ironlevels`, 5.3%), transferrin saturation (`tsat`, 5.3%), pulmonary artery systolic pressure (`pasp`, 5.2%) and ECQ QRS duration (`ecgqrsduration`, `5%`). These variables account for 41.1% of the missing values in the HFmrEF data. An overview of the variables with the most missing values in each data set can be found in table (3.2).

3.2.2 Little's Test for MCAR

The presence of missing values has to be addressed by any individual conducting data analysis. As missing values may make the data corrupted and introduce statistical bias that may lead to invalid results and inferences. This is vital for us as many of the statistical methods used later in this thesis cannot be conducted in the presence of missing values. When talking about missing values one typically mention three distinct types of missing values, see e.g. Sterne et al. (2009) and Kaushal (2014) for further explanation. These are as follows:

(i) Missing completely at random (MCAR): This type assumes that there is no systematic difference between the missing values and the observed values. An example can be if blood pressure values are missing due to breakdown in automatic sphygmomanometer, or if blood sugar values are missing due to a non working glucometer.

(ii) Missing at random (MAR): The second type of missing values assumes that any difference between the missing values and the observed values can be explained by differences in the observed values. Again, an example can be that missing blood pressure values or blood sugar values may be lower than the measured values, but only because younger people may be more likely to have missing blood pressure and blood sugar as missing.

(iii) Missing not at random (MNAR): The last and final type assumes that even after the observer data are taken into account, the systematic differences between the observed and missing values are still present. An example can be that people with high values of blood pressure or blood sugar may be less likely to attend an appointment due to headache.

The last type of missing value can only be speculated and thus never determined, see e.g. Rubin (1976), Schafer and Graham (2002) and Moons et al. (2006). In our data, we assume that the missing data is at least missing at random (MAR). This is an assumption that many in the literature place on their data without an attempt at supplying some arguments to support such an assumption. To this we have carried out Little's MCAR test (Little, 1988) on our data (separately on indicator and continuous variables). The test is structured with the following three steps :

(i) First the test starts by using the expectation-maximization (EM) algorithm (Dempster et al., 1977) to estimate the maximum likelihood of the population mean $\tilde{\boldsymbol{\mu}}_{obs,j}$ and variance-covariance matrix $\tilde{\boldsymbol{\Sigma}}_{obs,j}$. Here one enters the $Y : N \times p$ matrix of data into the EM algorithm.

(ii) Next step is to create a set of matrices S_j for $j = 1, \ldots, J$ where each matrix of the data set consists of all cases that are identified with particular missing patterns (0 = not-missing and 1 = missing). Define m_j to be the number of cases that belong to a given missing response pattern in S_j. From these $J-1$ cases, calculate the *observed* vector of means $\hat{\boldsymbol{y}}_{obs,j}$ for each random response pattern.

(iii) The final step comprises of calculating the difference between the observed means in step 2 with the estimated EM-means from step 1 weighted by m_j and the inverse variance-covariance matrix to obtain the following test statistics:

$$d^2 = \sum_{j=1}^{J} m_j \left(\hat{\boldsymbol{y}}_{obs,j} - \tilde{\boldsymbol{\mu}}_{obs,j}\right) \tilde{\boldsymbol{\Sigma}}_{obs,j}^{-1} \left(\hat{\boldsymbol{y}}_{obs,j} - \tilde{\boldsymbol{\mu}}_{obs,j}\right)^T \tag{3.1}$$

Little (1988) showed that d^2 is asymptotically χ^2-distributed with $f = \sum_{j=1}^{J} p_j - p$ degrees of freedom, where p_j is the number of observed variables for cases in S_j. Thus with the use of d^2, a large-sample test of the MCAR assumption compares d^2 with a chi-squared distribution with f df can be done, and rejecting the null hypothesis when d^2 is large. Following this procedure, we have carried out Little's MCAR test and the results are presented in table (3.3). The results were produced using the function `LittleMCAR()` in the `r` package `BaylorEdPsych` (Beaujean, 2012). We removed the variables that had more than 15% missing values from the

Table 3.3: Little's MCAR test

	num col	missing.patterns	Chi.squared (χ^2)	df	p-value
Panel I: Full Sample					
indicator	24	27	273.7770	242	0.07844
continuous	14	15	96.3276	96	0.47141
Panel II: HFpEF					
indicator.1	26	16	103.7992	109	0.62273
continuous.1	17	14	101.7398	103	0.51661
Panel III: HFmrEF					
indicator.2	24	19	141.8979	135	0.32518
continuous.2	14	11	53.9340	51	0.36284

HFpEF data set, 25% from the HFmrEF data set and 20% from the full data set (see table 3.2 for top 10 missing variables). Next, we split the variables into to two data sets, one for the continuous variables and one for the indicator variables. We also removed the variables that had near zero variance using the `nearZeroVar()` function in the `caret` package (Kuhn et al., 2018). As remarked by Beaujean (2012), the `LittleMCAR()` function can be very time inefficient for data sets with more than 50 variables. This time inefficiency is why we split the data sets into the two subsets, i.e. continuous and indicator and thus conducted separate tests on both subsets. The test assumes that the data is MCAR, and this is accordingly the null-hypothesis. From table (3.3), we can see that all the p-values are insignificant at 5% significance level. Suggesting that we cannot reject the null hypothesis of the missing data being MCAR. However, as argued by Allison (1999), just because the data passes this test does not mean that the MCAR assumption is satisfied. The assumptions for MCAR are strong, and a simple test such as the one suggested by Little (1988) does not in and of itself satisfy those assumptions. It merely lends evidence in its support, and given the test results presented in table (3.3) we consider this assumption to be intact. When it comes to the question regarding missing values, there exists many ways of dealing with this problem. Each of these ways have different advantages as well as disadvantages. One of the most used ways of dealing with missing values is through the use of imputation techniques.

3.2.3 Imputation

There exists a wide variety of methods that fall under the class of imputation. In general, all methods that attempt to replace each missing value in a data set with an estimate or a guess are typically classified as being an imputation method (Allison, 1999). A very popular and conventional method of imputing missing values is through the use of mean imputation. This method implies swapping each missing value with the mean of the observed values in the given variable column. The method is very easy to use and maintains the sample size, but it has a problem with underestimating both the variance and standard deviation estimates. This implies that the estimates that produce the imputed values are unbiased see e.g. Scheffer (2002), Enders (2010) and Eekhout et al. (2012). Another class of imputation methods that have proven to handle missing values in a wide variety of cases is the maximum likelihood methods. The use of set method requires that the assumption of MCAR is intact and as such if this is done can produce estimates that have the desirable properties normally associated with maximum likelihood. These properties are consistency (estimates will be approximately unbiased in large samples), asymptotic efficiency (estimates are close to being fully efficient i.e., having minimal standard errors) and asymptotic normality (allows the use of normal approximation to calculate confidence intervals and p-values). Additionally, the use of maximum likelihood methods can produce standard errors that fully account for the fact that some data is missing (Allison, 1999). It is exactly based on these qualities that we have chosen maximum likelihood based imputation as one of the strategies to address the problem with the missing values in our data set presented in subsection (3.2.1). We also showed that this is relevant as the assumption of MCAR is assumed intact, see subsection (3.2.2).

A maximum likelihood method typically starts out by expressing a likelihood function. This function expresses the probability of the data as a function of the unknown parameters. If we assume two discrete random variables: $\mathbf{X}$ and $\mathbf{Z}$ with a joint probability function defined by $p(x, z|\boldsymbol{\theta})$, where $\boldsymbol{\theta}$ is a vector of parameters. This joint probability function gives us the probability that $\mathbf{X} = x$ and $\mathbf{Z} = z$. If we assume there are no missing values and that the observations are independent, i.e. $cov(\mathbf{X}, \mathbf{Z}) = 0$. Then the likelihood function is defined by:

$$L(\boldsymbol{\theta}) = \prod_{i=1}^{n} p(x_i, z_i|\boldsymbol{\theta}) \tag{3.2}$$

To find an estimate of the maximum likelihood, we need to find the value for $\boldsymbol{\theta}$ that maximizes the likelihood function (eq. 3.2). This can be done using the log-likelihood function ($\mathcal{L}(\boldsymbol{\theta}) = \log L(\boldsymbol{\theta})$) and should give us an estimate defined by:

$$\hat{\theta} \in \left\{ \underset{\theta \in \Theta}{\arg\max} \sum_{i=1}^{n} \log p(x_i, z_i|\boldsymbol{\theta}) \right\} \tag{3.3}$$

If we assume that the data is MAR on $\boldsymbol{Z}$ for the first r cases, and MAR on $\boldsymbol{X}$ for the next s cases, we can then split the likelihood function into parts that correspond to each missing value pattern and accordingly factor these parts. This is in order to get a likelihood function that takes into account the missing data patterns. The likelihood function becomes as follows:

$$L(\boldsymbol{\theta}) = \prod_{i=1}^{r} g(x_i|\boldsymbol{\theta}) \prod_{i=r+1}^{r+s} h(z_i|\boldsymbol{\theta}) \prod_{i=r+s+1}^{n} p(x_i, z_i|\boldsymbol{\theta}) \tag{3.4}$$

where $g(x|\theta)$ and $h(z|\theta)$ are the marginal distributions of $\boldsymbol{X}$ and $\boldsymbol{Z}$, so that:

$$\prod_{i=1}^{r} g(x_i|\boldsymbol{\theta}) \prod_{i=r+1}^{r+s} h(z_i|\boldsymbol{\theta}) = \prod_{i=1}^{r+s} p(x_i, z_i|\boldsymbol{\theta}) \tag{3.5}$$

For each missing data pattern, the likelihood is found by summing the joint distribution over all possible values of the variables with missing data. The estimated maximum likelihood parameters in this particular example should therefore be defined by:

$$\hat{\theta} \in \left\{ \underset{\theta \in \Theta}{\arg\max} \left(\sum_{i=1}^{r+s} \log p(x_i, z_i|\boldsymbol{\theta}) + \sum_{i=r+s+1}^{n} \log p(x_i, z_i|\boldsymbol{\theta}) \right) \right\} \tag{3.6}$$

We assumed the variables were discrete in the begin, and as such if the variables were continuous, the summations would be replaced by integrals as the following. The extension to multiple variables is also relatively straightforward (Allison, 1999). In order to implement a maximum likelihood method on data that contains missing values it's important to have a model for the joint distribution for all variables in the data set, and accordingly have a numerical method for maximizing the likelihood of this distribution. Determining this model can vary with the type of data that one is dealing with.

In our data set, we have both continuous and indicator variables. When the data is continuous it's common to assume a multivariate-normal model, i.e. that all the variables are independently identically normally distributed (iid) and can be expressed as a linear function of all other variables (or subsets). There is also an assumption that the errors are homoscedastic, i.e. constant and have a mean of 0. In the case of the indicator variables, it's difficult to assume that these variables are normally distributed. However, according to Schafer (1997), Schafer and Olsen (1998) and Allison (1999) simulation evidence and practical experience have shown that maximum likelihood methods can do a good job in imputing missing values, even if the variables in question are indicator variables. Still, we opted to use a different imputation method for each of the types of data, i.e. we use a bootstrapped expectation-maximization (EM) imputation method for the variables that are continuous and a classification- and regression tree (CART) based imputation method for the indicator variables.

As we mentioned, one needs to have a numerical method for maximizing the likelihood of the joint probability distribution. One of the most common numerical methods is the expectation-maximization (EM) algorithm (Dempster et al., 1977). We mentioned it slightly in subsection (3.2.2), but it is an iterative algorithm that is used to maximize the likelihood function (eq. 3.2) of a number of missing data models. It is comprised of two steps; the expectation step (often called the E step) and the maximization step (called the M step). In the expectation step, the expected values of the log-likelihood is taken over the variables with missing values using the current estimated parameters (Allison, 1999). Afterwards the maximization step involves maximizing the expected log-likelihood in order to get new estimates of the parameters. These two steps are continued until convergence is achieved, i.e. until the estimated parameters of the joint probability distribution doesn't change from one iteration to the next. Most standard software packages using an EM implementation have as a principal output a set of maximum likelihood parameters related to the joint probability distribution. The imputed values are often included in addition, but are not recommended for further analysis. The reason for this is that these imputed values are not designed for that purpose and as such will produce biased estimates of many parameters if used in further analysis (Allison, 1999).

A way to get around this problem is using multiple-imputation. Honaker et al. (2011) introduced a bootstrapped EM algorithm that combines the nice properties of the EM algorithm, i.e. consistency, asymptotic efficiency etc. with the accuracy property of the bootstrap re-sampling method, see Efron (1992) and James et al. (2013). Honaker et al. (2011) also argue that the EMB algorithm they developed is much faster and more reliable than alternative algorithms, in addition to making valid and much more accurate imputations for cross-sectional data. The algorithm is implemented in the `Amalie II` package in `r`. The assumptions of the algorithm are: if we assume that the data set can be expressed as a matrix $\boldsymbol{D}$ consisting of dimensions $(n \times k)$. Let the matrix $\boldsymbol{D}$ be comprised of two parts, i.e. $\boldsymbol{D^{mis}}$ the missing part and $\boldsymbol{D^{obs}}$ the observer part. The matrix $\boldsymbol{D}$ is assumed to follow a multivariate distribution with mean vector $\boldsymbol{\mu}$ and covariance

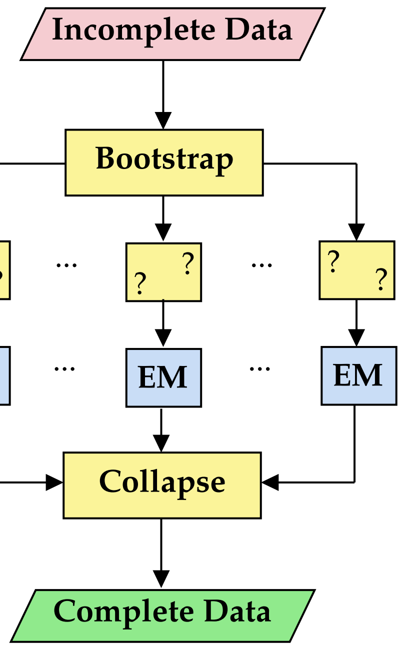

Figure 3.2: *Bootstrapped Expectation Maximization (BEM) procedure*

matrix $\mathbf{\Sigma}$. This assumption can be stated as $\boldsymbol{D} \sim N(\boldsymbol{\mu}, \mathbf{\Sigma})$. In addition to the multivariate normality assumption, the algorithm assumes that the data is MAR. The latter have we already shown to be intact, but the first assumption is somewhat difficult. As the data is by definition incomplete due to the missing data, we assume that this assumption intact. Typically one would test if this assumption is intact using a multivariate normality test similar to the ones mentioned by Mardia (1970), Henze and Zirkler (1990) or Royston (1982).

Most of these tests assume that the data is complete, and should the data be incomplete then its common to remove the missing observations and conduct the tests on the remaining data. The challenge for our part is that approximately 15% of our data set is missing which may cast doubt on the loss of statistical power that these tests may have. As a result of this we have chosen to assume that the normality assumption is intact. The schematic approach of this algorithm and the way it used in this thesis is described in Figure (3.2). The procedure starts by producing n bootstrapped data sets for which the EM algorithm is run on each bootstrapped data sets. In the full data sample (`HFfulldata set.Rdat`) we let the algorithm produce $n = 20$ bootstrapped data sets, whilst for the other data sets, i.e. `HFpEFdata set.Rdat` and `HFmrEFdata set.Rdat` we use $n = 100$. After the imputed data sets are produced they are collapsed by averaging all the imputed values produced by the EM algorithm. All the data from the incomplete data set that the procedure started with should be the same with the exception of the missing values, i.e. these have be replaced by the average of the imputed values.

For the indicator variables, the imputation technique is defined by a classification- and regression tree (CART) algorithm. This algorithm is implemented in the `mice` package in `r` (Buuren and Groothuis-Oudshoorn, 2010). The implementation proceeds as follows: for each variable $\boldsymbol{k}$ in the matrix $\boldsymbol{D}$, the algorithm fits a classification or regression three by recursive partitioning. Then for each missing value in $\boldsymbol{k}$ the algorithm finds the terminal nodes, i.e. the nodes the missing value can end up in according to the fitted tree. Lastly, the algorithm makes a random draw among the members in the nodes, and takes the observed value from that draw as the imputation. Rather than collapsing the multiple imputed data sets as with the BEM algorithm, we simply use the first imputed data sets for further analysis. For further description of the procedure of the algorithm can be found in Burgette and Reiter (2010). Our implementation of the algorithms with the source code can be found in appendix (B.1). This concludes our treatment of the challenge with missing data in this thesis. Next, we present our treatment of the challenge with the higher dimensional data in the thesis.

3.2.4 Dimensional Reduction

As we can see from figure (3.1), the number of features in each HF data sets are 92 and 87. After the consolidation process, we reduce the number of features to 55 in each data sets. As the number of features are so high, we need to have some process to address the challenge of higher dimensional data, i.e. when the number of features are more than the low-dimensional settings such as the three-dimensional physical space of everyday experience. The problem with such higher dimensional data is that some of these features may be noise features that are not truly associated with a given response. This may lead to a deterioration in a fitted model, and thus increase the uncertainty. Noise features may also exacerbate the risk of overfitting, i.e. having a statistical model that contains more parameters than can be justified by the data (Friedman et al., 2009), (James et al., 2013). One can also run the risk of drawing invalid inference as many of the features may be correlated with each other and thus one may face the case of multicollinearity, i.e. risking inflated standard errors.

We have chosen to address this problem with the use of Principal Component Analysis (PCA). The purpose of PCA is to express the information in the data set **D** by a less number of variables **Z**, called principal components. These principal components act as lower dimensional representation of the data that contains as much as possible of the variation. As each of the principal components are computed from the linear combination of the p features then this means that the components are orthogonal and linearly uncorrelated. This property is ideal for addressing the challenge with multicollinearity.

For a given $n \times p$ data set **D**, we assume that each of the variables in D has been centered to have mean zero. We then want the linear combination of the sample feature values of the form $z_{i1} = \theta_{11}x_{i1} + \theta_{21}x_{i2} + \ldots + \theta_{p1}x_{ip}$ that has the largest sample variance, subject to the constraint $\sum_{j=1}^{p} \theta_{j1}^2 = 1$. The optimization problem becomes (James et al., 2013):

$$\max_{\theta_{11},\ldots,\theta_{p1}} \left\{ \frac{1}{n} \sum_{i=1}^{n} \left(\sum_{j=1}^{p} \theta_{j1} x_{ij} \right)^2 \right\} \text{ subject to } \sum_{j=1}^{p} \theta_{j1}^2 = 1 \tag{3.7}$$

In the optimization problem above, we want to maximize the sample variance of the n values of z_{i1}. The elements $z_{11}, \ldots, z_{n1}$ are referred to as the *scores* of the first component and solving the optimization problem can be done using an eigen value decomposition. One can compute these principal components by using the estimated correlation or co-variance matrix of **D**. We have chosen to use the correlation matrix and the implementation of this is done using the `princomp()` function in the `stats`-package in `r`, (R Core Team, 2018b). We run the imputed data sets produced in subsection (3.2.3) through the PCA function and select the principal components that explain as much of the variance in the data set for further analysis. The number of components used for the full sample data set is 4. For the other data sets, i.e. the HFpEF and HFmrEF datasets we use the first two principal components. In the succeeding analysis, we use these principal components as input to the cluster analysis. We also use PCA to train the classification algorithms for predicting the clinical outcomes as mentioned in section (3.1). Much of the literature on the topic applies the same procedure to address the challenge of higher dimensional data, see e.g. Shah et al. (2014), Ahmad et al. (2014) and Katz et al. (2017). A full overview of the source code for the pre-processing procedure is listed in appendix (B.4).

3.3 Clustering Patient Groups

In this section, we present the unsupervised clustering algorithms used in the thesis. The clustering algorithms used are as mentioned: Hierarchical, K-Means and Expectation-Maximization (EM) clustering. As there exists many clustering algorithm, we follow the strategy defined in section (3.1) and try to keep to the ones most used in the literature. An overview of the implementation and the source code can be found in appendix (B.5).

3.3.1 Hierarchical

The first clustering algorithm evaluated in the Cluster Analysis process is the Hierarchical Clustering algorithm, Sibson (1973), Defays (1977) and (Rohlf, 1982). This algorithm uses a simple procedure that takes into account the dissimilarity between clusters and accordingly produces a graphical representation in the form of a dendrogram. The algorithm starts by

Algorithm 1: Hierarchical Clustering

initialization;
  n observations
  Distance measure
  Treat every observation n as its own cluster
for $i = n, n-1, \ldots, 2$ **do**
  Examine and fuse the most similar clusters
  Compute the pairwise inter-cluster dissimilarities
    among the $i-1$ remaining clusters
end
Cut dendrogram based on max relative loss of inertia criteria
return Clusters

defining a measure of the dissimilarity between each pairs of observations, i.e. the patients. A common measure is that of the euclidean distance between pairs of observations. This is the distance measure we also have chosen to use for all clustering algorithms used in this thesis. However, there exists many other distance measures, e.g. squared, polynomial, Manhattan, maximum and Mahalanobis distance. The algorithm the precedes by starts at the bottom of the dendrogram, i.e. with the observations that are most similar, and each of the n observations are treated as its own clusters. Next, the algorithm fuses the two clusters that are most similar and this continued iteratively for the remaining $n-1$ clusters. When the algorithm is finished and the dendrogram is complete, all the clusters are now part of the same cluster. Its then up to the user to choose where to cut the dendrogram. In our implementation, we cut the dendrogram based on the criteria of maximizing the relative loss of inertia. The pseudocode for this algorithm is presented above.

The advantage of the Hierarchical clustering algorithm is that one does not need to define the number of clusters a priori, and thus a user can cut the dendrogram at any given height based on a given index. There are many implementations of this algorithm, but since we use the principal components as input to this and all the other clustering algorithm we have chosen to use the Hierarchical Clustering on Principal Components function `HCPC()` in the `FactoMineR`-package in `r` (Lê et al., 2008). We have also defined an own function (`pca.var.plot()`) that visually presents the clustering results from all the clustering algorithms chosen for evaluation in this thesis. This function is very useful as it can supply the user with a visual illustration of the clustering results for each clustering algorithm. The evaluation criteria used to evaluate the clustering methods is something that we will be addressing in later sections. This is however just one of the algorithms that we use and accordingly we now move on to explaining the K-Means algorithm.

3.3.2 K-Means

The K-means clustering algorithm (Forgy, 1965) is a prototype-based technique for partitioning data into a pre-defined number of clusters (K). The clusters are represented by the centroids of the clusters (Tan et al., 2007). The algorithm assumes that each observations x_i belong to atleast one of the K clusters and that the clusters are non-overlapping, i.e. that no observations belong to more than one cluster. The idea behind the K-means clustering algorithm is that a good clustering is one that minimizes the within-cluster variation, i.e. a measure of how much the amount of observations within a cluster varies $W(C_k)$, where C_k is the set containing the indices of the observations in cluster K. Similar to the Hierarchical clustering algorithm this measure is often the euclidean distance between each pair of observations. Accordingly, the algorithm seeks to solve the following optimization problem (James et al., 2013):

$$\min_{C_i,\cdots,C_k} \left\{ \sum_{k=1}^{K} \frac{1}{|C_k|} \sum_{i,i^\star \in C_k} \sum_{j=1}^{P} \left(x_{ij} - x_{i^\star j}\right)^2 \right\} \quad \text{where} \quad i \neq i^\star \tag{3.8}$$

The solution to this optimization problem is very difficult as there exists K^n possible ways of partitioning n observations into K clusters. The K-means algorithm solves this problem with the following steps represented with the pseudocode:

Algorithm 2: K-Means Clustering

initialization;
  n observations
  Distance measure
  The number of clusters K to be produced
for $i = 1, \ldots, n$ **do**
  Randomly assign a number in $\{1, K\}$ to i
end
while *No changes in cluster assignment* **do**
  for *Each cluster* C_k **do**
    Compute cluster centroid
    for *Each observation in* C_k **do**
      Assign each observation to the cluster
        whose centroid is closest
    end
  end
end
return Clusters

The algorithm takes as input the number of observations n, the defined distance measure and the number of clusters K to be produces. Then all the observations n are assigned a number in the set of the number corresponding to the clusters. This assignment is done at random and serves as the initial cluster assignment for the observations. Then the algorithm iterates until there is no change in cluster assignment between cluster assignment a_t and a_{t-1}. The cluster assignments is done by computing the cluster centroid and assigning each observation in the cluster C to the cluster $C_1, \ldots, C_k$ whose centroid is closest, i.e. given the distance measure.

The disadvantage of the K-Means algorithm is that it requires a user to define the number clusters a priori, which in some cases may be seen as defeating the purpose of the cluster analysis, i.e. the results may vary with the number of clusters chosen. We have tried to address this problem by using the `r` function `NbClust()` (Charrad et al., 2014) that uses 30 indices for determining the number of clusters and proposes to the user the best clustering scheme by the use of the majority-rule of all the indices. As for the actual implementation of the K-Means clustering algorithm, we use the `kmeans()` function in the stats-package (R Core Team, 2018b). The implementation of this algorithm is wrapped in the `pca.cluster.plot()` function we mentioned in the preceding section.

3.3.3 Expectation-Maximization (EM)

The K-Means algorithm is closely related to the EM algorithm (Dempster et al., 1977) for estimating certain Gaussian mixture model(s). As we mentioned in section (3.2.3), the EM algorithm consists of estimating the maximum likelihood parameters of the given Gaussian(s) in question. This is done in the E-step of the algorithm and as such this is responsible for assigning the "responsibilities" for each data points based on its relative density under each mixture components. Whilst the M-step is responsible for recomputing the component density parameters based on the current responsibilities (Friedman et al., 2009). The aim of the EM clustering algorithm is to assign the data into K clusters according to the observations probability of belonging to each of the clusters. Its often stated that the EM algorithm is a "soft" version of the K-Means algorithm as the points are assigned based on a probabilistic (rather than a deterministic) approach (James et al., 2013). The pseudocode for the EM algorithm is given below. Accordingly, the algorithm starts by having the user defining the dataset matrix $\mathbf{D}$, a parametric model f_θ, an initial distribution π_0 and randomly selected parameter θ_0. The algorithm then computes the expected responsibilities on each observations and updates the parameters (π_t, θ_t) with the maximum likelihood estimates $(\pi_{max}, \theta_{max})$. This is done iteratively until convergence.

Algorithm 3: EM Clustering

initialization;
  Data set $\mathbf{D} = \{X_1, \ldots, X_n\}$
  Parametric model f_θ
  Choose an initial distribution π_0 and pick
  a parameter θ_0 at random.
  Denote $\Theta_0 = (\pi_0, \theta_0)$
while *No convergence* **do**
  E step:
  Compute expected responsibilities on each observation $\gamma_{i,k}^t$
  M step:
  Update the parameters $\Theta_t = (\pi_t, \theta_t)$ with the maximization
    parameters π_{max} and θ_{max}.
end
return Clusters produced by EM process

Being that the EM algorithm is similar to the K-Means algorithm it also has the same disadvantages, i.e. the user needs to define the number of clusters to be produced. In addition, it can sometimes be very time consuming or even impossible for the algorithm to achieve convergence, i.e. no change in cluster assignment between iterations. In theory, as the exit criteria of the EM algorithm is defined by convergence. This could mean that the algorithm may never stop as convergence is not guaranteed in all cases. As for the implementation, we use the `Mclust()` function in the `mclust` package in `r` (Scrucca et al., 2017). All the default setting are used in the implementation and as with the previous clustering algorithms, the EM algorithm is also wrapped in the `pca.var.plot()` function.

3.4 Classifying Clinical Outcomes

In this section, we present the supervised classification algorithms used in this thesis. As we mention in section (3.1), the classification algorithms that will be evaluated are: K-nearest neighbours, support vector machines, linear discriminant analysis and random forest.

3.4.1 k-Nearest Neighbours (k-NN)

The k-NN algorithm (Fix and Hodges Jr, 1951) is a widely used algorithm, and often for a good reason. It is very intuitive and simple to understand. In addition, the algorithm performs very well in many cases. This classifier is a memory-based algorithm used for classifying a given observation based on the k nearest neighbours of that observation in the feature space. Mathematically, we say that given a query point x_0, the k-NN algorithm tries to find the k training points $x_{(r)}, r = 1, \ldots, k$ closest in distance to x_0, and thus classify the point x_0 according to the majority rule of the k closest points to x_0, see (Friedman et al., 2009) and (James et al., 2013). The pseudocode for the algorithm is given by the following:

Algorithm 4: k-NN clustering algorithm

initialization;
  X: training data
  Y: class labels of **X**
  x: unknown sample
  Distance measure d
for $i = 1, \ldots, n$ **do**
  Compute distance $d(\mathbf{X}_i, x)$
end
Compute set I containing indices for the k smallest distances $d(\mathbf{X}_i, x)$.
return Majority label for $\{\mathbf{Y}_i \text{ where } i \in I\}$

Based on the pseudocode above, we can see that the k-NN algorithm starts by taking as input the training data **X** which is a subset of the full dataset $\mathbf{X} \subseteq \mathbf{D}$, the class labels **Y** of **X** and the distance measure to be used d. The distance measure between two data points are typically assumed to be a Minkowski distance, i.e. defined as

$$d[i,j] = \left(\sum_{i=1}^{n} |X_{i,k} - X_{j,k}|^p \right)^{1/q} \tag{3.9}$$

where if $p = 1$ or 2, the distance d will correspond to the Manhattan or the Euclidean distance mentioned in section (3.3.1). As q approaches infinity, the distance measure d convergence to the maximum distance, i.e. the largest coordinate difference between data points. After computing the distance between data points, the algorithm classifies the labels of the unknown sample x based on the mapping learned by the training data done with the majority rule.

The observant reader will probability wonder how the unknown sample x is determined? This is something we will address in a later section dealing with cross-validation. However, what we can say is that the implementation of the k-NN algorithm used in this thesis is that of the `knn()` function from the `stats` package in `r` (R Core Team, 2018b). We also need to emphasize that the k-NN algorithm is not without disadvantages, i.e. kNN is slow when you have a lot of observations, since it does not generalize over data in advance, it scans historical database each time a prediction is needed. It also has disadvantages with higher dimensional data as even when one normalizes the data, measures of the distance becomes more blurred as distance to all neighbors becomes more or less the same.

3.4.2 Linear Discriminant Analysis (LDA)

The linear discriminant analysis (LDA) algorithm is very similar to principal component analysis (PCA), i.e. they both try to look for linear combinations that best explain the data. However, LDA, tries explicitly to model the difference between the classes of data. This is done by modeling the distribution of the predictors X separately in each of the response classes (i.e. given Y). The objective of LDA is to perform dimension reduction (similar to PCA) while preserving as much of the class discrimination information as possible. Assuming we have a p dimensional random variable X, that is X follows multivariate normal distribution, i.e. $X \sim N(\mu, \sigma)$. This distribution is formally given by the following, i.e. an observation X is assigned to the class x with the following, see Friedman et al. (2009) and James et al. (2013):

$$f(x) = \frac{1}{(2\pi)^{p/1}|\sigma|^{1/2}} \exp\left\{-\frac{1}{2}(x-\mu)^T\sigma^{-1}(x-\mu)\right\} \tag{3.10}$$

The LDA classifier assumes that the observation in the kth class is drawn from a multivariate normal distribution. By using Bayes formula, the class-specific mean vector μ_k and a covariance matrix $\mathbf{\Sigma}$ one can define the LDA classifier as follows:

$$\gamma_k(x) = x^T\mathbf{\Sigma}^{-1}\mu_k - \frac{1}{2}\mu_k^T\mathbf{\Sigma}^{-1}\mu_k + \log \pi_k \tag{3.11}$$

The advantages of LDA is that for large samples and many variables, LDA is preferred to other discriminant classifiers due to its dimensional reduction nature. The same can be said in the opposite direction. LDA is one of the most common classification algorithms with a dimensional reduction nature. Accordingly, LDA suffers from two main problems: the small sample size and the linearity problem (Tharwat et al., 2017). The linearity problem is present if the underlying structure in the data is non-linear. Should this be the case (which is very common in many domains), then the LDA cannot find a LDA space where the discriminatory information exists in the mean, since it exists in the variance. In a two class situation with a non-linear structure in the data, this means that the means are equal. The small sample size problem is related to the low number of training samples available for each class compared with the dimensionality of the sample space (Tharwat et al., 2017). Either way, the LDA is one of the most popular classification algorithms used in the literature related to HF and accordingly, this is the reason why we choose to include it in the following review. The implementation of this algorithm is done using the `lda()` function from the `MASS`-package in `r`.

3.4.3 Support Vector Machines (SVM)

The next classification algorithm is that of the support vector machines (SVM) (Vapnik, 1963). This classifier is based on the concept of a separating hyperplane, i.e. a flat affine subspace of dimension $p-1$. The support vector machine can be generalized to classify clinical outcomes with non-linear decision boundaries. By choosing a radial kernel (function that quantifies the similarities between two observations), we can create a classifier that takes into the non-linear nature that is often assumed on higher dimensional data. This radial kernel is defined by the generalized inner product function:

$$K(x_i, x_i') = \exp\left\{-\gamma \sum_{j=1}^{p} \left(x_{ij} - x_{i'j}\right)^2\right\} \tag{3.12}$$

Where γ is a positive constant and is often described as a hyperparameter that controls the tradeoff between errors due to bias and variance in our model. The kernel function works by having training observations that are far away from the test observation $x^\star$ playing essentially no role in the predicted class label for $x^\star$. If the euclidean distance between the test observation and training observation is large, then the term $\exp\left(-\gamma \sum_{j=1}^{p}\left(x_{ij} - x_{i'j}\right)^2\right)$ becomes very small. This means that the radial kernel has very local behaviour, i.e. that only nearby training observations will have an effect on the class label of a test observation, see (Friedman et al., 2009) and (James et al., 2013). The advantage of using a svm with a kernel like e.g. the radial one described above is that computationally one only needs to compute $K(x_i, x_i')$ for all $\binom{n}{2}$ distinct pairs of i and i'. However, the disadvantages are that the classification results are very sensitive to the choose γ parameter. The algorithm is also very complex and requires extensive memory for large scale tasks. This is not as relevant for our thesis as our datasets are relatively small. Still, the implementation of the svm algorithm with the radial kernel is done with the help of the `svm()` function in the `e1071` package in `r` (Meyer et al., 2018).

3.4.4 Random Forest

The random forest algorithm (Ho, 1995) is a decision tree based ensemble learning classifier that is much used for both classification and regression tasks. The random forest algorithm used a multitude of decision trees to classify the outcome/class of a classification problem. There is also an important part about the decision trees that the algorithm generates,

Algorithm 5: Random forest

initialization;
  X: training data
  x: unknown sample
  Number of Bootstrap samples B
for $i = 1, \ldots, B$ **do**
  Draw a bootstrap sample $\mathbf{Z}^\star$ of size N from the training data **X**
  Grow a random forest T_b by the following:
  (i). Select m variables at random from the p variables.
  (ii). Pick the best variable/split-point among the m.
  (iii). Split the node into two daughter nodes.
end
Output the assembled trees $\{T_b\}_1^B$.
return $\hat{C}_{rf}^B(x) = \text{majority vote}\left\{\hat{C}_b(x)\right\}_i^B$.

and that is that they are decorrelated. The decision trees are build using the bootstrap re-sampling algorithm, and each time a split in the decision tree is considered, a random sample of m predictors are chosen from the full sample of

p predictors. At each split, a fresh sample of m predictors are chosen, where the number m is typically defined as $\sqrt{p}$. By doing this, the random forest algorithm overcomes the problem of small reductions in variance due to correlated decision trees as is often the case for algorithm like Bootstrap aggregating, i.e. bagging. The pseudo code for the random forest algorithm is mentioned above. As the random forest algorithm can be used for both classification and regression, we present only the pseudo code for the classification case. This is true for all classification algorithms used, see (Friedman et al., 2009) and (James et al., 2013).

Advantages of the random forest algorithm are, as mentioned, that one reduces the risk of overfitting since the algorithm averages the number of decision trees. It also reduces the overall variance since it splits the variables at random each time it builds a decision tree from the given bootstrapped data set. The disadvantages are that its often difficult to interpret how the algorithm works. The results may also vary significantly with the number of trees that are to be produced. Regardless, the random forest algorithm is one of the most popular algorithms for doing classification and accordingly has good performance on many problems including non-linear. The actual implementation of this algorithm is done using the `randomforest()` function in the `randomForest`-package in `r` (Liaw and Wiener, 2002).

3.5 K-fold Cross-Validation

When talking about evaluating the performance of a given classification algorithm one typically mentions the test error rate, i.e. the average prediction error that results from using a statistical learning algorithm. The most common way of estimating the average prediction error is through the way of cross-validation (CV). This is a direct method of estimating the expected extra-sample error $Err = E\left[L\left(Y, \hat{f}\left(X\right)\right),\right]$, i.e. the average generalization error when the method $\hat{f}\left(X\right)$ is applied to an independent test sample from the joint distribution of X and Y, see (Friedman et al., 2009) and (James et al., 2013). In the K-fold cross-validation method (Geisser, 1975) one typically splits the data into K roughly equal-sized parts (also called folds) and for a kth part, we fit the model on the remaining $K-1$ parts of the data. This is done for $k=1,\ldots,K$ and after this is done we are left with K estimates of the prediction error. This prediction error is typically defined by the mean square error (MSE). After producing the mean square error for the $k=1,\ldots,K$ folds, we average the MSEs to produce the k-fold cross-validation estimate. The formula is given by the following:

$$CV_k = \frac{1}{nk}\sum_{i=1}^{k}\sum_{j=1}^{n}\left(Y_{ij}-\hat{Y}_{ij}\right)^2 \tag{3.13}$$

The K-fold cross-validation estimate is one of many criteria used to evaluate the performance of various classifiers. One clear advantage of using the K-fold cross-validation estimate is computational, i.e. the runtime properties of the K-fold cross-validation algorithm are good as one limits the number of splits of the data to K-folds. This can also lower the variance of the prediction error since there is a high chance that all the K-folds are more different than if one was to chose $K=N$ (also called leave-one-out cross validation), see (Friedman et al., 2009). In the setting of this thesis, we will evaluate the classification algorithm mentioned earlier using only the K-fold cross validation algorithm. Furthermore, the psedo-code of the K-fold algorithm is given below. The implementation of the K-fold cross-validation method is typically included in each of the implementations of the classification algorithms mentioned in the previous sections.

Algorithm 6: K-fold cross validation

```
initialization;
    X: training data
    set of parameters Θ
    Learning algorithm A
    Number of folds K
Partition X into X_1, ..., X_k
for for each θ ∈ Θ do
    for i = 1, ..., K do
        h_{i,θ} = A(X_i, θ)
    end
    error(θ) = (1/k) Σ_{i=1}^{k} L_{X_i}(h_i, θ)
end
return θ* = arg min_θ [error(θ)] and h_{θ*} = A(S; θ*)
```

Chapter 4

Experiments

In this chapter, we present the results from the experiments that were done in this thesis. The results are split into two sections, i.e. the results from the cluster analysis and the classification sub procedure mentioned in figure (3.1). For each of the sections we present an overview of the statistical problems that the algorithms are to solve. In this, we also present the assumptions and the evaluation criteria that are used to rank the algorithms given the statistical learning problem that they need to solve.

4.1 Cluster Analysis

In the cluster analysis, we try to see how well the various clustering algorithms perform in producing phenotypically distinct clinical patient groups with HFpEF and HFmrEF. We will organize this section in the following way: we start out by looking at the full sample data set, i.e. `HFfullDataSet.Rdat`. After the pre-processing, we will run the principal components thought the clustering algorithms. The idea is to see how well the clustering algorithms perform in producing patient groups that are more unique compared to the physicians evaluation. Our measure of success is the number of unique baseline characteristics that are statistically significant using the Person χ^2 test for categorical variables, ANOVA for normally distributed variables and Kruskal–Wallis test for non-normally distributed variables (Kruskal and Wallis, 1952). Significance used is that of the conventional 5% level. The implementation is done using the `multigrps`-function from the `CBCgrps`-package in `r` (Zhang et al., 2018).

The algorithms are first going to be performed on the BI clustering HF problem, i.e. see how unique the patient groups produced are given that the only HF subtypes in the dataset is HFmrEF and HFpEF. After this is done, we will see how well the algorithms will perform in producing new clusters within the already defined patient groups from the first round. We will do the same analysis on both the groups that have been defined by the physicians and the first round clustering results.

4.1.1 The BI Clustering HF Problem

In the current clustering problem we assume that the dataset is comprised of two clusters HFmrEF and HFpEF. Accordingly, we allow the algorithms to determine the patients that best correspond to each group. We have plotted the results of the BI clustering problem in Figure (**??**). This plot can in many ways seem very misguiding as it only displays the results along the two first principal components. Still, as we can see from the table below, even if you only cluster based on the first four principal components (27.32% of variance explained), one can produce more unique phenotypically distinct patient groups than the physicians. As we can see from table (4.2), the Hierarchical

Table 4.1: Baseline characteristics of actual clustering

	Total	Cluster1	Cluster2	*p*-value
hb	109.34±20.29	107.85±21.22	110.93±19.18	0.141
pcv	0.34±0.06	0.33±0.06	0.34±0.06	0.159
age	78.64(69.22,84.17)	78.9(69.46,85.37)	78.08(68.73,82.74)	0.141
ewave	0.9(0.74,1.05)	0.92(0.8,1.1)	0.9(0.7,1.01)	0.056
gfr	48(32.5,70)	47(32,72)	51.96(33,67.77)	0.968
k	4.4(4,4.7)	4.4(4.1,4.7)	4.4(4,4.78)	0.664
los	10(4,22)	10(4,22)	10.5(4,21)	0.880
lvef	50(45,57.5)	57.5(55,60)	45(42,47.5)	0.000***
mcv	90.55(85.5,95)	89(85,94)	91.33(87,96)	0.011**
na	139(136,141)	139(136,141)	139(136,141)	0.650
ntprobnp	2848(1230.5,7374)	2217(997,5305)	4063.5(1886.5,9968.25)	0.000***
plts	204(156,268)	217(163,284)	190.87(148.5,241)	0.003**
wbc	7.8(5.9,10.5)	7.6(6,10.5)	8.1(5.9,10.4)	0.727
Total number of significant baseline char:			59	
Continuous:			4	
Categorical:			55	

Table 4.2: Baseline characteristics of Hierarchical and K-Means clustering

	Total	Cluster1	Cluster2	*p*-value
hb	109.34±20.29	106.79±21.29	111.73±19.06	0.019***
pcv	0.34±0.06	0.33±0.07	0.35±0.06	0.035*
age	78.64(69.22,84.17)	78.9(68.94,85.36)	78.26(69.73,82.8)	0.416
ewave	0.9(0.74,1.05)	0.97(0.8,1.1)	0.9(0.7,1)	0.002**
gfr	48(32.5,70)	46(31,70)	54.44(34,71)	0.205
k	4.4(4,4.7)	4.4(4,4.7)	4.4(4,4.8)	0.219
los	10(4,22)	10(4,22)	11(4.25,21)	0.889
lvef	50(45,57.5)	57.5(52.5,60)	45(42.5,47.5)	0.000***
mcv	90.55(85.5,95)	89(84,94)	91.14(87,96)	0.002**
na	139(136,141)	139(136,141)	139(136,141)	0.321
ntprobnp	2848(1230.5,7374)	2327(1007,5695)	3723.5(1731.5,9557.75)	0.000***
plts	204(156,268)	215(163,287)	194(151,241)	0.007**
wbc	7.8(5.9,10.5)	7.7(5.9,10.5)	8.05(5.92,10.47)	0.731
Total number of significant baseline char:			62	
Continuous:			7	
Categorical:			55	

and K-Means clustering algorithms both give the highest number of significant baseline characteristics (7 for only the cont. variables in the table and 62 in total) compare with the actual clustering done by the physicians (4 for the cont. variables and 59 in total). The EM algorithm produces overall the lowest number of significant baseline characteristics (5 for cont. variables and 54 in total). Both the Hierarchical and K-Means algorithm produce the same clustering configurations. The baseline characteristics in the clustering of the patient using the Hierarchical and K-Means clustering show that for the first clustering (HFpEF) the LVEF are on average 57.5% and for the second clustering (HFmrEF) the LVEF are on average 45%. These are very similar values as that which was done by the physicians. We can also see for other baseline characteristics such as ntprobnp the average is at 2327 ng/L for the HFpEF group which is significantly different than that of the HFmrEF group 3723.5 ng/L, also this is very similar to what the physicians concluded with. For characteristics that are significantly different in the clustering with Hierarchical and K-Means clustering, but not done so for the physicians one can include the following continuous variables: hemoglobin (`hb`), packed cell volume (`pcv`) and the ewave (`ewave`). This may suggest that both the clustering algorithms can be used as appropriate tools for physicians. The results from the EM algorithm (Table 4.3) show

Table 4.3: Baseline characteristics of EM clustering

	Total	Cluster1	Cluster2	*p*-value
hb	109.34±20.29	111.2±19.07	107.48±21.34	0.075
pcv	0.34±0.06	0.34±0.06	0.33±0.06	0.115
age	78.64(69.22,84.17)	77.81(69.22,82.76)	78.9(69.22,85.36)	0.199
ewave	0.9(0.74,1.05)	0.9(0.71,1.01)	0.93(0.8,1.1)	0.040*
gfr	48(32.5,70)	51.96(33,68.25)	47(32,72)	0.956
k	4.4(4,4.7)	4.4(4,4.8)	4.4(4,4.7)	0.363
los	10(4,22)	11(4,21)	10(4,22)	0.906
lvef	50(45,57.5)	45(42,47.5)	57.5(53.75,60)	0.000***
mcv	90.55(85.5,95)	91.14(87,96)	89(84.5,94)	0.007**
na	139(136,141)	139(136,141)	139(136,141)	0.330
ntprobnp	2848(1230.5,7374)	3985(1849.5,10038.25)	2226(990,5500)	0.000***
plts	204(156,268)	192.64(149.5,241.5)	217(163.5,286)	0.002**
wbc	7.8(5.9,10.5)	8.1(5.97,10.63)	7.6(5.9,10.35)	0.561
Total number of significant baseline char:			54	
Continuous:			5	
Categorical:			49	

that a lot of the similar baseline characteristics are not statistically significant. The LVEF (`lvef`) and NTproBNP (`ntprobnp`) is very similar to both the Hierarchical and K-means clustering, but other characteristics such as hemoglobin (`hb`) and the packed cell volume (`pcv`) are not. Throughout the analysis we have found that the EM algorithm does not a good job of clustering patient groups compared to the Hierarchical and K-Means clustering algorithms. This could be because of the assumption of multivariate normal distribution does not hold for this data set.

4.1.2 Clustering with and without Post-Diagnosis

In this section we will investigate the clustering results discussed previously. We will place an assumption of weather the physicians diagnosis is representative given an objective of producing the most unique patient groups. The clustering problem in this section assumes that the diagnosis done by the physicians is sufficient in regards to this objective, i.e. the clustering based on the *post-diagnosis* done by the physicians produces the most unique patient groups. We will compare these results to a clustering without an assumption of post-diagnosis done by the physicians and see if

Table 4.4: Number of significant baseline characteristics

	With Post-Diagnosis		Without Post-Diagnosis	
	HFpEF	HFmrEF	HFpEF	HFmrEF
Hierarchical	55	53	48	51
K-Means	55	53	48	53
EM	53	44	42	42

there are any substantial differences in results. We will only use the first two principal components (14.64% of variance explained) to cluster the patients. The evaluation criteria is the same as in the previous section.

We can see from table (4.4) that the Hierarchical and K-Means clustering algorithms almost always produce the same number of significant baseline characteristics. The only exception seems to be were one does not assume a post-diagnosis of the HFmrEF subgroup. The overall algorithm that produces the lowest number of significant baseline characteristics is the Expectation maximization algorithm. Starting with the algorithms performance given the assumption of post-diagnosis. We can see from table (A.3) and (A.4) that for the case with HFpEF, cluster 2 seems to contain patients that have a higher average age (85.45) with a packed cell volume (`pcv`) that is on average 0.33 ± 0.05. Given this information this puts cluster 2 right in the middle of clusters 1 and 3. The ntprobnp (`ntprobnp`) of cluster 3 is the lowest at 1417 ng/L which is also statistically significant. The average number of red blood cells, i.e. the mean corpusular volume (`mcv`) is on average 87 femtolitres which is less than clusters 2 and 3. Overall, we have 8 continuous significantly different baseline characteristics with 47 being categorical (each significant category is counted independently). The EM algorithm produced almost similar results for the subgroup HFpEF (table A.5). The second cluster produced by the EM algorithm is very similar to the first cluster produced by the Hierarchical and K-Means algorithm. The ntprobnp (`ntprobnp`) for cluster one and two produced by the EM are very similar. Both are approximately 2750 ng/L. The third cluster has the lowest values for the ntprobnp (1525 nl/L). Accordingly, we can see that cluster 3 produced by the EM algorithm is very similar to the third cluster produce by the Hierarchical and K-Means algorithms. The total number of significant baseline characteristics for the EM algorithm is 53 (8 cont. and 45 categorical). When looking at the HFmrEF clustering based on post-diagnosis, we can see that a somewhat different results shows up, i.e.

there are less significantly different baseline characteristics. For cluster 3, we see that the lowest ntprobnp (`ntprobnp`) at 2898.5 ng/L with a packed cell volume of 0.38 ± 0.04. This cluster also contains the patients with the lowest length of stay (7 days). The length of stay (LOS) is also a uniquely statistical significant baseline characteristic that is only significant in the HFmrEF subgroup of patients. This cluster also has the highest hemoglobin (`hb`) at 23.79 ± 12.89 g/100mL. Comparing the number of significant baseline characteristics between the HFmrEF groups both with and without the post-diagnosis assumption one can see that the latter has fewer in the case with the assumption, see table (4.4). The same goes for the HFpEF group, i.e. we have reasons to believe that assuming the physicians diagnosis is representative one can get additional clustered patient groups with higher degree of homogeneity compared to when this assumption is not intact. We have also demonstrated that the ML algorithms can be very useful in producing patient groups that are more phenotypically unique given that the objective is to challenge the diagnosis of the physicians, see section (4.1.1). Now that we have presented the results of the clustering analysis, we move on to the results of the classification of the clinical outcomes. The source code, relevant plots and tables can be found in the appendix (A).

4.2 Classification

In this section we will present the results of the classification analysis. As mentioned in ML the procedure (figure 3.1), we run the imputed data set through the various classification algorithms and accordingly run a 10-fold cross validation in order to estimate the accuracy of the various algorithms. The accuracy along with Cohen's kappa are the two evaluation criteria we use to rank the algorithms in this section.

4.2.1 Mortality Classifier

The statistical learning problem in this section is given by a two-class classification problem where mortality is the clinical outcome in question.

4.2.2 Re-admission Classifier

4.3 Discussion

Chapter 5

Conclusion

Appendix A

Data Description

In this appendix we present a descriptive overview of the data used in this thesis. This includes: an overview and explanation of the variables used (A.1), the R-packages (A.2) used and some relevant plots (A.4) to support the finding in the thesis. The source code used to produce the relevant plots can be found in appendix (B).

A.1 Variables

A.2 R-packages

A.3 Descriptive Statistics

Table A.1: Patient characteristics: HFpEF

Variable	n	#Na	Min	Max	$\bar{x}$	$\widetilde{x}$	s	q_1	q_3
PANEL II: Demographics									
age	193	0	29.0	100.8	76.3	78.9	12.1	69.5	85.4
gender	193	0	0.0	1.0	0.6	1.0	0.5	0.0	1.0
white	193	0	0.0	1.0	0.7	1.0	0.5	0.0	1.0
asian	193	0	0.0	1.0	0.1	0.0	0.2	0.0	0.0
black	193	0	0.0	1.0	0.3	0.0	0.4	0.0	1.0
PANEL III: Admission symptoms									
breathless	185	8	0.0	1.0	0.8	1.0	0.4	1.0	1.0
PANEL IV: Admission signs									

Table A.1: Patient characteristics: HFpEF (*continued*)

Variable	n	#Na	Min	Max	$\bar{x}$	$\widetilde{x}$	s	q_1	q_3
sbp	182	11	55.0	242.0	146.9	145.0	31.7	125.0	167.0
dbp	183	10	25.0	195.0	80.5	80.0	22.1	67.0	89.0
admissionwgt	160	33	41.5	158.0	78.9	76.7	23.3	60.1	93.9
bp	192	1	0.0	1.0	0.8	1.0	0.4	1.0	1.0
bmiadmission	148	45	16.8	107.1	30.7	29.3	10.5	23.6	35.4
pulse	182	11	44.0	211.0	84.7	83.0	22.1	70.0	95.0
PANEL V: Risk factors									
a-fib	189	4	0.0	1.0	0.5	0.0	0.5	0.0	1.0
copdasthma	190	3	0.0	1.0	0.4	0.0	0.5	0.0	1.0
irondef	69	124	0.0	1.0	0.6	1.0	0.5	0.0	1.0
dm	188	5	0.0	1.0	0.5	1.0	0.5	0.0	1.0
obesity	185	8	0.0	1.0	0.5	1.0	0.5	0.0	1.0
copdasthma.1	190	3	0.0	1.0	0.4	0.0	0.5	0.0	1.0
ihd	186	7	0.0	1.0	0.4	0.0	0.5	0.0	1.0
PANEL VI: Comorbidities									
comorbidities	193	0	0.0	9.0	4.2	4.0	1.8	3.0	5.0
PANEL VII: Electrocardiography									
ecgqrsduration	157	36	55.0	177.0	101.3	98.0	20.8	88.0	112.0
ecgqrsother	193	0	0.0	1.0	0.0	0.0	0.2	0.0	0.0
ecgrate	159	34	41.0	191.0	83.0	80.0	23.1	70.0	92.0
ecgrhythmother	193	0	0.0	1.0	0.1	0.0	0.2	0.0	0.0
lvh	169	24	0.0	1.0	0.1	0.0	0.3	0.0	0.0
normalecgqrs	193	0	0.0	1.0	0.6	1.0	0.5	0.0	1.0
lbbb	193	0	0.0	1.0	0.0	0.0	0.2	0.0	0.0
rbbb	193	0	0.0	1.0	0.1	0.0	0.3	0.0	0.0
sr	193	0	0.0	1.0	0.6	1.0	0.5	0.0	1.0
PANEL VIII: Laboratory tests									
hb	192	1	47.0	185.0	107.6	107.5	21.1	91.8	123.0
wbc	192	1	2.9	209.4	10.2	7.6	15.8	6.0	10.5
tsat	94	99	4.0	92.0	20.4	18.0	13.8	11.0	24.8
plts	192	1	51.0	497.0	229.4	217.0	89.5	163.0	284.2
pcv	193	0	0.2	0.6	0.3	0.3	0.1	0.3	0.4
ferritin	71	122	9.0	2223.0	378.2	173.0	533.8	61.5	443.5
k	189	4	2.4	8.7	4.4	4.4	0.6	4.1	4.7
ironlevels	95	98	2.0	23.0	8.6	7.0	4.8	5.0	11.0
chol	190	3	0.0	1.0	0.5	1.0	0.5	0.0	1.0
ntprobnp	193	0	81.0	70000.0	5047.3	2217.0	8487.4	997.0	5305.0
gfr	193	0	3.0	221.0	54.1	47.0	31.1	32.0	72.0
mcv	193	0	57.0	117.0	88.8	89.0	8.9	85.0	94.0
na	193	0	110.0	148.0	138.2	139.0	4.9	136.0	141.0
PANEL IX: Echocardiography									
lvef	191	2	50.0	72.5	57.1	57.5	4.5	55.0	60.0
ewave	174	19	0.4	1.6	0.9	0.9	0.3	0.7	1.1
pasp	122	71	14.0	85.0	43.5	42.5	14.2	34.0	51.8
ee	152	41	2.0	37.0	13.4	12.5	5.8	9.0	16.0
mr	193	0	0.0	2.0	0.5	0.0	0.7	0.0	1.0

Table A.1: Patient characteristics: HFpEF (*continued*)

Variable	n	#Na	Min	Max	$\bar{x}$	$\widetilde{x}$	s	q_1	q_3
tr	193	0	0.0	3.0	0.9	1.0	0.8	0.0	1.0
as	193	0	0.0	2.0	0.1	0.0	0.3	0.0	0.0
ai	193	0	0.0	2.0	0.2	0.0	0.5	0.0	0.0
rvfunction	192	1	0.0	4.0	0.6	0.0	1.2	0.0	0.2
af	193	0	0.0	1.0	0.2	0.0	0.4	0.0	0.0
PANEL X: Outcomes									
timetohfadm	69	124	3.8	718.8	192.5	122.7	197.8	33.0	270.0
hfhospitalisation	193	0	0.0	1.0	0.4	0.0	0.5	0.0	1.0
los	171	22	1.0	372.0	15.8	8.0	31.3	4.0	19.0

Table A.2: Patient characteristics: HFmrEF

Variable[i]	n	# Na	Min	Max	$\bar{x}$	$\widetilde{x}$	s	q_1	q_3
PANEL I: Identification									
patientid	182	0	1.0	193.0	96.9	97.5	56.6	47.2	146.5
PANEL II: Demographics									
gender	182	0	0.0	1.0	0.4	0.0	0.5	0.0	1.0
white	182	0	0.0	1.0	0.7	1.0	0.5	0.0	1.0
asian	182	0	0.0	1.0	0.1	0.0	0.3	0.0	0.0
black	182	0	0.0	1.0	0.2	0.0	0.4	0.0	0.0
PANEL III: Admission symptoms									
breathless	55	127	0.0	3.0	2.4	3.0	1.0	2.0	3.0
PANEL IV: Admission signs									
sbp	98	84	86.0	242.0	132.6	126.5	27.7	114.2	147.8
dbp	95	87	45.0	591.0	80.2	72.0	55.7	62.0	85.0
admissionwgt	51	131	21.0	134.9	80.6	80.6	21.8	66.7	96.4
bp	182	0	0.0	1.0	0.7	1.0	0.5	0.0	1.0
bmiadmission	4	178	18.7	36.1	26.0	24.7	8.0	20.2	30.5
pulse	98	84	54.0	144.0	88.8	85.0	21.9	71.2	100.0
PANEL V: Risk factors									
a-fib	182	0	0.0	1.0	0.4	0.0	0.5	0.0	1.0
copdasthma	181	1	0.0	1.0	0.3	0.0	0.5	0.0	1.0
irondef	52	130	0.0	1.0	0.4	0.0	0.5	0.0	1.0
dm	180	2	0.0	1.0	0.4	0.0	0.5	0.0	1.0
obesity	53	129	0.0	1.0	0.5	1.0	0.5	0.0	1.0
copdasthma.1	181	1	0.0	1.0	0.3	0.0	0.5	0.0	1.0
ihd	181	1	0.0	1.0	0.5	0.0	0.5	0.0	1.0
PANEL VI: Comorbidities									
comorbidities	182	0	0.0	7.0	3.2	3.0	1.7	2.0	4.0
PANEL VII: Electrocardiography									
ecgqrsduration	77	105	71.0	182.0	104.9	99.0	24.0	88.0	116.0

Table A.2: Patient characteristics: HFmrEF (*continued*)

Variable	n	#Na	Min	Max	$\bar{x}$	$\widetilde{x}$	s	q_1	q_3
ecgqrsother	182	0	0.0	1.0	0.1	0.0	0.2	0.0	0.0
ecgrate	88	94	42.0	135.0	86.2	83.5	21.5	72.2	99.2
ecgrhythmother	182	0	0.0	1.0	0.0	0.0	0.1	0.0	0.0
lvh	180	2	0.0	3.0	0.6	0.0	0.8	0.0	1.0
normalecgqrs	182	0	0.0	1.0	0.3	0.0	0.4	0.0	1.0
lbbb	182	0	0.0	1.0	0.0	0.0	0.2	0.0	0.0
rbbb	182	0	0.0	1.0	0.0	0.0	0.2	0.0	0.0
sr	182	0	0.0	1.0	0.0	0.0	0.2	0.0	0.0
PANEL VIII: Laboratory tests									
hb	168	14	54.0	153.0	110.7	111.0	19.9	98.0	125.0
wbc	166	16	1.5	39.2	8.3	7.6	4.2	5.9	9.4
tsat	71	111	1.0	65.0	20.4	19.0	12.5	14.0	25.0
plts	166	16	55.0	638.0	203.8	187.0	92.3	143.2	246.5
pcv	166	16	0.2	0.5	0.3	0.3	0.1	0.3	0.4
ferritin	54	128	17.0	3853.0	370.2	225.0	556.3	102.8	448.0
k	165	17	3.0	6.1	4.4	4.4	0.6	4.0	4.8
ironlevels	70	112	2.0	41.0	9.5	8.0	7.1	5.0	11.0
chol	181	1	0.0	1.0	0.4	0.0	0.5	0.0	1.0
ntprobnp	182	0	5.0	70000.0	9604.4	4063.5	14051.2	1886.5	9968.2
gfr	167	15	3.0	400.0	53.5	47.0	39.8	31.0	68.5
mcv	166	16	65.0	112.0	91.0	92.0	8.4	86.0	96.0
na	168	14	4.7	155.0	137.5	139.0	11.5	136.0	141.0
PANEL IX: Echocardiography									
lvef	182	0	40.0	50.0	44.0	45.0	2.9	42.0	47.5
ewave	139	43	0.3	5.0	0.9	0.9	0.5	0.7	1.0
pasp	72	110	18.0	251520.0	3856.5	40.0	29625.6	32.0	53.2
ee	88	94	3.0	43.0	14.9	13.5	7.3	9.0	19.2
mr	159	23	0.0	3.0	0.8	1.0	0.8	0.0	1.0
tr	157	25	0.0	3.0	0.9	1.0	0.9	0.0	1.0
as	140	42	0.0	2.0	0.2	0.0	0.5	0.0	0.0
ai	151	31	0.0	3.0	0.3	0.0	0.5	0.0	0.0
rvfunction	146	36	0.0	6.0	1.2	0.0	2.0	0.0	1.0
af	182	0	0.0	1.0	0.2	0.0	0.4	0.0	0.0
PANEL X: Outcomes									
timetohfadm	122	60	0.4	575.9	84.5	44.9	109.6	11.9	114.7
hfhospitalisation	182	0	0.0	1.0	0.2	0.0	0.4	0.0	0.0
los	169	13	1.0	196.0	16.9	9.0	24.2	4.0	19.0

[i] Note: n - number of observations, #Na - number of missing data, Min - minimal, Max - maximal, $\bar{x}$ - arithmetic mean, $\widetilde{x}$ - median, s - standard deviation, q_1 - first quartile and q_3 - third quartile.

Table A.3: Baseline characteristics of Hierarchical clustering HFpEF based on post-diagnosis

	Cluster1	Cluster2	Cluster3	*p*-value
hb	89.62±15.25	107.37±15.78	120.22±19.61	0.000***
pcv	0.28±0.05	0.33±0.05	0.37±0.06	0.000***
age	77.07(64.44,81.8)	85.45(77.81,88.81)	75.27(67.08,82.06)	0.000***
dbp	75(65,84)	80(67.25,85.65)	84(69.5,92)	0.053
ewave	1.02(0.9,1.2)	0.9(0.77,1)	0.9(0.7,1)	0.000***
gfr	31(22.75,45)	47(38.25,70.75)	60(41,84)	0.000***
k	4.45(4.17,4.8)	4.2(3.7,4.6)	4.4(4.1,4.7)	0.030*
los	11(5,21.29)	10.5(5,22.18)	7(4,20.87)	0.217
lvef	57.5(55,60)	55.5(52.5,57.5)	57.5(55,60)	0.063
mcv	87(80.75,92)	90(85.25,95)	90(86,95.5)	0.010**
na	138(134.75,141)	139(137,142)	139(137,141)	0.107
ntprobnp	2745(1622,7647.25)	2432(1269.75,5920.5)	1417(714.5,3601.5)	0.001***
plts	244.5(170,307.25)	212(164,247)	215(162,286.5)	0.393
pulse	82(69.75,92.75)	79.5(66.75,89.5)	88(71.5,97.5)	0.105
sbp	157(128.75,180)	144.5(129.25,153.94)	147(124.5,164.5)	0.200
wbc	7.65(5.37,10.35)	7.15(5.72,10.7)	8.1(6.45,10.3)	0.270
Total number of significant baseline char:			55	
Continuous:			8	
Categorical:			47	

Table A.4: Baseline characteristics of K-Means clustering HFpEF based on post-diagnosis

	Cluster1	Cluster2	Cluster3	*p*-value
hb	120.22±19.61	107.37±15.78	89.62±15.25	0.000***
pcv	0.37±0.06	0.33±0.05	0.28±0.05	0.000***
age	75.27(67.08,82.06)	85.45(77.81,88.81)	77.07(64.44,81.8)	0.000***
dbp	84(69.5,92)	80(67.25,85.65)	75(65,84)	0.053
ewave	0.9(0.7,1)	0.9(0.77,1)	1.02(0.9,1.2)	0.000***
gfr	60(41,84)	47(38.25,70.75)	31(22.75,45)	0.000***
k	4.4(4.1,4.7)	4.2(3.7,4.6)	4.45(4.17,4.8)	0.030**
los	7(4,20.87)	10.5(5,22.18)	11(5,21.29)	0.217
lvef	57.5(55,60)	55.5(52.5,57.5)	57.5(55,60)	0.063
mcv	90(86,95.5)	90(85.25,95)	87(80.75,92)	0.010**
na	139(137,141)	139(137,142)	138(134.75,141)	0.107
ntprobnp	1417(714.5,3601.5)	2432(1269.75,5920.5)	2745(1622,7647.25)	0.001*
plts	215(162,286.5)	212(164,247)	244.5(170,307.25)	0.393
pulse	88(71.5,97.5)	79.5(66.75,89.5)	82(69.75,92.75)	0.105
sbp	147(124.5,164.5)	144.5(129.25,153.94)	157(128.75,180)	0.200
wbc	8.1(6.45,10.3)	7.15(5.72,10.7)	7.65(5.37,10.35)	0.270
Total number of significant baseline char:			55	
Continuous:			8	
Categorical:			47	

Table A.5: Baseline characteristics of EM clustering HFpEF based on post-diagnosis

	Cluster1	Cluster2	Cluster3	*p*-value
hb	105.98±17.53	89.37±14.55	118.61±19.42	0.000***
pcv	0.33±0.05	0.28±0.04	0.37±0.06	0.000***
age	85.45(77.81,88.65)	75(63.26,80.41)	77.03(68.05,82.11)	0.000***
dbp	79.5(66.25,85.75)	75.5(65.5,83.75)	83(70,91)	0.061
ewave	0.9(0.79,1)	1.1(0.9,1.2)	0.9(0.7,1)	0.000***
gfr	45.5(38,67.75)	31(21.25,45)	60(40,84)	0.000***
k	4.2(3.7,4.6)	4.4(4.13,4.8)	4.5(4.1,4.7)	0.012**
los	11(5,22.18)	11.5(4.25,21.76)	8(4,20.14)	0.269
lvef	56.75(52.5,57.5)	57.5(55,60)	57.5(55,60)	0.194
mcv	89.5(85.25,94)	87(80.25,92)	90(86,97)	0.008**
na	139.5(137,142)	138(135,141)	139(136,141)	0.102
ntprobnp	2755(1451.5,6684.25)	2745(1566,7993.75)	1525(727,3590)	0.000***
plts	212(164,247)	235.5(170,309.75)	219(163,284)	0.402
pulse	79.5(69.25,90)	82(72,94.25)	85(71,97)	0.259
sbp	144(126.37,158.75)	160(131.25,180)	145(125,164)	0.057
wbc	7.15(5.8,10.77)	7.55(5.42,10.17)	7.9(6.4,10.3)	0.506
Total number of significant baseline char:			53	
Continuous:			8	
Categorical:			45	

Table A.6: Baseline characteristics of Hierarchical clustering HFmrEF based on post-diagnosis

	Cluster1	Cluster2	Cluster3	p-value
hb	91.01±14.72	109.11±16.81	123.79±12.89	0.000***
k	4.58±0.66	4.51±0.58	4.18±0.48	0.000***
pcv	0.28±0.04	0.34±0.05	0.38±0.04	0.000***
age	71.53(65.87,82.74)	77.15(70.09,82.04)	79.19(68.12,82.9)	0.455
ewave	0.8(0.7,1)	0.96(0.8,1.14)	0.83(0.67,0.96)	0.003**
gfr	49.98(22,77)	43(30,55)	62(44.5,77.25)	0.000***
los	17(10,36)	12(4,21.48)	7(3,14.25)	0.000***
lvef	45(42,47.5)	45(42,47.5)	42.75(42.5,45)	0.344
mcv	91(87,96)	90(85,95)	93(88,97)	0.159
na	138(135,141)	139(135,141)	139(137.02,142)	0.049*
ntprobnp	6598(2857,27818)	4953(1861,10914)	2898.5(1587.75,5163.5)	0.005**
plts	210(147,285)	204(153,250)	174.74(149.5,215.75)	0.107
wbc	8.2(6.3,9.6)	8.3(6.9,9.8)	7.31(5.7,8.6)	0.025*
Total number of significant baseline char:			53	
Continuous:			9	
Categorical:			44	

Table A.7: Baseline characteristics of K-Means clustering HFmrEF based on post-diagnosis

	Cluster1	Cluster2	Cluster3	*p*-value
hb	91.01±14.72	109.11±16.81	123.79±12.89	0.000***
k	4.58±0.66	4.51±0.58	4.18±0.48	0.000***
pcv	0.28±0.04	0.34±0.05	0.38±0.04	0.000***
age	71.53(65.87,82.74)	77.15(70.09,82.04)	79.19(68.12,82.9)	0.455
ewave	0.8(0.7,1)	0.96(0.8,1.14)	0.83(0.67,0.96)	0.003**
gfr	49.98(22,77)	43(30,55)	62(44.5,77.25)	0.000***
los	17(10,36)	12(4,21.48)	7(3,14.25)	0.000***
lvef	45(42,47.5)	45(42,47.5)	42.75(42.5,45)	0.344
mcv	91(87,96)	90(85,95)	93(88,97)	0.159
na	138(135,141)	139(135,141)	139(137.02,142)	0.049*
ntprobnp	6598(2857,27818)	4953(1861,10914)	2898.5(1587.75,5163.5)	0.005**
plts	210(147,285)	204(153,250)	174.74(149.5,215.75)	0.107
wbc	8.2(6.3,9.6)	8.3(6.9,9.8)	7.31(5.7,8.6)	0.025*
Total number of significant baseline char:			53	
Continuous:			9	
Categorical:			44	

Table A.8: Baseline characteristics of EM clustering HFmrEF based on post-diagnosis

	Cluster1	Cluster2	Cluster3	p-value
hb	83.17±14.19	105.83±17.56	120.8±14.91	0.000***
k	4.65±0.84	4.53±0.57	4.22±0.51	0.001***
pcv	0.26±0.04	0.33±0.05	0.37±0.04	0.000***
age	68.63(41.33,74.15)	76.91(69.62,82.01)	80.97(68.7,83.32)	0.031
ewave	0.88(0.7,1.01)	0.94(0.79,1.12)	0.82(0.67,0.95)	0.009**
gfr	27(13.75,82.25)	42.5(27.5,58)	61(45.75,76.25)	0.000***
los	27.5(13.75,62)	12.5(5,21)	8(3,16.5)	0.006**
lvef	45(42,45.62)	45(42,47.5)	43(42.5,45)	0.517
mcv	91.5(86,96)	89.94(85.06,94)	93(89,97)	0.027*
na	136.5(133.75,141)	139(135.25,141)	139(137,142)	0.175
ntprobnp	19446.5(4178.5,59423.25)	5640.5(1953.25,11400.75)	2898.5(1636,5163.5)	0.001**
plts	155(117,236.25)	205.36(157.25,256)	177.5(147,224)	0.081
wbc	6.55(5.6,7.77)	8.4(7.1,10.3)	7.25(5.7,8.8)	0.001**
Total number of significant baseline char:			44	
Continuous:			8	
Categorical:			36	

Table A.9: Baseline characteristics of Hierarchical clustering HFpEF without post-diagnosis

	Cluster1	Cluster2	Cluster3	*p*-value
hb	87.94±13.31	110.91±17.01	117.31±20.34	0.000***
pcv	0.28±0.04	0.34±0.05	0.37±0.06	0.000***
age	75(64.94,81.79)	84.31(77.36,88.63)	74.08(65.25,82.89)	0.000***
ewave	1.1(0.92,1.27)	0.9(0.7,1)	0.94(0.7,1.1)	0.000***
gfr	29(20.5,44.25)	47(38,68)	55.5(40.25,83.75)	0.000***
k	4.44(4.1,4.8)	4.2(3.7,4.5)	4.45(3.92,4.78)	0.008**
los	12(5,22.19)	10(4,23.83)	8(4,20.76)	0.246
lvef	55(52.5,58.12)	57.5(55,60)	57.5(55,60)	0.203
mcv	87(79.25,92)	89(85,94)	90.5(85.25,96)	0.039*
na	137.5(134.75,141)	140(137,142)	139.5(137,141)	0.024*
ntprobnp	3852(1879.5,9806.75)	1995(934,5573)	1653.5(870.25,3760.75)	0.000***
plts	226(162,303.75)	210(170,251.5)	217(160.25,296.5)	0.889
wbc	7.75(5.7,9.92)	7.2(5.55,10.55)	8.1(6.63,11.13)	0.121
Total number of significant baseline char:			51	
Continuous:			9	
Categorical:			42	

Table A.10: Baseline characteristics of K-Means clustering HFpEF without post-diagnosis

	Cluster1	Cluster2	Cluster3	p-value
hb	117.31±20.34	87.94±13.31	110.91±17.01	0.000***
pcv	0.37±0.06	0.28±0.04	0.34±0.05	0.000***
age	74.08(65.25,82.89)	75(64.94,81.79)	84.31(77.36,88.63)	0.000***
ewave	0.94(0.7,1.1)	1.1(0.92,1.27)	0.9(0.7,1)	0.000***
gfr	55.5(40.25,83.75)	29(20.5,44.25)	47(38,68)	0.000***
k	4.45(3.92,4.78)	4.44(4.1,4.8)	4.2(3.7,4.5)	0.008**
los	8(4,20.76)	12(5,22.19)	10(4,23.83)	0.246
lvef	57.5(55,60)	55(52.5,58.12)	57.5(55,60)	0.203
mcv	90.5(85.25,96)	87(79.25,92)	89(85,94)	0.039*
na	139.5(137,141)	137.5(134.75,141)	140(137,142)	0.024*
ntprobnp	1653.5(870.25,3760.75)	3852(1879.5,9806.75)	1995(934,5573)	0.000***
plts	217(160.25,296.5)	226(162,303.75)	210(170,251.5)	0.889
wbc	8.1(6.63,11.13)	7.75(5.7,9.92)	7.2(5.55,10.55)	0.121
Total number of significant baseline char:			53	
Continuous:			9	
Categorical:			44	

Table A.11: Baseline characteristics of EM clustering HFpEF without post-diagnosis

	Cluster1	Cluster2	Cluster3	p-value
hb	106.38±16.53	84.31±14.29	109.8±21.75	0.000***
pcv	0.33±0.05	0.27±0.04	0.34±0.07	0.000***
age	84.69(76.93,88.61)	71.3(60.76,82.33)	77.79(68.05,84.04)	0.001***
ewave	0.9(0.68,1)	1.1(0.96,1.2)	0.98(0.8,1.1)	0.007**
gfr	44(36.5,56.5)	26.5(10,38.5)	48(31,73)	0.000***
k	4.1(3.7,4.5)	4.4(4.17,4.75)	4.4(4.1,4.7)	0.024*
los	10(4,21.54)	8.5(4,16.5)	10(5,22.08)	0.652
lvef	57.5(54.38,60.62)	57.5(54.38,60.62)	57.5(52.5,60)	0.357
mcv	89(85,93.25)	89(83,93.25)	89(84,95)	0.914
na	140(137,142)	137(134,141.25)	139(136,141)	0.233
ntprobnp	2191.5(1048,5046.25)	4114.5(1707,10007.75)	2184(976,4895)	0.098
plts	206.5(163.75,243.75)	206.5(154,296.75)	221(163,301)	0.494
wbc	7.1(5.5,9.35)	7.05(4.75,9.1)	8.1(6.4,10.9)	0.045*
Total number of significant baseline char:			42	
Continuous:			7	
Categorical:			35	

Table A.12: Baseline characteristics of Hierarchical clustering HFmrEF without post-diagnosis

	Cluster1	Cluster2	Cluster3	*p*-value
hb	89.5±14.24	122.31±13.84	113.5±15.95	0.000***
k	4.55±0.66	4.32±0.49	4.46±0.61	0.114
age	72.08(67.43,82.83)	81.14(74.51,85.01)	77.02(67.73,81.93)	0.006**
ewave	0.8(0.7,1)	0.82(0.62,0.99)	0.9(0.8,1.05)	0.026**
gfr	41(21.75,76.25)	64(47,82.5)	44(31,60.86)	0.000***
los	15.5(8.75,34.5)	9(4,24)	9(3,18)	0.003**
lvef	45(41.61,47.5)	45(42.5,47.5)	42.5(40,47.5)	0.001***
mcv	91(86,96.25)	93(88.71,96)	90(86.75,94)	0.136
na	138(134.75,141)	139(136,141)	139(136.92,141)	0.663
ntprobnp	8937.5(3303.5,26619.5)	2898.5(1440.75,5004.5)	3817.5(1647.25,10311.75)	0.000***
pcv	0.28(0.26,0.31)	0.38(0.35,0.4)	0.36(0.33,0.38)	0.000***
plts	210.5(151.5,285.5)	193(148.5,231.5)	191.27(153.75,226.5)	0.466
wbc	8.65(6.25,12.45)	7.45(6.07,9.22)	8.3(5.87,11.15)	0.375
Total number of significant baseline char:			51	
Continuous:			8	
Categorical:			43	

Table A.13: Baseline characteristics of K-Means clustering HFmrEF without post-diagnosis

	Cluster1	Cluster2	Cluster3	*p*-value
hb	121.6±14.29	114.24±15.78	90.02±14.5	0.000***
k	4.32±0.48	4.47±0.62	4.54±0.65	0.106
age	81.23(75.02,85.37)	77.02(67.73,81.84)	72.08(66.39,82.08)	0.002**
ewave	0.82(0.63,1)	0.9(0.8,1.05)	0.8(0.71,1)	0.045*
gfr	64(46.75,81.5)	44(31,59.35)	44(22.25,76)	0.000***
los	9(4,24)	8.5(3,16.25)	16.5(9.25,35.5)	0.000***
lvef	45(42.5,47.5)	42.5(40,47.5)	45(42,47.5)	0.001**
mcv	93(88.9,96)	89.88(85.75,94)	91(86.25,96)	0.105
na	139(136,141)	139(136.66,141)	138(135,141)	0.804
ntprobnp	2898.5(1526.25,4967.5)	3817.5(1647.25,10807.75)	8656(3176.5,25270.5)	0.001***
pcv	0.38(0.34,0.4)	0.36(0.33,0.39)	0.28(0.26,0.31)	0.000***
plts	193(147,232.5)	193.67(153.75,226.5)	209(153.25,284.75)	0.598
wbc	7.55(6.22,9.32)	8.3(5.87,11.35)	8.5(6.15,12.18)	0.451
Total number of significant baseline char:			53	
Continuous:			8	
Categorical:			45	

A.4 Relevant Plots

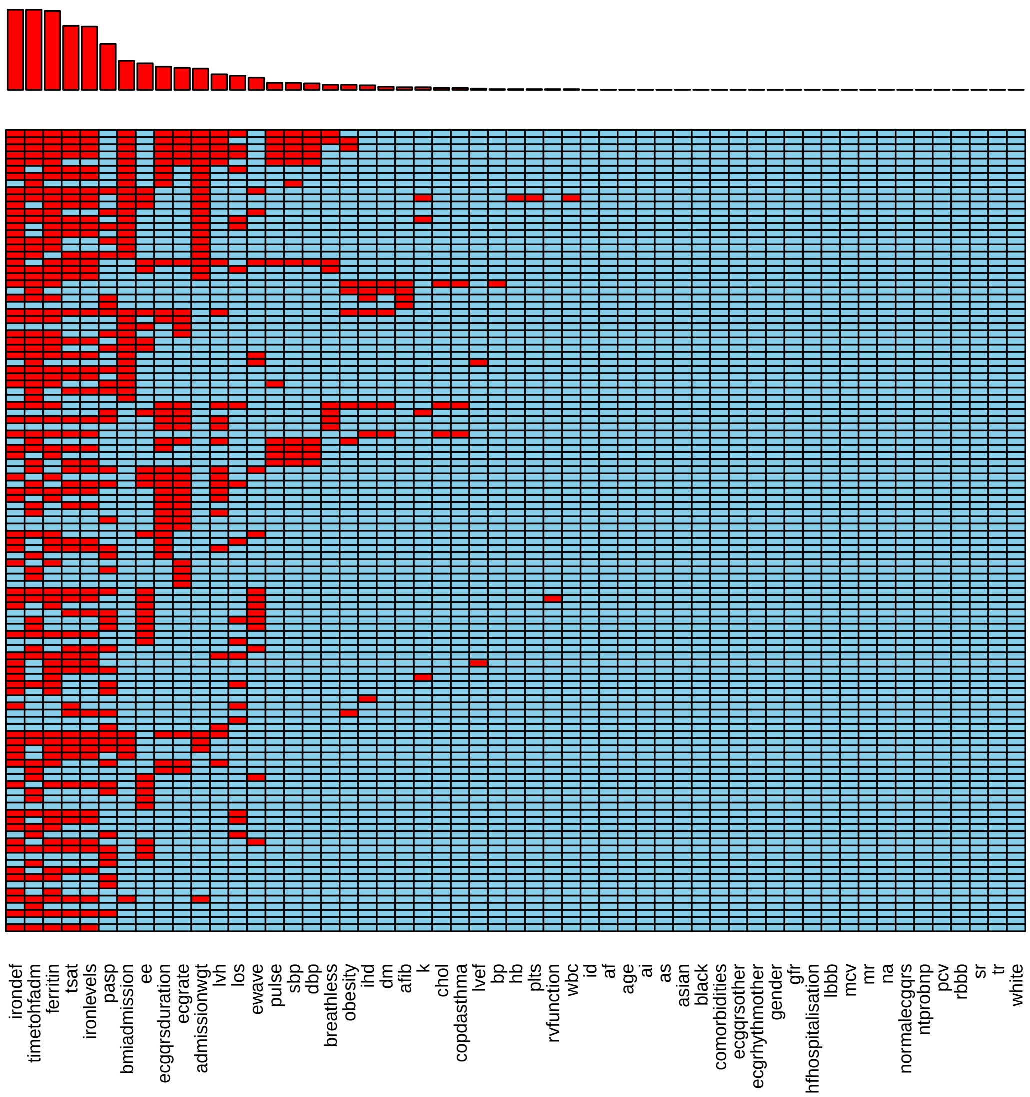

Figure A.1: *Missing values in HFpEF data set. Top: the amount of missing values in each variable sorted in ascending order. Bottom: plot of the combinations of missing (red) and non-missing (blue) values in the HFpEF data set.*

Figure A.2: *Missing values in HFmrEF data set. Top: the amount of missing values in each variable sorted in ascending order. Bottom: plot of the combinations of missing (red) and non-missing (blue) values in the HFmrEF data set.*

Appendix B

Source Code

The following appendix presents all the relevant R-code used in this thesis. We have organized the chapter in accordance with the various steps in the machine learning procedure adopted in this thesis, see figure (3.1). We have tried to comment as much of the source code in order to ensure that an eventual re-examination of the results can be as easy and smooth as possible. Inquires about the code can be forwarded to the author on request.

B.1 Utilities, `utilities.R`

```
# ---------------------------------------------------------------------- #
# Install packages (if not already installed)
# ---------------------------------------------------------------------- #
Packages <- c("docstring", "plotrix", "FactoMineR",
              "factoextra", "gridExtra", "NbClust",
              "ggpubr", "mclust", "CBCgrps")
# install.packages(Packages)

# ---------------------------------------------------------------------- #
# Load package for docstring
# ---------------------------------------------------------------------- #
lapply(Packages, library, character.only = TRUE)

# ---------------------------------------------------------------------- #
# Helper function used in this thesis
# ---------------------------------------------------------------------- #
if.not.class <- function(var, class){
  #' Utility function for error messages
```

```r
  #'
  #' @description Utility function for error messages given
  #' wrong input class as function argument.

  if (!any(class(var) %in% class)){
    stop(paste("first argument must be of class(es) ",
               class, "!", sep = ""))
  }
}

# ------------------------------------------------------------------ #
make.na <- function(data){
  #' Converts all the NaN in a matrix to NA
  #'
  #' @description This function returns a matrix in which all
  #' the NaN values are replaced with NA values. Note! NaN
  #' ("not a number") is not the R syntax for missing values.
  #' The correct syntax is NA ("not available").
  #'
  #' @param data matrix. Matrix containing NaN values

  data[is.nan(data)] <- NA
  return(data)
}

# ------------------------------------------------------------------ #
summary.missing <- function(data){
  #' Summary of the missing values in a dataset
  #'
  #' @description This function returns a list with the total
  #' number of na values and the total percentage in the entire
  #' dataset, including the percentage of missing values for
  #' all variables (columns) and the relative percentage of
  #' missing values to the total (both as vectors).
  #'
  #' @param data matrix. Matrix containing missing values

  num.na <- sum(is.na(data))
  tot.pmv <- num.na/prod(dim(data))
  num.na.vec <- apply(data, 2, function(col) sum(is.na(col)))
  pmv.vec <- num.na.vec / prod(dim(data))
  rel.pmv.vec <- num.na.vec / num.na
  rel.pmv.v <- num.na.vec / dim(data)[1]

  outp <- list(num.na, tot.pmv, num.na.vec, pmv.vec,
```

```r
                 rel.pmv.vec, rel.pmv.v)
  names(outp) <- c("num.na", "tot.pmv", "num.na.vec",
                   "pmv.vec", "rel.pmv.vec", "rel.pmv.v")
  return(outp)
}

# ------------------------------------------------------------------ #
summary.zeros <- function(data){
  #' Summary of the zero values in a dataset
  #'
  #' @description The function returns a list with the
  #' percentage of zero values for all variables in a dataset,
  #' including the total number of zero values and the total
  #' percentage and the relative percentage of zero values
  #' to the total.
  #'
  #' @param data matrix. Matrix containing zero values

  num.zeros <- sum(colSums(data == 0, na.rm = T))
  tot.pzv <- num.zeros / prod(dim(data))
  num.zeros.vec <- colSums(data == 0, na.rm = T)
  pzv.vec <- num.zeros.vec / nrow(data)
  rel.pzv.vec <- num.zeros.vec / num.zeros

  outp <- list(num.zeros, tot.pzv, num.zeros.vec, pzv.vec,
               rel.pzv.vec)
  names(outp) <- c("num.zeros", "tot.pzv", "num.zeros.vec",
                   "pzv.vec", "rel.pzv.vec")
  return(outp)
}

# ------------------------------------------------------------------ #
rm.indicator <- function(data, n.uniq){
  #' Removes indicator variable columns from a dataset based on
  #' predefined number of unique element in that column
  #'
  #' @description This function return a matrix without
  #' indicator variable columns. A indicator variable column is
  #' defined as a column containing less that a predefined
  #' number of unique elements (n.uniq)
  #'
  #' @param data matrix. Matrix containing indicator variables
  #' @param n.uniq integer. Number of unique element in a
  #' column needed for that column to be defined as a indicator
  #' variable column.
```

```
  non.indicator <- data[, apply(data, 2, function(col)
    length(unique(col)) > n.uniq)]
  ind.var.idx <- !(colnames(data) %in% colnames(non.indicator))
  indicator <- data[, ind.var.idx]

  outp <- list(non.indicator, indicator)
  names(outp) <- c("non.indicator", "indicator")
  return(outp)
}

# ---------------------------------------------------------------------- #
rm.missing <- function(data, cut.off = 0.8, near.zero.var = T){
  #' Remove variables with near zero variance or more missing
  #' values than a percentage threshold.
  #'
  #' @description This function removes all variables in a
  #' matrix or dataframe with suspected of having near zero
  #' variance or more missing values than a given percentage
  #' threshold.
  #'
  #' @param data matrix. Matrix like object
  #' @param cut.off integer. Percentage threshold for missing
  #' values.
  #' @param near.zero.var logical. Boolean indicating if
  #' criteria for near zero variance is to be used.

  if (near.zero.var){
    near.zero <- nearZeroVar(data)
    if (length(near.zero) != 0){
      data <- data[, -near.zero]
    }
  }
  miss.col <- summary.missing(data)$rel.pmv.v
  miss.cut <- miss.col < cut.off
  data <- data[, miss.cut]
  return(data)
}

# ---------------------------------------------------------------------- #
zero.to.na <- function(data, except=NULL){
 #' Convert zero datapoints to na in a dataset.
 #'
 #' @description This function converts all the zero datapoints
 #' in a dataset into na. One can also supply a vector of
```

```r
  #' columnames (except) corresponding to variables that this
  #' function should not be applied on.
  #'
  #' @param data matrix. Matrix containing zero datapoints
  #' @param except character vector. Names of matrix column not
  #' to apply function on.

  exp.idx <- colnames(data) %in% except
  exp.data <- data[, exp.idx]; not.exp.data <- data[, !exp.idx]
  not.exp.data[not.exp.data == 0] <- NA
  data <- cbind(not.exp.data, exp.data)
  return(data)
}

# --------------------------------------------------------------------- #
move.columns <- function(from.mat, to.mat, column.name){
  #' Move one column from one matric to another.
  #'
  #' @description This function moves one column with name
  #' column.name from matrix called from.mat to matrix called
  #' to.mat.
  #'
  #' @param from.mat matrix. Matrix to move column from
  #' @param to.mat matrix. Matrix to move column to
  #' @param column.name character. Name of column to be moved

   to.mat <- cbind(to.mat,from.mat[, colnames(from.mat) ==
                                     column.name])
   colnames(to.mat)[ncol(to.mat)] <- column.name
   from.mat <- from.mat[, colnames(from.mat) != column.name]
   outp <- list(from.mat, to.mat)
   names(outp) <- c("from.mat","to.mat")
   return(outp)
}
# --------------------------------------------------------------------- #
sort.column.names <- function(data, id.col = T){
   #' Sorts columns from data
   #'
   #' @description This function sorts the columns names of an
   #' matrix like object.
   #'
   #' @param data matrix. Matrix with columns names
   #' @id.col boolean. Logical indicating if data contains
   #' an id column.

```

```
  if(id.col){
    id <- data[, 1]
    data <- data[,-1]
    data <- cbind(id, data[,sort(colnames(data))])
  }else{
    data <- data[,sort(colnames(data))]
  }
  return(data)
}

# ------------------------------------------------------------------------ #
split.matrix <- function(data){
  #' Split matrix in two parts.
  #'
  #' @description This function splits a matrix into two parts.
  #' Both halfs can be accessed by the user as an output.
  #'
  #' @param data matrix. Matrix like object
  #'
  #' @note The function assumes that the input matrix has more
  #' than one column.

  if (ncol(data)==1){
    stop("data must have more than one column!")
  }
  mid <- trunc(ncol(data)/2); end <- ncol(data)
  first.half <- data[, 1:mid]
  second.half <- data[, (mid+1):end]
  outp <- list(first.half, second.half)
  names(outp) <- c("first.half", "second.half")
  return(outp)
}

# ------------------------------------------------------------------------ #
data.bounds <- function(data, lower.bound, upper.bound){
  #' Generate an Amelia compatible bound matrix
  #'
  #' @description This function produces a three column matrix
  #' to hold logical bounds on the imputations done in Amelia
  #' II. Each row of the matrix is of the form c(column.number,
  #' lower.bound,upper.bound).
  #'
  #' @param data matrix. Matrix like object
  #' @param lower.bound numeric.
  #' @param upper.bound numeric.
```

```
  len <- ncol(data); column.number <- seq(1, len)
  lower <- rep(lower.bound, len)
  upper <- rep(upper.bound, len)
  outp <- cbind(column.number, lower, upper)
  return(outp)
}

# ----------------------------------------------------------------------------- #
boot.em.impute <- function(data, bounds, n.boot = 30){
  #' Impute data using a mean collapsing bootstrapped EM
  #' algorithm.
  #'
  #' @description This function imputes a data matrix using the
  #' bootstrapped EM algorihm from the Amalie II package. The
  #' algorithm creates n.boot number of bootstrapped datasets
  #' after which the datasets are collapsed into one dataset
  #' using the mean of all imputted values as final estimate
  #' of the given missing value.
  #'
  #' @param data matrix. Matrix like object
  #' @param bounds matrix. Three column matrix of the form
  #' c(column.number, lower.bound,upper.bound).
  #' @param n.boot numeric. Number of bootstrapped datasets
  #' to create.

  data.em = list()
  for (i in 1:n.boot){
    print(paste("Bootstrap: ", i, " (", i/n.boot*100, " %)",
                sep=""))
    data.em[[i]] <- amelia(data, m = 1, p2s = 0,
                           bounds = bounds)$imputations$imp1
  }
  return(Reduce("+", data.em) / n.boot)
}

# ----------------------------------------------------------------------------- #
top.n.missing <- function(data, n, decreasing=T){
  #' Summary of top n missing variables in data set.
  #'
  #' @description This function produces a summary table of the
  #' top n missing variables in an inputed dataset.
  #'
  #' @param data matrix. Matrix like object
  #' @param n integer. Top n highest missing variables
```

```
  #' @param decreasing logical. Logical argument indicating
  #' wheater values should be sorted in decreasing order.

  missing <- summary.missing(data)
  count <- missing$num.na.vec
  if (sum(count) == 0){
    stop("no missing values!")
  }
  perc <- missing$pmv.vec
  relp <- missing$rel.pmv.vec
  relv <- missing$rel.pmv.v
  outp <- apply(as.matrix(cbind(count, perc, relp, relv)), 2,
                         sort, decreasing)[1:n,]
  grand.tot <- c(missing$num.na, missing$tot.pmv, sum(relp),
                      NA)
  outp <- rbind(grand.tot, outp)
  colnames(outp) <- c("#Na", "%N", "%Na", "%V")
  return(outp)
}

# ------------------------------------------------------------------ #
label.summary <- function(labels, label.col, col.names, digits,
                               sort.col, ignore.id.col = T,
                               decr = T){
  #' Summary of class labels in data set
  #'
  #' @description The function returns a table with the number
  #' unique labels in a labels matrix and the percentage of
  #' all the labels that occure.
  #'
  #' @param labels matrix. Matrix like object of characters
  #' @param label.col integer. Column number of primary labels
  #' @param col.names charachter vector. Vector of column
  #' names
  #' @param digits integer. Integer indicating the number of
  #' decimal places to be used.
  #' @param sort.col integer. Column number to sort
  #' @param ignore.id.col logical. Boolean indicating whether
  #' first column of id numbers should be ignored.
  #' @param decr logical. Boolean indicating if values in
  #' sort.col should be sorted in decreasing order.

  uniq <- unique(if(ignore.id.col){
    labels[order(labels[, label.col]),-1]}else{labels})
  tabl <- table(labels[, label.col])
```

```
  perc <- round(tabl/sum(tabl), digits)
  outp <- cbind(uniq, tabl, perc)
  colnames(outp) <- col.names
  return(outp[order(outp[, sort.col], decreasing = decr),])
}

# ------------------------------------------------------------------ #
little.mcar <- function(data){
  #' Little's test to assess for missing completely at
  #' random.
  #'
  #' @description This function uses Little's test (from
  #' BaylorEdPsych package) to assess for missing completely at
  #' random for multivariate data with missing values. It
  #' return the chi.squared test statistics, df and p.value.
  #'
  #' @param data matrix like object. Matrix or data frame with
  #' values that are missing.
  #'
  #' @note This function cannot accept data with more than 50
  #' variables, and may in some cases take long time to
  #' complete.

  l <- LittleMCAR(data[, summary.missing(data)$num.na.vec > 0])
  outp <- c(dim(data)[2], l$missing.patterns, l$chi.square,
            l$df, l$p.value)
  names(outp) <- c("n var", "missing.patterns", "chi.square", "df",
                   "p.value")
  outp[1:2] <- round(outp[1:2])
  return(outp)
}

# ------------------------------------------------------------------ #
pca.var.plot <- function(pca, n.comp=NA, digits=4, title = NA){
  #' Plot the explained and cumulative variance from a
  #' principal component analysis (PCA).
  #'
  #' @description This function produces a plot of the
  #' explained and cumulative variance extracted from a
  #' principal component analysis.
  #'
  #' @param pca princomp object.
  #' @param n.comp integer. Number of components to be plotted
  #' @param digits integer. Integer indicating the number of
```

```
  #' decimal places to be used.
  #' @param title character. Name of title.

  if.not.class(pca, "princomp")
  sd <- pca$sdev
  n <- 1:ifelse(is.na(n.comp), length(sd), n.comp)
  vr <- (sd^2/sum(sd^2))[n]
  cm <- cumsum(vr)
  colfunc <- colorRampPalette(c("lightblue","blue"))
  twoord.plot(n, vr, n, cm, type = c("bar", "s"),
              lcol = colfunc(length(n)), main = title,
              cex.axis = 0.5); grid()
  lines(vr); points(vr, pch = 20)
  leg <- c(paste("Number comp:", length(n)),
           paste("Cum.variance:", round(sum(vr),digits)))
  legend("top", legend = leg, bty = "n")
}

# -------------------------------------------------------------------------- #
pca.cluster.plot <- function(pca, ncp, km.clust = 2,
                             hc.clust = -1, em.clust = 2,
                             digits = 5, ellipse = T,
                             actual = NA, fcp=1, scp = 2,
                             ellipse.type = "convex",
                             ggtheme = theme_gray(),
                             return.clust=F){
  #' Side-by-side cluster plots with Hierarchical Clustering,
  #' kMeans and EM clustering on principal components.
  #'
  #' @description This function runs Hierarchical, kMeans and
  #' EM clustering on a predefined number of principal
  #' components. The results are scatterplots with the
  #' results from the clustering.
  #'
  #' @param pca princomp object.
  #' @param ncp numeric. Number of principal components
  #' @param km.clust numeric. Number of clusters to be used
  #' in the kMeans algorithm.
  #' @param hc.clust numeric. Number of clusters to be used
  #' in the Hierarchical clustering.
  #' @param em.clust numeric. Number of clusters to be used
  #' in the expectation maximization algorithm.
  #' @param digits numeric. Number of decimal places for
  #' cumulative variance in plot title.
  #' @param ellipse logical value. Boolean indicating if
```

```
#' ellipse around clusters should be drawn.
#' @param ellipse.type. Type of ellipse to be drawn.
#' See ggscatter for more information.
#' @param ggtheme. function, ggplot2 theme name.
#' @param return.clust. logical. Boolean indicating wheather
#' one want to return the cluster partioning.

if.not.class(pca, "princomp")
data <- as.data.frame(pca$scores[,1:ncp])
sdev <- pca$sdev
rdev <- sdev^2 / sum(sdev^2)
cdev <- cumsum(rdev)
subt <- paste("Cum.variance: ",round(cdev[ncp], digits))
hc.title <- labs(title=paste("Hierarchical Clustering"),
                 subtitle= subt)
km.title <- labs(title = paste("kMeans (k = ", km.clust,
              ") Clustering", sep = ""),subtitle = subt)
em.title <- labs(title = paste("EM Clustering"),
                 subtitle = subt)
xlab <- paste("Dim", fcp, "(",
              round((rdev[fcp])*100, 2),
              "%)", sep = "")
ylab <- paste("Dim", scp," (",
              round((rdev[scp])*100, 2),"%)",
              sep = "")
hc.cluster <- HCPC(data, nb.clust = hc.clust,
                   graph = F)$data.clust$clust
km.cluster <- as.factor(kmeans(data, km.clust)$cluster)
em.cluster <- as.factor(rev(Mclust(data[,1:ncp],
                              em.clust)$classification))
if (all(is.na(actual))){
  data <- cbind(data[, fcp:scp], hc.cluster, km.cluster,
                em.cluster)
}else{
  actual <- as.factor(actual)
  data <- cbind(data[, fcp:scp], hc.cluster, km.cluster,
                em.cluster,
                actual)
}
hc <- ggscatter(data, paste("Comp.", fcp, sep=""),
                paste("Comp.", scp, sep=""),
                color = "hc.cluster",ylab=ylab, xlab=xlab,
                shape = "hc.cluster", ellipse = ellipse,
                ellipse.type = ellipse.type,
                ggtheme = ggtheme, mean.point = T,
```

```r
                    label = seq(nrow(data)) + hc.title
  km <- ggscatter(data, paste("Comp.", fcp, sep=""),
                    paste("Comp.", scp, sep=""),
                    color = "km.cluster", ylab=ylab, xlab=xlab,
                    shape = "km.cluster", ellipse = ellipse,
                    ellipse.type = ellipse.type,
                    ggtheme = ggtheme, mean.point = T,
                    label = seq(nrow(data)) + km.title
  em <- ggscatter(data, paste("Comp.", fcp, sep=""),
                    paste("Comp.", scp, sep=""),
                    color = "em.cluster",ylab=ylab, xlab=xlab,
                    shape = "em.cluster", ellipse = ellipse,
                    ellipse.type = ellipse.type,
                    ggtheme = ggtheme, mean.point = T,
                    label = seq(nrow(data)) + em.title
  if (all(is.na(actual))){
    grid.arrange(hc, km, em, nrow = 2)
  }else{
    act <- ggscatter(data, paste("Comp.", fcp, sep=""),
                       paste("Comp.", scp, sep=""),
                       color = "actual", shape = "actual",
                       ellipse = ellipse,
                       ellipse.type = ellipse.type,
                       ggtheme = ggtheme,
                       label = seq(nrow(data)),ylab=hc$labels$y,
                       xlab = hc$labels$x) +
      labs(title = "Actual Clustering", subtitle = "")
    grid.arrange(act, hc, km, em, nrow = 2)
  }
  if (return.clust){
    clust.list <- list(as.numeric(actual),
                         as.numeric(hc.cluster),
                         as.numeric(km.cluster),
                         as.numeric(em.cluster))
    names(clust.list) <- c("ACT", "HC", "KMC", "EMC")
    return(clust.list)
  }
}

# ------------------------------------------------------------ #
compare.baseline <- function(data, grp, alpha=0.05){
  #' Compare baseline characteristics between two groups.
  #'
  #' @description This function compares the baseline charact-
  #' eristics between two sample groups using an automated
```

```
#' process for determining the distribution of continious
#' variabels and the appropriate tests. The Wilcoxon rank
#' sum test is applied for categorical variables.
#'
#'  @param data matrix like object. Matrix or data frame.
#'  @param grp. group variable
#'
#' @references Zhang Z. Univariate description and bivariate
#' statistical inference: the first step delving into data.
#' Ann Transl Med. 2016 Mar;4(5):91.

 if (length(unique(data[, grp]))>2){
   grp.table <- multigrps(data, grp, sim=T)$table
 }else{
   grp.table <- twogrps(data, grp, sim=T)$table
 }
 grp.list <- list(sum(grp.table[,ncol(grp.table)]<alpha),
                   grp.table)
 return(grp.list)
}

# --------------------------------------------------------------------- #
```

B.2 Consolidation, consolidation.R

```
# --------------------------------------------------------------------- #
# Install packages (if not already installed)
# --------------------------------------------------------------------- #
Packages <- c("R.matlab", "data.table","stringr")
# install.packages(Packages)

# --------------------------------------------------------------------- #
# Load relevant packages
# --------------------------------------------------------------------- #
lapply(Packages, library, character.only = TRUE)
source("../source/utilities.R")

# --------------------------------------------------------------------- #
# Read matlab files into R
# --------------------------------------------------------------------- #
dataSetHFpEF <- readMat('data_use_HFpEF.mat')
dataSetHFmrEF <- readMat('data_use_HFmrEF.mat')

# --------------------------------------------------------------------- #
# Extract the data matrix from matlab files
```

```r
# ---------------------------------------------------------------- #
HFpEFmat <- dataSetHFpEF$All.data
HFmrEFmat <- dataSetHFmrEF$All.data

# ---------------------------------------------------------------- #
# Add all column names
# ---------------------------------------------------------------- #
colnames(HFpEFmat) <- c(as.vector(unlist(
                          dataSetHFpEF$Varnames)))
colnames(HFmrEFmat) <- c(as.vector(unlist(
                          dataSetHFmrEF$Varnames)))

# ---------------------------------------------------------------- #
# Consolidate naming conventions for some variables
# ---------------------------------------------------------------- #
# In the HFpEF matrix
# ---------------------------------------------------------------- #
setnames(as.data.frame(HFpEFmat),
         old = c("E_e","LVfunction", "ECGRhythm_other",
                 "ECGQRS_other", "Other_ethnicity", "Plt",
                 "COPD"),
         new = c("Ee", "LVEF", "ECGRhythmother", "ECGQRSother",
                 "Otherethnicity", "Plts", "COPDasthma"))

# ---------------------------------------------------------------- #
# In the HFmrEF matrix
# ---------------------------------------------------------------- #
setnames(as.data.frame(HFmrEFmat),
         old=c("Admissionweight","BMI","Numberofcomorbidities",
               "Afrocaribbean", "Caucasian","Pulse","NtproBNP",
               "E", "ECGRhythm_other", "LVHand_orLAE",
               "ECGQRS_other", "iron", "Timetoadmission"),
         new = c("admissionwgt","BmIadmission","comorbidities",
                 "Black","White","pulse","NTproBNP", "Ewave",
                 "ECGRhythmother", "LVHandorLAE",
                 "ECGQRSother", "Ironlevels", "TimetoHFadm"))

# ---------------------------------------------------------------- #
# Lowercase letters for all the colnames
# ---------------------------------------------------------------- #
colnames(HFpEFmat) <- tolower(colnames(HFpEFmat))
colnames(HFmrEFmat) <- tolower(colnames(HFmrEFmat))

# ---------------------------------------------------------------- #
# Rename dupblicate names in variables af and ar
```

```r
# ---------------------------------------------------------------- #
if(all(colnames(HFmrEFmat)[c(2,4)] == c("af", "ar"))){
  colnames(HFmrEFmat)[c(2,4)] <- c("afib", "ai")
}
# ---------------------------------------------------------------- #
if(all(colnames(HFpEFmat)[c(3,7)] == c("af", "ar"))){
  colnames(HFpEFmat)[c(3,7)] <- c("afib", "ai")
}

# ---------------------------------------------------------------- #
# Address error in HFmrEF - lvef data point nr. 1
# ---------------------------------------------------------------- #
HFmrEFmat[1, "lvef"] <- 40.45

# ---------------------------------------------------------------- #
# Replace NaN values with NA using the make_na function
# ---------------------------------------------------------------- #
HFpEFmat <- make.na(HFpEFmat)
HFmrEFmat <- make.na(HFmrEFmat)

# ---------------------------------------------------------------- #
# Create one file with all the common variables in both
# HFpEF and HFmrEF data sets.
# ---------------------------------------------------------------- #
# Find common columns in both data sets
# ---------------------------------------------------------------- #
HFpEFcol <- colnames(HFpEFmat) %in% colnames(HFmrEFmat)
HFmrEFcol <- colnames(HFmrEFmat) %in% colnames(HFpEFmat)

# ---------------------------------------------------------------- #
# Test that all columns are equal
# ---------------------------------------------------------------- #
all(sort(colnames(HFpEFmat)[HFpEFcol]) ==
      sort(colnames(HFmrEFmat)[HFmrEFcol]))

# ---------------------------------------------------------------- #
# Get and sort the column names
# ---------------------------------------------------------------- #
HFpEFcol <- sort(colnames(HFpEFmat)[HFpEFcol])
HFmrEFcol <- sort(colnames(HFmrEFmat)[HFmrEFcol])
HFpEFsame <- HFpEFmat[, HFpEFcol]
HFmrEFsame  <- HFmrEFmat[, HFmrEFcol]

# ---------------------------------------------------------------- #
# Create syndrome class matrix
```

```r
# ------------------------------------------------------------------ #
syndrome <- rep(c(1, 2),
                times = c(nrow(HFpEFmat), nrow(HFmrEFmat)))
SyndName <- rep(c("HFpEF", "HFmrEF"),
                times = c(nrow(HFpEFmat), nrow(HFmrEFmat)))

# ------------------------------------------------------------------ #
# Add patient id, create full data set and syndrome classes
# ------------------------------------------------------------------ #
HFfullDataSet <- rbind(HFpEFsame, HFmrEFsame)
id <- seq(1, nrow(HFfullDataSet))
HFfullDataSet <- as.data.frame(cbind(id, HFfullDataSet))
SyndClass <- as.data.frame(cbind(id, syndrome, SyndName))

# ------------------------------------------------------------------ #
# Store indicator and non-indicator variables using the
# rm_indicator function
# ------------------------------------------------------------------ #
HFfullrmInd <- rm.indicator(HFfullDataSet, n.uniq = 8)

# ------------------------------------------------------------------ #
# Store the non-indicator and in variables for later
# ------------------------------------------------------------------ #
HFfullInd <- HFfullrmInd$indicator
HFfullNoInd <- HFfullrmInd$non.indicator

# ------------------------------------------------------------------ #
# Convert zeros to missings, the following variables are not to
# be converted.
# ------------------------------------------------------------------ #
notZeros <- c("comorbidities", "timetohfadm")
HFfullNoInd <- zero.to.na(HFfullNoInd, notZeros)

# ------------------------------------------------------------------ #
# Concatinate indicator and non-indicator variables to one
# data set and sort column names.
# ------------------------------------------------------------------ #
HFfullDataSet <- cbind(HFfullNoInd[, -1], HFfullInd)
HFfullDataSet <- HFfullDataSet[, sort(colnames(HFfullDataSet))]
HFfullDataSet <- cbind(id, HFfullDataSet)

# ------------------------------------------------------------------ #
# Split data according to syndroms
# ------------------------------------------------------------------ #
# Full data set
```

```r
# ---------------------------------------------------------------------- #
HFpEFrow <- SyndClass[,3] == "HFpEF"
HFmrEFrow <- SyndClass[,3] == "HFmrEF"
# ---------------------------------------------------------------------- #
HFpEFdataSet <- HFfullDataSet[HFpEFrow,]
HFmrEFdataSet <- HFfullDataSet[HFmrEFrow,]

# ---------------------------------------------------------------------- #
# Non-indicator variables
# ---------------------------------------------------------------------- #
HFpEFnoInd <- HFfullNoInd[HFpEFrow, ]
HFmrEFnoInd <- HFfullNoInd[HFmrEFrow, ]

# ---------------------------------------------------------------------- #
# Indicator variables
# ---------------------------------------------------------------------- #
HFpEFind <- HFfullInd[HFpEFrow, ]
HFmrEFind <- HFfullInd[HFmrEFrow, ]

# ---------------------------------------------------------------------- #
# Re-code patient group labels
# ---------------------------------------------------------------------- #
# Get patient groups
# ---------------------------------------------------------------------- #
patientGroupsHFpEF <- as.matrix(unlist(
                                dataSetHFpEF$Patient.group))
patientGroupsHFmrEF <- as.matrix(unlist(
                                 dataSetHFmrEF$Patient.group))

# ---------------------------------------------------------------------- #
# Labels of clinical outcomes
# ---------------------------------------------------------------------- #
deceased <- c("IN", "Z", "Y", "X")
reAdmission <- c("V", "U")

# ---------------------------------------------------------------------- #
# Split labels
# ---------------------------------------------------------------------- #
HFpEFsplit <- str_split_fixed(patientGroupsHFpEF,", ", n = 2)
HFmrEFsplit <- str_split_fixed(patientGroupsHFmrEF,", ",n = 2)

# ---------------------------------------------------------------------- #
# Re-coding mortality labels
# ---------------------------------------------------------------------- #
isDeceasedHFpEF <- HFpEFsplit[,1] %in% deceased
```

```r
isDeceasedHFmrEF <- HFmrEFsplit[,1] %in% deceased
deceasedHFpEF <- ifelse(isDeceasedHFpEF, "yes", "no")
deceasedHFmrEF <- ifelse(isDeceasedHFmrEF, "yes", "no")

# ------------------------------------------------------------------------ #
# Re-coding re-admission labels
# ------------------------------------------------------------------------ #
isReAdmittedHFpEF <- HFpEFsplit[,1] %in% reAdmission |
                     HFpEFsplit[,2] %in% reAdmission
isReAdmittedHFmrEF <- HFmrEFsplit[,1] %in% reAdmission |
                      HFmrEFsplit[,2] %in% reAdmission
reAdmissionHFpEF <- ifelse(isReAdmittedHFpEF,"yes","no")
reAdmissionHFmrEF <- ifelse(isReAdmittedHFmrEF,"yes","no")

# ------------------------------------------------------------------------ #
# Add outcomes to matrix
# ------------------------------------------------------------------------ #
HFpEFoutcomes <- cbind(id[HFpEFrow], patientGroupsHFpEF,
                       deceasedHFpEF, reAdmissionHFpEF)
HFmrEFoutcomes <- cbind(id[HFmrEFrow], patientGroupsHFmrEF,
                        deceasedHFmrEF, reAdmissionHFmrEF)

# ------------------------------------------------------------------------ #
# Add colnames to matrices
# ------------------------------------------------------------------------ #
colnames(HFpEFoutcomes) <- colnames(HFmrEFoutcomes) <-
  c("id", "patientgroup", "deceased", "readmitted")

# ------------------------------------------------------------------------ #
# Create outcomes data frames
# ------------------------------------------------------------------------ #
HFfullOutcomes <- as.data.frame(rbind(HFpEFoutcomes,
                                      HFmrEFoutcomes))
rownames(HFfullOutcomes) <- HFfullOutcomes[,1]
# ------------------------------------------------------------------------ #
HFpEFoutcomes <- HFfullOutcomes[HFpEFrow,]
HFmrEFoutcomes <- HFfullOutcomes[HFmrEFrow,]

# ------------------------------------------------------------------------ #
# Save all data frames (13 df in all)
# ------------------------------------------------------------------------ #
path <- "../source/data_files/"; r <- ".Rdat"
fileNames <- c("HFfullDataSet", "HFfullNoInd", "HFfullInd",
               "HFpEFdataSet", "HFpEFnoInd", "HFpEFind",
               "HFmrEFdataSet", "HFmrEFnoInd", "HFmrEFind",
```

```r
                    "HFfullOutcomes", "HFpEFoutcomes",
                    "HFmrEFoutcomes", "SyndClass")
for (name in fileNames){
  save(list = (name), file = paste(path, name, r, sep = ""))
}

# ------------------------------------------------------------------ #
```

B.3 Descriptive Statistics, `desc_stat.R`

```r
# ------------------------------------------------------------------ #
#  Install relevant packages (if not already done)
# ------------------------------------------------------------------ #
Packages <- c("reporttools", "VIM", "Hmisc", "xtable",
              "tikzDevice")
#  install.packages(Packages)

# ------------------------------------------------------------------ #
# Load relevant packages and source helper functions
# ------------------------------------------------------------------ #
lapply(Packages, library, character.only = T)
source("_helper_func.R")

# ------------------------------------------------------------------ #
# Load HFpEF and HFmrEF datafiles
# ------------------------------------------------------------------ #
path <- "data_files/"; r <- ".Rdat"
fileNames <- c("HFpEFdataSet", "HFmrEFdataSet",
               "HFpEFoutcomes", "HFmrEFoutcomes",
               "HFfullDataSet", "HFfullOutcomes")
lapply(gsub(" ", "", paste(path, fileNames, r)),
       load,.GlobalEnv)

# ------------------------------------------------------------------ #
#  Plot of missing values distribution
# ------------------------------------------------------------------ #
pathToImages <- "../../../doc/thesis/images/"

tikz(file=paste(c(pathToImages,"HFpEF_miss_dist.tex"),
                collapse = ""))
aggr(HFpEFdataSet, plot = T, sortVars = T,
     bars = F, combined = T, ylabs = "", cex.axis = 0.7)
dev.off()

```

```
tikz(file = paste(c(pathToImages, "HFmrEF_miss_dist.tex"),
                  collapse = ""))
aggr(HFmrEFdataSet, plot = T,
      sortVars = T, bars = F, combined = T, ylabs = "",
      cex.axis = 0.7)
dev.off()

# ------------------------------------------------------------------- #
# Summary of variables
# ------------------------------------------------------------------- #
# Reorder data matrix by phenotype domains
# ------------------------------------------------------------------- #
nameOrder <- c("age", "gender", "white", "asian", "black",
               "breathless", "sbp", "dbp", "admissionwgt",
               "bp", "bmiadmission", "pulse", "a-fib",
               "copdasthma", "irondef", "dm", "obesity",
               "copdasthma", "ihd", "comorbidities",
               "ecgqrsduration", "ecgqrsother", "ecgrate",
               "ecgrhythmother", "lvh", "normalecgqrs", "lbbb",
               "rbbb", "sr", "hb", "wbc", "tsat", "plts","pcv",
               "ferritin", "k", "ironlevels", "chol",
               "ntprobnp", "gfr", "mcv", "na", "lvef", "ewave",
               "pasp", "ee", "mr", "tr", "as", "ai",
               "rvfunction", "af", "timetohfadm",
               "hfhospitalisation", "los")

# ------------------------------------------------------------------- #
# Descriptive statistics
# ------------------------------------------------------------------- #
capHFpEF <- "Patient characteristics: HFpEF"
labHFpEF <- "tab:desc_stat_HFpEF"
tableContinuous(HFpEFdataSet[, nameOrder],
                stats = c("n", "na", "min", "max", "mean",
                          "median", "s", "q1", "q3"),
                cap = capHFpEF, lab = labHFpEF)

# ------------------------------------------------------------------- #
capHFmrEF <- "Patient characteristics: HFmrEF"
labHFmrEF <- "tab:desc_stat_HFmrEF"
tableContinuous(HFmrEFdataSet[, nameOrder],
                stats = c("n", "na", "min", "max", "mean",
                          "median", "s", "q1", "q3"),
                cap = capHFmrEF, lab = labHFmrEF)

# ------------------------------------------------------------------- #
```

```r
# Outcomes table
# ------------------------------------------------------------------ #
r <- rep("", 5)

tabOutHFfull <- rbind(label.summary(as.matrix(HFfullOutcomes),
                        2, cbind("Group", "Mort?", "Readm?", "n",
                                 "%Tot"), 3, 5))

tabOutHFpEF <- rbind(label.summary(as.matrix(HFpEFoutcomes),
                        2, c("Group", "Mort?", "Readm?", "n",
                             "% Tot"), 3, 5), r, r)

tabOutHFmrEF <- label.summary(as.matrix(HFmrEFoutcomes),
                    2, c("Group", "Mort?", "Readm?",
                         "n", "% Tot"), 3, 5)

print(xtable(tabOutHFfull), include.rownames = F)
print(xtable(cbind(tabOutHFpEF, tabOutHFmrEF)),
                    include.rownames = F)

# ------------------------------------------------------------------ #
# Tables of top 10 missing values variables in both data sets
# ------------------------------------------------------------------ #
HFfullMiss <- top.n.missing(HFfullDataSet, 10)
HFpEFmiss <- top.n.missing(HFpEFdataSet, 10)
HFmrEFmiss <- top.n.missing(HFmrEFdataSet, 10)

# ------------------------------------------------------------------ #
# Combine missing values table and convert to Latex code
# ------------------------------------------------------------------ #
xtable(HFfullMiss, digits = c(0,0,3,3,3))
xtable(cbind(round(HFpEFmiss,3), rownames(HFmrEFmiss),
       round(HFmrEFmiss,3)))

# ------------------------------------------------------------------ #
```

B.4 Pre-processing, pre_process.R

```r
# ------------------------------------------------------------------ #
# Install packages (if not already installed)
# ------------------------------------------------------------------ #
Packages <- c("BaylorEdPsych", "Amelia", "mice", "NbClust",
              "caret", "rlist", "xtable")
# install.packages(Packages)
```

```r

# --------------------------------------------------------------------------- #
# Load package for docstring
# --------------------------------------------------------------------------- #
lapply(Packages, library, character.only = TRUE)

# --------------------------------------------------------------------------- #
# Load data set with same variables and source helper functions
# --------------------------------------------------------------------------- #
allDataFiles <- c("HFpEFind", "HFmrEFind",
                  "HFpEFnoInd", "HFmrEFnoInd",
                  "HFfullDataSet", "SyndClass")
lapply(gsub(" ", "", paste("data_files/", allDataFiles,
                           ".Rdat")), load,.GlobalEnv)
source("utilities.R")

# --------------------------------------------------------------------------- #
# Summary of missing variables
# --------------------------------------------------------------------------- #
top.n.missing(HFfullDataSet, 10)
top.n.missing(cbind(HFmrEFnoInd, HFmrEFind), 10)
top.n.missing(cbind(HFpEFnoInd, HFpEFind), 10)

# --------------------------------------------------------------------------- #
# Split variables into indicator and categorical variables
# --------------------------------------------------------------------------- #
HFfullRmInd <- rm.indicator(HFfullDataSet, 8)
HFfullInd <- HFfullRmInd$indicator
HFfullNoInd <- HFfullRmInd$non.indicator

# --------------------------------------------------------------------------- #
# Little's test to assess for missing completely at random.
# Remove variables with more than a given cut.off missing
# values and that have near zero variance (not for indicator
# variables).
# --------------------------------------------------------------------------- #
# In Full data set
# --------------------------------------------------------------------------- #
CutOff <- 0.20 # cut.off percentage
HFfullInd <- rm.missing(HFfullInd, cut.off = CutOff,
                        near.zero.var = F)
HFfullNoInd <- rm.missing(HFfullNoInd, cut.off = CutOff)
HFfullList <- list(HFfullInd, HFfullNoInd)
HFfullMcar <- do.call(rbind, lapply(HFfullList, little.mcar))
HFfullCarNames <- c("indicator","continuous")
```

```
rownames(HFfullMcar) <- HFfullCarNames

# ------------------------------------------------------------------- #
# In HFpEF
# ------------------------------------------------------------------- #
CutOff <- 0.15 # cut.off percentage
HFpEFind <- rm.missing(HFpEFind, cut.off = CutOff,
                       near.zero.var = F)
HFpEFnoInd <- rm.missing(HFpEFnoInd, cut.off = CutOff)
HFpEFlist <- list(HFpEFind, HFpEFnoInd)
HFpEFmcar <- do.call(rbind, lapply(HFpEFlist, little.mcar))
HFpEFmcarNames <- c("indicator","continuous")
rownames(HFpEFmcar) <- HFpEFmcarNames

# ------------------------------------------------------------------- #
# In HFmrEF
# ------------------------------------------------------------------- #
CutOff <- 0.25 # cut.off percentage
HFmrEFind <- rm.missing(HFmrEFind, cut.off = CutOff,
                        near.zero.var = F)
HFmrEFnoInd <- rm.missing(HFmrEFnoInd, cut.off = CutOff)
HFmrEFlist <- list(HFmrEFind, HFmrEFnoInd)
HFmrEFmcar <- do.call(rbind, lapply(HFmrEFlist, little.mcar))
HFmrEFmcarNames <- c("indicator","continuous")
rownames(HFmrEFmcar) <- HFmrEFmcarNames
xtable(rbind(HFfullMcar, HFpEFmcar, HFmrEFmcar),
       digits = c(0,0,0,4,0,5))

# ------------------------------------------------------------------- #
# Report missing data after removing variables
# ------------------------------------------------------------------- #
top.n.missing(cbind(HFfullNoInd, HFfullInd), n = 10)
top.n.missing(cbind(HFpEFnoInd, HFpEFind), n = 10)
top.n.missing(cbind(HFmrEFnoInd, HFmrEFind), n = 10)

# ------------------------------------------------------------------- #
# Impute data using Bootstrap EM and CART
# ------------------------------------------------------------------- #
# In Full data set
# ------------------------------------------------------------------- #
m <- 100 # number of bootstrap samples
bnd <- data.bounds(HFfullNoInd, 0, Inf)
HFfullEm <- boot.em.impute(HFfullNoInd, bnd, n.boot = m)
HFfullCart <- complete(mice(HFfullInd, method = "cart"))
```

```r
# ---------------------------------------------------------------------- #
# In HFpEF
# ---------------------------------------------------------------------- #
HFpEFconImpEmList <- HFmrEFconImpEmList <- list()
HFpEFbound   <- data.bounds(HFpEFnoInd, 0, Inf)
HFpEFem <- boot.em.impute(HFpEFnoInd, bounds = HFpEFbound,
                          n.boot = m)
HFpEFcart <- complete(mice(HFpEFind, method ="cart"))

# ---------------------------------------------------------------------- #
# In HFmrEF
# ---------------------------------------------------------------------- #
HFmrEFbound <- data.bounds(HFmrEFnoInd, 0, Inf)
HFmrEFem <- boot.em.impute(HFmrEFnoInd,
                           bounds = HFmrEFbound,
                           n.boot = m)
HFmrEFcart <- complete(mice(HFmrEFind, method ="cart"))

# ---------------------------------------------------------------------- #
# Combine imputed data sets into one
# ---------------------------------------------------------------------- #
HFfullImp <- cbind(HFfullEm, HFfullCart)
HFpEFimp <- cbind(HFpEFem, HFpEFcart)
HFmrEFimp <- cbind(HFmrEFem, HFmrEFcart)

# ---------------------------------------------------------------------- #
# Sort columns and remove 328 outlier
# ---------------------------------------------------------------------- #
HFfullImp <- sort.column.names(HFfullImp, id.col = T)
HFpEFimp <- sort.column.names(HFpEFimp, id.col = T)
HFmrEFimp <- sort.column.names(HFmrEFimp, id.col = T)

# ---------------------------------------------------------------------- #
# Save full data set
# ---------------------------------------------------------------------- #
path <- "data_files/"; r <- ".Rdat"
fileNames <- c("HFfullImp", "HFpEFimp", "HFmrEFimp")

for (name in fileNames){
  save(list = (name), file = paste(path, name, r, sep = ""))
}

# ---------------------------------------------------------------------- #
# Principal component analysis
# ---------------------------------------------------------------------- #
```

```
HFfullpca <- princomp(HFfullImp, cor = T)
HFpEFpca <- princomp(HFpEFimp, cor = T)
HFmrEFpca <- princomp(HFmrEFimp, cor = T)

# ------------------------------------------------------------------ #
# Explained variance
# ------------------------------------------------------------------ #
pca.var.plot(HFfullpca, 31, title = "HF same variables")
pca.var.plot(HFpEFpca, 34, title = "HFpEF")
pca.var.plot(HFmrEFpca, 31, title = "HFmrEF")

# ------------------------------------------------------------------ #
# Save pca objects
# ------------------------------------------------------------------ #
path <- "data_files/"; r <- ".Rdat"
objects <- c("HFfullpca", "HFpEFpca", "HFmrEFpca")

for (object in objects){
  save(list = (object), file = paste(path, object, r, sep = ""))
}

# ------------------------------------------------------------------ #
```

B.5 Clustering, `clustering.R`

```
# ------------------------------------------------------------------ #
# Install relevant packages (if not already done)
# ------------------------------------------------------------------ #
Packages <- c("NbClust")
# install.packages(Packages)

# ------------------------------------------------------------------ #
# Load relevant packages
# ------------------------------------------------------------------ #
lapply(Packages, library, character.only = TRUE)
source("utilities.R")

# ------------------------------------------------------------------ #
# Load pca objects and data files
# ------------------------------------------------------------------ #
allDataFiles <- c("HFfullpca", "HFpEFpca", "HFmrEFpca",
                  "HFfullImp","HFpEFimp", "HFmrEFimp",
                  "SyndClass")
```

```
lapply(gsub(" ", "", paste("data_files/", allDataFiles,
                            ".Rdat")), load,.GlobalEnv)

# ---------------------------------------------------------------------- #
# Determine optimal number of clusters
# ---------------------------------------------------------------------- #
NbClust(HFfullpca$scores[,1:31], min.nc = 2, max.nc = 4,
        method = "kmeans")
NbClust(HFpEFpca$scores[,1:31], min.nc = 2, max.nc = 4,
        method = "kmeans")
NbClust(HFmrEFpca$scores[,1:31], min.nc = 2, max.nc = 4,
        method = "kmeans")

# ---------------------------------------------------------------------- #
# PCA cluster plot for all data sets
# ---------------------------------------------------------------------- #
pdf(file="../../../doc/thesis/images/clustFull.pdf")
clustFull <- pca.cluster.plot(HFfullpca, 31, km.clust = 2,
                              hc.clust = 2, em.clust = 2,
                              actual = SyndClass[,2],
                              return.clust = T, ellipse = F)
dev.off()

# ---------------------------------------------------------------------- #
# Extract cluster configuration and add to data frame
# ---------------------------------------------------------------------- #
ACTfull <- clustFull$ACT
HCfull <- clustFull$HC
KMfull <- clustFull$KM
EMfull <- clustFull$EM

# ---------------------------------------------------------------------- #
# Compare baseline characteristics
# ---------------------------------------------------------------------- #
compare.baseline(cbind(HFfullImp, ACTfull), "ACTfull")
compare.baseline(cbind(HFfullImp, HCfull), "HCfull")
compare.baseline(cbind(HFfullImp, KMfull), "KMfull")
compare.baseline(cbind(HFfullImp, EMfull), "EMfull")

# ---------------------------------------------------------------------- #
# Assuming clustering by physicians is correct
# ---------------------------------------------------------------------- #
clustPefFull <- pca.cluster.plot(HFpEFpca, 2, km.clust = 3,
                                 hc.clust = 3, em.clust = 3,
                                 return.clust = T, ellipse = F)
```

```r

clustMrFull <- pca.cluster.plot(HFmrEFpca, 2, km.clust = 3,
                                hc.clust = 3, em.clust = 3,
                                return.clust = T, ellipse = F)

# ---------------------------------------------------------------- #
# Compare baseline characteristics HFpEF
# ---------------------------------------------------------------- #
HCpEFphy <- clustPefFull$HC
KMpEFphy <- clustPefFull$KM
EMpEFphy <- clustPefFull$EM

compare.baseline(cbind(HFpEFimp, HCpEFphy), "HCpEFphy")
compare.baseline(cbind(HFpEFimp, KMpEFphy), "KMpEFphy")
compare.baseline(cbind(HFpEFimp, EMpEFphy), "EMpEFphy")

# ---------------------------------------------------------------- #
# Compare baseline characteristics HFmrEF
# ---------------------------------------------------------------- #
HCmrEFphy <- clustMrFull$HC
KMmrEFphy <- clustMrFull$KM
EMmrEFphy <- clustMrFull$EM

compare.baseline(cbind(HFmrEFimp, HCmrEFphy), "HCmrEFphy")
compare.baseline(cbind(HFmrEFimp, KMmrEFphy), "KMmrEFphy")
compare.baseline(cbind(HFmrEFimp, EMmrEFphy), "EMmrEFphy")

# ---------------------------------------------------------------- #
# Assumin clustering by physicians is incorrect
# ---------------------------------------------------------------- #
hiKmeansClust <- clustFull$HC
HFpEFhiKmeans <- HFfullImp[hiKmeansClust==1,]
HFmrEFhiKmeans <- HFfullImp[hiKmeansClust==2,]

# ---------------------------------------------------------------- #
# Re-calculate principal components
# ---------------------------------------------------------------- #
HFpEFNewpca <- princomp(HFpEFhiKmeans, cor = T)
HFmrEFNewpca <- princomp(HFmrEFhiKmeans, cor = T)

# ---------------------------------------------------------------- #
# Plot clusters
# ---------------------------------------------------------------- #
clustNewPef <- pca.cluster.plot(HFpEFNewpca, 2, km.clust = 3,
                                hc.clust = 3, em.clust = 3,
```

```
                                    return.clust = T, ellipse = F)

clustNewMr <- pca.cluster.plot(HFmrEFNewpca, 2, km.clust = 3,
                                   hc.clust = 3, em.clust = 3,
                                   return.clust = T, ellipse = F)

# -------------------------------------------------------------------- #
# Compare baseline characteristics HFpEF
# -------------------------------------------------------------------- #
HCpEFnoPhy <- clustNewPef$HC
KMpEFnoPhy <- clustNewPef$KM
EMpEFnoPhy <- clustNewPef$EM

compare.baseline(cbind(HFpEFhiKmeans,HCpEFnoPhy),"HCpEFnoPhy")
compare.baseline(cbind(HFpEFhiKmeans,KMpEFnoPhy),"KMpEFnoPhy")
compare.baseline(cbind(HFpEFhiKmeans,EMpEFnoPhy),"EMpEFnoPhy")

# -------------------------------------------------------------------- #
# Compare baseline characteristics HFmrEF
# -------------------------------------------------------------------- #
HCmrEFnoPhy <- clustNewMr$HC
KMmrEFnoPhy <- clustNewMr$KM
EMmrEFnoPhy <- clustNewMr$EM

compare.baseline(cbind(HFmrEFhiKmeans,HCmrEFnoPhy),
                 "HCmrEFnoPhy")
compare.baseline(cbind(HFmrEFhiKmeans,KMmrEFnoPhy),
                 "KMmrEFnoPhy")
compare.baseline(cbind(HFmrEFhiKmeans,EMmrEFnoPhy),
                 "EMmrEFnoPhy")

# -------------------------------------------------------------------- #
```

B.6 Classification, classification.R

```
# -------------------------------------------------------------------- #
# Install relevant packages (if not already done)
# -------------------------------------------------------------------- #
Packages <- c("mlbench", "caret", "elasticnet", "klarR")
# install.packages(Packages)

# -------------------------------------------------------------------- #
# Load relevant packages
# -------------------------------------------------------------------- #
lapply(Packages, library, character.only = TRUE)
```

```r
source("utilities.R")

# ------------------------------------------------------------------ #
# Load data files
# ------------------------------------------------------------------ #
allDataFiles <- c("HFfullImp", "HFfullOutcomes")
lapply(gsub(" ", "", paste("data_files/", allDataFiles,
                           ".Rdat")), load,.GlobalEnv)

# ------------------------------------------------------------------ #
# Add cross validation configuration
# ------------------------------------------------------------------ #
kfold <- trainControl(method = "cv", number = 5)
seed <- 902109
metric <- "Accuracy"

# ------------------------------------------------------------------ #
# Train and evaluate the classification algorithms with kfold
# ------------------------------------------------------------------ #
dataset <- HFfullImp[,-1]
mortality <- HFfullOutcomes[,3]
readmission <- HFfullOutcomes[,4]

# ------------------------------------------------------------------ #
# Mortality
# ------------------------------------------------------------------ #
# kfold CV evaluation of classifiers
# ------------------------------------------------------------------ #
set.seed(seed)
fitKnnKfoldMort <- train(dataset, mortality, method="knn",
                         metric=metric, trControl=kfold)

set.seed(seed)
fitLLKfoldMort <- train(dataset, mortality, method = "glm",
                        metric=metric, trControl = kfold)

set.seed(seed)
fitLDAKfoldMort <- train(dataset, mortality, method = "lda",
                         metric = metric, trControl = kfold)

set.seed(seed)
fitNbKfoldMort <- train(dataset, mortality, method = "nb",
                        metric = metric, trControl = kfold)

set.seed(seed)
```

```
fitSvmKfoldMort <- train(dataset, mortality,method="svmRadial",
                          metric=metric, trControl=kfold)

set.seed(seed)
fitRfKfoldMort <- train(dataset, mortality, method="rf",
                        metric = metric, trControl = kfold)

# ---------------------------------------------------------------- #
# Produce summary statistics and plots
# ---------------------------------------------------------------- #
# Kfold CV
# ---------------------------------------------------------------- #
resultsMortalityKfold <- resamples(list(knn = fitKnnKfoldMort,
                                        logr = fitLLKfoldMort,
                                        lda = fitLDAKfoldMort,
                                        nb = fitNbKfoldMort,
                                        svm = fitSvmKfoldMort,
                                        rf = fitRfKfoldMort))

summary(resultsMortalityKfold); dotplot(resultsMortalityKfold)

# ---------------------------------------------------------------- #
# Readmission
# ---------------------------------------------------------------- #
# kfold CV evaluation of classifiers
# ---------------------------------------------------------------- #
set.seed(seed)
fitKnnKfoldReadm <- train(dataset, readmission, method="knn",
                          metric=metric, trControl=kfold)

set.seed(seed)
fitLLKfoldReadm <- train(dataset, readmission, method = "glm",
                         metric=metric, trControl = kfold)

set.seed(seed)
fitLDAKfoldReadm <- train(dataset[,-19], readmission,
                          method = "lda", metric = metric,
                          trControl = kfold)

set.seed(seed)
fitNbKfoldReadm <- train(dataset, readmission, method = "nb",
                         metric = metric, trControl = kfold)

set.seed(seed)
fitSvmKfoldReadm <- train(dataset, readmission,
```

```r
                              method="svmRadial", metric=metric,
                              trControl=kfold)

set.seed(seed)
fitRfKfoldReadm <- train(dataset, readmission, method="rf",
                         metric = metric, trControl = kfold)

# ------------------------------------------------------------- #
# Produce summary statistics and plots
# ------------------------------------------------------------- #
# Kfold CV
# ------------------------------------------------------------- #
resultsReadmKfold <- resamples(list(knn = fitKnnKfoldReadm,
                                    lda = fitLDAKfoldReadm,
                                    nb = fitNbKfoldReadm,
                                    logr = fitLLKfoldReadm,
                                    svm = fitSvmKfoldReadm,
                                    rf = fitRfKfoldReadm))

summary(resultsReadmKfold); dotplot(resultsReadmKfold)

# ------------------------------------------------------------- #
```

Bibliography

Acharya, U. R., Fujita, H., Sudarshan, V. K., Oh, S. L., Muhammad, A., Koh, J. E., Tan, J. H., Chua, C. K., Chua, K. P., and San Tan, R. (2017). Application of empirical mode decomposition (emd) for automated identification of congestive heart failure using heart rate signals. *Neural Computing and Applications*, 28(10):3073–3094. Springer.

Aha, D. W., Kibler, D., and Albert, M. K. (1991). Instance-based learning algorithms. *Machine learning*, 6(1):37–66. Springer.

Ahmad, T., Desai, N., Wilson, F., Schulte, P., Dunning, A., Jacoby, D., Allen, L., Fiuzat, M., Rogers, J., Felker, G. M., et al. (2016). Clinical implications of cluster analysis-based classification of acute decompensated heart failure and correlation with bedside hemodynamic profiles. *PloS one*, 11(2):e0145881. Public Library of Science.

Ahmad, T., Pencina, M. J., Schulte, P. J., O'Brien, E., Whellan, D. J., Piña, I. L., Kitzman, D. W., Lee, K. L., O'Connor, C. M., and Felker, G. M. (2014). Clinical implications of chronic heart failure phenotypes defined by cluster analysis. *Journal of the American College of Cardiology*, 64(17):1765–1774. Elsevier.

Allison, P. D. (1999). *Missing data*. Sage Publications, Inc. Thousand Oaks, California.

Alonso-Betanzos, A., Bolón-Canedo, V., Heyndrickx, G. R., and Kerkhof, P. L. (2015). Exploring guidelines for classification of major heart failure subtypes by using machine learning. *Clinical Medicine Insights: Cardiology*, 9:CMC–S18746. SAGE Publications Sage UK: London, England.

Aune, E., Baekkevar, M., Roislien, J., Rodevand, O., and Otterstad, J. E. (2009). Normal reference ranges for left and right atrial volume indexes and ejection fractions obtained with real-time three-dimensional echocardiography. *European Journal of Echocardiography*, 10(6):738–744. Oxford University Press.

Austin, P. C., Lee, D. S., Steyerberg, E. W., and Tu, J. V. (2012). Regression trees for predicting mortality in patients with cardiovascular disease: What improvement is achieved by using ensemble-based methods? *Biometrical journal*, 54(5):657–673. Wiley Online Library.

Austin, P. C., Tu, J. V., Ho, J. E., Levy, D., and Lee, D. S. (2013). Using methods from the data-mining and machine-learning literature for disease classification and prediction: a case study examining classification of heart failure subtypes. *Journal of clinical epidemiology*, 66(4):398–407. Elsevier.

Awan, S. E., Sohel, F., Sanfilippo, F. M., Bennamoun, M., and Dwivedi, G. (2018). Machine learning in heart failure: ready for prime time. *Current opinion in cardiology*, 33(2):190–195. LWW.

Beaujean, A. A. (2012). *BaylorEdPsych: R Package for Baylor University Educational Psychology Quantitative Courses*. https://CRAN.R-project.org/package=BaylorEdPsych.

Beringer, J. Y. and Kerkhof, P. L. (1998). A unifying representation of ventricular volumetric indexes. *IEEE transactions on biomedical engineering*, 45(3):365–371. IEEE.

Braunwald, E. (2015). The war against heart failure: the lancet lecture. *The Lancet*, 385(9970):812–824. Elsevier.

Burgette, L. F. and Reiter, J. P. (2010). Multiple imputation for missing data via sequential regression trees. *American journal of epidemiology*, 172(9):1070–1076.

Buuren, S. v. and Groothuis-Oudshoorn, K. (2010). mice: Multivariate imputation by chained equations in r. *Journal of statistical software*, pages 1–68.

Carlson, K. J., Lee, D. C.-S., Goroll, A. H., Leahy, M., and Johnson, R. A. (1985). An analysis of physicians' reasons for prescribing long-term digitalis therapy in outpatients. *Journal of chronic diseases*, 38(9):733–739. Elsevier.

Charrad, M., Ghazzali, N., Boiteau, V., and Niknafs, A. (2014). NbClust: An R package for determining the relevant number of clusters in a data set. *Journal of Statistical Software*, 61(6):1–36.

Cheng, R. K., Cox, M., Neely, M. L., Heidenreich, P. A., Bhatt, D. L., Eapen, Z. J., Hernandez, A. F., Butler, J., Yancy, C. W., and Fonarow, G. C. (2014). Outcomes in patients with heart failure with preserved, borderline, and reduced ejection fraction in the medicare population. *American heart journal*, 168(5):721–730. Elsevier.

Cikes, M. and Solomon, S. D. (2015). Beyond ejection fraction: an integrative approach for assessment of cardiac structure and function in heart failure. *European heart journal*, 37(21):1642–1650. Oxford University Press.

Cowie, M. R., Struthers, A. D., Wood, D. A., Coats, A. J., Thompson, S. G., Poole-Wilson, P. A., and Sutton, G. C. (1997). Value of natriuretic peptides in assessment of patients with possible new heart failure in primary care. *The Lancet*, 350(9088):1349–1353. Elsevier.

Cox, D. R. (1972). Regression models and life-tables. *Journal of the Royal Statistical Society. Series B (Methodological)*, 34(2):87–22.

Crespo-Leiro, M. G., Anker, S. D., Maggioni, A. P., Coats, A. J., Filippatos, G., Ruschitzka, F., Ferrari, R., Piepoli, M. F., Delgado Jimenez, J. F., Metra, M., et al. (2016). European society of cardiology heart failure long-term registry (esc-hf-lt): 1-year follow-up outcomes and differences across regions. *European journal of heart failure*, 18(6):613–625. Wiley Online Library.

Defays, D. (1977). An efficient algorithm for a complete link method. *The Computer Journal*, 20(4):364–366.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society. Series B (methodological)*, pages 1–38. JSTOR.

Deo, R. C. (2015). Machine learning in medicine. *Circulation*, 132(20):1920–1930. Am Heart Assoc.

Dunderdale, K., Thompson, D. R., Miles, J. N., Beer, S. F., and Furze, G. (2005). Quality-of-life measurement in chronic heart failure: do we take account of the patient perspective? *European journal of heart failure*, 7(4):572–582. Wiley Online Library.

Eekhout, I., de Boer, M. R., Twisk, J. W., de Vet, H. C., and Heymans, M. W. (2012). Brief report: Missing data: A systematic review of how they are reported and handled. *Epidemiology*, pages 729–732. JSTOR.

Efron, B. (1992). Bootstrap methods: another look at the jackknife. In *Breakthroughs in statistics*, pages 569–593. Springer.

Enders, C. K. (2010). *Applied missing data analysis*. Guilford press.

Eriksson, H., Caidaul, K., Larsson, B., Ohlson, L.-O., Welin, L., Wilhelmsen, L., and Svärdsudd, K. (1987). Cardiac and pulmonary causes of dyspnoea—validation of a scoring test for clinical-epidemiological use: the study of men born in 1913. *European heart journal*, 8(9):1007–1014. Oxford University Press.

Ernst, G. (2016). *Heart rate variability*. Springer.

Fix, E. and Hodges Jr, J. L. (1951). Discriminatory analysis-nonparametric discrimination: consistency properties. Technical report, California Univ Berkeley.

Fleg, J. L., Piña, I. L., Balady, G. J., Chaitman, B. R., Fletcher, B., Lavie, C., Limacher, M. C., Stein, R. A., Williams, M., and Bazzarre, T. (2000). Assessment of functional capacity in clinical and research applications: An advisory from the committee on exercise, rehabilitation, and prevention, council on clinical cardiology, american heart association. *Circulation*, 102(13):1591–1597. Am Heart Assoc.

Forgy, E. W. (1965). Cluster analysis of multivariate data: efficiency versus interpretability of classifications. *biometrics*, 21:768–769.

Friedman, J., Hastie, T., and Tibshirani, R. (2009). *The elements of statistical learning: data mining, inference, and prediction*, volume 2. Springer series in statistics New York.

Fuat, A., Murphy, J. J., Hungin, A. P. S., Curry, J., Mehrzad, A. A., Hetherington, A., Johnston, J. I., Smellie, W. S. A., Duffy, V., and Cawley, P. (2006). The diagnostic accuracy and utility of a b-type natriuretic peptide test in a community population of patients with suspected heart failure. *Br J Gen Pract*, 56(526):327–333. British Journal of General Practice.

Geisser, S. (1975). The predictive sample reuse method with applications. *Journal of the American statistical Association*, 70(350):320–328.

Gharehcho-pogh, F. S. and Khalifelu, Z. A. (2011). Neural network application in diagnosis of patient: a case study. In *Computer Networks and Information Technology (ICCNIT), 2011 International Conference on*, pages 245–249. IEEE.

Grossman, W. (1990). Diastolic dysfunction and congestive heart failure. *Circulation*, 81(2 Suppl):III1–7.

Hay, S. et al. (2017). Global, regional, and national incidence, prevalence, and years lived with disability for 328 diseases and injuries for 195 countries, 1990–2016: a systematic analysis for the global burden of disease study 2016. *The Lancet*, 390(10100):1211 – 1259. Elsevier.

Henein, M. Y. (2010). *Heart failure in clinical practice*. Springer.

Henze, N. and Zirkler, B. (1990). A class of invariant consistent tests for multivariate normality. *Communications in Statistics-Theory and Methods*, 19(10):3595–3617.

Ho, T. K. (1995). Random decision forests. In *Document analysis and recognition, 1995., proceedings of the third international conference on*, volume 1, pages 278–282. IEEE.

Honaker, J., King, G., Blackwell, M., et al. (2011). Amelia ii: A program for missing data. *Journal of statistical software*.

Hsu, J. J., Ziaeian, B., and Fonarow, G. C. (2017). Heart failure with mid-range (borderline) ejection fraction: Clinical implications and future directions. *JACC: Heart Failure*. Elsevier.

Hunt, S. A., Baker, D. W., Chin, M. H., Cinquegrani, M. P., Feldmanmd, A. M., Francis, G. S., Ganiats, T. G., Goldstein, S., Gregoratos, G., Jessup, M. L., et al. (2001). Acc/aha guidelines for the evaluation and management of chronic heart failure in the adult: executive summary a report of the american college of cardiology/american heart association task force on practice guidelines (committee to revise the 1995 guidelines for the evaluation and management of heart failure). *Circulation*, 104(24):2996–3007. Am Heart Assoc.

Ibrahim, J. G., Chu, H., and Chen, M.-H. (2012). Missing data in clinical studies: issues and methods. *Journal of clinical oncology*, 30(26):3297. American Society of Clinical Oncology.

Inamdar, A. A. and Inamdar, A. C. (2016). Heart failure: diagnosis, management and utilization. *Journal of clinical medicine*, 5(7):62. Multidisciplinary Digital Publishing Institute.

Isler, Y. (2016). Discrimination of systolic and diastolic dysfunctions using multi-layer perceptron in heart rate variability analysis. *Computers in biology and medicine*, 76:113–119. Elsevier.

James, G., Witten, D., Hastie, T., and Tibshirani, R. (2013). *An introduction to statistical learning*, volume 112. Springer.

Kao, D. P., Lewsey, J. D., Anand, I. S., Massie, B. M., Zile, M. R., Carson, P. E., McKelvie, R. S., Komajda, M., McMurray, J. J., and Lindenfeld, J. (2015). Characterization of subgroups of heart failure patients with preserved ejection fraction with possible implications for prognosis and treatment response. *European journal of heart failure*, 17(9):925–935. Wiley Online Library.

Kaplan, E. L. and Meier, P. (1958). Nonparametric estimation from incomplete observations. *Journal of the American statistical association*, 53(282):457–481. Taylor & Francis.

Katz, D. H., Deo, R. C., Aguilar, F. G., Selvaraj, S., Martinez, E. E., Beussink-Nelson, L., Kim, K.-Y. A., Peng, J., Irvin, M. R., Tiwari, H., et al. (2017). Phenomapping for the identification of hypertensive patients with the myocardial substrate for heart failure with preserved ejection fraction. *Journal of cardiovascular translational research*, 10(3):275–284. Springer.

Kaushal, S. (2014). Missing data in clinical trials: Pitfalls and remedies. *International journal of applied & basic medical research*, 4(Suppl 1):S6–7. Medknow Publications.

Kelly, J. P., Mentz, R. J., Mebazaa, A., Voors, A. A., Butler, J., Roessig, L., Fiuzat, M., Zannad, F., Pitt, B., O'Connor, C. M., et al. (2015). Patient selection in heart failure with preserved ejection fraction clinical trials. *Journal of the American College of Cardiology*, 65(16):1668–1682. Elsevier.

Ketchum, E. S. and Levy, W. C. (2011). Multivariate risk scores and patient outcomes in advanced heart failure. *Congestive Heart Failure*, 17(5):205–212. Wiley Online Library.

Koulaouz-idis, G., Iakovidis, D., and Clark, A. (2016). Telemonitoring predicts in advance heart failure admissions. *International journal of cardiology*, 216:78–84. Elsevier.

Krishnaswamy, P., Lubien, E., Clopton, P., Koon, J., Kazanegra, R., Wanner, E., Gardetto, N., Garcia, A., DeMaria, A., and Maisel, A. S. (2001). Utility of b-natriuretic peptide levels in identifying patients with left ventricular systolic or diastolic dysfunction. *The American journal of medicine*, 111(4):274–279. Elsevier.

Kruskal, W. H. and Wallis, W. A. (1952). Use of ranks in one-criterion variance analysis. *Journal of the American statistical Association*, 47(260):583–621.

Kuhn, M., Wing, J., Weston, S., Williams, A., Keefer, C., Engelhardt, A., Cooper, T., Mayer, Z., Kenkel, B., the R Core Team, Benesty, M., Lescarbeau, R., Ziem, A., Scrucca, L., Tang, Y., Candan, C., and Hunt, T. (2018). *caret: Classification and Regression Training*. R package version 6.0-79.

Lam, C. S. and Solomon, S. D. (2014). The middle child in heart failure: heart failure with mid-range ejection fraction (40–50%). *European journal of heart failure*, 16(10):1049–1055. Wiley Online Library.

Lê, S., Josse, J., and Husson, F. (2008). FactoMineR: A package for multivariate analysis. *Journal of Statistical Software*, 25(1):1–18.

Lee, D. S., Austin, P. C., Rouleau, J. L., Liu, P. P., Naimark, D., and Tu, J. V. (2003). Predicting mortality among patients hospitalized for heart failure:

derivation and validation of a clinical model. *Jama*, 290(19):2581–2587. American Medical Association.

Levy, W. C., Mozaffarian, D., Linker, D. T., Sutradhar, S. C., Anker, S. D., Cropp, A. B., Anand, I., Maggioni, A., Burton, P., Sullivan, M. D., et al. (2006). The seattle heart failure model: prediction of survival in heart failure. *Circulation*, 113(11):1424–1433. Am Heart Assoc.

Liaw, A. and Wiener, M. (2002). Classification and regression by randomforest. *R News*, 2(3):18–22.

Little, R. J. (1988). A test of missing completely at random for multivariate data with missing values. *Journal of the American Statistical Association*, 83(404):1198–1202. Taylor & Francis.

Liu, G., Wang, L., Wang, Q., Zhou, G., Wang, Y., and Jiang, Q. (2014). A new approach to detect congestive heart failure using short-term heart rate variability measures. *PloS one*, 9(4):e93399. Public Library of Science.

Maisel, A., Mueller, C., Adams, K., Anker, S. D., Aspromonte, N., Cleland, J. G., Cohen-Solal, A., Dahlstrom, U., DeMaria, A., Di Somma, S., et al. (2008). State of the art: using natriuretic peptide levels in clinical practice. *European journal of heart failure*, 10(9):824–839. Wiley Online Library.

Mardia, K. V. (1970). Measures of multivariate skewness and kurtosis with applications. *Biometrika*, 57(3):519–530.

Masetic, Z. and Subasi, A. (2016). Congestive heart failure detection using random forest classifier. *Computer methods and programs in biomedicine*, 130:54–64. Elsevier.

McKee, P. A., Castelli, W. P., McNamara, P. M., and Kannel, W. B. (1971). The natural history of congestive heart failure: the framingham study. *New England Journal of Medicine*, 285(26):1441–1446. Mass Medical Soc.

McMurray, J. J., Adamopoulos, S., Anker, S. D., Auricchio, A., Böhm, M., Dickstein, K., Falk, V., Filippatos, G., Fonseca, C., et al. (2012). Esc guidelines for the diagnosis and treatment of acute and chronic heart failure 2012: The task force for the diagnosis and treatment of acute and chronic heart failure 2012 of the european society of cardiology.

developed in collaboration with the heart failure association (hfa) of the esc. *European heart journal*, 33(14):1787–1847. Oxford University Press.

Meyer, D., Dimitriadou, E., Hornik, K., Weingessel, A., and Leisch, F. (2018). *e1071: Misc Functions of the Department of Statistics, Probability Theory Group (Formerly: E1071), TU Wien*. R package version 1.7-0.

Moons, K. G., Donders, R. A., Stijnen, T., and Harrell, F. E. (2006). Using the outcome for imputation of missing predictor values was preferred. *Journal of clinical epidemiology*, 59(10):1092–1101. Elsevier.

Murphy, K. P. (2012). *Machine learning, a probabilistic perspective*. The MIT press.

Myers, W. R. (2000). Handling missing data in clinical trials: an overview. *Drug Information Journal*, 34(2):525–533. SAGE Publications Sage CA: Los Angeles, CA.

Nagueh, S. F., Appleton, C. P., Gillebert, T. C., Marino, P. N., Oh, J. K., Smiseth, O. A., Waggoner, A. D., Flachskampf, F. A., Pellikka, P. A., and Evangelista, A. (2009). Recommendations for the evaluation of left ventricular diastolic function by echocardiography. *Journal of the American Society of Echocardiography*, 22(2):107–133. Elsevier.

Narin, A., Isler, Y., and Ozer, M. (2014). Investigating the performance improvement of hrv indices in chf using feature selection methods based on backward elimination and statistical significance. *Computers in biology and medicine*, 45:72–79. Elsevier.

NYHA (1994). The criteria committee of the new york heart association - nomenclature and criteria for diagnosis of diseases of the heart and great vessels. *Little, Brown Medical Division*, 7:253–256. Boston, Mass.

Panahiazar, M., Taslimitehrani, V., Pereira, N., and Pathak, J. (2015). Using ehrs and machine learning for heart failure survival analysis. *Studies in health technology and informatics*, 216:40.

Pandit, K., Mukhopadhyay, P., Ghosh, S., and Chowdhury, S. (2011). Natriuretic peptides: Diagnostic and therapeutic use. *Indian journal of endocrinology and metabolism*, 15(Suppl4):S345.

Peterson, P. N., Rumsfeld, J. S., Liang, L., Albert, N. M., Hernandez, A. F., Peterson, E. D., Fonarow, G. C., Masoudi, F. A., et al. (2010). A validated risk score for in-hospital mortality in patients with heart failure from the american heart association get with the guidelines program. *Circulation: Cardiovascular Quality and Outcomes*, 3(1):25–32. Am Heart Assoc.

Ponikowski, P., Voors, A. A., Anker, S. D., Bueno, H., Cleland, J. G., Coats, A. J., Falk, V., González-Juanatey, J. R., Harjola, V.-P., Jankowska, E. A., et al. (2016). 2016 esc guidelines for the diagnosis and treatment of acute and chronic heart failure: The task force for the diagnosis and treatment of acute and chronic heart failure of the european society of cardiology (esc) developed with the special contribution of the heart failure association (hfa) of the esc. *European heart journal*, 37(27):2129–2200. Oxford University Press.

R Core Team (2018a). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. http://www.R-project.org/.

R Core Team (2018b). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Roger, V. L. (2010). The heart failure epidemic. *International journal of environmental research and public health*, 7(4):1807–1830.

Rohlf, F. J. (1982). 12 single-link clustering algorithms. *Handbook of statistics*, 2:267–284.

Royston, J. (1982). An extension of shapiro and wilk's w test for normality to large samples. *Applied Statistics*, pages 115–124.

RStudio Team (2018). *RStudio: Integrated Development Environment for R*. RStudio, Inc., Boston, MA. http://www.rstudio.com/.

Rubin, D. B. (1976). Inference and missing data. *Biometrika*, 63(3):581–592. Oxford University Press.

Savarese, G. and Lund, L. H. (2017). Global public health burden of heart failure. *Cardiac failure review*, 3(1):7. Radcliffe Cardiology.

Schafer, J. L. (1997). *Analysis of incomplete multivariate data*. Chapman and Hall/CRC.

Schafer, J. L. and Graham, J. W. (2002). Missing data: our view of the state of the art. *Psychological methods*, 7(2):147. American Psychological Association.

Schafer, J. L. and Olsen, M. K. (1998). Multiple imputation for multivariate missing-data problems: A data analyst's perspective. *Multivariate behavioral research*, 33(4):545–571.

Scheffer, J. (2002). Dealing with missing data. *Massey University*.

Scrucca, L., Fop, M., Murphy, T. B., and Raftery, A. E. (2017). mclust 5: clustering, classification and density estimation using Gaussian finite mixture models. *The R Journal*, 8(1):205–233.

Shah, S. J., Katz, D. H., Selvaraj, S., Burke, M. A., Yancy, C. W., Gheorghiade, M., Bonow, R. O., Huang, C.-C., and Deo, R. C. (2014). Phenomapping for novel classification of heart failure with preserved ejection fraction. *Circulation*, pages CIRCULATIONAHA–114. Am Heart Assoc.

Sibson, R. (1973). Slink: an optimally efficient algorithm for the single-link cluster method. *The computer journal*, 16(1):30–34.

Son, C.-S., Kim, Y.-N., Kim, H.-S., Park, H.-S., and Kim, M.-S. (2012). Decision-making model for early diagnosis of congestive heart failure using rough set and decision tree approaches. *Journal of biomedical informatics*, 45(5):999–1008. Elsevier.

Sterne, J. A., White, I. R., Carlin, J. B., Spratt, M., Royston, P., Kenward, M. G., Wood, A. M., and Carpenter, J. R. (2009). Multiple imputation for missing data in epidemiological and clinical research: potential and pitfalls. *Bmj*, 338:b2393. British Medical Journal Publishing Group.

Swedberg, K., Cleland, J., Dargie, H., Drexler, H., Follath, F., Komajda, M., Tavazzi, L., Smiseth, O. A., Gavazzi, A., Haverich, A., et al. (2005). Guidelines for the diagnosis and treatment of chronic heart failure: executive summary (update 2005) the task force for the diagnosis and treatment of chronic heart failure of the european society of cardiology. *European heart journal*, 26(11):1115–1140. Oxford University Press.

Tan, P.-N. et al. (2007). *Introduction to data mining*. Pearson Education India.

Tharwat, A., Gaber, T., Ibrahim, A., and Hassanien, A. E. (2017). Linear discriminant analysis: A detailed tutorial. *AI communications*, 30(2):169–190.

Tripoliti, E. E., Papadopoulos, T. G., Karanasiou, G. S., Naka, K. K., and Fotiadis, D. I. (2017). Heart failure: diagnosis, severity estimation and prediction of adverse events through machine learning techniques. *Computational and structural biotechnology journal*, 15:26–47. Elsevier.

van Ravenswaaij-Arts, C. M., Kollee, L. A., Hopman, J. C., Stoelinga, G. B., and van Geijn, H. P. (1993). Heart rate variability. *Annals of internal medicine*, 118(6):436–447. Am Coll Physicians.

Vapnik, V. (1963). Pattern recognition using generalized portrait method. *Automation and remote control*, 24:774–780.

Wang, T. J., Larson, M. G., Levy, D., Benjamin, E. J., Leip, E. P., Omland, T., Wolf, P. A., and Vasan, R. S. (2004). Plasma natriuretic peptide levels and the risk of cardiovascular events and death. *New England Journal of Medicine*, 350(7):655–663. Mass Medical Soc.

Wasserman, L. (2013). *All of statistics: a concise course in statistical inference*. Springer Science & Business Media.

Yamamoto, K., Burnett, J. C., Bermudez, E. A., Jougasaki, M., Bailey, K. R., and Redfield, M. M. (2000). Clinical criteria and biochemical markers for the detection of systolic dysfunction. *Journal of cardiac failure*, 6(3):194–200. Elsevier.

Yancy, C. W., Jessup, M., Bozkurt, B., Butler, J., Casey, D. E., Drazner, M. H., Fonarow, G. C., Geraci, S. A., Horwich, T., Januzzi, J. L., et al. (2013). 2013 accf/aha guideline for the management of heart failure: a report of the american college of cardiology foundation/american heart association task force on practice guidelines. *Journal of the American College of Cardiology*, 62(16):e147–e239. Journal of the American College of Cardiology.

Yang, G., Ren, Y., Pan, Q., Ning, G., Gong, S., Cai, G., Zhang, Z., Li, L., and Yan, J. (2010). A heart failure diagnosis model based on support vector machine. In *Biomedical Engineering and Informatics (BMEI), 2010 3rd International Conference on*, volume 3, pages 1105–1108. IEEE.

Zaphiriou, A., Robb, S., Murray-Thomas, T., Mendez, G., Fox, K., McDonagh, T., Hardman, S., Dargie, H. J., and Cowie, M. R. (2005). The diagnostic accuracy of plasma bnp and ntprobnp in patients referred from primary care with suspected heart failure: results of the uk natriuretic peptide study. *European journal of heart failure*, 7(4):537–541. Wiley Online Library.

Zhang, Z., hospital, S. R.-R. S., and university school of medicine, Z. (2018). *CBCgrps: Compare Baseline Characteristics Between Groups*. R package version 2.3.

Zolfaghar, K., Meadem, N., Teredesai, A., Roy, S. B., Chin, S.-C., and Muckian, B. (2013). Big data solutions for predicting risk-of-readmission for congestive heart failure patients. In *Big Data, 2013 IEEE International Conference on*, pages 64–71. IEEE.