INFO-Assignment2

Daniel Prvanov

2022-08-14

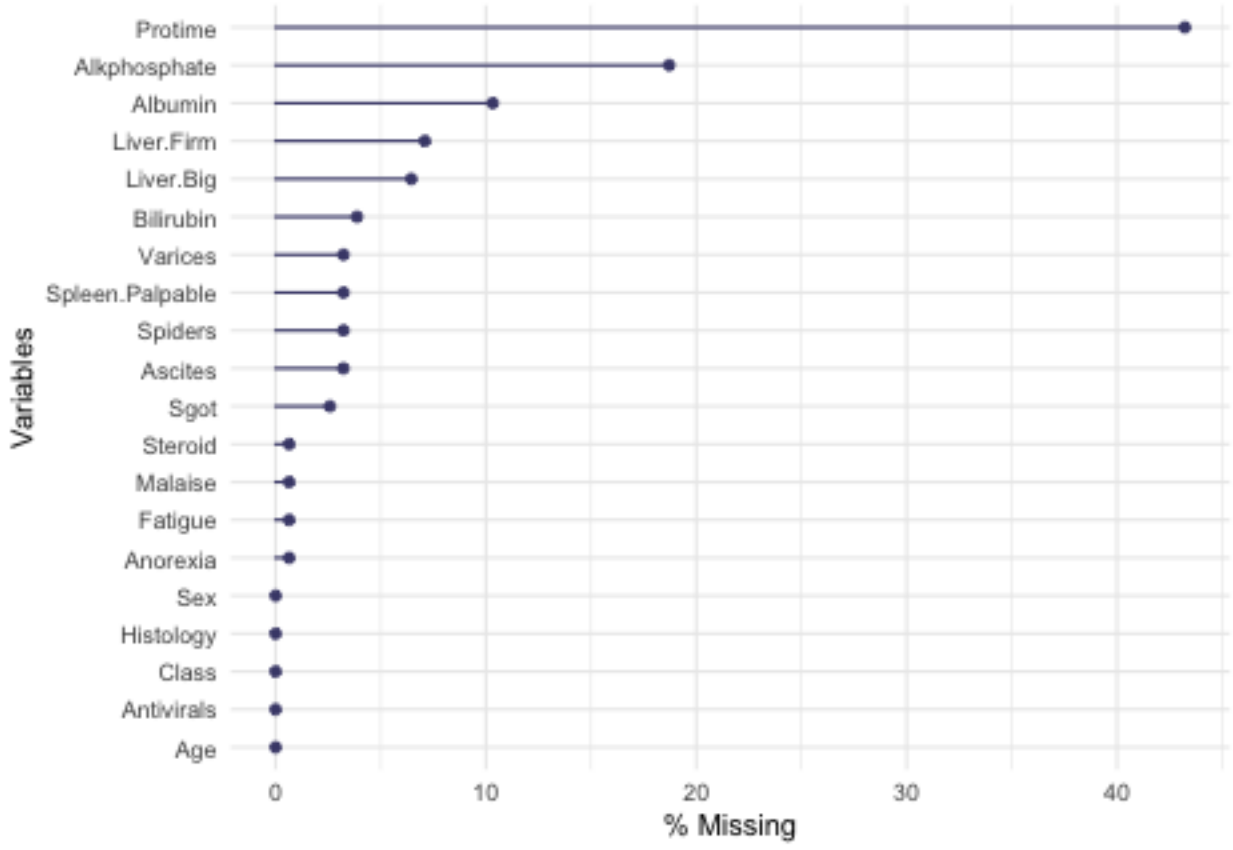
R Markdown

```
library(naniar)  # install.packages("naniar") if you don't have it.
library(ggplot2) # package naniar produces ggplots
```

```
hepatitis <-read.csv("hepatitis.data")
hepatitis <- data.factorise(hepatitis, factor.cols = c(1,3:14,20))
```

Question One (10 marks)

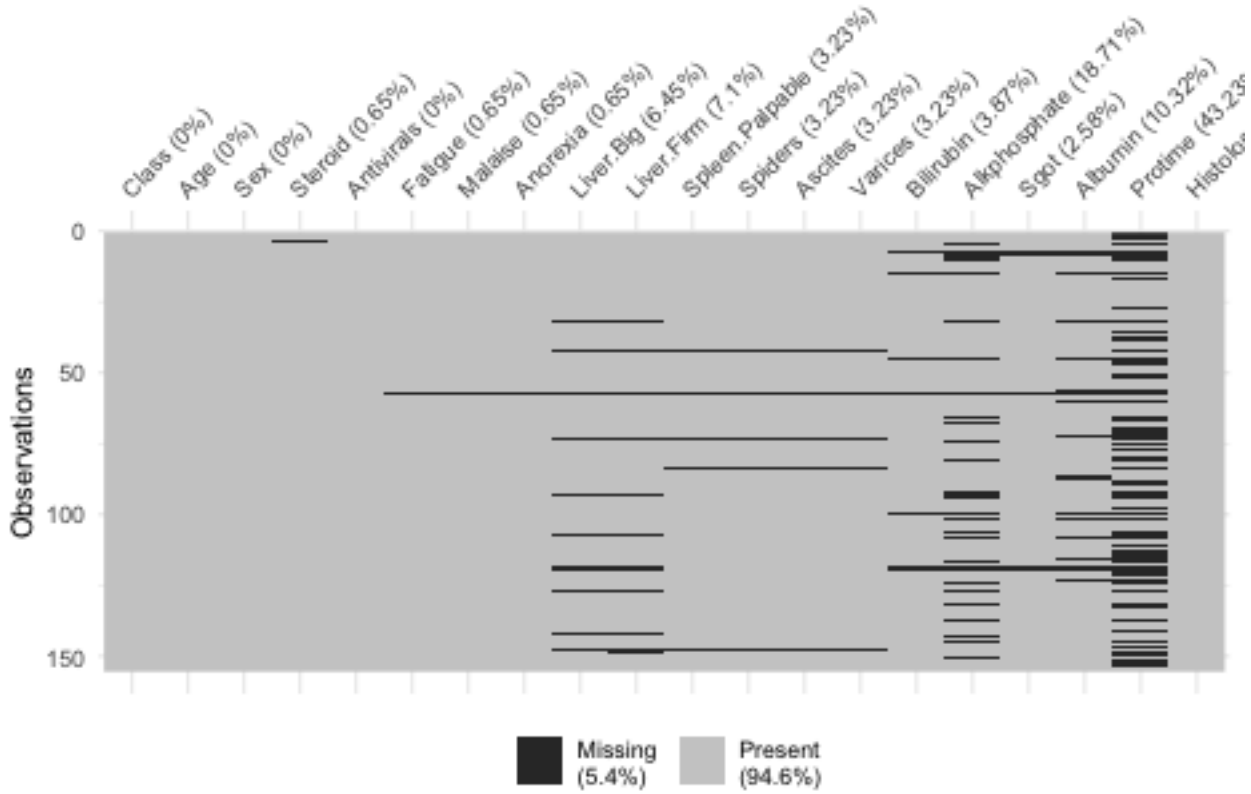
```
gg_miss_var(hepatitis,show_pct=TRUE)
```



```
print(paste("Number of NA's in Hepatitis:",length(which(is.na(hepatitis)==TRUE))))
```

```
## [1] "Number of NA's in Hepatitis: 167"
```

```
vis_miss(hepatitis)
```



From this we can see that a large percentage of variables of "Protime" are missing (43.23%) and a total of 167 variables (5.4%) are missing. We can also see that that a few observations are missing several rows worth of data, there is a row around the 50th observation missing all variables apart "Class", "Age", "Sex" and "Steroid" it probably would be smart to remove this observation as it does not contain much information. The variables "Age", "Antivirals", "Class", "Histology", and "Sex" are all missing in zero observations.

Question Two (10 marks)

The explanatory variable "Sex" needs to be removed because all observations that Sex is female (two), class is also equal to Survived (two) Therefore they are directly correlated and the model would predict that if someone is female that they will also always survive, which in the real world is not correct and is only correct here to due to the small sample size of females. That is why "Sex" needs to be removed.

```
hepatitis <- subset(hepatitis, select = -c(Sex))
```

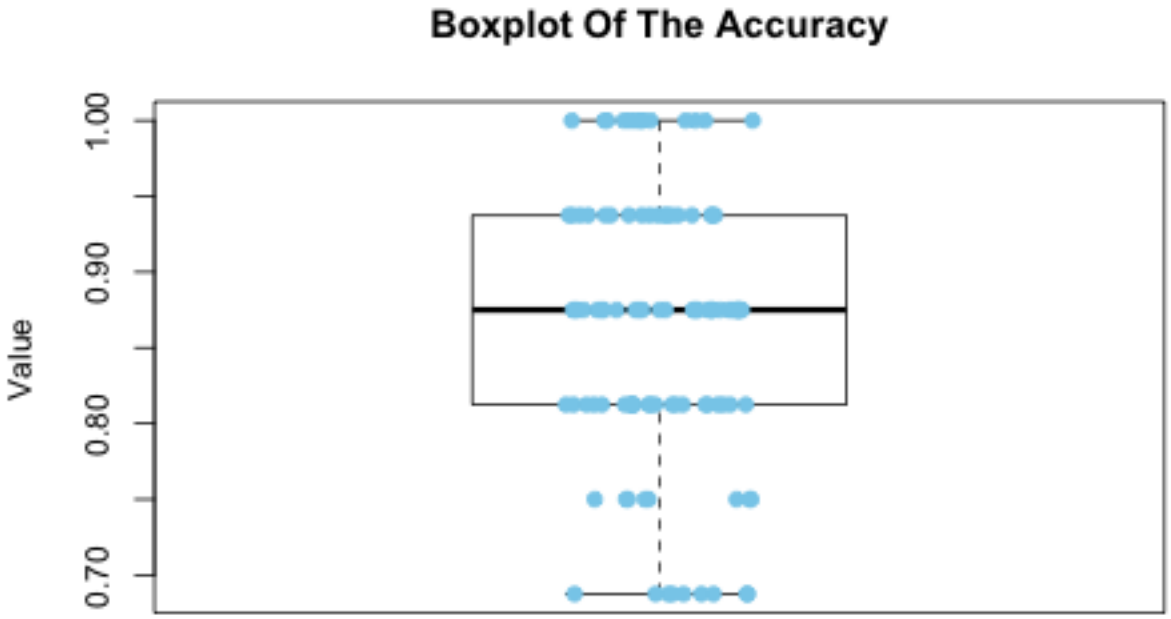
Question Three (45 marks) - Logistic Regression

I Have already turned variables to factors and removed the explanatory variable "Sex".

```
library(mice)
imps <- make.imputations(hepatitis) #3.3 Create an imputed data set for the hepatitis data set
x = replicate(100,test.logistic(imps, Class ~ .)) #3.4 Estimate the accuracy of a logistic model using
```

3.4 Comment On The Accuracy Of The Model

```
boxplot(x, ylab = "Value", main = "Boxplot Of The Accuracy", col = "white") #boxplot
stripchart(x, method = "jitter", vertical = TRUE, add = TRUE, pch = 19, col = "skyblue")
```




```
print(paste("the mean of the accuracy is",mean(x))) #mean accuracy
```

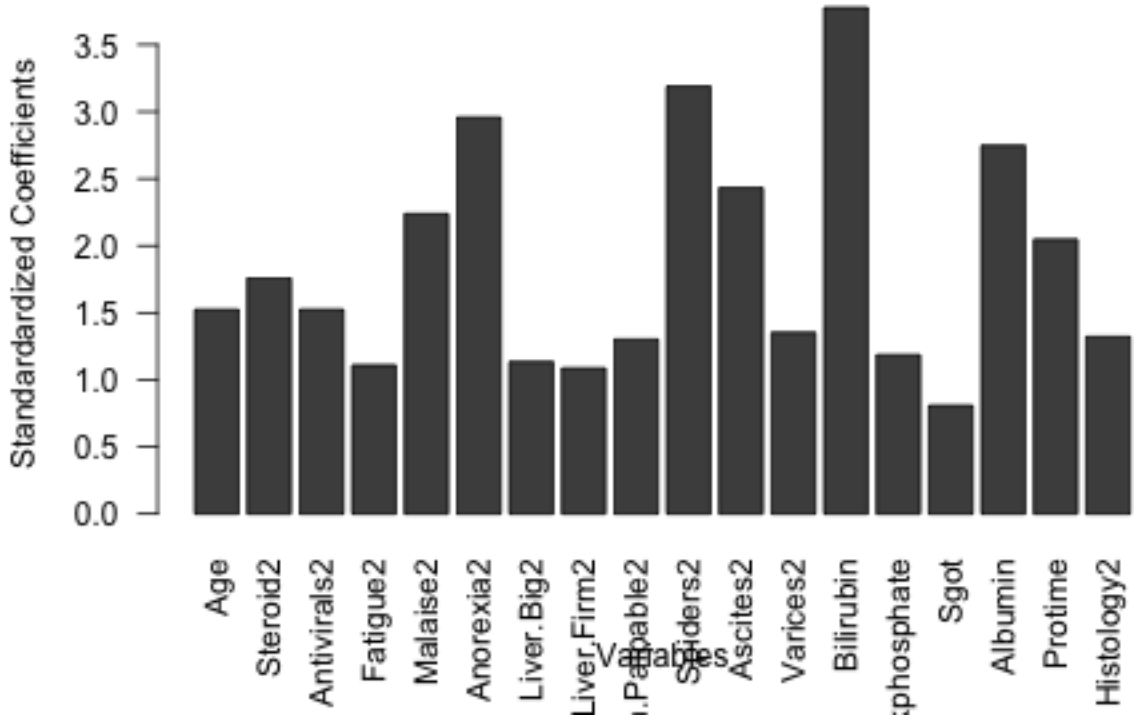
```
## [1] "the mean of the accuracy is 0.861875"
```

When running the function 100 times I got a mean accuracy of 0.85875 (Note: the accuracy is not always 0.85875, as if rerun make.imputations I get similar but different results). In the Breiman paper they had a predictive error rate of 17.4, or an accuracy of 0.826. From this we can see that my results have a slightly better accuracy then the paper, this could be due to the removing of the "Sex" variables, as well as imputing the data. The paper does not state what they did to deal with NA values, they might have imputed the data as well not I am not sure. The Scientific America paper produces a result that will misclassify a given patient 20% of the time, this is a pretty bad result considering theres roughly a 80% survive rate so just stating everyone will survive will give a similar misclassification rate.

3.5 Create a Barplot Showing Variable Importance For Each Explanatory

```
glm100 = replicate(100,glm.coefficients(imps, Class ~ .))
glm100 = abs(glm100)
sd = apply(glm100, 1, sd)
mean = apply(glm100, 1, mean)
barplot.values = mean/sd
```

```
barplot.values <- barplot.values[c(-1)]
barplot.values_ordered <- sort(barplot.values, decreasing = F)
barplot(t(barplot.values), las = 2, xlab = "Variables", ylab = "Standardardized Coefficients")
```



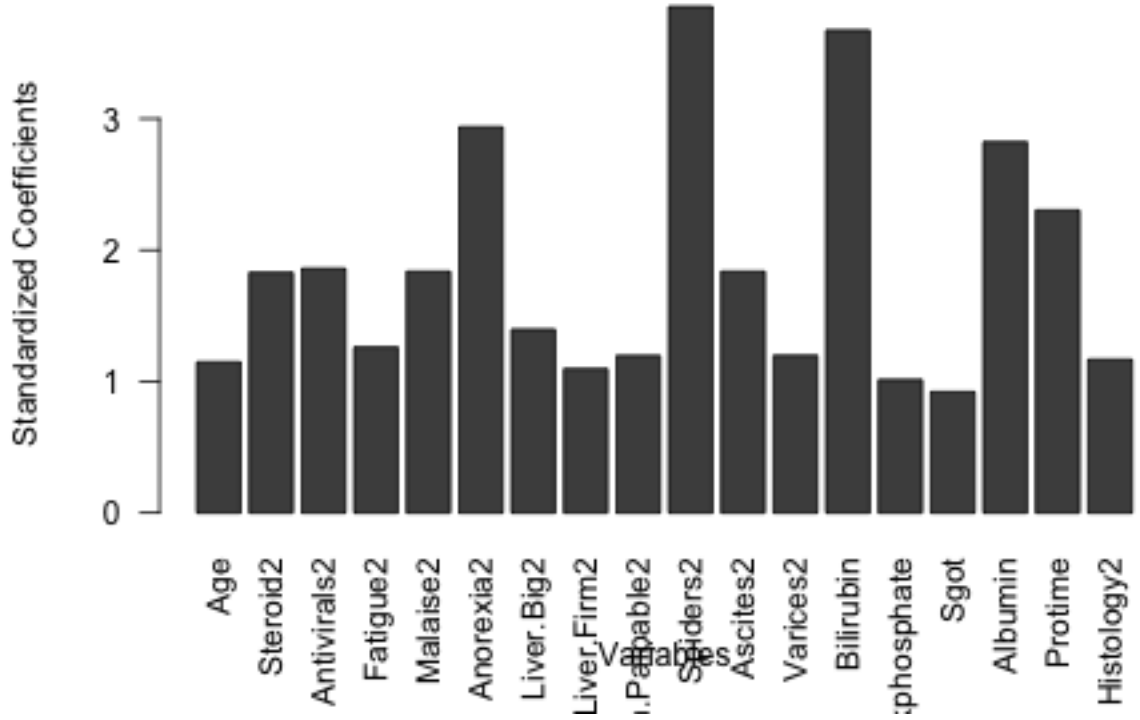
In my first graph that I created the three most important covariates were Anorexia, Ascites, and Bilrubin. In the Beriman paper it was stated variable 7 and 11 are the most important variables (they should relate to Anorexia and Spiders) In my model Anerexia is also one of the two most important covariates but Spiders is only my models four most important variable. The biggest difference between the two graphs is that my model thinks Ascites is a very important covariates while Beriman does not think it is. Note: all graphs that I describe will be different when viewed later after knitting.

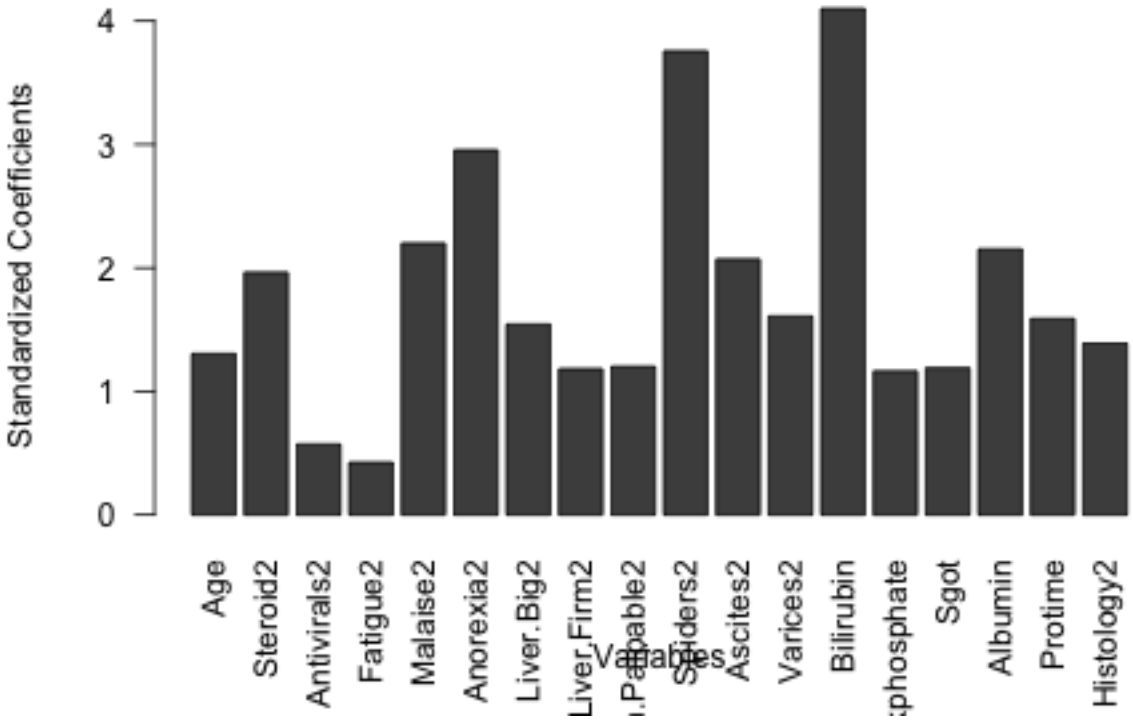
The Scientific America paper concludes that four most important variables are Malaise, Ascites, Bilirubin, and physicians prognosis (which we do not have). It is suprsing that the Beriman paper and Scientific America both have differnet variables that they consider important. But now we have a list of five variables that should be important Anorexia, Spiders, Malaise, Ascities, and Bilirubin.

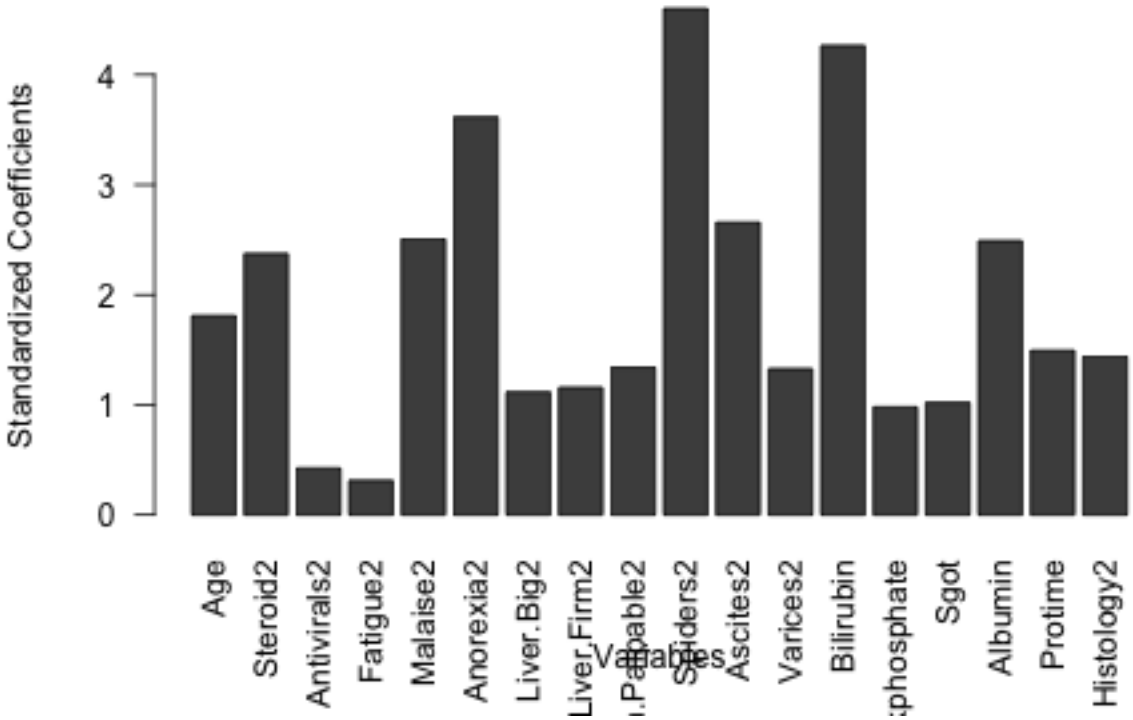
If we create five more different graphs we will get different results and check for the presence of these variables.

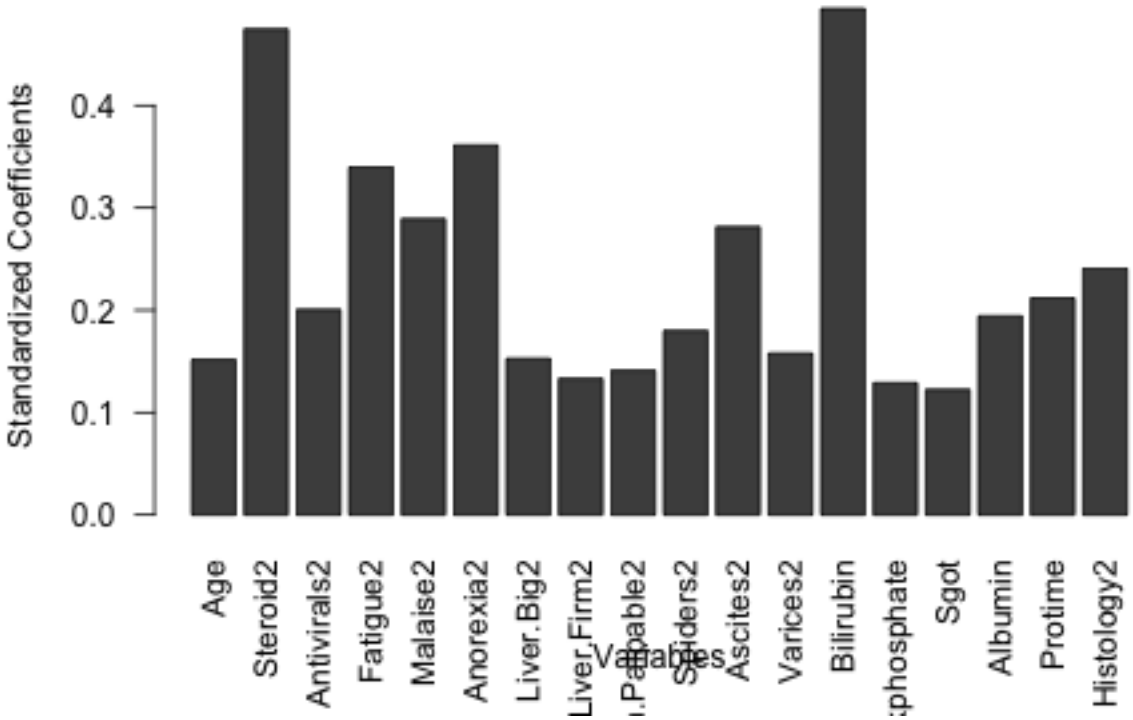
```
build.graph.values <- function(dataset, formula){
  imps <- make.imputations(dataset)
  glm100 <- replicate(100,glm.coefficients(imps, formula))
  glm100 <- abs(glm100)
  sd <- apply(glm100,1,sd)
  mean <- apply(glm100,1,mean)
  barplot.values <- mean/sd
  barplot.values <- barplot.values[c(-1)]
  barplot(t(barplot.values), las =2, xlab = "Variables", ylab = "Standardized Coefficients")
}
```

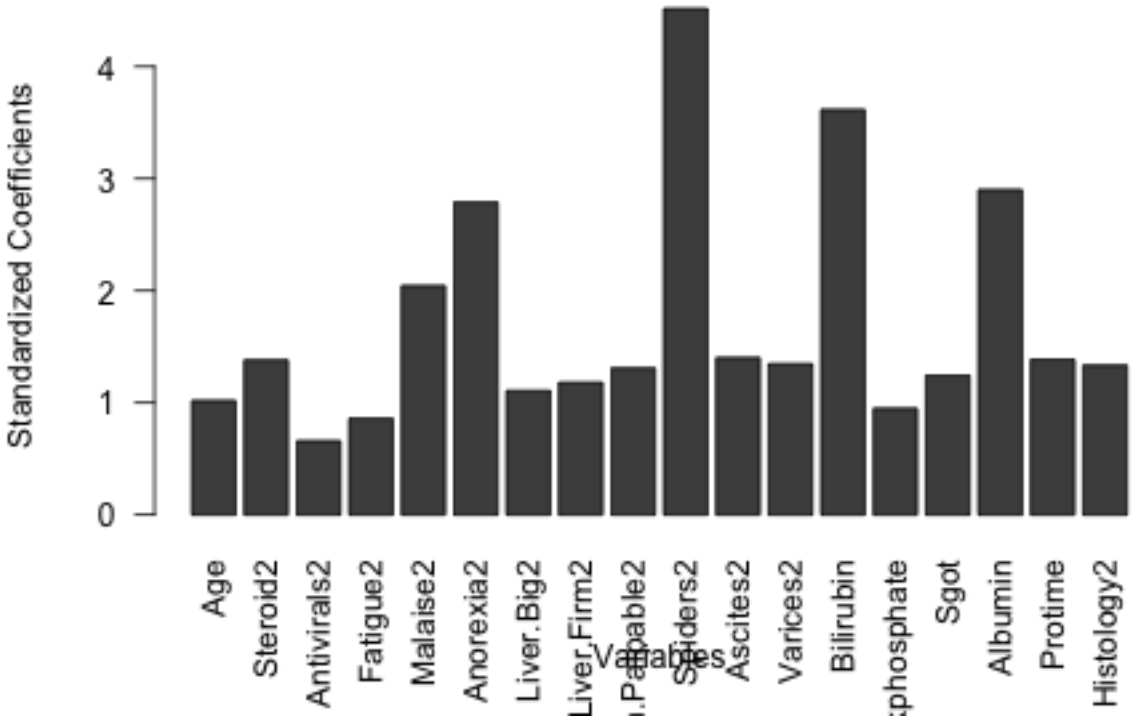
```
replicate(5,build.graph.values(hepatitis, Class ~ .))
```




Standardized Coefficients
0
1
2
3
4
Age
Steroid2
Antivirals2
Fatigue2
Malaise2
Anorexia2
Liver.Big2
Liver.Firm2
Spleen.Palpable2
Spiders2
Ascites2
Varices2
Bilirubin
Sgot
Albumin
Protime
Histology2
Variables


Standardized Coefficients
0
1
2
3
4
Age
Steroid2
Antivirals2
Fatigue2
Malaise2
Anorexia2
Liver.Big2
Liver.Firm2
Spiders2
Ascites2
Varices2
Bilirubin
Sgot
Albumin
Protime
Histology2
Variables


Standardized Coefficients
0.0
0.1
0.2
0.3
0.4
Age
Steroid2
Antivirals2
Fatigue2
Malaise2
Anorexia2
Liver.Big2
Liver.Firm2
Spiders2
Ascites2
Varices2
Bilirubin
Sgot
Albumin
Protime
Histology2




```
##       [,1] [,2] [,3] [,4] [,5]
##  [1,]  0.7  0.7  0.7  0.7  0.7
##  [2,]  1.9  1.9  1.9  1.9  1.9
##  [3,]  3.1  3.1  3.1  3.1  3.1
##  [4,]  4.3  4.3  4.3  4.3  4.3
##  [5,]  5.5  5.5  5.5  5.5  5.5
##  [6,]  6.7  6.7  6.7  6.7  6.7
##  [7,]  7.9  7.9  7.9  7.9  7.9
##  [8,]  9.1  9.1  9.1  9.1  9.1
##  [9,] 10.3 10.3 10.3 10.3 10.3
## [10,] 11.5 11.5 11.5 11.5 11.5
## [11,] 12.7 12.7 12.7 12.7 12.7
## [12,] 13.9 13.9 13.9 13.9 13.9
## [13,] 15.1 15.1 15.1 15.1 15.1
## [14,] 16.3 16.3 16.3 16.3 16.3
## [15,] 17.5 17.5 17.5 17.5 17.5
## [16,] 18.7 18.7 18.7 18.7 18.7
## [17,] 19.9 19.9 19.9 19.9 19.9
## [18,] 21.1 21.1 21.1 21.1 21.1
```

When rerunning the model again, we get different results with different variables becoming the most important.

When running the model again we get different results, this means that the stability of logistic regression for the hepatitis data set isn't the best. I wouldn't be confident using this approach to assess variable importance due to the randomness of it. But it could be used to give a rough ball park of the variables importance. For

example we might not know how important the "Spiders" variable is but both times it has appeared near the top of the graph so it most likely is important even if we don't know how important it is and the same would be true for variables that have a low score.

In the new graphs that I have created the variables that tend to be of most importance are Spiders, Ascites, Bilirubin and Malaise. Four out of the five variables that the Breiman and Diaconis/Efron papers suggested.

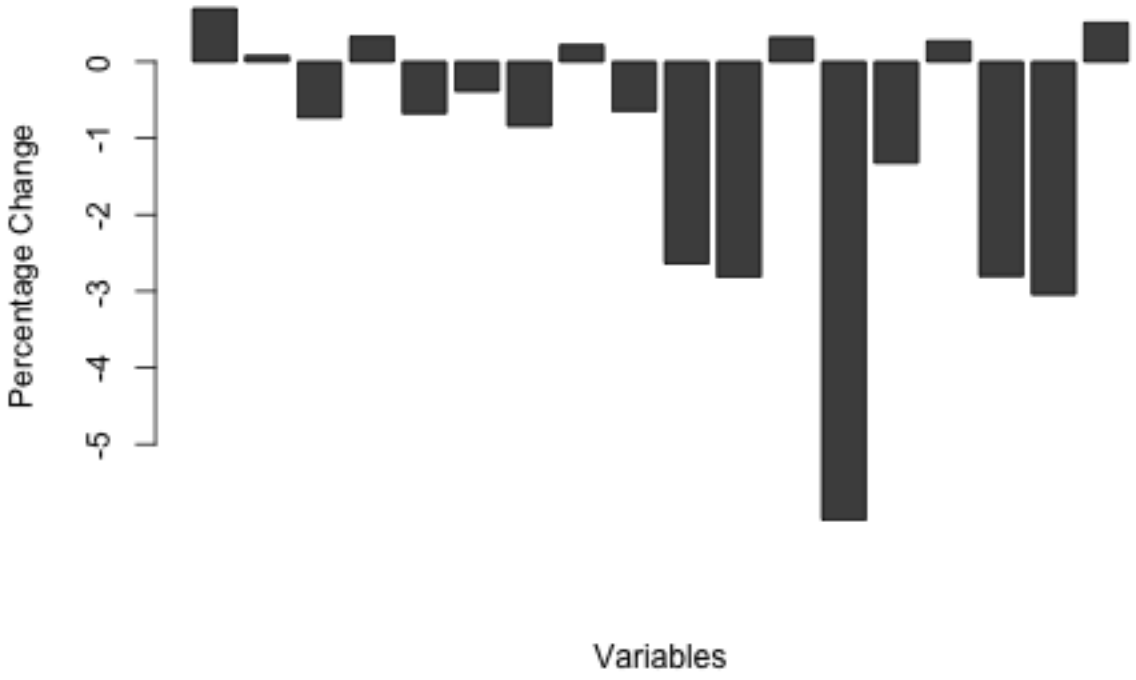
Question 3.6 (25 marks) - Implement Variable Importance Using Permutation

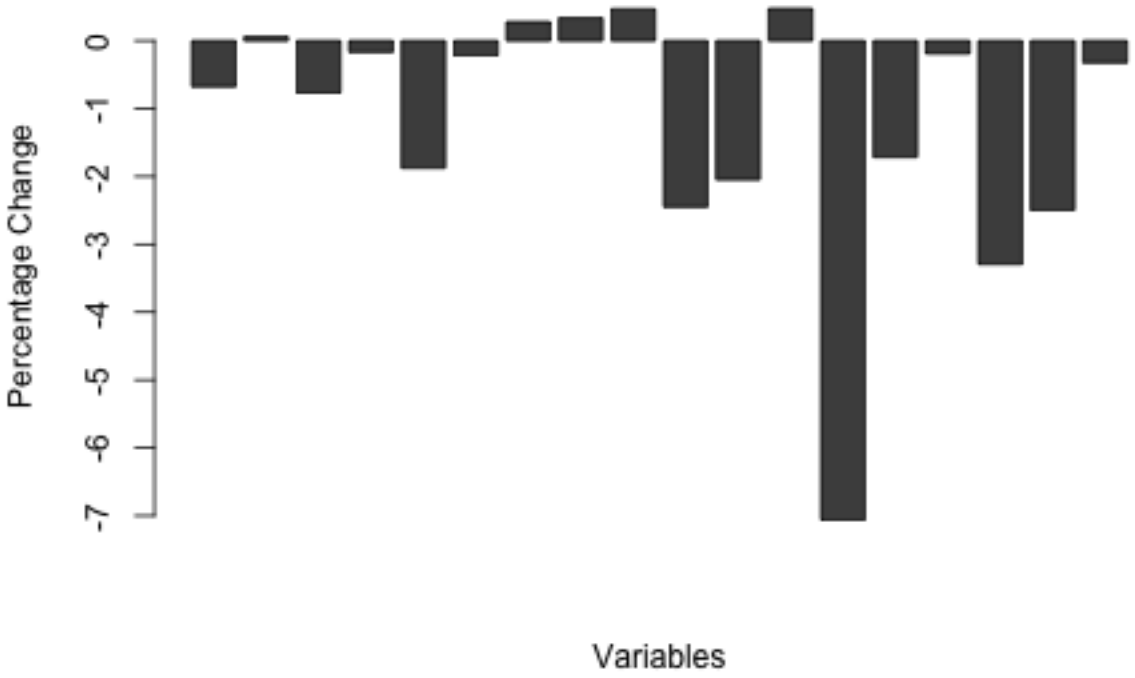
```
variable.importance <- function(imp,formula,var.col=2,perc.train=0.9)
{
  # get the training/test data randomly from one imputed datasets

  ttdata <- get.imp.train.test(imp,perc.train=perc.train)
  train <- ttdata[[1]]
  test <- ttdata[[2]]

  # Build the model...
  log.mod <- glm(formula, data=train, family="binomial")
  #
  # and now find the threshold to maximise accuracy...
  # uses the same training data (since we can't use test)
  train.pred <- predict(log.mod,newdata=train,type="response")

  # But now we can get back the threshold
  threshold <- fast.threshold(train$Class,train.pred)

  # and now test the model...
  test.pred <- predict(log.mod,newdata=test,type="response")

  # and use the threshold to calculate the final predicted values
  #
  pred.vals <- as.factor(ifelse(test.pred >=threshold,2,1))

  conf.mat <- caret::confusionMatrix(data = pred.vals,
                                     reference=test$Class)

  # computes the logistic model accuracy between 0 and 1
  original.test <- (conf.mat$table[1]+conf.mat$table[4])/nrow(test)

  #imputing the data
  newr <- sample(1:nrow(test),nrow(test), replace = FALSE)
  test[,var.col] <- test[newr,var.col] # imputed dataset
  log.mod <- glm(formula, data=train, family="binomial")
  #
  # and now find the threshold to maximise accuracy...
  # uses the same training data (since we can't use test)
  train.pred <- predict(log.mod,newdata=train,type="response")

  # But now we can get back the threshold
  threshold <- fast.threshold(train$Class,train.pred)

  # and now test the model...
```

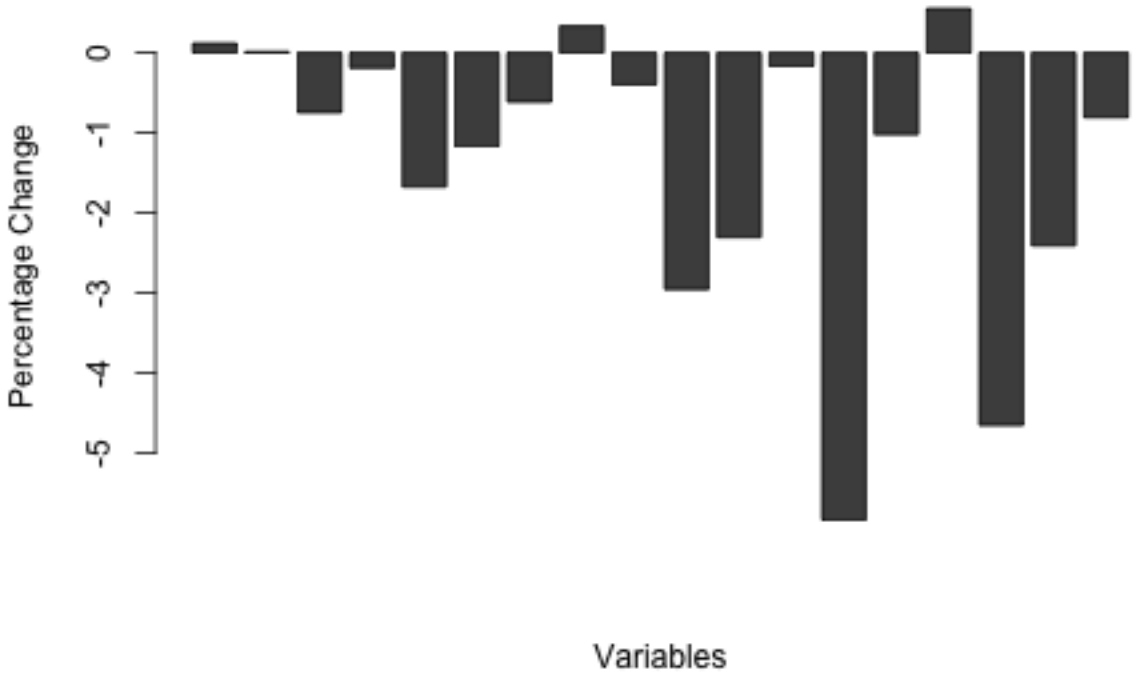
```
  test.pred <- predict(log.mod,newdata=test,type="response")

  # and use the threshold to calculate the final predicted values
  #
  pred.vals <- as.factor(ifelse(test.pred >=threshold,2,1))

  conf.mat <- caret::confusionMatrix(data = pred.vals,
                                     reference=test$Class)

  # Return accuracy between 0 and 1
  imputed.test <- (conf.mat$table[1]+conf.mat$table[4])/nrow(test)


  #calculate percentage increase
  dif <- imputed.test-original.test
  percentage.change <- dif/imputed.test*100
  percentage.change
}
```

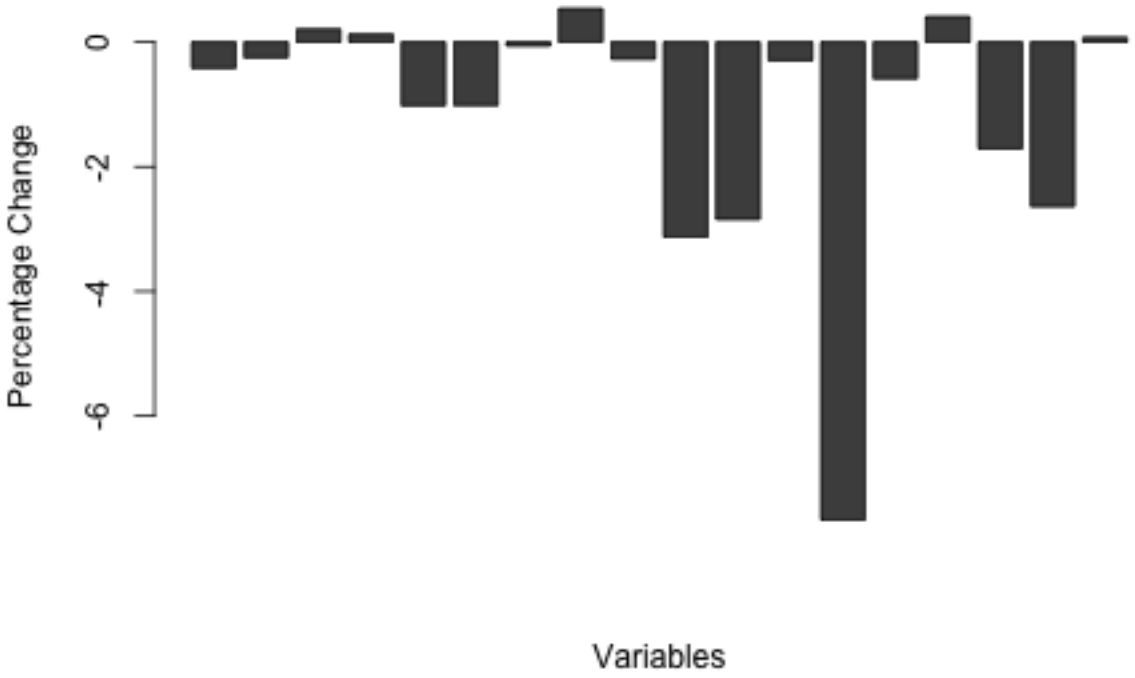
function to build graphs.

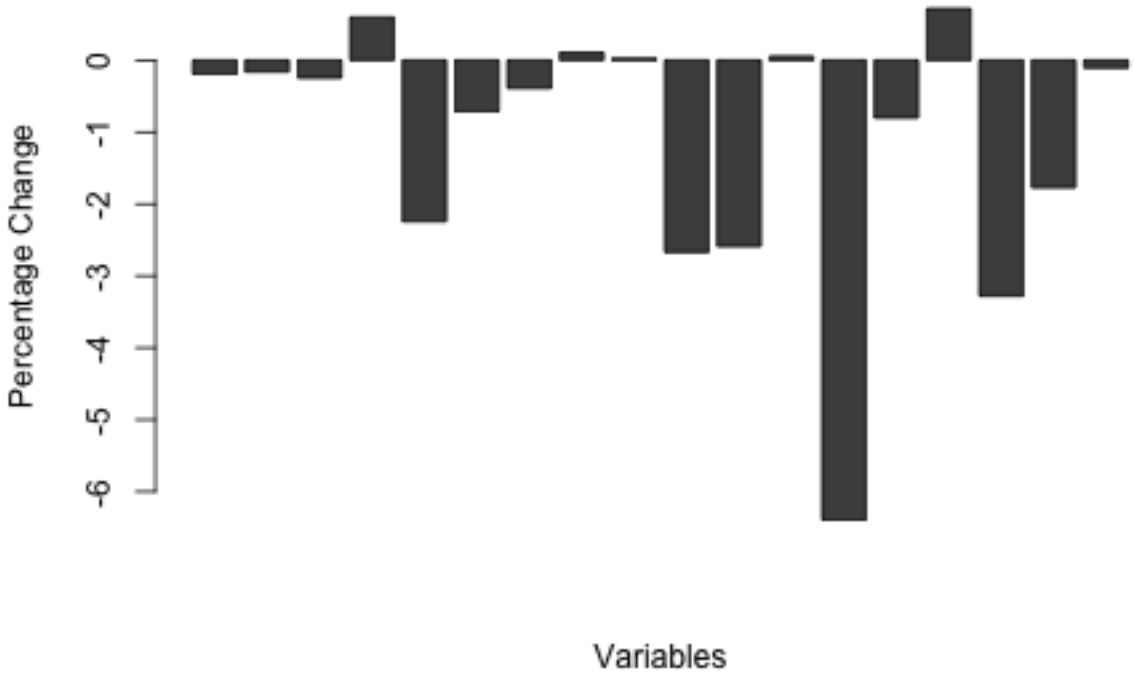
```
build.permuted.graph <- function(imp, formula){
  var.imps <- collect.var.imp(imp, formula)
  col.means <- colMeans(var.imps)
  barplot(t(col.means),xlab = "Variables", ylab = "Percentage Change")
}

replicate(5,build.permuted.graph(imps,Class ~ .))
```


Percentage Change
0
-1
-2
-3
-4
-5
Variables


Percentage Change
0
-1
-2
-3
-4
-5
-6
-7
Variables


Percentage Change
0
-1
-2
-3
-4
-5
Variables


Percentage Change
0
-2
-4
-6
Variables




```
##       [,1] [,2] [,3] [,4] [,5]
##  [1,]  0.7  0.7  0.7  0.7  0.7
##  [2,]  1.9  1.9  1.9  1.9  1.9
##  [3,]  3.1  3.1  3.1  3.1  3.1
##  [4,]  4.3  4.3  4.3  4.3  4.3
##  [5,]  5.5  5.5  5.5  5.5  5.5
##  [6,]  6.7  6.7  6.7  6.7  6.7
##  [7,]  7.9  7.9  7.9  7.9  7.9
##  [8,]  9.1  9.1  9.1  9.1  9.1
##  [9,] 10.3 10.3 10.3 10.3 10.3
## [10,] 11.5 11.5 11.5 11.5 11.5
## [11,] 12.7 12.7 12.7 12.7 12.7
## [12,] 13.9 13.9 13.9 13.9 13.9
## [13,] 15.1 15.1 15.1 15.1 15.1
## [14,] 16.3 16.3 16.3 16.3 16.3
## [15,] 17.5 17.5 17.5 17.5 17.5
## [16,] 18.7 18.7 18.7 18.7 18.7
## [17,] 19.9 19.9 19.9 19.9 19.9
## [18,] 21.1 21.1 21.1 21.1 21.1
```

We can use permutation to check for variable importance, When we permute a variable, if that variable is important there should be a large negative percentage change in the accuracy. From the graphs that I ran, in all five of them the same three variables all had significant accuracy decreases. and addition two variables showed significant change in three of the runs. Because of this we can say the stability of permutation is pretty good as all graphs produce similiar results. The three variables that were always considered important

were Spiders, Ascities and Bilirubin and the two variables that sometimes were considering important were Malaise and Anorexia. These are the five variables that the Breiman and Diaconis/Efron papers considered to be the most important.

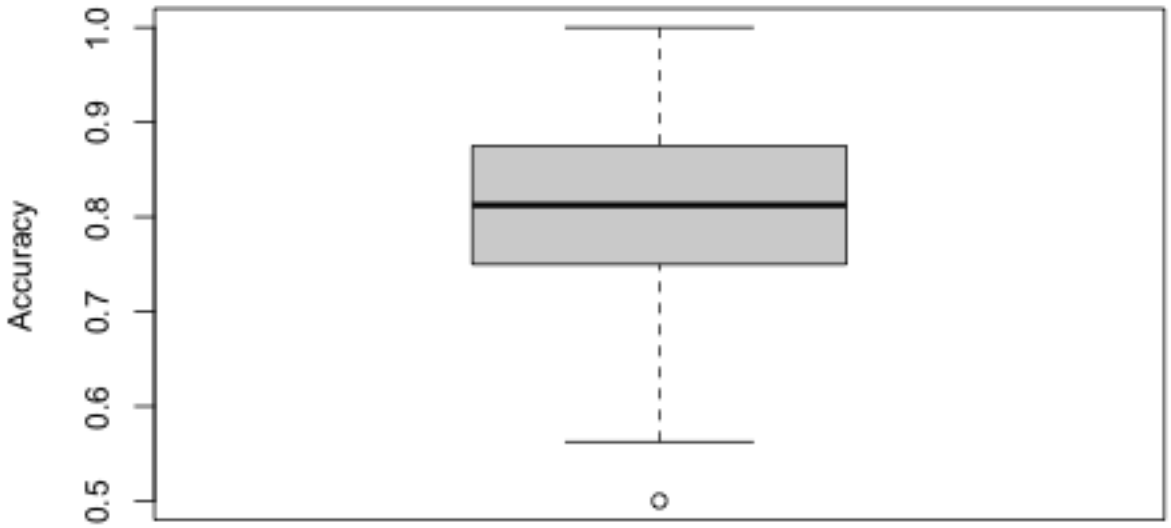
compare to other literature tomorrow

Question Four (20 marks) - Decision Trees and Random Forest

4.1

```
test.rpart <- function(imp, formula,perc.train=0.9)
{
  ttdata <- get.imp.train.test(imp, perc.train = perc.train)
  train <- ttdata[[1]]
  test <- ttdata[[2]]

  # Build the model
  log.mod <- rpart(formula = formula, data = train)

  train.pred <- predict(log.mod,newdata=train,type="vector")
  test.pred <- predict(log.mod,newdata=test,type="vector")
  pred.vals <- as.factor(test.pred)
  conf.mat <- caret::confusionMatrix(data = pred.vals,
                                     reference=test$Class)
  # Return accuracy between 0 and 1
  (conf.mat$table[1]+conf.mat$table[4])/nrow(test)
}

single.decisiontree <- replicate(100,test.rpart(imps, Class ~ . ))
```

```
boxplot(single.decisiontree, ylab = "Accuracy")
```




```
print(paste("The mean is",mean(single.decisiontree)))
```

```
## [1] "The mean is 0.79375"
```

```
print(paste("The sd is",sd(single.decisiontree)))
```

```
## [1] "The sd is 0.105670956322449"
```

A single decision tree roughly has a mean accuracy rate of only 80%, this is really bad considering predicting that everyone will survive will have give you an accuracy of around 80% as well. When comparing this to logistic regression this is noticeably worse (logistic regression has roughly mean accuracy of 85%). The standard deviation of the single decision tree and logistic regression are both roughly 0.08.

##4.2 The random forest function works but first splitting the data into training and testing data sets then bootstraps the training data set for each tree. Using the bootstrapped data to create a tree. Then testing the test data on the bootstrapped training data to get the accuracy.

```
tree.one <- replicate(100,test.rf(imps, Class~., ntrees = 1))
tree.five <- replicate(100,test.rf(imps, Class~., ntrees = 5))
tree.ten <- replicate(100,test.rf(imps, Class~., ntrees = 10))
tree.twenty <- replicate(100,test.rf(imps, Class~., ntrees = 20))
```

```
print(paste("the mean when having one tree is",mean(tree.one)))
```

```
## [1] "the mean when having one tree is 0.785625"
```

```
print(paste("the mean when having five trees is",mean(tree.five)))
```

```
## [1] "the mean when having five trees is 0.823125"
```

```
print(paste("the mean when having ten trees is",mean(tree.ten)))
```

```
## [1] "the mean when having ten trees is 0.8325"
```

```
print(paste("the mean when having twenty trees is",mean(tree.twenty)))
```

```
## [1] "the mean when having twenty trees is 0.82875"
```

From the results we can see that adding more trees typically increases the accuracy of the results. The mean accuracy of the model with twenty trees when I ran it was 0.838125 this is greater then the single decision tree but is still worse then the logistic regression.

Question Five (15 marks) - Explain The Steps That You Would Take To Understand The Dataset, Prior To Building A Model For Prediction

If I was given this dataset I would first look at the raw data in a text editor file, checking for missing variables. Then checking how many missing values there exactly are and where they are in Rstudio, doing something similar to what I did in question one. If necessary either imputing the data or deleting observations. Imputing the data if an observation is only missing a few variables and deleting it if it is missing a large percentage of variables. But before imputing or deleting the data, I would create a correlation matrix. Checking if any variables are correlated. Just looking at the data I would assume that Income of customer and Average monthly credit card spending might be correlated. And if they were it might be possible to delete one of these variables. I would also visualize each variable as a boxplot, checking for outliers in the data, making sure that none of the numerical variables are negative or too great of an outlier. I would also check to make sure no one's years of experience is greater than their Age and no one that has a credit card has any credit card spending, as these should not be possible. I would also need to check to make sure no variables are perfectly correlated to the response, like the "Sex" variable in the Hepatitis data set. For example if everyone that does not own a credit card are also bad credit risks the Credit Card variable will need to be deleted. And finally before building the model I would convert Credit Card, Educational Level and Credit Risk into factor variables, all of the other variables would stay as continuous variables. And standardize the data if the model I want to build will require it.