FinalProject

Team 10

4/25/2022

Introduction (Quang Du)

Cardiovascular diseases are the number one cause of death globally, taking an estimated 17.9 million lives each year, amounting to 31% of all deaths worldwide. The United States alone ranked 4th in the world for the number of death dues to heart disease. Unfortunately, with our medical knowledge and technological advance, we're still unable to find a cure for coronary disease. The next best thing to do is configure models that can accurately predict who is more likely to encounter coronary diseases. Through trials and errors, our team came up with three models: Multiple Regression, Decisions tree, and Random forest.

During our research and experimenting with data, many questions aroused our curiosity. Such as which predictor/s are significant to our model? Which model is the best for accomplishing our goal amongst the created models? And which gender is more likely to develop coronary diseases? With that in mind, we designed our project in ways that'll satisfy these data questions.

About the Data (Quang Du)

Totally 918 observations with 12 variables.

The attribute for each variable are as follow, with expected output in bracket:

(1) `Age`: age of the patient

(2) `Sex` [M: Male, F: Female]

(3) `ChestPainType` [TA: Typical Angina, ATA: Atypical Angina, NAP: Non-Anginal Pain, ASY: Asymptomatic]

(4) `RestingBP`: resting blood pressure (mm Hg)

(5) `Cholesterol`: serum cholesterol (mm/dl)

(6) `FastingBS`: fasting blood sugar [1: if FastingBS > 120 mg/dl, 0: otherwise]

(7) `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]

(8) `MaxHR`: maximum heart rate achieved [between 60 and 202]

(9) `ExerciseAngina`: exercise-induced angina [Y: Yes, N: No]

(10) `Oldpeak`: oldpeak = ST [Numeric value measured in depression]

(11) `ST_Slope`: the slope of the peak exercise ST segment [Up: upsloping, Flat: flat, Down: downsloping]

Finally, the variable `HeartDisease` will be our response variable, with output of 1 meaning patient has heart disease and 0 for normal. This told us that our response variable is qualitative. Therefore moving forward, we will be using models that deal with classification problem.

Multiple Logistic Regression (Ngoc Tran, Lac Tran)

The goal of a multiple logistic regression is to find an equation that best predicts the probability of a value of the response variable as a function of the predictor variables. Once this equation is formed, we can use it to understand the functional relationship between the independent variables and the dependent variable, to try to understand what might cause the probability of the dependent variable to change.

```
heart.glm = glm(HeartDisease ~ ., family = "binomial", data = heart)
summary(heart.glm)
```

```
##
## Call:
## glm(formula = HeartDisease ~ ., family = "binomial", data = heart)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.6531  -0.3747   0.1745   0.4457   2.5778
##
## Coefficients:
##                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)      -1.163656   1.416003  -0.822 0.411197
## Age               0.016550   0.013197   1.254 0.209803
## SexM              1.466477   0.279834   5.241 1.60e-07 ***
## ChestPainTypeATA -1.830289   0.326293  -5.609 2.03e-08 ***
## ChestPainTypeNAP -1.685682   0.266001  -6.337 2.34e-10 ***
## ChestPainTypeTA  -1.488392   0.432572  -3.441 0.000580 ***
## RestingBP         0.004194   0.006010   0.698 0.485296
## Cholesterol      -0.004115   0.001087  -3.785 0.000154 ***
## FastingBS         1.136482   0.274999   4.133 3.59e-05 ***
## RestingECGNormal -0.177033   0.271925  -0.651 0.515022
## RestingECGST     -0.268546   0.350020  -0.767 0.442945
## MaxHR            -0.004288   0.005023  -0.854 0.393249
## ExerciseAnginaY   0.900292   0.244513   3.682 0.000231 ***
## Oldpeak           0.380643   0.118466   3.213 0.001313 **
## ST_SlopeFlat      1.453902   0.429086   3.388 0.000703 ***
## ST_SlopeUp       -0.994101   0.450196  -2.208 0.027234 *
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1262.14  on 917  degrees of freedom
## Residual deviance:  594.19  on 902  degrees of freedom
## AIC: 626.19
##
## Number of Fisher Scoring iterations: 6
```

We first fitted the model with full set of predictors. With the summary of the model, we can easily obtain the significant variable by observing the p-value in the last column. Any variable which has a p-value less than or equal to 0.05 is considered a significant variable. Therefore, there are 7 siginificant variables in this model, which are `Sex`, `ChestPainType`, `Cholesterol`, `FastingBS`, `ExerciseAngina`, `Oldpeak` and `ST_Slope`. Additionally, the AIC obtained here is 626.19. Next, we proceed to fit the model with only significant variables and derive the following summary of the coefficients:

```
heart.glm2 = glm(HeartDisease ~ Sex + ChestPainType + Cholesterol + FastingBS + ExerciseAngina + Oldpea
summary(heart.glm2)$coefficient
```

```
##                       Estimate  Std. Error    z value     Pr(>|z|)
## (Intercept)       -0.481858738 0.562252272 -0.8570152 3.914365e-01
## SexM               1.454586076 0.278086263  5.2307009 1.688685e-07
## ChestPainTypeATA  -1.878770980 0.322001652 -5.8346625 5.389954e-09
## ChestPainTypeNAP  -1.706719929 0.260758111 -6.5452228 5.940676e-11
## ChestPainTypeTA   -1.458703372 0.424978841 -3.4324141 5.982333e-04
## Cholesterol       -0.004124281 0.001026123 -4.0192866 5.837462e-05
## FastingBS          1.193157495 0.271641728  4.3923940 1.121093e-05
## ExerciseAnginaY    0.991359078 0.235370313  4.2119121 2.532181e-05
## Oldpeak            0.410093723 0.115694355  3.5446304 3.931640e-04
## ST_SlopeFlat       1.443532229 0.425674959  3.3911608 6.959726e-04
## ST_SlopeUp        -1.060365102 0.443634279 -2.3901785 1.684019e-02
```

```
summary(heart.glm2)$aic
```

```
## [1] 621.6088
```

The summary of the second model shows that every predictor in this model is significant and the AIC here is 621.61, which is smaller than the value we got from fitting model with a full set of predictors. Therefore, we will proceed to use the 7 significant predictors for training and testing the data to find out the error rate. Also, our multiple logistic regression model formula with significant predictors would be:

$$
\begin{aligned}
log(\frac{p(X)}{1-p(X)}) = & -0.481859 + 1.454586 \cdot X_{SexM} - 1.878771 \cdot X_{ChestPainTypeATA} - 1.706720 \cdot X_{ChestPainTypeNAP} \\
& - 1.458703 \cdot X_{ChestPainTypeTA} - 0.004124 \cdot Cholesterol + 1.193157 \cdot FastingBS \\
& + 0.991359 \cdot X_{ExerciseAnginaY} + 0.410094 \cdot Oldpeak + 1.443532 \cdot X_{ST_{Slope}Flat} - 1.060365 \cdot X_{ST_{Slope}Up}
\end{aligned}
$$

Note that among the significant predictors, `Sex`, `ChestPainType`, `ExerciseAngina` and `ST_Slope` are qualitative variables, so we will use dummy variables to represent them (e.g. $X_{SexM}, X_{ChestPainTypeATA}, ...$). Hence, we have $X_{\text{predictor category}}$ is 1 if the predictor is correctly categorized and is 0, otherwise. For example,

$$
X_{SexM} = \begin{cases} 1 & \text{if patient is male} \\ 0 & \text{if patient is female} \end{cases}
$$

Training and Testing

After loading in the data, the data is split into training data set and testing data set to allow easier manipulation to the data. 80% of the data is split into training data and the remaining 20% of the data lies in the testing data as shown below.

```
set.seed(10) #set seed
#Select 80% of the data for training data
sample = sample.int(n = nrow(heart),size = round(.8*nrow(heart)),
                    replace = FALSE)
```

```
train = heart[sample,]
test = heart[-sample,]
#Multiple Logistic Regression on training data
heart.glm.train= glm(HeartDisease~ Sex + ChestPainType + Cholesterol + FastingBS
                     + ExerciseAngina + Oldpeak + ST_Slope,
                     family="binomial",
                     data= train)
```

After that, we proceed to calculate the training error rate and average MSE.

```
#Creating Confusion Matrix on Train Data
predict.train = predict.glm(heart.glm.train,type = "response")
predict.hd.train = ifelse(predict.train < 0.5, "No Heart Disease","Yes Heart Disease")
(conf.mat.train = table(predict.hd.train,train$HeartDisease))
```

```
##
## predict.hd.train      0   1
##   No Heart Disease  278  42
##   Yes Heart Disease  53 361
```

Here, the probability of heart disease is classified into two categories. When the probability is higher than 50%, the prediction will be classified as having heart disease. On the other hand, the probability of having heart disease that is lower than 50% is an indication that there is no presence of heart failure. The training error rate obtained from the confusion matrix is 12.94%.

```
cf=NA
for(i in 1:10) {
  set.seed(i) #set seed
  #Select 80% of the data for training data
  sample = sample.int(n = nrow(heart),size = round(.80*nrow(heart)),
                  replace = FALSE)
  train = heart[sample,]
  test = heart[-sample,]

  heart.glm= glm(HeartDisease~ Sex+ Age+ MaxHR+ Cholesterol+ ChestPainType,
               family="binomial", data= train)

  predict.test = predict.glm(heart.glm,type = "response", newdata = test)
  predict.hd.test = ifelse(predict.test< 0.5,"No Heart Disease","Yes Heart Disease")

  #Confustion Matrix
  (conf.mat = table(predict.hd.test,test$HeartDisease))

  #Testing Error Rate
  cf[i]= (conf.mat[1,2]+conf.mat[2,1])/sum(conf.mat)
}
cf
```

```
##  [1] 0.1630435 0.2010870 0.1630435 0.1956522 0.1684783 0.2119565 0.2282609
##  [8] 0.2500000 0.2282609 0.2500000
```

```
#Average of all test error rates
mean(cf)
```

```
## [1] 0.2059783
```

When the calculation for the test error rate is repeated 10 times, with an 80-20% split for the training data and testing data, the test error rates came out differently each time. The average calculated for all ten test error rates is 20.60%, which is higher than the training error rate. This is as expected because the training data set tends to perform better than the testing data set. Nonetheless, when looking at the different test error rates obtained, they appear to have quite low accuracy and high variance among each other. This indicates that a better model should be used for prediction.

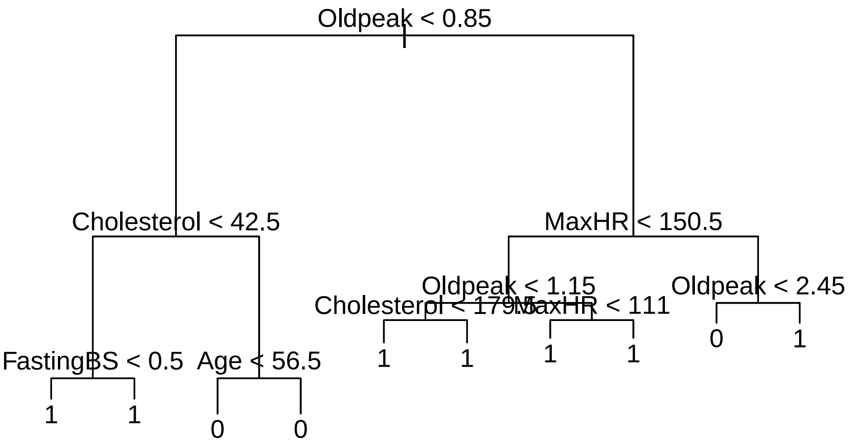
Classification Tree Model (Jinangkumar Shah)

The fundamental difficulty with logistic regression is accurately interpreting the data, whereas decision trees are easy to comprehend. Ease of Decision Making is enhanced with decision trees. Decision trees, unlike logistic regression, are pruned to avoid overfitting. For the classification tree, we'll use `HeartDisease` as our response variable and all other variables as our predictors. This is the formula we will use for the classification tree:

$$
Heart\hat{D}isease \sim Age + Sex + ChestPainType + RestingBP + Cholesterol + FastingBS + RestingECG \\
+ MaxHR + ExerciseAngina + Oldpeak + ST\_Slope
$$

Now we use the training data to create a decision tree:

```
#set.seed(10)
tree.heart = tree(as.factor(HeartDisease)~., data = train)
plot(tree.heart)
text(tree.heart, pretty = 1)
```



```r
summary(tree.heart)
```

```
##
## Classification tree:
## tree(formula = as.factor(HeartDisease) ~ ., data = train)
## Variables actually used in tree construction:
## [1] "Oldpeak"     "Cholesterol" "FastingBS"   "Age"         "MaxHR"
## Number of terminal nodes:  10
## Residual mean deviance:  0.8884 = 643.2 / 724
## Misclassification error rate: 0.2003 = 147 / 734
```

For our tree, `FastingBS`, `Oldpeak`, `Age`, `Cholesterol`, and `MaxHR` are used in this tree. We have a residual mean deviance of `0.8884` and a misclassification error rate of `20.03%`. With 10 terminal nodes. we will evaluate the performance of our tree by using the testing set of our data. First, we will repeatedly calculating the test error rate 10 times with different subsets of training and testing data and then find the mean value.

```r
mse.unprune = NA
for (i in 1:10) {
  set.seed(i)
  sample = sample(nrow(heart), round(nrow(heart)*.80))
  train = heart[sample,]
  test = heart[-sample,]
  tree.heart = tree(as.factor(HeartDisease)~., data = train)
  tree.pred = predict(tree.heart, test, type = 'class')
  table(tree.pred, test$HeartDisease)
  test.matrix = table(tree.pred, test$HeartDisease)
  mse.unprune[i] = (test.matrix[2] + test.matrix[3])/sum(test.matrix)
}
mse.unprune
```

```
##  [1] 0.1902174 0.2391304 0.2010870 0.1902174 0.2010870 0.1902174 0.2228261
##  [8] 0.2282609 0.2336957 0.2065217
```

```r
mean(mse.unprune)
```

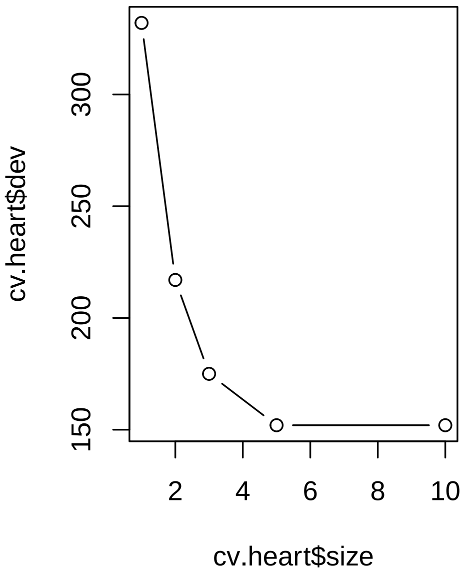
```
## [1] 0.2103261
```

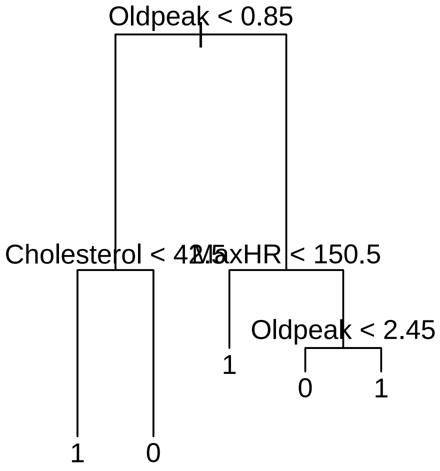
After iterating 10 times we get mean error to be 0.210326 ≈ `21.03%` and conversely an accuracy rate of `79.97%`. With this accuracy rate, our tree performs alright in predicting if a patient may have heart disease or not.

Pruning Classification Tree

It is better to prune our tree and lower the number of nodes to avoid overfitting and have a better interpretation out of the tree.

```r
set.seed(10)
cv.heart = cv.tree(tree.heart, FUN = prune.misclass)
prune5 = prune.misclass(tree.heart, best = 5)
par(mfrow=c(1,2))
plot(cv.heart$size, cv.heart$dev, type = "b")
plot(prune5); text(prune5, pretty = 0)
```





Pruning decision trees minimizes their size by deleting sections of the tree that don't have enough ability to categorize instances. From the plot on the left hand side, it appears the optimal tree size would be 5 since it corresponds the smallest cross-valication error. With 5 terminal nodes, our pruned tree is left with `Oldpeak`, `MaxHR`, and `Cholesterol` variables. Therefore, these predictors will be considered significant in our decision tree model.

To evaluate the performance of the pruned tree, we will use the same approach as for the unpruned tree, which is repeatedly calculating the test error rate 10 times with different subsets of training and testing data and find the average.

```
##  [1] 0.1902174 0.2336957 0.1956522 0.1793478 0.1847826 0.1847826 0.2173913
##  [8] 0.2282609 0.2336957 0.2065217
```

```
## [1] 0.2054348
```

After 10 iterations, we obtained all 10 MSEs with the same value of `0.2054348`. Hence, the average MSE is `20.54%`, which is slightly better than what we had initially with the unpruned tree. We can affirm that tree with 5 nodes is performing better in predicting heart diseases.

Random Forest (Han Hoang)

Previously, we can see that the decision trees can possibly give lower prediction accuracy and higher variance every time we fit a classification tree. To tackle these issues, we decided to use random forest approach to analyze the data and obtain the test MSE. Random forests first construct B = 500 large un-pruned trees, and each time a tree split is considered, it picks a random subset of $m = \sqrt{p}$, which is approximately 3 predictors from the full set of 11 predictors. This is an improvement over bagging via decorrelation, hence stabilizing the variance of the estimate. Finally, the final prediction is determined by counting the majority vote across all B trees since we are working on a classification task. Similar to the classification tree, the form of the random forest model is:

$$HeartDisease \sim Age + Sex + ChestPainType + RestingBP + Cholesterol + FastingBS$$

$$+RestingECG + MaxHR + ExerciseAngina + Oldpeak + ST\_Slope$$

```
set.seed(10)
rf.model = randomForest(as.factor(HeartDisease)~.,
                         data = heart,
                         mtry = sqrt(11),
                         importance = TRUE)
rf.model
```

```
##
## Call:
##  randomForest(formula = as.factor(HeartDisease) ~ ., data = heart,      mtry = sqrt(11), importance =
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 12.85%
## Confusion matrix:
##     0   1 class.error
## 0 340  70  0.17073171
## 1  48 460  0.09448819
```

With random forests, we attain an Out-of-bag (OOB) estimate of error rate of 12.85%. Here, OOB error rate is the mean prediction error on each training sample x_i, using only the trees that did not have x_i in their bootstrap sample. Figure 1 shows that OOB error rate (black line) is stabilized with an increase in the number of trees.

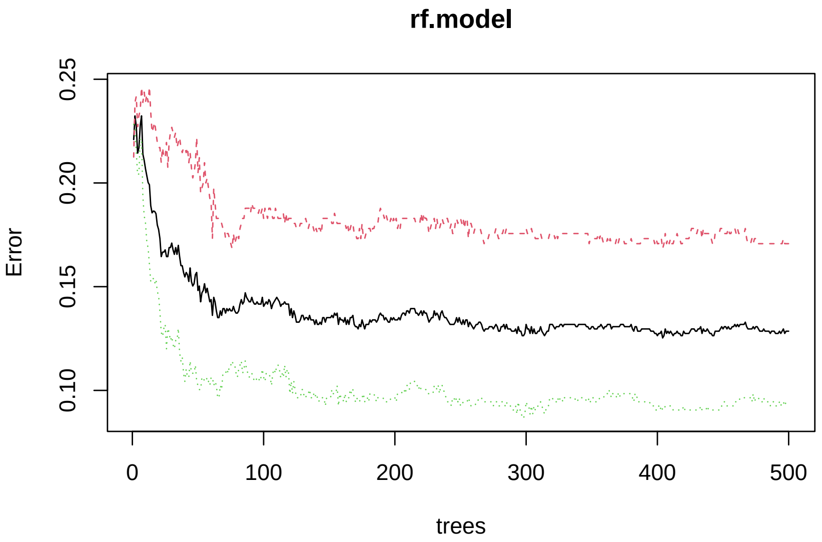


Figure 1: Number of trees vs. OOB error rate

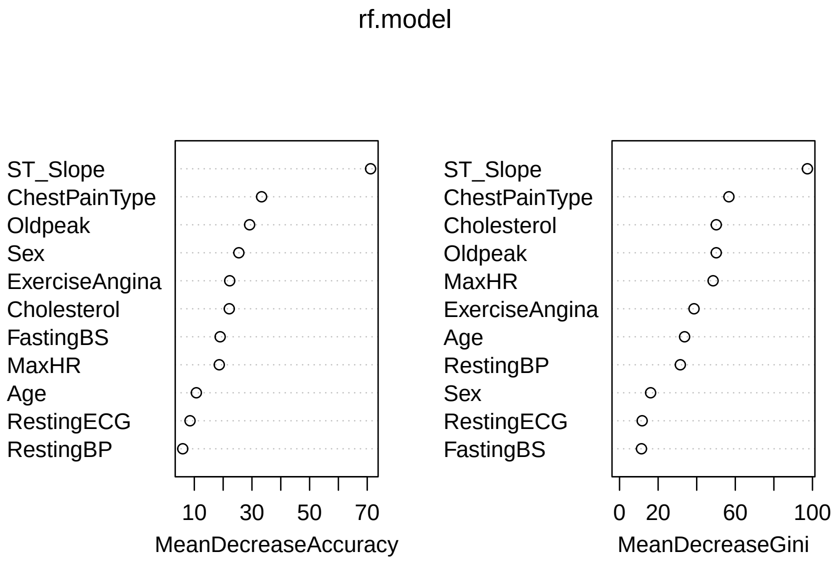


Figure 2: Variable Importance Measurement

In terms of variable importance measurement, we use `varImpPlot()` function to determine the importance of each variable. Note that the larger the value of DecreaseGini, the more important that variable is. Therefore, from the *MeanDecreaseGini* plot, it is visible that the most important variables is `ST_Slope`, followed by `ChestPainType`, `Cholesterol`, `MaxHR` and `Oldpeak`. These high-ranking predictors appear to be quite similar to those important predictors in the decision tree. In particular, the pruned tree picks `Cholesterol`, `MaxHR` and `Oldpeak` as the most important variables.

```
MSE = rep(0,10)
for (i in 1:10){
  set.seed(i)
  train = sample.int(n = nrow(heart),size = round(.8*nrow(heart)),replace = FALSE)
  Heart.train = heart[train,]
  Heart.test = heart[-train,]
  rf.model = randomForest(as.factor(HeartDisease)~.,
                          data = Heart.train,
                          mtry = sqrt(11),
                          importance = TRUE)
  rf.yhat = predict(rf.model, newdata = Heart.test, type = "class")
  (conf.mat = table(rf.yhat, Heart.test$HeartDisease))
  MSE[i] = (conf.mat[1,2]+conf.mat[2,1])/sum(conf.mat)
}
MSE
```

```
##  [1] 0.1195652 0.1576087 0.1358696 0.1032609 0.1304348 0.1358696 0.1250000
##  [8] 0.1358696 0.1467391 0.1521739
```

```
mean(MSE)
```

```
## [1] 0.1342391
```

Finally, we randomly divide the dataset using 80/20 split to train the model on the 80% training data, and record the test error rate on left out 20% testing data. After 10 iterations of training and testing, we obtained an average test MSE of 0.1342 (13.42%). Notably, this is a better rate than the results obtained from the other two models.

Performance evaluation

Model	Test error
Multiple Linear Regression	20.60%
Decision Tree	20.54%
Random Forest	13.42%

After obtaining the MSE of all three models, we can conclude that Multiple Linear Regression has the largest error rate of 20.60%, followed by Decision Trees with MSE of 20.54%. Finally, Random Forests have the best test error rate of 13.42%.

Heart Failure by gender (Seonjae Baek)

In addition to the important variables, we want to explore the dataset further to see whether male or female patients have a higher chance of heart failure. Hence, we proceed to fit a logistics model with `HeartDisease` as response and `Sex` as predictor to determine which gender has a higher risk of heart disease. Our initial model formula would be:

$$P(x) = \begin{cases} \frac{exp(\beta_0+\beta_1)}{(1+exp(\beta_0+\beta_1))} & \text{if Male} \\ \\ \frac{exp(\beta_0)}{(1+exp(\beta_0))} & \text{if Female} \end{cases}$$

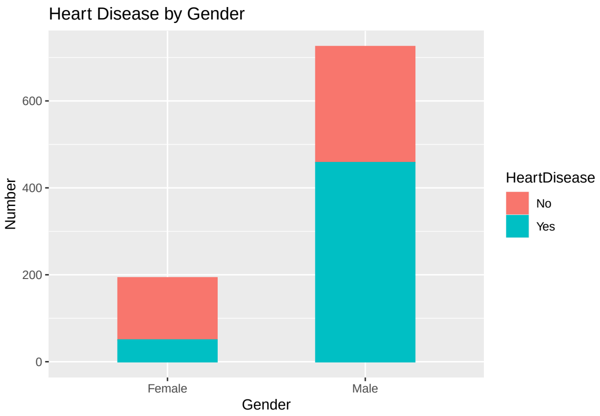
```r
model = glm(HeartDisease ~ Sex, family = "binomial", data = heart)
```

```
##               Estimate Std. Error   z value     Pr(>|z|)
## (Intercept) -1.050822  0.1642953 -6.395931 1.595726e-10
## SexM         1.590442  0.1814433  8.765503 1.859444e-18
```

After fitting the model, we obtain that $\beta_0 = -1.0508$ and $\beta_1 = 1.5904 > 0$. This means there is a higher chance of heart failure in male patients. With this data, we can now calculate the probability of male and female patients having heart disease in the data set.

$$P(x) = \begin{cases} \frac{exp(-1.0508+1.5904)}{(1+exp(-1.0508+1.5904))} = 0.6317 & \text{if Male} \\ \\ \frac{exp(-1.0508)}{(1+exp(-1.0508))} = 0.2591 & \text{if Female} \end{cases}$$

The result shows that there is a 63.17% chance of male patients having heart diseart and the rate is much lower for female patients, which is 25.91%. These data can also be visualized in the following figure.



Conclusion (Quang Du)

In conclusion, out of the three models above, Random Forest is our best model for predicting if the patient has heart disease because it has the lowest MSE of 13.42%. Unfortunately, we did not have time to do a neural network model because it would be interesting to see what MSE that model would yield. Regarding significant predictors, we determined predictors `Oldpeak`, `ChestPainType`, `Cholesterol`, and `MaxHR` to be the best indicator to determine if someone has heart disease since these four repeatedly appear in all three of our models. While working on the data, our group also inferred that males are almost three times more likely to have heart disease than females. Further research suggests this is also one of the significant factors in contributing to females having a life expectancy of five years longer than guys on average.