Josh Blakely

Project

ISYE 6420
June 27, 2023

Introduction

Cardiovascular diseases (CVDs) are the number one cause of death globably, taking an estimated 17.9 million lives each year. This accounts for 31% of all deaths worldwide. Four out of five CVD deaths are due to hear attacks and strokes, and one-third of these deaths occur prematurely in people under 70 years of age. Cardiovascular disease are conditions that affect the structures or function of your heart, such as: abnormal heart rhythms or arrhythmias, aorta discease, marfen syndrome, congenital heart disease, coronary artery disease, etc. Heart failure is a common event caused by CVDs. The dataset that I am using contains eleven predictive values that can be used to predict a possible heart disease. The dataset was retrieved from Kaggle[1]. The data is the combination of five different datasets [2] that total 918 non-duplicate observations.

The data is broken down into the eleven predictive values below:

- Age: age of the patient [years]
- Sex: sex of the patient [M: Male, F: Female]
- ChestPainType: chest pain type [TA: Typical Angina, ATA: Atypical Angina, NAP: Non-Anginal Pain, ASY: Asymptomatic]
- RestingBP: resting blood pressure [mm Hg]
- Cholesterol: serum cholesterol [mg/dl]
- FastingBS: fasting blood sugar [1: if FastingBS $>$ 120 mg/dl, 0: otherwise]
- RestingECG: resting electrocardiogram results [Normal: Normal, ST: having ST-T wave abnormality (T wave inversions and/or ST elevation or depression of $>$ 0.05 mV), LVH: showing probable or definite left ventricular hypertrophy by Estes' criteria]
- MaxHR: maximum heart rate achieved [Numeric value between 60 and 202]
- ExerciseAngina: exercise-induced angina [Y: Yes, N: No]
- Oldpeak: oldpeak = ST [Numeric value measured in depression]
- ST_Slope: the slope of the peak exercise ST segment [Up: upsloping, Flat: flat, Down: downsloping]
- HeartDisease: output class [1: heart disease, 0: Normal]

People with cardiovascular disease or who are at high cardiovascular risk (due to the presence of one or more risk factors such as hypertension, diabetes, hyperlipidaemia or already established disease) need early detection and management. Therefore, can we determine if someone is more prone to cardiovascular disease, given the features?

I will be using 20% of the data to test and verify the model and will be verifying the accuracy using AUC from ROC curve.

Analysis

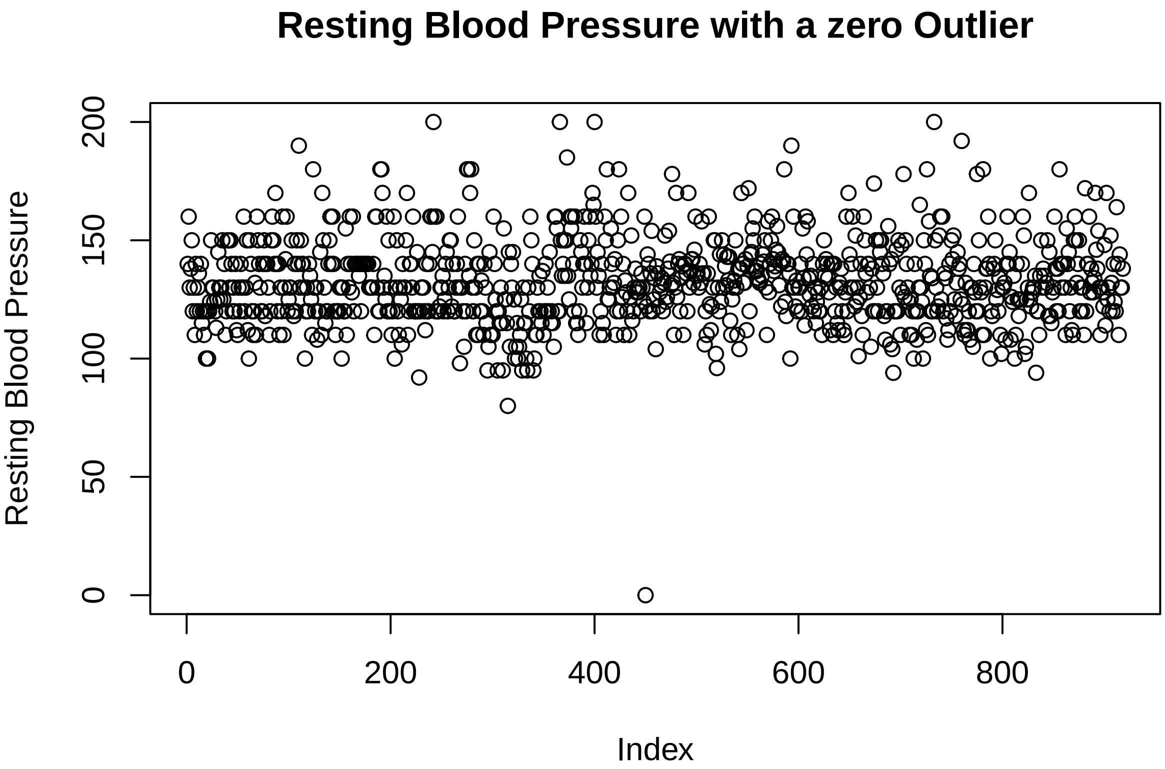
Data Cleaning:

The first thing that should be done on any dataset is to plot the data and run basic analysis on each of the predictors. When reviewing the data, it was noticed that there were two glaring issues with the data. The first occurred in the `RestingBP`.

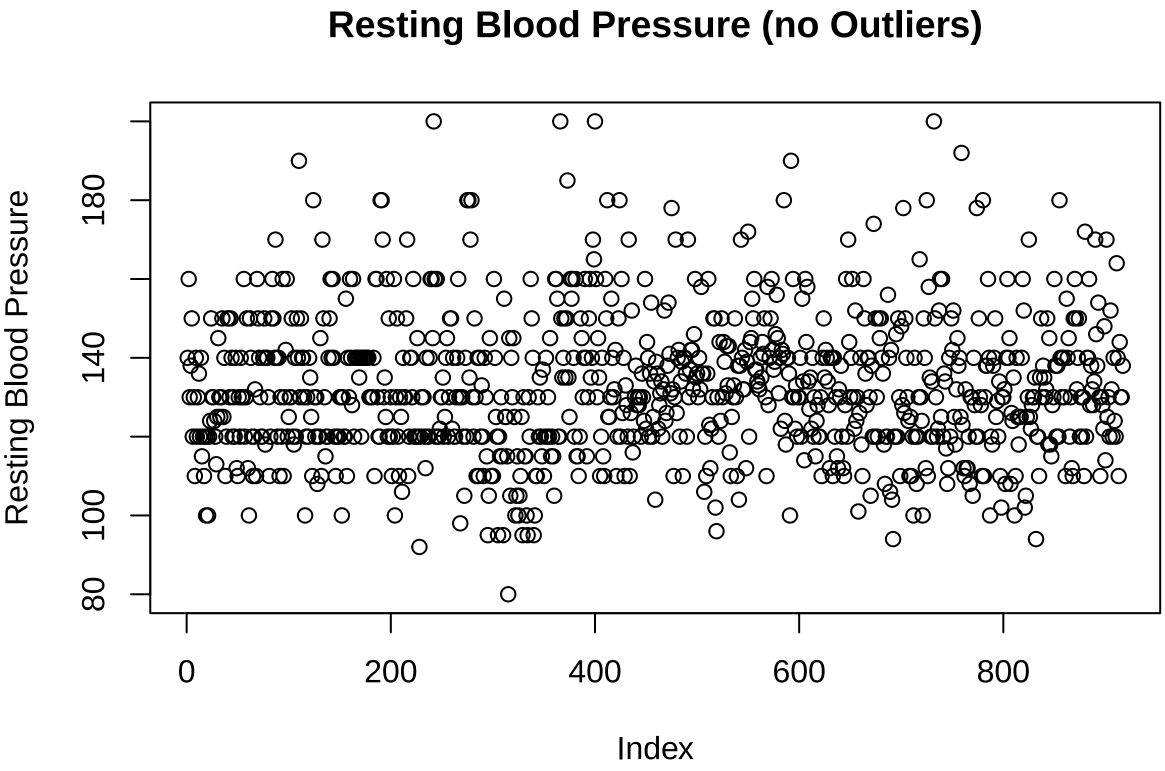
[1] https://www.kaggle.com/datasets/fedesoriano/heart-failure-prediction?sort=recent-comments

[2] fedesoriano. (September 2021). Heart Failure Prediction Dataset. Retrieved 4/24/2022 from https://www.kaggle.com/fedesoriano/heart-failure-prediction.

Resting Blood Pressure

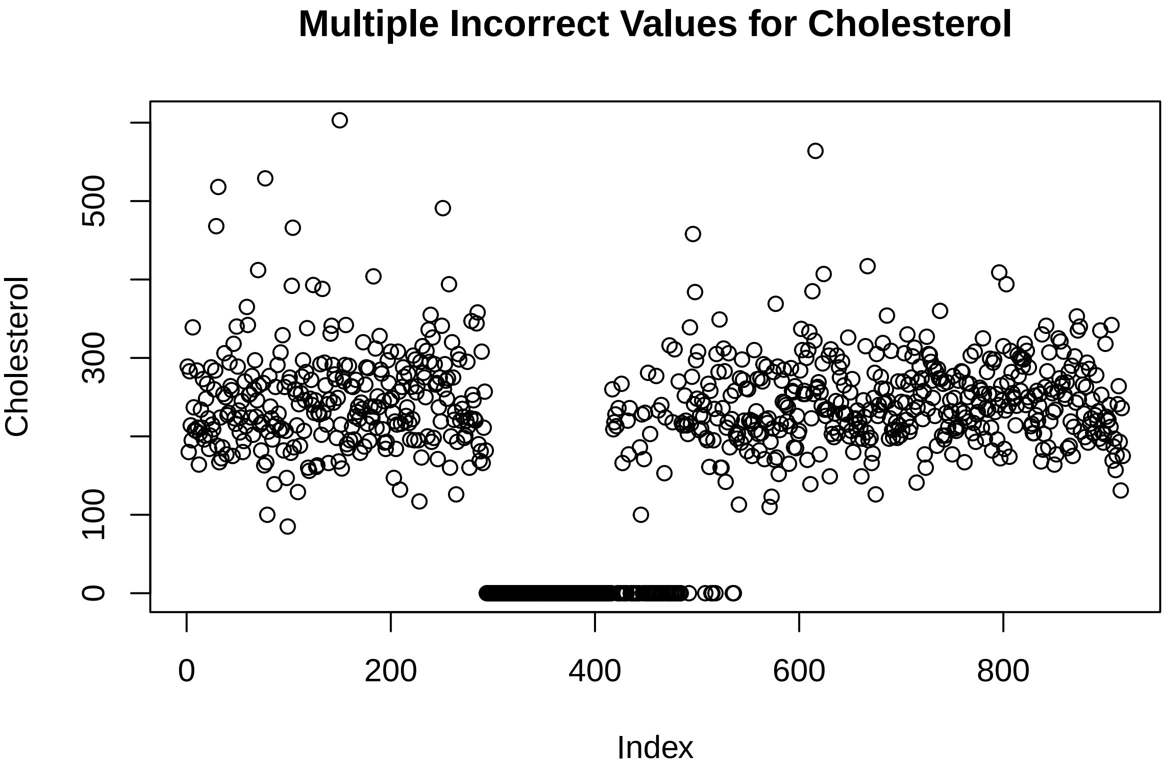


We can see in the above that we have a resting blood pressure value of **0**, which would mean that this person is dead. I could have used Bayes to impute the incorrect/missing value, but I decided to remove it because in the end one less data point won't truly affect the analysis. Also there really isn't a point to set up a Bayes model to impute a single point, in my opinion, especially with how many good points we still have. After removing the outlier, we can see the new data appears to be within the necessary values for resting blood pressure.



Cholesterol

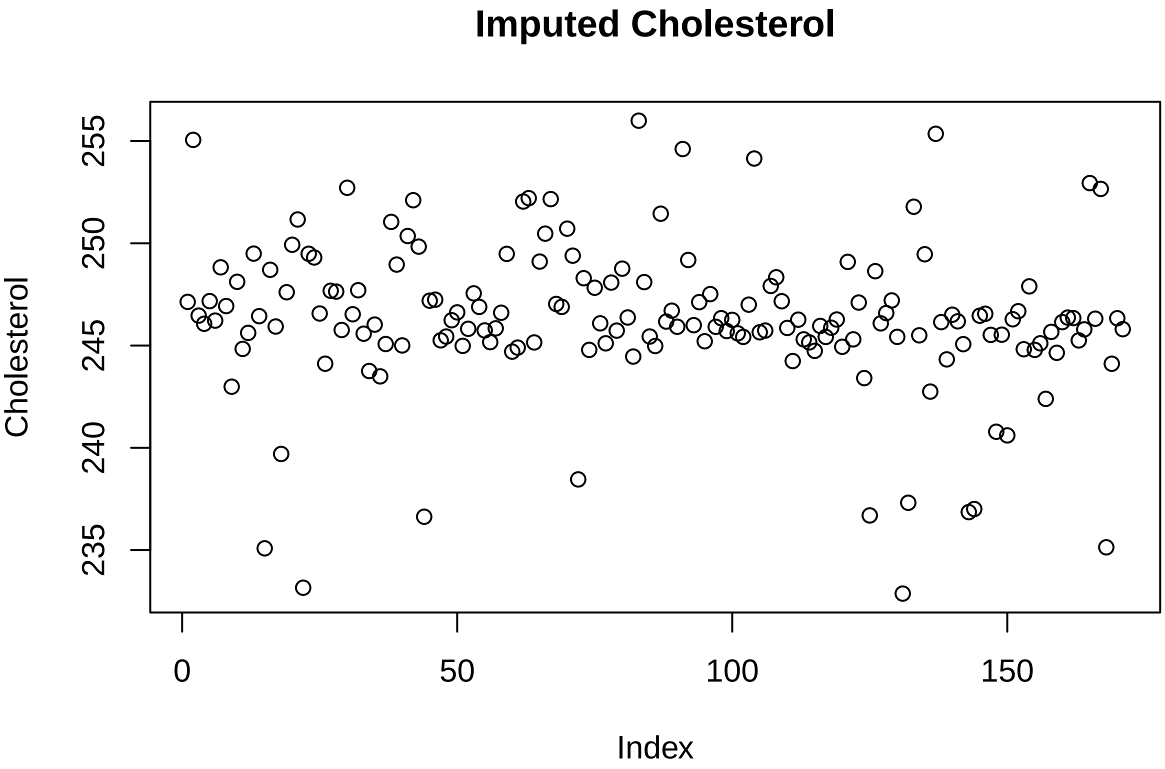
The second issue I found while exploring the data was with the `Cholesterol`. The dataset shows that a range of indexes have a `Cholesterol` value of zero, which also like resting blood pressure is impossible.

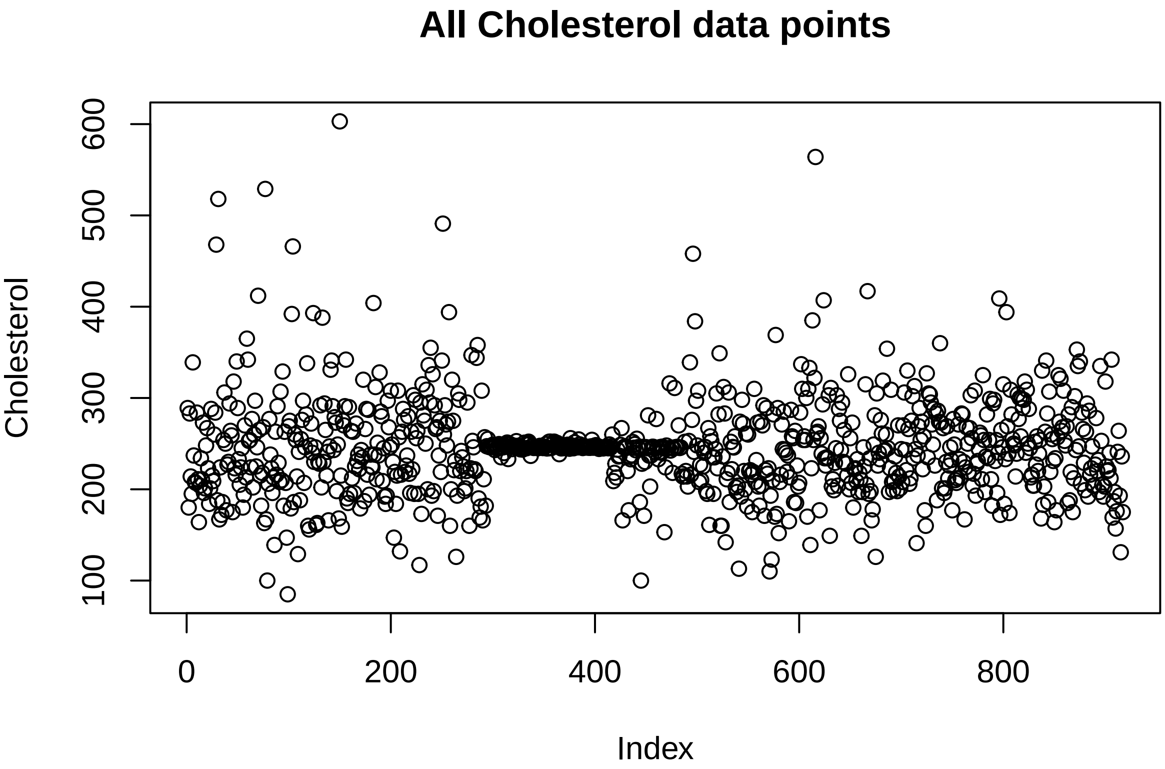


These zero values for `Cholesterol` account for *171* values, which is approximately **18%** of the data set. This is a significant value. Normally when encountering a data set with this large of incorrect data, the goal would be to either throw out the data that is incorrect or supply the mean of the observed values for all the missing data points. Even then this is a significant amount of incorrect data. I have decided to use Bayes MCMC to try and impute the data. I used non-informative priors allowing the data set to determine what the values should be. The model that was used for this can be seen in `Appendix A: Cholesterol Imputed`. Looking at the missing values, it appears that one of the five data sets didn't have correct values for `Cholesterol`. Again normally these would be thrown out, but I decided to use a Bayesian model to find what the imputed values would be. Due to the assumption that it is an incorrect data set, I didn't split the data into training and test sets for the imputation. I am using different models for each to make sure that there isn't any leakage for determining the imputed values to later predicting the test set.

The reason I have decided to split them was due to the fact that this data set is a combination of 5 different data sets from different areas that have been combined to create a single data set. I am making the assumption that because the missing values are all relatively next to each other, that they all came from the same data set. More than likely the data would not have the missing values and, therefore, would not need to be imputed before running through the model. I could have made adjustments to the model to impute the missing Cholesterol-values for a training data set and then using this distribution determine what it would be for the test data set. I decided instead to first impute all the values and then split them into training and test sets, which could then be run through a prediction model.

After running the data through the stan model, I was able to get imputed data points. Below you can see the graph of the missing values by themselves.





We can see that there is slight variation as compared to the mean value, but they aren't as varied as the original data points. No matter what was changed in the model, I was unable to get the wide variation that is seen the other data points, but the imputed data appears to make logical sense.

Results

Predictive Model:

Now that I have a "cleaned" data set with filled in missing values for Cholesterol, I was able to create a logit regression model. This Stan model can be seen in `Appendix B: Logit Regression`.

I am building a simple Bernoulli logit model with normal prior for the betas.

$$\alpha = \beta_0 + X * \beta_{1..k}$$

Where: * X: predictive variables * β_0: intercept coefficient * $\beta_{1..k}$: coefficients for each of the predictive variables

$$BernoulliLogit(y \mid \alpha) = Bernoulli(y \mid logit^{-1}(\alpha)) = \begin{cases} logit^{-1}(\alpha), & ify = 1 \\ 1 - logit^{-1}(\alpha), & ify = 0 \end{cases} \tag{1}$$

$$Bernoulli(y \mid logit^{-1}(\alpha)) = \begin{cases} logit^{-1}(\beta_0 + X * \beta_{1..k}), & ify = 1 \\ 1 - logit^{-1}(\beta_0 + X * \beta_{1..k}), & ify = 0 \end{cases} \tag{2}$$

Once the Bayesian model has been trained on the training set, I was able to supply the test variables (`X.test`) to the model. This allows me to predict the values for y on the test set. Grabbing the mean for all the different MCMC chains we are able to get the predicted values for y upon the test set.

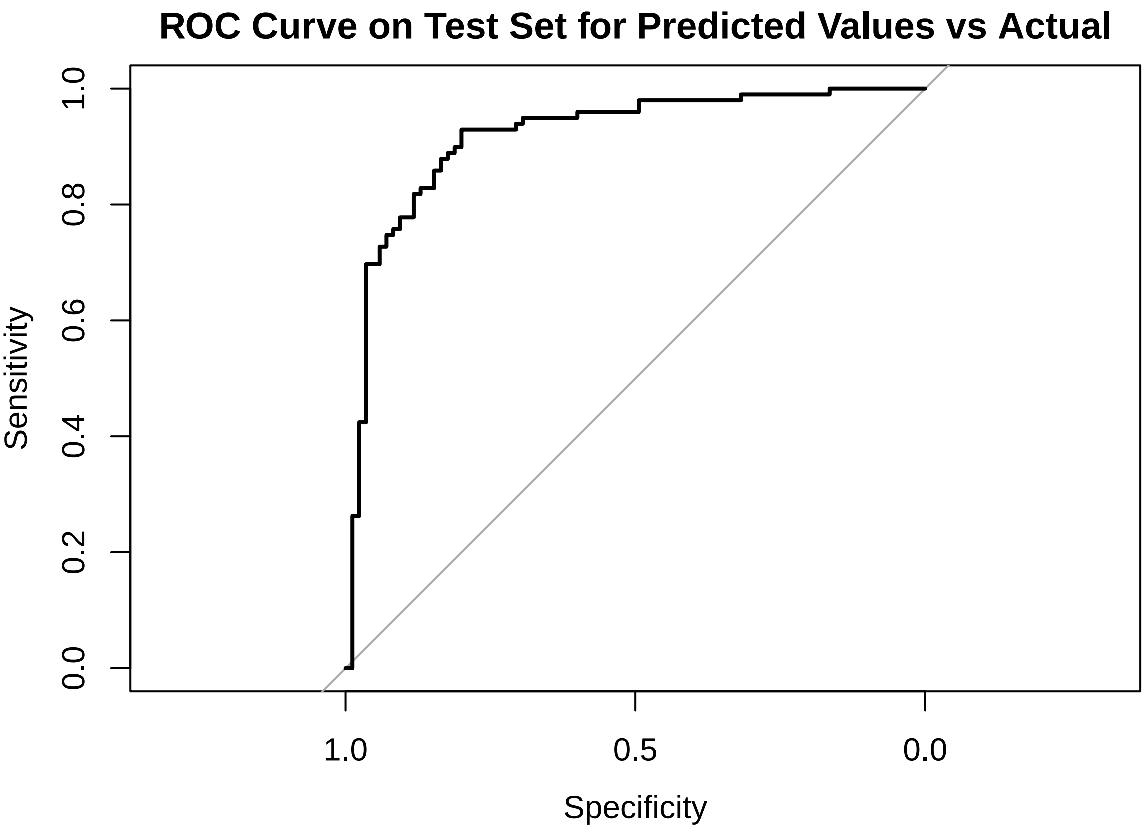
```
# statistics
ypred <- fit$summary('y_pred')$mean
ypred
```

```
##   [1] 0.12975 0.24800 0.02675 0.86600 0.88950 0.47700 0.05925 0.96350 0.27300
##  [10] 0.08125 0.02500 0.10925 0.16450 0.96450 0.03875 0.46525 0.94075 0.91250
##  [19] 0.11450 0.95750 0.30125 0.17150 0.94400 0.23600 0.97900 0.92100 0.28850
##  [28] 0.08075 0.08350 0.99725 0.37225 0.99400 0.79075 0.94525 0.13375 0.98050
##  [37] 0.18325 0.17975 0.06475 0.45550 0.92650 0.39375 0.01625 0.93400 0.00525
##  [46] 0.01625 0.04450 0.93575 0.86600 0.71275 0.30475 0.12625 0.94150 0.36350
##  [55] 0.13675 0.04875 0.93525 0.95275 0.89175 0.73950 0.60225 0.49225 0.92300
##  [64] 0.02525 0.96475 0.91225 0.97250 0.93775 0.49175 0.44425 0.79775 0.51175
##  [73] 0.74600 0.95700 0.68725 0.97775 0.04650 0.91350 0.68850 0.82975 0.98175
##  [82] 0.71425 0.99200 0.93475 0.97700 0.98875 0.98800 0.97625 0.53075 0.09150
##  [91] 0.99900 0.60600 0.97650 0.86125 0.97975 0.78875 0.97125 0.29725 0.95575
## [100] 0.90125 0.92750 0.99200 0.48625 0.52925 0.92800 0.92750 0.11925 0.95625
## [109] 0.96500 0.87325 0.94500 0.98575 0.95425 0.15375 0.95275 0.98350 0.95575
## [118] 0.92350 0.86225 0.69300 0.90825 0.59975 0.46950 0.06925 0.01075 0.88350
## [127] 0.08650 0.04375 0.14600 0.12975 0.05325 0.93900 0.42675 0.11200 0.87150
## [136] 0.90375 0.07450 0.72000 0.60875 0.99775 0.96325 0.06675 0.14100 0.73075
## [145] 0.13550 0.97550 0.01450 0.30150 0.34050 0.73550 0.01775 0.26075 0.17550
## [154] 0.35900 0.06975 0.50025 0.66050 0.04700 0.14325 0.93600 0.72350 0.35925
## [163] 0.22125 0.45200 0.12725 0.96025 0.08925 0.96000 0.92475 0.08700 0.41450
## [172] 0.48275 0.55200 0.96900 0.08625 0.96375 0.32675 0.03375 0.02900 0.17425
## [181] 0.37375 0.24450 0.97850 0.31450
```

Now that we have the predicted values, let us check the AUC for the ROC curve to determine the accuracy of the model.

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```



The ROC curve looks really good for this model. Let's see what the AUC for this model is:

```
auc(roc.score)
```

```
## Area under the curve: 0.9181
```

We can see that we have a decent score for the Bayesian predictive model with imputed Cholesterol values.

Conclusion

In my project I have shown that we can use a Bayesian model to impute missing/incorrect values. This allows us to have more data to model in order to determine accuracy. We can see that using the imputed results can allow for a fairly accurate model that could be used to determine cardiovascular disease. Ideally, we would want cleaner data especially working with something so vital as the health of patients. Using Bayesian models has allowed us to retain data to help the model perform better overall.

Building a Bayesian model is fairly straightforward for imputing the incorrect data. This actually makes me wonder why the general consensus with the data science community is just to use the mean or median value. Determining which to use would be determined by the distribution of the data. It would be significantly easier to use the mean if the data is normally distributed, but if you have left or right shifted distribution it would affect the accuracy of using the mean/median. Whereas by imputing with a Bayesian model, we would be able to get much more accurate data that is related to the data's distribution.

I would like to expand on this in the future to determine how a Bayesian logit predictive model compares to a general logit model. I would also like to compare the difference between dropping the missing values vs actual.

Appendix

Appendix A: Cholesterol Imputed

```
## // The input data is a vector ’y’ of length ’N’.
## data {
##   int<lower=0> N;       // observations
##   int<lower=0> N_obs;   // observed variables
##   int<lower=0> N_mis;   // missing variables
##   array[N_obs] int<lower=1, upper=N_obs + N_mis> ix_obs;  // index of obs x’s
##   array[N_mis] int<lower=1, upper=N_obs + N_mis> ix_mis;  // index of mis x’s
##
##   int<lower=1> K;       // predictor variables (reduced by one for Cholesterol)
##
##   matrix[N, K] X;       // The X’s without the imputed column (Cholesterol)
##   vector[N_obs] x_obs;  // Actual x values that were observed
##   array[N] int<lower=0, upper=1> y;
## }
##
## parameters {
##   real alpha;           // intercept
##   vector[K] betas;      // coefficients
##   real b_chol;          // Cholesterol coefficient
##   vector[N_mis] x_mis;  // Missing values for cholesterol
##
##   real mu;
##   real<lower=0> sigma;
## }
##
## transformed parameters {
##   vector[N] X_chol;        // Imputed Cholesterol values
##   X_chol[ix_obs] = x_obs;  // Observed Cholesterol values
##   X_chol[ix_mis] = x_mis;  // Missing Cholesterol values
## }
##
## model {
##   alpha ~ normal(0, 100);
##   betas ~ normal(0, 100);  // Non-informative prior
##   b_chol ~ normal(0, 100); // Non-informative prior
##
##   mu ~ normal(300, 50);    // Non-informative prior, but expecting to be around the mean of dataset
##   sigma ~ normal(0, 10);
##   X_chol ~ normal(mu, sigma);  // Impute the missing X values
##
##   y ~ bernoulli_logit(alpha + X * betas + X_chol * b_chol);
## }
```

Appendix B: Logit Regression

```
## // The input data is a vector 'y' of length 'N'.
## data {
##   int<lower=0> N;       // observations
##   int<lower=0> N_t;     // test oberservations
##   int<lower=1> K;       // coefficients
##
##   matrix[N, K] X;
##
##   matrix[N_t, K] X_test;
##   array[N] int<lower=0, upper=1> y;
##   vector[N_t] y_test;
## }
##
## parameters {
##   real alpha;    // intercept
##   vector[K] betas;
## }
##
## model {
##   alpha ~ uniform(-50, 50);
##   // alpha ~ normal(0, 10);
##   betas ~ normal(0, 10);
##   y ~ bernoulli_logit(alpha + X * betas);
## }
##
## generated quantities {
##   vector[N_t] y_pred;
##
##   for (n in 1:N_t) {
##     y_pred[n] = bernoulli_logit_rng(alpha + X_test[n] * betas);
##   }
##
##   // In order to calculate Bayesian R2:
##   // http://www.stat.columbia.edu/~gelman/research/unpublished/bayes_R2.pdf
##   real BR2;
##   BR2 = variance(y_pred) / (variance(y_pred) + variance(y_test - y_pred));
## }
```