*This is the main submission document**. Save and rename this document filename with your registered full name as Prefix before submission.***

| | |
|---|---|
| Class | 6 |
| Full Name | Lim Qing Rui |
| Matriculation Number | U2010816G |

** : Delete and replace as appropriate.*

## Declaration of Academic Integrity

By submitting this assignment for assessment, I declare that this submission is my own work, unless otherwise quoted, cited, referenced or credited. I have read and understood the Instructions to CBA.PDF provided and the Academic Integrity Policy.

I am aware that failure to act in accordance with the University's Academic Integrity Policy may lead to the imposition of penalties which may include the requirement to revise and resubmit an assignment, receiving a lower grade, or receiving an F grade for the assignment; suspension from the University or termination of my candidature.

I consent to the University copying and distributing any or all of my work in any form and using third parties to verify whether my work contains plagiarised material, and for quality assurance purposes.

***Please insert an "X" within the square brackets below to indicate your selection.***

**[X] I have read and accept the above.**

## Table of Contents

# Answer to Q1:

**1a) Import data as data1 and ensure that all categorical data are treated as categories instead of integers, numeric or text string characters. Show your code.**

In this dataset, identified categorical data columns are (i) group, (ii) outcome, (iii) gendera, (iv) hypertensive, (v) atrialfibrillation, (vi) CHD with no MI, (vii) diabetes, (viii) deficiencyanemias, (ix) depression, (X) Hyperlipemia, (xi) Renal failure, (xii) COPD.

```
summary(data1) # for this data, na values will not be removed as NA values will be replaced in subsequent code
nrow(data1) # 1177 rows

length(unique(data1$ID)) # 1177 unique rows/ID, so ID does not have to be categorised
```

ID in data1 was not identified as a categorical as we identified 1177 unique data points (rows) and this is same as the unique variables length in ID. Hence there is no need to create 1177 unique ID categories.

Str() used to check each column class type and ensure that the 12 variables identified are factorised into categorical data.

```
library(tidyverse)
library(magrittr) # library to mutate (change column class type) multiple columns
cols <- c("group", "outcome", "gendera", "hypertensive","atrialfibrillation", "CHD with no MI", "diabetes",
          "deficiencyanemias","depression", "Hyperlipemia", "Renal failure", "COPD") #12 variables for factorisation

data1 %<>% mutate_at(cols, factor) # Assignment pipe %<>% to assign columns into dataframe to update category
str(data1) # check data type and 12 columns are factorised
```

Magrittr is a library that was used to allow the assignment pipe operator %<>% to assign column names identified above for factorisation in data1 dataframe.

**1b) Purpose of Derivation group and Validation group and how is this reflected in the dataset?**

The purpose of the Derivation Group (n= 825, 70% of 1177 patients, group = "1") was used as a training data to train the models such as XGBoost and LASSO, while the Validation Group (n= 352, 30% of 1177 patients, group = "2") was used to test the model's predictability and results.

This is reflected in the dataset 'data01.csv' as the dataset contains 1177 patients (1177 Unique IDs) which is similar to the study performed by Li F, et al (2021).

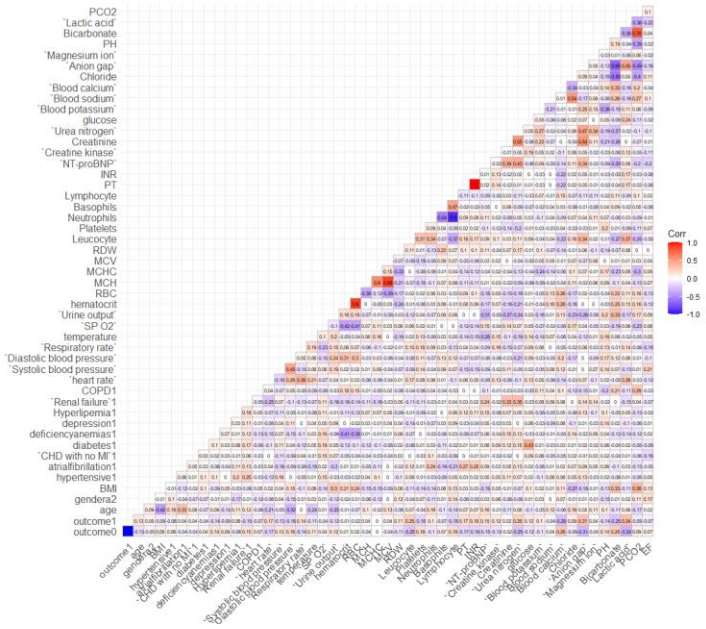
**1c) Show a table of missing value counts that shows all those variables that has missing values, it's data type (numeric, integer, factor, character, etc.) and it's missing value count**

Refer to R file for the codes (Line 40 to 69). The completed table is as follows (20 variables with NA values):

| | Variable.Name | Data.Type | NA.Count |
|---|---|---|---|
| 1 | Basophils | numeric | 259 |
| 2 | Blood calcium | numeric | 1 |
| 3 | BMI | numeric | 215 |
| 4 | Creatine kinase | numeric | 165 |
| 5 | Diastolic blood pressure | numeric | 16 |
| 6 | glucose | numeric | 18 |
| 7 | heart rate | numeric | 13 |
| 8 | INR | numeric | 20 |
| 9 | Lactic acid | numeric | 229 |
| 10 | Lymphocyte | numeric | 145 |
| 11 | Neutrophils | numeric | 144 |
| 12 | outcome | factor | 1 |
| 13 | PCO2 | numeric | 294 |
| 14 | PH | numeric | 292 |
| 15 | PT | numeric | 20 |
| 16 | Respiratory rate | numeric | 13 |
| 17 | SP O2 | numeric | 13 |
| 18 | Systolic blood pressure | numeric | 16 |
| 19 | temperature | numeric | 19 |
| 20 | Urine output | numeric | 36 |

**1d) Explore data1. Produce charts, tables or/and statistics to explain 3 interesting findings.**

Finding 1: Most variables exhibited low correlation (Positive or Negative) in Trainset data

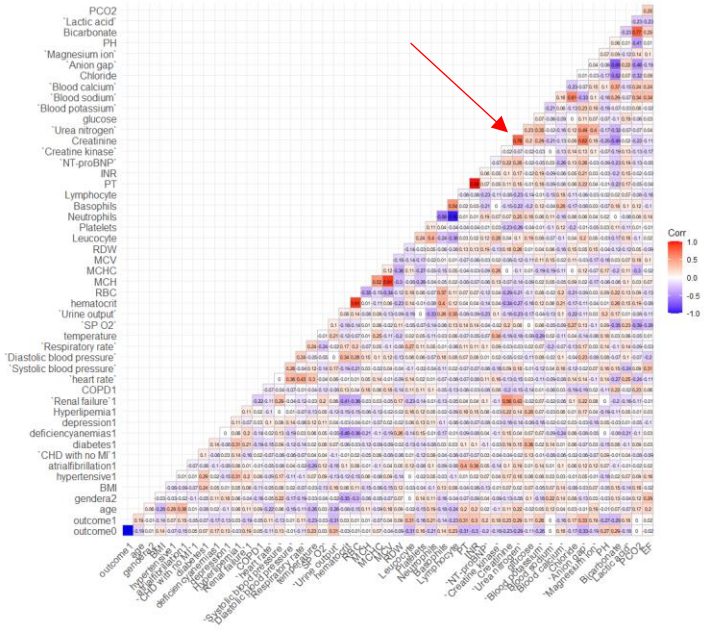



In the correlation plot, we can see that most variables used to investigate the outcome of Heart Failure showed low inter-variable correlation for trainset data. This would increase reliability of results as multicollinearity of the dependent variables will not affect the stability of the models, where positive change in one variable results in a positive change in another variable and cause the model to fluctuate significantly with similar variables preferred and viewed as important in the Machine Learning model. However, strong positive (>0.75, bright red squares) and negative correlations (< -0.75, bright blue squares) can be seen for the following data:

| **Variable 1** | **Variable 2** | **Quantitative correlation** | **Qualitative correlation** |
|---|---|---|---|
| Haematocrit | RBC | 0.90 | Strong Positive |
| MCH | MCV | 0.88 | Strong Positive |
| PT | INR | 1.00 | Strong Positive |
| Bicarbonate | PCO2 | 0.78 | Strong Positive |
| Neutrophils | Lymphocytes | 0.90 | Strong Negative |

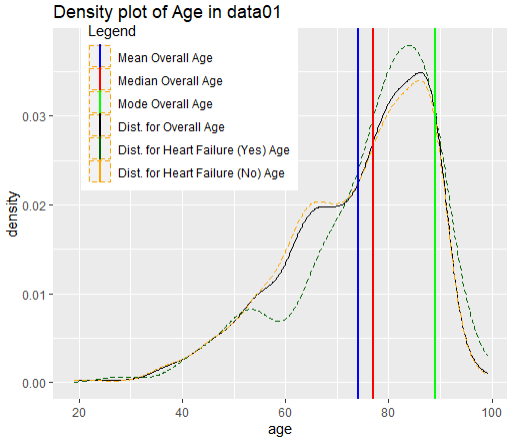
Based on the table above, we can see that Variables 1 and 2 are highly related. For example, Neutrophils and Lymphocytes are a form of White Blood Cells, which are activated when infection in the body occurs. *(Cancer.org., n.d)* MCH (Mean Corpuscular Haemoglobin) and MCV (Mean Corpuscular Volume) are also similar as larger red blood cells (MCV) tend to have greater Haemoglobin content (MCH). *(Jill S.S., 2021)* The employment of these variables in Machine Learning could lead to skewed variable predictor selection across models.

Finding 2: Creatinine and Urea Nitrogen displayed strong positive correlation in Trainset data




In addition to the variables in the trainset data, Creatinine and Urea nitrogen showed high positive correlation of 0.76 (Red Arrow). From additional research, these 2 variables are highly related as they are end of metabolism products and will usually occur when the other variable increases. *(Higgins C., 2016)* However, this could result in high bias in predictability of results and lower predictability and applicability of data for prediction onto the model.

Finding 3: Skewness of participants age group selected in dataset




From the graph above, we can see that the distribution of all participants selected for the study is a left skewed (negatively skewed curve). This is seen as the Mean age is 74.06 years, median age is 77 years, and the modal age is 89 years. The usage of participants with high age groups can result in a possible illusory correlation that age may be highly correlated to Heart Failure death. Hence, the skewness does not justify the Death Occurrence risk that may occur in younger participants.

This can be further shown as there are there are higher occurrences of older people who passed away from Heart Failure (Dark Green dashed line) than in people who survived. This could lead to the selection of predictors that Age is an important determining factor of Heart Failure death in XGBoost and LASSO as shown in the Li F., et al (2021) research paper.

## Answer to Q2:

**2a) Replace all missing values and check that data2 has no missing values. Show the code used to check for missing value count and the output.**

```
##### Question 2 #####

### a)
data2 = data1
str(data2)

# na.roughfix used in RF package to impute continuous variables by median and categorical variable by mode
data2 = na.roughfix(data2)
summary(data2)

colSums(is.na(data2)) # no missing values in each column for data2
View(data2) #Output after imputing respective values into NA values
```

In lines 125 to 136 of the code, data1 is stored in data2 as a dataframe. Str(data2) is used to check for the class type of each column in data2. na.roughfix is extracted from the randomForest library to encode NA continuous variables to median and mode if the NA variable is categorical. To check the output and ensure that no NA values are in data2, colSums(is.na(data2)) is used to check each column (Red box). The output is as follows:

```
> colSums(is.na(data2)) # no missing values in each column for data2
                group                   ID              outcome                  age              gendera                  BMI 
                    0                    0                    0                    0                    0                    0 
         hypertensive      atrialfibrillation       CHD with no MI             diabetes     deficiencyanemias           depression 
                    0                    0                    0                    0                    0                    0 
         Hyperlipemia         Renal failure                 COPD           heart rate Systolic blood pressure Diastolic blood pressure 
                    0                    0                    0                    0                    0                    0 
   Respiratory rate          temperature                SP O2           Urine output           hematocrit                  RBC 
                    0                    0                    0                    0                    0                    0 
                  MCH                 MCHC                  MCV                  RDW           Leucocyte           Platelets 
                    0                    0                    0                    0                    0                    0 
          Neutrophils           Basophils          Lymphocyte                   PT                  INR           NT-proBNP 
                    0                    0                    0                    0                    0                    0 
       Creatine kinase           Creatinine       Urea nitrogen              glucose     Blood potassium        Blood sodium 
                    0                    0                    0                    0                    0                    0 
        Blood calcium             Chloride          Anion gap      Magnesium ion                   PH          Bicarbonate 
                    0                    0                    0                    0                    0                    0 
          Lactic acid                 PCO2                   EF 
                    0                    0                    0 
```

**b) Produce a trainset using group = 1, remove group and ID from trainset and show the proportion of cases who died vs alive**

```
### b)
trainset = subset(data2, data2$group =='1')
summary(trainset) #Verify that there are 825 individuals in trainset
trainset = subset(trainset,select= -c(group, ID)) # subset to remove group and ID columns
table1 = as.table(table(trainset$outcome))
prop.table(table1) #Proportion of outcomes for alive vs dead
```

Trainset obtained from data2 using the subset() function. The negative vector "-c()" in the third line is used to remove group and ID from the trainset data. The outcome column is stored in table1 to derive the counted observations of outcome and create the proportion in a table using prop.table(). The proportions are shown below:

```
> prop.table(table1) #Proportion of outcomes for alive vs dead

        0         1 
0.8593939 0.1406061 
```

Cases who are alive are denoted as '0' and comprised of 85.94% (0.8594) while cases who are dead are denoted as '1' and comprised of 14.06% (0.1406)

**c) Produce a testset using group = 2, remove group and ID from trainset and show the proportion of cases who died vs alive**

test set obtained in similar method to trainset. The proportions of alive and dead cases are shown below:

```
> prop.table(table2) #Proportion of outcomes for alive vs dead

        0         1 
0.8778409 0.1221591 
```

Cases who are alive are denoted as '0' and comprised of 87.78% (0.8778) while cases who are dead are denoted as '1' and comprised of 12.22% (0.1222)

# Answer to Q3:

**3. Briefly explain (in bullet points) how you would compute the GWTG predicted outcome on the dataset**

- The dataset consists of 1177 patients, which will be split into a 70-30 train-test split where 825 patients (group = 1) will be used for training the model and 352 patients (group = 2) will be used as testset data.
- Key predictors of Heart failure to be extracted from the data would be information about the patient's age, Systolic Blood pressure, Blood Urea Nitrogen, Heart Rate, serum sodium, Chronic obstructive pulmonary disease (COPD) history and non-African American ethnicity* to predict the risk of in-hospital mortality for patients hospitalised with HF. The variables listed above will be obtained and parsed into a new dataframe.

  * Race is assumed to be non-black in the data given, although it is used as a predictor of HF.
- Continuous variables will be passed through the model as splines. Once the trainset data is parsed through the GWTG model with Bootstrapping with 500 samples (as we do not know the stableness of the model), we can use the predict() function to parse the testset data into the model containing the trainset data.
- Similar to the 2009 Paper provided, the risk score for each variable will be scored with its respective bin range, and this will be parsed into a new dataframe for GWTG. *(See Appendix B for Bin ranges in Codes and from the 2009 Paper)*
- trainGWTG$ttlscore used to store the total score across all variables identified for Heart Failure. Ttlscore is factored into mortality (outcome = 1) and no mortality (outcome = 0) by setting probability of death at 50% which occurs when ttlscore>= 79
- We can obtain a confusion matrix for the false positive, false, negative, true positive and true negative prediction of the predicted data onto the original data and determine the proportion of predicted Heart failure results using the confusionMatrix() function in 'caret' package with positive = "1" set to establish patients who are at risk of mortality on the trainset and testset data.

## Answer to Q4:

**4) Briefly explain (in bullet points) how you would compute the Nomogram predicted outcome on the dataset.**

- The dataset consists of 1177 patients, which will be split into a 70-30 train-test split where 825 patients (group = 1) will be used for training the model and 352 patients (group = 2) will be used as testset data.
- Variables significantly associated with in-hospital mortality for Heart Failure will be used for multivariate binary logistic regression in the XGBoost-Nomogram equation (See bolded statement below).
- We can impute the values of each row in the data to the following equation (XGBoost equation used here as it is used for Nomogram computation):

  **odds of mortality = 10 ^ (4.62536 + 0.24559 × anion gap + 0.61542 × lactate−1.04993 × calcium+ 0.02687 × BUN −1.76330 × CKD − 0.05633 × DBP)**

- After that, we will set the threshold for patients who survive to be 50% and predict that patients who do not survive have odds of mortality > 0.5 (outcome = 1). *(See Appendix C for Bin ranges in Codes and from the 2021 Paper)*
- To obtain the confusion matrix for the false positive, false, negative, true positive and true negative prediction of the predicted data onto the original data to determine the proportion of predicted Heart failure results, the confusionMatrix() function in 'caret' package will be used with positive = "1" set to establish patients who are at risk of mortality on the trainset and testset data.

## Answer to Q5:

**5) Show the table of trainset errors (false positive rate, false negative rate, overall error) from Logistic Regression with Backward Elimination, Random Forest with default settings, GWTG and Nomogram. Briefly state your findings.**

| | Model | FPR | FNR | Err |
|---|---|---|---|---|
| 1 | Logistic Reg (BE) | 0.02257 | 0.60345 | 0.10424 |
| 2 | Random Forest | 0.00282 | 0.89655 | 0.12848 |
| 3 | GWTG | 0 | 1 | 0.14061 |
| 4 | Nomogram | 0.00987 | 0.74138 | 0.11273 |

Logistic Regression with Backward elimination has the least overall error and least False Negative Rate of 0.10424 and 0.60345 respectively. However, GWTG, amongst the 4 techniques, has the lowest False positive rate of 0 while Random Forest* is next with a FPR of 0.00282. The GWTG is unable to determine any False Positives rates (0) and has high false negative rates (1) showing that it has low sensitivity and high specificity in data detection, which is supplemented with its highest overall error of 0.14061. Nomogram compares moderately for trainset errors with its FPR, FNR and Overall Errors being in the mid-range across all 4 models.

*Random Forest Variable Importance hyperparameter has been turned on to standardise across all questions (Q5 to 8)

## Answer to Q6:

**6) Show the table of testset errors (false positive rate, false negative rate, overall error) from Logistic Regression with Backward Elimination, Random Forest with default settings, GWTG and Nomogram. Briefly state your findings.**

| | Model | FPR | FNR | Err |
|---|---|---|---|---|
| 1 | Logistic Reg (BE) | 0.03883 | 0.67442 | 0.11648 |
| 2 | Random Forest | 0 | 0.90698 | 0.1108 |
| 3 | GWTG | 0 | 1 | 0.12216 |
| 4 | Nomogram | 0.01618 | 0.74419 | 0.10511 |

In testset error, Logistic Regression with Backward elimination has the least False Negative Rate of 0.67442 and highest False Positive Rate of 0.03883. However, GWTG, amongst the 4 techniques, has the lowest False positive rate of 0 while Random Forest is next with a same False positive rate and an FPR of 0.00324. GWTG is unable to determine any False Positives rates (0) and has high false negative rates (1) showing that it has low accuracy and precision in data detection in the test set data compared to the other models which can be shown with its highest overall testset error of 0.12216. Similar to the trainset error, Nomogram testset error performs moderately to the other 3 models, with error range in the mid-range for FPR, FNR and Overall Error.

## Answer to Q7:

**7) Balance the trainset by sampling from the majority to obtain 50-50 distribution of alive vs death in the trainset. Show the table of testset errors (false positive rate, false negative rate, overall error) from Logistic Regression with Backward Elimination and Random Forest with default settings. Briefly state your findings.**

| | Model | FPR | FNR | Err |
|---|---|---|---|---|
| 1 | Logistic Reg (BE) | 0.30421 | 0.32558 | 0.30682 |
| 2 | Random Forest | 0.28803 | 0.23256 | 0.28125 |

The balanced trainset data onto the model resulted in a higher testset error prediction. It caused logistic regression with backward elimination and random forest to have an increased overall error. The balanced sampling method caused both Logistic Regression and Random Forest to have an increased False Positive Rate of 0.30421 and 0.28803 respectively, while False Negative Rates are 0.32558 and 0.23256 respectively. However, the overall error of Random Forest (0.28125) is lower than Logistic Regression (0.30682), and this could represent **higher accuracy and precision of Random Forest with higher stability than Logistic Regression in Mortality prediction.**

## Answer to Q8:

**8) Extract the top 20 variable importance (permutation approach) from Random Forest trained on balanced trainset and fit them as predictors into logistic regression. Append the testset results and show the table. Is this model superior than stand-alone logistic regression (with backward elimination) or Random Forest? Briefly state your findings.**

| | Model | FPR | FNR | Err |
|---|---|---|---|---|
| 1 | Logistic Reg (BE) | 0.30421 | 0.32558 | 0.30682 |
| 2 | Random Forest | 0.28803 | 0.23256 | 0.28125 |
| 3 | RF VarImpt into Logistic Reg | 0.32362 | 0.39535 | 0.33239 |

The top 20 variables with Variable Importance from Random Forest with the balanced trainset data are shown in **Appendix D**. For Logistic Regression with Random Forest Variable Importance, it is not a superior model than the Logistic Regression with Backward Elimination or Random Forest. This is because it has a high overall error rate of 0.33239, False Positive Rate of 0.32362 and False Negative Rate of 0.39535, resulting **in lower accuracy and precision than the other models**.

## Answer to Q9:

**9) A hospital in Singapore is thinking of using a risk scoring system to assess ICU patient mortality. What is your recommendation?**

Across all the models within the error comparison, we can see that Random Forest performed consistently, being either the model with the least overall error or displaying lower error than 1 or 2 other models. In addition, Random Forest displayed the least divergence and discrepancies amongst the False Positive and False Negative Error rates across the 4 testing scenarios. Hence, we can see that Random Forest provides consistency across data fitting and testset prediction with significance in threshold acceptance of mortality outcomes (P (Mortality) >50%) and shows favourability in selection as a predictive model for risk scoring system.

Random Forest can be combined with the Nomogram as it can be used to ranked and order the variables in terms of significance, where the hospital staff and data experts can possess the ability to visualise and determine cut-off limits for variable selection and importance according to rankings and business decisions where they can omit variables in usage for the Nomogram. As shown in Data exploration and Variable selection *(Appendix D),* RDW (Red Blood Cell Distribution Width) and MCV are both similar as they both measure Red Blood Cell volume, and hence one of them may be eliminated to reduce the biasness in Nomogram, while a heavier weightage can be given in determination of mortality as the variable shows that the element of the human pathology (RDW/MCV) is important in determining Heart Failure as they both appeared in the top 20 variables, with advice and involvement from domain experts.

Random Forest also reduces error rates through Bootstrap Aggregation, which generates each model through resampling data independently. The multitude resampling allows for minimisation of error and variable selection bias and stabilisation through the Bootstrap function, decreasing data overfitting onto the model where majority ranking and the average of variables is present to determine the final output. In other studies, Random Forest has been employed as a predictor of heart diseases, allowing for decreased wastage and efficient allocation of resources to cases through low False Positive cases and higher accuracy, precision and timely prediction in missing out the detection of cases in lower False Negative Rates. *(Pal M., et al., 2020)*

Through the ability to reduce resource utilization in low-risk patients (lower FPR), while accurately predicting more than other models (lower FNR), **Random Forest is recommended** as it excels with the ability to classify variables and quantify and qualify the variables/output through regressive techniques for fitting onto sub-models such as the Nomogram.

## Answer to Q10:

**10) Suggest other ideas that may improve the accuracy of the model.**

To improve the accuracy of the model (Random Forest), we can employ several techniques: (i) Increase data variables collected and amount of data, (ii) Imputing missing values using K-Nearest Neighbours or MissForest Algorithm, (iii) Model settings adjustment (mtry and ntrees) and (iv) Utilise several datasets across various samples with similar variables composition.

**(i) Increase data variables collected and amount of data:** For this project, we can observe that out of the variables amongst the 1177 patients, some variables used had as much as 290 NA values. The presence of NA values would result in possible skewed data analysis when imputed with median or modal values. An increase in amount of data (E.g. 7000 patients) would decrease the proportion of NA values occurrence and allow Random Forest to have a wider range of data points to prevent biased decisions while identifying and eliminating outliers *(Mishra A., 2020)*. Increasing variables would also provide greater robustness and realism in applicability, as clinical studies have shown that external factors such as the inclusion of Smoking, physical inactivity *(Yang H., et al., 2015)* have also aided in providing meaningful outputs for mortality risk.

**(ii) Imputing missing values using K-Nearest Neighbours (KNN) or MissForest Algorithm:** The current method of imputing missing values in R was to use na.roughfix, which replaced data by median if the missing value was continuous or by mode if the variable was categorical, which could lead to skewed data as shown in Age variable. Two machine learning methodologies to replace missing values would be KNN and MissForest, which calculates the missing value based on train and test data to replace the missing value and using random Forest to replace the value respectively *(Ye A., 2020)*. However, MissForest has been observed to be more robust to noisy data and multicollinearity, which is advantageous as it decreases the distortion from noisy variables such as high correlation or weak predictors through using an algorithm similar to random forest for value imputation.

**(iii) Model settings adjustment (mtry and ntrees):** Based on the data used, the number of random variables used in each tree is floor(sqrt(48)) = 6 as outcome is a categorical dependent variable. To improve accuracy, we can increase the number of random variables (mtry) as it causes correlation and strength to increase between trees when greater number of variables are input into the model. Increasing the ntrees will reduce the variance of the error score and increase stability of the model, leading to higher accuracy across predicted values when Bagging occurs. To obtain optimal values, the RF model can be run with 10 different combinations of hyperparameters (mtry and ntrees) and the optimal feature selection obtained at minimal Out-Of-Bag error where the model has stabilized. *(Bhalla D., n.d.)*

**(iv) Utilise several datasets across various samples with similar variables composition:** Using the dataset given, 1177 individuals may not be reflective of parameters of a population, or a larger sample. In this case, the model can be validated and trained using other similar datasets that contain similar variables to data01.csv, but containing different data values and patients within. This data models can then be cross trained with other models or testsets through transfer learning *(Brownlee J., 2017)*. In addition, when this occurs the model performance is continuously revalidated and repurposed, which helps to update its performance with the input of new data that ensures output is current and learning improves causing accuracy to increase with predicted data relevance to the population.

# References


Cancer.org. (n.d.) Low White Blood Cells Count (Neutropenia). Accessed 23 March 2022. Retrieved from: https://www.cancer.org/treatment/treatments-and-side-effects/physical-side-effects/low-blood-counts/neutropenia.html

Jill S.S. (14 December 2021). What is MCH and What do high and low Values mean? Accessed 23 March 2022. Retrieved from: https://www.healthline.com/health/mch

Higgins C. (October 2016). Urea and Creatinine Concentration: the Urea: Creatinine ratio. Accessed 23 March 2022. Retrieved from: https://acutecaretesting.org/en/articles/urea-and-creatinine-concentration-the-urea-creatinine-ratio

Pal M., et al. (2020) Prediction of Heart Diseases using Random Forest. Accessed 27 March 2022. Retrieved from: https://iopscience.iop.org/article/10.1088/1742-6596/1817/1/012009/pdf#:~:text=Using%20random%20forest%20algorithm%2C%20we,using%20random%20forest%20is%2093.3%25.

Mishra A. (4 April 2020) Ways to Improve the Accuracy of Machine Learning Models. Accessed 27 March 2022. Retrieved from: https://datascience.foundation/datatalk/ways-to-improve-the-accuracy-of-machine-learning-models

Yang H. (17 March 2015). Clinical Prediction of incident heart failure risk: a systematic review and meta-analysis. Accessed 27 March 2022. Retrieved from: https://openheart.bmj.com/content/2/1/e000222

Ye A. (1 September 2020). MissForest: The Best Missing Data Imputation Algorithm? Accessed 27 March 2022. Retrieved from: https://towardsdatascience.com/missforest-the-best-missing-data-imputation-algorithm-4d01182aed3

Bhalla D. (n.d.D0. A complete guide to random forest in R. Accessed 27 Match 2022. Retrieved from: https://www.listendata.com/2014/11/random-forest-with-r.html

Brownlee J. (20 December 2017). A gentle introduction to transfer learning for deep learning. Accessed 27 March 2022. Retrieved from: https://machinelearningmastery.com/transfer-learning-for-deep-learning/

CDC. (n.d.) Smoking and COPD. Accessed 28 March 2022. Retrieved from: https://www.cdc.gov/tobacco/campaign/tips/diseases/copd.html#:~:text=How%20Is%20Smoking%20Related%20to,can%20also%20contribute%20to%20COPD.&text=As%20many%20as%201%20out,with%20COPD%20never%20smoked%20cigarettes.

# Appendix

**Appendix A: Key Assumptions**

1. Race is assumed to be non-black in the data for GWTG Risk Scoring Model as race/ethnicity is not available in the data but Li F., et al (2021) shows that non-blacks is the great majority (approx. 86%).
2. XGBoost model was used in Nomogram construction and the multivariate equation used was **log(odds of mortality) = (4.62536 + 0.24559 × anion gap + 0.61542 × lactate−1.04993 × calcium+ 0.02687 × BUN −1.76330 × CKD − 0.05633 × DBP),** where the equation units of measurement are same as the variable units of measurement.
3. Probability and distribution of data and participants for the study are selected at random.
4. Presence or Absence of Smoking is linked to Heart Failure mortality as it is related to COPD *(CDC., n.d)*. For the data, we will assume that Smoking is taken into account by COPD variable.
5. Key predictors of Heart Failure include lifestyle (physical inactivity, diet) *(Yang H., et al., 2015)*. For this analysis, we will assume that diets were same as patients were in-hospital and physical activity levels are negligible in mortality prediction.
6. Patient family history has also been taken into account into account with variables CHD with no MI, renal Failure and Hyperlipemia. *(Yang H., et al., 2015)*
7. Patients have no other existing disease that can cause elevated results of white blood cells, neutrophils or variables that can cause distortion in the existing levels of variables in the dataset.

## Appendix B: GWTG Bin Ranges for Variable Scoring and codes

| Systolic BP | Points | BUN | Points | Sodium | Points | Age | Points |
|---|---|---|---|---|---|---|---|
| 50-59 | 28 | ≤9 | 0 | ≤130 | 4 | ≤19 | 0 |
| 60-69 | 26 | 10-19 | 2 | 131 | 3 | 20-29 | 3 |
| 70-79 | 24 | 20-29 | 4 | 132 | 3 | 30-39 | 6 |
| 80-89 | 23 | 30-39 | 6 | 133 | 3 | 40-49 | 8 |
| 90-99 | 21 | 40-49 | 8 | 134 | 2 | 50-59 | 11 |
| 100-109 | 19 | 50-59 | 9 | 135 | 2 | 60-69 | 14 |
| 110-119 | 17 | 60-69 | 11 | 136 | 2 | 70-79 | 17 |
| 120-129 | 15 | 70-79 | 13 | 137 | 1 | 80-89 | 19 |
| 130-139 | 13 | 80-89 | 15 | 138 | 1 | 90-99 | 22 |
| 140-149 | 11 | 90-99 | 17 | ≥139 | 0 | 100-109 | 25 |
| 150-159 | 9 | 100-109 | 19 | | | ≥110 | 28 |
| 160-169 | 8 | 110-119 | 21 | | | | |
| 170-179 | 6 | 120-129 | 23 | | | | |
| 180-189 | 4 | 130-139 | 25 | | | | |
| 190-199 | 2 | 140-149 | 27 | | | | |
| ≥200 | 0 | ≥150 | 28 | | | | |

| Heart Rate | Points | Black Race | Points | COPD | Points |
|---|---|---|---|---|---|
| ≤79 | 0 | Yes | 0 | Yes | 2 |
| 80-84 | 1 | No | 3 | No | 0 |
| 85-89 | 3 | | | | |
| 90-94 | 4 | | | | |
| 95-99 | 5 | | | | |
| 100-104 | 6 | | | | |
| ≥105 | 8 | | | | |

| Total Score | Probability of Death |
|---|---|
| 0-33 | <1% |
| 34-50 | 1-5% |
| 51-57 | >5-10% |
| 58-61 | >10-15% |
| 62-65 | >15-20% |
| 66-70 | >20-30% |
| 71-74 | >30-40% |
| 75-78 | >40-50% |
| ≥79 | >50% |

GWTG Variables imputed into R code for determination of GWTG Score in next page

```
trainGWTG = trainset
trainGWTG$`Systolic blood pressure score`=cut(trainGWTG$`Systolic.blood.pressure`,
                                              breaks = c(0,59,69,79,89,99,109,119,129,139,149,159,169,179,189,199,
                                                         max(trainGWTG$`Systolic.blood.pressure`)),
                                              labels = c("28","26","24","23","21","19","17","15","13","11","9","8","6","4","2","0"),
                                              include.lowest = TRUE)

trainGWTG$BUN=cut(trainGWTG$Urea.nitrogen, breaks = c(0,9,19,29,39,49,59,69,79,89,99,109,119,129,139,149,
                                                      max(trainGWTG$Urea.nitrogen)),
                  labels = c("0","2","4","6","8","9","11","13","15","17","19","21","23","25","27","28"),
                  include.lowest = TRUE)

trainGWTG$SodiumScore =cut(trainGWTG$Blood.sodium,
                           breaks = c(0,130,131,132,133,134,135,136,137,138,
                                      max(trainGWTG$Blood.sodium)),
                           labels = c("4","3","3","3","2","2","2","1","1","0"),
                           include.lowest = TRUE)

trainGWTG$Age1 = cut(trainGWTG$age, breaks = c(0,19,29,39,49,59,69,79,89,
                                               max(trainGWTG$age)),
                     labels = c("0","3","6","8","11","14","17","19","22"),
                     include.lowest = TRUE)

trainGWTG$HeartRate = cut(trainGWTG$heart.rate, breaks = c(0,79,84,89,94,99,104,
                                                           max(trainGWTG$heart.rate)),
                          labels = c("0","1","3","4","5","6","8"),
                          include.lowest = TRUE)

trainGWTG$COPDScore = ifelse(trainGWTG$COPD =="1", 2,0)
```

trainGWTG$ttlscore used to store the total score across all variables identified for Heart Failure

```
trainGWTG$ttlscore = as.numeric(as.character(trainGWTG$Age1)) + as.numeric(as.character(trainGWTG$BUN))+
  as.numeric(as.character(trainGWTG$`Systolic blood pressure score`)) + as.numeric(as.character(trainGWTG$SodiumScore)) +
  as.numeric(as.character(trainGWTG$HeartRate)) + as.numeric(as.character(trainGWTG$COPDScore)) + 3

trainGWTG$outcome2 = factor(ifelse(trainGWTG$ttlscore >= 79, 1,0))
class(trainGWTG$outcome2)
levels(trainGWTG$outcome2)
summary(trainGWTG)
```

Ttlscore is factored into mortality (outcome = 1) and no mortality (outcome = 0) by setting probability of death at 50% which occurs when ttlscore>= 79

**Appendix C: Nomogram variable equation for Mortality Scoring and codes**

p-
in
rs

odds of mortality=4.62536+0.24559×anion gap+0.61542×-lactate−1.04993×calcium+0.02687×BUN−1.76330×CK-D−0.05633×DBP.

The variance inflation factors (VIFs) for these variables

Log (Odds of mortality) scoring equation for Nomogram calculation

**Obtained from: Li F, et al (2021)**

```
trainNOMO = trainset
trainNOMO$outcome1 = (4.62536+0.24559*trainNOMO$Anion.gap +0.61542*(trainNOMO$Lactic.acid)- 1.04993*(trainNOMO$Blood.calcium)+
  0.02687*(trainNOMO$Urea.nitrogen)-1.76330*as.numeric(as.character(trainNOMO$Renal.failure))-0.05633*(trainNOMO$Diastolic.blood.pressure))

trainNOMO$prob1 = exp(trainNOMO$outcome1)/(1+exp(trainNOMO$outcome1))
summary(trainNOMO)

trainNOMO$outcome2 = factor(ifelse(trainNOMO$prob1 > 0.5, 1,0))

NOMO_conf_matrix=confusionMatrix(data= as.factor(trainNOMO$outcome2), trainNOMO$outcome, positive = "1")
NOMO_conf_matrix
```

Creating probability for odds of mortality by removing logarithmic function in Log (Odds of mortality) Dependent Y variable.

Nomogram Scoring equation for odds of Mortality determined by XGBoost which was selected (shown in the article).

Factorisation of Odds of mortality probability and setting threshold that death of patient occurs of P>0.5 (Outcome = 1)

**Appendix D: Top 20 predicted variables by Random Forest**