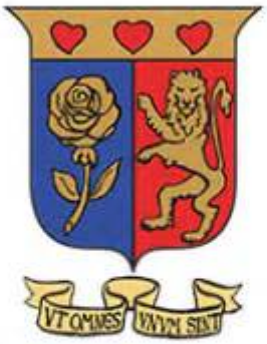
2020

Identifying the best method to correct for missing data, a case of HIV/TB co-infection in Kenya.

Mwaro, Joshua Owuori
Strathmore Institute of Mathematical Sciences
Strathmore University



Strathmore
UNIVERSITY

Identifying the Best Method to Correct for Missing Data, A Case of HIV/TB Co-infection in Kenya.

Joshua Owuori Mwaro

Master of Science in Statistical Science.

[2020]

Identifying the Best Method to Correct for Missing Data, A Case of HIV/TB Co-infection in Kenya

Joshua Owuori Mwaro

Submitted in partial fulfillment of the requirements for the Degree of Master of Science in Statistical Science at Strathmore University.

[Strathmore Institute of Mathematical Sciences]
Strathmore University
Nairobi, Kenya

[June, 2020]

I declare that this work has not been previously submitted and approved for the award of a degree by this or any other University. To the best of my knowledge and belief, the thesis contains no material previously published or written by another person except where due reference is made in the thesis itself.

Joshua Owuori Mwaro

Date: October 22, 2020

Approval

The thesis of Joshua Owuori Mwaro was reviewed and approved by the following:

Dr. Collins Odhiambo
Strathmore Institute of Mathematical Science.
Strathmore University

Dr. Linda Chaba
Strathmore Institute of Mathematical Science.
Strathmore University

Ferdinand Othieno
Dean, Strathmore Institute of Mathematical Sciences,
Strathmore University

Professor Ruth Kiraka,
Dean, School of Graduate Studies,
Strathmore University

Abstract

Having missing information is almost inevitable in research, but many researchers only report on complete cases. Here we review the missing data theory, missingness characteristics, look at the background information, importance of studying missing data, the most common ways of correcting for missing data then extend to Kenyan HIV/ TB co-infection setting. We review most of the existing methods of dealing with missing data and what other scholars have done in the missing data area. In the methodology section, we outline and give characteristics and features of the four methods for dealing with missing data(Analysis of complete cases only, Mean/Single imputation method, MLE method, and Multiple Imputation method.) which our study is focused on. We also test the four methods on the simulated data then apply the same procedure on the real Kenyan HIV/TB co-infection data.

Results showed that analysis of data that was corrected for missingness using: complete cases only, weighted method, likelihood-based, and multiple imputation estimated the Kenyan HIV/TB co-infection rate to be 29%, 27%, 26%, and 21% respectively. The results indicate that MI is the best approach to correct for missing data as it does not overestimate the HIV & TB co-infection rate.

Keywords
Missing data, HIV/TB Co-infection, Imputation, Missing completely at random, Missing at random, Missing not at random.

Contents

List of Figures

List of Tables

Abbreviations

MCAR: Missing completely at random.
MAR: Missing at random.
MNAR: Missing not at random.
MLE: Maximum Likelihood Estimation
MI: Multiple Imputation.
MIDS: Multiply Imputed Datasets
MIRA: Multiply Imputed Repeated Analysis
MIPO: Multiple Imputed Pooled Outcomes
PLWHIV:People Living With HIV
HAART: Highly Active Antiretroviral Therapy
TB :tuberculosis
HIV: human immunodeficiency virus
MCMC:Markov chain Monte Carlo
MICE:Multiple imputation by chained equations
NTLD-P:National Tuberculosis, Leprosy and Lung Disease Program
ISLR: Introduction to Statistical Learning with applications in R
AUC: Area Under the receiver-operator Curve

Acknowledgements

Foremost, I thank God for giving an opportunity to undertake my masters studies, keeping me in good health, and providing resources that enabled me successfully complete the studies.

My sincere gratitude goes to my supervisors: Dr. Collins Odhiambo and Dr. Linda Chaba for their guidance, patience, motivation and immense knowledge that enabled me write this thesis.

I also thank the Strathmore Institute of Mathematical Sciences for the good administration during the study period, very knowledgeable lecturers, very good internal and external examiners and good panelists during my thesis defence, they enabled me shape up this final document.

My sincere gratitude also goes to my mentor Dr. David Khaoya who provided me with guidance, knowledge, advice and opportunities that enabled me to:successfully complete my masters studies and apply class work in my career.

I thank my classmates Susan Mutua and Jean Jesang for the support you gave me, the exchange of ideas and the time you took to review my work or help me fix an erroneous code whenever I requested you.

Special thanks go to my thank my family: Mum, Dad, Aunts and brothers for your spiritual and financial support and constant follow up on my progress.

Lastly I sincerely thank my friends: Susan Mutua, Stephen Onteri, Happiness Kemunto, Shirley Serem, Arthur Otieno, James Karabu, Camilla Wasike, Tracy Ndonji, Nickson Mukamani and Ian Muyumba, you were there for me when I needed you most. I thank all who directly and/or indirectly contributed to my success and any omission in this brief acknowledgement does not mean lack of gratitude.

Chapter 1

Introduction

1.1 Background

In any research field, the main challenge is incomplete data. Some of the causes of missing data range from outright refusals, such as when asking more personal information(like income)or when asking for information that make a respondent feel embarrassed. Another cause could be in self administered surveys in which respondents may overlook or forget to answer some questions. Missing data can also arise when some questions are not applicable to some respondents or they could genuinely be lacking knowledge of the information being asked.The causes could also be from the researcher's end during data collection or data entry. Another common cause of missing data occurs when respondents die or drop out from longitudinal studies(Rubin, 1976). Missing data always occur in research, they undermine the validity of research findings by introducing bias and reduction of statistical power. It is therefore important to carefully examine any given data to identify the missingness proportion, mechanism and pattern and consequently correct the missingness. This will enable application of the correct method in a given setting.Addressing missing data problems became robust in the late 80's when Little & Rubin published the book Statistical Analysis with Missing Data (Graham, 2009). Most of the researchers understand the existence of missing data but never report how they dealt with it, some address the issue by assuming that the missingness completely occured at random (MCAR) which rarely happens in practice, and apply ad hoc means(e.g complete case analysis and single imputation) when analyzing (Peng et al., 2006; Karahalios et al., 2012).
A few researchers go further to apply improved imputation and likelihood based methods. However, they tend to apply them across all the three missingness mechanisms. Approaches to correcting missingness have not applied to Kenyan HIV/TB co-infection data setting. Here we seek to identify the best method to correct for missingness under different mechanisms in Kenyan HIV/TB co-infection setting.

1.2 Problem statement

TB is the leading comorbidity disease among the people living with HIV (PLWHIV) and a leading morbidity source in HIV patients. Globally, these two diseases are the main transmissible diseases causing death. This makes it important to have valid and accurate statistics about the co-infection rates of these killer diseases.
An important statistic is the prevalence rate of the co-infection. Globally, there are wide variations from the true to the reported prevalence rates both in-country and between countries. One of the reasons attributed to the variations is under-reporting which results from missing data.
Missing data is mainly caused by irregular collection of information from HIV patients, no show of the patients for the baseline or consecutive checkup and nondisclosure of some information.

Majority of the researchers in public health address the missing data problem by using the default methods(ad-hoc) in the statistical software. Other researchers improve and apply single imputation. These techniques are commonly applied without considering the proportion, pattern or mechanism under which data are missing thus yielding biasness and loss of power in the study.
We therefore need to identify the best method to correct for missingness in each mechanism (MCAR, MAR & MNAR). As we do this, we are considering varying proportions of missing data as well as varying sample sizes.

1.3 Objectives

1.3.1 Main objective

Identifying the best method to correct for missing data in Kenyan TB/HIV co-infection setting.

Specific objectives

1. To systematically evaluate how each of the following methods performs in correcting missingness:
   (a) Analysis of complete cases only.
   (b) Mean/Single imputation method.
   (c) MLE method.
   (d) Multiple Imputation method.
2. To apply the four methods in Kenyan TB-HIV coinfection data setting.
3. To identify the best method to correct for missingness on Kenyan TB-HIV coinfection data.

1.4 Justification

The occurrence of missing data in research is almost inevitable but researchers are non cognisant of this and fail to report on the same. They unknowingly apply the default ad-hoc methods found in the analysis software. Most of the ad-hoc methods to correct for missing data have been applied more generally without considering the missing mechanism or pattern hence the need to identify the best method under each mechanism. Further more, the methods have not been applied in a Kenyan HIV/TB co-infection data setting therefore we fill this gap.

Chapter 2

Literature review

Having incomplete data when conducting research is unavoidable. Data can be entirely missing for a given observation or can be missing for one or more items on an observation i.e when no information is recorded from a respondent and when incomplete information is recorded from a respondent respectively. Our focus is on item level(hence forth referred to as missing data in this work) missing information because of the problems it causes such as:reduction in statistical power, bias in parameters, reduction in representativeness of the selected sample, and complication of the analysis (Kang, 2013).
Dong and Peng suggested three aspects of dealing with missing data, the; mechanism, proportion and how missinges occur between and/or among variables within a dataset(pattern). Researchers are expected to consider each aspect before deciding how to correct for missing data (Dong and Peng, 2013).

2.0.1 Missingness mechanisms

Nakagawa (2015) described missing data mechanisms as the statistical relationship between observations and the probability of missing data. Mechanisms that bring about missingess in data include: missingness at random (MAR), missingness occuring completely at random (MCAR), and missingness not occuring at random (MNAR.The mechanisms represent statistical relationship between observations and the probability of missing data. We explain the mechanisms(classes of missing data) by partitioning a matrix of data D into observed set($D_{observed}$) and unobserved set($D_{missing}$) parts (Rubin, 1976).

Missing data is MAR if the chance of a data value being unobserved relys only on the available observations; $D_{observed}$, not on unobserved; $D_{missing}$, after accounting for the $D_{observed}$ (Nakai and Ke, 2011).
Mathematically, if M is a matrix of missingness whose dimensions are like those of D. Each element of M is coded 1/0, representing D being observed/missing respectively. Let the M follow the distribution : $P(M|D, \varphi)$, φ is the parameter for missingness. Schafer (1997) suggested that if the distribution of M is in form of the model:

$$P(M|D,\ \varphi) = P(M|D_{observed}, D_{missing}, \varphi) = (M|D_{observed},\ \varphi) \tag{2.1}$$

then this missing mechanism will be MAR.
Allison (2001) further explains missing data on M being MAR if after accounting for all variables, chances of no observation on a given variable in D is not associated with value of the observation.
Mathematically, let there be two variables M & X, if X is always completely observed and M occasionally incompletely observed.Then the occassional missingness in M is MAR if:

$$Pr(M_{missing}|M,\ X) = Pr(M_{missing}|\ X) \tag{2.2}$$

i.e the conditional chance of having incomplete data on M given both X and M, is the chance of having incomplete data on M given X only. To illustrate MAR mechanism in relation to our work: suppose the chance of having incomplete data on HIV/TB coinfection is based on a persons attendance of clinics, however in each clinic attendance category, the risk of incomplete HIV/TB coinfection data is unrelated to HIV/TB coinfection. Generally, missing data fail to fullfil the MAR assumption if cases with incomplete data on a given variable have lower/higher values on the variable than the cases with complete data on the variable, after considering other observed variables (Allison, 2001).
Data is MCAR if the chance of being unobserved neither rely on the completely available information; $D_{observed}$ nor incomplete information; $D_{missing}$. Under MCAR, M's distribution is modeled as:
:

$$P(M|D,\ \varphi) = P(M|D_{observed}, D_{missing},\ \varphi) = (M|\varphi) \tag{2.3}$$

A simplified definition of MCAR was given by Allison (2001), given the chance of incomplete data on a variable neither relates to its value nor value(s) of another variable(s) within a given data matrix then the missingness mechanism is MCAR.

Finally, MNAR mechanism refers to when; conditional on the observed data, the chance of having missing data on a variable depends on the value of the unobserved data. An example is being likely to have missing data in an HIV & TB co-infection variable if HIV TB co-infected clients who do not attend clinics are more likely not to provide information as compared to those who attend clinics (Galimard et al., 2018).

2.0.2 Missing Data Patterns.

Missing data pattern refers to how the incomplete data occur/are arranged between and/or among variables and individual cases (McNeish, 2017). Three missing data patterns occur: univariate, monotone and arbitrary.

Dong and Peng (2013) explained the three patterns as: Letting our data set have m variables designated: $X_1, X_2, ..., X_m$. Missingness pattern in a data set is said to be **univariate** if identical records have incomplete data on one/more of the m variables.A pattern is **monotone** if the variables can occur in an arrangement such that, when X_i is missing, then X_{i+1},X_{i+2},...,X_m are missing. This pattern frequently happens in longitudinal studies after dropouts.The **arbitrary** missing data pattern refers to when missingness randomly occur in any variable(s) for any participant(s), this is computationally harder to handle than the other two patterns.

How to Diagnose the Missing Data patterns and Mechanism

Missing data patterns can be visualized using R's missingmap function of the Amelia library (Honaker et al., 2011). To identify if data is MCAR, for each variable with missing data we create a dummy variable with 1 representing missing observation and 0 otherwise. T-tests are conducted between the observed and unobserved groups. If all the t-tests are not significant then the missing values are MCAR, else are MAR or MNAR. This method becomes tedious when the variables are many in a data set (McKnight et al., 2007). Little (1988) solved the problem of conducting several t-tests by introducing a multivariate procedure which produces a Chi-square statistic for the entire data set.This procedure can be done by the LittleMCAR function in the BaylorEdPsych library of R. The null hypothesis is that the data is MCAR, if the resultant chi-square value is significant then the missingness is not MCAR, could be MAR or MNAR. The method has two major disadvantages: (1) when the observed and unobserved

groups dont balance, the statistical power of the data may be weak. (2)there can be a non-significant result even if missingness is MAR or MNAR, particularly if missing values in a variable are related to the high and low values of another variable.
Nakagawa (2015) stated that there are neither tests nor visual techniques to distinguish between MAR and MNAR. Therefore there is need to ascertain whether or not missing values are MNAR based on the systems under investigation. Graphical methods are also useful in diagnosing missingness. This can be done in R by the pairs.panels function from the psych library. Figure 2.1 below illustrates a visual assessment of our simulation data(7% MAR). Paired panel plots of the data matrix and missingness matrix for the simulated data set, Gender, Age, weight, and HIVTBConifection are the variables in the data matrix, while Gender.1, Age.1, weight.1, and HIVTBConifection.1 is the missingness for the variables in the data matrix. The upper triangle panels displays Spearman correlations, the lower triangle panels show scatterplots with lowess (locally weighted scatterplot smoothing) lines. The diagonals show histograms. There is some evidence for MAR because of the moderate correlation between HIVTBConifection and Age.1, Gender.1 and weight.1 (Nakagawa, 2015).

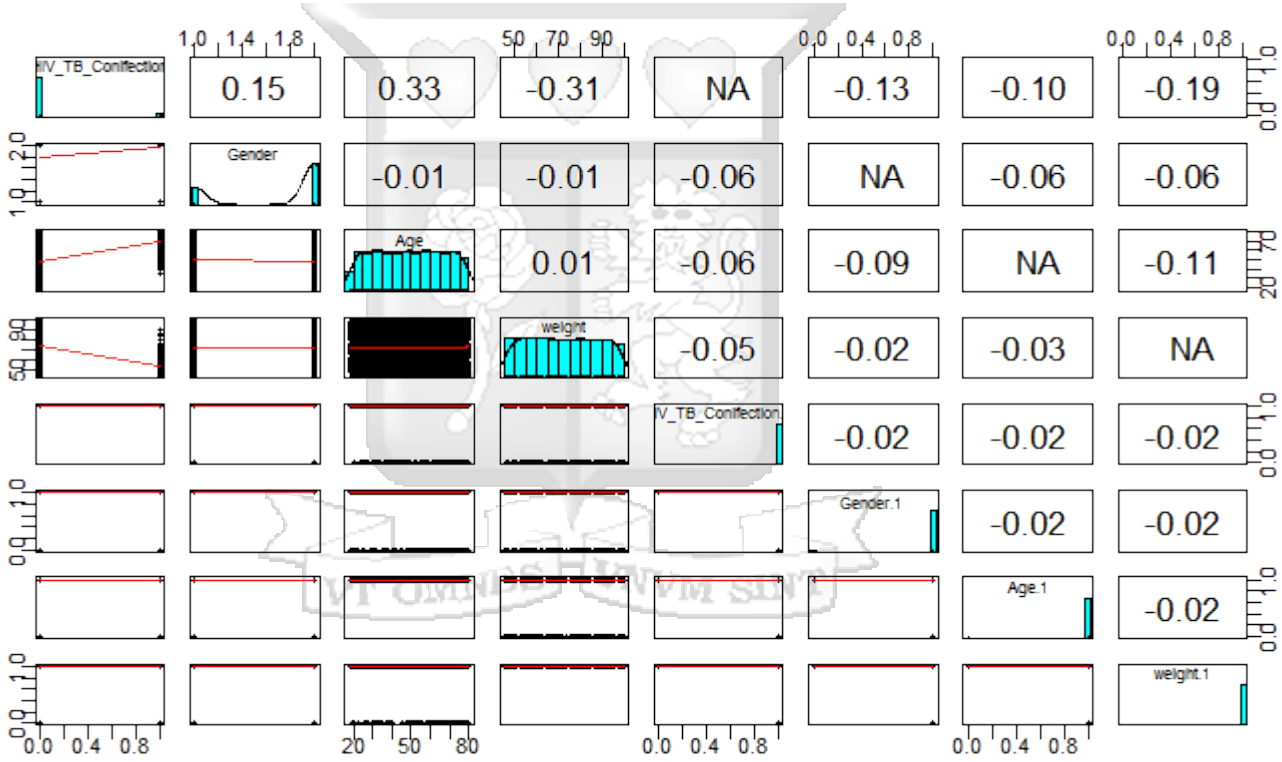


Figure 2.1: Pairs panels

2.0.3 Methods to correct for Missing data.

Due to the serious consequences of incomplete data (Kang, 2013), researchers have come up with ways to solve the issues of having missing data. García-Laencina et al. (2010) classified approaches of dealing with missing data into four groups: Case deletion, Imputation, Model-based procedures, and machine learning methods for handling missing data.

Case Deletion

The most common method which is always the default way in analysis softwares is available-case analysis, this can be in form of: Listwise deletion or pairwise deletion.

Listwise deletion.

This is the easiest method, units with unavailable information on any variable are left out during analysis. However, it greatly reduces the sample size and assumes that the information is missing completely at random.

Pair-wise deletion.

This is an improvement of list-wise deletion. This technique works with pairs of variables in that: it calculates the covariance estimates for records with completely observed data on both variables, i.e records are left out if they contain incomplete data on any of the variables being analyzed(Carter, 2006).

Pairwise deletion can be problematic as it results into varying sample sizes for different variable combinations within the same data set.

Case deletion approaches have the advantage of being simple and readily available as default settings in most statistical softwares. Their main disadvantage is the loss of information due to the reduced sample size.Van Buuren (2012) indicated that case deletion approaches do not always work. The American Psychological Association(APA) task force on statistical inference described case deletion approaches as being among the worst methods for practical applications (Wilkinson, 1999).

Imputation

Imputation is the estimation and filling in of the possible missing values. The imputation process can be one value for each missing value(single imputation) or more than one value for each missing value(multiple imputation). The four main imputation methods are:Mean imputation, Regression imputation, Hot and cold deck imputation, and Multiple imputation.

Mean imputation.

Here, missing values on a variable are filled in by the average(or mode for categorical variables) value of the observed cases in the variable.

Given that x_i^j is the j^{th} missing attribute of the i^{th} instance, the mean imputation is given by:

$$x_i^j = \sum_{k \epsilon I(complete)} \frac{x_k^j}{n_{I(complete)}} \tag{2.4}$$

Where I(complete) is a set of indices that are complete in x_i and $n_{I(complete)}$ is the total number of instances where the j^{th} attribute is observed (Sim et al., 2015).

H ¡retrun¿ Mean imputation has the advantage of improving on the pitfalls of case deletion methods and being simple to execute. However, it has the disadvantages of ignoring the variability and correlation in the data (Schafer, 1997) and assuming that missing data is MCAR which is rare in practice (Rubin, 2004).

Regression imputation.

In regression imputation, the missing values are predicted from the observed values. Observed values are used in the estimation of a regression equation whereby the incomplete variable is the dependent variable and the complete variable(s) the independent variable.

Suppose the i^{th} variable has missing values, and the remaining p-1 variables are complete. The regression model $f(.)$ is trained to approximate the unknown function using the available data.

$$\widetilde{x_i} = f(X_{obs}) \approx x_i \tag{2.5}$$

Where X_{obs} is the input vector made up of the complete variables(p-1) which is used to compute the unknown parameters that defines f(.). If there are more than one incomplete variable, a multivariate regression model is used to perform imputation (García-Laencina et al., 2010).

Regression imputation has a disadvantage of producing biased parameter estimates because the method assumes that imputed values fall on the regression line with a nonzero slope as would be shown on a scatter plot of a regression imputation. This implies a correlation of 1 between the predictors and the missing outcome variable, it also shows lack of variability in the imputed data as would been the case in the hypothetically complete data set. The imputed values which are perfectly correlated can overestimate the overall correlation but underestimates the variances and covariances.However, regression imputation has the advantage of producing unbiased estimates of the mean when the data are MCAR or MAR (Baraldi and Enders, 2010). Some researchers have attempted to remedy the disadvantage of regression imputation by use of a modified version called stochastic regression imputation.

In stochastic regression, after computing the predicted values, a random error term(a random number generated from a normal distribution with zero mean and a variance equal to the residual variance from the preceding regression analysis) is added to each predicted score and the result is used to replace missing values. The addition of the random error terms restores lost variability to the data. This modification produces parameter estimates that are unbiased under both the MCAR and MAR mechanisms.

Stochastic regression has the advantage of producing comparable estimates to maximum likelihood and multiple imputation. Stochastic regression suffers the disadvantage of having inappropriately small which would make the significance tests have excess Type I error rates(Baraldi and Enders, 2010).

Hot and cold deck imputation.

Andridge and Little (2010) described Hot Hot deck imputation as the replacement of missing values of one or more variables for a non-respondent(recipient) with observed values from a respondent(donor)that has similar observed characteristics. The donor may be selected randomly from a set of potential donors(donor pool); which is called random hot deck methods. The other hot deck procedure of selecting a donor is to identify a single donor (in most cases the “nearest neighbour” based on some metric) and values are imputed from that case,this method is called deterministic hot deck method because it lacks randomness in donor selection. Another form of hot deck imputation is the ”last observation carried forward” (LOCF). LOCF involves creating an ordered dataset by sorting according to any of the variables. Then find the first missing value and use the cell value immediately prior to the data that are missing to impute the missing value, this process is repeated for the next cell with a missing value until all missing values have been imputed. In longitudinal studies, the method replaces each missing value with the last observed value from the same subject (Hamer and Simpson, 2009). It represents the assumption that if a measurement is missing, it hasn't changed from the last time it was measured. This method increases risk of bias and false conclusions hence not recommended for use (Molnar et al., 2008).

The hot deck imputation has the advantage of not relying on model fitting thus less sensitive to model misspecification. The method also reduces non-response bias because there is an association between the variables defining imputation classes and both the tendency to respond and the variable to be imputed.

Cold-deck imputation, selects donors from a different dataset. It replaces missing values with response values of similar items in past surveys. It is common in surveys that measure time intervals and has same advantages as hot deck imputation.

Despite the methods being easy and simple to execute, the National Research Council's Panel on Missing Data in Clinical Trials discouraged the uncritical use of Single imputation methods as the primary approach to the treatment of missing data unless the assumptions that underlie them are scientifically justified (Council et al., 2010).

Multiple imputation(MI).
Chinomona and Mwambi (2015), described MI as a technique based on simulations that replaces unavailable values with some plausible values. Standard statistical analysis procedures are applied to each of complete data-set then estimates of parameter and their standard errors are obtained. Outcomes are merged to generate overal estimates(multiple imputation estimates), the confidence intervals cover the uncertainty of unobserved data.

White et al. (2011) outlined the three steps of Multiple imputation :
The first step is the creation of plausible values.
MI creates $p > 1$ probable values to fill in unavailable values. MI procedure draws the plausible values from a distribution specifically modeled for a missing value.For a given incomplete variable v, an imputation model is constructed that regresses v on variables with complete data,e.g $v_1, v_2, \ldots, v_k$, among individuals with the observed v. This results into m imputed complete datasets, the datasets are identical in observed values but differ in imputed values.
The second step involves parameter estimation. In this step, the standard analytical procedures are applied on to the imputed datasets as would ha ve been on a complete data set. Variations are expected in the results due to the different imputed values.

The final step is the pooling of results obtained in the second step. Rubin's rules (Rubin, 2004) are used to combine the m parameter estimates into one estimate and one variance–covariance matrix. The resultant variance–covariance matrix incorporates both within(sampling variance) variance and between-imputation(caused by missing data) variance. Mathematically:
Suppose $\widehat{\phi_i}$ estimates the multivariate/univariate quantity the researcher is interested in, for example regression coefficient resulting fromi^{th} imputed data and W_i the variance approximated of $\widehat{\phi_i}$. The pooled estimate $\widehat{\phi}$ is given by:

$$\widehat{\phi} = \frac{1}{n} \sum \sum_{i=1}^{n} \widehat{\phi_j} \tag{2.6}$$

equation image (1) Within-imputation variance of $\widehat{\phi}$ is:

$$w = \frac{1}{n} \sum_{i=1}^{n} w_i \tag{2.7}$$

and the variation between imputations is combined as:

$$b = \frac{1}{n-1} \sum_{i=1}^{n} (\widehat{\phi_j} - \widehat{\phi})^2 \tag{2.8}$$

and the total variance of $\widehat{\phi}$ is given by combining variance between imputations and the variance within imputations:

$$Var(\widehat{\phi}) = w + (1 + \frac{1}{n})b \tag{2.9}$$

Figure 2.2 illustrates multiple imputation steps for three(m) imputed data sets. The three data sets are kept in an object belonging to class MIDS. A function with() is used to analyze imputed datasets and analysis results are stored in class MIRA. Finally Rubin's rules are applied to pool the results of imputed datasets. The last procedure is done using mice's pool() function, pooled results are stored in an object contained in class MIPO (Buuren and Groothuis-Oudshoorn, 2010).

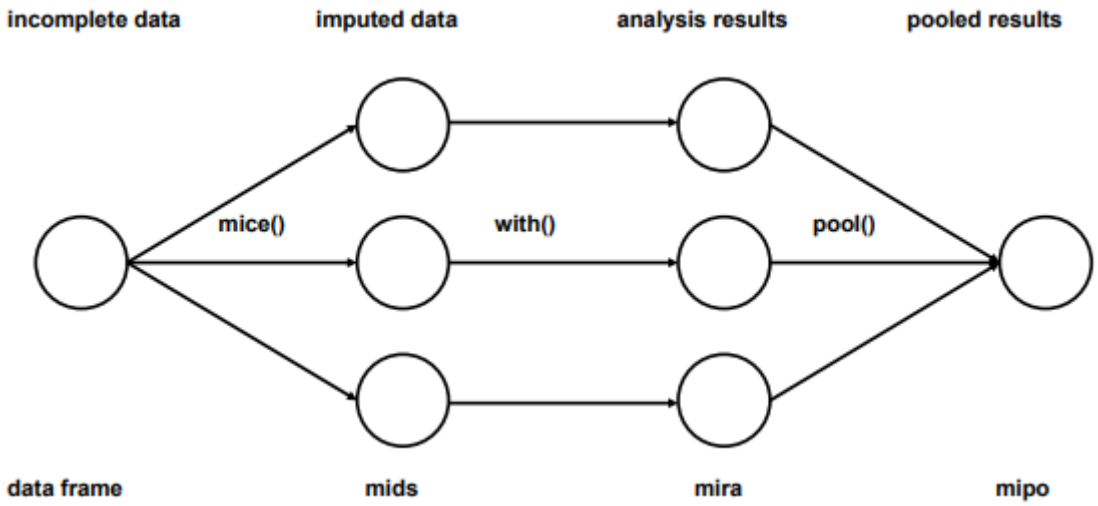


Figure 2.2: Multiple imputation steps

Imputation methods based on machine learning

These methods model the missing data estimation based on information available in the dataset.
Imputation with K-nearest neighbor (K -nn).
In this method, K nearest neighbors (donors) are obtained from the complete cases. The nearest and most similar neighbors are selected by minimizing a distance function (Batista and Monard, 2003). If we have an incomplete pattern x, the set of K nearest neighbors (based on a distance metric) organized in increasing order of its distance are obtained. After obtaining these neighbors, a replacement value to substitute the missing value is estimated. The replacement depends on the type of data; mode for discrete data and mean for continous data (García-Laencina et al., 2010). An improvement of the K-nn method is to weight the contribution of each of the K neighbors. A key aspect of the K-nn method is the distance measure. The accurate distance function is the heterogeneous euclidean overlap metric (HEOM)(Batista and Monard, 2003; Batista et al., 2002).

Self-organizing map (SOM)imputation.
This method describes a mapping from a higher dimensional (dimension d) input data space to a lower dimensional (dimension d L) map space, being d L = 2 the most extended approach. The SOM entails nodes placed in a d L-dimensional array with each node having and associated d-dimensional weight vector. The assigned weights(to each node) are representative of the input data, in such a way that nodes that are spatially close in the array have similar weight vectors (Kohonen, 2012). The neuron with weight vector most similar to the training input vector x is called the best matching unit (BMU) or image node. The weights of the BMU and its neighbor nodes close to x in the SOM lattice are adjusted towards the input vector. A neighborhood function(Gaussian is the common choice) is defined. In general, a SOM can be

considered a non-linear version of PCA .
Self-organizing map (SOM)imputation is conducted in three steps. First, the incomplete pattern is presented to the SOM, its image-node is chosen ignoring the distances in the missing variables; the second step entails selecting an activation group composed of image-node's neighbors; in the last step, each imputed value is computed depending on the weights of the activation group of nodes in the missing dimensions (Fessant and Midenet, 2002). Piela (2002) implemented missing data imputation in a tree structured self-organizing map(TS-SOM), this is composed of several SOMs arranged to a tree structure. The major advantages of TS-SOM over the basic SOM are is the faster convergence and its computational benefit when the number of input vector is large (García-Laencina et al., 2010).

Multi-layer perceptron(MLP) imputation.
The approach entails training an MLP by use of the complete cases only as regression model. Given p input variables, each incomplete variable is learned (used as output) using the remaining complete attributes given as inputs. The MLP imputation scheme is described in two steps.
The first step is to separate the incomplete dataset X, into the observed component(X_o;input vectors not containing any missing data) and the vector with missing values(X_m). In the second step,

for each possible combination of incomplete variables in X_m, an MLP is constructed using X_o. The target variables are those with missing data, and the input variables are the ones that are completely observed (Sharpe and Solly, 1995). There is one MLP model per missing variables combination. Depending on the type of the variables to be imputed (continuous or discrete), different error functions (sum of squares error or cross-entropy error) are minimized during the training process.

The MLP approach has the main disadvantage that when missing items appear in several variables, several MLP models have to be designed, one per missing variables combination. Other MLP methods have been proposed to solve the disadvantage (Yoon and Lee, 1999; Kallin, 2002). Yoon and Lee (1999) propose the Training-Estimation-Training(TEST) algorithm as a way of using MLP to impute missing data. TEST is made up of three steps. Step 1, the network is trained with all the complete patterns. Step 2, the parameters (weights) are used to estimate the missing data in all incomplete patterns by use of back-propagation in the missing variables. Step 3, the MLP network is trained again using the whole data set(complete and imputed patterns). The disadvantage of TEST is that it cannot estimate missing values in the test set. To solve that, Kallin (2002) implements an imputation procedure in a single layer perceptron (SLP). First, the SLP is trained using only the complete cases; then imputation is done with the inverse of the obtained equation for the desired classification output (it requires that the pre-processing and activation functions can be inverted). This solution produces good results which are comparable to multiple imputation.

Recurrent neural network(RNN) imputation.
In RNN, missing values are imputed using the feedback connections from the hidden neurons. Missing values are initialized with the mean imputation, and updated using the feedback connections as the network is trained to learn the classification task. Missing values are modified as a function of the missing input in the last iteration and the weighted sum of a set of recurrent links from the other units (hidden and missing) to the missing unit with a unit delay. When the input variables depend on each other, the output prediction is improved by accounting for the dependencies. RNN performs better than standard network with missing values replaced by their mean (Bengio and Gingras, 1996).

Auto-associative neural network(AANN) imputation.
In AANN, missing data imputation is done by use of the output unit that learns the corresponding incomplete attribute.

An auto-associative neural network (AANN) is a set of neurons that are completely connected such that each neuron receives input from, and sends output to, all the other neurons. Missing data imputation using this kind of networks is executed using the following procedure: the network learns from complete cases, so as to replicate all of the inputs as outputs. When unknown values are detected, the weights are not updated. Instead, the missing values are replaced by the network outputs (Narayanan et al., 2002; Chung and Merat, 1996; Marseguerra and Zoia, 2005). We propose further reading from (García-Laencina et al., 2007) so as to understand Multi-task learning(MTL) approaches.

Model-based procedures.
Maximum likelihood methods
In maximum likelihood method, it is assumed that the observed data are drawn from a multivariate normal distribution. Parameters are estimated using the available data and the missing data are estimated based on the estimated parameters. the missing data may be estimated by using the conditional distribution of the other variables (Kang, 2013). The statistics explaining the relationship between complete and incomplete variables is computed by maximum likelihood method. We reviewed two maximum likelihood methods:Full information maximum-likelihood (FIML) and Expectation-Maximization (EM).

Full information maximum-likelihood (FIML)
FIML is used frequently in structural equating modeling (SEM). (Dong and Peng, 2013) stated that in the literature that they reviewed, 26.1% studies used FIML to deal with missing data. FIML does not impute any missing data, but estimates parameters using the information contained in the incomplete data set (Hartley and Hocking, 1971). FIML obtains parameter estimates by maximizing the likelihood function of the incomplete data. Assuming multivariate normality, the log likelihood function of each observation i is:

$$logL_i = K_i - \frac{1}{2}log|\sum| - \frac{1}{2}(x_i - \mu)' \sum^{-1} (x_i - \mu) \tag{2.10}$$

x_i is the vector of observed values for case i, K_i is a constant determined by the number of observed variables for case i,μ is the mean vector to be estimated, and $\sum$ is the covariance matrix to be estimated too.
Overall sample log likelihood is the sum of the individual log likelihood across n cases. The standard ML algorithm is used to estimate μ and $\sum$. The corresponding standard errors are estimated by maximizing the total sample log likelihood function. FIML assumes MAR and multivariate normality for the joint distribution of all the variables making it produce unbiased estimates (Enders and Bandalos, 2001).

Expectation-Maximization (EM)
Expectation-Maximization is a general iterative computation of maximum likelihood estimates when observations are incomplete.The process is called Expectation-Maximization because each iteration of the algorithm is made up of an expectation step followed by a maximization step.It creates a new data set where all missing values are imputed with values estimated by the maximum likelihood methods (Dempster et al., 1977). The process begins with the expectation step in which the parameters are estimated. The estimated parameters are then used to create a regression equation which predicts the missing values. The next step is the maximization step which uses equations from the first step to fill in the missing data. The two steps are repeated until stability is achieved (the covariance matrix for the subsequent iteration become the same

as that for the preceding iteration).

The disadvantages expectation-maximization are: when the proportion of missingness is large, the process can take a very long time to reach convergence. The process is too complex to be acceptable by some statisticians. This approach can lead to the biased parameter estimates and can underestimate the standard error (Kang, 2013; Dong and Peng, 2013).

Machine learning methods for handling missing data

Neural network ensembles.
In this method, a set of multilayer perceptron (MLPs) is created, and each MLP classifies based on each possible combination of complete features. The main disadvantage of this method is that it requires a large number of neurons if multiple combinations of incomplete attributes are presented. The incomplete dataset is divided into two groups, one containing the complete data sets and the other group is the incomplete data set. The complete data set is used as the training sets for the neural networks. The neural network ensembles method can utilize all information from the data thus maintaining maximum consistency and avoiding the dependency on distribution assumption (Jiang et al., 2005).

Decision trees.
This method employs three approaches: ID3, C4.5, and CN2. ID3 is a top–down decision tree algorithm that handles an unknown observation by generating an additional edge for the unknown. The unknown edge is taken as the new possible value for each attribute and it is treated in the same way as other values. C4.5 is an extension of ID3 proposed that employs a probabilistic approach to handle missing values in the training and test data set. Here, during training, each value of an attribute is assigned a weight of 1 if the attribute value is known otherwise the weight of any other value for that attribute is the relative frequency of that attribute. During the testing phase, if a test case is incomplete, all the available branches are explored and a decision is made on the class label depending on the most probabilistic value. The CN2 is an algorithm that induces propositional classification rules (Quinlan, 1993, 1989; Webb, 1998; Zheng and Low, 1999; Clark and Niblett, 1989).

Fuzzy approaches.
Fuzzy approaches that handle missing data include the MLP classification in which missing values are represented by interval inputs, the interval inputs includes all possible values of the attribute. Observed values are also represented by intervals and the network is trained by means of back-propagation algorithm for fuzzy input vectors (Ishibuchi and Tanaka, 1991).
Another fuzzy approach called the general fuzzy min–max (GFMM) neural network using hyperbox fuzzy sets was developed by Gabrys (Petit-renaud and Denoeux, 1998; Gabrys, 2000). The min-point and the max point of the d-dimensional space is defined by a hyperbox.The creation and adjustment of the hyperboxes happens during learning in the GFMM neural network for classification (Gabrys, 2000). Missing values are handled by modelling the missing attribute by use of a real valued interval over the whole range of values.

Other techniques that tolerate missing values are based on a system of fuzzy rules for classification (Berthold and Huber, 1998; Nauck and Kruse, 1999). A fuzzy rule based classifier has a set of rules for each possible category and each rule can be decomposed into one-dimensional membership functions that correspond to the fuzzy sets. Other fuzzy approaches for handling missing values are based on fuzzy C-means (Hathaway and Bezdek, 2001; Ichihashi and Honda, 2005; Sarkar and Leong, 2001).

Support vector machines(SVM)
In this method the standard Support vector machines classifier with all the training data

completely observed is extended to handle missing values. This method generalizes the linear mean imputation and the resulting kernel machine reduces to the standard Support Vector Machine (SVM) when all input are not missing. The linear classification constraints are replaced by probabilistic constraints. Missing variables are modelled as random variables and model parameters estimated by EM algorithm. (Smola et al., 2005; Pelckmans et al., 2005; TNLJWD et al., 2004).

2.0.4 Statistical Learning.

James et al. (2013) defined statistical learning as a wide range of instruments for interpreting data. Statistical learning is categorized into Supervised and Unsupervised.
Supervised refers to the case whereby a statistical model is built to estimate/predict an outcome based on one/more predictors .
Unsupervised statistical learning refers to a statistical learning method in which we have input(s) but no supervising output, the main aim here is to understand data structure and relationships James et al. (2013).
Let Y be a numeric outcome with m predictors: $X_1, X_2, ..., X_m$. Assuming that Y is associated with $X = X_1, X_2, ..., X_m$. Equation 2.8 represents interrelation between X and Y:

$$Y = f(X) + \in \tag{2.11}$$

Here f is a fixed unknown mathematical relation of $X_1, X_2, ..., X_m$ representing systematic information that X yields about Y and $\in$ is an error term which does not depend on X and has a mean of zero. **Statisical learning** is the collection of approaches for estimating f.
We estimate f for prediction and inference. Prediction is stimulated by X being readily available as opposed to Y. With reference to equation 2.8 above, given that $\in$ has a mean of zero, we predict Y using equation 2.9

$$\widehat{Y} = \widehat{f}(X) \tag{2.12}$$

Where $\widehat{f}$ and $\widehat{Y}$ are the estimates of f and Y respectively. We are not interested in the precise form of $\widehat{f}$, as long as resultant estimate accurately predicts Y hence we treat $\widehat{f}$ as a black box. The **reducible** and **irreducible** errors determine accuracy level of $\widehat{Y}$ predicting Y.
Generally, f cannot be perfectly estimated by $\widehat{f}$, we attribute this to an error that is reducible. The error is called reducible error because we can improve $\widehat{f}$'s prediction accuracy by applying best($\widehat{Y} = f(X)$) statistical learning technique for estimating f. The prediction would still contain an error because Y is a function of $\in$(from the definition cannot be predicted by X) , this is irreducible James et al. (2013). We can't control the error introduced by $\in$.
Under inference, of interest is understanding how Y is altered by changes in X. Estimation of f is done but main focus is not predicting Y, the focus is understanding the relationship between X and Y, that is; how Y varies in relation to X. The exact form of $\widehat{f}$ needs to be established. Consequently, we don't treat $\widehat{f}$ as a black box.

Unsupervised & Supervised learning.
If there exists an associated response measurement y_i for every occurrence of explanatory items(s)$x_i, i = 1, 2, ...n$, then the statistical method is called supervised learning. A model is created for the relationship between the response and predictors. Goal is to make accurate predictions and inferences. Models that have a numeric outcome are called **regression models**, but the models with a qualitative response are called **classification models**.

If there exists a vector of measurements x_i for every observation $i = 1, 2, ..., n$ but lack response y_i then we refer to this case as unsupervised learning. In this scenario cluster analysis

is used to find out, based on $x_1, x_2, ..., x_n$, whether observed values are in easily distinguishable groups.
In statistical learning methods predictor variable type is less important as long as proper coding of the categorical predictor(s) is done before analysis.

2.0.5 HIV/TB Coinfection

Tuberculosis is leading opportunistic infection that cause morbidity & mortality among PLWHIV. TB is a global health problem with yearly new infections of 9 million and around 2 million annual deaths.The most affected countries are in Africa which account for 85% of the global rates of infection. Kenya is among the twenty two high burden countries accounting for 80% of global TB infections (Organization et al., 2010). Approximately thirty percent of PLWHIV are estimated to be infected with TB (Getahun et al., 2010) .
Globally, thirteen million persons have HIV and TB co-infection, 70% of worldwide HIV-TB co-infections are African residents. (Papathakis and Piwoz, 2008).

HIV highly interact with Mycobacterium tuberculosis each increasing progression of the other.Treatment of TB is made difficult due to interactions of the drugs with HAART resulting into adverse drug reactions.
For PLWHIV, active TB is an indication of AIDS. Corbett et al. (2003) stated that both active TB and HIV accelerate the progression of the other with the former decreasing the number of CD4+ lymphocytes thereby increasing the reproduction of HIV that eventually shortens the lifespan of PLWHIV.The fatality rate of HIV-related TB is over 50% .Further information on HIV and/or TB can be found at Organization (2013).

In Kenya, the NTLD-P (2018) report indicates that most(98%) of the TB patients were screened for HIV, **27%** of the patients infected with TB were also living with HIV.This co-infection rate was close to the 28% co-infection rate reported in 2017 (NTLD-P, 2017).
The prevalence rates for the two years are questionable due to what was reported on strategies for finding missing people with TB. The key strategy was the Active Case Finding (ACF) which entails a systematic screening for TB among all patients presenting to health facilities regardless of whether they present with TB symptoms or not.The key motivation to our study comes from one of the challenges encountered during the ACF strategy implementation the challenge was called "System challenges" and quoted as: **"incomplete documentation in the presumptive TB registers leading to leakage e.g. missing lab results was also noted during the implementation"**.

Chapter 3

Methodology

3.0.1 Statistical Learning Algorithm

Let ψ be an observed response which is quantitative and n different predictors, $X_1, X_2, \cdots, X_\eta$. Assuming there is a relationship between ψ with $X = (X_1, X_2, \cdots, X_n)$, one can write the relationship as:

$$\psi = f(X) + \epsilon. \tag{3.1}$$

Where f is an unknown function of $X = (X_1, X_2, \cdots, X_n)$, ϵ is the error term, that is not dependent on X with mean value zero. In this formulation, f is the structured details that X gives concerning ψ.
Assuming that there is a true underlying parameter vector $\Omega \in \mathbb{R}^d$ which governs the outputs (Vapnik, 2013). For each $i = 1, 2, \cdots, n$:

$$\psi_i = X_i\Omega + \epsilon_i$$

The problem of statistical learning involves a set of approaches for approximating f and evaluation of obtained estimates. The motives of estimating f are prediction as well as inference. For prediction, the inputs (X) are readily available, but outputs ψ can't be obtained with ease. The error term has a mean of zero, therefore we predict ψ by:

$$\hat{\psi} = \hat{f}(X), \tag{3.2}$$

where $\hat{f}$ is f's estimate and $\hat{\psi}$ the resultant prediction of ψ . Here, $\hat{f}$ is handled like a black box, that is, we are concerned with $\hat{f}$'s exact form provided it gives accurate predictions for ψ. At training segment, we observe one realization of $\psi_1, \psi_2, \cdots, \psi_n$. The data matrices are given as:

$$X = [X_1, X_1, \cdots, X_n]^I \in \mathbb{R}^{n\times n}.$$
$$\epsilon = [\epsilon_1, \epsilon_2, \cdots, \epsilon_n]^I \in \mathbb{R}^n.$$
$$\psi = [\psi_1, \psi_2, \cdots, \psi_n]^I \in \mathbb{R}^n.$$
$$\Sigma = \frac{1}{n}X^I X \in \mathbb{R}^{n\times n}.$$

The goal is to naturally minimize the expected risk. See (Vapnik, 2013; Xuegong, 2000) for more coverage on ML

3.0.2 Handling Missing Data

Here we employ the complete case scenario and compare with three methods i.e. Weighted Method, Maximum likelihood approach and MI.

Complete Case

By complete cases, we refer to analyzing available data in generating parameters of interest. This approach is illustrated in equation 2 where required statistics is generated on different sets of cases.
For pairwise deletion, all cases are used to estimate mean of ϕ however, only the complete cases are used to estimate: Φ and correlation of ϕ and Φ. Distinct collections of cases are used in estimation of parameters that are of in the data. In situations where variables happen to be strongly associated, analysis of present cases gives estimations not superior to estimates of fully observed cases (Haitovsky, 1968).
Let ψ_i be a given i^{th} observation. Then

$$\Psi_{m,n} = \begin{pmatrix} \varphi_{1,1} & \varphi_{1,2} & \cdots & \varphi_{1,n} \\ \varphi_{2,1} & \varphi_{2,2} & \cdots & \varphi_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ \varphi_{m,1} & \varphi_{m,2} & \cdots & \varphi_{m,n} \end{pmatrix} \tag{3.3}$$

$$\Psi_{p,q} = \begin{pmatrix} \varphi_{m1,1} & \varphi_{m1,2} & \cdots & \varphi_{1,1n} \\ \varphi_{m2,1} & \varphi_{m2,2} & \cdots & \varphi_{2,2n} \\ \vdots & \vdots & \ddots & \vdots \\ \varphi_{mp,1} & \varphi_{mp,2} & \cdots & \varphi_{mp,nq} \end{pmatrix} \tag{3.4}$$

Where $\Psi_{m,n}$ are complete cases and $\Psi_{p,q}$ refers to incomplete cases. The estimates for the mean, standard error can be obtained as:

$$\bar{\varphi_1} = \frac{\sum_n^i \varphi_{1i}}{n} \tag{3.5}$$

$$s.\bar{e}(\varphi) = \frac{\sum_n^i (\varphi_{1i} - \bar{\varphi_1})^2}{n} \tag{3.6}$$

Weighted Method

In this approach, unobserved value(s) is replaced with some plausible value, like mean of observed cases.This approach provides for incorporation of all observations during analysis. Nonetheless, replacement of unavailable data with mean/mode changes distribution of a given variable; it decreases the variance.
Impute all missing values of ψ by (weighted) respondent mean of $\bar{\psi}$, if ψ continuous. Impute by respondent mode if categorical variable. Define homogeneous classes and impute mean or mode within class ψ mean (or mode) same for respondents and missing values within classes. We may use segmentation algorithm to develop homogenous classes.

Maximum Likelihood Method

The principle here is to base estimation of the missing value on the likelihood of available data. The problem with this method is the difficulty of specifying the likelihood. To solve the difficulty, the EM algorithm is used to estimate parameters (like means and covariance matrix).
The intention is maximization of conditional likelihood by use of a set of respondents with $\varphi_{iT} = 1$, their response possibility is changed in the sense that, rather than original probability, the conditional likelihood of $\varphi_{it-1} = 1$ given $\varphi_{iT} = 1$ is considered. Conditional chance for the special circumstance when T = 1, implying no followup.
Conditional likelihood approach involves two steps. In the initial step, reverse conditional probability $q_{it} = pr(\varphi_{i,t} = 1 | \varphi_{i,t+1} = 1, \psi_i, \upsilon_i)$ derived by employing Baye's formula from the

assumed response model.
We obtain $q_{it} = \frac{\Xi_{it}}{1+\Xi_{it}}$ where

$$\Xi_{it} = \frac{pr(\varphi_{it} = 1|\psi_i, \upsilon_i, \varphi_{i,t+1} = 1)}{pr(\varphi_{it} = 0|\psi_i, \upsilon_i, \varphi_{i,t+1} = 1)}$$

$$= \frac{pr(\varphi_{it} = 1, \varphi_{i,t+1} = 1|\psi_i, \upsilon_i)}{pr(\varphi_{it} = 0, \varphi_{i,t+1} = 1|\psi_i, \upsilon_i)}$$

$$= \frac{pr(\varphi_{it} = 1|\psi_i, \upsilon_i, \varphi_{i,t+1} = 1)}{pr(\varphi_{it} = 1|\psi_i, \upsilon_i, \varphi_{i,t+1} = 0)} \frac{pr(\varphi_{it} = 1|\psi_i, \upsilon_i)}{pr(\varphi_{it} = 1|\psi_i, \upsilon_i)}$$

$$= \frac{1}{p_{i,t+1}} \frac{\bar{\Upsilon}_{it}}{1 - \bar{\Upsilon}_{it}}$$

where $\bar{\Upsilon}_{it} = \sum_t^{j=1}[p_{ij} \prod_{j-}^{k=1}(1 - p_{ij})] = \sum_t^{j=1} \upsilon_j$. More of the parameter estimation can be found in Im (2015).

Multiple Imputation

Generally, there are three steps for executing MI. The first step is to create m ($m > 1$) plausible values for each missing value. The second step is to analyze the fully observed datasets. Final step is combination of results from m analyses. Step 1 depends on assumptions of missingness principle which created the observed sample. Imputation aims at accounting for the association of unobserved and observed values, at the same time considering uncertainty of imputation.The Missing At Random assumption that is generally assumed for many missing data methods, is important to the validity of MI method. For monotone missingness, easy methods(propensity methods,predictive mean matching,discriminant analysis or logistic regression) are employed. However, for complicated missingness, Markov Chain Monte Carlo (MCMC) methods are applied (Horton and Lipsitz, 2001). All these methods require multivariate normality assumption.

Simulation.

We adopted the incomplete data methodology suggested by Schouten et al. (2018), it comprises of four steps. Step 1. Simulation of a multivariate, complete data set to be considered the population of interest.Step 2. Making the dataset incomplete.Step 3.Estimating the incomplete data by methods of correcting for missing data.Step 4.Comparing the Statistical inferences obtained for the original, complete data set and after dealing with the missing values to get an indication of the performance of the missing data method. We then apply real HIV-TB coinfection data.

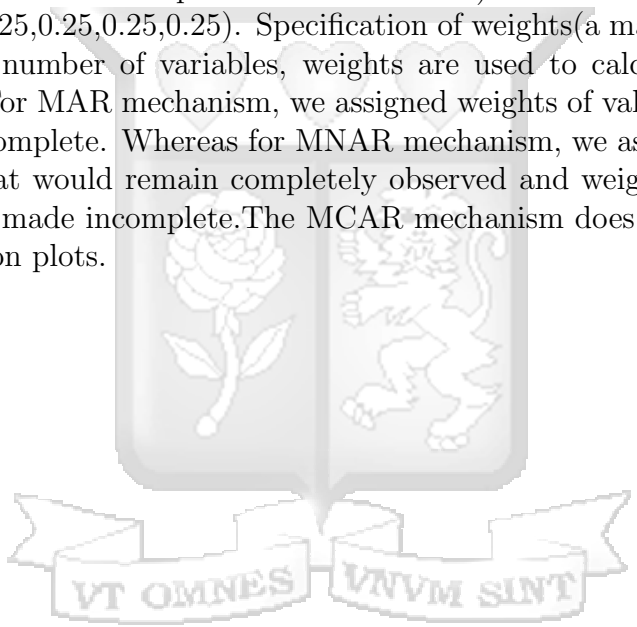
3.0.3 Step 1: Simulated multivariate complete Data set

We used R to simulate a data set that mimics real HIV/TB coinfection data set. The simulated data set contained 10000 observations with four variables :"Gender";categories "1: Male" and "2: Female" , "HIVTBConifection"; levels "1:coinfected" and "0:not coinfected", "weight"; patient's weight and "Age"; patient's age.
Our aim is to predict patients who would be coinfected based on Gender, Age, and weight. The statistical inferences of interest are the Accuracy, Sensitivity, Specificity and AUC of the original complete dataset that we would compare with the statistical inferences of the original dataset after creating missingness(under MAR, MNAR and MCAR) and correcting for the missingness by: available case analysis, replacement with mean or mode, multiple imputation and maximum likelihood methods.

3.0.4 Step 2: Missing Data Amputation.

The amputation(generation of missing values) was done using the "ampute" function which is available in mice package of R software, a multivariate amputation procedure as explained by Schouten et al. (2018) was used. Missing data were generated in three mechanisms: MCAR, MAR and MNAR with varying proportion of missingness (7%, 10%, 30%, 50% and 80%).
The choice of the missingness proportions was arbitrary because from literature, there is no established cutoff of acceptable or unacceptable proportion of missing data (Dong and Peng, 2013). Schafer (1999) stated that when the proportion of missing data is less than 5% then single imputation is sufficient enough to make accurate estimates. Bennett (2001) mentioned that statistical analyses are likely to be biased if the proportion of missing data is over 10%. However, the proportion is not the only criterion for assesing the missind data problem, the mechanism and pattern have more impact on the results than the proportion (Tabachnick et al., 2007).

The default patterns(combinations of variables with;coded 0 and without;coded 1 missing values) in ampute function of R was adopted. Here the arbitrary missingness pattern was for records on a specific variable. Our frequency(a vector of length number of patterns containing the relative frequency with which the patterns should occur.) too was default (equal probability for each pattern) i.e c(0.25,0.25,0.25,0.25). Specification of weights(a matrix/data frame of size number of patterns by number of variables, weights are used to calculate weighted sum of scores) was as follows: For MAR mechanism, we assigned weights of value zero to the variables that would be made incomplete. Whereas for MNAR mechanism, we assigned weights of value zero to the variables that would remain completely observed and weights of value one to the variables that would be made incomplete.The MCAR mechanism does not use weights. Table 3.1 shows the amputation plots.



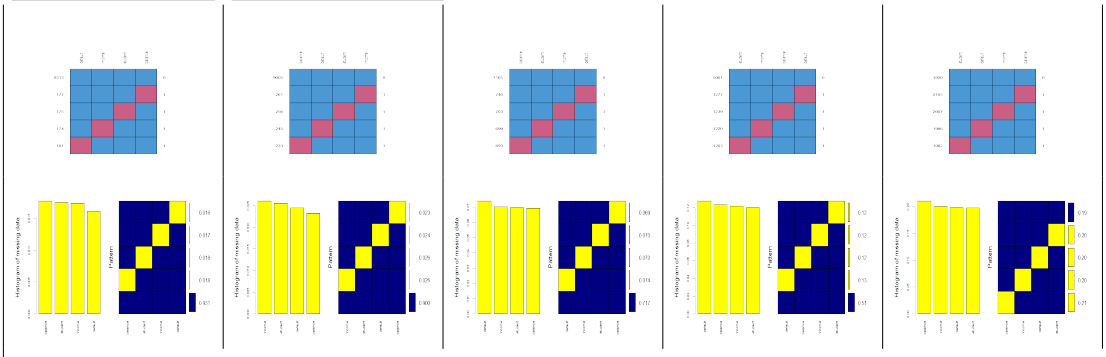
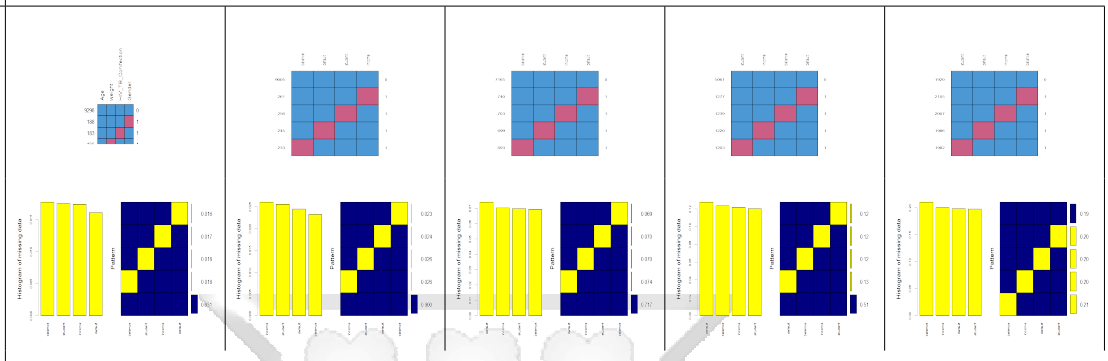
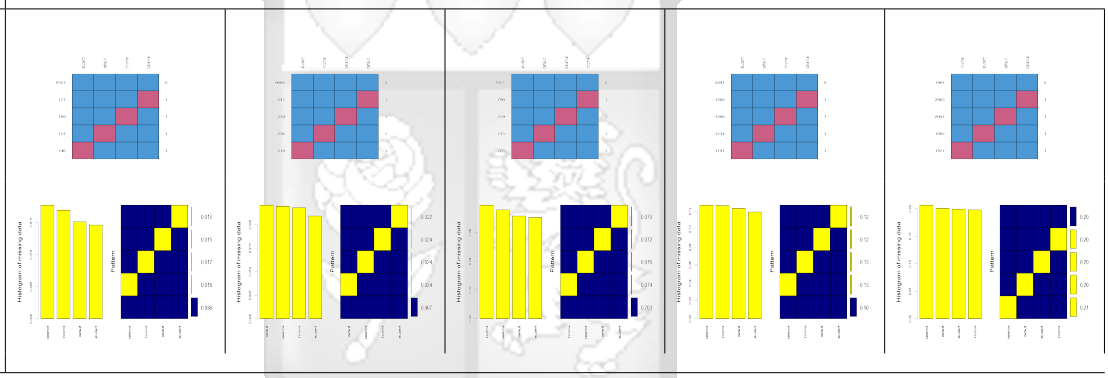
Missing Data Mechanism	7% Missingness	10% Missingness	30% Missingness	50% Missingness	80% Missingness
MNAR					
MAR					
MCAR					

Table 3.1: Graphical presentation of the amputations

3.0.5 Step 3: Correcting for the missing Data

We started the procedure of correcting for missing values by first reconverting the categorical which had been converted(during amputation) into numerical into their original categorical form. Checked to ensure the data was coded correctly, identified missing values and patterns within each variable and graphically represented the missingness. Finally, we corrected for the missingness in each of the amputed(under the three mechanisms)datasets using each of these methods: Complete case analysis, mode/mean replacement, MI and MLE.

3.0.6 Step 4: Comparing the Statistical inferences obtained for the original, complete data set and after correcting for the missing values

We repeated the analysis in step 1 on each of the datasets that we had corrected for missingness. The goal was to obtain similar statistical inferences(Accuracy, Sensitivity, Specificity and AUC) for comparison with those obtained in the first step.

Chapter 4

Results and Conclusion

4.0.1 Simulation Results

Prediction of "HIVTBConifection" using "Gender", "Age", and "weight" of the complete simulated data set resulted into 0.96, 0.56, 0.99 and 0.98 as the Accuracy, Sensitivity, Specificity and AUC of our prediction model respectively. These are the standard inferences that we used to compare to get an indication of the performance of the missing data method.
We generated missingness in the simulated data set. The proportion of missingness ranged from 7% to 80%. Then used four approaches of dealing with missing data: Complete case analysis (List wise deletion), Mean/Single imputation, Multiple imputation and Maximum Likelihood Estimation Imputation method. We re-analyzed the data sets to obtain new: Accuracy, Sensitivity, Specificity and the area under the receiver-operator curve (AUC) for comparison with the complete case values.
Table4.1 shows analysis where missing data are corrected for by List wise deletion; complete case analysis. Under MAR, We see similar (compared to Complete data) values of sensitivity at 7% which which increases at 10% and systematically decrease as the proportion of missingness increase from 30% to 80%. However, we observe similar AUC values across all the various proportions of missingness.
Similar inference values are observed for both MAR and MNAR, this is due to the impossibility of distinguishing between MAR and MNAR using observed data.
When the mechanism is MCAR, we observe irregular sensitivity values, this is caused by the complete randomness in the amputation of the missing values. The specificity and AUC values are similar to the complete data.

Listwise deletion: Complete Case Analysis(CCA)				
Missingness under MAR				
Percentage Missing	Accuracy	Sensitivity	Specificity	AUC
Original Complete Data	**0.96**	**0.56**	**0.99**	**0.98**
7%	0.96	0.56	0.99	0.98
10%	0.97	0.65	0.99	0.98
30%	0.97	0.56	0.99	0.98
50%	0.98	0.54	1.0	0.98
80%	0.98	0.45	1.0	0.99
Missingness under MNAR				
Original Complete Data	**0.96**	**0.56**	**0.99**	**0.98**
7%	0.96	0.56	0.99	0.98
10%	0.97	0.65	0.99	0.98
30%	0.97	0.56	0.99	0.98
50%	0.98	0.54	1.00	0.98
80%	0.98	0.39	1.00	0.99
Missingness under MCAR				
Original Complete Data	**0.96**	**0.56**	**0.99**	**0.98**
7%	0.96	0.70	0.98	0.98
10%	0.96	0.72	0.99	0.98
30%	0.96	0.73	0.98	0.98
50%	0.96	0.69	0.98	0.98
80%	0.96	0.70	0.98	0.97

Table 4.1: Complete Case Analysis

Table 4.2 shows analysis when missing data are corrected for by Mean/Mode single imputation. We replaced missing values with mean and mode of observed cases for continuous and categorical variables respectively. Across all missingness mechanisms, we observe a systematic decrease in the Accuracy, Specificity,and AUC values. We observe similar values of sensitivity across all the mechanisms, the sensitivity values are higher than those observed during complete data analysis.

Mean/Mode Single Imputation Method				
Missingness under MAR				
Percentage Missing	Accuracy	Sensitivity	Specificity	AUC
Original Complete Data	**0.96**	**0.56**	**0.99**	**0.98**
7%	0.94	0.99	0.48	0.90
10%	0.93	0.99	0.42	0.87
30%	0.87	0.99	0.19	0.77
50%	0.81	0.99	0.13	0.73
80%	0.76	0.97	0.17	0.70
Missingness under MNAR				
Original Complete Data	**0.96**	**0.56**	**0.99**	**0.98**
7%	0.94	0.99	0.48	0.90
10%	0.93	0.99	0.42	0.87
30%	0.87	0.99	0.19	0.73
50%	0.82	0.99	0.13	0.73
80%	0.76	0.97	0.17	0.70
Missingness under MCAR				
Original Complete Data	**0.96**	**0.56**	**0.99**	**0.98**
7%	0.95	0.99	0.55	0.90
10%	0.94	0.99	0.48	0.88
30%	0.88	0.99	0.20	0.75
50%	0.83	1.0	0.12	0.70
80%	0.77	0.99	0.12	0.68

Table 4.2: Mean/Mode Single Imputation Method

Table 4.3 shows analysis when missing data are corrected for by Multiple imputation. We used the MICE package in R to carry out multiple imputations. We imputed three data sets using all the variables ("Gender", "Age", "weight", "HIVTBConifection"). Results indicate similar characteristics among the three mechanisms: the values for Accuracy, Specificity and AUC are very close to the complete data set values. The Sensitivity values are similar across the mechanism but higher than the complete data set Sensitivity values.

Multiple Imputation Method				
Missingness under MAR				
Percentage Missing	Accuracy	Sensitivity	Specificity	AUC
Original Complete Data	**0.96**	**0.56**	**0.99**	**0.98**
7%	0.96	0.70	0.98	0.98
10%	0.96	0.70	0.98	0.98
30%	0.96	0.68	0.98	0.98
50%	0.96	0.72	0.99	0.98
80%	0.96	0.69	0.99	0.98
Missingness under MNAR				
Original Complete Data	**0.96**	**0.56**	**0.99**	**0.98**
7%	0.96	0.70	0.98	0.98
10%	0.96	0.70	0.98	0.98
30%	0.96	0.68	0.98	0.98
50%	0.96	0.71	0.99	0.98
80%	0.96	0.69	0.99	0.98
Missingness under MCAR				
Original Complete Data	**0.96**	**0.56**	**0.99**	**0.98**
7%	0.96	0.72	0.98	0.98
10%	0.96	0.71	0.99	0.98
30%	0.97	0.72	0.99	0.98
50%	0.96	0.70	0.98	0.98
80%	0.96	0.67	0.98	0.98

Table 4.3: Multiple Imputation Method

CRUDE Coinfection rates are shown in table 4.4

No. Tested	$TBHIV$	HIV/TB Coinfection Rate
7,379,664	2,112,688	29% (LCI 25%, UCI 33%)

Table 4.4: CRUDE Coinfection rate

Figures 4.1 and 4.2 shows the yearly Box plots and line graph of HIV/TB coinfection rates. The irregular spikes could be as a result of missing information.

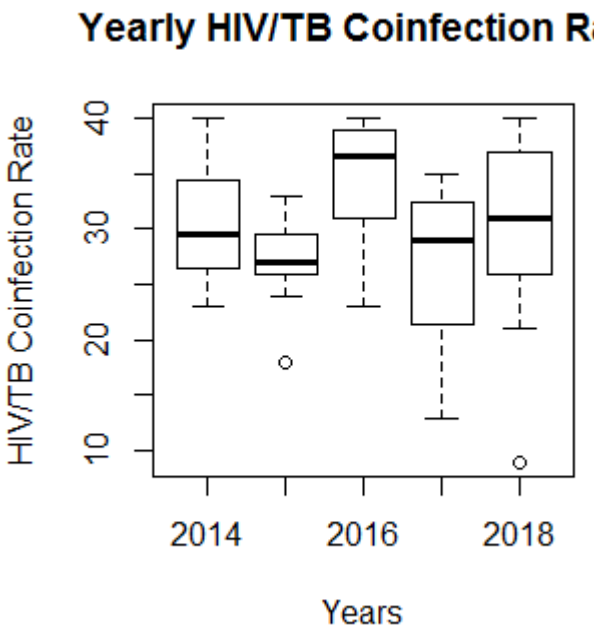


Figure 4.1: Box plots of Yearly HIV/TB Coinfection

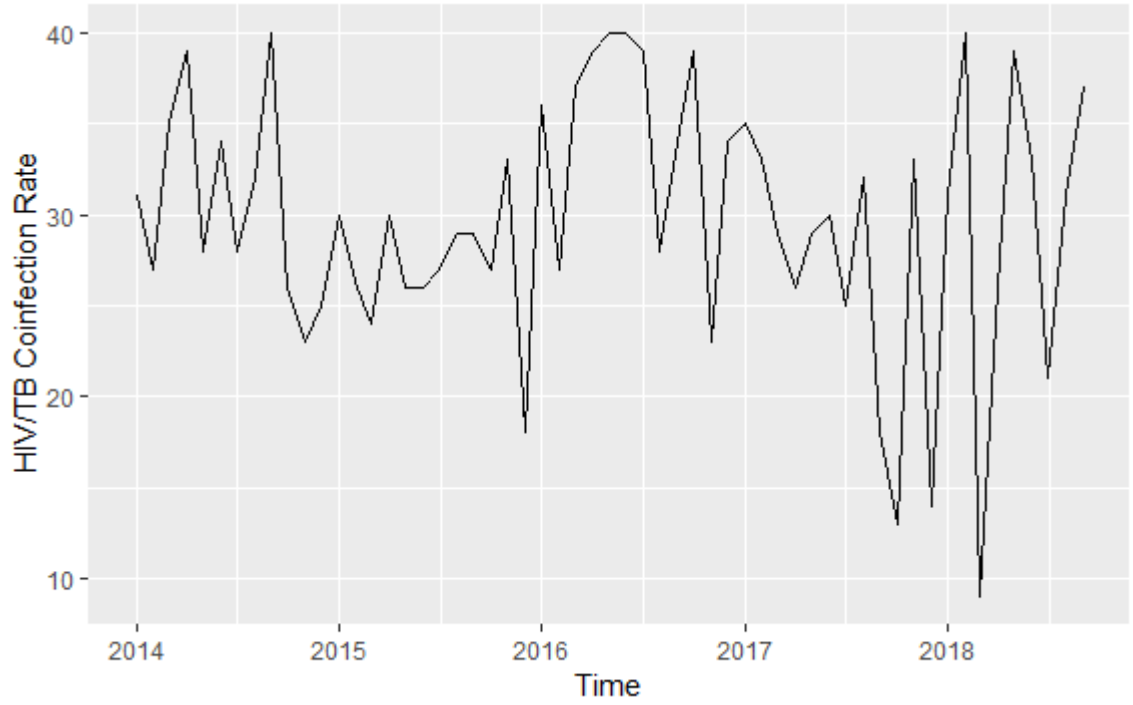


Figure 4.2: Line graph of Yearly HIV/TB Coinfection

Plotting Missingness

We observe mixed arbitrary patterns of missingness for both the data on HIV patients screened for TB and the data on HIV/TB co-infection. The similarity in the patterns can be attributed to fact that one data set is used to yield the other. Figure 4.3 shows a plot the missing values for HIV Patients Screened for TB

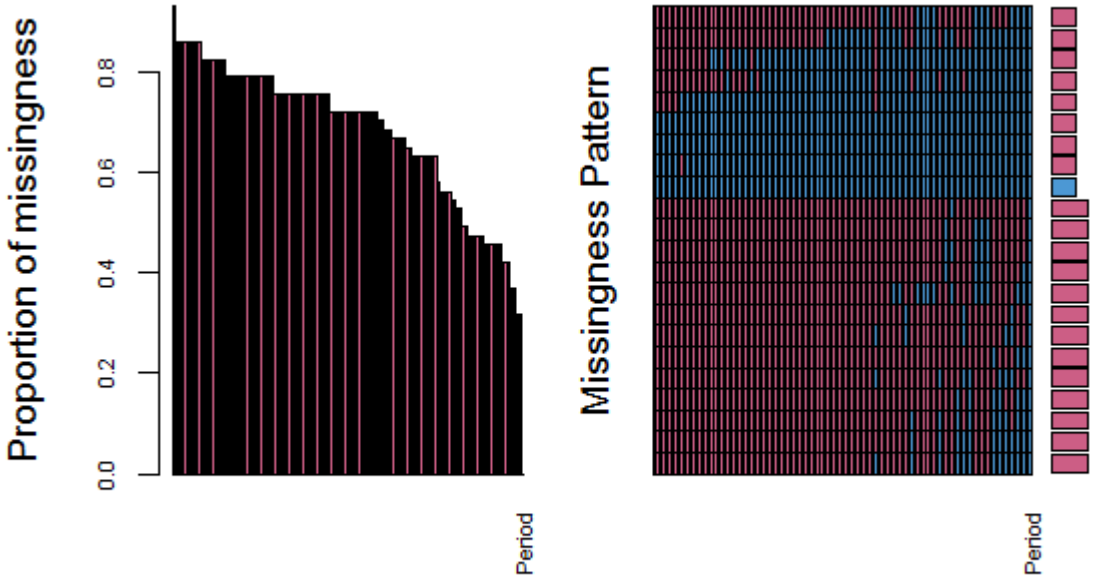


Figure 4.3: Missingness patterns for HIV Patients Screened for TB

Figure 4.4 shows a plot the missing values for HIV/TB Coinfection

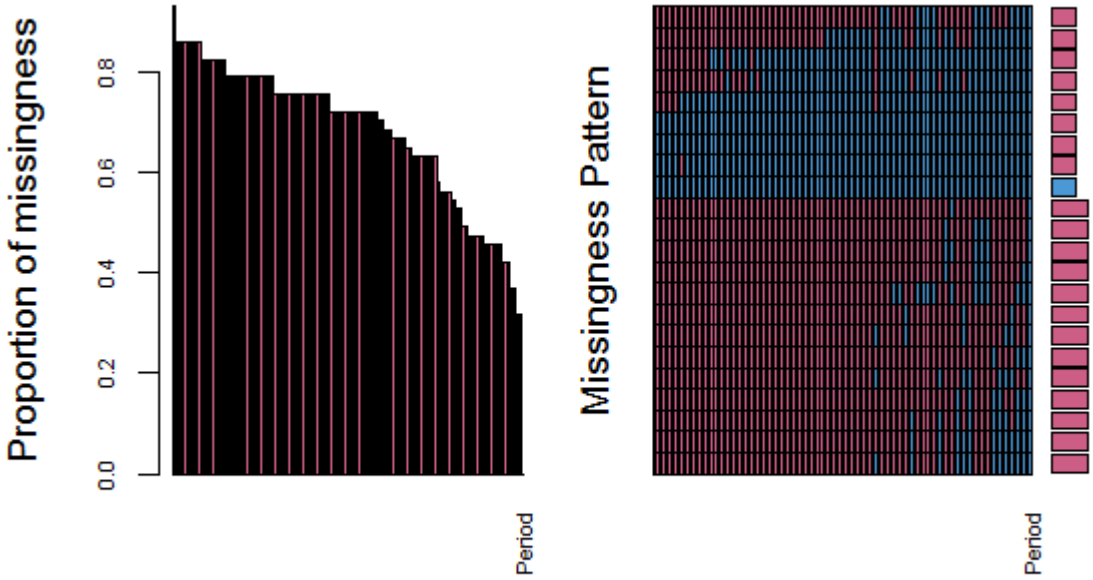


Figure 4.4: Missingness patterns for HIV/TB Coinfection

Results in Table 4.5 show that; complete cases only had a co-infection rate (95% Confidence Interval band) of 29%(25%, 33%), weighted method 27%(23%, 31%), likelihood-based approach 26%(24%, 28%) and multiple imputation approach 21%(20%, 22%). In conclusion, MI remains leading approach to rectify incomplete data and failure to apply it results to overestimation of HIV & TB co-infection rate by 8%.

Comparing Missing Data approaches

Approach	HIV & TB Co-infection Rate	95% LCI	95% UCI
Complete Cases Only	29%	25%	33%
Weighted Method	27%	23%	31%
Likelihood Based approach	26%	24%	28%
Multiple Imputation Approach	21%	20%	22%

Table 4.5: Comparison of four approaches

Complete cases only: Figure 4.5
Results show a cyclic trend

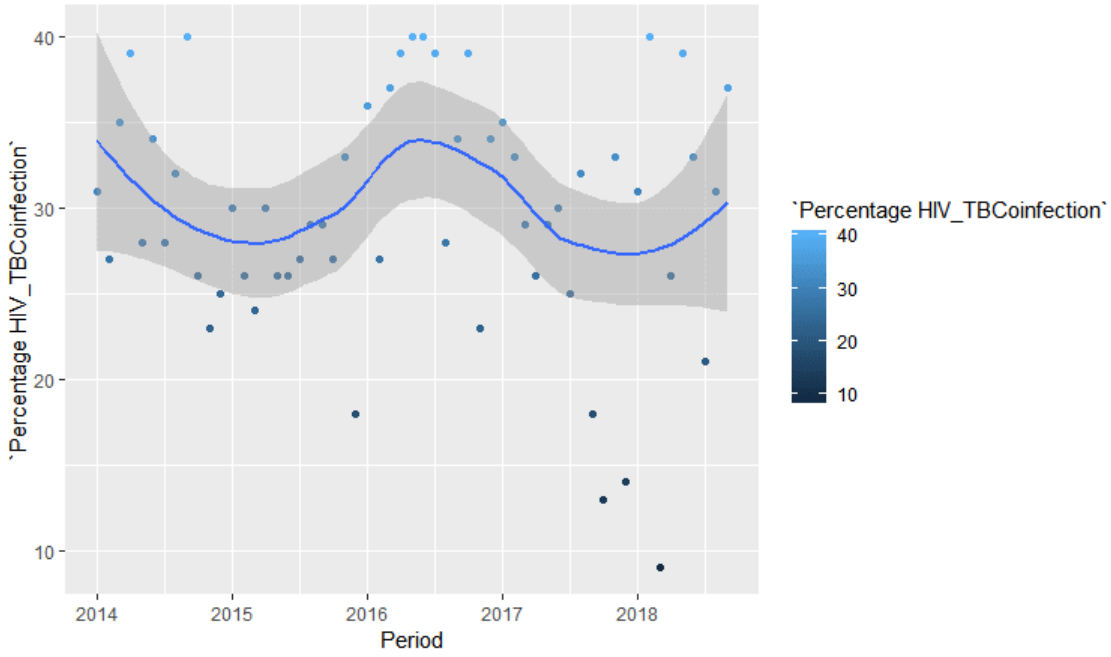


Figure 4.5: Plot of the complete cases only

Weighted Method: Figure 4.6
Shows a deep downward trend in the recent months

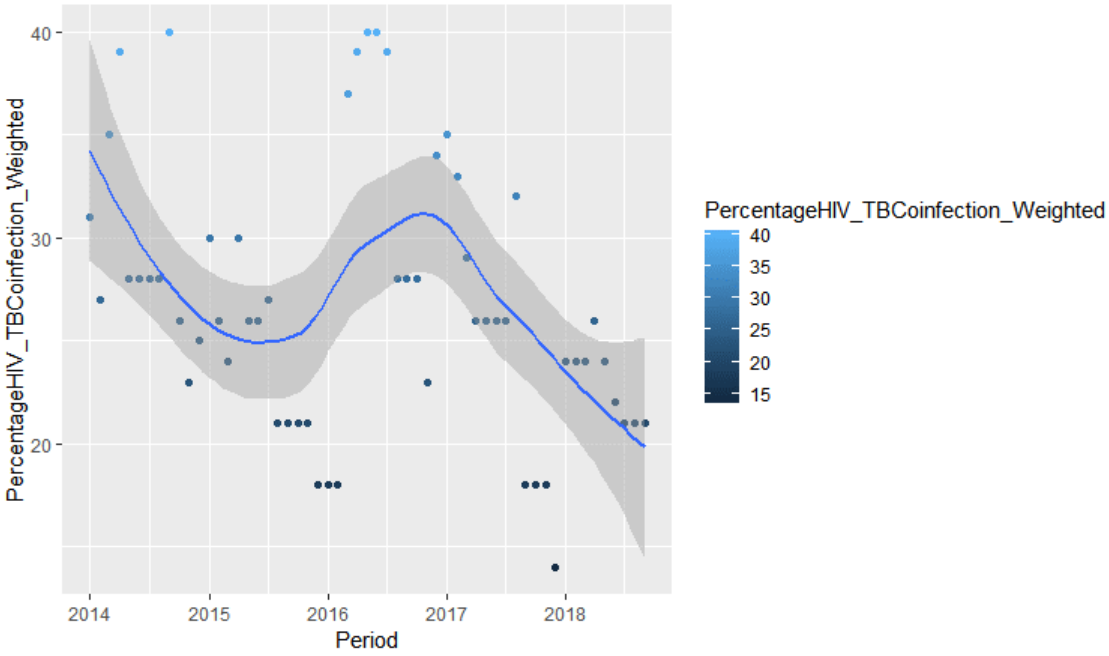


Figure 4.6: Plot of the weighted data

Likelihood Method: Figure 4.7

Shows an upward trend in the recent months

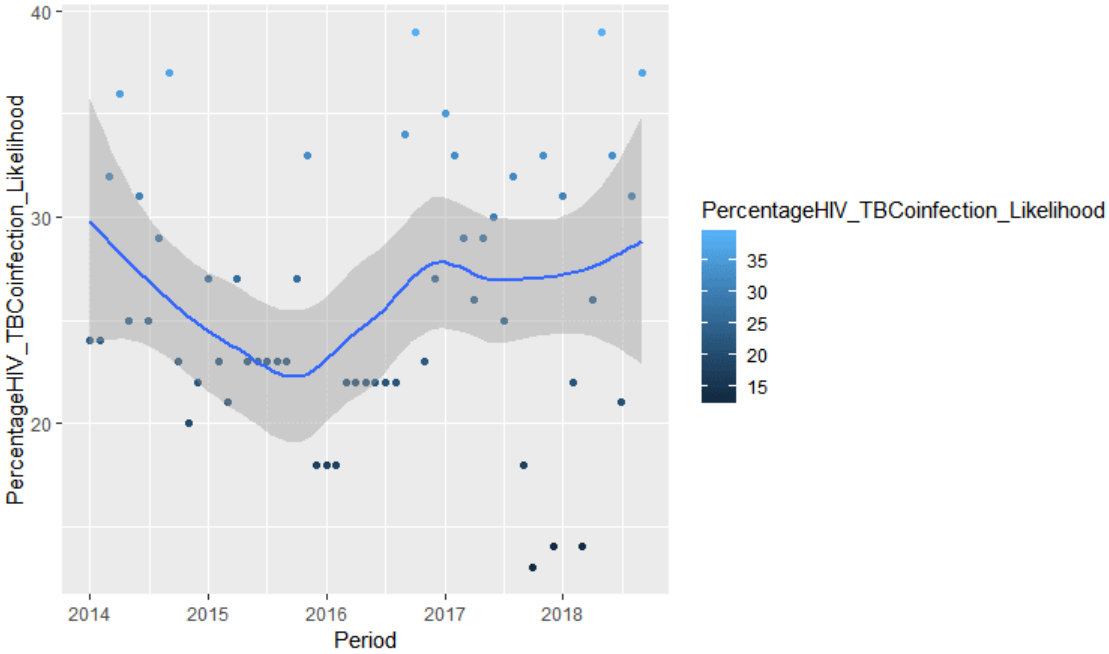


Figure 4.7: Plot of the likelihood estimated data

Multiple Imputation:Figure 4.8

Cyclic trend but generally lower

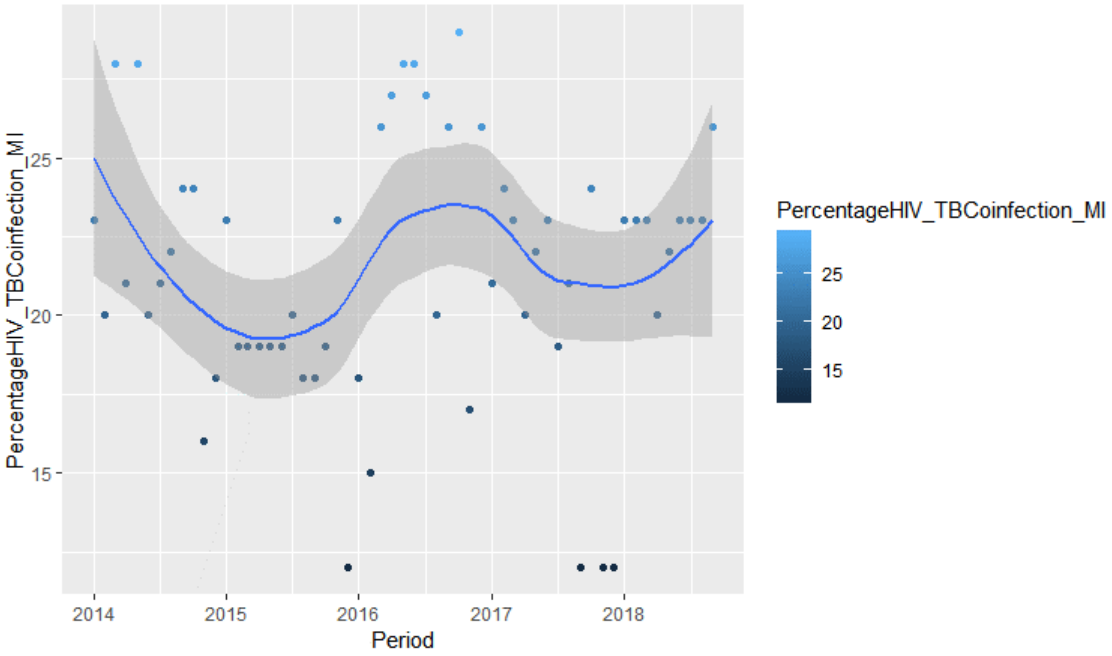


Figure 4.8: Plot of the Multiply Imputed data

Chapter 5

Discussion, Conclusion and Recommendations

5.1 Discussions

We revisited and reviewed the concept of correcting for missing data under the theme of estimating national HIV/TB co-infection. Moreover, we targeted to assess MI's accuracy in situations where models from imputed data are compared with those derived from actual data using statistical learning approach.

It is estimated globally that HIV/TB co-infection is 14 million (Getahun et al., 2010), and TB remains the leading cause of death among PLHIV. HIV infected individuals are at a higher risk of contracting TB compared with HIV-seronegative individuals in high HIV-prevalence countries(Organization, 2013). Of the estimated 8.7 million people who developed TB globally in 2012, 1.1 million (13%) were estimated to be HIV-coinfected. Of the 2.8 million people with TB who actually were screened for HIV in 2012, 20% tested HIV-positive, including 42% of people with TB in sub-Saharan Africa. More than 75% of the estimated HIV-positive incident TB cases live in just 10 countries (Ethiopia, India, Kenya, Mozambique, Nigeria, South Africa, United Republic of Tanzania, Uganda, Zambia, and Zimbabwe) (Organization, 2013).

The increased incidence of active TB in HIV-infected individuals can be attributed to at least two mechanisms: the increased reactivation of latent TB or increased susceptibility to tuberculosis infection. The increased risk of active TB among HIV-infected persons was initially mainly attributed to an increased risk of reactivation of a latent infection.

With all these HIV/TB co-infection estimates at national and international levels failing to deal with missing data can bring dire consequences in HIV programming contexts. Unavailable data are effectively overlooked when applying 'complete case' analyses. Ignoring unavailable data causes problems when the data are not MCAR, as is likely mechanism in HIV/TB setting. We illustrated using extensive simulation, how to fill in missing data with different settings and adjustments. Our methodology entailed:identifying missing data using descriptive statistics; investigation of missingness patterns; defining variables which may be related to missing values to be used for the imputation model; impute unavailable data to give 'm' complete data sets; run the models of interest using the 'm' imputed data sets; combine the 'm' models' parameters; report the final model (as you would have done for any regression model). This methodology can be applied in any research area facing similar missing data problems involves .

The assumptions made by the Statistical Learning Theory framework include the future (i.e. test) observations are related to the past (i.e. training) ones, so that the feature is stationary. At the core of the theory is a probabilistic model of the phenomenon (or data generation process). Within this model, the relationship between past and future observations is that they both are sampled independently from the same distribution (i.i.d.). The independence assumption means that each new observation yields maximum information. The identical distribution

means that the observations give information about the underlying phenomenon (here a probability distribution). An immediate consequence of this very general setting is that one can construct algorithms (e.g. k-nearest neighbors with appropriate k) that are consistent, which means that, as one gets more and more data, the predictions of the algorithm are closer and closer to the optimal ones. So this seems to indicate that we can have some sort of universal algorithm.

5.1.1 Comparison of four approaches

Complete Case Analysis

One major difficulty experienced in available case analysis, is that there is a possibility of generating implausible estimates of covariance matrices in which correlations lie outside of the range of Φ. Inaccuracy in estimation is due to the differing numbers of responses used to estimate covariance matrix. When missingness is MCAR relative performance of complete-case and available case analysis,relys on correlation between the variables; available case paradigm will yield consistent estimates only when the correlation between variables is weak. Utmost difficulty for available case analysis is the impossibility of predicting if available case analysis will give adequate results, in general the method is therefore not useful.

Weighted Method

Normally, the mean imputation brings about overall means that are equal to the complete case values, but variance of variables is underestimated.Underestimation results from two sources. First, filling in unavailable values a single mean value does not account for the variation that would likely occur if the variables were observed because true values almost surely differ from the mean. Second source of underestimation (smaller standard errors) is decreased sample size which inadequately reflects uncertainty existing in the data. A researcher cannot have the same amount of information when some cases are missing important information on some variables as would have been the situation where data are completely observed. Impartiality when estimating variances and standard errors is worsened when estimating multivariate parameters such as regression coefficients. Unless the sample is extremely small, mean imputation always produce biased results.

Likelihood Based approach

Maximum likelihood methods for missing multivariate normal data focus on the estimation of the parameters of the observed data, namely the mean vector and variance-covariance matrix. Because we assume the data multivariate normal, we can utilize the well-known properties of conditional normal distributions to estimate the expected values of the sums and cross products of the variables. Using maximum likelihood with the EM algorithm does not result in values for individual missing variables. The estimates obtained for the means and the variance-covariance matrix of the variables of interest, and then uses of these parameter estimates to obtain model parameters such as the coefficients of a linear regression model.
The one major difficulty with treatment methods for missing data is the computation of the standard errors of estimates (such as the standard error of the mean). Testing whether a mean is significantly different from zero, for example, requires an estimate of how accurate our estimate is. In maximum likelihood theory, the negative second derivative of the observed data log likelihood is needed to obtain standard errors of the estimated mean vector and covariance matrix. This quantity requires algebraic analysis to compute, and is unique to every set of multivariate data.

Multiple Imputation Approach

Multiple imputation avoids two of the difficulties associated with maximum likelihood methods using the EM algorithm. With multiple imputation, a researcher will use standard methods of analysis once imputations are computed, and can easily obtain standard errors of estimates. Though specialized computing is required in multiple imputation, the method provides much more flexibility than in the method described in the previous section. While MI appears to be the most promising method of solving missing data problem, critics of the method center on its expense in terms of: amount of computing and analysis time, heavy costs of analyzing more than one set of data, and the method does require specialized software. Any model fitted by the analyst by use of imputed data sets must include the same variables as the model fitted on originally imputed the data.

Conclusion and Recommendations for further studies

Analysis of missing data without addressing missing values problem yields biased results which may consequentially bring about poor intervention policies.
There is need to understand the structure of missing values in terms of proportion, patterns and mechanisms so as to identify the best method to correct for the missingness. When few cases have unobserved values, available case analysis methods can produce unbiased estimates. In other circumstances, like HIV/TB co-infection setting, available cases make a small fraction of the total cases. Considering expenses incurred during investment in studies and some programs it warrants using methods that utilize as much complete data as possible. When missing data occur, there is need to acknowledge the limitations of our data and use appropriate method(s) to fill in the missing values. In this study we aimed to test the four methods of correcting for missingness on a simulated data set and apply them on a real data, however, we encountered a limitation because the code for MLE did not run on the simulated data set but did on real data set. This is an area that requires more time for further research to help solve the problem. Another area of further research is the replication of this study in machine learning methods for handling missing data.

Bibliography

Allison, P. D. (2001). *Missing data*, volume 136. Sage publications.

Andridge, R. R. and Little, R. J. (2010). A review of hot deck imputation for survey non-response. *International statistical review*, 78(1):40–64.

Baraldi, A. N. and Enders, C. K. (2010). An introduction to modern missing data analyses. *Journal of school psychology*, 48(1):5–37.

Batista, G. and Monard, M. (2003). Experimental comparison of k-nearest neighbor and mean or mode imputation methods with the internal strategies used by c4. 5 and cn2 to treat missing data. *University of Sao Paulo*, 34.

Batista, G. E., Monard, M. C., et al. (2002). A study of k-nearest neighbour as an imputation method. *His*, 87(251-260):48.

Bengio, Y. and Gingras, F. (1996). Recurrent neural networks for missing or asynchronous data. In *Advances in neural information processing systems*, pages 395–401.

Bennett, D. A. (2001). How can i deal with missing data in my study? *Australian and New Zealand journal of public health*, 25(5):464–469.

Berthold, M. R. and Huber, K.-P. (1998). Missing values and learning of fuzzy rules. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 6(02):171–178.

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

Carter, R. L. (2006). Solutions for missing data in structural equation modeling. *Research & Practice in Assessment*, 1.

Chinomona, A. and Mwambi, H. (2015). Multiple imputation for non-response when estimating hiv prevalence using survey data. *BMC public health*, 15(1):1059.

Chung, D. and Merat, F. L. (1996). Neural network based sensor array signal processing. In *1996 IEEE/SICE/RSJ International Conference on Multisensor Fusion and Integration for Intelligent Systems (Cat. No. 96TH8242)*, pages 757–764. IEEE.

Clark, P. and Niblett, T. (1989). The cn2 induction algorithm. *Machine learning*, 3(4):261–283.

Corbett, E. L., Watt, C. J., Walker, N., Maher, D., Williams, B. G., Raviglione, M. C., and Dye, C. (2003). The growing burden of tuberculosis: global trends and interactions with the hiv epidemic. *Archives of internal medicine*, 163(9):1009–1021.

Council, N. R. et al. (2010). *The prevention and treatment of missing data in clinical trials*. National Academies Press.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22.

Dong, Y. and Peng, C.-Y. J. (2013). Principled missing data methods for researchers. *SpringerPlus*, 2(1):222.

Enders, C. K. and Bandalos, D. L. (2001). The relative performance of full information maximum likelihood estimation for missing data in structural equation models. *Structural equation modeling*, 8(3):430–457.

Fessant, F. and Midenet, S. (2002). Self-organising map for data imputation and correction in surveys. *Neural Computing & Applications*, 10(4):300–310.

Gabrys, B. (2000). Pattern classification for incomplete data. In *KES'2000. Fourth International Conference on Knowledge-Based Intelligent Engineering Systems and Allied Technologies. Proceedings (Cat. No. 00TH8516)*, volume 1, pages 454–457. IEEE.

Galimard, J.-E., Chevret, S., Curis, E., and Resche-Rigon, M. (2018). Heckman imputation models for binary or continuous mnar outcomes and mar predictors. *BMC medical research methodology*, 18(1):90.

García-Laencina, P. J., Sancho-Gómez, J.-L., and Figueiras-Vidal, A. R. (2010). Pattern classification with missing data: a review. *Neural Computing and Applications*, 19(2):263–282.

García-Laencina, P. J., Serrano, J., Figueiras-Vidal, A. R., and Sancho-Gómez, J.-L. (2007). Multi-task neural networks for dealing with missing inputs. In *International Work-Conference on the Interplay Between Natural and Artificial Computation*, pages 282–291. Springer.

Getahun, H., Gunneberg, C., Granich, R., and Nunn, P. (2010). Hiv infection—associated tuberculosis: the epidemiology and the response. *Clinical Infectious Diseases*, 50(Supplement_3):S201–S207.

Graham, J. W. (2009). Missing data analysis: Making it work in the real world. *Annual review of psychology*, 60:549–576.

Haitovsky, Y. (1968). Missing data in regression analysis. *Journal of the Royal Statistical Society: Series B (Methodological)*, 30(1):67–82.

Hamer, R. M. and Simpson, P. M. (2009). Last observation carried forward versus mixed models in the analysis of psychiatric clinical trials.

Hartley, H. and Hocking, R. (1971). The analysis of incomplete data. *Biometrics*, pages 783–823.

Hathaway, R. J. and Bezdek, J. C. (2001). Fuzzy c-means clustering of incomplete data. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 31(5):735–744.

Honaker, J., King, G., Blackwell, M., et al. (2011). Amelia ii: A program for missing data. *Journal of statistical software*, 45(7):1–47.

Horton, N. J. and Lipsitz, S. R. (2001). Multiple imputation in practice: comparison of software packages for regression models with missing variables. *The American Statistician*, 55(3):244–254.

Ichihashi, H. and Honda, K. (2005). Fuzzy c-means classifier for incomplete data sets with outliers and missing values. In *International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce (CIMCA-IAWTIC'06)*, volume 2, pages 457–464. IEEE.

Im, J. (2015). Some methods for handling missing data in surveys.

Ishibuchi, H. and Tanaka, H. (1991). An extension of the bp-algorithm to interval input vectors-learning from numerical data and expert's knowledge. In *[Proceedings] 1991 IEEE International Joint Conference on Neural Networks*, pages 1588–1593. IEEE.

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

Jiang, K., Chen, H., and Yuan, S. (2005). Classification for incomplete data using classifier ensembles. In *2005 International Conference on Neural Networks and Brain*, volume 1, pages 559–563. IEEE.

Kallin, L. (2002). Missing data and the preprocessing perceptron. Technical report, Tech. Rep., Umeaå University.

Kang, H. (2013). The prevention and handling of the missing data. *Korean journal of anesthesiology*, 64(5):402.

Karahalios, A., Baglietto, L., Carlin, J. B., English, D. R., and Simpson, J. A. (2012). A review of the reporting and handling of missing data in cohort studies with repeated assessment of exposure measures. *BMC medical research methodology*, 12(1):96.

Kohonen, T. (2012). *Self-organizing maps*, volume 30. Springer Science & Business Media.

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

Marseguerra, M. and Zoia, A. (2005). The autoassociative neural network in signal analysis: Ii. application to on-line monitoring of a simulated bwr component. *Annals of Nuclear Energy*, 32(11):1207–1223.

McKnight, P. E., McKnight, K. M., Sidani, S., and Figueredo, A. J. (2007). *Missing data: A gentle introduction*. Guilford Press.

McNeish, D. (2017). Missing data methods for arbitrary missingness with small samples. *Journal of Applied Statistics*, 44(1):24–39.

Molnar, F. J., Hutton, B., and Fergusson, D. (2008). Does analysis using "last observation carried forward" introduce bias in dementia research? *Cmaj*, 179(8):751–753.

Nakagawa, S. (2015). Missing data: mechanisms, methods and messages. *Ecological statistics: Contemporary theory and application*, pages 81–105.

Nakai, M. and Ke, W. (2011). Review of the methods for handling missing data in longitudinal data analysis. *International Journal of Mathematical Analysis*, 5(1):1–13.

Narayanan, S., Marks, R., Vian, J. L., Choi, J., El-Sharkawi, M., and Thompson, B. B. (2002). Set constraint discovery: missing sensor data restoration using autoassociative regression machines. In *Proceedings of the 2002 International Joint Conference on Neural Networks. IJCNN'02 (Cat. No. 02CH37290)*, volume 3, pages 2872–2877. IEEE.

Nauck, D. and Kruse, R. (1999). Learning in neuro-fuzzy systems with symbolic attributes and missing values. In *ICONIP'99. ANZIIS'99 & ANNES'99 & ACNN'99. 6th International Conference on Neural Information Processing. Proceedings (Cat. No. 99EX378)*, volume 1, pages 142–147. IEEE.

NTLD-P (2017). *Annual Report 2017*. https://www.nltp.co.ke/annual-reports/#.

NTLD-P (2018). *Annual Report 2018*. https://www.nltp.co.ke/annual-reports/#.

Organization, W. H. (2013). *Global tuberculosis report 2013*. World Health Organization.

Organization, W. H. et al. (2010). The global plan to stop tb 2011-2015: transforming the fight towards elimination of tuberculosis.

Papathakis, P. and Piwoz, E. (2008). Nutrition and tuberculosis: A review of the literature and considerations for tb control programs. *United States Agency for International Development, Africa's Health 2010 Project*, page 1.

Pelckmans, K., De Brabanter, J., Suykens, J. A., and De Moor, B. (2005). Handling missing values in support vector machine classifiers. *Neural Networks*, 18(5-6):684–692.

Peng, C.-Y. J., Harwell, M., Liou, S.-M., Ehman, L. H., et al. (2006). Advances in missing data methods and implications for educational research. *Real data analysis*, 3178.

Petit-renaud, S. and Denoeux, T. (1998). A neuro-fuzzy model for missing data reconstruction.

Piela, P. (2002). Introduction to self-organizing maps modelling for imputation—techniques and technology. *Research in Official Statistics*, 2:5–19.

Quinlan, J. R. (1989). Unknown attribute values in induction. In *Proceedings of the sixth international workshop on Machine learning*, pages 164–168. Elsevier.

Quinlan, R. (1993). 4.5: Programs for machine learning morgan kaufmann publishers inc. *San Francisco, USA*.

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

Rubin, D. B. (2004). *Multiple imputation for nonresponse in surveys*, volume 81. John Wiley & Sons.

Sarkar, M. and Leong, T.-Y. (2001). Fuzzy k-means clustering with missing values. In *Proceedings of the AMIA Symposium*, page 588. American Medical Informatics Association.

Schafer, J. L. (1997). *Analysis of incomplete multivariate data*. CRC press.

Schafer, J. L. (1999). Multiple imputation: a primer. *Statistical methods in medical research*, 8(1):3–15.

Schouten, R. M., Lugtig, P., and Vink, G. (2018). Generating missing values for simulation purposes: a multivariate amputation procedure. *Journal of Statistical Computation and Simulation*, 88(15):2909–2930.

Sharpe, P. K. and Solly, R. (1995). Dealing with missing values in neural network-based diagnostic systems. *Neural Computing & Applications*, 3(2):73–77.

Sim, J., Lee, J. S., and Kwon, O. (2015). Missing values and optimal selection of an imputation method and classification algorithm to improve the accuracy of ubiquitous computing applications. *Mathematical problems in engineering*, 2015.

Smola, A. J., Vishwanathan, S., and Hofmann, T. (2005). Kernel methods for missing variables. In *AISTATS*. Citeseer.

Tabachnick, B. G., Fidell, L. S., and Ullman, J. B. (2007). *Using multivariate statistics*, volume 5. Pearson Boston, MA.

TNLJWD, Z., Bousquet, O., and Scholkopf, B. (2004). Adv. neural inf. process. syst. *Learning with local and global consistency*, 2.

Van Buuren, S. (2012). *Flexible imputation of missing data*. CRC press.

Vapnik, V. (2013). *The nature of statistical learning theory*. Springer science & business media.

Webb, G. I. (1998). The problem of missing values in decision tree grafting. In *Australian Joint Conference on Artificial Intelligence*, pages 273–283. Springer.

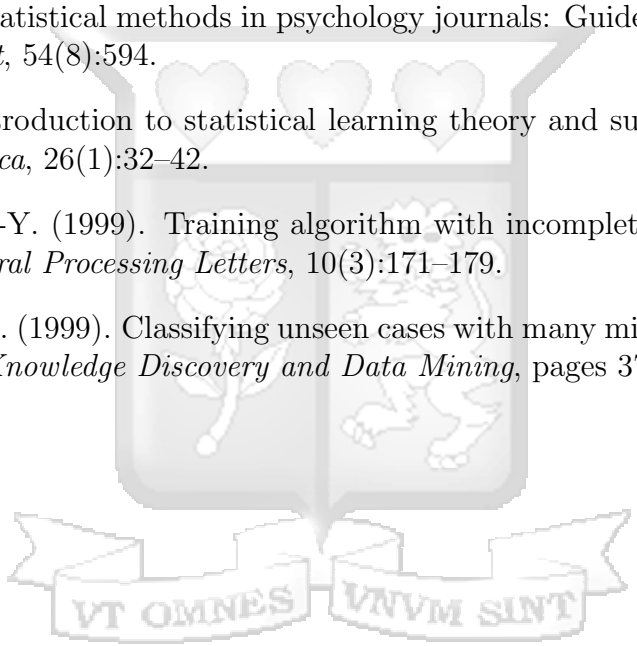
White, I. R., Royston, P., and Wood, A. M. (2011). Multiple imputation using chained equations: issues and guidance for practice. *Statistics in medicine*, 30(4):377–399.

Wilkinson, L. (1999). Statistical methods in psychology journals: Guidelines and explanations. *American psychologist*, 54(8):594.

Xuegong, Z. (2000). Introduction to statistical learning theory and support vector machines. *Acta Automatica Sinica*, 26(1):32–42.

Yoon, S.-Y. and Lee, S.-Y. (1999). Training algorithm with incomplete data for feed-forward neural networks. *Neural Processing Letters*, 10(3):171–179.

Zheng, Z. and Low, B. T. (1999). Classifying unseen cases with many missing values. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 370–375. Springer.



Appendices

R Scripts

```
[H]
rm(list = ls())
library(mice)
library(VIM)
library(ISLR)
library(MASS)
library(caret)

#Simulations
set.seed(1)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
frq(HIV_TB_Conifection)
mod <- glm(HIV_TB_Conifection ~ Gender + Age+weight, family = "binomial")
summary(mod)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
str(Default2)
#View(Default)
#Original complete data
attach(Default2)
library(tibble)
as_tibble(Default2)
#Check for missingness
md.pattern(Default2)
#Predictive modelling
# We split the data into:  testing and  training sets.
set.seed(42)
default_idx = sample(nrow(Default2), 5000)
default_trn = Default2[default_idx, ]
default_tst = Default2[-default_idx, ]
#view(default_trn)

model_glm_1 = glm(HIV_TB_Conifection~., data = default_trn, family = "binomial")
```

```
model_glm_1
summary(model_glm_1)

model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_2
summary(model_glm_2)

#obtain the fitted coefficients
coef(model_glm_2)

#Make predictions
head(predict(model_glm_2, type = "link"))

#To obtain the predicted probabilities
head(predict(model_glm_2, type = "response"))

#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0.5, "Yes", "No")
frq(model_glm_pred2)
#model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the training classificati
calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
}

#training classification error RATE
calc_class_err(actual = default_trn$HIV_TB_Conifection, predicted = model_glm_pred2)

#We use table() and confusionMatrix() functions to obtain many more metrics.
train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes")
library(caret)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")

c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])

#ROC Curve
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)

################################################################################
#CREATING MISSINGNESS
################################################################################

require("mice")
library(mice)
```

```
#Function to make sure that the graphs are fully displayed
resetPar <- function() {
  dev.new()
  op <- par(no.readonly = TRUE)
  dev.off()
  op
}
par(resetPar())

#CREATING MISSINGNESS

#1.)  Creating 7% Missingness Under MAR

#rm(list = ls())
set.seed(100)
attach(Default2)
Amputation_MAR_.07_1<-ampute(Default2, prop = 0.07,freq = NULL, mech = "MAR",std = TRUE,
Amputation_MAR_.07_1

#ASSESSING MISSINGNESS

Amputation_MAR_.07_1$amp
str(Amputation_MAR_.07_1)#Realize that all variables have been converted into ints(need
Amputation_MAR_.07_1$patterns
Amputation_MAR_.07_1$weights
Amputation_MAR_.07_1$prop

Amputation_MAR_.07_1<-as.data.frame(Amputation_MAR_.07_1$amp)
P1<-function(x) {sum(is.na(x))/length(x)*100}

apply(Amputation_MAR_.07_1,2,P1) #Gives proportion of missingness in each variable
md.pattern(Amputation_MAR_.07_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MAR_.07_1)# Bivariate comparison
aggr_plot <- aggr(Amputation_MAR_.07_1, col=c('navyblue','yellow'), numbers=TRUE, sortVa

###Diagnosing Missingness  pattern and mechanism
#install.packages("Amelia")
library(Amelia)
#AmeliaView()
missmap(Amputation_MAR_.07_1)
Missingness <- ifelse (is.na (Amputation_MAR_.07_1) == TRUE, 0, 1)
# create the missingness matrix
MissData <- data.frame (Amputation_MAR_.07_1, Missingness)
library (psych) # loading the psych library
pairs.panels (MissData, ellipses = FALSE, method = "spearman")

#CORRECTING FOR MISSINGNESS: COMPLETE CASE ANALYSIS
set.seed(42)
dim(Amputation_MAR_.07_1)
str(Amputation_MAR_.07_1)
complete_cases_0.07_MAR<-Amputation_MAR_.07_1[complete.cases(Amputation_MAR_.07_1),]
```

```
View(complete_cases_0.07_MAR)
dim(complete_cases_0.07_MAR)
str(complete_cases_0.07_MAR)
md.pattern(complete_cases_0.07_MAR)

#CONVERTING BACK TO ORIGINAL DATA TYPES

library(tibble)
as_tibble(complete_cases_0.07_MAR)

complete_cases_0.07_MAR<-data.frame(complete_cases_0.07_MAR)
library(dplyr)

drops<-c("Age","weight")
DefaultStd<-complete_cases_0.07_MAR[ , !(names(complete_cases_0.07_MAR) %in% drops)]
str(DefaultStd)
con.names = DefaultStd %>% select_if(is.numeric) %>% colnames()
print(con.names)

DefaultStdConverted_MAR_.07<-complete_cases_0.07_MAR
DefaultStdConverted_MAR_.07[,con.names] = data.frame(apply(DefaultStdConverted_MAR_.07[c
#str(DefaultStdConverted[,con.names])

DefaultStdConverted_MAR_.07
dim(DefaultStdConverted_MAR_.07)
str(DefaultStdConverted_MAR_.07)

##ANALYSIS: Complete cases 7% MAR

# We split the data into:  testing and  training sets.
set.seed(42)
default_idx = sample(nrow(DefaultStdConverted_MAR_.07), 4656)
default_trn = DefaultStdConverted_MAR_.07[default_idx, ]
default_tst = DefaultStdConverted_MAR_.07[-default_idx, ]

model_glm_1 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_1
summary(model_glm_1)

model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_2
summary(model_glm_2)

#obtain the fitted coefficients
coef(model_glm_2)

#Make predictions
head(predict(model_glm_2, type = "link"))
#To obtain the predicted probabilities
head(predict(model_glm_2, type = "response"))
```

```
#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the training classificati
calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
}

#training classification error RATE
calc_class_err(actual = default_trn$default, predicted = model_glm_pred2)

#We use table() and confusionMatrix() functions to obtain many more metrics.
train_tab = table(predicted = model_glm_pred2, actual = default_trn$default)
train_tab # The row labels should be same as column labels. We currently have "No, yes"
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")

c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])

#ROC Curve
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)

#2.)  Creating 7% Missingness Under MNAR

rm(list = ls())
set.seed(100)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
Amputation_MNAR_.07_1<-ampute(Default2, prop = 0.07,freq = NULL, mech = "MNAR",std = TRU
Amputation_MNAR_.07_1

#ASSESSING MISSINGNESS
Amputation_MNAR_.07_1$amp
str(Amputation_MNAR_.07_1)#Realize that all variables have been converted into ints(need
Amputation_MNAR_.07_1$patterns
```

```
Amputation_MNAR_.07_1$weights
Amputation_MNAR_.07_1$prop

Amputation_MNAR_.07_1<-as.data.frame(Amputation_MNAR_.07_1$amp)
P1<-function(x) {sum(is.na(x))/length(x)*100}

apply(Amputation_MNAR_.07_1,2,P1) #Gives proportion of missingness in each variable
md.pattern(Amputation_MNAR_.07_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MNAR_.07_1)# Bivariate comparison
aggr_plot <- aggr(Amputation_MNAR_.07_1, col=c('navyblue','yellow'), numbers=TRUE, sortV

#CORRECTING FOR MISSINGNESS: COMPLETE CASE ANALYSIS
set.seed(42)
dim(Amputation_MNAR_.07_1)
str(Amputation_MNAR_.07_1)
complete_cases_0.07_MNAR<-Amputation_MNAR_.07_1[complete.cases(Amputation_MNAR_.07_1),]
View(complete_cases_0.07_MNAR)
dim(complete_cases_0.07_MNAR)
str(complete_cases_0.07_MNAR)
md.pattern(complete_cases_0.07_MNAR)

#CONVERTING BACK TO ORIGINAL DATA TYPES
library(tibble)
as_tibble(complete_cases_0.07_MNAR)
complete_cases_0.07_MNAR<-data.frame(complete_cases_0.07_MNAR)
library(dplyr)

drops<-c("Age","weight")
DefaultStd<-complete_cases_0.07_MNAR[ , !(names(complete_cases_0.07_MNAR) %in% drops)]
str(DefaultStd)
con.names = DefaultStd %>% select_if(is.numeric) %>% colnames()
print(con.names)

DefaultStdConverted_MNAR_.07<-complete_cases_0.07_MNAR
DefaultStdConverted_MNAR_.07[,con.names] = data.frame(apply(DefaultStdConverted_MNAR_.07
#str(DefaultStdConverted[,con.names])

DefaultStdConverted_MNAR_.07
dim(DefaultStdConverted_MNAR_.07)
str(DefaultStdConverted_MNAR_.07)

##ANALYSIS: Complete cases 7% MNAR

# We split the data into:  testing and  training sets.
set.seed(42)
default_idx = sample(nrow(DefaultStdConverted_MNAR_.07), 4656)
default_trn = DefaultStdConverted_MNAR_.07[default_idx, ]
default_tst = DefaultStdConverted_MNAR_.07[-default_idx, ]

model_glm_1 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_1
```

```
summary(model_glm_1)

model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_2
summary(model_glm_2)

#obtain the fitted coefficients
coef(model_glm_2)

#Make predictions
head(predict(model_glm_2, type = "link"))

#To obtain the predicted probabilities
head(predict(model_glm_2, type = "response"))
#Note that these are probabilities, not classifications.
#To obtain classifications, we will need to compare to the correct cutoff value with an


#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the training classificati
calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
}

#training classification error RATE
calc_class_err(actual = default_trn$default, predicted = model_glm_pred2)

#We use table() and confusionMatrix() functions to obtain many more metrics.
train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")

c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])

#ROC Curves
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)

#2.)  Creating 7% Missingness Under MCAR

rm(list = ls())
```

```
set.seed(100)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
Amputation_MCAR_.07_1<-ampute(Default2, prop = 0.07,freq = NULL, mech = "MCAR",std = TRU
Amputation_MCAR_.07_1

#ASSESSING MISSINGNESS
Amputation_MCAR_.07_1$amp
str(Amputation_MCAR_.07_1)
Amputation_MCAR_.07_1$patterns
Amputation_MCAR_.07_1$weights
Amputation_MCAR_.07_1$prop

Amputation_MCAR_.07_1<-as.data.frame(Amputation_MCAR_.07_1$amp)
P1<-function(x) {sum(is.na(x))/length(x)*100}

apply(Amputation_MCAR_.07_1,2,P1) #Gives proportion of missingness in each variable
md.pattern(Amputation_MCAR_.07_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MCAR_.07_1)# Bivariate comparison
aggr_plot <- aggr(Amputation_MCAR_.07_1, col=c('navyblue','yellow'), numbers=TRUE, sortV

#CORRECTING FOR MISSINGNESS: COMPLETE CASE ANALYSIS
set.seed(42)
dim(Amputation_MCAR_.07_1)
str(Amputation_MCAR_.07_1)
complete_cases_0.07_MCAR<-Amputation_MCAR_.07_1[complete.cases(Amputation_MCAR_.07_1),]
View(complete_cases_0.07_MCAR)
dim(complete_cases_0.07_MCAR)
str(complete_cases_0.07_MCAR)
md.pattern(complete_cases_0.07_MCAR)

#ANALYSIS: Complete cases 7% MCAR

# We split the data into:  testing and  training sets.
set.seed(42)
default_idx = sample(nrow(complete_cases_0.07_MCAR), 4678)
default_trn = complete_cases_0.07_MCAR[default_idx, ]
default_tst = complete_cases_0.07_MCAR[-default_idx, ]

model_glm_1 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_1
summary(model_glm_1)

model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
```

```
model_glm_2
summary(model_glm_2)

#obtain the fitted coefficients
coef(model_glm_2)

#Make predictions
head(predict(model_glm_2, type = "link"))

#To obtain the predicted probabilities
head(predict(model_glm_2, type = "response"))
#Note that these are probabilities, not classifications.
#To obtain classifications, we will need to compare to the correct cutoff value with an

#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the training classificati
calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
}

#training classification error RATE
calc_class_err(actual = default_trn$HIV_TB_Conifection, predicted = model_glm_pred2)

#We use table() and confusionMatrix() functions to obtain many more metrics.
train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")

c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])

#ROC Curves
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)

################################################################################
#1.)  Creating 10% Missingness Under MAR
rm(list = ls())
set.seed(100)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
```

```
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
attach(Default)
Amputation_MAR_.10_1<-ampute(Default2, prop = 0.10,freq = NULL, mech = "MAR",std = TRUE,
Amputation_MAR_.10_1

#ASSESSING MISSINGNESS

Amputation_MAR_.10_1$amp
str(Amputation_MAR_.10_1)
Amputation_MAR_.10_1$patterns
Amputation_MAR_.10_1$weights
Amputation_MAR_.10_1$prop

Amputation_MAR_.10_1<-as.data.frame(Amputation_MAR_.10_1$amp)
P1<-function(x) {sum(is.na(x))/length(x)*100}

apply(Amputation_MAR_.10_1,2,P1) #Gives proportion of missingness in each variable
md.pattern(Amputation_MAR_.10_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MAR_.10_1)# Bivariate comparison
aggr_plot <- aggr(Amputation_MAR_.10_1, col=c('navyblue','yellow'), numbers=TRUE, sortVa

#CORRECTING FOR MISSINGNESS: COMPLETE CASE ANALYSIS
set.seed(42)
dim(Amputation_MAR_.10_1)
str(Amputation_MAR_.10_1)
complete_cases_0.10_MAR<-Amputation_MAR_.10_1[complete.cases(Amputation_MAR_.10_1),]
View(complete_cases_0.10_MAR)
dim(complete_cases_0.10_MAR)
str(complete_cases_0.10_MAR)
md.pattern(complete_cases_0.10_MAR)

#CONVERTING BACK TO ORIGINAL DATA TYPES

library(tibble)
as_tibble(complete_cases_0.10_MAR)

complete_cases_0.10_MAR<-data.frame(complete_cases_0.10_MAR)
library(dplyr)

#drops<-c("balance","income")
#DefaultStd<-complete_cases_0.10_MAR[ , !(names(complete_cases_0.10_MAR) %in% drops)]
#str(DefaultStd)
#con.names = DefaultStd %>% select_if(is.numeric) %>% colnames()
#print(con.names)

DefaultStdConverted_MAR_.10<-complete_cases_0.10_MAR
```

```
#DefaultStdConverted_MAR_.10[,con.names] = data.frame(apply(DefaultStdConverted_MAR_.10[
#str(DefaultStdConverted[,con.names])

DefaultStdConverted_MAR_.10
dim(DefaultStdConverted_MAR_.10)
str(DefaultStdConverted_MAR_.10)

##ANALYSIS: Complete cases 10% MAR
# We split the data into:  testing and  training sets.
set.seed(42)
default_idx = sample(nrow(DefaultStdConverted_MAR_.10), 4502)
default_trn = DefaultStdConverted_MAR_.10[default_idx, ]
default_tst = DefaultStdConverted_MAR_.10[-default_idx, ]

model_glm_1 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_1
summary(model_glm_1)

model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_2
summary(model_glm_2)

#obtain the fitted coefficients
coef(model_glm_2)

#Make predictions
head(predict(model_glm_2, type = "link"))

#To obtain the predicted probabilities
head(predict(model_glm_2, type = "response"))

#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the training classificati
calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
}

#training classification error RATE
calc_class_err(actual = default_trn$HIV_TB_Conifection, predicted = model_glm_pred2)

#We use table() and confusionMatrix() functions to obtain many more metrics.
train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")
```

```
c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])

#ROC Curves
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)

#2.)  Creating 10% Missingness Under MNAR
rm(list = ls())
set.seed(100)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
attach(Default2)
Amputation_MNAR_.10_1<-ampute(Default2, prop = 0.10,freq = NULL, mech = "MNAR",std = TRU
Amputation_MNAR_.10_1

#ASSESSING MISSINGNESS
Amputation_MNAR_.10_1$amp
str(Amputation_MNAR_.10_1)
Amputation_MNAR_.10_1$patterns
Amputation_MNAR_.10_1$weights
Amputation_MNAR_.10_1$prop

Amputation_MNAR_.10_1<-as.data.frame(Amputation_MNAR_.10_1$amp)
P1<-function(x) {sum(is.na(x))/length(x)*100}

apply(Amputation_MNAR_.10_1,2,P1) #Gives proportion of missingness in each variable
md.pattern(Amputation_MNAR_.10_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MNAR_.10_1)# Bivariate comparison
aggr_plot <- aggr(Amputation_MNAR_.10_1, col=c('navyblue','yellow'), numbers=TRUE, sortV

#CORRECTING FOR MISSINGNESS: COMPLETE CASE ANALYSIS
set.seed(42)
dim(Amputation_MNAR_.10_1)
str(Amputation_MNAR_.10_1)
complete_cases_0.10_MNAR<-Amputation_MNAR_.10_1[complete.cases(Amputation_MNAR_.10_1),]
View(complete_cases_0.10_MNAR)
dim(complete_cases_0.10_MNAR)
str(complete_cases_0.10_MNAR)
md.pattern(complete_cases_0.10_MNAR)
```

```
#CONVERTING BACK TO ORIGINAL DATA TYPES
library(tibble)
as_tibble(complete_cases_0.10_MNAR)

complete_cases_0.10_MNAR<-data.frame(complete_cases_0.10_MNAR)
library(dplyr)

#drops<-c("balance","income")
#DefaultStd<-complete_cases_0.10_MNAR[ , !(names(complete_cases_0.10_MNAR) %in% drops)]
#str(DefaultStd)
#con.names = DefaultStd %>% select_if(is.numeric) %>% colnames()
#print(con.names)

DefaultStdConverted_MNAR_.10<-complete_cases_0.10_MNAR
#DefaultStdConverted_MNAR_.10[,con.names] = data.frame(apply(DefaultStdConverted_MNAR_.1
#str(DefaultStdConverted[,con.names])

DefaultStdConverted_MNAR_.10
dim(DefaultStdConverted_MNAR_.10)
str(DefaultStdConverted_MNAR_.10)

##ANALYSIS: Complete cases 10% MNAR

# We split the data into:  testing and  training sets.
set.seed(42)
default_idx = sample(nrow(DefaultStdConverted_MNAR_.10), 4502)
default_trn = DefaultStdConverted_MNAR_.10[default_idx, ]
default_tst = DefaultStdConverted_MNAR_.10[-default_idx, ]

model_glm_1 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_1
summary(model_glm_1)

model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_2
summary(model_glm_2)

#obtain the fitted coefficients
coef(model_glm_2)

#Make predictions
head(predict(model_glm_2, type = "link"))

#To obtain the predicted probabilities
head(predict(model_glm_2, type = "response"))

#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the training classificati
```

```
calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
}

#training classification error RATE
calc_class_err(actual = default_trn$HIV_TB_Conifection, predicted = model_glm_pred2)

#We use table() and confusionMatrix() functions to obtain many more metrics.
train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")

c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])

#ROC Curves
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)

#3.)  Creating 10% Missingness Under MCAR

rm(list = ls())
set.seed(100)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
attach(Default2)
Amputation_MCAR_.10_1<-ampute(Default2, prop = 0.10,freq = NULL, mech = "MCAR",std = TRU
Amputation_MCAR_.10_1

#ASSESSING MISSINGNESS

Amputation_MCAR_.10_1$amp
str(Amputation_MCAR_.10_1)
Amputation_MCAR_.10_1$patterns
Amputation_MCAR_.10_1$weights
Amputation_MCAR_.10_1$prop

Amputation_MCAR_.10_1<-as.data.frame(Amputation_MCAR_.10_1$amp)
```

```
P1<-function(x) {sum(is.na(x))/length(x)*100}

apply(Amputation_MCAR_.10_1,2,P1) #Gives proportion of missingness in each variable
md.pattern(Amputation_MCAR_.10_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MCAR_.10_1)# Bivariate comparison
aggr_plot <- aggr(Amputation_MCAR_.10_1, col=c('navyblue','yellow'), numbers=TRUE, sortV

#CORRECTING FOR MISSINGNESS: COMPLETE CASE ANALYSIS
set.seed(42)
dim(Amputation_MCAR_.10_1)
str(Amputation_MCAR_.10_1)
complete_cases_0.10_MCAR<-Amputation_MCAR_.10_1[complete.cases(Amputation_MCAR_.10_1),]
View(complete_cases_0.10_MCAR)
dim(complete_cases_0.10_MCAR)
str(complete_cases_0.10_MCAR)
md.pattern(complete_cases_0.10_MCAR)

#ANALYSIS: Complete cases 10% MCAR

# We split the data into:  testing and  training sets.
set.seed(42)
default_idx = sample(nrow(complete_cases_0.10_MCAR), 4533)
default_trn = complete_cases_0.10_MCAR[default_idx, ]
default_tst = complete_cases_0.10_MCAR[-default_idx, ]

model_glm_1 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_1
summary(model_glm_1)

model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_2
summary(model_glm_2)

#obtain the fitted coefficients
coef(model_glm_2)

#Make predictions
head(predict(model_glm_2, type = "link"))

#To obtain the predicted probabilities
head(predict(model_glm_2, type = "response"))

#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the training classificati
calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
}
```

```
#training classification error RATE
calc_class_err(actual = default_trn$HIV_TB_Conifection, predicted = model_glm_pred2)

#We use table() and confusionMatrix() functions to obtain many more metrics.
train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")

c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])

#ROC Curves
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)

################################################################################################
#1.)  Creating 30% Missingness Under MAR
rm(list = ls())
set.seed(100)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
attach(Default)
Amputation_MAR_.30_1<-ampute(Default2, prop = 0.30,freq = NULL, mech = "MAR",std = TRUE,
Amputation_MAR_.30_1

#ASSESSING MISSINGNESS
Amputation_MAR_.30_1$amp
str(Amputation_MAR_.30_1)
Amputation_MAR_.30_1$patterns
Amputation_MAR_.30_1$weights
Amputation_MAR_.30_1$prop

Amputation_MAR_.30_1<-as.data.frame(Amputation_MAR_.30_1$amp)
P1<-function(x) {sum(is.na(x))/length(x)*100}

apply(Amputation_MAR_.30_1,2,P1) #Gives proportion of missingness in each variable
md.pattern(Amputation_MAR_.30_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MAR_.30_1)# Bivariate comparison
```

```
aggr_plot <- aggr(Amputation_MAR_.30_1, col=c('navyblue','yellow'), numbers=TRUE, sortVa

#CORRECTING FOR MISSINGNESS: COMPLETE CASE ANALYSIS
set.seed(42)
dim(Amputation_MAR_.30_1)
str(Amputation_MAR_.30_1)
complete_cases_0.30_MAR<-Amputation_MAR_.30_1[complete.cases(Amputation_MAR_.30_1),]
View(complete_cases_0.30_MAR)
dim(complete_cases_0.30_MAR)
str(complete_cases_0.30_MAR)
md.pattern(complete_cases_0.30_MAR)

#CONVERTING BACK TO ORIGINAL DATA TYPES

library(tibble)
as_tibble(complete_cases_0.30_MAR)

complete_cases_0.30_MAR<-data.frame(complete_cases_0.30_MAR)
library(dplyr)

DefaultStdConverted_MAR_.30<-complete_cases_0.30_MAR
DefaultStdConverted_MAR_.30
dim(DefaultStdConverted_MAR_.30)
str(DefaultStdConverted_MAR_.30)

##ANALYSIS: Complete cases 30% MAR
# We split the data into:  testing and  training sets.
set.seed(42)
default_idx = sample(nrow(DefaultStdConverted_MAR_.30), 6600)
default_trn = DefaultStdConverted_MAR_.30[default_idx, ]
default_tst = DefaultStdConverted_MAR_.30[-default_idx, ]

model_glm_1 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_1
summary(model_glm_1)

model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_2
summary(model_glm_2)

#obtain the fitted coefficients
coef(model_glm_2)

#Make predictions
head(predict(model_glm_2, type = "link"))

#To obtain the predicted probabilities
head(predict(model_glm_2, type = "response"))

#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
```

```
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the training classificati
calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
}

#training classification error RATE
calc_class_err(actual = default_trn$HIV_TB_Conifection, predicted = model_glm_pred2)

#We use table() and confusionMatrix() functions to obtain many more metrics.
train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")

c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])

#ROC Curves
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)

#2.)  Creating 30% Missingness Under MNAR
rm(list = ls())
set.seed(100)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
attach(Default2)
Amputation_MNAR_.30_1<-ampute(Default2, prop = 0.30,freq = NULL, mech = "MNAR",std = TRU
Amputation_MNAR_.30_1

#ASSESSING MISSINGNESS
Amputation_MNAR_.30_1$amp
str(Amputation_MNAR_.30_1)
Amputation_MNAR_.30_1$patterns
Amputation_MNAR_.30_1$weights
Amputation_MNAR_.30_1$prop
```

```
Amputation_MNAR_.30_1<-as.data.frame(Amputation_MNAR_.30_1$amp)
P1<-function(x) {sum(is.na(x))/length(x)*100}

apply(Amputation_MNAR_.30_1,2,P1) #Gives proportion of missingness in each variable
md.pattern(Amputation_MNAR_.30_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MNAR_.30_1)# Bivariate comparison
aggr_plot <- aggr(Amputation_MNAR_.30_1, col=c('navyblue','yellow'), numbers=TRUE, sortV

#CORRECTING FOR MISSINGNESS: COMPLETE CASE ANALYSIS
set.seed(42)
dim(Amputation_MNAR_.30_1)
str(Amputation_MNAR_.30_1)
complete_cases_0.30_MNAR<-Amputation_MNAR_.30_1[complete.cases(Amputation_MNAR_.30_1),]
View(complete_cases_0.30_MNAR)
dim(complete_cases_0.30_MNAR)
str(complete_cases_0.30_MNAR)
md.pattern(complete_cases_0.30_MNAR)

#CONVERTING BACK TO ORIGINAL DATA TYPES
library(tibble)
as_tibble(complete_cases_0.30_MNAR)

complete_cases_0.30_MNAR<-data.frame(complete_cases_0.30_MNAR)
library(dplyr)

DefaultStdConverted_MNAR_.30<-complete_cases_0.30_MNAR
DefaultStdConverted_MNAR_.30
dim(DefaultStdConverted_MNAR_.30)
str(DefaultStdConverted_MNAR_.30)

##ANALYSIS: Complete cases 30% MNAR
# We split the data into:  testing and  training sets.
set.seed(42)
default_idx = sample(nrow(DefaultStdConverted_MNAR_.30), 6600)
default_trn = DefaultStdConverted_MNAR_.30[default_idx, ]
default_tst = DefaultStdConverted_MNAR_.30[-default_idx, ]

model_glm_1 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_1
summary(model_glm_1)

model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_2
summary(model_glm_2)

#obtain the fitted coefficients
coef(model_glm_2)

#Make predictions
head(predict(model_glm_2, type = "link"))
```

```
#To obtain the predicted probabilities
head(predict(model_glm_2, type = "response"))

#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the training classificati
calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
}

#training classification error RATE
calc_class_err(actual = default_trn$HIV_TB_Conifection, predicted = model_glm_pred2)

#We use table() and confusionMatrix() functions to obtain many more metrics.
train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")

c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])

#ROC Curves
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)

#3.)  Creating 30% Missingness Under MCAR
rm(list = ls())
set.seed(100)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
attach(Default2)
Amputation_MCAR_.30_1<-ampute(Default2, prop = 0.30,freq = NULL, mech = "MCAR",std = TRU
Amputation_MCAR_.30_1

#ASSESSING MISSINGNESS
Amputation_MCAR_.30_1$amp
```

```
str(Amputation_MCAR_.30_1)
Amputation_MCAR_.30_1$patterns
Amputation_MCAR_.30_1$weights
Amputation_MCAR_.30_1$prop

Amputation_MCAR_.30_1<-as.data.frame(Amputation_MCAR_.30_1$amp)
P1<-function(x) {sum(is.na(x))/length(x)*100}

apply(Amputation_MCAR_.30_1,2,P1) #Gives proportion of missingness in each variable
md.pattern(Amputation_MCAR_.30_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MCAR_.30_1)# Bivariate comparison
aggr_plot <- aggr(Amputation_MCAR_.30_1, col=c('navyblue','yellow'), numbers=TRUE, sortV

#CORRECTING FOR MISSINGNESS: COMPLETE CASE ANALYSIS
set.seed(42)
dim(Amputation_MCAR_.30_1)
str(Amputation_MCAR_.30_1)
complete_cases_0.30_MCAR<-Amputation_MCAR_.30_1[complete.cases(Amputation_MCAR_.30_1),]
View(complete_cases_0.30_MCAR)
dim(complete_cases_0.30_MCAR)
str(complete_cases_0.30_MCAR)
md.pattern(complete_cases_0.30_MCAR)

#ANALYSIS: Complete cases 30% MCAR
# We split the data into:  testing and  training sets.
set.seed(42)
default_idx = sample(nrow(complete_cases_0.30_MCAR), 3515)
default_trn = complete_cases_0.30_MCAR[default_idx, ]
default_tst = complete_cases_0.30_MCAR[-default_idx, ]

model_glm_1 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_1
summary(model_glm_1)

model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_2
summary(model_glm_2)

#obtain the fitted coefficients
coef(model_glm_2)
#Make predictions
head(predict(model_glm_2, type = "link"))
#To obtain the predicted probabilities
head(predict(model_glm_2, type = "response"))
#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the training classificati
calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
```

```
}

#training classification error RATE
calc_class_err(actual = default_trn$HIV_TB_Conifection, predicted = model_glm_pred2)

#We use table() and confusionMatrix() functions to obtain many more metrics.
train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")

c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])
#ROC Curves
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)


###50%#############################################################################
#1.)  Creating 50% Missingness Under MAR
rm(list = ls())
set.seed(100)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
attach(Default)
Amputation_MAR_.50_1<-ampute(Default2, prop = 0.50,freq = NULL, mech = "MAR",std = TRUE,
Amputation_MAR_.50_1

#ASSESSING MISSINGNESS
Amputation_MAR_.50_1$amp
str(Amputation_MAR_.50_1)
Amputation_MAR_.50_1$patterns
Amputation_MAR_.50_1$weights
Amputation_MAR_.50_1$prop

Amputation_MAR_.50_1<-as.data.frame(Amputation_MAR_.50_1$amp)
P1<-function(x) {sum(is.na(x))/length(x)*100}

apply(Amputation_MAR_.50_1,2,P1) #Gives proportion of missingness in each variable
```

```
md.pattern(Amputation_MAR_.50_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MAR_.50_1)# Bivariate comparison
aggr_plot <- aggr(Amputation_MAR_.50_1, col=c('navyblue','yellow'), numbers=TRUE, sortVa

#CORRECTING FOR MISSINGNESS: COMPLETE CASE ANALYSIS
set.seed(42)
dim(Amputation_MAR_.50_1)
str(Amputation_MAR_.50_1)
complete_cases_0.50_MAR<-Amputation_MAR_.50_1[complete.cases(Amputation_MAR_.50_1),]
View(complete_cases_0.50_MAR)
dim(complete_cases_0.50_MAR)
str(complete_cases_0.50_MAR)
md.pattern(complete_cases_0.50_MAR)

#CONVERTING BACK TO ORIGINAL DATA TYPES
library(tibble)
as_tibble(complete_cases_0.50_MAR)

complete_cases_0.50_MAR<-data.frame(complete_cases_0.50_MAR)
library(dplyr)

DefaultStdConverted_MAR_.50<-complete_cases_0.50_MAR
DefaultStdConverted_MAR_.50
dim(DefaultStdConverted_MAR_.50)
str(DefaultStdConverted_MAR_.50)

##ANALYSIS: Complete cases 50% MAR
# We split the data into:  testing and  training sets.
set.seed(42)
default_idx = sample(nrow(DefaultStdConverted_MAR_.50), 3000)
default_trn = DefaultStdConverted_MAR_.50[default_idx, ]
default_tst = DefaultStdConverted_MAR_.50[-default_idx, ]

model_glm_1 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_1
summary(model_glm_1)

model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_2
summary(model_glm_2)

#obtain the fitted coefficients
coef(model_glm_2)

#Make predictions
head(predict(model_glm_2, type = "link"))

#To obtain the predicted probabilities
head(predict(model_glm_2, type = "response"))

#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
```

```
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the training classificati
calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
}

#training classification error RATE
calc_class_err(actual = default_trn$HIV_TB_Conifection, predicted = model_glm_pred2)

#We use table() and confusionMatrix() functions to obtain many more metrics.
train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")

c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])

#ROC Curves
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)

#2.)  Creating 50% Missingness Under MNAR
rm(list = ls())
set.seed(100)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
attach(Default2)
Amputation_MNAR_.50_1<-ampute(Default2, prop = 0.50,freq = NULL, mech = "MNAR",std = TRU
Amputation_MNAR_.50_1

#ASSESSING MISSINGNESS
Amputation_MNAR_.50_1$amp
str(Amputation_MNAR_.50_1)
Amputation_MNAR_.50_1$patterns
Amputation_MNAR_.50_1$weights
Amputation_MNAR_.50_1$prop
```

```
Amputation_MNAR_.50_1<-as.data.frame(Amputation_MNAR_.50_1$amp)
P1<-function(x) {sum(is.na(x))/length(x)*100}

apply(Amputation_MNAR_.50_1,2,P1) #Gives proportion of missingness in each variable
md.pattern(Amputation_MNAR_.50_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MNAR_.50_1)# Bivariate comparison
aggr_plot <- aggr(Amputation_MNAR_.50_1, col=c('navyblue','yellow'), numbers=TRUE, sortV

#CORRECTING FOR MISSINGNESS: COMPLETE CASE ANALYSIS
set.seed(42)
dim(Amputation_MNAR_.50_1)
str(Amputation_MNAR_.50_1)
complete_cases_0.50_MNAR<-Amputation_MNAR_.50_1[complete.cases(Amputation_MNAR_.50_1),]
View(complete_cases_0.50_MNAR)
dim(complete_cases_0.50_MNAR)
str(complete_cases_0.50_MNAR)
md.pattern(complete_cases_0.50_MNAR)

#CONVERTING BACK TO ORIGINAL DATA TYPES
library(tibble)
as_tibble(complete_cases_0.50_MNAR)

complete_cases_0.50_MNAR<-data.frame(complete_cases_0.50_MNAR)
library(dplyr)

DefaultStdConverted_MNAR_.50<-complete_cases_0.50_MNAR
DefaultStdConverted_MNAR_.50
dim(DefaultStdConverted_MNAR_.50)
str(DefaultStdConverted_MNAR_.50)

##ANALYSIS: Complete cases 50% MNAR
# We split the data into:  testing and  training sets.
set.seed(42)
default_idx = sample(nrow(DefaultStdConverted_MNAR_.50), 3000)
default_trn = DefaultStdConverted_MNAR_.50[default_idx, ]
default_tst = DefaultStdConverted_MNAR_.50[-default_idx, ]

model_glm_1 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_1
summary(model_glm_1)

model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_2
summary(model_glm_2)

#obtain the fitted coefficients
coef(model_glm_2)

#Make predictions
head(predict(model_glm_2, type = "link"))
```

```
#To obtain the predicted probabilities
head(predict(model_glm_2, type = "response"))

#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the training classificati
calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
}

#training classification error RATE
calc_class_err(actual = default_trn$HIV_TB_Conifection, predicted = model_glm_pred2)

#We use table() and confusionMatrix() functions to obtain many more metrics.
train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")

c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])

#ROC Curves
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)

#3.)  Creating 50% Missingness Under MCAR
rm(list = ls())
set.seed(100)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
attach(Default2)
Amputation_MCAR_.50_1<-ampute(Default2, prop = 0.50,freq = NULL, mech = "MCAR",std = TRU
Amputation_MCAR_.50_1

#ASSESSING MISSINGNESS
```

```
Amputation_MCAR_.50_1$amp
str(Amputation_MCAR_.50_1)
Amputation_MCAR_.50_1$patterns
Amputation_MCAR_.50_1$weights
Amputation_MCAR_.50_1$prop

Amputation_MCAR_.50_1<-as.data.frame(Amputation_MCAR_.50_1$amp)
P1<-function(x) {sum(is.na(x))/length(x)*100}

apply(Amputation_MCAR_.50_1,2,P1) #Gives proportion of missingness in each variable
md.pattern(Amputation_MCAR_.50_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MCAR_.50_1)# Bivariate comparison
aggr_plot <- aggr(Amputation_MCAR_.50_1, col=c('navyblue','yellow'), numbers=TRUE, sortV

#CORRECTING FOR MISSINGNESS: COMPLETE CASE ANALYSIS
set.seed(42)
dim(Amputation_MCAR_.50_1)
str(Amputation_MCAR_.50_1)
complete_cases_0.50_MCAR<-Amputation_MCAR_.50_1[complete.cases(Amputation_MCAR_.50_1),]
View(complete_cases_0.50_MCAR)
dim(complete_cases_0.50_MCAR)
str(complete_cases_0.50_MCAR)
md.pattern(complete_cases_0.50_MCAR)

#ANALYSIS: Complete cases 50% MCAR
# We split the data into:  testing and  training sets.
set.seed(42)
default_idx = sample(nrow(complete_cases_0.50_MCAR), 3515)
default_trn = complete_cases_0.50_MCAR[default_idx, ]
default_tst = complete_cases_0.50_MCAR[-default_idx, ]

model_glm_1 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_1
summary(model_glm_1)

model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_2
summary(model_glm_2)

#obtain the fitted coefficients
coef(model_glm_2)
#Make predictions
head(predict(model_glm_2, type = "link"))
#To obtain the predicted probabilities
head(predict(model_glm_2, type = "response"))
#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the training classificati
calc_class_err = function(actual, predicted) {
```

```
  mean(actual != predicted)
}

#training classification error RATE
calc_class_err(actual = default_trn$HIV_TB_Conifection, predicted = model_glm_pred2)

#We use table() and confusionMatrix() functions to obtain many more metrics.
train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")

c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])
#ROC Curves
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)

###80%#########################################################################
#1.)  Creating 80% Missingness Under MAR
rm(list = ls())
set.seed(100)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
attach(Default)
Amputation_MAR_.80_1<-ampute(Default2, prop = 0.80,freq = NULL, mech = "MAR",std = TRUE,
Amputation_MAR_.80_1

#ASSESSING MISSINGNESS
Amputation_MAR_.80_1$amp
str(Amputation_MAR_.80_1)
Amputation_MAR_.80_1$patterns
Amputation_MAR_.80_1$weights
Amputation_MAR_.80_1$prop

Amputation_MAR_.80_1<-as.data.frame(Amputation_MAR_.80_1$amp)
P1<-function(x) {sum(is.na(x))/length(x)*100}

apply(Amputation_MAR_.80_1,2,P1) #Gives proportion of missingness in each variable
```

```
md.pattern(Amputation_MAR_.80_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MAR_.80_1)# Bivariate comparison
aggr_plot <- aggr(Amputation_MAR_.80_1, col=c('navyblue','yellow'), numbers=TRUE, sortVa

#CORRECTING FOR MISSINGNESS: COMPLETE CASE ANALYSIS
set.seed(42)
dim(Amputation_MAR_.80_1)
str(Amputation_MAR_.80_1)
complete_cases_0.80_MAR<-Amputation_MAR_.80_1[complete.cases(Amputation_MAR_.80_1),]
View(complete_cases_0.80_MAR)
dim(complete_cases_0.80_MAR)
str(complete_cases_0.80_MAR)
md.pattern(complete_cases_0.80_MAR)

#CONVERTING BACK TO ORIGINAL DATA TYPES
library(tibble)
as_tibble(complete_cases_0.80_MAR)

complete_cases_0.80_MAR<-data.frame(complete_cases_0.80_MAR)
library(dplyr)

DefaultStdConverted_MAR_.80<-complete_cases_0.80_MAR
DefaultStdConverted_MAR_.80
dim(DefaultStdConverted_MAR_.80)
str(DefaultStdConverted_MAR_.80)

##ANALYSIS: Complete cases 80% MAR
# We split the data into:  testing and  training sets.
set.seed(42)
default_idx = sample(nrow(DefaultStdConverted_MAR_.80), 1915)
default_trn = DefaultStdConverted_MAR_.80[default_idx, ]
default_tst = DefaultStdConverted_MAR_.80[-default_idx, ]

model_glm_1 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_1
summary(model_glm_1)

model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_2
summary(model_glm_2)

#obtain the fitted coefficients
coef(model_glm_2)

#Make predictions
head(predict(model_glm_2, type = "link"))

#To obtain the predicted probabilities
head(predict(model_glm_2, type = "response"))

#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
```

```
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the training classificati
calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
}

#training classification error RATE
calc_class_err(actual = default_trn$HIV_TB_Conifection, predicted = model_glm_pred2)

#We use table() and confusionMatrix() functions to obtain many more metrics.
train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")

c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])

#ROC Curves
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)

#2.)  Creating 80% Missingness Under MNAR
rm(list = ls())
set.seed(100)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
attach(Default2)
Amputation_MNAR_.80_1<-ampute(Default2, prop = 0.80,freq = NULL, mech = "MNAR",std = TRU
Amputation_MNAR_.80_1

#ASSESSING MISSINGNESS
Amputation_MNAR_.80_1$amp
str(Amputation_MNAR_.80_1)
Amputation_MNAR_.80_1$patterns
Amputation_MNAR_.80_1$weights
Amputation_MNAR_.80_1$prop
```

```
Amputation_MNAR_.80_1<-as.data.frame(Amputation_MNAR_.80_1$amp)
P1<-function(x) {sum(is.na(x))/length(x)*100}

apply(Amputation_MNAR_.80_1,2,P1) #Gives proportion of missingness in each variable
md.pattern(Amputation_MNAR_.80_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MNAR_.80_1)# Bivariate comparison
aggr_plot <- aggr(Amputation_MNAR_.80_1, col=c('navyblue','yellow'), numbers=TRUE, sortV

#CORRECTING FOR MISSINGNESS: COMPLETE CASE ANALYSIS
set.seed(42)
dim(Amputation_MNAR_.80_1)
str(Amputation_MNAR_.80_1)
complete_cases_0.80_MNAR<-Amputation_MNAR_.80_1[complete.cases(Amputation_MNAR_.80_1),]
View(complete_cases_0.80_MNAR)
dim(complete_cases_0.80_MNAR)
str(complete_cases_0.80_MNAR)
md.pattern(complete_cases_0.80_MNAR)

#CONVERTING BACK TO ORIGINAL DATA TYPES
library(tibble)
as_tibble(complete_cases_0.80_MNAR)

complete_cases_0.80_MNAR<-data.frame(complete_cases_0.80_MNAR)
library(dplyr)

DefaultStdConverted_MNAR_.80<-complete_cases_0.80_MNAR
DefaultStdConverted_MNAR_.80
dim(DefaultStdConverted_MNAR_.80)
str(DefaultStdConverted_MNAR_.80)

##ANALYSIS: Complete cases 30% MNAR
# We split the data into:  testing and  training sets.
set.seed(42)
default_idx = sample(nrow(DefaultStdConverted_MNAR_.80), 1800)
default_trn = DefaultStdConverted_MNAR_.80[default_idx, ]
default_tst = DefaultStdConverted_MNAR_.80[-default_idx, ]

model_glm_1 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_1
summary(model_glm_1)

model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_2
summary(model_glm_2)

#obtain the fitted coefficients
coef(model_glm_2)

#Make predictions
head(predict(model_glm_2, type = "link"))
```

```
#To obtain the predicted probabilities
head(predict(model_glm_2, type = "response"))

#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the training classificati
calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
}

#training classification error RATE
calc_class_err(actual = default_trn$HIV_TB_Conifection, predicted = model_glm_pred2)

#We use table() and confusionMatrix() functions to obtain many more metrics.
train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")

c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])

#ROC Curves
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)

#3.)  Creating 80% Missingness Under MCAR
rm(list = ls())
set.seed(100)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
attach(Default2)
Amputation_MCAR_.80_1<-ampute(Default2, prop = 0.80,freq = NULL, mech = "MCAR",std = TRU
Amputation_MCAR_.80_1

#ASSESSING MISSINGNESS
```

```
Amputation_MCAR_.80_1$amp
str(Amputation_MCAR_.80_1)
Amputation_MCAR_.80_1$patterns
Amputation_MCAR_.80_1$weights
Amputation_MCAR_.80_1$prop

Amputation_MCAR_.80_1<-as.data.frame(Amputation_MCAR_.80_1$amp)
P1<-function(x) {sum(is.na(x))/length(x)*100}

apply(Amputation_MCAR_.80_1,2,P1) #Gives proportion of missingness in each variable
md.pattern(Amputation_MCAR_.80_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MCAR_.80_1)# Bivariate comparison
aggr_plot <- aggr(Amputation_MCAR_.80_1, col=c('navyblue','yellow'), numbers=TRUE, sortV

#CORRECTING FOR MISSINGNESS: COMPLETE CASE ANALYSIS
set.seed(42)
dim(Amputation_MCAR_.80_1)
str(Amputation_MCAR_.80_1)
complete_cases_0.80_MCAR<-Amputation_MCAR_.80_1[complete.cases(Amputation_MCAR_.80_1),]
View(complete_cases_0.80_MCAR)
dim(complete_cases_0.80_MCAR)
str(complete_cases_0.80_MCAR)
md.pattern(complete_cases_0.80_MCAR)

#ANALYSIS: Complete cases 80% MCAR
# We split the data into:  testing and  training sets.
set.seed(42)
default_idx = sample(nrow(complete_cases_0.80_MCAR), 1050)
default_trn = complete_cases_0.80_MCAR[default_idx, ]
default_tst = complete_cases_0.80_MCAR[-default_idx, ]

model_glm_1 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_1
summary(model_glm_1)

model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_2
summary(model_glm_2)

#obtain the fitted coefficients
coef(model_glm_2)
#Make predictions
head(predict(model_glm_2, type = "link"))
#To obtain the predicted probabilities
head(predict(model_glm_2, type = "response"))
#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the training classificati
calc_class_err = function(actual, predicted) {
```

```
  mean(actual != predicted)
}

#training classification error RATE
calc_class_err(actual = default_trn$HIV_TB_Conifection, predicted = model_glm_pred2)

#We use table() and confusionMatrix() functions to obtain many more metrics.
train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")

c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])
#ROC Curves
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)

############MEAN/SINGLE IMPUTATION METHOD#####################################%%%%%%%%%

#1.)  Creating 7% Missingness Under MAR
rm(list = ls())
set.seed(100)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
attach(Default)
Amputation_MAR_.07_1<-ampute(Default2, prop = 0.07,freq = NULL, mech = "MAR",std = TRUE,
Amputation_MAR_.07_1

#ASSESSING MISSINGNESS
Amputation_MAR_.07_1$amp
str(Amputation_MAR_.07_1)
Amputation_MAR_.07_1$patterns
Amputation_MAR_.07_1$weights
Amputation_MAR_.07_1$prop

Amputation_MAR_.07_1<-as.data.frame(Amputation_MAR_.07_1$amp)
P1<-function(x) {sum(is.na(x))/length(x)*100}
```

```
apply(Amputation_MAR_.07_1,2,P1) #Gives proportion of missingness in each variable
md.pattern(Amputation_MAR_.07_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MAR_.07_1)# Bivariate comparison
aggr_plot <- aggr(Amputation_MAR_.07_1, col=c('navyblue','yellow'), numbers=TRUE, sortVa

library(Hmisc)
#2.1.a) 7% MAR
str(Amputation_MAR_.07_1)
drops<-c("weight","Age","HIV_TB_Conifection")
drops<-c("weight","Age")
DefaultStd<-Amputation_MAR_.07_1[ , !(names(Amputation_MAR_.07_1) %in% drops)]
str(DefaultStd)
con.names = DefaultStd %>% select_if(is.character) %>% colnames()
print(con.names)
Amputation_MAR_.07_1<-Amputation_MAR_.07_1
Amputation_MAR_.07_1[,con.names] = data.frame(apply(Amputation_MAR_.07_1[con.names], 2,
Amputation_MAR_.07_1
dim(Amputation_MAR_.07_1)
str(Amputation_MAR_.07_1)
summary(Amputation_MAR_.07_1)

Age<-impute(Amputation_MAR_.07_1$Age, mean)  # replace with mean
weight<-impute(Amputation_MAR_.07_1$weight, mean)  # replace with mean
HIV_TB_Conifection<-impute(Amputation_MAR_.07_1$HIV_TB_Conifection, mode)  # replace wit
HIV_TB_Conifection<-factor(HIV_TB_Conifection)
Gender<-impute(Amputation_MAR_.07_1$Gender, mode)  # replace with mode
Gender<-factor(Gender)
Single_imp_Data_MAR_.07<-as.data.frame(cbind(Age,weight,HIV_TB_Conifection,Gender))
cols <- c("Gender")
Single_imp_Data_MAR_.07[,cols] <- data.frame(apply(Single_imp_Data_MAR_.07[cols], 2, as.
Single_imp_Data_MAR_.07$HIV_TB_Conifection[Single_imp_Data_MAR_.07$HIV_TB_Conifection ==
Single_imp_Data_MAR_.07$HIV_TB_Conifection[Single_imp_Data_MAR_.07$HIV_TB_Conifection ==
Single_imp_Data_MAR_.07

dim(Single_imp_Data_MAR_.07)
md.pattern(Single_imp_Data_MAR_.07)
str(Single_imp_Data_MAR_.07)
view(Single_imp_Data_MAR_.07)
#ANALYSIS: MEAN/SINGLE IMPUTATION 7% MAR
# We split the data into:  testing and  training sets.
set.seed(42)
default_idx = sample(nrow(Single_imp_Data_MAR_.07), 5000)
default_trn = Single_imp_Data_MAR_.07[default_idx, ]
default_tst = Single_imp_Data_MAR_.07[-default_idx, ]

model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_2
summary(model_glm_2)

#Make predictions
head(predict(model_glm_2, type = "link"))
```

```
#the predicted probabilities
head(predict(model_glm_2, type = "response"))

#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the trainging classificat
calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
}
calc_class_err(actual = default_trn$default, predicted = model_glm_pred2)

train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
set.seed(100)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")
c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])

#ROC Curves
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)

#2.)  Creating 7% Missingness Under MNAR
rm(list = ls())
set.seed(100)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
attach(Default)
Amputation_MNAR_.07_1<-ampute(Default2, prop = 0.07,freq = NULL, mech = "MNAR",std = TRU
Amputation_MNAR_.07_1

#ASSESSING MISSINGNESS
Amputation_MNAR_.07_1$amp
str(Amputation_MNAR_.07_1)
Amputation_MNAR_.07_1$patterns
Amputation_MNAR_.07_1$weights
```

```
Amputation_MNAR_.07_1$prop

Amputation_MNAR_.07_1<-as.data.frame(Amputation_MNAR_.07_1$amp)
P1<-function(x) {sum(is.na(x))/length(x)*100}

apply(Amputation_MNAR_.07_1,2,P1) #Gives proportion of missingness in each variable
md.pattern(Amputation_MNAR_.07_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MNAR_.07_1)# Bivariate comparison
aggr_plot <- aggr(Amputation_MNAR_.07_1, col=c('navyblue','yellow'), numbers=TRUE, sortV

library(Hmisc)
#2.1.a) 7% MAR
str(Amputation_MNAR_.07_1)
drops<-c("weight","Age","HIV_TB_Conifection")
drops<-c("weight","Age")
DefaultStd<-Amputation_MNAR_.07_1[ , !(names(Amputation_MNAR_.07_1) %in% drops)]
str(DefaultStd)
con.names = DefaultStd %>% select_if(is.character) %>% colnames()
print(con.names)
Amputation_MAR_.07_1<-Amputation_MNAR_.07_1
Amputation_MNAR_.07_1[,con.names] = data.frame(apply(Amputation_MNAR_.07_1[con.names], 2
Amputation_MNAR_.07_1
dim(Amputation_MNAR_.07_1)
str(Amputation_MNAR_.07_1)
summary(Amputation_MNAR_.07_1)

Age<-impute(Amputation_MNAR_.07_1$Age, mean)  # replace with mean
weight<-impute(Amputation_MNAR_.07_1$weight, mean)  # replace with mean
HIV_TB_Conifection<-impute(Amputation_MNAR_.07_1$HIV_TB_Conifection, mode)  # replace wi
HIV_TB_Conifection<-factor(HIV_TB_Conifection)
Gender<-impute(Amputation_MNAR_.07_1$Gender, mode)  # replace with mode
Gender<-factor(Gender)
Single_imp_Data_MNAR_.07<-as.data.frame(cbind(Age,weight,HIV_TB_Conifection,Gender))
cols <- c("Gender")
Single_imp_Data_MNAR_.07[,cols] <- data.frame(apply(Single_imp_Data_MNAR_.07[cols], 2, a
Single_imp_Data_MNAR_.07$HIV_TB_Conifection[Single_imp_Data_MNAR_.07$HIV_TB_Conifection
Single_imp_Data_MNAR_.07$HIV_TB_Conifection[Single_imp_Data_MNAR_.07$HIV_TB_Conifection
Single_imp_Data_MNAR_.07

dim(Single_imp_Data_MNAR_.07)
md.pattern(Single_imp_Data_MNAR_.07)
str(Single_imp_Data_MNAR_.07)
view(Single_imp_Data_MNAR_.07)
#ANALYSIS: MEAN/SINGLE IMPUTATION 7% MNAR
# We split the data into:  testing and  training sets.
set.seed(42)
default_idx = sample(nrow(Single_imp_Data_MNAR_.07), 5000)
default_trn = Single_imp_Data_MNAR_.07[default_idx, ]
default_tst = Single_imp_Data_MNAR_.07[-default_idx, ]

model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
```

```
model_glm_2
summary(model_glm_2)

#Make predictions
head(predict(model_glm_2, type = "link"))
#the predicted probabilities
head(predict(model_glm_2, type = "response"))

#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the trainging classificat
calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
}
calc_class_err(actual = default_trn$default, predicted = model_glm_pred2)

train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
set.seed(100)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")
c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])

#ROC Curves
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)

#3.)  Creating 7% Missingness Under MCAR
rm(list = ls())
set.seed(100)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
attach(Default)
Amputation_MCAR_.07_1<-ampute(Default2, prop = 0.07,freq = NULL, mech = "MCAR",std = TRU
Amputation_MCAR_.07_1
```

```
#ASSESSING MISSINGNESS
Amputation_MCAR_.07_1$amp
str(Amputation_MCAR_.07_1)
Amputation_MCAR_.07_1$patterns
Amputation_MCAR_.07_1$weights
Amputation_MCAR_.07_1$prop

Amputation_MCAR_.07_1<-as.data.frame(Amputation_MCAR_.07_1$amp)
P1<-function(x) {sum(is.na(x))/length(x)*100}

apply(Amputation_MCAR_.07_1,2,P1) #Gives proportion of missingness in each variable
md.pattern(Amputation_MCAR_.07_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MCAR_.07_1)# Bivariate comparison
aggr_plot <- aggr(Amputation_MCAR_.07_1, col=c('navyblue','yellow'), numbers=TRUE, sortV

library(Hmisc)
#2.1.a) 7% MAR
str(Amputation_MCAR_.07_1)
drops<-c("weight","Age","HIV_TB_Conifection")
drops<-c("weight","Age")
DefaultStd<-Amputation_MCAR_.07_1[ , !(names(Amputation_MCAR_.07_1) %in% drops)]
str(DefaultStd)
con.names = DefaultStd %>% select_if(is.character) %>% colnames()
print(con.names)
Amputation_MCAR_.07_1<-Amputation_MCAR_.07_1
Amputation_MCAR_.07_1[,con.names] = data.frame(apply(Amputation_MCAR_.07_1[con.names], 2
Amputation_MCAR_.07_1
dim(Amputation_MCAR_.07_1)
str(Amputation_MCAR_.07_1)
summary(Amputation_MCAR_.07_1)

Age<-impute(Amputation_MCAR_.07_1$Age, mean)  # replace with mean
weight<-impute(Amputation_MCAR_.07_1$weight, mean)  # replace with mean
HIV_TB_Conifection<-impute(Amputation_MCAR_.07_1$HIV_TB_Conifection, mode)  # replace wi
HIV_TB_Conifection<-factor(HIV_TB_Conifection)
Gender<-impute(Amputation_MCAR_.07_1$Gender, mode)  # replace with mode
Gender<-factor(Gender)
Single_imp_Data_MCAR_.07<-as.data.frame(cbind(Age,weight,HIV_TB_Conifection,Gender))
cols <- c("Gender")
Single_imp_Data_MCAR_.07[,cols] <- data.frame(apply(Single_imp_Data_MCAR_.07[cols], 2, a
Single_imp_Data_MCAR_.07$HIV_TB_Conifection[Single_imp_Data_MCAR_.07$HIV_TB_Conifection
Single_imp_Data_MCAR_.07$HIV_TB_Conifection[Single_imp_Data_MCAR_.07$HIV_TB_Conifection
Single_imp_Data_MCAR_.07

dim(Single_imp_Data_MCAR_.07)
md.pattern(Single_imp_Data_MCAR_.07)
str(Single_imp_Data_MCAR_.07)
view(Single_imp_Data_MCAR_.07)
#ANALYSIS: MEAN/SINGLE IMPUTATION 7% MCAR
# We split the data into:  testing and  training sets.
set.seed(42)
```

```
default_idx = sample(nrow(Single_imp_Data_MCAR_.07), 5000)
default_trn = Single_imp_Data_MCAR_.07[default_idx, ]
default_tst = Single_imp_Data_MCAR_.07[-default_idx, ]

model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_2
summary(model_glm_2)

#Make predictions
head(predict(model_glm_2, type = "link"))
#the predicted probabilities
head(predict(model_glm_2, type = "response"))

#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the trainging classificat
calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
}
calc_class_err(actual = default_trn$default, predicted = model_glm_pred2)

train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
set.seed(100)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")
c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])

#ROC Curves
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)

##########################10%
#1.)  Creating 10% Missingness Under MAR
rm(list = ls())
set.seed(100)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
```

```
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
attach(Default)
Amputation_MAR_.10_1<-ampute(Default2, prop = 0.1,freq = NULL, mech = "MAR",std = TRUE,
Amputation_MAR_.10_1

#ASSESSING MISSINGNESS
Amputation_MAR_.10_1$amp
str(Amputation_MAR_.07_1)
Amputation_MAR_.10_1$patterns
Amputation_MAR_.10_1$weights
Amputation_MAR_.10_1$prop

Amputation_MAR_.10_1<-as.data.frame(Amputation_MAR_.10_1$amp)
P1<-function(x) {sum(is.na(x))/length(x)*100}

apply(Amputation_MAR_.10_1,2,P1) #Gives proportion of missingness in each variable
md.pattern(Amputation_MAR_.10_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MAR_.10_1)# Bivariate comparison
aggr_plot <- aggr(Amputation_MAR_.10_1, col=c('navyblue','yellow'), numbers=TRUE, sortVa

library(Hmisc)
#2.1.a) 7% MAR
str(Amputation_MAR_.10_1)
drops<-c("weight","Age","HIV_TB_Conifection")
drops<-c("weight","Age")
DefaultStd<-Amputation_MAR_.10_1[ , !(names(Amputation_MAR_.10_1) %in% drops)]
str(DefaultStd)
con.names = DefaultStd %>% select_if(is.character) %>% colnames()
print(con.names)
Amputation_MAR_.10_1<-Amputation_MAR_.10_1
Amputation_MAR_.10_1[,con.names] = data.frame(apply(Amputation_MAR_.10_1[con.names], 2,
Amputation_MAR_.10_1
dim(Amputation_MAR_.10_1)
str(Amputation_MAR_.10_1)
summary(Amputation_MAR_.10_1)

Age<-impute(Amputation_MAR_.10_1$Age, mean)  # replace with mean
weight<-impute(Amputation_MAR_.10_1$weight, mean)  # replace with mean
HIV_TB_Conifection<-impute(Amputation_MAR_.10_1$HIV_TB_Conifection, mode)  # replace wit
HIV_TB_Conifection<-factor(HIV_TB_Conifection)
Gender<-impute(Amputation_MAR_.10_1$Gender, mode)  # replace with mode
Gender<-factor(Gender)
Single_imp_Data_MAR_.10<-as.data.frame(cbind(Age,weight,HIV_TB_Conifection,Gender))
cols <- c("Gender")
Single_imp_Data_MAR_.10[,cols] <- data.frame(apply(Single_imp_Data_MAR_.10[cols], 2, as.
Single_imp_Data_MAR_.10$HIV_TB_Conifection[Single_imp_Data_MAR_.10$HIV_TB_Conifection ==
Single_imp_Data_MAR_.10$HIV_TB_Conifection[Single_imp_Data_MAR_.10$HIV_TB_Conifection ==
Single_imp_Data_MAR_.10

dim(Single_imp_Data_MAR_.10)
```

```
md.pattern(Single_imp_Data_MAR_.10)
str(Single_imp_Data_MAR_.10)
view(Single_imp_Data_MAR_.10)
#ANALYSIS: MEAN/SINGLE IMPUTATION 10% MAR
# We split the data into:  testing and  training sets.
set.seed(42)
default_idx = sample(nrow(Single_imp_Data_MAR_.10), 5000)
default_trn = Single_imp_Data_MAR_.10[default_idx, ]
default_tst = Single_imp_Data_MAR_.10[-default_idx, ]

model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_2
summary(model_glm_2)

#Make predictions
head(predict(model_glm_2, type = "link"))
#the predicted probabilities
head(predict(model_glm_2, type = "response"))

#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the trainging classificat
calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
}
calc_class_err(actual = default_trn$default, predicted = model_glm_pred2)

train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
set.seed(100)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")
c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])

#ROC Curves
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)

#2.)  Creating 10% Missingness Under MNAR
rm(list = ls())
set.seed(100)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
```

```
weight<-round(runif(10000, 45, 100))
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
attach(Default)
Amputation_MNAR_.10_1<-ampute(Default2, prop = 0.10,freq = NULL, mech = "MNAR",std = TRU
Amputation_MNAR_.10_1

#ASSESSING MISSINGNESS
Amputation_MNAR_.10_1$amp
str(Amputation_MNAR_.10_1)
Amputation_MNAR_.10_1$patterns
Amputation_MNAR_.10_1$weights
Amputation_MNAR_.10_1$prop

Amputation_MNAR_.10_1<-as.data.frame(Amputation_MNAR_.10_1$amp)
P1<-function(x) {sum(is.na(x))/length(x)*100}

apply(Amputation_MNAR_.10_1,2,P1) #Gives proportion of missingness in each variable
md.pattern(Amputation_MNAR_.10_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MNAR_.10_1)# Bivariate comparison
aggr_plot <- aggr(Amputation_MNAR_.10_1, col=c('navyblue','yellow'), numbers=TRUE, sortV

library(Hmisc)
#2.1.a) 10% MNAR
str(Amputation_MNAR_.10_1)
drops<-c("weight","Age","HIV_TB_Conifection")
drops<-c("weight","Age")
DefaultStd<-Amputation_MNAR_.10_1[ , !(names(Amputation_MNAR_.10_1) %in% drops)]
str(DefaultStd)
con.names = DefaultStd %>% select_if(is.character) %>% colnames()
print(con.names)
Amputation_MAR_.10_1<-Amputation_MNAR_.10_1
Amputation_MNAR_.10_1[,con.names] = data.frame(apply(Amputation_MNAR_.10_1[con.names], 2
Amputation_MNAR_.10_1
dim(Amputation_MNAR_.10_1)
str(Amputation_MNAR_.10_1)
summary(Amputation_MNAR_.10_1)

Age<-impute(Amputation_MNAR_.10_1$Age, mean)  # replace with mean
weight<-impute(Amputation_MNAR_.10_1$weight, mean)  # replace with mean
HIV_TB_Conifection<-impute(Amputation_MNAR_.10_1$HIV_TB_Conifection, mode)  # replace wi
HIV_TB_Conifection<-factor(HIV_TB_Conifection)
Gender<-impute(Amputation_MNAR_.10_1$Gender, mode)  # replace with mode
Gender<-factor(Gender)
Single_imp_Data_MNAR_.10<-as.data.frame(cbind(Age,weight,HIV_TB_Conifection,Gender))
cols <- c("Gender")
Single_imp_Data_MNAR_.10[,cols] <- data.frame(apply(Single_imp_Data_MNAR_.10[cols], 2, a
```

```
Single_imp_Data_MNAR_.10$HIV_TB_Conifection[Single_imp_Data_MNAR_.10$HIV_TB_Conifection
Single_imp_Data_MNAR_.10$HIV_TB_Conifection[Single_imp_Data_MNAR_.10$HIV_TB_Conifection
Single_imp_Data_MNAR_.10

dim(Single_imp_Data_MNAR_.10)
md.pattern(Single_imp_Data_MNAR_.10)
str(Single_imp_Data_MNAR_.10)
view(Single_imp_Data_MNAR_.10)
#ANALYSIS: MEAN/SINGLE IMPUTATION 10% MNAR
# We split the data into:  testing and  training sets.
set.seed(42)
default_idx = sample(nrow(Single_imp_Data_MNAR_.10), 5000)
default_trn = Single_imp_Data_MNAR_.10[default_idx, ]
default_tst = Single_imp_Data_MNAR_.10[-default_idx, ]

model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_2
summary(model_glm_2)

#Make predictions
head(predict(model_glm_2, type = "link"))
#the predicted probabilities
head(predict(model_glm_2, type = "response"))

#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the trainging classificat
calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
}
calc_class_err(actual = default_trn$default, predicted = model_glm_pred2)

train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
set.seed(100)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")
c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])

#ROC Curves
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)
```

```
#3.)  Creating 10% Missingness Under MCAR
rm(list = ls())
set.seed(100)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
attach(Default2)
Amputation_MCAR_.10_1<-ampute(Default2, prop = 0.10,freq = NULL, mech = "MCAR",std = TRU
Amputation_MCAR_.10_1

#ASSESSING MISSINGNESS
Amputation_MCAR_.10_1$amp
str(Amputation_MCAR_.10_1)
Amputation_MCAR_.10_1$patterns
Amputation_MCAR_.10_1$weights
Amputation_MCAR_.10_1$prop

Amputation_MCAR_.10_1<-as.data.frame(Amputation_MCAR_.10_1$amp)
P1<-function(x) {sum(is.na(x))/length(x)*100}

apply(Amputation_MCAR_.10_1,2,P1) #Gives proportion of missingness in each variable
md.pattern(Amputation_MCAR_.10_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MCAR_.10_1)# Bivariate comparison
aggr_plot <- aggr(Amputation_MCAR_.10_1, col=c('navyblue','yellow'), numbers=TRUE, sortV

library(Hmisc)
#2.1.a) 7% MAR
str(Amputation_MCAR_.10_1)
drops<-c("weight","Age","HIV_TB_Conifection")
drops<-c("weight","Age")
DefaultStd<-Amputation_MCAR_.10_1[ , !(names(Amputation_MCAR_.10_1) %in% drops)]
str(DefaultStd)
con.names = DefaultStd %>% select_if(is.character) %>% colnames()
print(con.names)
Amputation_MCAR_.10_1<-Amputation_MCAR_.10_1
Amputation_MCAR_.10_1[,con.names] = data.frame(apply(Amputation_MCAR_.10_1[con.names], 2
Amputation_MCAR_.10_1
dim(Amputation_MCAR_.10_1)
str(Amputation_MCAR_.10_1)
summary(Amputation_MCAR_.10_1)

Age<-impute(Amputation_MCAR_.10_1$Age, mean)  # replace with mean
weight<-impute(Amputation_MCAR_.10_1$weight, mean)  # replace with mean
HIV_TB_Conifection<-impute(Amputation_MCAR_.10_1$HIV_TB_Conifection, mode)  # replace wi
HIV_TB_Conifection<-factor(HIV_TB_Conifection)
```

```
Gender<-impute(Amputation_MCAR_.10_1$Gender, mode)  # replace with mode
Gender<-factor(Gender)
Single_imp_Data_MCAR_.10<-as.data.frame(cbind(Age,weight,HIV_TB_Conifection,Gender))
cols <- c("Gender")
Single_imp_Data_MCAR_.10[,cols] <- data.frame(apply(Single_imp_Data_MCAR_.10[cols], 2, a
Single_imp_Data_MCAR_.10$HIV_TB_Conifection[Single_imp_Data_MCAR_.10$HIV_TB_Conifection
Single_imp_Data_MCAR_.10$HIV_TB_Conifection[Single_imp_Data_MCAR_.10$HIV_TB_Conifection
Single_imp_Data_MCAR_.10

dim(Single_imp_Data_MCAR_.10)
md.pattern(Single_imp_Data_MCAR_.10)
str(Single_imp_Data_MCAR_.10)
view(Single_imp_Data_MCAR_.10)
#ANALYSIS: MEAN/SINGLE IMPUTATION 10% McAR
# We split the data into:  testing and  training sets.
set.seed(42)
default_idx = sample(nrow(Single_imp_Data_MCAR_.10), 5000)
default_trn = Single_imp_Data_MCAR_.10[default_idx, ]
default_tst = Single_imp_Data_MCAR_.10[-default_idx, ]

model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_2
summary(model_glm_2)

#Make predictions
head(predict(model_glm_2, type = "link"))
#the predicted probabilities
head(predict(model_glm_2, type = "response"))

#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the trainging classificat
calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
}
calc_class_err(actual = default_trn$default, predicted = model_glm_pred2)

train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
set.seed(100)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")
c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])

#ROC Curves
```

```
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)

##30%!!!!!!!!!!!!!!!!
#1.)  Creating 30% Missingness Under MAR
rm(list = ls())
set.seed(100)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
attach(Default)
Amputation_MAR_.30_1<-ampute(Default2, prop = 0.3,freq = NULL, mech = "MAR",std = TRUE,
Amputation_MAR_.30_1

#ASSESSING MISSINGNESS
Amputation_MAR_.30_1$amp
str(Amputation_MAR_.30_1)
Amputation_MAR_.30_1$patterns
Amputation_MAR_.30_1$weights
Amputation_MAR_.30_1$prop

Amputation_MAR_.30_1<-as.data.frame(Amputation_MAR_.30_1$amp)
P1<-function(x) {sum(is.na(x))/length(x)*100}

apply(Amputation_MAR_.30_1,2,P1) #Gives proportion of missingness in each variable
md.pattern(Amputation_MAR_.30_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MAR_.30_1)# Bivariate comparison
aggr_plot <- aggr(Amputation_MAR_.30_1, col=c('navyblue','yellow'), numbers=TRUE, sortVa

library(Hmisc)
#2.1.a) 30% MAR
str(Amputation_MAR_.30_1)
drops<-c("weight","Age","HIV_TB_Conifection")
drops<-c("weight","Age")
DefaultStd<-Amputation_MAR_.30_1[ , !(names(Amputation_MAR_.30_1) %in% drops)]
str(DefaultStd)
con.names = DefaultStd %>% select_if(is.character) %>% colnames()
print(con.names)
Amputation_MAR_.30_1<-Amputation_MAR_.30_1
Amputation_MAR_.30_1[,con.names] = data.frame(apply(Amputation_MAR_.30_1[con.names], 2,
Amputation_MAR_.30_1
dim(Amputation_MAR_.30_1)
str(Amputation_MAR_.30_1)
```

```
summary(Amputation_MAR_.30_1)

Age<-impute(Amputation_MAR_.30_1$Age, mean)  # replace with mean
weight<-impute(Amputation_MAR_.30_1$weight, mean)  # replace with mean
HIV_TB_Conifection<-impute(Amputation_MAR_.30_1$HIV_TB_Conifection, mode)  # replace wit
HIV_TB_Conifection<-factor(HIV_TB_Conifection)
Gender<-impute(Amputation_MAR_.30_1$Gender, mode)  # replace with mode
Gender<-factor(Gender)
Single_imp_Data_MAR_.30<-as.data.frame(cbind(Age,weight,HIV_TB_Conifection,Gender))
cols <- c("Gender")
Single_imp_Data_MAR_.30[,cols] <- data.frame(apply(Single_imp_Data_MAR_.30[cols], 2, as.
Single_imp_Data_MAR_.30$HIV_TB_Conifection[Single_imp_Data_MAR_.30$HIV_TB_Conifection ==
Single_imp_Data_MAR_.30$HIV_TB_Conifection[Single_imp_Data_MAR_.30$HIV_TB_Conifection ==
Single_imp_Data_MAR_.30

dim(Single_imp_Data_MAR_.30)
md.pattern(Single_imp_Data_MAR_.30)
str(Single_imp_Data_MAR_.30)
view(Single_imp_Data_MAR_.30)
#ANALYSIS: MEAN/SINGLE IMPUTATION 30% MAR
# We split the data into:  testing and  training sets.
set.seed(42)
default_idx = sample(nrow(Single_imp_Data_MAR_.30), 5000)
default_trn = Single_imp_Data_MAR_.30[default_idx, ]
default_tst = Single_imp_Data_MAR_.30[-default_idx, ]

model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_2
summary(model_glm_2)

#Make predictions
head(predict(model_glm_2, type = "link"))
#the predicted probabilities
head(predict(model_glm_2, type = "response"))

#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the trainging classificat
calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
}
calc_class_err(actual = default_trn$default, predicted = model_glm_pred2)

train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
set.seed(100)
```

```
train_con_mat = confusionMatrix(train_tab, positive = "Yes")
c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])

#ROC Curves
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)

#2.)  Creating 30% Missingness Under MNAR
rm(list = ls())
set.seed(100)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
attach(Default)
Amputation_MNAR_.30_1<-ampute(Default2, prop = 0.30,freq = NULL, mech = "MNAR",std = TRU
Amputation_MNAR_.30_1

#ASSESSING MISSINGNESS
Amputation_MNAR_.30_1$amp
str(Amputation_MNAR_.30_1)
Amputation_MNAR_.30_1$patterns
Amputation_MNAR_.30_1$weights
Amputation_MNAR_.30_1$prop

Amputation_MNAR_.30_1<-as.data.frame(Amputation_MNAR_.30_1$amp)
P1<-function(x) {sum(is.na(x))/length(x)*100}

apply(Amputation_MNAR_.30_1,2,P1) #Gives proportion of missingness in each variable
md.pattern(Amputation_MNAR_.30_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MNAR_.30_1)# Bivariate comparison
aggr_plot <- aggr(Amputation_MNAR_.30_1, col=c('navyblue','yellow'), numbers=TRUE, sortV

library(Hmisc)
#2.1.a) 30% MNAR
str(Amputation_MNAR_.30_1)
drops<-c("weight","Age","HIV_TB_Conifection")
drops<-c("weight","Age")
DefaultStd<-Amputation_MNAR_.30_1[ , !(names(Amputation_MNAR_.30_1) %in% drops)]
str(DefaultStd)
con.names = DefaultStd %>% select_if(is.character) %>% colnames()
print(con.names)
```

```
Amputation_MAR_.30_1<-Amputation_MNAR_.30_1
Amputation_MNAR_.30_1[,con.names] = data.frame(apply(Amputation_MNAR_.30_1[con.names], 2
Amputation_MNAR_.30_1
dim(Amputation_MNAR_.30_1)
str(Amputation_MNAR_.30_1)
summary(Amputation_MNAR_.30_1)

Age<-impute(Amputation_MNAR_.30_1$Age, mean)  # replace with mean
weight<-impute(Amputation_MNAR_.30_1$weight, mean)  # replace with mean
HIV_TB_Conifection<-impute(Amputation_MNAR_.30_1$HIV_TB_Conifection, mode)  # replace wi
HIV_TB_Conifection<-factor(HIV_TB_Conifection)
Gender<-impute(Amputation_MNAR_.30_1$Gender, mode)  # replace with mode
Gender<-factor(Gender)
Single_imp_Data_MNAR_.30<-as.data.frame(cbind(Age,weight,HIV_TB_Conifection,Gender))
cols <- c("Gender")
Single_imp_Data_MNAR_.30[,cols] <- data.frame(apply(Single_imp_Data_MNAR_.30[cols], 2, a
Single_imp_Data_MNAR_.30$HIV_TB_Conifection[Single_imp_Data_MNAR_.30$HIV_TB_Conifection
Single_imp_Data_MNAR_.30$HIV_TB_Conifection[Single_imp_Data_MNAR_.30$HIV_TB_Conifection
Single_imp_Data_MNAR_.30

dim(Single_imp_Data_MNAR_.30)
md.pattern(Single_imp_Data_MNAR_.30)
str(Single_imp_Data_MNAR_.30)
view(Single_imp_Data_MNAR_.30)
#ANALYSIS: MEAN/SINGLE IMPUTATION 30% MNAR
# We split the data into:  testing and  training sets.
set.seed(42)
default_idx = sample(nrow(Single_imp_Data_MNAR_.30), 5000)
default_trn = Single_imp_Data_MNAR_.30[default_idx, ]
default_tst = Single_imp_Data_MNAR_.30[-default_idx, ]

model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_2
summary(model_glm_2)

#Make predictions
head(predict(model_glm_2, type = "link"))
#the predicted probabilities
head(predict(model_glm_2, type = "response"))

#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the trainging classificat
calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
}
calc_class_err(actual = default_trn$default, predicted = model_glm_pred2)

train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
```

```
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
set.seed(100)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")
c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])

#ROC Curves
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)

#3.)  Creating 30% Missingness Under MCAR
rm(list = ls())
set.seed(100)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
attach(Default2)
Amputation_MCAR_.30_1<-ampute(Default2, prop = 0.30,freq = NULL, mech = "MCAR",std = TRU
Amputation_MCAR_.30_1

#ASSESSING MISSINGNESS
Amputation_MCAR_.30_1$amp
str(Amputation_MCAR_.30_1)
Amputation_MCAR_.30_1$patterns
Amputation_MCAR_.30_1$weights
Amputation_MCAR_.30_1$prop

Amputation_MCAR_.30_1<-as.data.frame(Amputation_MCAR_.30_1$amp)
P1<-function(x) {sum(is.na(x))/length(x)*100}

apply(Amputation_MCAR_.30_1,2,P1) #Gives proportion of missingness in each variable
md.pattern(Amputation_MCAR_.30_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MCAR_.30_1)# Bivariate comparison
aggr_plot <- aggr(Amputation_MCAR_.30_1, col=c('navyblue','yellow'), numbers=TRUE, sortV

library(Hmisc)
#2.1.a) 30% MCAR
str(Amputation_MCAR_.30_1)
drops<-c("weight","Age","HIV_TB_Conifection")
```

```
drops<-c("weight","Age")
DefaultStd<-Amputation_MCAR_.30_1[ , !(names(Amputation_MCAR_.30_1) %in% drops)]
str(DefaultStd)
con.names = DefaultStd %>% select_if(is.character) %>% colnames()
print(con.names)
Amputation_MCAR_.30_1<-Amputation_MCAR_.30_1
Amputation_MCAR_.30_1[,con.names] = data.frame(apply(Amputation_MCAR_.30_1[con.names], 2
Amputation_MCAR_.30_1
dim(Amputation_MCAR_.30_1)
str(Amputation_MCAR_.30_1)
summary(Amputation_MCAR_.30_1)

Age<-impute(Amputation_MCAR_.30_1$Age, mean)  # replace with mean
weight<-impute(Amputation_MCAR_.30_1$weight, mean)  # replace with mean
HIV_TB_Conifection<-impute(Amputation_MCAR_.30_1$HIV_TB_Conifection, mode)  # replace wi
HIV_TB_Conifection<-factor(HIV_TB_Conifection)
Gender<-impute(Amputation_MCAR_.30_1$Gender, mode)  # replace with mode
Gender<-factor(Gender)
Single_imp_Data_MCAR_.30<-as.data.frame(cbind(Age,weight,HIV_TB_Conifection,Gender))
cols <- c("Gender")
Single_imp_Data_MCAR_.30[,cols] <- data.frame(apply(Single_imp_Data_MCAR_.30[cols], 2, a
Single_imp_Data_MCAR_.30$HIV_TB_Conifection[Single_imp_Data_MCAR_.30$HIV_TB_Conifection
Single_imp_Data_MCAR_.30$HIV_TB_Conifection[Single_imp_Data_MCAR_.30$HIV_TB_Conifection
Single_imp_Data_MCAR_.30

dim(Single_imp_Data_MCAR_.30)
md.pattern(Single_imp_Data_MCAR_.30)
str(Single_imp_Data_MCAR_.30)
view(Single_imp_Data_MCAR_.30)
#ANALYSIS: MEAN/SINGLE IMPUTATION 30% MCAR
# We split the data into:  testing and  training sets.
set.seed(42)
default_idx = sample(nrow(Single_imp_Data_MCAR_.30), 5000)
default_trn = Single_imp_Data_MCAR_.30[default_idx, ]
default_tst = Single_imp_Data_MCAR_.30[-default_idx, ]

model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_2
summary(model_glm_2)

#Make predictions
head(predict(model_glm_2, type = "link"))
#the predicted probabilities
head(predict(model_glm_2, type = "response"))

#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the trainging classificat
calc_class_err = function(actual, predicted) {
```

```
  mean(actual != predicted)
}
calc_class_err(actual = default_trn$default, predicted = model_glm_pred2)

train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
set.seed(100)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")
c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])

#ROC Curves
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)

##??????????????50%
#1.)  Creating 50% Missingness Under MAR
rm(list = ls())
set.seed(100)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
attach(Default)
Amputation_MAR_.50_1<-ampute(Default2, prop = 0.5,freq = NULL, mech = "MAR",std = TRUE,
Amputation_MAR_.50_1

#ASSESSING MISSINGNESS
Amputation_MAR_.50_1$amp
str(Amputation_MAR_.50_1)
Amputation_MAR_.50_1$patterns
Amputation_MAR_.50_1$weights
Amputation_MAR_.50_1$prop

Amputation_MAR_.50_1<-as.data.frame(Amputation_MAR_.50_1$amp)
P1<-function(x) {sum(is.na(x))/length(x)*100}

apply(Amputation_MAR_.50_1,2,P1) #Gives proportion of missingness in each variable
md.pattern(Amputation_MAR_.50_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MAR_.50_1)# Bivariate comparison
```

```
aggr_plot <- aggr(Amputation_MAR_.50_1, col=c('navyblue','yellow'), numbers=TRUE, sortVa

library(Hmisc)
#2.1.a) 50% MAR
str(Amputation_MAR_.50_1)
drops<-c("weight","Age","HIV_TB_Conifection")
drops<-c("weight","Age")
DefaultStd<-Amputation_MAR_.50_1[ , !(names(Amputation_MAR_.50_1) %in% drops)]
str(DefaultStd)
con.names = DefaultStd %>% select_if(is.character) %>% colnames()
print(con.names)
Amputation_MAR_.50_1<-Amputation_MAR_.50_1
Amputation_MAR_.50_1[,con.names] = data.frame(apply(Amputation_MAR_.50_1[con.names], 2,
Amputation_MAR_.50_1
dim(Amputation_MAR_.50_1)
str(Amputation_MAR_.50_1)
summary(Amputation_MAR_.50_1)

Age<-impute(Amputation_MAR_.50_1$Age, mean)  # replace with mean
weight<-impute(Amputation_MAR_.50_1$weight, mean)  # replace with mean
HIV_TB_Conifection<-impute(Amputation_MAR_.50_1$HIV_TB_Conifection, mode)  # replace wit
HIV_TB_Conifection<-factor(HIV_TB_Conifection)
Gender<-impute(Amputation_MAR_.50_1$Gender, mode)  # replace with mode
Gender<-factor(Gender)
Single_imp_Data_MAR_.50<-as.data.frame(cbind(Age,weight,HIV_TB_Conifection,Gender))
cols <- c("Gender")
Single_imp_Data_MAR_.50[,cols] <- data.frame(apply(Single_imp_Data_MAR_.50[cols], 2, as.
Single_imp_Data_MAR_.50$HIV_TB_Conifection[Single_imp_Data_MAR_.50$HIV_TB_Conifection ==
Single_imp_Data_MAR_.50$HIV_TB_Conifection[Single_imp_Data_MAR_.50$HIV_TB_Conifection ==
Single_imp_Data_MAR_.50

dim(Single_imp_Data_MAR_.50)
md.pattern(Single_imp_Data_MAR_.50)
str(Single_imp_Data_MAR_.50)
view(Single_imp_Data_MAR_.50)
#ANALYSIS: MEAN/SINGLE IMPUTATION 50% MAR
# We split the data into:  testing and  training sets.
set.seed(42)
default_idx = sample(nrow(Single_imp_Data_MAR_.50), 5000)
default_trn = Single_imp_Data_MAR_.50[default_idx, ]
default_tst = Single_imp_Data_MAR_.50[-default_idx, ]

model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_2
summary(model_glm_2)

#Make predictions
head(predict(model_glm_2, type = "link"))
#the predicted probabilities
head(predict(model_glm_2, type = "response"))
```

```
#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the trainging classificat
calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
}
calc_class_err(actual = default_trn$default, predicted = model_glm_pred2)

train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
set.seed(100)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")
c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])

#ROC Curves
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)

#2.)  Creating 50% Missingness Under MNAR
rm(list = ls())
set.seed(100)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
attach(Default)
Amputation_MNAR_.50_1<-ampute(Default2, prop = 0.50,freq = NULL, mech = "MNAR",std = TRU
Amputation_MNAR_.50_1

#ASSESSING MISSINGNESS
Amputation_MNAR_.50_1$amp
str(Amputation_MNAR_.50_1)
Amputation_MNAR_.50_1$patterns
Amputation_MNAR_.50_1$weights
Amputation_MNAR_.50_1$prop

Amputation_MNAR_.50_1<-as.data.frame(Amputation_MNAR_.50_1$amp)
```

```
P1<-function(x) {sum(is.na(x))/length(x)*100}

apply(Amputation_MNAR_.50_1,2,P1) #Gives proportion of missingness in each variable
md.pattern(Amputation_MNAR_.50_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MNAR_.50_1)# Bivariate comparison
aggr_plot <- aggr(Amputation_MNAR_.50_1, col=c('navyblue','yellow'), numbers=TRUE, sortV

library(Hmisc)
#2.1.a) 50% MNAR
str(Amputation_MNAR_.50_1)
drops<-c("weight","Age","HIV_TB_Conifection")
drops<-c("weight","Age")
DefaultStd<-Amputation_MNAR_.50_1[ , !(names(Amputation_MNAR_.50_1) %in% drops)]
str(DefaultStd)
con.names = DefaultStd %>% select_if(is.character) %>% colnames()
print(con.names)
Amputation_MAR_.50_1<-Amputation_MNAR_.50_1
Amputation_MNAR_.50_1[,con.names] = data.frame(apply(Amputation_MNAR_.50_1[con.names], 2
Amputation_MNAR_.50_1
dim(Amputation_MNAR_.50_1)
str(Amputation_MNAR_.50_1)
summary(Amputation_MNAR_.50_1)

Age<-impute(Amputation_MNAR_.50_1$Age, mean)  # replace with mean
weight<-impute(Amputation_MNAR_.50_1$weight, mean)  # replace with mean
HIV_TB_Conifection<-impute(Amputation_MNAR_.50_1$HIV_TB_Conifection, mode)  # replace wi
HIV_TB_Conifection<-factor(HIV_TB_Conifection)
Gender<-impute(Amputation_MNAR_.50_1$Gender, mode)  # replace with mode
Gender<-factor(Gender)
Single_imp_Data_MNAR_.50<-as.data.frame(cbind(Age,weight,HIV_TB_Conifection,Gender))
cols <- c("Gender")
Single_imp_Data_MNAR_.50[,cols] <- data.frame(apply(Single_imp_Data_MNAR_.50[cols], 2, a
Single_imp_Data_MNAR_.50$HIV_TB_Conifection[Single_imp_Data_MNAR_.50$HIV_TB_Conifection
Single_imp_Data_MNAR_.50$HIV_TB_Conifection[Single_imp_Data_MNAR_.50$HIV_TB_Conifection
Single_imp_Data_MNAR_.50

dim(Single_imp_Data_MNAR_.50)
md.pattern(Single_imp_Data_MNAR_.50)
str(Single_imp_Data_MNAR_.50)
view(Single_imp_Data_MNAR_.50)
#ANALYSIS: MEAN/SINGLE IMPUTATION 50% MNAR
# We split the data into:  testing and  training sets.
set.seed(42)
default_idx = sample(nrow(Single_imp_Data_MNAR_.50), 5000)
default_trn = Single_imp_Data_MNAR_.50[default_idx, ]
default_tst = Single_imp_Data_MNAR_.50[-default_idx, ]

model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_2
summary(model_glm_2)
```

```
#Make predictions
head(predict(model_glm_2, type = "link"))
#the predicted probabilities
head(predict(model_glm_2, type = "response"))

#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the trainging classificat
calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
}
calc_class_err(actual = default_trn$default, predicted = model_glm_pred2)

train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
set.seed(100)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")
c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])

#ROC Curves
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)

#3.)  Creating 50% Missingness Under MCAR
rm(list = ls())
set.seed(100)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
attach(Default2)
Amputation_MCAR_.50_1<-ampute(Default2, prop = 0.50,freq = NULL, mech = "MCAR",std = TRU
Amputation_MCAR_.50_1

#ASSESSING MISSINGNESS
Amputation_MCAR_.50_1$amp
str(Amputation_MCAR_.50_1)
```

```
Amputation_MCAR_.50_1$patterns
Amputation_MCAR_.50_1$weights
Amputation_MCAR_.50_1$prop

Amputation_MCAR_.50_1<-as.data.frame(Amputation_MCAR_.50_1$amp)
P1<-function(x) {sum(is.na(x))/length(x)*100}

apply(Amputation_MCAR_.50_1,2,P1) #Gives proportion of missingness in each variable
md.pattern(Amputation_MCAR_.50_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MCAR_.50_1)# Bivariate comparison
aggr_plot <- aggr(Amputation_MCAR_.50_1, col=c('navyblue','yellow'), numbers=TRUE, sortV

library(Hmisc)
#2.1.a) 50% MCAR
str(Amputation_MCAR_.50_1)
drops<-c("weight","Age","HIV_TB_Conifection")
drops<-c("weight","Age")
DefaultStd<-Amputation_MCAR_.50_1[ , !(names(Amputation_MCAR_.50_1) %in% drops)]
str(DefaultStd)
con.names = DefaultStd %>% select_if(is.character) %>% colnames()
print(con.names)
Amputation_MCAR_.50_1<-Amputation_MCAR_.50_1
Amputation_MCAR_.50_1[,con.names] = data.frame(apply(Amputation_MCAR_.50_1[con.names], 2
Amputation_MCAR_.50_1
dim(Amputation_MCAR_.50_1)
str(Amputation_MCAR_.50_1)
summary(Amputation_MCAR_.50_1)

Age<-impute(Amputation_MCAR_.50_1$Age, mean)  # replace with mean
weight<-impute(Amputation_MCAR_.50_1$weight, mean)  # replace with mean
HIV_TB_Conifection<-impute(Amputation_MCAR_.50_1$HIV_TB_Conifection, mode)  # replace wi
HIV_TB_Conifection<-factor(HIV_TB_Conifection)
Gender<-impute(Amputation_MCAR_.50_1$Gender, mode)  # replace with mode
Gender<-factor(Gender)
Single_imp_Data_MCAR_.50<-as.data.frame(cbind(Age,weight,HIV_TB_Conifection,Gender))
cols <- c("Gender")
Single_imp_Data_MCAR_.50[,cols] <- data.frame(apply(Single_imp_Data_MCAR_.50[cols], 2, a
Single_imp_Data_MCAR_.50$HIV_TB_Conifection[Single_imp_Data_MCAR_.50$HIV_TB_Conifection
Single_imp_Data_MCAR_.50$HIV_TB_Conifection[Single_imp_Data_MCAR_.50$HIV_TB_Conifection
Single_imp_Data_MCAR_.50

dim(Single_imp_Data_MCAR_.50)
md.pattern(Single_imp_Data_MCAR_.50)
str(Single_imp_Data_MCAR_.50)
view(Single_imp_Data_MCAR_.50)
#ANALYSIS: MEAN/SINGLE IMPUTATION 50% MCAR
# We split the data into:  testing and  training sets.
set.seed(42)
default_idx = sample(nrow(Single_imp_Data_MCAR_.50), 5000)
default_trn = Single_imp_Data_MCAR_.50[default_idx, ]
default_tst = Single_imp_Data_MCAR_.50[-default_idx, ]
```

```
model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_2
summary(model_glm_2)

#Make predictions
head(predict(model_glm_2, type = "link"))
#the predicted probabilities
head(predict(model_glm_2, type = "response"))

#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the trainging classificat
calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
}
calc_class_err(actual = default_trn$default, predicted = model_glm_pred2)

train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
set.seed(100)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")
c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])

#ROC Curves
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)

##??????????????80%
#1.)  Creating 80% Missingness Under MAR
rm(list = ls())
set.seed(100)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
attach(Default)
```

```
Amputation_MAR_.80_1<-ampute(Default2, prop = 0.8,freq = NULL, mech = "MAR",std = TRUE,
Amputation_MAR_.80_1

#ASSESSING MISSINGNESS
Amputation_MAR_.80_1$amp
str(Amputation_MAR_.80_1)
Amputation_MAR_.80_1$patterns
Amputation_MAR_.80_1$weights
Amputation_MAR_.80_1$prop

Amputation_MAR_.80_1<-as.data.frame(Amputation_MAR_.80_1$amp)
P1<-function(x) {sum(is.na(x))/length(x)*100}

apply(Amputation_MAR_.80_1,2,P1) #Gives proportion of missingness in each variable
md.pattern(Amputation_MAR_.80_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MAR_.80_1)# Bivariate comparison
aggr_plot <- aggr(Amputation_MAR_.80_1, col=c('navyblue','yellow'), numbers=TRUE, sortVa

library(Hmisc)
#2.1.a) 80% MAR
str(Amputation_MAR_.80_1)
drops<-c("weight","Age","HIV_TB_Conifection")
drops<-c("weight","Age")
DefaultStd<-Amputation_MAR_.80_1[ , !(names(Amputation_MAR_.80_1) %in% drops)]
str(DefaultStd)
con.names = DefaultStd %>% select_if(is.character) %>% colnames()
print(con.names)
Amputation_MAR_.80_1<-Amputation_MAR_.80_1
Amputation_MAR_.80_1[,con.names] = data.frame(apply(Amputation_MAR_.80_1[con.names], 2,
Amputation_MAR_.80_1
dim(Amputation_MAR_.80_1)
str(Amputation_MAR_.80_1)
summary(Amputation_MAR_.80_1)

Age<-impute(Amputation_MAR_.80_1$Age, mean)  # replace with mean
weight<-impute(Amputation_MAR_.80_1$weight, mean)  # replace with mean
HIV_TB_Conifection<-impute(Amputation_MAR_.80_1$HIV_TB_Conifection, mode)  # replace wit
HIV_TB_Conifection<-factor(HIV_TB_Conifection)
Gender<-impute(Amputation_MAR_.80_1$Gender, mode)  # replace with mode
Gender<-factor(Gender)
Single_imp_Data_MAR_.80<-as.data.frame(cbind(Age,weight,HIV_TB_Conifection,Gender))
cols <- c("Gender")
Single_imp_Data_MAR_.80[,cols] <- data.frame(apply(Single_imp_Data_MAR_.80[cols], 2, as.
Single_imp_Data_MAR_.80$HIV_TB_Conifection[Single_imp_Data_MAR_.80$HIV_TB_Conifection ==
Single_imp_Data_MAR_.80$HIV_TB_Conifection[Single_imp_Data_MAR_.80$HIV_TB_Conifection ==
Single_imp_Data_MAR_.80

dim(Single_imp_Data_MAR_.80)
md.pattern(Single_imp_Data_MAR_.80)
str(Single_imp_Data_MAR_.80)
view(Single_imp_Data_MAR_.80)
```

```
#ANALYSIS: MEAN/SINGLE IMPUTATION 80% MAR
# We split the data into:  testing and  training sets.
set.seed(42)
default_idx = sample(nrow(Single_imp_Data_MAR_.80), 5000)
default_trn = Single_imp_Data_MAR_.80[default_idx, ]
default_tst = Single_imp_Data_MAR_.80[-default_idx, ]

model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_2
summary(model_glm_2)

#Make predictions
head(predict(model_glm_2, type = "link"))
#the predicted probabilities
head(predict(model_glm_2, type = "response"))

#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the trainging classificat
calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
}
calc_class_err(actual = default_trn$default, predicted = model_glm_pred2)

train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
set.seed(100)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")
c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])

#ROC Curves
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)

#2.)  Creating 80% Missingness Under MNAR
rm(list = ls())
set.seed(100)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
```

```
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
attach(Default)
Amputation_MNAR_.80_1<-ampute(Default2, prop = 0.80,freq = NULL, mech = "MNAR",std = TRU
Amputation_MNAR_.80_1

#ASSESSING MISSINGNESS
Amputation_MNAR_.80_1$amp
str(Amputation_MNAR_.80_1)
Amputation_MNAR_.80_1$patterns
Amputation_MNAR_.80_1$weights
Amputation_MNAR_.80_1$prop

Amputation_MNAR_.80_1<-as.data.frame(Amputation_MNAR_.80_1$amp)
P1<-function(x) {sum(is.na(x))/length(x)*100}

apply(Amputation_MNAR_.80_1,2,P1) #Gives proportion of missingness in each variable
md.pattern(Amputation_MNAR_.80_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MNAR_.80_1)# Bivariate comparison
aggr_plot <- aggr(Amputation_MNAR_.850_1, col=c('navyblue','yellow'), numbers=TRUE, sort

library(Hmisc)
#2.1.a) 80% MNAR
str(Amputation_MNAR_.80_1)
drops<-c("weight","Age","HIV_TB_Conifection")
drops<-c("weight","Age")
DefaultStd<-Amputation_MNAR_.80_1[ , !(names(Amputation_MNAR_.80_1) %in% drops)]
str(DefaultStd)
con.names = DefaultStd %>% select_if(is.character) %>% colnames()
print(con.names)
Amputation_MAR_.80_1<-Amputation_MNAR_.80_1
Amputation_MNAR_.80_1[,con.names] = data.frame(apply(Amputation_MNAR_.80_1[con.names], 2
Amputation_MNAR_.80_1
dim(Amputation_MNAR_.80_1)
str(Amputation_MNAR_.80_1)
summary(Amputation_MNAR_.80_1)

Age<-impute(Amputation_MNAR_.80_1$Age, mean)  # replace with mean
weight<-impute(Amputation_MNAR_.80_1$weight, mean)  # replace with mean
HIV_TB_Conifection<-impute(Amputation_MNAR_.80_1$HIV_TB_Conifection, mode)  # replace wi
HIV_TB_Conifection<-factor(HIV_TB_Conifection)
Gender<-impute(Amputation_MNAR_.80_1$Gender, mode)  # replace with mode
Gender<-factor(Gender)
Single_imp_Data_MNAR_.80<-as.data.frame(cbind(Age,weight,HIV_TB_Conifection,Gender))
cols <- c("Gender")
Single_imp_Data_MNAR_.80[,cols] <- data.frame(apply(Single_imp_Data_MNAR_.80[cols], 2, a
Single_imp_Data_MNAR_.80$HIV_TB_Conifection[Single_imp_Data_MNAR_.80$HIV_TB_Conifection
Single_imp_Data_MNAR_.80$HIV_TB_Conifection[Single_imp_Data_MNAR_.80$HIV_TB_Conifection
Single_imp_Data_MNAR_.80
```

```
dim(Single_imp_Data_MNAR_.80)
md.pattern(Single_imp_Data_MNAR_.80)
str(Single_imp_Data_MNAR_.80)
view(Single_imp_Data_MNAR_.80)
#ANALYSIS: MEAN/SINGLE IMPUTATION 80% MNAR
# We split the data into:  testing and  training sets.
set.seed(42)
default_idx = sample(nrow(Single_imp_Data_MNAR_.80), 5000)
default_trn = Single_imp_Data_MNAR_.80[default_idx, ]
default_tst = Single_imp_Data_MNAR_.80[-default_idx, ]

model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_2
summary(model_glm_2)

#Make predictions
head(predict(model_glm_2, type = "link"))
#the predicted probabilities
head(predict(model_glm_2, type = "response"))

#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the trainging classificat
calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
}
calc_class_err(actual = default_trn$default, predicted = model_glm_pred2)

train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
set.seed(100)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")
c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])

#ROC Curves
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)

#3.)  Creating 80% Missingness Under MCAR
rm(list = ls())
set.seed(100)
```

```
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
attach(Default2)
Amputation_MCAR_.80_1<-ampute(Default2, prop = 0.80,freq = NULL, mech = "MCAR",std = TRU
Amputation_MCAR_.80_1

#ASSESSING MISSINGNESS
Amputation_MCAR_.80_1$amp
str(Amputation_MCAR_.80_1)
Amputation_MCAR_.80_1$patterns
Amputation_MCAR_.80_1$weights
Amputation_MCAR_.80_1$prop

Amputation_MCAR_.80_1<-as.data.frame(Amputation_MCAR_.80_1$amp)
P1<-function(x) {sum(is.na(x))/length(x)*100}

apply(Amputation_MCAR_.80_1,2,P1) #Gives proportion of missingness in each variable
md.pattern(Amputation_MCAR_.80_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MCAR_.80_1)# Bivariate comparison
aggr_plot <- aggr(Amputation_MCAR_.80_1, col=c('navyblue','yellow'), numbers=TRUE, sortV

library(Hmisc)
#2.1.a) 80% MCAR
str(Amputation_MCAR_.80_1)
drops<-c("weight","Age","HIV_TB_Conifection")
drops<-c("weight","Age")
DefaultStd<-Amputation_MCAR_.80_1[ , !(names(Amputation_MCAR_.80_1) %in% drops)]
str(DefaultStd)
con.names = DefaultStd %>% select_if(is.character) %>% colnames()
print(con.names)
Amputation_MCAR_.80_1<-Amputation_MCAR_.80_1
Amputation_MCAR_.80_1[,con.names] = data.frame(apply(Amputation_MCAR_.80_1[con.names], 2
Amputation_MCAR_.80_1
dim(Amputation_MCAR_.80_1)
str(Amputation_MCAR_.80_1)
summary(Amputation_MCAR_.80_1)

Age<-impute(Amputation_MCAR_.80_1$Age, mean)  # replace with mean
weight<-impute(Amputation_MCAR_.80_1$weight, mean)  # replace with mean
HIV_TB_Conifection<-impute(Amputation_MCAR_.80_1$HIV_TB_Conifection, mode)  # replace wi
HIV_TB_Conifection<-factor(HIV_TB_Conifection)
Gender<-impute(Amputation_MCAR_.80_1$Gender, mode)  # replace with mode
Gender<-factor(Gender)
Single_imp_Data_MCAR_.80<-as.data.frame(cbind(Age,weight,HIV_TB_Conifection,Gender))
```

```
cols <- c("Gender")
Single_imp_Data_MCAR_.80[,cols] <- data.frame(apply(Single_imp_Data_MCAR_.80[cols], 2, a
Single_imp_Data_MCAR_.80$HIV_TB_Conifection[Single_imp_Data_MCAR_.80$HIV_TB_Conifection
Single_imp_Data_MCAR_.80$HIV_TB_Conifection[Single_imp_Data_MCAR_.80$HIV_TB_Conifection
Single_imp_Data_MCAR_.80

dim(Single_imp_Data_MCAR_.80)
md.pattern(Single_imp_Data_MCAR_.80)
str(Single_imp_Data_MCAR_.80)
view(Single_imp_Data_MCAR_.80)
#ANALYSIS: MEAN/SINGLE IMPUTATION 80% MCAR
# We split the data into:  testing and  training sets.
set.seed(42)
default_idx = sample(nrow(Single_imp_Data_MCAR_.80), 5000)
default_trn = Single_imp_Data_MCAR_.80[default_idx, ]
default_tst = Single_imp_Data_MCAR_.80[-default_idx, ]

model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_2
summary(model_glm_2)

#Make predictions
head(predict(model_glm_2, type = "link"))
#the predicted probabilities
head(predict(model_glm_2, type = "response"))

#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the trainging classificat
calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
}
calc_class_err(actual = default_trn$default, predicted = model_glm_pred2)

train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
set.seed(100)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")
c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])

#ROC Curves
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
```

```
as.numeric(test_roc$auc)

############MULTIPLE IMPUTATION#########################################################
#####USING LONG FORMAT OF THE IMPUTED DATA SET  7% Missingness
##MAR 0.07
#1.)  Creating 7% Missingness Under MAR
rm(list = ls())
set.seed(100)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
attach(Default2)
Amputation_MAR_.07_1<-ampute(Default2, prop = 0.07,freq = NULL, mech = "MAR",std = TRUE,
Amputation_MAR_.07_1

#ASSESSING MISSINGNESS
Amputation_MAR_.07_1$amp
str(Amputation_MAR_.07_1)
Amputation_MAR_.07_1$patterns
Amputation_MAR_.07_1$weights
Amputation_MAR_.07_1$prop

Amputation_MAR_.07_1<-as.data.frame(Amputation_MAR_.07_1$amp)
P1<-function(x) {sum(is.na(x))/length(x)*100}

apply(Amputation_MAR_.07_1,2,P1) #Gives proportion of missingness in each variable
md.pattern(Amputation_MAR_.07_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MAR_.07_1)# Bivariate comparison
aggr_plot <- aggr(Amputation_MAR_.07_1, col=c('navyblue','yellow'), numbers=TRUE, sortVa

library(Hmisc)
#2.1.a) 7% MAR
str(Amputation_MAR_.07_1)
drops<-c("weight","Age","HIV_TB_Conifection")
drops<-c("weight","Age")
DefaultStd<-Amputation_MAR_.07_1[ , !(names(Amputation_MAR_.07_1) %in% drops)]
str(DefaultStd)
con.names = DefaultStd %>% select_if(is.character) %>% colnames()
print(con.names)
Amputation_MAR_.07_1<-Amputation_MAR_.07_1
Amputation_MAR_.07_1[,con.names] = data.frame(apply(Amputation_MAR_.07_1[con.names], 2,
Amputation_MAR_.07_1
dim(Amputation_MAR_.07_1)
str(Amputation_MAR_.07_1)
summary(Amputation_MAR_.07_1)
```

```
Amputation_MAR_.07_1_imp.LR <- mice(Amputation_MAR_.07_1, m=3, maxit=10, seed=2268, prin
summary(Amputation_MAR_.07_1_imp.LR)
densityplot(Amputation_MAR_.07_1_imp.LR)
stripplot(Amputation_MAR_.07_1_imp.LR)
head(Amputation_MAR_.07_1_imp.LR$imp$HIV_TB_Conifection)
tail(Amputation_MAR_.07_1_imp.LR$imp$HIV_TB_Conifection)
str(Amputation_MAR_.07_1_imp.LR)
#lOOking at each imputed data set
anescomp1 <- mice::complete(Amputation_MAR_.07_1_imp.LR, 1)
head(anescomp1)
md.pattern(anescomp1, plot = TRUE,rotate.names = TRUE)
anescomp2 <- mice::complete(Amputation_MAR_.07_1_imp.LR, 2)
head(anescomp2)
md.pattern(anescomp2, plot = TRUE,rotate.names = TRUE)
anescomp3 <- mice::complete(Amputation_MAR_.07_1_imp.LR, 3)
head(anescomp3)
md.pattern(anescomp3, plot = TRUE,rotate.names = TRUE)
#long matrix with stacked complete data
library(dplyr)
MAR.07.LR_1 <- complete(Amputation_MAR_.07_1_imp.LR, 'long')
#Dropping some cloumns
#https://www.listendata.com/2015/06/r-keep-drop-columns-from-data-frame.html
MAR.07.LR_1<- MAR.07.LR_1[,!(names(MAR.07.LR_1) %in% c(".imp",".id"))]
dim(MAR.07.LR_1)
view(MAR.07.LR_1)
str(MAR.07.LR_1)

set.seed(42)
default_idx = sample(nrow(MAR.07.LR_1), 15000)
default_trn = MAR.07.LR_1[default_idx, ]
default_tst = MAR.07.LR_1[-default_idx, ]
dim(default_trn)
dim(default_tst)
################
model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_2
summary(model_glm_2)
#Make predictions
head(predict(model_glm_2, type = "link"))
#the predicted probabilities
head(predict(model_glm_2, type = "response"))
#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the trainging classificat
calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
}
calc_class_err(actual = default_trn$default, predicted = model_glm_pred2)
```

```
train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
set.seed(100)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")
c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])
#ROC Curves
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)

#######MNAR .07
#1.)  Creating 7% Missingness Under MNAR
rm(list = ls())
set.seed(100)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
attach(Default2)
Amputation_MNAR_.07_1<-ampute(Default2, prop = 0.07,freq = NULL, mech = "MNAR",std = TRU
Amputation_MNAR_.07_1

#ASSESSING MISSINGNESS
Amputation_MNAR_.07_1$amp
str(Amputation_MNAR_.07_1)
Amputation_MNAR_.07_1$patterns
Amputation_MNAR_.07_1$weights
Amputation_MNAR_.07_1$prop

Amputation_MNAR_.07_1<-as.data.frame(Amputation_MNAR_.07_1$amp)
P1<-function(x) {sum(is.na(x))/length(x)*100}

apply(Amputation_MNAR_.07_1,2,P1) #Gives proportion of missingness in each variable
md.pattern(Amputation_MNAR_.07_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MNAR_.07_1)# Bivariate comparison
aggr_plot <- aggr(Amputation_MNAR_.07_1, col=c('navyblue','yellow'), numbers=TRUE, sortV

library(Hmisc)
#2.1.a) 7% MNAR
```

```
str(Amputation_MNAR_.07_1)
drops<-c("weight","Age","HIV_TB_Conifection")
drops<-c("weight","Age")
DefaultStd<-Amputation_MNAR_.07_1[ , !(names(Amputation_MNAR_.07_1) %in% drops)]
str(DefaultStd)
con.names = DefaultStd %>% select_if(is.character) %>% colnames()
print(con.names)
Amputation_MNAR_.07_1<-Amputation_MNAR_.07_1
Amputation_MNAR_.07_1[,con.names] = data.frame(apply(Amputation_MNAR_.07_1[con.names], 2
Amputation_MNAR_.07_1
dim(Amputation_MNAR_.07_1)
str(Amputation_MNAR_.07_1)
summary(Amputation_MNAR_.07_1)

Amputation_MNAR_.07_1_imp.LR <- mice(Amputation_MNAR_.07_1, m=3, maxit=10, seed=2268, pr
summary(Amputation_MNAR_.07_1_imp.LR)
densityplot(Amputation_MNAR_.07_1_imp.LR)
stripplot(Amputation_MNAR_.07_1_imp.LR)
head(Amputation_MNAR_.07_1_imp.LR$imp$HIV_TB_Conifection)
tail(Amputation_MNAR_.07_1_imp.LR$imp$HIV_TB_Conifection)
str(Amputation_MNAR_.07_1_imp.LR)
#lOOking at each imputed data set
anescomp1 <- mice::complete(Amputation_MNAR_.07_1_imp.LR, 1)
head(anescomp1)
md.pattern(anescomp1, plot = TRUE,rotate.names = TRUE)
anescomp2 <- mice::complete(Amputation_MNAR_.07_1_imp.LR, 2)
head(anescomp2)
md.pattern(anescomp2, plot = TRUE,rotate.names = TRUE)
anescomp3 <- mice::complete(Amputation_MNAR_.07_1_imp.LR, 3)
head(anescomp3)
md.pattern(anescomp3, plot = TRUE,rotate.names = TRUE)
#long matrix with stacked complete data
library(dplyr)
MNAR.07.LR_1 <- complete(Amputation_MNAR_.07_1_imp.LR, 'long')
#Dropping some cloumns
#https://www.listendata.com/2015/06/r-keep-drop-columns-from-data-frame.html
MNAR.07.LR_1<- MNAR.07.LR_1[,!(names(MNAR.07.LR_1) %in% c(".imp",".id"))]
dim(MNAR.07.LR_1)
view(MNAR.07.LR_1)
str(MNAR.07.LR_1)

set.seed(42)
default_idx = sample(nrow(MNAR.07.LR_1), 15000)
default_trn = MNAR.07.LR_1[default_idx, ]
default_tst = MNAR.07.LR_1[-default_idx, ]
dim(default_trn)
dim(default_tst)
################
model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_2
summary(model_glm_2)
```

```
#Make predictions
head(predict(model_glm_2, type = "link"))
#the predicted probabilities
head(predict(model_glm_2, type = "response"))
#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the trainging classificat
calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
}
calc_class_err(actual = default_trn$default, predicted = model_glm_pred2)

train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
set.seed(100)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")
c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])
#ROC Curves
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)

###MCAR .07
#1.)  Creating 7% Missingness Under MCAR
rm(list = ls())
set.seed(100)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
attach(Default2)
Amputation_MCAR_.07_1<-ampute(Default2, prop = 0.07,freq = NULL, mech = "MCAR",std = TRU
Amputation_MCAR_.07_1

#ASSESSING MISSINGNESS
Amputation_MCAR_.07_1$amp
str(Amputation_MCAR_.07_1)
Amputation_MCAR_.07_1$patterns
```

```
Amputation_MCAR_.07_1$weights
Amputation_MCAR_.07_1$prop

Amputation_MCAR_.07_1<-as.data.frame(Amputation_MCAR_.07_1$amp)
P1<-function(x) {sum(is.na(x))/length(x)*100}

apply(Amputation_MCAR_.07_1,2,P1) #Gives proportion of missingness in each variable
md.pattern(Amputation_MCAR_.07_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MCAR_.07_1)# Bivariate comparison
aggr_plot <- aggr(Amputation_MCAR_.07_1, col=c('navyblue','yellow'), numbers=TRUE, sortV

library(Hmisc)
#2.1.a) 7% MNAR
str(Amputation_MCAR_.07_1)
drops<-c("weight","Age","HIV_TB_Conifection")
drops<-c("weight","Age")
DefaultStd<-Amputation_MCAR_.07_1[ , !(names(Amputation_MCAR_.07_1) %in% drops)]
str(DefaultStd)
con.names = DefaultStd %>% select_if(is.character) %>% colnames()
print(con.names)
Amputation_MCAR_.07_1<-Amputation_MCAR_.07_1
Amputation_MCAR_.07_1[,con.names] = data.frame(apply(Amputation_MCAR_.07_1[con.names], 2
Amputation_MCAR_.07_1
dim(Amputation_MCAR_.07_1)
str(Amputation_MCAR_.07_1)
summary(Amputation_MCAR_.07_1)

Amputation_MCAR_.07_1_imp.LR <- mice(Amputation_MCAR_.07_1, m=3, maxit=10, seed=2268, pr
summary(Amputation_MCAR_.07_1_imp.LR)
densityplot(Amputation_MCAR_.07_1_imp.LR)
stripplot(Amputation_MCAR_.07_1_imp.LR)
head(Amputation_MCAR_.07_1_imp.LR$imp$HIV_TB_Conifection)
tail(Amputation_MCAR_.07_1_imp.LR$imp$HIV_TB_Conifection)
str(Amputation_MCAR_.07_1_imp.LR)
#lOOking at each imputed data set
anescomp1 <- mice::complete(Amputation_MCAR_.07_1_imp.LR, 1)
head(anescomp1)
md.pattern(anescomp1, plot = TRUE,rotate.names = TRUE)
anescomp2 <- mice::complete(Amputation_MCAR_.07_1_imp.LR, 2)
head(anescomp2)
md.pattern(anescomp2, plot = TRUE,rotate.names = TRUE)
anescomp3 <- mice::complete(Amputation_MCAR_.07_1_imp.LR, 3)
head(anescomp3)
md.pattern(anescomp3, plot = TRUE,rotate.names = TRUE)
#long matrix with stacked complete data
library(dplyr)
MCAR.07.LR_1 <- complete(Amputation_MCAR_.07_1_imp.LR, 'long')
#Dropping some cloumns
#https://www.listendata.com/2015/06/r-keep-drop-columns-from-data-frame.html
MCAR.07.LR_1<- MCAR.07.LR_1[,!(names(MCAR.07.LR_1) %in% c(".imp",".id"))]
dim(MCAR.07.LR_1)
```

```
view(MCAR.07.LR_1)
str(MCAR.07.LR_1)

set.seed(42)
default_idx = sample(nrow(MCAR.07.LR_1), 15000)
default_trn = MCAR.07.LR_1[default_idx, ]
default_tst = MCAR.07.LR_1[-default_idx, ]
dim(default_trn)
dim(default_tst)
#################
model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_2
summary(model_glm_2)
#Make predictions
head(predict(model_glm_2, type = "link"))
#the predicted probabilities
head(predict(model_glm_2, type = "response"))
#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the trainging classificat
calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
}
calc_class_err(actual = default_trn$default, predicted = model_glm_pred2)

train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
set.seed(100)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")
c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])
#ROC Curves
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)

#####USING LONG FORMAT OF THE IMPUTED DATA SET  10% Missingness
##MAR 0.10
#1.)  Creating 10% Missingness Under MAR
rm(list = ls())
set.seed(100)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
```

```
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
attach(Default2)
Amputation_MAR_.10_1<-ampute(Default2, prop = 0.10,freq = NULL, mech = "MAR",std = TRUE,
Amputation_MAR_.10_1

#ASSESSING MISSINGNESS
Amputation_MAR_.10_1$amp
str(Amputation_MAR_.10_1)
Amputation_MAR_.10_1$patterns
Amputation_MAR_.10_1$weights
Amputation_MAR_.10_1$prop

Amputation_MAR_.10_1<-as.data.frame(Amputation_MAR_.10_1$amp)
P1<-function(x) {sum(is.na(x))/length(x)*100}

apply(Amputation_MAR_.10_1,2,P1) #Gives proportion of missingness in each variable
md.pattern(Amputation_MAR_.10_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MAR_.10_1)# Bivariate comparison
aggr_plot <- aggr(Amputation_MAR_.10_1, col=c('navyblue','yellow'), numbers=TRUE, sortVa

library(Hmisc)
#2.1.a) 10% MAR
str(Amputation_MAR_.10_1)
drops<-c("weight","Age","HIV_TB_Conifection")
drops<-c("weight","Age")
DefaultStd<-Amputation_MAR_.10_1[ , !(names(Amputation_MAR_.10_1) %in% drops)]
str(DefaultStd)
con.names = DefaultStd %>% select_if(is.character) %>% colnames()
print(con.names)
Amputation_MAR_.10_1<-Amputation_MAR_.10_1
Amputation_MAR_.10_1[,con.names] = data.frame(apply(Amputation_MAR_.10_1[con.names], 2,
Amputation_MAR_.10_1
dim(Amputation_MAR_.10_1)
str(Amputation_MAR_.10_1)
summary(Amputation_MAR_.10_1)

Amputation_MAR_.10_1_imp.LR <- mice(Amputation_MAR_.10_1, m=3, maxit=10, seed=2268, prin
summary(Amputation_MAR_.10_1_imp.LR)
densityplot(Amputation_MAR_.10_1_imp.LR)
stripplot(Amputation_MAR_.10_1_imp.LR)
head(Amputation_MAR_.10_1_imp.LR$imp$HIV_TB_Conifection)
tail(Amputation_MAR_.10_1_imp.LR$imp$HIV_TB_Conifection)
str(Amputation_MAR_.10_1_imp.LR)
#lOOking at each imputed data set
anescomp1 <- mice::complete(Amputation_MAR_.10_1_imp.LR, 1)
head(anescomp1)
```

```
md.pattern(anescomp1, plot = TRUE,rotate.names = TRUE)
anescomp2 <- mice::complete(Amputation_MAR_.10_1_imp.LR, 2)
head(anescomp2)
md.pattern(anescomp2, plot = TRUE,rotate.names = TRUE)
anescomp3 <- mice::complete(Amputation_MAR_.10_1_imp.LR, 3)
head(anescomp3)
md.pattern(anescomp3, plot = TRUE,rotate.names = TRUE)
#long matrix with stacked complete data
library(dplyr)
MAR.10.LR_1 <- complete(Amputation_MAR_.10_1_imp.LR, 'long')
#Dropping some cloumns
#https://www.listendata.com/2015/06/r-keep-drop-columns-from-data-frame.html
MAR.10.LR_1<- MAR.10.LR_1[,!(names(MAR.10.LR_1) %in% c(".imp",".id"))]
dim(MAR.10.LR_1)
view(MAR.10.LR_1)
str(MAR.10.LR_1)

set.seed(42)
default_idx = sample(nrow(MAR.10.LR_1), 15000)
default_trn = MAR.10.LR_1[default_idx, ]
default_tst = MAR.10.LR_1[-default_idx, ]
dim(default_trn)
dim(default_tst)
#################
model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_2
summary(model_glm_2)
#Make predictions
head(predict(model_glm_2, type = "link"))
#the predicted probabilities
head(predict(model_glm_2, type = "response"))
#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the trainging classificat
calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
}
calc_class_err(actual = default_trn$default, predicted = model_glm_pred2)

train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
set.seed(100)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")
c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])
```

```
#ROC Curves
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)

#######MNAR .10
#1.)  Creating 10% Missingness Under MNAR
rm(list = ls())
set.seed(100)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
attach(Default2)
Amputation_MNAR_.10_1<-ampute(Default2, prop = 0.10,freq = NULL, mech = "MNAR",std = TRU
Amputation_MNAR_.10_1

#ASSESSING MISSINGNESS
Amputation_MNAR_.10_1$amp
str(Amputation_MNAR_.10_1)
Amputation_MNAR_.10_1$patterns
Amputation_MNAR_.10_1$weights
Amputation_MNAR_.10_1$prop

Amputation_MNAR_.10_1<-as.data.frame(Amputation_MNAR_.10_1$amp)
P1<-function(x) {sum(is.na(x))/length(x)*100}

apply(Amputation_MNAR_.10_1,2,P1) #Gives proportion of missingness in each variable
md.pattern(Amputation_MNAR_.10_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MNAR_.10_1)# Bivariate comparison
aggr_plot <- aggr(Amputation_MNAR_.10_1, col=c('navyblue','yellow'), numbers=TRUE, sortV

library(Hmisc)
#2.1.a) 10% MNAR
str(Amputation_MNAR_.10_1)
drops<-c("weight","Age","HIV_TB_Conifection")
drops<-c("weight","Age")
DefaultStd<-Amputation_MNAR_.10_1[ , !(names(Amputation_MNAR_.10_1) %in% drops)]
str(DefaultStd)
con.names = DefaultStd %>% select_if(is.character) %>% colnames()
print(con.names)
Amputation_MNAR_.10_1<-Amputation_MNAR_.10_1
Amputation_MNAR_.10_1[,con.names] = data.frame(apply(Amputation_MNAR_.10_1[con.names], 2
Amputation_MNAR_.10_1
dim(Amputation_MNAR_.10_1)
```

```
str(Amputation_MNAR_.10_1)
summary(Amputation_MNAR_.10_1)

Amputation_MNAR_.10_1_imp.LR <- mice(Amputation_MNAR_.10_1, m=3, maxit=10, seed=2268, pr
summary(Amputation_MNAR_.10_1_imp.LR)
densityplot(Amputation_MNAR_.10_1_imp.LR)
stripplot(Amputation_MNAR_.10_1_imp.LR)
head(Amputation_MNAR_.10_1_imp.LR$imp$HIV_TB_Conifection)
tail(Amputation_MNAR_.10_1_imp.LR$imp$HIV_TB_Conifection)
str(Amputation_MNAR_.10_1_imp.LR)
#lOOking at each imputed data set
anescomp1 <- mice::complete(Amputation_MNAR_.10_1_imp.LR, 1)
head(anescomp1)
md.pattern(anescomp1, plot = TRUE,rotate.names = TRUE)
anescomp2 <- mice::complete(Amputation_MNAR_.10_1_imp.LR, 2)
head(anescomp2)
md.pattern(anescomp2, plot = TRUE,rotate.names = TRUE)
anescomp3 <- mice::complete(Amputation_MNAR_.10_1_imp.LR, 3)
head(anescomp3)
md.pattern(anescomp3, plot = TRUE,rotate.names = TRUE)
#long matrix with stacked complete data
library(dplyr)
MNAR.10.LR_1 <- complete(Amputation_MNAR_.10_1_imp.LR, 'long')
#Dropping some cloumns
#https://www.listendata.com/2015/06/r-keep-drop-columns-from-data-frame.html
MNAR.10.LR_1<- MNAR.10.LR_1[,!(names(MNAR.10.LR_1) %in% c(".imp",".id"))]
dim(MNAR.10.LR_1)
view(MNAR.10.LR_1)
str(MNAR.10.LR_1)

set.seed(42)
default_idx = sample(nrow(MNAR.10.LR_1), 15000)
default_trn = MNAR.10.LR_1[default_idx, ]
default_tst = MNAR.10.LR_1[-default_idx, ]
dim(default_trn)
dim(default_tst)
#################
model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_2
summary(model_glm_2)
#Make predictions
head(predict(model_glm_2, type = "link"))
#the predicted probabilities
head(predict(model_glm_2, type = "response"))
#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the trainging classificat
calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
```

```
}
calc_class_err(actual = default_trn$default, predicted = model_glm_pred2)

train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
set.seed(100)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")
c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])
#ROC Curves
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)

###MCAR .10
#1.)  Creating 10% Missingness Under MCAR
rm(list = ls())
set.seed(100)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
attach(Default2)
Amputation_MCAR_.10_1<-ampute(Default2, prop = 0.10,freq = NULL, mech = "MCAR",std = TRU
Amputation_MCAR_.10_1

#ASSESSING MISSINGNESS
Amputation_MCAR_.10_1$amp
str(Amputation_MCAR_.10_1)
Amputation_MCAR_.10_1$patterns
Amputation_MCAR_.10_1$weights
Amputation_MCAR_.10_1$prop

Amputation_MCAR_.10_1<-as.data.frame(Amputation_MCAR_.10_1$amp)
P1<-function(x) {sum(is.na(x))/length(x)*100}

apply(Amputation_MCAR_.10_1,2,P1) #Gives proportion of missingness in each variable
md.pattern(Amputation_MCAR_.10_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MCAR_.10_1)# Bivariate comparison
aggr_plot <- aggr(Amputation_MCAR_.10_1, col=c('navyblue','yellow'), numbers=TRUE, sortV
```

```
library(Hmisc)
#2.1.a) 10% MCAR
str(Amputation_MCAR_.10_1)
drops<-c("weight","Age","HIV_TB_Conifection")
drops<-c("weight","Age")
DefaultStd<-Amputation_MCAR_.10_1[ , !(names(Amputation_MCAR_.10_1) %in% drops)]
str(DefaultStd)
con.names = DefaultStd %>% select_if(is.character) %>% colnames()
print(con.names)
Amputation_MCAR_.10_1<-Amputation_MCAR_.10_1
Amputation_MCAR_.10_1[,con.names] = data.frame(apply(Amputation_MCAR_.10_1[con.names], 2
Amputation_MCAR_.10_1
dim(Amputation_MCAR_.10_1)
str(Amputation_MCAR_.10_1)
summary(Amputation_MCAR_.10_1)

Amputation_MCAR_.10_1_imp.LR <- mice(Amputation_MCAR_.10_1, m=3, maxit=10, seed=2268, pr
summary(Amputation_MCAR_.10_1_imp.LR)
densityplot(Amputation_MCAR_.10_1_imp.LR)
stripplot(Amputation_MCAR_.10_1_imp.LR)
head(Amputation_MCAR_.10_1_imp.LR$imp$HIV_TB_Conifection)
tail(Amputation_MCAR_.10_1_imp.LR$imp$HIV_TB_Conifection)
str(Amputation_MCAR_.10_1_imp.LR)
#lOOking at each imputed data set
anescomp1 <- mice::complete(Amputation_MCAR_.10_1_imp.LR, 1)
head(anescomp1)
md.pattern(anescomp1, plot = TRUE,rotate.names = TRUE)
anescomp2 <- mice::complete(Amputation_MCAR_.10_1_imp.LR, 2)
head(anescomp2)
md.pattern(anescomp2, plot = TRUE,rotate.names = TRUE)
anescomp3 <- mice::complete(Amputation_MCAR_.10_1_imp.LR, 3)
head(anescomp3)
md.pattern(anescomp3, plot = TRUE,rotate.names = TRUE)
#long matrix with stacked complete data
library(dplyr)
MCAR.10.LR_1 <- complete(Amputation_MCAR_.10_1_imp.LR, 'long')
#Dropping some cloumns
#https://www.listendata.com/2015/06/r-keep-drop-columns-from-data-frame.html
MCAR.10.LR_1<- MCAR.10.LR_1[,!(names(MCAR.10.LR_1) %in% c(".imp",".id"))]
dim(MCAR.10.LR_1)
view(MCAR.10.LR_1)
str(MCAR.10.LR_1)

set.seed(42)
default_idx = sample(nrow(MCAR.10.LR_1), 15000)
default_trn = MCAR.10.LR_1[default_idx, ]
default_tst = MCAR.10.LR_1[-default_idx, ]
dim(default_trn)
dim(default_tst)
################
model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
```

```
model_glm_2
summary(model_glm_2)
#Make predictions
head(predict(model_glm_2, type = "link"))
#the predicted probabilities
head(predict(model_glm_2, type = "response"))
#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the trainging classificat
calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
}
calc_class_err(actual = default_trn$default, predicted = model_glm_pred2)

train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
set.seed(100)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")
c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])
#ROC Curves
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)

#####USING LONG FORMAT OF THE IMPUTED DATA SET  30% Missingness
##MAR 0.30
#1.)  Creating 30% Missingness Under MAR
rm(list = ls())
set.seed(100)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
attach(Default2)
Amputation_MAR_.30_1<-ampute(Default2, prop = 0.30,freq = NULL, mech = "MAR",std = TRUE,
Amputation_MAR_.30_1

#ASSESSING MISSINGNESS
```

```
Amputation_MAR_.30_1$amp
str(Amputation_MAR_.30_1)
Amputation_MAR_.30_1$patterns
Amputation_MAR_.30_1$weights
Amputation_MAR_.30_1$prop

Amputation_MAR_.30_1<-as.data.frame(Amputation_MAR_.30_1$amp)
P1<-function(x) {sum(is.na(x))/length(x)*100}

apply(Amputation_MAR_.30_1,2,P1) #Gives proportion of missingness in each variable
md.pattern(Amputation_MAR_.30_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MAR_.30_1)# Bivariate comparison
aggr_plot <- aggr(Amputation_MAR_.30_1, col=c('navyblue','yellow'), numbers=TRUE, sortVa

library(Hmisc)
#2.1.a) 30% MAR
str(Amputation_MAR_.30_1)
drops<-c("weight","Age","HIV_TB_Conifection")
drops<-c("weight","Age")
DefaultStd<-Amputation_MAR_.30_1[ , !(names(Amputation_MAR_.30_1) %in% drops)]
str(DefaultStd)
con.names = DefaultStd %>% select_if(is.character) %>% colnames()
print(con.names)
Amputation_MAR_.30_1<-Amputation_MAR_.30_1
Amputation_MAR_.30_1[,con.names] = data.frame(apply(Amputation_MAR_.30_1[con.names], 2,
Amputation_MAR_.30_1
dim(Amputation_MAR_.30_1)
str(Amputation_MAR_.30_1)
summary(Amputation_MAR_.30_1)

Amputation_MAR_.30_1_imp.LR <- mice(Amputation_MAR_.30_1, m=3, maxit=10, seed=2268, prin
summary(Amputation_MAR_.30_1_imp.LR)
densityplot(Amputation_MAR_.30_1_imp.LR)
stripplot(Amputation_MAR_.30_1_imp.LR)
head(Amputation_MAR_.30_1_imp.LR$imp$HIV_TB_Conifection)
tail(Amputation_MAR_.30_1_imp.LR$imp$HIV_TB_Conifection)
str(Amputation_MAR_.30_1_imp.LR)
#lOOking at each imputed data set
anescomp1 <- mice::complete(Amputation_MAR_.30_1_imp.LR, 1)
head(anescomp1)
md.pattern(anescomp1, plot = TRUE,rotate.names = TRUE)
anescomp2 <- mice::complete(Amputation_MAR_.30_1_imp.LR, 2)
head(anescomp2)
md.pattern(anescomp2, plot = TRUE,rotate.names = TRUE)
anescomp3 <- mice::complete(Amputation_MAR_.30_1_imp.LR, 3)
head(anescomp3)
md.pattern(anescomp3, plot = TRUE,rotate.names = TRUE)
#long matrix with stacked complete data
library(dplyr)
MAR.30.LR_1 <- complete(Amputation_MAR_.30_1_imp.LR, 'long')
#Dropping some cloumns
```

```
#https://www.listendata.com/2015/06/r-keep-drop-columns-from-data-frame.html
MAR.30.LR_1<- MAR.30.LR_1[,!(names(MAR.30.LR_1) %in% c(".imp",".id"))]
dim(MAR.30.LR_1)
view(MAR.30.LR_1)
str(MAR.30.LR_1)

set.seed(42)
default_idx = sample(nrow(MAR.30.LR_1), 15000)
default_trn = MAR.30.LR_1[default_idx, ]
default_tst = MAR.30.LR_1[-default_idx, ]
dim(default_trn)
dim(default_tst)
#################
model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_2
summary(model_glm_2)
#Make predictions
head(predict(model_glm_2, type = "link"))
#the predicted probabilities
head(predict(model_glm_2, type = "response"))
#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the trainging classificat
calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
}
calc_class_err(actual = default_trn$default, predicted = model_glm_pred2)

train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
set.seed(100)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")
c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])
#ROC Curves
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)

#######MNAR .30
#1.)  Creating 30% Missingness Under MNAR
rm(list = ls())
set.seed(100)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
```

```
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
attach(Default2)
Amputation_MNAR_.30_1<-ampute(Default2, prop = 0.30,freq = NULL, mech = "MNAR",std = TRU
Amputation_MNAR_.30_1

#ASSESSING MISSINGNESS
Amputation_MNAR_.30_1$amp
str(Amputation_MNAR_.30_1)
Amputation_MNAR_.30_1$patterns
Amputation_MNAR_.30_1$weights
Amputation_MNAR_.30_1$prop

Amputation_MNAR_.30_1<-as.data.frame(Amputation_MNAR_.30_1$amp)
P1<-function(x) {sum(is.na(x))/length(x)*100}

apply(Amputation_MNAR_.30_1,2,P1) #Gives proportion of missingness in each variable
md.pattern(Amputation_MNAR_.30_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MNAR_.30_1)# Bivariate comparison
aggr_plot <- aggr(Amputation_MNAR_.30_1, col=c('navyblue','yellow'), numbers=TRUE, sortV

library(Hmisc)
#2.1.a) 30% MNAR
str(Amputation_MNAR_.30_1)
drops<-c("weight","Age","HIV_TB_Conifection")
drops<-c("weight","Age")
DefaultStd<-Amputation_MNAR_.30_1[ , !(names(Amputation_MNAR_.30_1) %in% drops)]
str(DefaultStd)
con.names = DefaultStd %>% select_if(is.character) %>% colnames()
print(con.names)
Amputation_MNAR_.30_1<-Amputation_MNAR_.30_1
Amputation_MNAR_.30_1[,con.names] = data.frame(apply(Amputation_MNAR_.30_1[con.names], 2
Amputation_MNAR_.30_1
dim(Amputation_MNAR_.30_1)
str(Amputation_MNAR_.30_1)
summary(Amputation_MNAR_.30_1)

Amputation_MNAR_.30_1_imp.LR <- mice(Amputation_MNAR_.30_1, m=3, maxit=10, seed=2268, pr
summary(Amputation_MNAR_.30_1_imp.LR)
densityplot(Amputation_MNAR_.30_1_imp.LR)
stripplot(Amputation_MNAR_.30_1_imp.LR)
head(Amputation_MNAR_.30_1_imp.LR$imp$HIV_TB_Conifection)
tail(Amputation_MNAR_.30_1_imp.LR$imp$HIV_TB_Conifection)
str(Amputation_MNAR_.30_1_imp.LR)
#lOOking at each imputed data set
```

```
anescomp1 <- mice::complete(Amputation_MNAR_.30_1_imp.LR, 1)
head(anescomp1)
md.pattern(anescomp1, plot = TRUE,rotate.names = TRUE)
anescomp2 <- mice::complete(Amputation_MNAR_.30_1_imp.LR, 2)
head(anescomp2)
md.pattern(anescomp2, plot = TRUE,rotate.names = TRUE)
anescomp3 <- mice::complete(Amputation_MNAR_.30_1_imp.LR, 3)
head(anescomp3)
md.pattern(anescomp3, plot = TRUE,rotate.names = TRUE)
#long matrix with stacked complete data
library(dplyr)
MNAR.30.LR_1 <- complete(Amputation_MNAR_.30_1_imp.LR, 'long')
#Dropping some cloumns
#https://www.listendata.com/2015/06/r-keep-drop-columns-from-data-frame.html
MNAR.30.LR_1<- MNAR.30.LR_1[,!(names(MNAR.30.LR_1) %in% c(".imp",".id"))]
dim(MNAR.30.LR_1)
view(MNAR.30.LR_1)
str(MNAR.30.LR_1)

set.seed(42)
default_idx = sample(nrow(MNAR.30.LR_1), 15000)
default_trn = MNAR.30.LR_1[default_idx, ]
default_tst = MNAR.30.LR_1[-default_idx, ]
dim(default_trn)
dim(default_tst)
#################
model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_2
summary(model_glm_2)
#Make predictions
head(predict(model_glm_2, type = "link"))
#the predicted probabilities
head(predict(model_glm_2, type = "response"))
#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the trainging classificat
calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
}
calc_class_err(actual = default_trn$default, predicted = model_glm_pred2)

train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
set.seed(100)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")
c(train_con_mat$overall["Accuracy"],
```

```
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])
#ROC Curves
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)

###MCAR .30
#1.)  Creating 30% Missingness Under MCAR
rm(list = ls())
set.seed(100)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
md.pattern(Default2)
attach(Default2)
Amputation_MCAR_.30_1<-ampute(Default2, prop = 0.30,freq = NULL, mech = "MCAR",std = TRU
Amputation_MCAR_.30_1

#ASSESSING MISSINGNESS
Amputation_MCAR_.30_1$amp
str(Amputation_MCAR_.30_1)
Amputation_MCAR_.30_1$patterns
Amputation_MCAR_.30_1$weights
Amputation_MCAR_.30_1$prop

Amputation_MCAR_.30_1<-as.data.frame(Amputation_MCAR_.30_1$amp)
P1<-function(x) {sum(is.na(x))/length(x)*100}

apply(Amputation_MCAR_.30_1,2,P1) #Gives proportion of missingness in each variable
md.pattern(Amputation_MCAR_.30_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MCAR_.30_1)# Bivariate comparison
aggr_plot <- aggr(Amputation_MCAR_.30_1, col=c('navyblue','yellow'), numbers=TRUE, sortV

library(Hmisc)
#2.1.a) 30% MCAR
str(Amputation_MCAR_.30_1)
drops<-c("weight","Age","HIV_TB_Conifection")
drops<-c("weight","Age")
DefaultStd<-Amputation_MCAR_.30_1[ , !(names(Amputation_MCAR_.30_1) %in% drops)]
str(DefaultStd)
con.names = DefaultStd %>% select_if(is.character) %>% colnames()
print(con.names)
Amputation_MCAR_.30_1<-Amputation_MCAR_.30_1
```

```
Amputation_MCAR_.30_1[,con.names] = data.frame(apply(Amputation_MCAR_.30_1[con.names], 2
Amputation_MCAR_.30_1
dim(Amputation_MCAR_.30_1)
str(Amputation_MCAR_.30_1)
summary(Amputation_MCAR_.30_1)

Amputation_MCAR_.30_1_imp.LR <- mice(Amputation_MCAR_.30_1, m=3, maxit=10, seed=2268, pr
summary(Amputation_MCAR_.30_1_imp.LR)
densityplot(Amputation_MCAR_.30_1_imp.LR)
stripplot(Amputation_MCAR_.30_1_imp.LR)
head(Amputation_MCAR_.30_1_imp.LR$imp$HIV_TB_Conifection)
tail(Amputation_MCAR_.30_1_imp.LR$imp$HIV_TB_Conifection)
str(Amputation_MCAR_.30_1_imp.LR)
#lOOking at each imputed data set
anescomp1 <- mice::complete(Amputation_MCAR_.30_1_imp.LR, 1)
head(anescomp1)
md.pattern(anescomp1, plot = TRUE,rotate.names = TRUE)
anescomp2 <- mice::complete(Amputation_MCAR_.30_1_imp.LR, 2)
head(anescomp2)
md.pattern(anescomp2, plot = TRUE,rotate.names = TRUE)
anescomp3 <- mice::complete(Amputation_MCAR_.30_1_imp.LR, 3)
head(anescomp3)
md.pattern(anescomp3, plot = TRUE,rotate.names = TRUE)
#long matrix with stacked complete data
library(dplyr)
MCAR.30.LR_1 <- complete(Amputation_MCAR_.30_1_imp.LR, 'long')
#Dropping some cloumns
#https://www.listendata.com/2015/06/r-keep-drop-columns-from-data-frame.html
MCAR.30.LR_1<- MCAR.30.LR_1[,!(names(MCAR.30.LR_1) %in% c(".imp",".id"))]
dim(MCAR.30.LR_1)
view(MCAR.30.LR_1)
str(MCAR.30.LR_1)

set.seed(42)
default_idx = sample(nrow(MCAR.30.LR_1), 15000)
default_trn = MCAR.30.LR_1[default_idx, ]
default_tst = MCAR.30.LR_1[-default_idx, ]
dim(default_trn)
dim(default_tst)
#################
model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_2
summary(model_glm_2)
#Make predictions
head(predict(model_glm_2, type = "link"))
#the predicted probabilities
head(predict(model_glm_2, type = "response"))
#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")
```

```
#After obtaining classifications, we calculate metrics such as the trainging classificat
calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
}
calc_class_err(actual = default_trn$default, predicted = model_glm_pred2)

train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
set.seed(100)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")
c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])
#ROC Curves
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)

#####USING LONG FORMAT OF THE IMPUTED DATA SET  50% Missingness
##MAR 0.50
#1.)  Creating 50% Missingness Under MAR
rm(list = ls())
set.seed(100)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
attach(Default2)
Amputation_MAR_.50_1<-ampute(Default2, prop = 0.50,freq = NULL, mech = "MAR",std = TRUE,
Amputation_MAR_.50_1

#ASSESSING MISSINGNESS
Amputation_MAR_.50_1$amp
str(Amputation_MAR_.50_1)
Amputation_MAR_.50_1$patterns
Amputation_MAR_.50_1$weights
Amputation_MAR_.50_1$prop

Amputation_MAR_.50_1<-as.data.frame(Amputation_MAR_.50_1$amp)
P1<-function(x) {sum(is.na(x))/length(x)*100}

apply(Amputation_MAR_.50_1,2,P1) #Gives proportion of missingness in each variable
```

```
md.pattern(Amputation_MAR_.50_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MAR_.50_1)# Bivariate comparison
aggr_plot <- aggr(Amputation_MAR_.50_1, col=c('navyblue','yellow'), numbers=TRUE, sortVa

library(Hmisc)
#2.1.a) 50% MAR
str(Amputation_MAR_.50_1)
drops<-c("weight","Age","HIV_TB_Conifection")
drops<-c("weight","Age")
DefaultStd<-Amputation_MAR_.50_1[ , !(names(Amputation_MAR_.50_1) %in% drops)]
str(DefaultStd)
con.names = DefaultStd %>% select_if(is.character) %>% colnames()
print(con.names)
Amputation_MAR_.50_1<-Amputation_MAR_.50_1
Amputation_MAR_.50_1[,con.names] = data.frame(apply(Amputation_MAR_.50_1[con.names], 2,
Amputation_MAR_.50_1
dim(Amputation_MAR_.50_1)
str(Amputation_MAR_.50_1)
summary(Amputation_MAR_.50_1)

Amputation_MAR_.50_1_imp.LR <- mice(Amputation_MAR_.50_1, m=3, maxit=10, seed=2268, prin
summary(Amputation_MAR_.50_1_imp.LR)
densityplot(Amputation_MAR_.50_1_imp.LR)
stripplot(Amputation_MAR_.50_1_imp.LR)
head(Amputation_MAR_.50_1_imp.LR$imp$HIV_TB_Conifection)
tail(Amputation_MAR_.50_1_imp.LR$imp$HIV_TB_Conifection)
str(Amputation_MAR_.50_1_imp.LR)
#lOOking at each imputed data set
anescomp1 <- mice::complete(Amputation_MAR_.50_1_imp.LR, 1)
head(anescomp1)
md.pattern(anescomp1, plot = TRUE,rotate.names = TRUE)
anescomp2 <- mice::complete(Amputation_MAR_.50_1_imp.LR, 2)
head(anescomp2)
md.pattern(anescomp2, plot = TRUE,rotate.names = TRUE)
anescomp3 <- mice::complete(Amputation_MAR_.50_1_imp.LR, 3)
head(anescomp3)
md.pattern(anescomp3, plot = TRUE,rotate.names = TRUE)
#long matrix with stacked complete data
library(dplyr)
MAR.50.LR_1 <- complete(Amputation_MAR_.50_1_imp.LR, 'long')
#Dropping some cloumns
#https://www.listendata.com/2015/06/r-keep-drop-columns-from-data-frame.html
MAR.50.LR_1<- MAR.30.LR_1[,!(names(MAR.50.LR_1) %in% c(".imp",".id"))]
dim(MAR.50.LR_1)
view(MAR.50.LR_1)
str(MAR.50.LR_1)

set.seed(42)
default_idx = sample(nrow(MAR.50.LR_1), 15000)
default_trn = MAR.50.LR_1[default_idx, ]
default_tst = MAR.50.LR_1[-default_idx, ]
```

```
dim(default_trn)
dim(default_tst)
################
model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_2
summary(model_glm_2)
#Make predictions
head(predict(model_glm_2, type = "link"))
#the predicted probabilities
head(predict(model_glm_2, type = "response"))
#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the trainging classificat
calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
}
calc_class_err(actual = default_trn$default, predicted = model_glm_pred2)

train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
set.seed(100)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")
c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])
#ROC Curves
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)

#######MNAR .50
#1.)  Creating 50% Missingness Under MNAR
rm(list = ls())
set.seed(100)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
attach(Default2)
Amputation_MNAR_.50_1<-ampute(Default2, prop = 0.50,freq = NULL, mech = "MNAR",std = TRU
```

```
Amputation_MNAR_.50_1

#ASSESSING MISSINGNESS
Amputation_MNAR_.50_1$amp
str(Amputation_MNAR_.50_1)
Amputation_MNAR_.50_1$patterns
Amputation_MNAR_.50_1$weights
Amputation_MNAR_.50_1$prop

Amputation_MNAR_.50_1<-as.data.frame(Amputation_MNAR_.50_1$amp)
P1<-function(x) {sum(is.na(x))/length(x)*100}

apply(Amputation_MNAR_.50_1,2,P1) #Gives proportion of missingness in each variable
md.pattern(Amputation_MNAR_.50_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MNAR_.50_1)# Bivariate comparison
aggr_plot <- aggr(Amputation_MNAR_.50_1, col=c('navyblue','yellow'), numbers=TRUE, sortV

library(Hmisc)
#2.1.a) 50% MNAR
str(Amputation_MNAR_.50_1)
drops<-c("weight","Age","HIV_TB_Conifection")
drops<-c("weight","Age")
DefaultStd<-Amputation_MNAR_.50_1[ , !(names(Amputation_MNAR_.50_1) %in% drops)]
str(DefaultStd)
con.names = DefaultStd %>% select_if(is.character) %>% colnames()
print(con.names)
Amputation_MNAR_.50_1<-Amputation_MNAR_.50_1
Amputation_MNAR_.50_1[,con.names] = data.frame(apply(Amputation_MNAR_.50_1[con.names], 2
Amputation_MNAR_.50_1
dim(Amputation_MNAR_.50_1)
str(Amputation_MNAR_.50_1)
summary(Amputation_MNAR_.50_1)

Amputation_MNAR_.50_1_imp.LR <- mice(Amputation_MNAR_.50_1, m=3, maxit=10, seed=2268, pr
summary(Amputation_MNAR_.50_1_imp.LR)
densityplot(Amputation_MNAR_.50_1_imp.LR)
stripplot(Amputation_MNAR_.50_1_imp.LR)
head(Amputation_MNAR_.50_1_imp.LR$imp$HIV_TB_Conifection)
tail(Amputation_MNAR_.50_1_imp.LR$imp$HIV_TB_Conifection)
str(Amputation_MNAR_.50_1_imp.LR)
#lOOking at each imputed data set
anescomp1 <- mice::complete(Amputation_MNAR_.50_1_imp.LR, 1)
head(anescomp1)
md.pattern(anescomp1, plot = TRUE,rotate.names = TRUE)
anescomp2 <- mice::complete(Amputation_MNAR_.50_1_imp.LR, 2)
head(anescomp2)
md.pattern(anescomp2, plot = TRUE,rotate.names = TRUE)
anescomp3 <- mice::complete(Amputation_MNAR_.50_1_imp.LR, 3)
head(anescomp3)
md.pattern(anescomp3, plot = TRUE,rotate.names = TRUE)
#long matrix with stacked complete data
```

```
library(dplyr)
MNAR.50.LR_1 <- complete(Amputation_MNAR_.50_1_imp.LR, 'long')
#Dropping some cloumns
#https://www.listendata.com/2015/06/r-keep-drop-columns-from-data-frame.html
MNAR.50.LR_1<- MNAR.50.LR_1[,!(names(MNAR.50.LR_1) %in% c(".imp",".id"))]
dim(MNAR.50.LR_1)
view(MNAR.50.LR_1)
str(MNAR.50.LR_1)

set.seed(42)
default_idx = sample(nrow(MNAR.50.LR_1), 15000)
default_trn = MNAR.50.LR_1[default_idx, ]
default_tst = MNAR.50.LR_1[-default_idx, ]
dim(default_trn)
dim(default_tst)
#################
model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_2
summary(model_glm_2)
#Make predictions
head(predict(model_glm_2, type = "link"))
#the predicted probabilities
head(predict(model_glm_2, type = "response"))
#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the trainging classificat
calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
}
calc_class_err(actual = default_trn$default, predicted = model_glm_pred2)

train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
set.seed(100)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")
c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])
#ROC Curves
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)

###MCAR .50
#1.)  Creating 50% Missingness Under MCAR
```

```
rm(list = ls())
set.seed(100)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
attach(Default2)
Amputation_MCAR_.50_1<-ampute(Default2, prop = 0.50,freq = NULL, mech = "MCAR",std = TRU
Amputation_MCAR_.50_1

#ASSESSING MISSINGNESS
Amputation_MCAR_.50_1$amp
str(Amputation_MCAR_.50_1)
Amputation_MCAR_.50_1$patterns
Amputation_MCAR_.50_1$weights
Amputation_MCAR_.50_1$prop

Amputation_MCAR_.50_1<-as.data.frame(Amputation_MCAR_.50_1$amp)
P1<-function(x) {sum(is.na(x))/length(x)*100}

apply(Amputation_MCAR_.50_1,2,P1) #Gives proportion of missingness in each variable
md.pattern(Amputation_MCAR_.50_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MCAR_.50_1)# Bivariate comparison
aggr_plot <- aggr(Amputation_MCAR_.50_1, col=c('navyblue','yellow'), numbers=TRUE, sortV

library(Hmisc)
#2.1.a) 50% MCAR
str(Amputation_MCAR_.50_1)
drops<-c("weight","Age","HIV_TB_Conifection")
drops<-c("weight","Age")
DefaultStd<-Amputation_MCAR_.50_1[ , !(names(Amputation_MCAR_.50_1) %in% drops)]
str(DefaultStd)
con.names = DefaultStd %>% select_if(is.character) %>% colnames()
print(con.names)
Amputation_MCAR_.50_1<-Amputation_MCAR_.50_1
Amputation_MCAR_.50_1[,con.names] = data.frame(apply(Amputation_MCAR_.50_1[con.names], 2
Amputation_MCAR_.50_1
dim(Amputation_MCAR_.50_1)
str(Amputation_MCAR_.50_1)
summary(Amputation_MCAR_.50_1)

Amputation_MCAR_.50_1_imp.LR <- mice(Amputation_MCAR_.50_1, m=3, maxit=10, seed=2268, pr
summary(Amputation_MCAR_.50_1_imp.LR)
densityplot(Amputation_MCAR_.50_1_imp.LR)
stripplot(Amputation_MCAR_.50_1_imp.LR)
head(Amputation_MCAR_.50_1_imp.LR$imp$HIV_TB_Conifection)
```

```
tail(Amputation_MCAR_.50_1_imp.LR$imp$HIV_TB_Conifection)
str(Amputation_MCAR_.50_1_imp.LR)
#lOOking at each imputed data set
anescomp1 <- mice::complete(Amputation_MCAR_.50_1_imp.LR, 1)
head(anescomp1)
md.pattern(anescomp1, plot = TRUE,rotate.names = TRUE)
anescomp2 <- mice::complete(Amputation_MCAR_.50_1_imp.LR, 2)
head(anescomp2)
md.pattern(anescomp2, plot = TRUE,rotate.names = TRUE)
anescomp3 <- mice::complete(Amputation_MCAR_.50_1_imp.LR, 3)
head(anescomp3)
md.pattern(anescomp3, plot = TRUE,rotate.names = TRUE)
#long matrix with stacked complete data
library(dplyr)
MCAR.50.LR_1 <- complete(Amputation_MCAR_.50_1_imp.LR, 'long')
#Dropping some cloumns
#https://www.listendata.com/2015/06/r-keep-drop-columns-from-data-frame.html
MCAR.50.LR_1<- MCAR.50.LR_1[,!(names(MCAR.50.LR_1) %in% c(".imp",".id"))]
dim(MCAR.50.LR_1)
view(MCAR.50.LR_1)
str(MCAR.50.LR_1)

set.seed(42)
default_idx = sample(nrow(MCAR.50.LR_1), 15000)
default_trn = MCAR.50.LR_1[default_idx, ]
default_tst = MCAR.50.LR_1[-default_idx, ]
dim(default_trn)
dim(default_tst)
#################
model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_2
summary(model_glm_2)
#Make predictions
head(predict(model_glm_2, type = "link"))
#the predicted probabilities
head(predict(model_glm_2, type = "response"))
#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the trainging classificat
calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
}
calc_class_err(actual = default_trn$default, predicted = model_glm_pred2)

train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
```

```
set.seed(100)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")
c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])
#ROC Curves
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)

#####USING LONG FORMAT OF THE IMPUTED DATA SET  80% Missingness
##MAR 0.80
#1.)  Creating 80% Missingness Under MAR
rm(list = ls())
set.seed(100)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
attach(Default2)
Amputation_MAR_.80_1<-ampute(Default2, prop = 0.80,freq = NULL, mech = "MAR",std = TRUE,
Amputation_MAR_.80_1

#ASSESSING MISSINGNESS
Amputation_MAR_.80_1$amp
str(Amputation_MAR_.80_1)
Amputation_MAR_.80_1$patterns
Amputation_MAR_.80_1$weights
Amputation_MAR_.80_1$prop

Amputation_MAR_.80_1<-as.data.frame(Amputation_MAR_.80_1$amp)
P1<-function(x) {sum(is.na(x))/length(x)*100}

apply(Amputation_MAR_.80_1,2,P1) #Gives proportion of missingness in each variable
md.pattern(Amputation_MAR_.80_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MAR_.80_1)# Bivariate comparison
aggr_plot <- aggr(Amputation_MAR_.80_1, col=c('navyblue','yellow'), numbers=TRUE, sortVa

library(Hmisc)
#2.1.a) 80% MAR
str(Amputation_MAR_.80_1)
drops<-c("weight","Age","HIV_TB_Conifection")
drops<-c("weight","Age")
DefaultStd<-Amputation_MAR_.80_1[ , !(names(Amputation_MAR_.80_1) %in% drops)]
str(DefaultStd)
```

```
con.names = DefaultStd %>% select_if(is.character) %>% colnames()
print(con.names)
Amputation_MAR_.80_1<-Amputation_MAR_.80_1
Amputation_MAR_.80_1[,con.names] = data.frame(apply(Amputation_MAR_.80_1[con.names], 2,
Amputation_MAR_.80_1
dim(Amputation_MAR_.80_1)
str(Amputation_MAR_.80_1)
summary(Amputation_MAR_.80_1)

Amputation_MAR_.80_1_imp.LR <- mice(Amputation_MAR_.80_1, m=3, maxit=10, seed=2268, prin
summary(Amputation_MAR_.80_1_imp.LR)
densityplot(Amputation_MAR_.80_1_imp.LR)
stripplot(Amputation_MAR_.80_1_imp.LR)
head(Amputation_MAR_.80_1_imp.LR$imp$HIV_TB_Conifection)
tail(Amputation_MAR_.80_1_imp.LR$imp$HIV_TB_Conifection)
str(Amputation_MAR_.80_1_imp.LR)
#lOOking at each imputed data set
anescomp1 <- mice::complete(Amputation_MAR_.80_1_imp.LR, 1)
head(anescomp1)
md.pattern(anescomp1, plot = TRUE,rotate.names = TRUE)
anescomp2 <- mice::complete(Amputation_MAR_.80_1_imp.LR, 2)
head(anescomp2)
md.pattern(anescomp2, plot = TRUE,rotate.names = TRUE)
anescomp3 <- mice::complete(Amputation_MAR_.80_1_imp.LR, 3)
head(anescomp3)
md.pattern(anescomp3, plot = TRUE,rotate.names = TRUE)
#long matrix with stacked complete data
library(dplyr)
MAR.80.LR_1 <- complete(Amputation_MAR_.80_1_imp.LR, 'long')
#Dropping some cloumns
#https://www.listendata.com/2015/06/r-keep-drop-columns-from-data-frame.html
MAR.80.LR_1<- MAR.80.LR_1[,!(names(MAR.80.LR_1) %in% c(".imp",".id"))]
dim(MAR.80.LR_1)
view(MAR.80.LR_1)
str(MAR.80.LR_1)

set.seed(42)
default_idx = sample(nrow(MAR.80.LR_1), 15000)
default_trn = MAR.80.LR_1[default_idx, ]
default_tst = MAR.80.LR_1[-default_idx, ]
dim(default_trn)
dim(default_tst)
################
model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_2
summary(model_glm_2)
#Make predictions
head(predict(model_glm_2, type = "link"))
#the predicted probabilities
head(predict(model_glm_2, type = "response"))
#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
```

```
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the trainging classificat
calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
}
calc_class_err(actual = default_trn$default, predicted = model_glm_pred2)

train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
set.seed(100)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")
c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])
#ROC Curves
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)

#######MNAR .80
#1.)  Creating 80% Missingness Under MNAR
rm(list = ls())
set.seed(100)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
attach(Default2)
Amputation_MNAR_.80_1<-ampute(Default2, prop = 0.80,freq = NULL, mech = "MNAR",std = TRU
Amputation_MNAR_.80_1

#ASSESSING MISSINGNESS
Amputation_MNAR_.80_1$amp
str(Amputation_MNAR_.80_1)
Amputation_MNAR_.80_1$patterns
Amputation_MNAR_.80_1$weights
Amputation_MNAR_.80_1$prop

Amputation_MNAR_.80_1<-as.data.frame(Amputation_MNAR_.80_1$amp)
P1<-function(x) {sum(is.na(x))/length(x)*100}
```

```
apply(Amputation_MNAR_.80_1,2,P1) #Gives proportion of missingness in each variable
md.pattern(Amputation_MNAR_.80_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MNAR_.80_1)# Bivariate comparison
aggr_plot <- aggr(Amputation_MNAR_.80_1, col=c('navyblue','yellow'), numbers=TRUE, sortV

library(Hmisc)
#2.1.a) 80% MNAR
str(Amputation_MNAR_.80_1)
drops<-c("weight","Age","HIV_TB_Conifection")
drops<-c("weight","Age")
DefaultStd<-Amputation_MNAR_.80_1[ , !(names(Amputation_MNAR_.80_1) %in% drops)]
str(DefaultStd)
con.names = DefaultStd %>% select_if(is.character) %>% colnames()
print(con.names)
Amputation_MNAR_.80_1<-Amputation_MNAR_.80_1
Amputation_MNAR_.80_1[,con.names] = data.frame(apply(Amputation_MNAR_.80_1[con.names], 2
Amputation_MNAR_.80_1
dim(Amputation_MNAR_.80_1)
str(Amputation_MNAR_.80_1)
summary(Amputation_MNAR_.80_1)

Amputation_MNAR_.80_1_imp.LR <- mice(Amputation_MNAR_.80_1, m=3, maxit=10, seed=2268, pr
summary(Amputation_MNAR_.80_1_imp.LR)
densityplot(Amputation_MNAR_.80_1_imp.LR)
stripplot(Amputation_MNAR_.80_1_imp.LR)
head(Amputation_MNAR_.80_1_imp.LR$imp$HIV_TB_Conifection)
tail(Amputation_MNAR_.80_1_imp.LR$imp$HIV_TB_Conifection)
str(Amputation_MNAR_.80_1_imp.LR)
#lOOking at each imputed data set
anescomp1 <- mice::complete(Amputation_MNAR_.80_1_imp.LR, 1)
head(anescomp1)
md.pattern(anescomp1, plot = TRUE,rotate.names = TRUE)
anescomp2 <- mice::complete(Amputation_MNAR_.80_1_imp.LR, 2)
head(anescomp2)
md.pattern(anescomp2, plot = TRUE,rotate.names = TRUE)
anescomp3 <- mice::complete(Amputation_MNAR_.80_1_imp.LR, 3)
head(anescomp3)
md.pattern(anescomp3, plot = TRUE,rotate.names = TRUE)
#long matrix with stacked complete data
library(dplyr)
MNAR.80.LR_1 <- complete(Amputation_MNAR_.80_1_imp.LR, 'long')
#Dropping some cloumns
#https://www.listendata.com/2015/06/r-keep-drop-columns-from-data-frame.html
MNAR.80.LR_1<- MNAR.80.LR_1[,!(names(MNAR.80.LR_1) %in% c(".imp",".id"))]
dim(MNAR.80.LR_1)
view(MNAR.80.LR_1)
str(MNAR.80.LR_1)

set.seed(42)
default_idx = sample(nrow(MNAR.80.LR_1), 15000)
```

```
default_trn = MNAR.80.LR_1[default_idx, ]
default_tst = MNAR.80.LR_1[-default_idx, ]
dim(default_trn)
dim(default_tst)
################
model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_2
summary(model_glm_2)
#Make predictions
head(predict(model_glm_2, type = "link"))
#the predicted probabilities
head(predict(model_glm_2, type = "response"))
#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the trainging classificat
calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
}
calc_class_err(actual = default_trn$default, predicted = model_glm_pred2)

train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
set.seed(100)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")
c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])
#ROC Curves
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)

###MCAR .80
#1.)  Creating 80% Missingness Under MCAR
rm(list = ls())
set.seed(100)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
```

```
attach(Default2)
Amputation_MCAR_.80_1<-ampute(Default2, prop = 0.80,freq = NULL, mech = "MCAR",std = TRU
Amputation_MCAR_.80_1

#ASSESSING MISSINGNESS
Amputation_MCAR_.80_1$amp
str(Amputation_MCAR_.80_1)
Amputation_MCAR_.80_1$patterns
Amputation_MCAR_.80_1$weights
Amputation_MCAR_.80_1$prop

Amputation_MCAR_.80_1<-as.data.frame(Amputation_MCAR_.80_1$amp)
P1<-function(x) {sum(is.na(x))/length(x)*100}

apply(Amputation_MCAR_.80_1,2,P1) #Gives proportion of missingness in each variable
md.pattern(Amputation_MCAR_.80_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MCAR_.80_1)# Bivariate comparison
aggr_plot <- aggr(Amputation_MCAR_.80_1, col=c('navyblue','yellow'), numbers=TRUE, sortV

library(Hmisc)
#2.1.a) 80% MCAR
str(Amputation_MCAR_.80_1)
drops<-c("weight","Age","HIV_TB_Conifection")
drops<-c("weight","Age")
DefaultStd<-Amputation_MCAR_.80_1[ , !(names(Amputation_MCAR_.80_1) %in% drops)]
str(DefaultStd)
con.names = DefaultStd %>% select_if(is.character) %>% colnames()
print(con.names)
Amputation_MCAR_.80_1<-Amputation_MCAR_.80_1
Amputation_MCAR_.80_1[,con.names] = data.frame(apply(Amputation_MCAR_.80_1[con.names], 2
Amputation_MCAR_.80_1
dim(Amputation_MCAR_.80_1)
str(Amputation_MCAR_.80_1)
summary(Amputation_MCAR_.80_1)

Amputation_MCAR_.80_1_imp.LR <- mice(Amputation_MCAR_.80_1, m=3, maxit=10, seed=2268, pr
summary(Amputation_MCAR_.80_1_imp.LR)
densityplot(Amputation_MCAR_.80_1_imp.LR)
stripplot(Amputation_MCAR_.80_1_imp.LR)
head(Amputation_MCAR_.80_1_imp.LR$imp$HIV_TB_Conifection)
tail(Amputation_MCAR_.80_1_imp.LR$imp$HIV_TB_Conifection)
str(Amputation_MCAR_.80_1_imp.LR)
#lOOking at each imputed data set
anescomp1 <- mice::complete(Amputation_MCAR_.80_1_imp.LR, 1)
head(anescomp1)
md.pattern(anescomp1, plot = TRUE,rotate.names = TRUE)
anescomp2 <- mice::complete(Amputation_MCAR_.80_1_imp.LR, 2)
head(anescomp2)
md.pattern(anescomp2, plot = TRUE,rotate.names = TRUE)
anescomp3 <- mice::complete(Amputation_MCAR_.80_1_imp.LR, 3)
head(anescomp3)
```

```
md.pattern(anescomp3, plot = TRUE,rotate.names = TRUE)
#long matrix with stacked complete data
library(dplyr)
MCAR.80.LR_1 <- complete(Amputation_MCAR_.80_1_imp.LR, 'long')
#Dropping some cloumns
#https://www.listendata.com/2015/06/r-keep-drop-columns-from-data-frame.html
MCAR.80.LR_1<- MCAR.80.LR_1[,!(names(MCAR.80.LR_1) %in% c(".imp",".id"))]
dim(MCAR.80.LR_1)
View(MCAR.80.LR_1)
str(MCAR.80.LR_1)

set.seed(42)
default_idx = sample(nrow(MCAR.80.LR_1), 15000)
default_trn = MCAR.80.LR_1[default_idx, ]
default_tst = MCAR.80.LR_1[-default_idx, ]
dim(default_trn)
dim(default_tst)
################
model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_2
summary(model_glm_2)
#Make predictions
head(predict(model_glm_2, type = "link"))
#the predicted probabilities
head(predict(model_glm_2, type = "response"))
#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the trainging classificat
calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
}
calc_class_err(actual = default_trn$default, predicted = model_glm_pred2)

train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
set.seed(100)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")
c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])
#ROC Curves
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)
```

```
################################### MLE
####Missing Data:       7%
################################### MLE
rm(list = ls())
library(norm)
library(mice)
library(dplyr)
library(VIM)
library(mitools)
library(MASS)
library(mix)
library(ISLR)
##########Missing Data: MAR 7%
#1.)  Creating 30% Missingness Under MAR
rm(list = ls())
set.seed(100)
Gender<-sample(c(1,2), size = 10000, replace = TRUE, prob = c(0.29, 0.71))
Age<-round(runif(10000, 18, 80))
weight<-round(runif(10000, 45, 100))
xb <- -9 + 3.5*Gender + 0.2*Age-0.2*weight
#xb <- -9 + 3.5*Gender-0.2*weight
p <- 1/(1 + exp(-xb))
HIV_TB_Conifection <- rbinom(n = 10000, size = 1, prob = p)
Default2<-cbind(HIV_TB_Conifection,Gender,Age,weight)
Default2<-as.data.frame(Default2)
attach(Default2)
View(Default2)
Amputation_MAR_.30_1<-ampute(Default2, prop = 0.8,freq = NULL, mech = "MAR",std = TRUE,
#View(Amputation_MAR_.30_1)

#ASSESSING MISSINGNESS
Amputation_MAR_.30_1$amp
str(Amputation_MAR_.30_1)
Amputation_MAR_.30_1$patterns
Amputation_MAR_.30_1$weights
Amputation_MAR_.30_1$prop

Amputation_MAR_.30_1<-as.data.frame(Amputation_MAR_.30_1$amp)
P1<-function(x) {sum(is.na(x))/length(x)*100}

apply(Amputation_MAR_.30_1,2,P1) #Gives proportion of missingness in each variable
md.pattern(Amputation_MAR_.30_1, plot = TRUE,rotate.names = TRUE)
md.pairs(Amputation_MAR_.30_1)# Bivariate comparison
aggr_plot <- aggr(Amputation_MAR_.30_1, col=c('navyblue','yellow'), numbers=TRUE, sortVa

library(Hmisc)
#2.1.a) 30% MAR
str(Amputation_MAR_.30_1)
drops<-c("weight","Age","HIV_TB_Conifection")
drops<-c("weight","Age")
DefaultStd<-Amputation_MAR_.30_1[ , !(names(Amputation_MAR_.30_1) %in% drops)]
```

```
str(DefaultStd)
con.names = DefaultStd %>% select_if(is.character) %>% colnames()
print(con.names)
Amputation_MAR_.30_1<-Amputation_MAR_.30_1
Amputation_MAR_.30_1[,con.names] = data.frame(apply(Amputation_MAR_.30_1[con.names], 2,
Amputation_MAR_.30_1
dim(Amputation_MAR_.30_1)
str(Amputation_MAR_.30_1)
summary(Amputation_MAR_.30_1)
#Single_imp_Data_MAR_.30<-as.data.frame(cbind(Age,weight,HIV_TB_Conifection,Gender))
#cols <- c("Gender")
#Single_imp_Data_MAR_.30[,cols] <- data.frame(apply(Single_imp_Data_MAR_.30[cols], 2, as
#Single_imp_Data_MAR_.30$HIV_TB_Conifection[Single_imp_Data_MAR_.30$HIV_TB_Conifection =
#Single_imp_Data_MAR_.30$HIV_TB_Conifection[Single_imp_Data_MAR_.30$HIV_TB_Conifection =
#Single_imp_Data_MAR_.30
#**********************************
library(mix)
library(rngtools)
View(Amputation_MAR_.30_1)
str(Amputation_MAR_.30_1)
Amputation_MAR_.30_1$HIV_TB_Conifection<-as.factor(Amputation_MAR_.30_1$HIV_TB_Conifecti
Amputation_MAR_.30_1$Gender<-as.factor(Amputation_MAR_.30_1$Gender)
#Amputation_MAR_.30_1[, 1:2] <- sapply(Amputation_MAR_.30_1[, 1:2],as.factor)
str(Amputation_MAR_.30_1)
md.pattern(Amputation_MAR_.30_1)
s <- prelim.mix(Amputation_MAR_.30_1,3) # do preliminary manipulations
thetahat <- em.mix(s) # ML estimate for unrestricted model
#set.seed(100)
rngseed(12345678) # set random number generator seed
newtheta <- da.mix(s,thetahat,steps=100,showits=TRUE) # data augmentation
MLE_MAR_.07 <- imp.mix(s, newtheta, Amputation_MAR_.07_1) # impute under newtheta

data(stlouis)
View(stlouis)
dim(stlouis)
str(stlouis)
md.pattern(stlouis)
s <- prelim.mix(stlouis,3) # preliminary manipulations
thetahat <- em.mix(s) # find ML estimate
rngseed(1234567) # set random number generator seed
newtheta <- da.mix(s, thetahat, steps=100, showits=TRUE) # take 100 steps
ximp1 <- imp.mix(s, newtheta) # impute under newtheta
md.pattern(ximp1)




data(stlouis)
s <- prelim.mix(Amputation_MAR_.30_1,3) # preliminary manipulations
margins <- c(1,2,3) # saturated loglinear model
design <- diag(rep(1,12)) # identity matrix, D=no of cells
```

```
thetahat <- ecm.mix(s,margins,design) # should be same as em.mix(s)
loglik.mix(s,thetahat) # loglikelihood at thetahat

library(imp4p)

#Simulating data
res.sim=sim.data(nb.pept=2000,nb.miss=600,nb.cond=2);

#Imputation of missing values with the mle algorithm
dat.mle=impute.mle(tab=Amputation_MAR_.30_1,conditions=Amputation_MAR_.30_1$condition);
#http://uu.diva-portal.org/smash/get/diva2:940656/FULLTEXT01.pdf
#https://medium.com/coinmonks/dealing-with-missing-data-using-r-3ae428da2d17
#http://www.statisticalhorizons.com/wp-content/uploads/MissingDataByML.pdf
#https://cran.r-project.org/web/packages/mix/mix.pdf
#CONVERTING BACK TO ORIGINAL DATA TYPES
library(tibble)
as_tibble(MLE_MAR_.07)

MLE_MAR_.07<-data.frame(MLE_MAR_.07)
library(dplyr)

drops<-c("balance","income")
DefaultStd<-MLE_MAR_.07[ , !(names(MLE_MAR_.07) %in% drops)]
str(DefaultStd)
con.names = DefaultStd %>% select_if(is.numeric) %>% colnames()
print(con.names)

MLE_MAR_.07<-MLE_MAR_.07
MLE_MAR_.07[,con.names] = data.frame(apply(MLE_MAR_.07[con.names], 2, as.factor))
str(MLE_MAR_.07[,con.names])

MLE_MAR_.07
dim(MLE_MAR_.07)
str(MLE_MAR_.07)
view(MLE_MAR_.07)
md.pattern(MLE_MAR_.07)

set.seed(42)
default_idx = sample(nrow(MLE_MAR_.07), 5000)
default_trn = MLE_MAR_.07[default_idx, ]
default_tst = MLE_MAR_.07[-default_idx, ]
dim(default_trn)
dim(default_tst)
str(default_trn)
View(default_trn)

################
model_glm_2 = glm(HIV_TB_Conifection ~., data = default_trn, family = "binomial")
model_glm_2
summary(model_glm_2)
```

```
#Make predictions
head(predict(model_glm_2, type = "link"))
#the predicted probabilities
head(predict(model_glm_2, type = "response"))

#We obtain classifications, by comparing to the correct cutoff value with an ifelse() st
model_glm_pred2 = ifelse(predict(model_glm_2, type = "link") > 0, "Yes", "No")
model_glm_pred2 = ifelse(predict(model_glm_2, type = "response") > 0.5, "Yes", "No")

#After obtaining classifications, we calculate metrics such as the trainging classificat
calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
}
calc_class_err(actual = default_trn$default, predicted = model_glm_pred2)

train_tab = table(predicted = model_glm_pred2, actual = default_trn$HIV_TB_Conifection)
dimnames(train_tab)[[2]] = c("No","Yes") # This code renames the 1, 2 to No, Yes
train_tab

library(caret)
set.seed(100)
train_con_mat = confusionMatrix(train_tab, positive = "Yes")
c(train_con_mat$overall["Accuracy"],
  train_con_mat$byClass["Sensitivity"],
  train_con_mat$byClass["Specificity"])

#ROC Curves
library(pROC)
test_prob = predict(model_glm_2, newdata = default_tst, type = "response")
test_roc = roc(default_tst$HIV_TB_Conifection ~ test_prob, plot = TRUE, print.auc = TRUE
as.numeric(test_roc$auc)
```