university of groningen / faculty of science and engineering / mathematics and applied mathematics

A sample size based comparison of the frequentist and Bayesian logistic regression

Bachelor's Project Mathematics

July 2021

Student: A.J. Tissing

First supervisor: prof. dr. M.A. Grzegorczyk

Second assessor: dr. W.P. Krijnen

Abstract

This research studies the performance of the frequentist and Bayesian logistic regression. When comparing the performance of both methods, the effect of the sample size is discussed, to see whether there is an effect in the difference in performance when the sample size is decreased. The measure of performance used in this research is the Area Under the Receiver Operating Characteristic curve, which measures a model's ability to classify data with a binary outcome variable. 5-fold cross-validation is used when doing this comparison of different methods. Before doing the sample size reduction, within each of the approaches, multiple methods are compared. For the frequentist approach, these are the AIC, BIC and p-values methods and for the Bayesian approach, these are the MCMC and RJMCMC algorithms. After comparing these, to study the performance when decreasing the sample size, the data set is split in half multiple times and both regression approaches are performed on each individual part. Throughout this research, the Framingham data set is used as a benchmark data set.

1 Introduction

In statistics, regression models are commonly used to describe the relation between explanatory variables and an outcome variable. There are many different types of regressions and every type comes with its own characteristics. One well-known type of regression is the linear regression. A generalisation of the linear regression is the Generalised Linear Model (GLM), which relates the mean of the outcome variable to the linear model through a monotone link function.

One of regression models that is common to use, is the logistic regression. The logistic regression relates the linear model with the mean through the logit link function, ensuring the mean to be between 0 and 1. This is especially useful, because many data sets have a binary outcome variable. For example, when looking at medical data, to determine whether a patients gets a certain disease, the outcome variable is either a 1, meaning the patient gets the disease, or a 0, meaning the patient does not get the disease. Other applications are for example risk estimations in business or predictions whether a patient's injuries will be lethal. But next to those, there are many other applications one can think of.

When doing a logistic regression, there is still a variety of methods that can be applied and there are even different paradigms that approach this regression in a totally different manner. These two paradigms are the frequentist approach and the Bayesian approach. Although there are many differences between the two approaches, one essential difference is that the frequentist approach strives to find the 'best' set of parameters through maximum likelihood estimation, while the Bayesian approach sees every parameter as a variable that has its own distribution and that does not necessarily have one fixed value in nature. To find be able to use this unknown distribution, the Bayesian approach samples from the distribution. Exact details on how exactly these approaches work, are described in this thesis.

This thesis studies the difference in performance between the frequentist and the Bayesian approach. On the Framingham data set, which functions as a benchmark data set, the two methods are compared to see how well the approaches are able to classify the data. When doing this comparison, the sample size of the data set is halved five times, to establish whether the sample size influences the performance of both methods. This is the main focus of this research, to determine whether there is a difference in performance between the two approaches and to establish whether this difference is dependent on sample size. For this reason, the research question is as follows:

> *Is there a difference in performance between the frequentist and Bayesian logistic regression and does the difference in performance depend on the sample size?*

Within both the frequentist and Bayesian methods, many methods can be used. For the frequentist approach, model selection methods that are considered are the Akaike Information Criterion (AIC), the Bayesian Information Criterion (BIC) and the method of p-values. For the determination of parameters for the Bayesian method, the Markov Chain Monte Carlo (MCMC) algorithm and an alternative that automatically does model selection, the Reversible Jump Markov Chain Monte Carlo (RJMCMC) algorithm, are discussed.

1.1 Literature review

A lot of literature has been published on both approaches. However, most literature does not discuss differences in performance of the two approaches, but rather the different intuition behind both approaches and other advantages. Dobson and Barnett [2018] mention that the Bayesian approach allows ideal definitions of the p-value and the confidence interval, contrary to the frequentist methods. For example, it states that the p-value in Bayesian methods truly means the probability that the null hypothesis is true, where the p-value in frequentist methods is the probability of observing more extreme data given that the null hypothesis is correct. Bayarri and Berger [2004] argue that there are, in fact, differences between the Bayesian and frequentist methodologies and that it is context-dependent to determine which method is superior. Furthermore, Samaniego [2010] states that Bayesian and frequentist approaches behave in a similar manner asymptotically, so when the sample size gets big enough, the difference between the two methods disappears. Wakefield [2013] argues that inferences of both methods are mostly equivalent, but that the Bayesian approach gives better results for small samples. This is because the frequentist approach is not able to give a good confidence interval in this case, while this is no issue for the Bayesian method. Next to that, McNeish [2016] states that Bayesian methods have properties that make them better for modelling small data sets, but that it is important to take the prior distributions into account. McNeish also mentions that methods are able to outperform frequentist methods because they are less biased. However, despite these discussed advantages, the writer does not favour one of the two methods above the other. McNeish does warn that if the prior contains incorrect information, this might give worse results than in the frequentist methods, as samples might not contain enough information to override the prior distribution.

It becomes clear that a lot has been written about the differences between the two approaches. However, most literature does not specify one of them to be superior to the other. Therefore, it will be interesting to see whether it is possible to fix a relationship between the sample size of the data set and the performance. Many different methods are used to make this comparison of performance. In this bachelor's thesis, the focus is on how well the logistic regression performs at classifying the data. Hosmer Jr et al. [2013] describe the Area Under the Receiver Operating Characteristic Curve (AUC) as a good method for doing this. According to these researchers, this is a unit of measure that specifies how well the model discriminates between subjects with and without a 'success'. The AUC uses the predicted probabilities and compares these to the outcome variables in the data. For all different cut-points, it measures whether the the prediction is equal to the outcome, resulting in a number between 0.5 and 1. The optimal value is 1, while a value of 0.5 means that the model does not perform better than tossing a coin. When applying the AUC to separated test data, this provides a good way of measuring the fit of the model. This method of separating test data is called cross-validation. Browne [2000] discusses that cross-validation is a suitable method to find a method with a small amount of misclassifications. The writer also mentions that with cross-validation, it is possible to evaluate methods on small samples. This means that cross-validation is still useful when decreasing the sample size of the data set used in this research.

Looking at the application of the two approaches, there are multiple methods, each with its own advantages. When determining the parameters of the Bayesian model, a normal MCMC algorithm can only find parameter values for a fixed set of covariates. As an alternative, Green [1995] describes the RJMCMC algorithm, using which it is possible

to jump between different models, which extends the possibilities of the MCMC algorithm to model selection. This has the potential of saving computational power, as the normal MCMC algorithm has to be performed many times for different possible models to find an efficient parsimonious model. For the frequentist approach, multiple methods to perform model selection exist as well. Two of these methods are, like mentioned before, the AIC and BIC. Akaike [1974] describes a method for model identification, 'punishing' models with many parameters, called the AIC. The BIC is an alternative to the AIC with similar features. It has a higher penalty for additional covariates. A model with a lower AIC or BIC is considered to be a better, sparser, model. According to Gelman et al. [2013] and Gelman et al. [2014], the fact that the BIC has a higher penalty makes it less useful for predicting values outside of the model, which could ben an issue when using cross-validation. For this reason, it can be expected that in this research the AIC gives better predictions than the BIC and that the AIC is the preferred method.

1.2 Conclusion

Based on this literature review, it can be concluded that there exists enough basis to do this research. Enough studies emphasise the differences between the Bayesian and frequentist approach, but these studies do not give a clear picture of the effect of sample size in both methods for the logistic regression. Therefore, there is reason enough to focus on this specific aspect in this bachelor's project. Looking at the literature, a reasonable hypothesis would be to assume that the methods behave similarly for a large data set, but that the Bayesian approach becomes superior when the sample size is decreased.

1.3 Overview of Thesis

This thesis is structured as follows. In Section 2, a description of the Framingham data set is given and the process of preparing the data for both methods is described. Then, in Section 3, the theoretical framework of both methods is given. This section also discusses the measure of performance and the process of reducing the sample size. In Section 4, the results from all methods are described and compared. In Section 5, a conclusion is drawn from the results. And lastly, in Section 6, the processes in this thesis are discussed and possible improvements are proposed for future research. Also, the computational expense of especially the Bayesian methods is discussed.

2 Preparation of the data

Before the Framingham data set is analysed using both methods, some preparations need to be made in order to be able to properly analyse the data set. Especially the Bayesian logistic regression requires the data to be pre-processed. To make sure that both methods are still comparable, this is also done for the frequentist logistic regression.

First of all, it is necessary to take a look at the columns of the data. In Table 1, all column names are stated, as well as whether they are continuous, categorical or binary (which is a special case of categorical). The header of the data set, can be found in Appendix A.

Continuous	Binary	Categorical
age	male	education
cigsPerDay	currentSmoker	
totChol	BPMeds	
sysBP	prevalentStroke	
diaBP	prevalentHyp	
BMI	diabetes	
heartRate	TenYearCHD	
glucose		

Table 1: Classification of variables in the Framingham data set.

To get a better idea of what the data set looks like, in Table 2, an overview can be found about all variables. This overview includes a description of all covariates, what kind of inputs it can take and the mean value. To get a better feel of how the variables are distributed, the histograms of all variables can be found in Appendix B.

Here, a few things must be noted. First of all, TenYearCHD is the outcome variable, determining whether a patient gets the coronary heart disease within ten years. Second, some of the variables that are continuous, could also be classified as categorical. An example of this is the variable age. This is rounded off to a full amount of years, meaning that there could be around 100 possible categories, assuming people are usually between the age of 0 and 100. However, because there are so many of them, it is more reasonable to consider this variable to be continuous instead of categorical.

The education variable is considered as categorical. This variable has only four categories, being the numbers 1 to 4. In Section 2.1, the procedure for handling categorical variables is discussed.

2.1 Categorical variables

When working with continuous variables, it is usually clear what the ordering is in the data. For example, someone with age 90 is older than someone with age 30, so it is possible to look at the influence of age on the patients. However, in case of categorical variables, this distinction is not always so clear, for example in the case of 'colour'. Hence, another process needs to be taken, to differentiate between all cases of the categorical variable.

Name	Description	Minimum value	Maximum value	Mean
male	Indicates whether the patient is male or not	0	1	0.44
age	Age	32	70	49.55
education	Education level	1	4	1.98
currentSmoker	Indicates whether the patient smokes or not	0	1	0.49
cigsPerDay	Amount of cigarettes the patient smokes per day	0	70	9.03
BPMeds	Indicates whether the patient takes blood pressure medication	0	1	0.03
prevalentStroke	Indicates whether the patient has had a stroke in the past	0	1	0.006
prevalentHyp	Indicates whether the patient is hypertensive	0	1	0.31
diabetes	Indicates whether the instance has diabetes	0	1	0.03
totChol	Total cholesterol level	113	600	236.85
sysBP	Systolic blood pressure	83.5	295	132.37
diaBP	Diastolic blood pressure	48	142.5	82.92
BMI	Body Mass Index	15.54	56.8	25.78
heartRate	Heart rate	44	143	75.73
glucose	Glucose level	40	394	81.85
TenYearCHD	Indicates whether the patient got the Coronary Heart Disease within ten years	0	1	0.15

Table 2: Description of all variables in the Framingham data set with their minimum, maximum and mean value.

This is where *design variables* are used, as described by Hosmer Jr et al. [2013]. A categorical variable with k categories is converted into $k-1$ design variables. The process is as follows. Let the $k-1$ design variables be denoted with D_j. When an instance has the first category, $D_j = 0$ for all $j = 1, \ldots, k-1$. In all other cases, when an instance has the j-th category, $D_{j-1} = 1$ and all other design variables are equal to zero. In this case, it possible to make a distinction between all categories, giving them their own coefficient. The coefficient for the first category is then incorporated in the constant, this is considered the 'base'. For the education variable in the Framingham data set, the design variables are set as in Table 3. These are added to the data set and the original variable education is removed.

2.2 Standardising variables

According to Hosmer Jr et al. [2013], standardising the explanatory variables is common practice. One advantage of this, is that this makes it easier to specify priors in the Bayesian logistic regression, a concept that is discussed in Section 3. For this reason, all continuous

education	D_1	D_2	D_3
1	0	0	0
2	1	0	0
3	0	1	0
4	0	0	1

Table 3: Design variables for the education variable in the Framingham data set.

variables are standardised, to ensure that they have a mean of zero and a standard deviation equal to one. For binary variables, including the design variables of categorical variables, this does not happen.

2.3 Other preparations

The next preparation that is performed, is removing any null values. If an instance in the data has a null value in any category, this instance is removed completely. There are other methods to work with null values, but due to the fact that these are not in the scope of this research, these instances are not taken into consideration. From the 4240 initial instances, 582 instances were removed. This means that there are still 3658 instances left.

Another preparation is randomising the order of the data. Before analysing the data, the data is split up into a training and testing data set. The reason for doing this, is discussed in Section 3.6. Randomising the order of the data is then necessary, to prevent any unknown ordering of the data from having an effect on the results.

An R script has been written to do this whole process of preparing, randomising and splitting the data. This code is displayed in Appendix C. When this function is run, all steps above are performed and both the training and testing data can be accessed from the output list. One additional feature of this program, is that it is able to halve the data as well. When the sample size is decreased, the data set is first randomly shuffled and then a small part of the data set is returned. When halving the data set k times, the user can choose which of the 2^k bins to choose.

3 Methods

3.1 Logistic Regression

First of all, before discussing the frequentist and Bayesian approach, the logistic regression is discussed. Even though the methods of the two approaches are completely different, there is some common basis among the two logistic regressions. The theory in this description is based on Hosmer Jr et al. [2013].

In all types of regressions, the key quantity to estimate is the mean value of the outcome variable given the values of the explanatory variables. When Y is taken to be the outcome variable and $\mathbf{x}$ is taken to be the set of explanatory variables, the expression to estimate is called the conditional mean and is denoted by $E(Y|\mathbf{x})$. This is the expected value of Y, given $\mathbf{x}$.

In a linear regression, this value is estimated to be a linear combination of a set of parameters β and the explanatory variables $\mathbf{x}$, which is denoted as

$$E(Y|\mathbf{x}) = \mathbf{x}^T\beta = \beta_0 + \sum_{k=1}^{p} \beta_k x_k.$$

In this linear regression, the goal is to estimate the set of parameters β.

In the case of a binary outcome variable, the mean of outcome variable Y given the explanatory variables $\mathbf{x}$, $E(Y|\mathbf{x})$, must be on the interval from 0 to 1. This is because the outcome variable can only take values of 0 and 1, so the mean cannot be outside this interval. This is an issue for the linear regression, where $E(Y|\mathbf{x})$ can take any value between $-\infty$ and ∞. To avoid this issue, the logistic regression is used.

First of all, in the logistic regression a simplified notation is used, so the quantity $E(Y|\mathbf{x})$ is denoted as $\pi(\mathbf{x}) = P(Y = 1|\mathbf{x})$. This is called the probability of having a 'success', meaning the probability for the outcome variable to be equal to 1, given the set of explanatory variables. The logistic regression model, which ensures this probability to be between 0 and 1, is denoted as follows

$$\pi(\mathbf{x}) = \frac{e^{\mathbf{x}^T\beta}}{1 + e^{\mathbf{x}^T\beta}}. \tag{1}$$

The goal in this formula is to find the set of parameters β. For this, a transformation is used. Equation 1 is rewritten to the equation

$$g(\mathbf{x}) = log\left(\frac{\pi(\mathbf{x})}{1 - \pi(\mathbf{x})}\right) = \mathbf{x}^T\beta = \beta_0 + \sum_{k=1}^{p} \beta_k x_k.$$

$g(\mathbf{x})$ is called the link function and is in this case the logit function. $\mathbf{x} = (1, x_1, x_2, \ldots, x_p)^T$ is the vector of explanatory variables, together with a 1 term for the intercept. $\beta = (\beta_0, \beta_1, \ldots, \beta_p)^T$ is the vector of parameters, each corresponding to one explanatory variable. This equation has many of the same properties as the linear regression model and is used to build the model. In order to do this, both the frequentist and the Bayesian approach require a likelihood function to be defined.

Let's define the data to be independent observations $(\mathbf{x}_i, y_i)$ for $i = 1, \ldots, n$, where y_i is the binary outcome variable. When $y_i = 1$ for some i, the probability of this event happening is equal to $\pi(\mathbf{x}_i)$. When $y_i = 0$, the probability of this event happening is equal to $1 - \pi(\mathbf{x}_i)$. This means that, to describe the likelihood for any event to happen for an instance in the data $(\mathbf{x}_i, y_i)$ can be written as

$$\pi(\mathbf{x}_i)^{y_i} \cdot (1 - \pi(\mathbf{x}_i))^{1-y_i}, \tag{2}$$

which has the same value as the description above for any y_i. Because the instances in the data are assumed to be independent, to obtain the likelihood function for the full data set, every instance of Equation 2 is multiplied. This leads to the following likelihood function

$$L(\beta) = \prod_{i=1}^{n} \pi(\mathbf{x}_i)^{y_i} \cdot (1 - \pi(\mathbf{x}_i))^{1-y_i}, \tag{3}$$

which is a function of β, because $\pi(\mathbf{x}_i)$ is defined in Equation 1 in terms of β.

3.2 Frequentist Logistic Regression

The first approach that is discussed is the frequentist logistic regression. The theory in this description is based on Hosmer Jr et al. [2013].

In the frequentist approach, the method to find these parameters β is called the *maximum likelihood estimation*. The maximum likelihood estimator is the set of values for β that maximises the likelihood of obtaining the data. To find the maximum likelihood estimate, the value of β must be found such that Equation 3 is maximised. This is equivalent to maximising the log-likelihood, which is often an easier computation. The log-likelihood is obtained by taking the natural logarithm of Equation 3 and is given by

$$l(\beta) = \sum_{i=1}^{n} [y_i log(\pi(\mathbf{x}_i)) + (1 - y_i) log(1 - \pi(\mathbf{x}_i))]. \tag{4}$$

To maximise Equation 4, the derivative with respect to every parameter β_j for $j = 0, 1, \ldots, p$ is set equal to zero. This is done separately for β_0 and for β_j for $j = 1, 2, \ldots, p$. However, before doing this, Equation 4 first needs to be written in a form that includes β, using Equation 1.

$$
\begin{aligned}
l(\beta) &= \sum_{i=1}^{n} [y_i log(\pi(\mathbf{x}_i)) + (1-y_i) log(1-\pi(\mathbf{x}_i))] \\
&= \sum_{i=1}^{n} \left[y_i log\left(\frac{e^{\mathbf{x}_i^T \beta}}{1+e^{\mathbf{x}_i^T \beta}} \right) + (1-y_i) log\left(1 - \frac{e^{\mathbf{x}_i^T \beta}}{1+e^{\mathbf{x}_i^T \beta}} \right) \right] \\
&= \sum_{i=1}^{n} \left[y_i (\mathbf{x}_i^T \beta - log(1+e^{\mathbf{x}_i^T \beta})) + (1-y_i) log\left(\frac{1}{1+e^{\mathbf{x}_i^T \beta}} \right) \right] \\
&= \sum_{i=1}^{n} \left[y_i \mathbf{x}_i^T \beta - y_i log(1+e^{\mathbf{x}_i^T \beta}) + (1-y_i) \cdot -log(1+e^{\mathbf{x}_i^T \beta}) \right] \\
&= \sum_{i=1}^{n} \left[y_i \mathbf{x}_i^T \beta - log(1+e^{\mathbf{x}_i^T \beta}) \right]
\end{aligned}
$$

This is differentiated with respect to β_0. Note that $\mathbf{x}_i^T \beta = \beta_0 + \sum_{k=1}^{p} \beta_k x_{ik}$. Because β_0 represents the intercept, this is differentiated separately from the other parameters β_j.

$$
\begin{aligned}
\frac{\partial l(\beta)}{\partial \beta_0} &= \frac{\partial}{\partial \beta_0} \sum_{i=1}^{n} \left[y_i \mathbf{x}_i^T \beta - log(1+e^{\mathbf{x}_i^T \beta}) \right] \\
&= \sum_{i=1}^{n} \left[y_i \cdot 1 - \frac{1}{1+e^{\mathbf{x}_i^T \beta}} \cdot e^{\mathbf{x}_i^T \beta} \cdot 1 \right] \\
&= \sum_{i=1}^{n} \left[y_i - \frac{e^{\mathbf{x}_i^T \beta}}{1+e^{\mathbf{x}_i^T \beta}} \right] \\
&= \sum_{i=1}^{n} [y_i - \pi(\mathbf{x}_i)]
\end{aligned}
$$

When differentiating with respect to β_j for $j = 1, 2, \ldots, p$, the following is obtained.

$$
\begin{aligned}
\frac{\partial l(\beta)}{\partial \beta_j} &= \frac{\partial}{\partial \beta_j} \sum_{i=1}^{n} \left[y_i \mathbf{x}_i^T \beta - log(1+e^{\mathbf{x}_i^T \beta}) \right] \\
&= \sum_{i=1}^{n} \left[y_i \cdot x_{ij} - \frac{1}{1+e^{\mathbf{x}_i^T \beta}} \cdot e^{\mathbf{x}_i^T \beta} \cdot x_{ij} \right] \\
&= \sum_{i=1}^{n} x_{ij} \left[y_i - \frac{e^{\mathbf{x}_i^T \beta}}{1+e^{\mathbf{x}_i^T \beta}} \right] \\
&= \sum_{i=1}^{n} x_{ij} [y_i - \pi(\mathbf{x}_i)]
\end{aligned}
$$

These derivatives are set equal to zero to obtain the maximum. For this reason, the following equations need to be solved to obtain the maximum likelihood estimators.

$$\sum_{i=1}^{n} [y_i - \pi(\mathbf{x}_i)] = 0 \tag{5}$$

and

$$\sum_{i=1}^{n} x_{ij} [y_i - \pi(\mathbf{x}_i)] = 0 \tag{6}$$

for $j = 1, 2, \ldots, p$.

3.2.1 Iterative Weighted Least Squares estimates

According to Dobson and Barnett [2018], most statistical packages use an Iterative Weighted Least Squares (IWLS) procedure for estimating the maximum likelihood estimators. This also holds for the statistical software used for the frequentist logistic regression in this research, which is the glm function in the programming language R. According to Charnes et al. [1976], if the distribution functions of the outcome variables are part of the regular exponential family, the maximum likelihood estimation and IWLS estimates lead to the same result. The authors describe that a distribution of a random variable Y belongs to the regular exponential family, if it can be written in the form

$$f(y; \pi) = exp[y \cdot b(\pi) + c(\pi) + d(y)],$$

for known functions b, c and d where the equality $E(Y) = \pi$ holds.

For individual outcome variables that have the Bernoulli distribution, as written in Equation 2, this likelihood can be written as

$$\begin{aligned} f(y_i; \pi(\mathbf{x}_i)) &= exp\left[y_i log(\pi(\mathbf{x}_i)) + (1 - y_i) log(1 - \pi(\mathbf{x}_i))\right] \\ &= exp\left[y_i log\left(\frac{\pi(\mathbf{x}_i)}{1 - \pi(\mathbf{x}_i)}\right) + log(1 - \pi(\mathbf{x}_i))\right] \\ &= exp(y_i \cdot b(\pi(\mathbf{x}_i)) + c(\pi(\mathbf{x}_i))), \end{aligned}$$

where $b(\pi(\mathbf{x}_i)) = log\left(\frac{\pi(\mathbf{x}_i)}{1-\pi(\mathbf{x}_i)}\right)$ and $c(\pi(\mathbf{x}_i)) = log(1 - \pi(\mathbf{x}_i))$. For this reason, it can be concluded that the likelihood distribution used in this logistic regression is indeed part of the regular exponential family. For this reason, the maximum likelihood estimation and the IWLS estimation lead to the same results, meaning that the IWLS algorithm can be used as well.

The following description of the IWLS algorithm is based on the explanation by Dobson and Barnett [2018]. Before stating the algorithm, some notation needs to be explained. Define $g(\pi(\mathbf{x}_i)) = log\left(\frac{\pi(\mathbf{x}_i)}{1-\pi(\mathbf{x}_i)}\right) = \mathbf{x}_i^T \beta = \eta_i$. Note that from the properties of the Bernoulli distribution, it can be derived that for an outcome variable Y_i, it follows that

$$Var(Y_i) = \pi(\mathbf{x}_i)(1 - \pi(\mathbf{x}_i))$$

(while the same result could be derived from the general derivation by Dobson and Barnett [2018]). Using this, define $\mathbf{z}^{(m-1)}$ to be a vector of length n with elements

$$z_i^{(m-1)} = \mathbf{x}_i^T \mathbf{b}^{(m-1)} + (y_i - \pi(\mathbf{x}_i)) \left(\frac{\partial \eta_i}{\partial \pi(\mathbf{x}_i)} \right) = \mathbf{x}_i^T \mathbf{b}^{(m-1)} + \frac{(y_i - \pi(\mathbf{x}_i))}{\pi(\mathbf{x}_i)(1 - \pi(\mathbf{x}_i))},$$

with $\pi(\mathbf{x}_i)$ and η_i evaluated using $\mathbf{b}^{(m-1)}$. Also define $\mathbf{W}^{(m-1)}$ to be an $n \times n$ diagonal matrix with elements

$$w_{ii}^{(m-1)} = \frac{1}{Var(Y_i)} \left(\frac{\partial \pi(\mathbf{x}_i)}{\partial \eta_i} \right)^2 = \frac{1}{\pi(\mathbf{x}_i)(1 - \pi(\mathbf{x}_i))} \left(\pi(\mathbf{x}_i)(1 - \pi(\mathbf{x}_i)) \right)^2 = \pi(\mathbf{x}_i)(1 - \pi(\mathbf{x}_i)).$$

Define $\mathbf{X}$ to be the $n \times (p + 1)$ matrix that has $\mathbf{x}_i^T$ as rows. Using these definitions, the algorithm is as follows [Dobson and Barnett, 2018].

1. Set $m = 0$. Start with some initial approximation of β denoted by $\mathbf{b}^{(m)} = \mathbf{b}^{(0)}$.
2. Increase m by 1.
3. Calculate $\mathbf{z}^{(m-1)}$ and $\mathbf{W}^{(m-1)}$ using $\mathbf{b}^{(m-1)}$.
4. Calculate $\mathbf{b}^{(m)} = (\mathbf{X}^T \mathbf{W}^{(m-1)} \mathbf{X})^{-1} \mathbf{X}^T \mathbf{W}^{(m-1)} \mathbf{z}^{(m-1)}$.
5. Repeat steps 2-5.

Once the difference between two successive approximations of $\mathbf{b}$ becomes small enough, the iterations are stopped. A more thorough explanation and the derivation of the IWLS algorithm can be found in literature, for example by Dobson and Barnett [2018].

3.2.2 Model selection methods

When applying the logistic regression, usually not all covariates contribute to the ability to predict the outcome variable. There exist multiple approaches that determine which coefficients are relevant for the prediction in the model. Below, three of these methods are discussed.

The first method, is the Akaike Information Criterion (AIC) [Akaike, 1974]. The AIC is a measure of goodness of fit. It is defined as follows

$$AIC = -2l + 2(p + 1).$$

In this formula, l is the maximum log-likelihood and p is the number of regression coefficients, meaning that the 1 is written to take the constant β_0 into account. The AIC tries to find the best trade-off between having a sparse model while still having a good fit. Models with a lower AIC are for this reason preferred over models with a higher AIC when using this method. To ensure getting a sparse model with a good fit, a higher likelihood means a lower AIC and the the inclusion of unnecessary covariates is 'punished'. This is one of the methods that is be considered to determine which covariates have to be included in the model.

A second way to determine the covariates that have to be included, is using the Bayesian Information Criterion (BIC) [Dobson and Barnett, 2018]. The reasoning behind this method

is similar to that of the AIC, as the formula is only slightly different. There are differences in the exact definition of the formula of the BIC between different literature. However, the following definition, which is also used in the BIC function in the programming language R, is stated by Gelman et al. [2013]

$$BIC = -2l + (p+1) \cdot log(n),$$

where n is the number of observations. From this extra term, it can be seen that the BIC uses a higher 'punishment' when determining the model with the best fit.

The last method of model selection discussed in this section, is the method of p-values. This method is described by Dobson and Barnett [2018], among others. For every variable in the full model, the significance of each variable can be indicated using a p-value. The lower the p-value, the more certainly it can be said that the value is not equal to zero, meaning that the covariate is relevant to be included. A common significance level is 0.05, which will also be taken in this analysis. This approach is more 'obsolete' than the approaches of the information criterions, as it does not take into account whether information is lost in the process. However, it can be interesting to see which method performs better.

3.2.3 Algorithm for model selection

For the Framingham data set, the three methods described above are used to determine the covariates that should be included. The approach for this differs slightly for the different methods.

AIC and BIC
For the AIC and BIC methods, the glm function is first applied to the full model, meaning that all covariates are included at first. Then, the algorithm iterates over all covariates. In every of these steps, the covariate is excluded from the model and a new model is built. If this covariate is the design variable of a categorical variable, the algorithm exludes all design variables related to the categorical variable [Hosmer Jr et al., 2013]. After building this model, using the respective method, AIC or BIC, an estimate of the goodness of fit is made. Whenever the AIC or BIC is lower for the reduced model, meaning that this model is sparser than the previous model while maintaining a good fit, this covariate is excluded from the model. This step is repeated until there are no covariates left that give a better model. This process is called the backward elimination algorithm. Lastly, to be sure that no covariates were excluded that were important to the model, a final check is made whether the inclusion of any removed covariate or set of design variables gives a better AIC or BIC, which is a test for forward selection. This process is described by Hosmer Jr et al. [2013]. When this is done, the remaining covariates form the best model according to the chosen method.

According to Gelman et al. [2013], the BIC is not intended to predict performance of the model on data that is not in the sample. Therefore, a lower performance is expected of this information criterion and thus it will most likely not be the method performing best. This is to be expected, as it has a higher penalty. Therefore, it is more likely to delete important covariates.

p-values
For the p-values, a similar method is applied, as a backward elimination algorithm is used as well. An iteration is done over all covariates again. In this case, if the p-value of a covariate that is selected exceeds 0.05, this covariate is excluded. If this covariate with a p-value exceeding 0.05 is found and excluded, the iteration is stopped and the process is repeated, starting from the beginning with a new iteration. Similar to the method for the AIC and the BIC, if this covariate is a design variable of a categorical variable, all design variables are included or excluded simultaniously and they are excluded only when all three p-values exceed 0.05. When no more covariates have an insignificant p-value, a short forward selection algorithm is performed to check whether no significant covariates have been excluded, as this is a risk when removing covariates with a p-value slightly above 0.05.

The code that was used to do model selection and generate fitted values on the whole data using the AIC can be found in Appendix D. The procedures for the BIC and the p-values follow in a similar manner.

3.2.4 Fitted values of the model

When a model selection method has been performed, the result is a list of the significant covariates, all with a coefficient β_j. This result can then be used to compute a fitted value for every instance in the data. Recall that the predicted probability of 'success' for an instance in the data is denoted by

$$\pi(\mathbf{x}_i) = \frac{exp(\mathbf{x}_i^T \beta)}{1 + exp(\mathbf{x}_i^T \beta)}.$$

Because an estimate of β is known, for every instance in the data set, it is possible to predict the probability of success using this equation. This probability can be used to compare the performance of different methods. In Section 3.5, a measure of performance is discussed that uses this predicted probability.

3.3 Bayesian Logistic Regression

The Bayesian Logistic Regression is part of Bayesian statistics. This is a completely different paradigm of statistics, which comes with multiple advantages, like discussed in Hosmer Jr et al. [2013]. One of those advantages is that it is possible to incorporate prior knowledge into the analysis. If previous studies have shown a certain distribution of the parameters, it is possible to use this information while updating the belief through the Bayesian model. A second advantage is that there is no need to assume an initial distribution, where this is necessary in frequentist approaches when for example calculating a confidence interval. The last advantage is that the interpretation of Bayesian models comes is often described as more natural. For example, as described by Dobson and Barnett [2018], p-values in Bayesian models are the probability that the null hypothesis is true given the observed data, where in frequentist models the p-value is the probability that more extreme data is observed. For confidence intervals, similar differences hold and the Bayesian approach can be considered to be intuitive.

First of all, because this is a logistic regression again, this means that the same formulae from Section 3.1 are used again. Recall that the likelihood function is described by

$$L(\beta) = \prod_{i=1}^{n} \pi(\mathbf{x}_i)^{y_i} \cdot (1 - \pi(\mathbf{x}_i))^{1-y_i}.$$

There is an essential difference between the frequentist and Bayesian approach, as described by Dobson and Barnett [2018]. The frequentist approach assume that there is a fixed set of parameters β and that the data is dependent on this parameter. The Bayesian approach reverses this, which is an arguably more intuitive description. In a Bayesian model, given the data, the parameter β has a distribution and is a random variable, instead of one fixed variable, dependent on the data. In order to do this, Bayesian statistics strives to estimate $P(\beta|\mathbf{y})$. This estimation is done using Bayes' equation, which is as follows [Dobson and Barnett, 2018]

$$P(\beta|\mathbf{y}) = \frac{P(\mathbf{y}|\beta)P(\beta)}{P(\mathbf{y})}. \tag{7}$$

In this formula, β is the set of parameters and $\mathbf{y}$ is the observed data. Because $P(\mathbf{y})$ does not have a closed form solution, it cannot be calculated. However, for a set of data, this probability does not change. Therefore, it can be said that

$$P(\beta|\mathbf{y}) \propto P(\mathbf{y}|\beta)P(\beta), \tag{8}$$

meaning that the two sides are proportional up to a scalar multiple. The three distributions in this equation are described in Section 3.3.1, which is where the logistic regression formulae are used.

3.3.1 Prior, likelihood and posterior

The three distributions in Equation 8 are the prior, likelihood and posterior. These are discussed separately.

Prior
The prior distribution is denoted by $P(\beta)$. This distribution indicates the prior belief about the parameters, which is determined before using the data. There are many different possible choices of priors. As nothing is known about the parameters before doing the analysis of the Framingham data set, the choice is made to make this a normal distribution with zero mean and a variance of 1. Due to the fact that the categorical variables were standardised, this also seems to be a reasonable choice.

Likelihood
The likelihood is denoted by $P(\mathbf{y}|\beta)$. This is the probability of the data to appear given the set of parameters β. This is equal to the likelihood distribution. As mentioned before, because the log-likelihood function is easier to do computations with, the log-likelihood is used here as well. Therefore, the relation with $P(\mathbf{y}|\beta)$ is as follows

$$log(P(\mathbf{y}|\beta)) = l(\beta) = \sum_{i=1}^{n} [y_i log(\pi(\mathbf{x}_i)) + (1 - y_i) log(1 - \pi(\mathbf{x}_i))]. \tag{9}$$

When filling in Equation 1, this gives the following derivation. This derivation is based on the lecture notes of the course Statistical Reasoning at the University of Groningen [Grzegorczyk, b].

$$\pi(\mathbf{x}_i) = \frac{exp(\mathbf{x}_i^T\beta)}{1 + exp(\mathbf{x}_i^T\beta)} = \frac{1}{1 + exp(-\mathbf{x}_i^T\beta)}.$$

This gives

$$1 - \pi(\mathbf{x}_i) = \frac{exp(-\mathbf{x}_i^T\beta)}{1 + exp(-\mathbf{x}_i^T\beta)}.$$

When filling in these two equations in Equation 9, this gives the following.

$$\begin{aligned}
l(\beta) &= \sum_{i=1}^{n} [y_i log(\pi(\mathbf{x}_i)) + (1 - y_i) log(1 - \pi(\mathbf{x}_i))] \\
&= \sum_{i=1}^{n} \left[y_i log\left(\frac{1}{1 + exp(-\mathbf{x}_i^T\beta)}\right) + (1 - y_i) log\left(\frac{exp(-\mathbf{x}_i^T\beta)}{1 + exp(-\mathbf{x}_i^T\beta)}\right)\right] \\
&= \sum_{i=1}^{n} \left[-y_i log\left(1 + exp(-\mathbf{x}_i^T\beta)\right) + (1 - y_i) log\left(exp(-\mathbf{x}_i^T\beta)\right) - (1 - y_i) log\left(1 + exp(-\mathbf{x}_i^T\beta)\right)\right] \\
&= \sum_{i=1}^{n} \left[(1 - y_i) log\left(exp(-\mathbf{x}_i^T\beta)\right) - log\left(1 + exp(-\mathbf{x}_i^T\beta)\right)\right] \\
&= \sum_{i=1}^{n} \left[(1 - y_i)(-\mathbf{x}_i^T\beta) - log\left(1 + exp(-\mathbf{x}_i^T\beta)\right)\right] \\
&= \sum_{i=1}^{n} \left[(y_i - 1) \cdot \mathbf{x}_i^T\beta - log\left(1 + exp(-\mathbf{x}_i^T\beta)\right)\right]
\end{aligned}$$

Posterior
The posterior is denoted by $P(\beta|\mathbf{y})$. The prior and likelihood are used to calculate what the posterior is proportional to, using Equation 8. This is where Markov Chain Monte Carlo simulations are used to generate a sample from the posterior distribution, as described in Section 3.4.

3.4 Markov Chain Monte Carlo Simulations

Doing a Markov Chain Monte Carlo simulation is a method to sample from the posterior distribution, without knowing $P(\mathbf{y})$. The strategy is to take a sample from the distribution. Hosmer Jr et al. [2013] describe this as a "random walk" around the posterior distribution of the parameters. The authors describe the following eight steps.

1. Pick starting values for the parameters, which are taken as the current values, $\beta_{current}$.
2. Use this set of parameters to compute the unstandardised posterior density $P(\beta_{current}|\mathbf{y})$, as in Equation 8. This can be directly calculated, as the formula for the prior and likelihood are known.
3. Generate proposed parameter values, $\beta_{proposed}$. This can be taken from for example a normal distribution centered around the current parameter values. In this research, a uniform distribution centered around the current parameter values is used.

4. Use $\beta_{proposed}$ to compute the unstandardised posterior density $P(\beta_{proposed}|\mathbf{y})$.

5. Compute the probability of moving to the proposed parameter values as $p_{move} = min\left[\frac{P(\beta_{proposed}|\mathbf{y})}{P(\beta_{current}|\mathbf{y})}, 1\right]$. If the posterior likelihood is higher for the proposed set of parameters, the probability of moving is equal to 1. If it is smaller, then it can still move to that set of parameters, but with a lower probability. Note here that this ratio is the same for the unstandardised posterior and the standardised posterior because

$$\frac{P(\beta_{proposed}|\mathbf{y})}{P(\beta_{current}|\mathbf{y})} = \frac{\frac{P(\mathbf{y}|\beta_{proposed})P(\beta_{proposed})}{P(\mathbf{y})}}{\frac{P(\mathbf{y}|\beta_{current})P(\beta_{current})}{P(\mathbf{y})}} = \frac{P(\mathbf{y}|\beta_{proposed})P(\beta_{proposed})}{P(\mathbf{y}|\beta_{current})P(\beta_{current})}.$$

6. Generate a random value u between 0 and 1 using the uniform distribution.

7. Accept the proposed set of parameters $\beta_{proposed}$ if $u < p_{move}$, otherwise keep the current set of values $\beta_{current}$.

8. Repeat step 2-7 with the set of parameters β that was chosen in step 7.

When doing many iterations, this walk around the posterior distribution gives a sample for the posterior distribution. In Section 3.4.5, it is explained how this sample can be used to do inference.

The code that was used to do the Bayesian logistic regression, using the MCMC algorithm, can be found in Appendix E. This code has been written by prof. dr. Grzegorczyk, but some changes have been made to ensure it suits the Framingham data set and to increase the speed of the algorithm.

3.4.1 Convergence MCMC

There are multiple methods to determine whether this algorithm has converged. These methods require doing multiple MCMC simulations. In this way, it can be checked whether there are differences between the simulations, which is an indication that more iterations are necessary to achieve convergence. The methods described below, are based on Hosmer Jr et al. [2013].

The first method is a visual inspection of the trace plot. In an MCMC simulation, every variable continues to update, in order to get a sampling distribution. A trace plot shows for every iteration in the MCMC simulation what the value of the parameter is at this specific iteration. For example, in Figure 1, the trace plot for the parameter 'male' for the Framingham data set is shown. This plot shows that the method quickly converges to a mean that seems to be around 0.5, after which is randomly fluctuates.

In order to assess convergence by using trace plots, two of these plots need to be compared. If both plots seem to randomly fluctuate around the same values, then this is an indication that the MCMC simulation has converged.

According to Hosmer Jr et al. [2013], there exists a second method of assessing convergence as well. This is called the Brooks-Gelman-Rubin (BGR) statistic. This statistic checks whether the variation between different chains is not much bigger than the variation

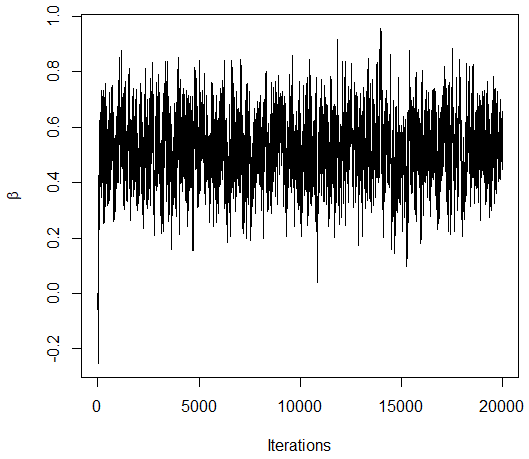


Figure 1: Trace plot for 20000 MCMC iterations for slope parameter male.

within one chain. This would be an indication that the chains are not fluctuating around the same value, meaning that they have not yet converged. This value is described as follows.

Let there be $j = 1, \ldots, m$ chains for one specific parameter, all consisting of $i = 1, \ldots, q$ sampled values denoted by β_{ij}. Then the variance of the parameter values in chain j is given by

$$s_j^2 = \frac{1}{q-1} \sum_{i=1}^{q} (\beta_{ij} - \bar{\beta}_{.j})^2$$

where $\bar{\beta}_{.j}$ is the average of the parameter values in chain j. Then W, which is called the within chain variability, is the average of the variances for all chain runs. This is denoted by

$$W = \frac{1}{m} \sum_{j=1}^{m} s_j^2.$$

The variability between chains, B, is defined by

$$B = \frac{1}{m-1} \sum_{j=1}^{m} q(\bar{\beta}_{.j} - \bar{\beta}_{..})^2,$$

where $\bar{\beta}_{..}$ is the average of all sampled values from all chains. Then, the marginal posterior variance for this parameter is estimated with a weighted average of these two variances. The formula for this is given as

$$\hat{V}_\beta = \frac{q-1}{q} W + \frac{1}{q} B.$$

Using this, the BGR statistic is given by

$$\hat{R} = \sqrt{\frac{\hat{V}_\beta}{W}}.$$

According to Gelman et al. [2013], the value of $\hat{R}$ tends to 1 as the amount of iterations goes to infinity. The authors say that if the value of $\hat{R}$ is below 1.1, the simulation can be considered to have converged for this specific parameter. One small remark to make here is that they do mention that this is dependent on the situation.

3.4.2 Alternative: Reversible Jump Markov Chain Monte Carlo Simulations

MCMC simulations are a method to sample from the distribution of the parameters. To determine which parameters are relevant, one can do a backwards elimination algorithm, removing one covariate at a time, until all parameters that do not contribute to the prediction of the outcome variable have been removed. However, this requires to run the MCMC algorithm many times, while one run is already computationally expensive. An alternative procedure that solves this issue is described by Green [1995], this is called a Reversible Jump Markov Chain Monte Carlo (RJMCMC) simulation.

While the RJMCMC algorithm does require many iterations, it is not necessary to do multiple runs. The main idea is that the algorithm performs normal MCMC simulations, but that with some probability, it jumps to another set of covariates. This means, for every covariate, it determines whether it wants to update the value or whether it wants to remove that value from the set of covariates (or add it if it was not in the set of covariates). The algorithm can be described as follows, based on the description in the lecture notes of the course Statistical Reasoning at the University of Groningen [Grzegorczyk, a].

Before defining the algorithm, the indicator vector $\mathbf{v}$ needs to be defined.

$$\mathbf{v}^{(t)} = (v_1^{(t)}, \ldots, v_k^{(t)})$$

$v_j^{(t)}$ equals 1 if covariate j is in the set of covariates at iteration t, otherwise it equals 0.

Then the RJMCMC algorithm is defined as follows. First, select a covariate $j \in \{1, \ldots, k\}$. With probability p, for $0 < p < 1$, perform an MCMC step, as described before. If this does not happen, with probability $1 - p$, the algorithm perform the following reversible jump steps.

1.
   - If $v_j^{(t)} = 1$, propose to delete the j-th covariate, so remove this entry from the parameter vector and from the indicator vector.

     $$\mathbf{v}^{(*)} = (v_1^{(t)}, \ldots, v_{j-1}^{(t)}, 0, v_{j+1}^{(t)}, \ldots, v_k^{(t)})$$

     $$\beta^{(*)} = (\beta_1^{(t)}, \ldots, \beta_{j-1}^{(t)}, 0, \beta_{j+1}^{(t)}, \ldots, \beta_k^{(t)})$$

   - If $v_j^{(t)} = 0$, propose to add the j-th covariate, so add this entry to the parameter vector and to the indicator vector. The parameter added, $\beta_j^{(*)}$, is sampled from $\mathcal{N}(0, \sigma^2)$ for some value σ.

     $$\mathbf{v}^{(*)} = (v_1^{(t)}, \ldots, v_{j-1}^{(t)}, 1, v_{j+1}^{(t)}, \ldots, v_k^{(t)})$$

     $$\beta^{(*)} = (\beta_1^{(t)}, \ldots, \beta_{j-1}^{(t)}, \beta_j^{(*)}, \beta_{j+1}^{(t)}, \ldots, \beta_k^{(t)})$$

2. Compute the acceptance probability

$$A = min\left[1, \frac{P(\mathbf{y}|\beta^{(*)})}{P(\mathbf{y}|\beta^{(t)})}\right]$$

3. Draw a random number between 0 and 1.

   - If $u \leq A$ set $\mathbf{v}^{(t+1)} = \mathbf{v}^{(*)}$ and $\beta^{(t+1)} = \beta^{(*)}$
   - If $u > A$ set $\mathbf{v}^{(t+1)} = \mathbf{v}^{(t)}$ and $\beta^{(t+1)} = \beta^{(t)}$

Lastly, repeat this process of determining whether to perform an MCMC step or a reversible jump step for the amount of iterations done.

3.4.3 Convergence RJMCMC

For the RJMCMC algorithm, a different convergence check must be applied. Because the algorithm switches between sets of covariates, the variances will become bigger, because different chains can be in different sets of covariates at the same iteration. Therefore, the convergence check that is used for the MCMC algorithm cannot be used.

An approach that can be taken, is to check whether two different chains spend the same amount iterations in a certain set of covariates. The way to do this, is as follows. For every covariate, check the percentage that this covariate is in the model. This is done for at least two chains. Then, this is compared for all covariates. If these percentages are close, then it can be concluded that the RJMCMC chain has converged.

3.4.4 Burn-in and thinning

When sampling the posterior distribution, there are a few issues that can appear. First of all, the sample has usually not converged for the first iterations of the MCMC or RJMCMC simulation. If part of the sample has not been converged, then this part should not be used, as this does not provide a good random sample from the posterior distribution. Therefore, it is common to discard the first iterations. The iterations that are discarded, are named the *burn-in*. Gelman et al. [2013] mention that it is good practice to discard the first half of the iterations. Despite this being a conservative practice, in this way there is, in combination with a convergence check, less risk of using data that has not converged.

Another issue with the sample is that its values depend on previous values. This is the case, because new values are always a small change from the previous values. Therefore, it is possible that the values from the sample are correlated. This makes the sample 'less random'. To avoid this, *thinning* is used. Dobson and Barnett [2018] describe that thinning is the process of keeping every j-th sample and discarding all other samples. For example, in case of thinning with j=2, this would mean that every odd sample is discarded, while all the even samples are kept. In this research, j=10 is used.

3.4.5 Use the (RJ)MCMC sample

Independent of which sampling method was used, the sample is used in the same way. The sample has multiple uses, like discussed in Hosmer Jr et al. [2013]. First of all, it is possible

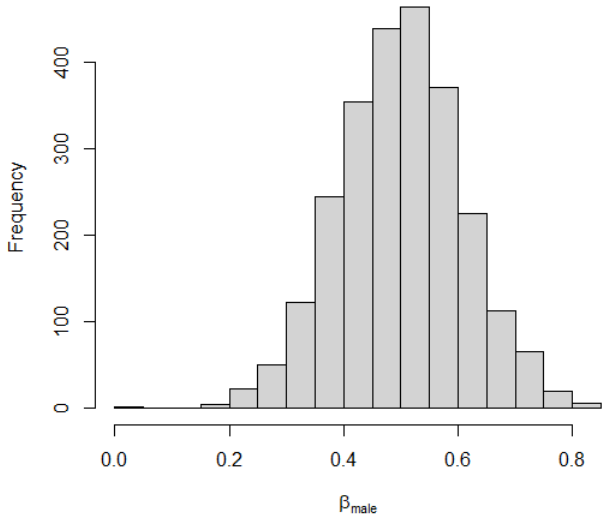


Figure 2: Histogram for the β_{male} parameter from a sample of 50000 RJMCMC iterations after burn-in and thinning.

to do inferences about the individual covariates. For example, the sample, after burn-in and thinning, produces a distribution for each parameter. This comes with a mean value and a 95% confidence interval can be constructed for further inferences. A histogram can be constructed for this as well, like for example Figure 2.

However, more is possible than only doing inferences on the individual parameters. It is also possible to use the whole sample to compute fitted values. In this case, every iteration from the sample, after burn-in and fitting, is used to compute its own fitted value, which is the probability of 'success'. For each instance in the test data, this gives a distribution on the fitted values as well. In this way, it is also possible to determine a confidence interval on the probability that a patience will get the coronary heart disease. An example of a sample of such a distribution is shown in Figure 3. When taking either the mean or the median of this sample, this can be regarded as the estimated 'true value'. This value can be used to do predictions and this prediction can be compared with the predictions of other methods.

3.5 Area Under the Receiver Operating Characteristic Curve

After applying the methods described above, the performance of different methods needs to be assessed. The method used for this is the Area Under the Receiver Operating Characteristic Curve (AUC) [Hosmer Jr et al., 2013]. This method is used for the comparison of the frequentist and Bayesian methods, but also for the comparisons within each method. This method is a test of performance when the goal is to classify instances of a data set. In the case of the Framingham data set, and in data sets with a binary outcome variable, this

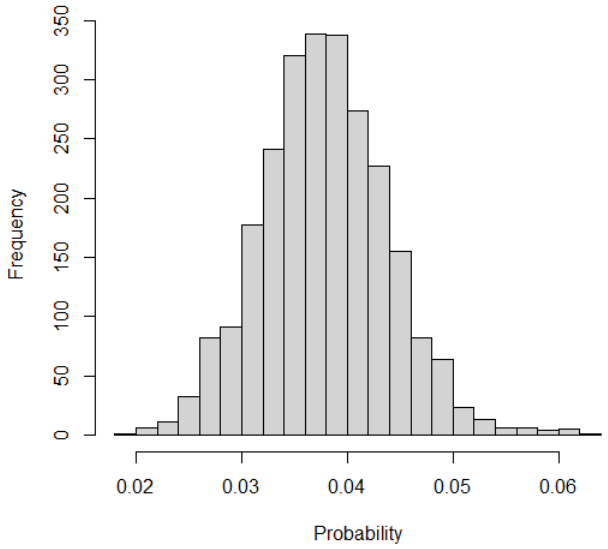


Figure 3: Probability distribution for one patient from a sample of 50000 RJMCMC iterations after burn-in and thinning.

is ideal, as the goal is to determine whether a success is predicted, meaning that the goal is to predict which patients get the coronary heart disease.

In order to explain the AUC, two other concepts need to be explained, which are *sensitivity* and *specificity*.

Sensitivity
The sensitivity is the probability of correctly detecting a true signal given a certain cutpoint. This is determined in the following way. A model that was chosen to do predictions gives fitted values. These are predicted probabilities of getting a success given an instance in the data set. In the Framingham data set, this is the predicted probability of getting the heart disease given someone's medical data. Then, to classify an instance as a success or not, a cutpoint is taken. For example, for a cutpoint of 0.5, all instances with a predicted probability smaller than 0.5 are classified as a person who will not get the coronary heart disease. All other instances, of people with a predicted probability bigger than or equal to 0.5, are classified as having the heart disease. Then, the sensitivity, given a certain cutpoint, is the probability of correctly detecting a true signal. For the Framingham data set, this can be described using the definitions in Table 4.
Using the definitions in this table, the sensitivity is defined to be the following

$$Sensitivity = \frac{TP}{TP + FN},$$

which is the amount of people who got the heart disease who were predicted to get the heart

Notation	Name	Description
TP	True Positives	Number of patients correctly predicted to get CHD
FP	False Positives	Number of patients incorrectly predicted to get CHD
TN	True Negatives	Number of patients correctly predicted to not get CHD
FN	False Negatives	Number of patients incorrectly predicted to not get CHD

Table 4: Classification of predictions in categories True Positives, False Positives, True Negatives and False Negatives, given a certain cutpoint.

disease divided by the total amount of people who got the heart disease.

Specificity
The specificity is the probability of correctly detecting a false signal given a certain cutpoint. The intuition behind this term is analogous to the intuition of the sensitivity. The equation for this is as follows

$$Specificity = \frac{TN}{TN + FP}.$$

Before determining the AUC, the Receiver Operating Characteristic Curve (ROC curve) needs to be explained. The ROC curve is a plot of one minus the specificity against the sensitivity. This is plotted for all specific cutpoints. So for every specific cutpoint, it demonstrates how many of the true positives and true negatives a method is able to find proportionally. An example of an ROC curve can be found in Figure 4.

In a ROC curve, a higher line means that the model performs better at classification. If it is along the diagonal line, then it is not able to classify at all. For example, take a cutpoint of 0.5. If random guesses are chosen as a classification method, then fifty percent of the times a success is predicted and fifty percent of the times a failure is predicted. However, because not any information is used, in around half of the cases a success is predicted correctly and in around half of the cases a failure is predicted correctly. So in the worst case, for all possible cutpoints, the specificity and sensitivity will be on the diagonal line. The further the line is from the diagonal, the better the model performs. A good way of measuring how far the line is from the diagonal, is by taking the area under this curve, which is the AUC. This is a way to determine how well the model is at classification without specifying a single cutpoint, as it takes all cutpoints into account. When the line is along the diagonal, the AUC is 0.5, which means no discrimination of results. The closer this value gets to 1, the better the discrimination. Although there is no perfect way to determine which values are good and which are not, the following can be taken as a rule of thumb, according to Hosmer Jr et al. [2013].

$$\text{If} \begin{cases} AUC = 0.5 & \text{No discrimination} \\ 0.5 < AUC < 0.7 & \text{Poor discrimination} \\ 0.7 \leq AUC < 0.8 & \text{Acceptable discrimination} \\ 0.8 \leq AUC < 0.9 & \text{Excellent discrimination} \\ AUC \geq 0.9 & \text{Outstanding discrimination} \end{cases} \tag{10}$$

The AUC is used to determine which model selection method works better. When

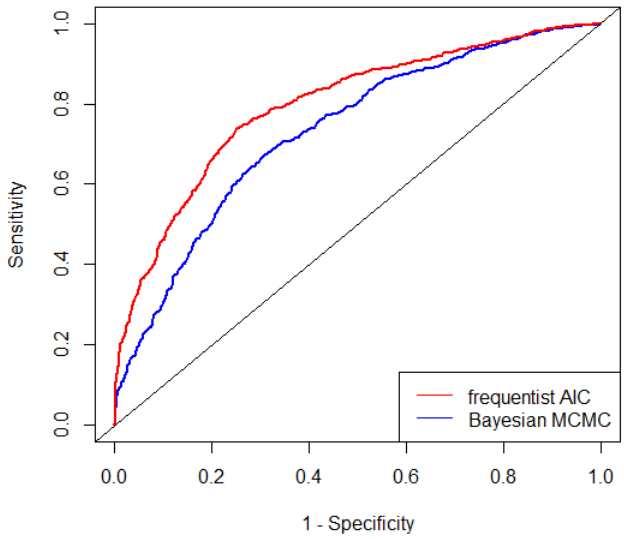


Figure 4: An example of two ROC curves, from the fitted values of the frequentist approach with the AIC and the fitted values of the Bayesian approach with the MCMC algorithm.

comparing the frequentist and the Bayesian approaches, it is also be used, as it is a good way to determine performance in terms of classification of totally different methods as well.

3.6 Cross-validation

When building a model using a regression, this gives a set of parameters is used to predict probabilities of success, in the case of the Framingham data set the probability that a person gets the coronary heart disease. When the model is 'trained', the data set would be used to generate the set of parameters. Then, the AUC is used to see how good the model performs at classification. Here, one issue arises. If the model is trained on the same data set that it is tested on, this is not a fair comparison, because the model has already 'seen' this data before. One problem that can arise is overfitting, a model with many parameters can seem to perform well on the data set, but it might just be adjusted to the data it has already seen.

This is where *cross-validation* is used, as for example described by Rodriguez et al. [2009]. In cross-validation, the model is trained on a part of the data, but the testing of performance happens on a different data set. This gives a comparison that is more fair, as it prevents overfitting and it gives a good indication of whether the model will generalise to other data that can be found. To summarise, it answers the question whether the model is actually good at predicting heart diseases for new subjects.

One specific way of doing cross-validation, is *K-fold cross-validation*. $\frac{K-1}{K}$ part of the data is used to train the model and get specific parameters, for K=5 this would be 80%.

Then, $\frac{1}{K}$ part is used to see how well the model performs, in the case of K=5 this is 20%. This generates a series of fitted values of, in this case, 20% of the data set. This process is repeated K times, so that all data is used as test data once. Then, these fitted values can be combined again to get fitted values for the whole data set. The advantage of this is that fitted values for the whole data set can be generated, while all data was trained with 'unseen' data. This set of fitted values can then be compared using the AUC or any other test of performance.

K-fold cross-validation is not the only way of doing cross-validation. There are many other methods, for example where only one data point is left out and this is repeated for all entries of the data set. However, due to the fact that mostly the Bayesian logistic regression is computationally very expensive, this turns out to be infeasible to do in this research, with limited computational power. For that reason, a 5-fold cross-validation was chosen. A higher number for K usually gives a better representation of the fit of the model, but this also comes with a vast increase in computational expenses. According to Rodriguez et al. [2009] and Arlot and Lerasle [2012], a 5-fold cross-validation does not perform significantly worse compared to using higher numbers for K. Contrary to that, 2-fold cross-validation does perform significantly worse according to those authors. This means that the 5-fold cross-validation chosen in this research seems to be a good choice that balances computational cost with good performance.

3.7 Sample Size Reduction

For both the frequentist and Bayesian approach, different methods have been discussed to build models and to compare the results. These methods can be used on the full data set, to see how well all the methods perform on the full data set using the AUC. After doing this comparison, the next step is to do the sample size reduction. When the sample size decreases, the performance of the frequentist and Bayesian approach are compared. For the Bayesian approach, either the MCMC or RJMCMC simulation is chosen, due to its computational expense. For the frequentist approach, all three methods are compared to see how well they maintain performance when decreasing the sample size.

As described before, the sample size is reduced in the following way. The order of the data set is randomised and is split in half. Then, all the procedures take place in the same way, by doing the same analysis as on the full data set. This is done a few times, until no relevant conclusions can be drawn from the data set. The goal of this repeated analysis on different sample sizes, is to establish a relationship in the performance of the methods based on sample size.

When reducing the sample size, the amount of data in the new data set becomes smaller. This gives a bigger risk of measuring a performance that is either too high or too low, as the testing and the training data might not be a balanced representation of the full data set. To prevent that the 'coincidence' of choosing a bad part of the data has an influence on the measured performance, the analysis is done on all parts of the data set after halving. For example, let's take take the data set after halving once. Then the analysis is performed on both halves using the AUC and cross-validation. This gives two arrays of fitted values, of which the performance can be assessed. But before assessing the performance on the individual halves, these two arrays of fitted values are concatenated, generating an array

of fitted values for the whole initial data set. When assessing the performance on this full array of fitted values using the AUC, this gives a balanced view of how well the method performs, contrary to when only one part is used. The same procedure is taken when the data set is halved more times. For example, in the case of halving 5 times, all 32 parts are used to generate separate fitted values.

To understand what halving a certain number of times means, this relation is shown in Table 5.

Halving number	Percentage of data	Number of instances
0	100%	3658
1	50%	1829
2	25%	914
3	12.5%	457
4	6.25%	228
5	3.125%	114

Table 5: Relation between halving number, percentage of data and number of instances in the reduced data set.

4 Results

Before the comparison of the frequentist and Bayesian logistic regression can be done, the performance of all methods within both paradigms is compared. As mentioned before, the performance when decreasing the sample size is discussed for the three frequentist methods. For the Bayesian approach, one method is chosen that of which the performance will be discussed. Therefore, the results in both approaches are examined before comparing the two approaches.

4.1 Frequentist

For the frequentist approach, like discussed in Section 3, the three methods that are considered are the method of the AIC, BIC and the method of p-values. After applying the algorithms for these methods, an output is produced for each method. This output can be found in Appendix F.

In this output, it is interesting to note that the BIC and p-values methods lead to the same outcome in this particular case. These methods selected the same set of variables. Due to the fact that there is one best outcome for each fixed set of covariates according to the frequentist approach, this automatically also leads to the same output of the glm function. It can also be seen that the BIC and p-values give a sparser model. It is to be expected that the AIC method gives a model with more parameters than the BIC method, as the BIC has a larger penalty for additional covariates.

The BIC and the p-values methods give the exact same result. For this reason, these methods are considered together from this point on. The two different outcomes are compared to determine which method is best at classifying outcomes in the Framingham data set. As discussed before, this is done using the Area Under the Receiver Operating Characteristic Curve, the AUC.

After applying these methods to the separated test data, as discussed in Section 3.6, the ROC curve can be plotted as seen in Figure 5, and the AUC can be calculated. The AUC's of both methods are given in Table 6. Recall that a higher ROC curve means that the model is better at classifying the data. Also, recall that the AUC is on a scale from 0.5 to 1, where values closer to 1 are considered better. Therefore, based on visual inspection of the graph and based on the AUC values, it can be concluded that the AIC performs best on this data. As mentioned before, this was to be expected for according to Gelman et al. [2013] and Gelman et al. [2014], as the AIC method performs better at out of sample prediction.

	AIC	BIC & p-values
AUC	0.7966429	0.7315301

Table 6: AUC of applying the frequentist methods AIC, BIC and p-values to the full Framingham data set.

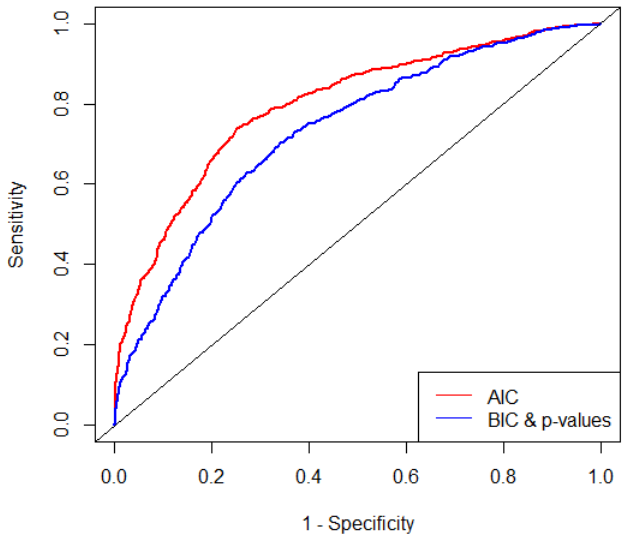


Figure 5: ROC curves of the frequentist approach using the AIC, BIC and p-values methods for the full data set.

4.2 Bayesian

For the Bayesian approach, like discussed in Section 3.6, the MCMC and RJMCMC algorithms are considered and compared. The output of these methods is a large amount of samples for the coefficients. On this set of samples, the previously discussed method of burn-in and thinning is used. The first half is discarded and 1 in every 10 values is kept in the thinning process. This thinned data set is then used to generate fitted values on a test data set; for every sample one fitted value is generated. However, before these can be compared, it must be checked whether the MCMC and RJMCMC algorithms converge.

4.2.1 MCMC convergence

The previously discussed MCMC algorithm is used with 50000 iterations. In Section 3.4.1, multiple methods to assess convergence are discussed for the MCMC algorithm. First of all, a visual inspection of the trace plots can be done. For all 18 parameters (including the design variables), such a comparison is done. An example of this, for the intercept, is shown in Figure 6. In this figure, both the sequence in red and black seem to converge to one set of values. The same holds for all other trace plots, which can be found in Appendix H. Both MCMC simulations are performed on the data set with the fifth bin excluded, both with different random seeds. A second method of establishing that an MCMC sequence has converged is the BGR statistic, which has also been discussed previously. The output of the BGR statistic for each variable can be found in Table 7. Any value above 1.1 would be an indication of a non-convergent MCMC sequence. It can be seen that this is not the case, so this sequence is considered to have converged. Therefore, 50000 MCMC iterations are

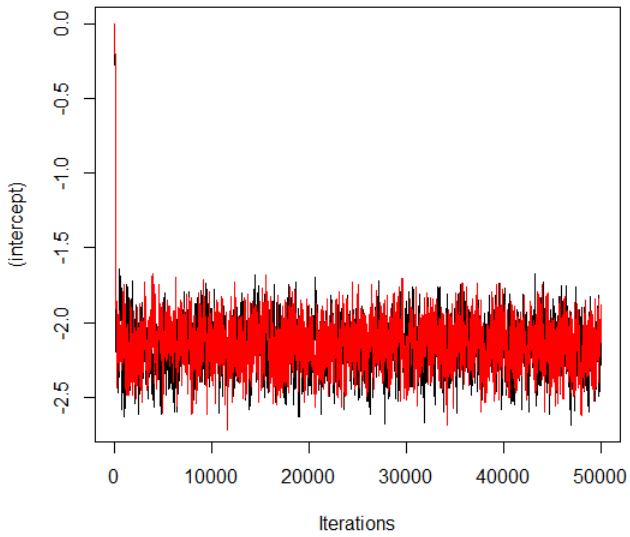


Figure 6: Trace plot of intercept parameter from a sample of 50000 MCMC iterations on the full data set.

considered to be sufficient for the Framingham data set.

Variable	BGR-statistic	Variable	BGR-statistic
intercept	1.002676	totChol	0.9999901
male	1.000494	sysBP	1.00019
age	1.000879	diaBP	1.001012
currentSmoker	1.001425	BMI	1.001147
cigsPerDay	1.001446	heartRate	0.9999987
BPMeds	1.000663	glucose	1.000409
prevalentStroke	1.004721	education_2	1.000354
prevalentHyp	1.001849	education_3	1.000365
diabetes	1.001124	education_4	1.002011

Table 7: BGR-statistics of 50000 MCMC iterations on the full Framingham data set.

Note: all trace plots and the code used to generate the trace plots and the BGR statistic can be found in Appendix H.

4.2.2 RJMCMC convergence

For the RJMCMC simulations, the convergence check is based on inclusion probabilities, as discussed in Section 3.4.3. These inclusion probabilities are computed for every individual variable. For two RJMCMC chains of 300000 iterations each, the inclusion probabilities

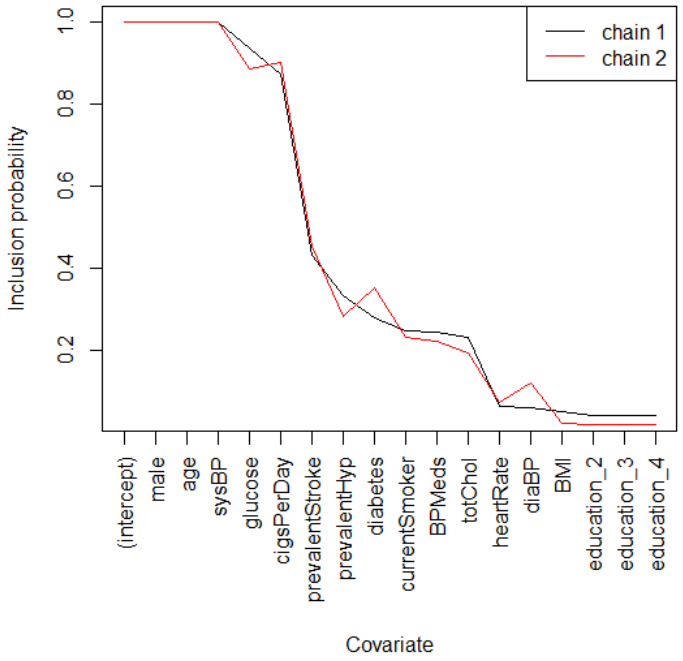


Figure 7: Inclusion probabilities for each parameter of two RJMCMC chains, ordered by highest inclusion probability in chain 1.

for two chains are plotted in Figure 7. This is done in decreasing order of the inclusion probabilities for chain 1. Although the differences between the two chains in terms of inclusion probabilities are not very large, there are some differences that may indicate that the simulations have not converged. For example, when looking at the variable diaBP, the inclusion probability of chain 2 is more than twice as big as the inclusion probability of chain 1 (i.e. 0.12037592 and 0.05958808, respectively). When complete convergence has been achieved, these numbers would most likely be closer. The same holds for some of the other variables. This means that it cannot be concluded that the chains have converged completely, meaning that 300000 iterations are most likely not enough.

4.2.3 Comparison of MCMC and RJMCMC

For determining which of the two approaches among MCMC and RJMCMC performs best, the AUC must be calculated. First, in Figure 8 the ROC curves are displayed for both methods. Just as before, for the MCMC simulation 50000 iterations were used and for the RJMCMC simulation 300000 iterations were used. This might seem to be in the advantage of the RJMCMC simulation, but this method takes more iterations in general, as it also needs to find the best set of covariates.

From the figure, it seems clear that neither of the methods is superior to the other; the ROC curves are very close to each other. This also becomes clear from the AUC's of both methods, which can be found in Table 8. They are very close to each other, so based on the

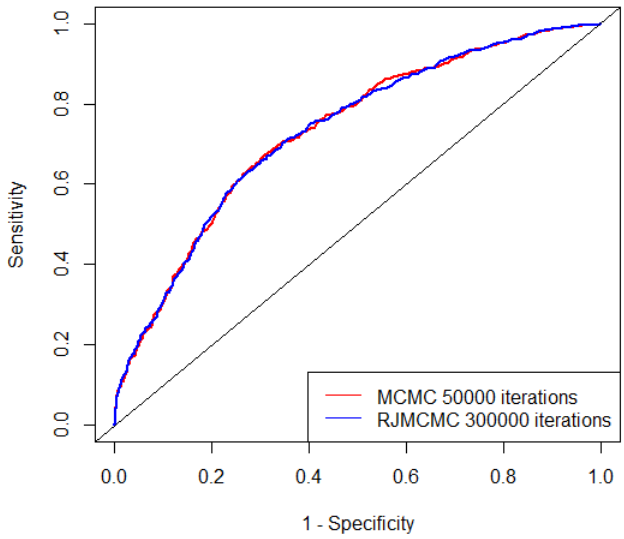


Figure 8: ROC curves of a sample of 50000 MCMC iterations and a sample of 300000 RJMCMC iterations after burn-in and thinning on the full Framingham data set.

AUC, no preference for one method can be given. However, the MCMC method has converged after the 50000 iterations, while this cannot be concluded for the RJMCMC method after 300000 iterations. Also, the MCMC algorithm is less computationally expensive. The RJMCMC algorithm is computationally very expensive, especially running this algorithm for all the halving steps. Therefore, this algorithm is infeasible with the resources of this research, which is discussed in more detail in Section 6. For this reason, the MCMC algorithm is chosen for the further comparisons between the frequentist and Bayesian approach.

Method	AUC
MCMC	0.7320448
RJMCMC	0.7314398

Table 8: AUC of applying the Bayesian approach using 50000 MCMC iterations and 300000 RJMCMC iterations on the full Framingham data set.

4.3 Comparison

Before reducing the sample size, the frequentist and Bayesian approaches are compared on the full data set, using the AUC. The ROC curves of these methods can be found in Figure 9. The frequentist method outperforms the Bayesian method in this case. In Table 9 this also becomes clear. Although both methods perform well according to Equation 10, the

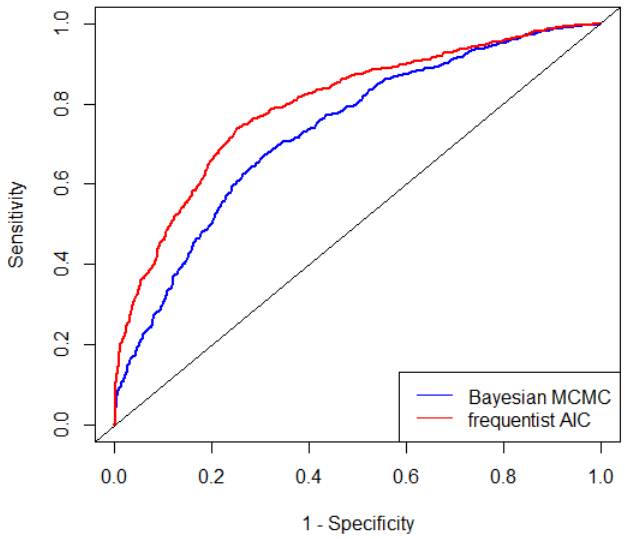


Figure 9: ROC curve of the frequentist approach with the AIC and the Bayesian approach with 50000 MCMC iterations on the full Framingham data set.

frequentist approach has a significantly higher AUC. For this reason, on the full data set, the frequentist approach performs better than the Bayesian approach.

Method	AUC
frequentist AIC	0.7966429
Bayesian MCMC	0.7320448

Table 9: AUC of applying the frequentist approach using the AIC and the Bayesian approach using 50000 MCMC iterations on the full Framingham data set.

4.4 Sample size reduction

When reducing the sample size of the data set, the goal is to determine within each approach what the effects are of reducing the sample size. After doing this, the results of this can be compared, so see which method performs better on average and to see which method performs better when reducing the sample size. For this comparison, the data set is halved five times.

4.4.1 Frequentist

For the frequentist approach, every time the data set is halved, an ROC curve can be produced. As described in Section 3.7, this ROC curve is based on a concatenation of the fitted values of all parts after halving. These are all plotted in Figure 10. It is to be expected

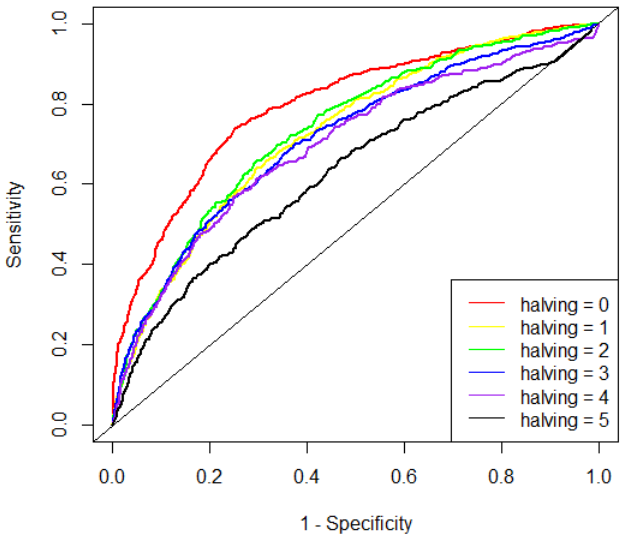


Figure 10: ROC curves for the frequentist approach with the AIC for all different sample sizes, for halving 0 to 5 times.

that when halving the data set multiple times, the prediction becomes less accurate, which would show a lower ROC curve. For the biggest part, this is the trend that is seen in this figure. The red line, representing the frequentist logistic regression using the AIC on the full data set, is the highest line for the biggest part. Below that, there is quite a large gap, but on average, the ROC curves seem to become lower for the smaller sample sizes. This same result can also be seen in Table 10, where the AUC values are given for these ROC curves. It must be noted that the AUC does not go down every time when halving the data set, because in some steps it slightly increases again.

Halving	Percentage of data	AUC
0	100%	0.7966429
1	50%	0.7268606
2	25%	0.7358715
3	12.5%	0.7121422
4	6.25%	0.6969054
5	3.125%	0.6140157

Table 10: AUC of applying the frequentist approach with the AIC for halving the data set 0 to 5 times.

When performing the frequentist logistic regression on the full data set using the three methods that were discussed, the AIC performs best, as shown in Section 4.1. To see whether this difference between the frequentist methods disappears when decreasing the sample size,

the methods using the BIC and p-values are performed on all sample sizes as well. This produces the ROC curves that are found in Appendix G. The AUC of both methods are shown in Table 11. From these figures and the table, it seems that when reducing the sample size for the BIC and p-values methods, the performance decreases less compared to the performance of the frequentist approach using the AIC. However, it must be noted that in every step the AIC method is either superior or almost equal to the BIC and p-values methods when measuring performance with the AUC. Because the goal is always to find the best possible model, the method using the AIC is still the preferred method. For this reason, this method is used to compare the effect of sample size on the performance of the frequentist approach with the Bayesian approach.

		AUC	
Halving	Percentage of data	BIC method	p-values method
0	100%	0.7315301	0.7315301
1	50%	0.7229586	0.7284748
2	25%	0.7073004	0.7093850
3	12.5%	0.6773977	0.6800091
4	6.25%	0.6793421	0.6693955
5	3.125%	0.6189264	0.6189264

Table 11: AUC of applying the frequentist approach with the BIC and p-values methods for halving the data set 0 to 5 times.

4.4.2 Bayesian

For the Bayesian approach with 50000 MCMC iterations, the same procedure is taken. Due to the computational expense of RJMCMC simulations, it is infeasible to compare this method on all sample sizes with the resources of this research. For this reason, this procedure is only taken for the MCMC simulations. The ROC curves can be found in Figure 11. As mentioned before, the ROC curve for the full model is lower for the Bayesian approach, compared to the frequentist approach. However, based on a visual comparison of the graphs, it seems like the decrease in performance is less than for the frequentist approach using the AIC, as the ROC curves decrease slightly for this method. This is discussed in more detail in Section 4.4.3. The values for the AUC can be found in Table 12.

Halving	Percentage of data	AUC
0	100%	0.7320448
1	50%	0.7250983
2	25%	0.7110672
3	12.5%	0.6867748
4	6.25%	0.6856306
5	3.125%	0.6473341

Table 12: AUC of applying the Bayesian approach with 50000 MCMC iterations for halving the data set 0 to 5 times.

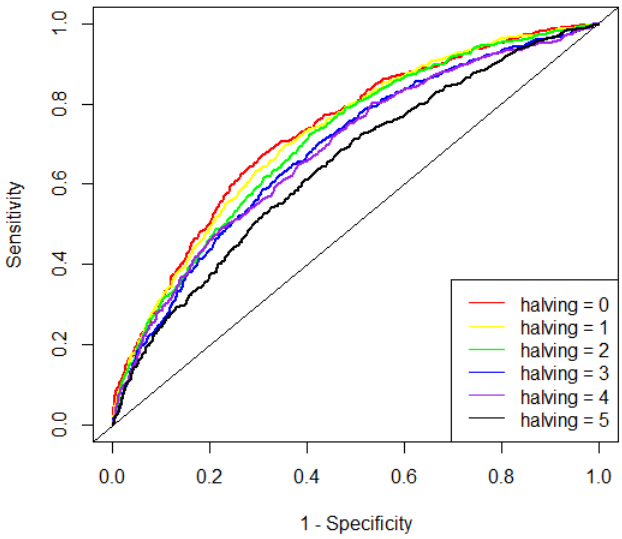


Figure 11: ROC curves for the Bayesian approach with 50000 MCMC iterations for all different sample sizes, for halving 0 to 5 times.

4.4.3 Comparison

To see how well the two approaches perform compared to each other, the AUC's are compared for each sample size. These can be found in Table 13, together with the mean values.

	AUC	
Halving	frequentist AIC	Bayesian MCMC
0	0.7966429	0.7320448
1	0.7268606	0.7250983
2	0.7358715	0.7110672
3	0.7121422	0.6867748
4	0.6969054	0.6856306
5	0.6140157	0.6473341
Mean	0.7137397	0.6979917

Table 13: Comparison of the AUC of both the frequentist approach with the AIC and the Bayesian approach with 50000 MCMC iterations for each sample size, for halving 0 to 5 times.

In the table, it can be seen that for the chosen sample sizes, the frequentist approach using the AIC usually performs better, due to its higher mean for the AUC. In Figure 12, the AUC values are plotted for each amount of times the data set was halved and a linear regression line is drawn for each individual method. This plot shows that both approaches have a different slope, and that the Bayesian approach loses less performance when halving

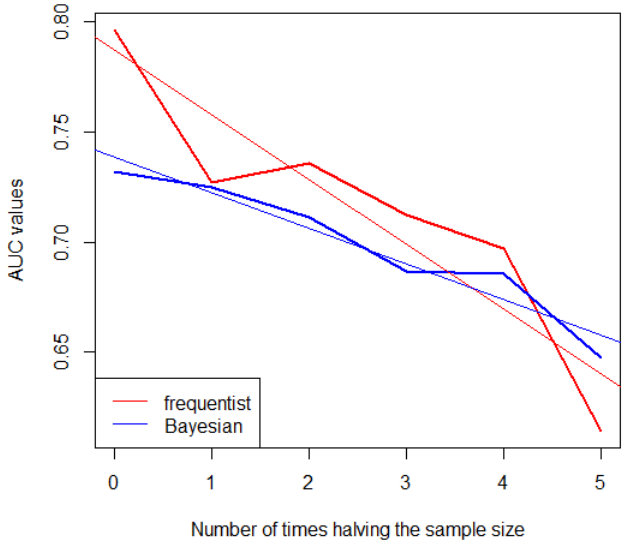


Figure 12: AUC for each amount of times halving the sample size for both approaches, the thin lines represent a linear regression line for the AUC's of each approach. Note: the x-axis is not a linear scale, because halving 0 times represents the full data set and halving 5 times represents 3.125% of the data set.

the sample size. For this reason, it can be concluded that the frequentist approach performs better on average, but that the Bayesian approach performs better when the sample size decreases. Note that the slope of the frequentist approach using the BIC and p-values methods would be less steep. However, because the AIC is better able to identify a model that can describe the data, this is more likely to be the full potential of the frequentist approach, so only the AIC is considered when comparing slopes.

4.5 Interpretation of results

When comparing the different methods, the fitted values of the different models are mainly of interest, because the AUC is a measure of performance that solely uses fitted values. However, it is interesting to see what the models tell about the Framingham data set, in terms of which covariates have an effect on the outcome variable according to the different models. For this comparison between the frequentist and Bayesian approach, only the results from the full sample size are used.

4.5.1 Frequentist

When looking at the frequentist approach, the methods to consider are the AIC, BIC and p-values methods. The output of these methods is found in Appendix F. This shows that the frequentist approach using the AIC finds that the parameters that have an effect on the

outcome variable are male, age, cigsPerDay, totChol, sysBP, glucose and prevalentStroke. The other covariates do not contribute enough to the outcome variable according to this method. When considering the BIC and the p-values methods, the covariates that have an effect are male, age, cigsPerDay, sysBP and glucose. All parameters have a positive value in both outputs, meaning that these covariates positively influence the risk of getting CHD.

4.5.2 Bayesian

For the Bayesian approach, the parameters that have an influence on the outcome variable can be determined through the Highest Posterior Density (HPD) interval. [Hosmer Jr et al., 2013] To construct the HPD interval for a 95% confidence level, 95% of the parameter values with the highest posterior density is selected. The intuition behind this method can be compared with the frequentist confidence interval, even though there are some differences. A variable is considered to have influence on the outcome variable, if this HPD interval does not contain 0. For the MCMC simulations, the HPD intervals can be found in Table 14. The intervals that do not contain 0, meaning they are considered to have an influence on the outcome variable, are male, age, cigsPerDay, sysBP and glucose. It is interesting to see that there are differences with the frequentist approach using the AIC, but that the parameters for the BIC and p-values methods are exactly the same. The HPD intervals of the covariates that are considered to have an influence, are all positive. This means that the Bayesian approach using MCMC simulations also suggests that these relevant parameters positively influence the risk of getting CHD.

Covariate	Lower limit	Upper limit
male	0.2693836	0.7671341
age	0.3960651	0.6716471
currentSmoker	-0.3383827	0.4137985
cigsPerDay	0.04024895	0.41366261
BPMeds	-0.3780389	0.6668417
prevalentStroke	-0.485715	1.544342
prevalentHyp	-0.0931024	0.5305319
diabetes	-0.7175552	0.7219530
totChol	-0.01750111	0.22360268
sysBP	0.1493250	0.5403408
diaBP	-0.2226500	0.1326203
BMI	-0.09520654	0.13451565
heartRate	-0.1498569	0.0777651
glucose	0.0538445	0.3023914
education_2	-0.49905235	0.07625783
education_3	-0.5864782	0.1346342
education_4	-0.4448337	0.3054564

Table 14: HPD intervals of all variables using the 5 cross-validated samples with 50000 iterations each on the full Framingham data set, after burn-in and thinning. These values were generated using the HDInterval package in R.

5 Conclusion

To answer the research question, multiple methods were used. The research question was to determine whether there is a difference in performance between the frequentist and Bayesian logistic regression and whether the sample size influences the performance of both approaches. The benchmark data set used in this research was the Framingham data set.

The measure of performance that was used, is the Area Under the Receiver Operating Characteristic curve, which is a measure of how well a model is able to classify instances of a data set as a success or a failure. Using 5-fold cross-validation, this should have given a balanced view on the performance on the complete data set, as training data and testing data were split from each other. First of all, this was used to determine which method performs best within each of the both approaches. For the frequentist approach, the AIC turned out to be the best method. For the Bayesian approach, due to convergence issues and computational expense of the RJMCMC method, the MCMC method was chosen.

When applying these two approaches, every splitting of the data came with two new values of the AUC, one for each approach. On average, the frequentist approach using the AIC outperformed the Bayesian approach on the Framingham data set. However, when plotting these AUC's as a function of the amount of times the data set was halved, it became clear that the AUC's of the frequentist approach decreased faster than the AUC's of the Bayesian approach. This means that the results on the Framingham data set suggest that the Bayesian approach performs better when decreasing the sample size.

6 Discussion

The goal of this thesis was to see whether there is a difference in the performance of the frequentist and Bayesian logistic regression. The aim was to find out whether the sample size determines the difference in performance between both approaches. When doing this, there are limitations to this research that are important to discuss as well.

First of all, looking at the Bayesian logistic regression, there are a few methods that could influence the performance of the approach. First of all, the MCMC algorithm is only applied to the full set of covariates. Although this did make sure that no important covariates were removed, it is most likely not the sparsest model with a good fit. One way to do this, is to remove covariates one by one and assess the performance of the new model. However, this backwards elimination algorithm would be very computationally expensive, which made it infeasible. Another approach would be to go for the RJMCMC algorithm instead. This model automatically does model selection, which is an advantage. However, due to the large amount of iterations that are required to achieve convergence, it is very computationally expensive. Doing the 300000 iterations that were done in this research took 18.5 hours on the computer[1] used in this research and that was not enough to reach convergence. Therefore, doing even more iterations would not be feasible. However, to make this feasible, a computer with more computational power could be used or more computational time could be spend to do the amount of iterations that are necessary to reach convergence.

Another improvement to this research that could be made, is to consider other methods to determine performance. The AUC is purely focused on performance in terms of classification. Considering other methods of assessing performance would be a possibility, as there are other goodness of fit tests available. When doing this, another possibility would be to generate data of which the true set of parameters or the distribution is known. In this way, it would be possible to assess how close the parameter values predicted are to the actual values.

Lastly, the performance of the two approaches was only tested on the Framingham data set. This cannot be automatically generalised to all data sets with a binary outcome variable. However, due to the fact that this data set does seem to be very suitable for doing a logistic regression, it seems likely that reproducing this analysis for similar data sets would lead to comparable results. This would be something to research in more detail in the future.

[1] The computer used in this research has an Intel Core i5 processor with 8GB of RAM

References

Hirotugu Akaike. A new look at the statistical model identification. *IEEE transactions on automatic control*, 19(6):716–723, 1974.

Sylvain Arlot and Matthieu Lerasle. Why v= 5 is enough in v-fold cross-validation. *arXiv preprint arXiv:1210.5830*, 2012.

M Jésus Bayarri and James O Berger. The interplay of bayesian and frequentist analysis. *Statistical Science*, pages 58–80, 2004.

Michael W Browne. Cross-validation methods. *Journal of mathematical psychology*, 44(1): 108–132, 2000.

Abraham Charnes, Edward L Frome, and Po-Lung Yu. The equivalence of generalized least squares and maximum likelihood estimates in the exponential family. *Journal of the American Statistical Association*, 71(353):169–171, 1976.

Annette J Dobson and Adrian G Barnett. *An introduction to generalized linear models*. CRC press, 2018.

Andrew Gelman, John B Carlin, Hal S Stern, David B Dunson, Aki Vehtari, and Donald B Rubin. *Bayesian data analysis*. CRC press, 2013.

Andrew Gelman, Jessica Hwang, and Aki Vehtari. Understanding predictive information criteria for bayesian models. *Statistics and computing*, 24(6):997–1016, 2014.

Peter J Green. Reversible jump markov chain monte carlo computation and bayesian model determination. *Biometrika*, 82(4):711–732, 1995.

M.A. Grzegorczyk. Lecture notes 10, project statistical reasoning at the university of groningen. a.

M.A. Grzegorczyk. Lecture notes 8, project statistical reasoning at the university of groningen. b.

David W Hosmer Jr, Stanley Lemeshow, and Rodney X Sturdivant. *Applied logistic regression*, volume 398. John Wiley & Sons, 2013.

Daniel McNeish. On using bayesian methods to address small sample problems. *Structural Equation Modeling: A Multidisciplinary Journal*, 23(5):750–773, 2016.

Juan D Rodriguez, Aritz Perez, and Jose A Lozano. Sensitivity analysis of k-fold cross validation in prediction error estimation. *IEEE transactions on pattern analysis and machine intelligence*, 32(3):569–575, 2009.

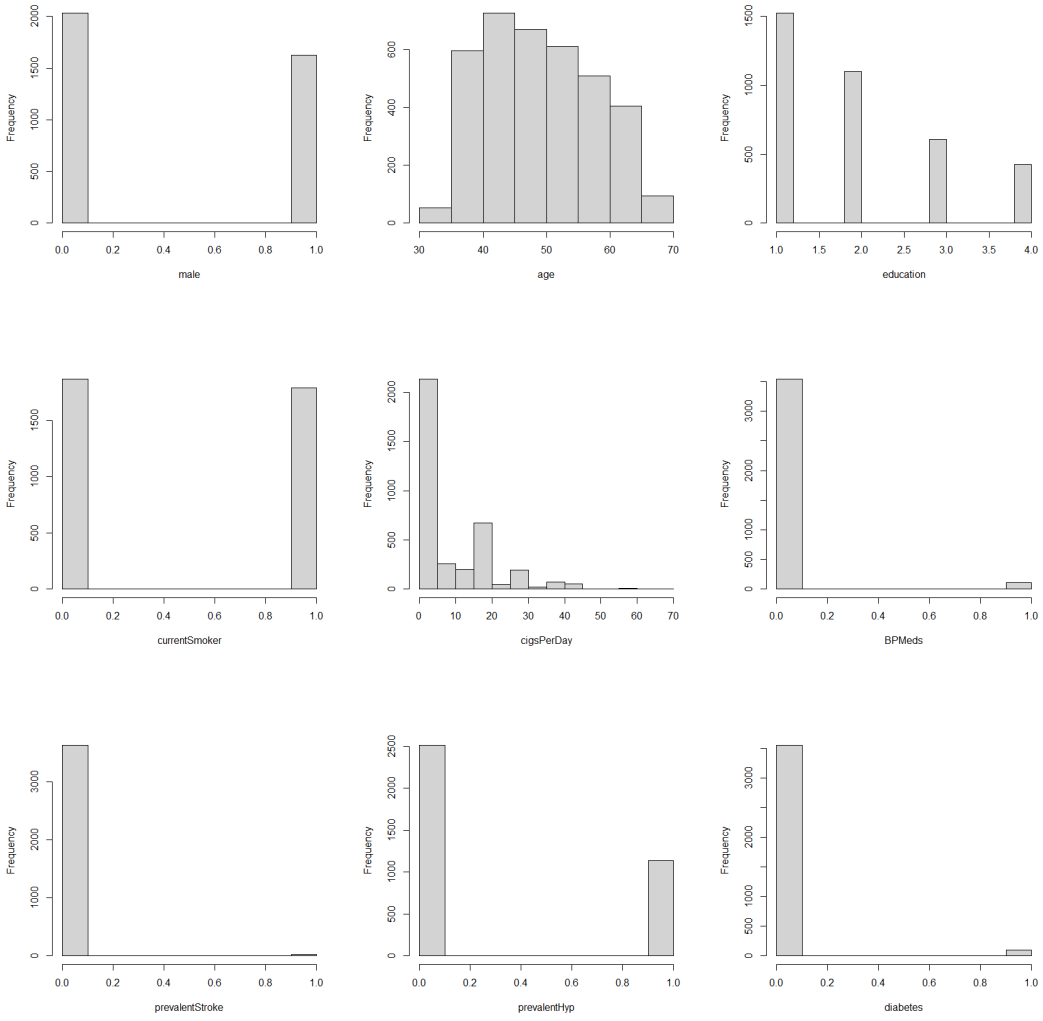
Francisco J Samaniego. *A comparison of the Bayesian and frequentist approaches to estimation*. Springer Science & Business Media, 2010.

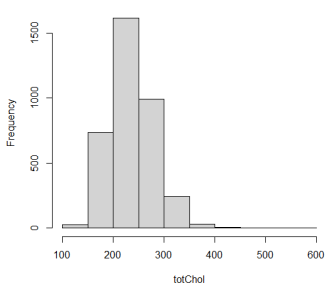
Jon Wakefield. *Bayesian and frequentist regression methods*. Springer Science & Business Media, 2013.

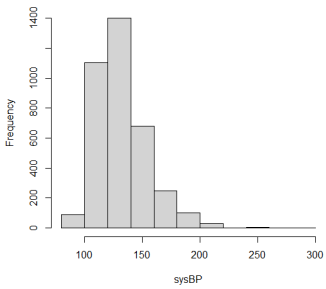
A Header of Framingham data set

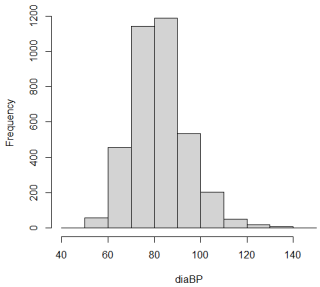
```
> head(framingham)
  male age education currentSmoker cigsPerDay BPMeds prevalentStroke
1    1  39         4             0          0      0                0
2    0  46         2             0          0      0                0
3    1  48         1             1         20      0                0
4    0  61         3             1         30      0                0
5    0  46         3             1         23      0                0
6    0  43         2             0          0      0                0
 prevalentHyp diabetes totChol sysBP diaBP   BMI heartRate glucose
            0        0     195 106.0    70 26.97        80      77
            0        0     250 121.0    81 28.73        95      76
            0        0     245 127.5    80 25.34        75      70
            1        0     225 150.0    95 28.58        65     103
            0        0     285 130.0    84 23.10        85      85
            1        0     228 180.0   110 30.30        77      99
 TenYearCHD
          0
          0
          0
          1
          0
          0
```

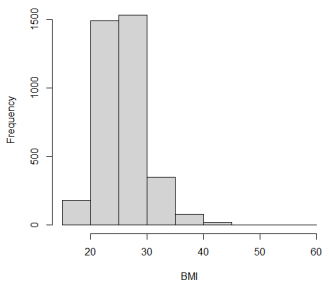
B Histograms of Framingham data set

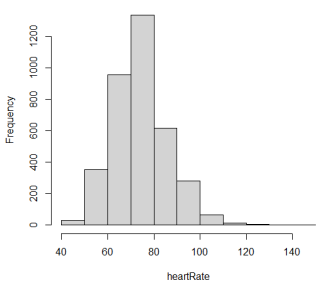


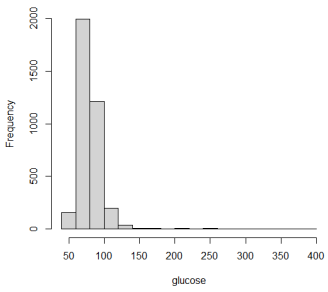

Frequency
0
500
1000
1500
100
200
300
400
500
600
totChol

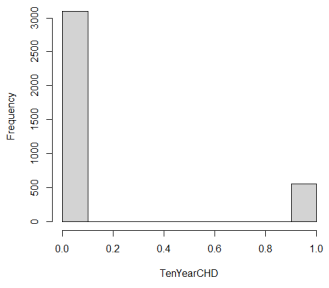

Frequency
0
200
400
600
800
1000
1400
100
150
200
250
300
sysBP


Frequency
0
200
400
600
800
1000
1200
40
60
80
100
120
140
diaBP


Frequency
0
500
1000
1500
20
30
40
50
60
BMI


Frequency
0
200
400
600
800
1000
1200
40
60
80
100
120
140
heartRate


Frequency
0
500
1000
1500
2000
50
100
150
200
250
300
350
400
glucose


Frequency
0
500
1000
1500
2000
2500
3000
0.0
0.2
0.4
0.6
0.8
1.0
TenYearCHD

C Appendix: Code for generating testing and training data

```
# INPUT VARIABLES
# nbins      = number of bins that the data is split up in
# bin_nr     = number of the bin, between 1 and nbins
# halving_nr = number of times the data is halved before
#              splitting up, if not present, it is not haved
# selectquarter = which part of the 2^halving_nr parts is selected

# OUTPUT VARIABLES
# output_list= list of train data and test data

get_train_test_data_selectpart <- function(nbins, bin_nr, halving_nr,
selectquarter) {

  framingham <- read.csv("~/Scriptie/framingham.csv")     # import file
  framingham <- framingham[complete.cases(framingham),]   # remove NULL values

  # standardise all continuous variables
  for (i in c(2,5,10,11,12,13,14,15)) {
    framingham[,i] = scale(framingham[,i], center = TRUE, scale = TRUE)
  }

  # making the design variables for the education categorical variable
  framingham$education_2 <- rep(0, length(framingham$education))
  framingham$education_3 <- rep(0, length(framingham$education))
  framingham$education_4 <- rep(0, length(framingham$education))
  for (i in 1:length(framingham$education)) {
    if (framingham$education[i] == 2) {
      framingham$education_2[i] = 1
    } else if (framingham$education[i] == 3) {
      framingham$education_3[i] = 1
    } else if (framingham$education[i] == 4) {
      framingham$education_4[i] = 1
    }
  }

  # remove the original education column
  framingham$education <- NULL

  # paste TenYearCHD at the end again to keep this as the final column
  TenYearCHD <- framingham$TenYearCHD
  framingham$TenYearCHD <- NULL
  framingham$TenYearCHD <- TenYearCHD

  # set random seed for reproducability and to obtain the same data set
  # every time for comparable results
```

```
set.seed(23)

# shuffle data set
row_order <- sample(nrow(framingham))
framingham <- framingham[row_order,]

# do the halving, only if halving_nr != NULL
if (!missing(halving_nr)) { # if a halving_nr was filled in

  # find the binary representation of the halving number, in reverse order
  rev_binary_select_quarter<- vector()
  num_to_bin = selectquarter - 1
  i=1
  while (num_to_bin > 0) {
    remainder = num_to_bin %% 2
    rev_binary_select_quarter[i] = remainder
    divisor = (num_to_bin - remainder)/2
    num_to_bin = divisor
    i = i + 1
  }

  # shuffle (again) before halving to get different sets every time
  for (k in 1:halving_nr) {
    row_order <- sample(nrow(framingham))
    framingham <- framingham[row_order,]
    if (k > length(rev_binary_select_quarter)) {
      rev_binary_select_quarter[k] = 0
    }

    # select the correct bin, using the binary representation
    if (rev_binary_select_quarter[k] == 0) {
      framingham <- framingham[1:floor(dim(framingham)[1]/2),]
    } else {
      framingham <- framingham[ceiling(dim(framingham)[1]/2):dim(framingham)[1],]
    }
  }
}

# if bin_nr equals 0, then return the full shuffled Framingham data set
# after halving
if (bin_nr == 0) {
  return(framingham)
}

# split up the data in training and testing data
row_nr <- nrow(framingham)
bin_begin <- floor(row_nr/nbins*(bin_nr-1))+1
bin_end <- floor(row_nr/nbins*bin_nr)
```

```
  train_data <- framingham[-(bin_begin:bin_end),]
  test_data <- framingham[bin_begin:bin_end,]

  output_list = list(train_data, test_data)

  return(output_list)
}
```

D Appendix: AIC model selection code

```
# fitted_values_AIC_halves builds the model and selects the best
# model according to the AIC.
# halving_nr      =   the number of times the data set is halved

# returns         =   a list of full_fitted_bin and full_test_data
# full_fitted_bin =   a vector of fitted values
# full_test_data  =   the data this vector of fitted values is for


fitted_values_AIC_halves <- function(halving_nr) {

  full_fitted_bin <- vector()
  full_test_data <- NULL

  for (w in 1:(2^halving_nr)) {

    nr_bins = 5

    model_AIC_list <- list()
    fitted_AIC_list <- list()

    for (j in 1:nr_bins) {

      # add halving number when needed
      train_test_data = get_train_test_data_selectpart(nr_bins,j, halving_nr, w)
      test_data = train_test_data[[2]]
      data = train_test_data[[1]]

      df <- data
      proposedDf <- data

      model <- glm(TenYearCHD ~ ., family = binomial(link="logit"), data=data)

      nr_vars = ncol(df)-2

      education_in_data = TRUE

      i = 1

      # backwards elimination procedure
      while (i < nr_vars) {
        if ((i == nr_vars - 1)&&education_in_data) {

          # take out all design variables at once if this is the education variable
          proposedDf <- subset(df, select=-c(i, i+1, i+2))
          proposedModel <- glm(TenYearCHD ~., family = binomial(link="logit"),
```

```
                    data=proposedDf)

    if (AIC(proposedModel) <= AIC(model)) {
      df <- proposedDf
      model <- proposedModel
      columnnames <- colnames(df)
      education_in_data = FALSE
      i = 0
    }
  } else {
    proposedDf <- subset(df, select=-c(i))
    proposedModel <- glm(TenYearCHD ~., family = binomial(link="logit"),
                    data=proposedDf)

    if (AIC(proposedModel) <= AIC(model)) {
      df <- proposedDf
      model <- proposedModel
      columnnames <- colnames(df)
      i = 0
    }
  }

  i = i + 1
  nr_vars = ncol(df)-2
}

nr_vars = ncol(data)-2

# forwards selection test run
for (i in 1:(nr_vars - 1)) {
  if ((colnames(data)[i] %in% colnames(df)) == FALSE) {

    if (i == nr_vars - 1) {
      new_df <- df
      new_df$education_2 <- data$education_2
      new_df$education_3 <- data$education_3
      new_df$education_4 <- data$education_4

      # add all education design variables
      biggerModel <- glm(TenYearCHD ~ .,
                         family = binomial(link = "logit"),
                         data = new_df)
    } else {
      new_df <- df
      new_df[,dim(df)[2]+1] <- data[,i]
      colnames(new_df) <- c(colnames(df), colnames(data)[i])

      biggerModel <- glm(TenYearCHD ~ .,
```

```
                              family = binomial(link = "logit"),
                              data = new_df)
          }

          if(AIC(biggerModel) < AIC(model)) {
            df <- new_df
            columnnames <- colnames(df)
            model <- biggerModel
            i=0
          }
        }
      }

      # generate fitted values on the testing data that was split up
      df <- test_data[,columnnames]
      n = dim(df)[1]
      p = dim(df)[2]
      X_mat <- as.matrix(cbind(rep(1,n),df[,1:(p-1)]))
      g_x_AIC <- X_mat%*%as.matrix(coef(model))
      test_fitted_vals_AIC <- exp(g_x_AIC)/(1+exp(g_x_AIC))
      model_AIC_list[[length(model_AIC_list) + 1]] <- model
      fitted_AIC_list[[length(fitted_AIC_list) + 1]] <- test_fitted_vals_AIC
    }

    # concatenate all fitted values
    for (i in 1:nr_bins) {
      full_fitted_bin <- c(full_fitted_bin, fitted_AIC_list[[i]])
    }

    # concatenate all testing data to compare with the fitted values
    full_test_data <- rbind(full_test_data,
            get_train_test_data_selectpart(nr_bins, 0, halving_nr, w))
  }
  return(list(full_fitted_bin, full_test_data))
}
```

E Bayesian Logistic Regression code

Note: the code below was written by prof. dr. Grzegorczyk. Some changes have been made to ensure it suits the Framingham data set and to increase the speed of the algorithm.

```
library(tictoc)

Compute_log_likelihood <- function(y, X_full, betas) {
  return(sum((y-1)*(X_full%*%betas))-sum(log(1+exp(-X_full%*%betas))))
}

################################################################################

Compute_log_prior  <- function(betas,mu,sigma2){

  log_prior = 0;

  for (i in 1:length(betas)){
    if(betas[i]!=0){
      log_prior = log_prior - log(sqrt(2*pi)) - log(sqrt(sigma2))  -0.5/sigma2 *
      (betas[i]-mu)^2
    }
  }
  return(log_prior)
}

################################################################################

# INPUT VARIABLES
# y          = binary response vector
# X_full     = full design matrix (with intercept and all covariates in)
# iterations = no. of MCMC iterations to be performed
# eps        = interval length for MH-MCMC moves is [-eps,eps]
# mu, sigma2 = hyperparameters of regression parameter priors: beta_j ~N(mu,sigma2)
# p          = probability that a MH-MCMC move is performed
#              with probability (1-p) a RJMCMC move is performed


Log_regression_MCMC <- function(y, X_full, iterations,eps,mu,sigma2){

  k = dim(X_full)[2] - 1 # no. of covariates

  betas = rep(0,1+k)     # initialise all k+1 regression coefficients with 0


  #Compute initial log score = log_likelihood + log_prior
  log_likelihood = Compute_log_likelihood(y,X_full,betas)
  log_prior      = Compute_log_prior(betas,mu,sigma2)
```

```
  log_score = log_likelihood + log_prior

  # Store initialisation
  BETAS      = betas
  LOG_SCORES = log_score


  ######
  # ITERATIONS
  ######
  for (t in 2:iterations){

      for (j in 1:(k+1)){
          betas_new    = betas
          betas_new[j] = betas_new[j] + runif(1,-eps,eps)

          log_likelihood_new = Compute_log_likelihood(y,X_full,betas_new)
          log_prior_new      = Compute_log_prior(betas_new,mu,sigma2)

          A = min(c(1,exp(log_likelihood_new - log_likelihood + log_prior_new -
                    log_prior)))

          u = runif(1)
          if (u<A){
            betas = betas_new
            log_likelihood = log_likelihood_new
            log_prior = log_prior_new
            log_score = log_likelihood + log_prior
          }

      }

    LOG_SCORES = rbind(LOG_SCORES,log_score)

    BETAS  = rbind(BETAS,  betas)
  }

  return(list(BETAS,LOG_SCORES))

}

#################################################################################

iterations = 50000
eps        = 0.1
mu         = 0
sigma2     = 1
```

```
# Run the MCMC simulation

nr_bins = 5
halving = 2
quarter = 3
OUTPUT_MCMC = list()
fitted_MCMC = list()

for (i in 1:nr_bins) {

  train_test_data = get_train_test_data_selectquarter(nr_bins,i,halving, quarter)
  test_data = train_test_data[[2]]
  data = train_test_data[[1]]

  # set a random seed to ensure it is different from the seed set in the
  # algorithm to split up data
  set.seed(i*107)

  n = dim(data)[1]   # no. of observations
  p = dim(data)[2]   # no. of covariates + 1 (last column is response y)

  # Prepare response vector
  y = data[,p]

  # Add a first column for the intercept
  X_full = as.matrix(cbind(rep(1,n),data[,1:(p-1)]))

  tic()
  OUT_MCMC = Log_regression_MCMC(y, X_full, iterations, eps, mu, sigma2)
  toc()

  colnames(OUT_MCMC[[1]]) <- c("(intercept)", colnames(data)[1:(p-1)])

  OUTPUT_MCMC[[length(OUTPUT_MCMC) + 1]] <- OUT_MCMC

  # thinning and burn-in
  OUT_MCMC[[1]] <- OUT_MCMC[[1]][seq(25000,50000,by=10),]

  n = dim(test_data)[1]   # no. of observations,
  p = dim(test_data)[2]   # no. of covariates + 1 (last column is response y)

  X_test = as.matrix(cbind(rep(1,n),test_data[,1:(p-1)]))

  g_x <- X_test%*%t(OUT_MCMC[[1]])
  fitted_values_MCMC <- exp(g_x)/(1+exp(g_x))
  fitted_MCMC <- vector()
  for (i in 1:dim(fitted_values_MCMC)[1]) {
    fitted_MCMC[i] = median(fitted_values_MCMC[i,])
```

```
  }
  fitted_MCMC[[length(fitted_MCMC)+1]] <- fitted_MCMC
}

save(OUTPUT_MCMC,file="Output MCMC 50000it.RData")
save(fitted_MCMC,file="Fitted MCMC 50000it.RData")
```

F Output AIC, BIC and p-values Methods

AIC

```
Call:  glm(formula = TenYearCHD ~ ., family = binomial(link =
"logit"), data = df)

Coefficients:
    (Intercept)             male              age       cigsPerDay
        -2.2631           0.5997           0.5238           0.2444
    totChol            sysBP          glucose  prevalentStroke
    0.1055           0.4005           0.1920           0.8658

Degrees of Freedom: 2925 Total (i.e. Null);  2918 Residual
Null Deviance:     2495
Residual Deviance: 2208  AIC: 2224
```

BIC & p-values

```
Call:  glm(formula = TenYearCHD ~ ., family = binomial(link =
"logit"), data = df)

Coefficients:
(Intercept)         male          age   cigsPerDay        sysBP
    -2.2350       0.5685       0.5393       0.2470       0.4143
    glucose
    0.1929

Degrees of Freedom: 2925 Total (i.e. Null);  2920 Residual
Null Deviance:     2495
Residual Deviance: 2214  AIC: 2226
```

G ROC curves BIC and p-values methods

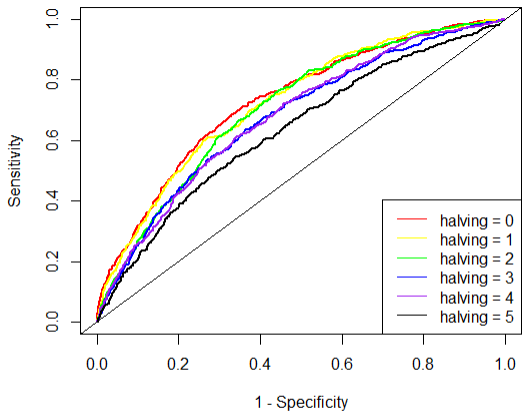


Figure 13: ROC curves for the frequentist approach with the BIC for all different sample sizes, for halving 0 to 5 times.

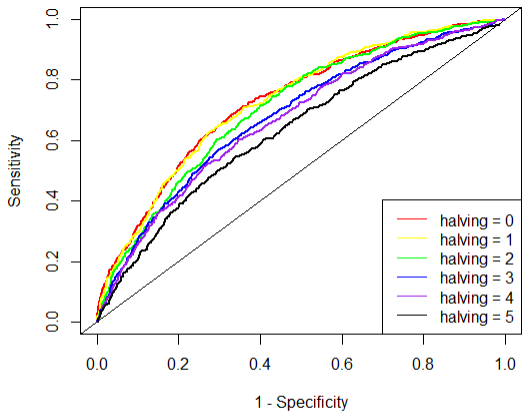


Figure 14: ROC curves for the frequentist approach with the p-values method for all different sample sizes, for halving 0 to 5 times.

H Appendix: MCMC convergence check

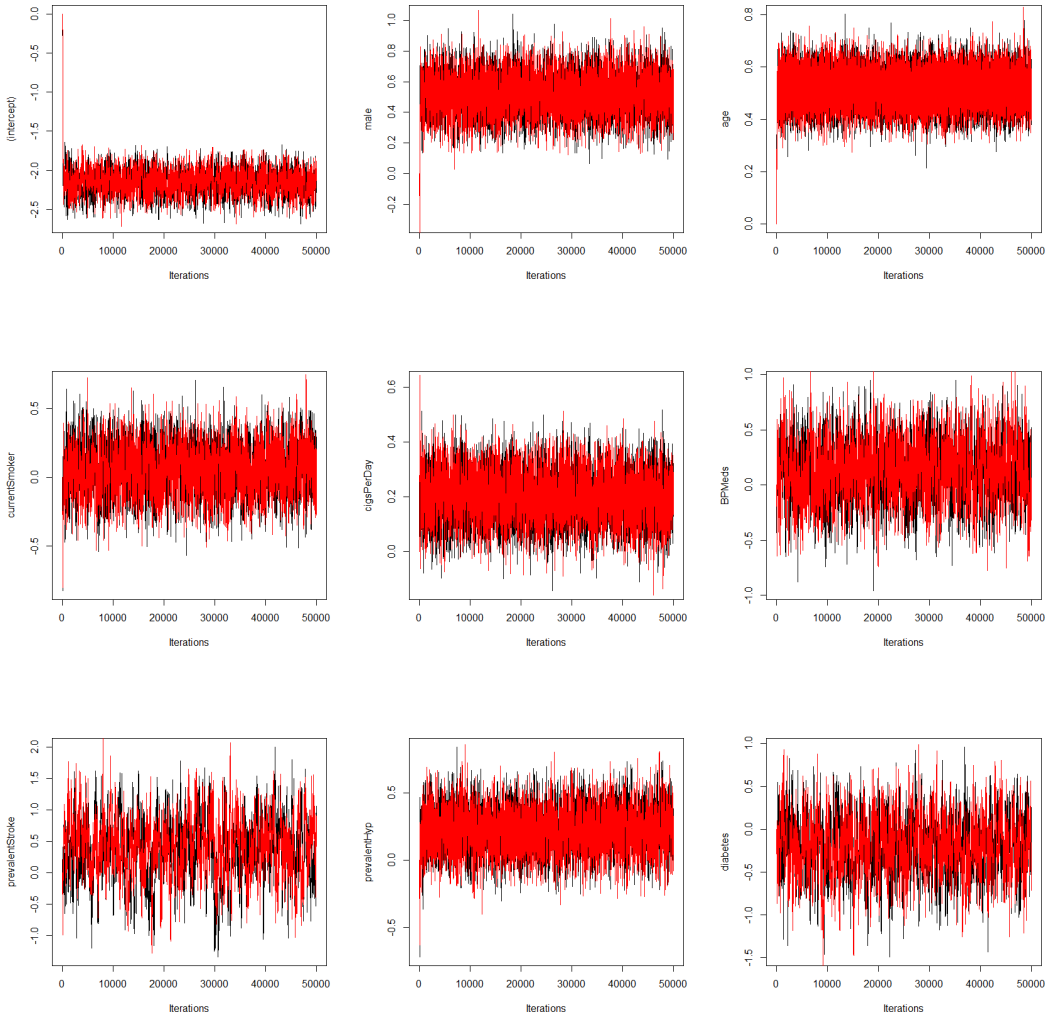
H.1 R code

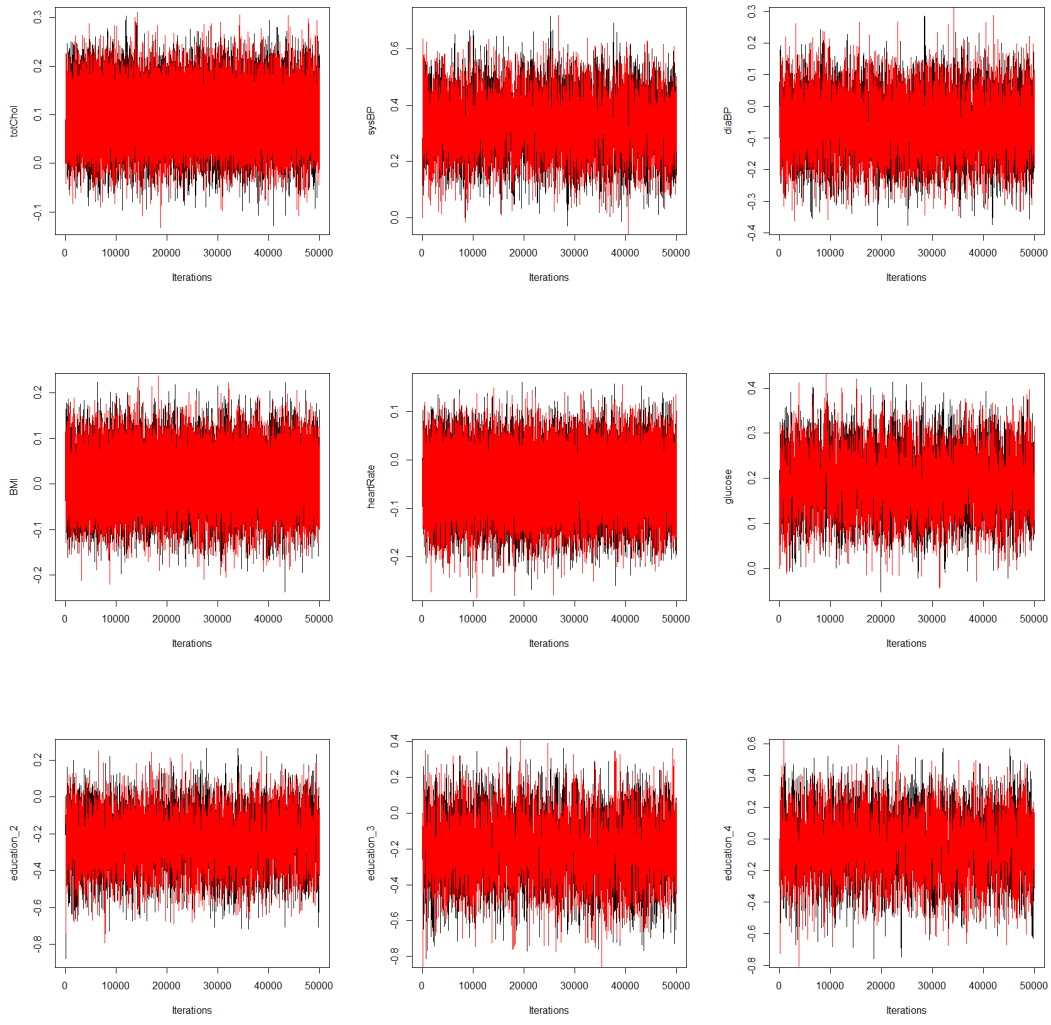
The code for the BGR statistic and the trace plots is as follows.

```
BGR_statistics <- function(chain_1, chain_2) {
  data <- get_train_test_data(5,1)
  colnam <- colnames(data[[1]])
  colnam <- colnam[1:(length(colnam)-1)]
  colnam <- c("(intercept)", colnam)

  n = dim(chain_1)[1] # number of iterations
  nr_vars = dim(chain_1)[2]

  m = 2       # number of chains

  BGR_stats <- vector()


  for (k in 1:nr_vars) {
    chain_list <- list(chain_1[,k], chain_2[,k])
    # make the trace plot
    plot(x=1:n, y=chain_list[[1]], type='l', xlab="Iterations",
        ylab=colnam[k])
    lines(x=1:n, y=chain_list[[2]], type='l', xlab="Iterations",
        ylab=colnam[k], col="red")


    s_j2 <- vector()
    avg_chain <- vector()
    W = 0
    B = 0
    tot_avg <- (mean(chain_list[[1]]) + mean(chain_list[[2]]))/2

    for (j in 1:m) {
      s_j2[j] = 0
      avg_chain[j] = mean(chain_list[[j]])
      for (i in 1:n) {
        s_j2[j] = s_j2[j] + 1/(n-1)*(chain_list[[j]][i] -
            avg_chain[j])^2
      }
      W = W + 1/m*s_j2[j]
      B = B + 1/(m-1)*n*(avg_chain[j] - tot_avg)^2
    }

    V = (n-1)/n*W + 1/n*B

    BGR_stats[k] <- sqrt(V/W)
```

```
  }
  BGR_stats <- t(BGR_stats)
  colnames(BGR_stats) <- colnam

  return(BGR_stats)
}
```

H.2 Trace plots




totChol
sysBP
diaBP
BMI
heartRate
glucose
education_2
education_3
education_4
Iterations