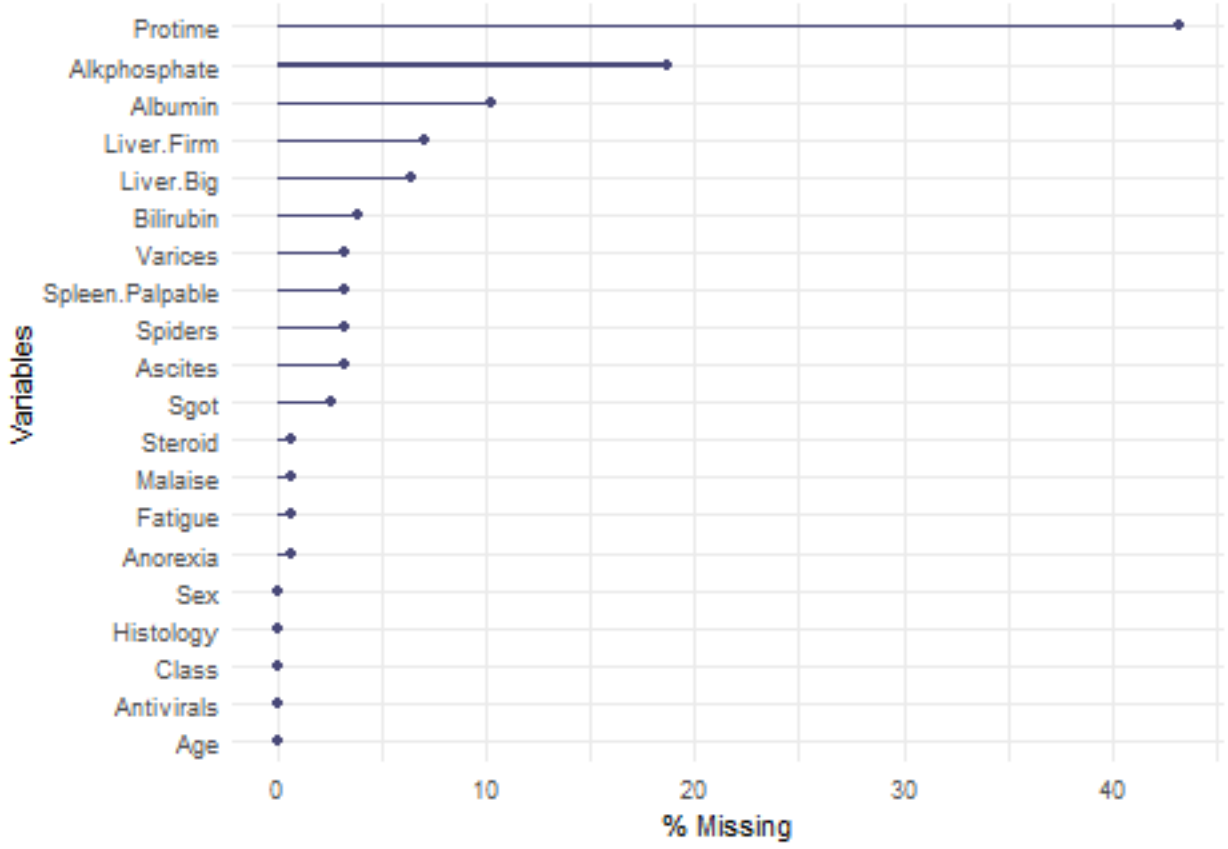
INFO304 Assignment 2 2022

Ethan Smith - 5652106

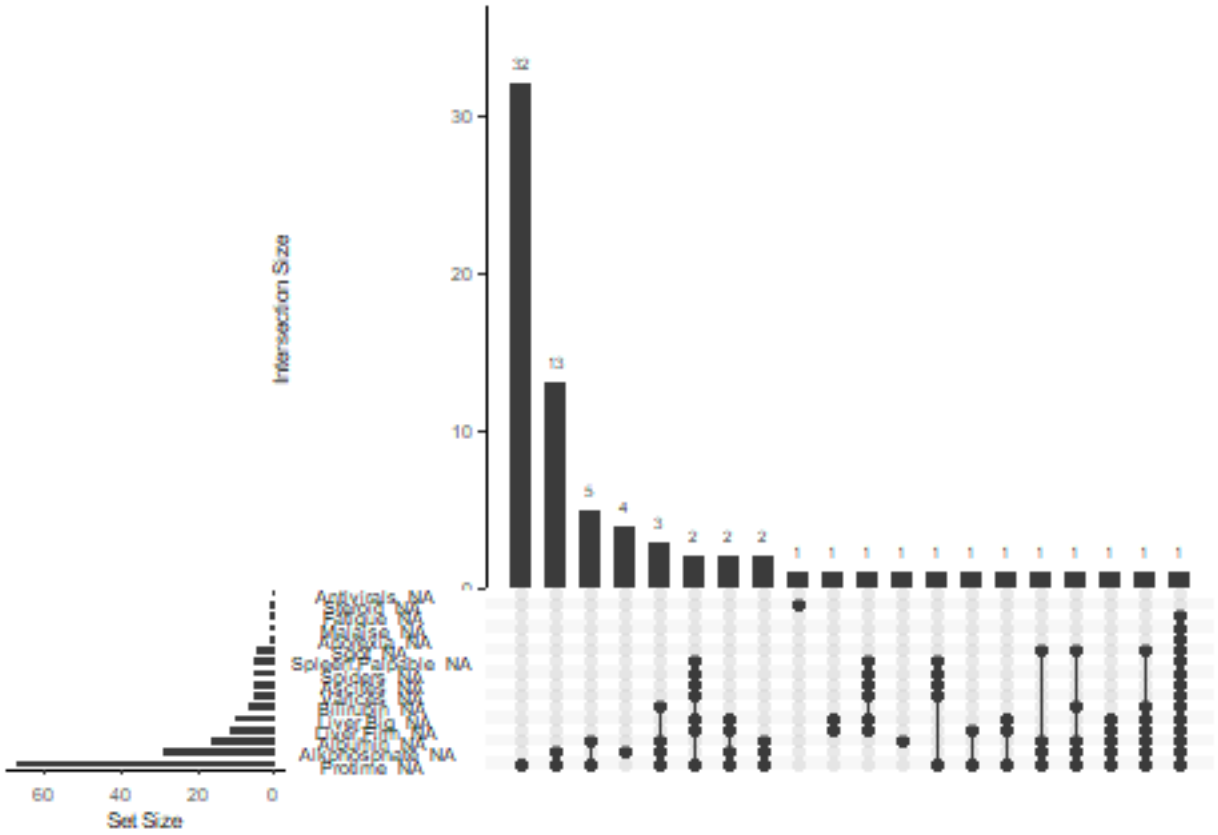
QUESTION ONE (10 MARKS)

Visualise the patterns of missing data for hepatitis.data and briefly comment on the patterns you observe.

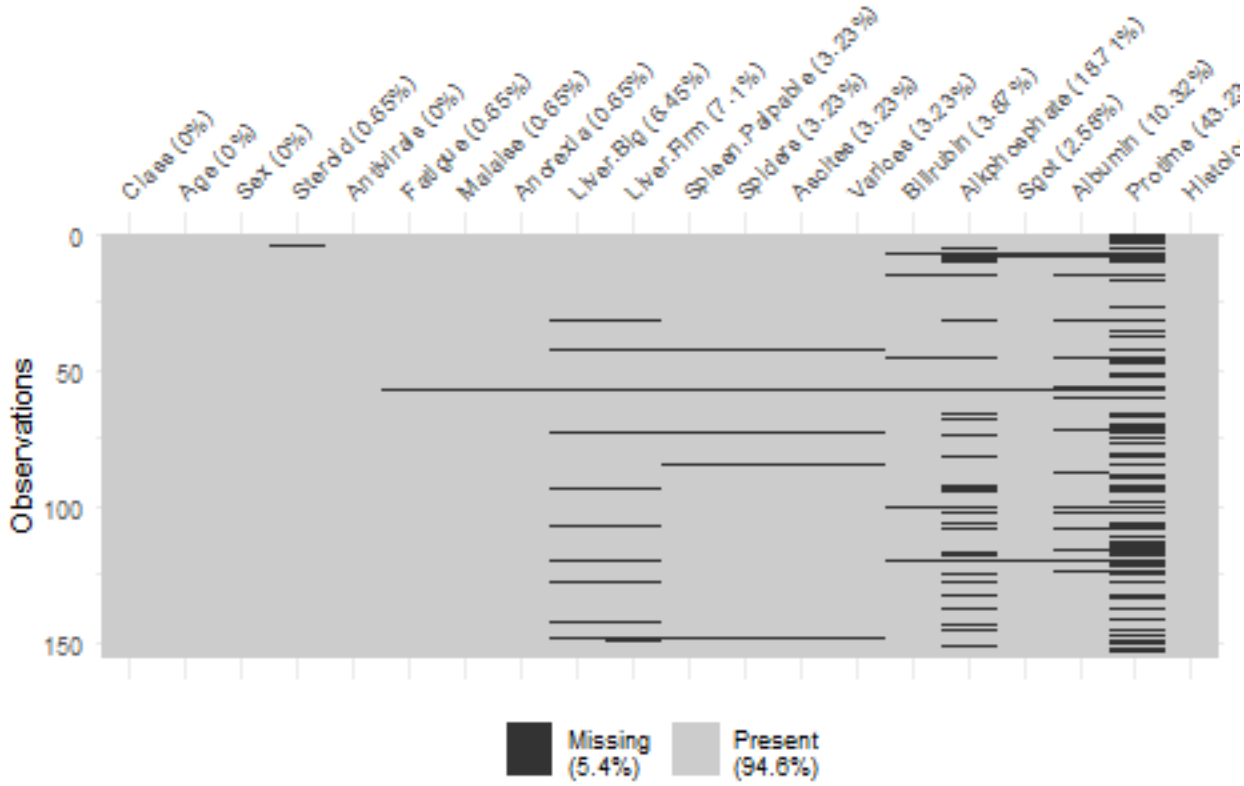
```
library(naniar)
library(ggplot2)
Hep <- read.csv("hepatitis.data")
gg_miss_var(Hep,show_pct=TRUE)
```




```
gg_miss_upset(Hep,nsets=ncol(Hep),nintersects=NA)
```


Intersection Size
32
13
5
4
3
2
2
2
1
1
1
1
1
1
1
1
1
1
1
30
20
10
0
Antivirals NA
Prednisone NA
Set Size
60
40
20
0

```
vis_miss(Hep)
```



Here we can see that within the entire dataset 5.4% of the values are missing. By summing the counts of each observation with NA values from the 2nd plot there are 75 observations in the dataset that are missing at least one value for a predictor (just under 50% of all observations). The three variables which are missing the most values are protime, alkphosphate and albumin. From the 2nd plot we can also see that combinations of these variables are also the most likely to be missing in comparison to other variables.

I researched into these variables and found they are all found in blood so within this sample there may have been an issue around obtaining results from blood tests of patients. The biggest concern here is for the protime predictor variable which has not been recorded for almost half the observations in the dataset. Even with methods such as imputation to help predict values for missing data this variable should be treated very carefully as there is no certainity these value will be representative of the true distribution.

It appears overall that medically based measurements that will be more unique to a patient are more likely to be missing than basic patient information but the majority of them have a small proportion of missing values that should be reasonable to account for.

QUESTION TWO (10 MARKS)

Justify why the explanatory variable "Sex" should be removed from the data prior to modelling.

```
t <- table(Hep$Class, Hep$Sex)
colnames(t) = c("Male", "Female")
rownames(t) = c("Died", "Lived")
t
```

```
##
##         Male Female
##   Died    32      0
##   Lived  107     16
```

From this table between class and sex it can be shown that sex is irrelevant for predicting whether a patient lives or dies. Overall we can see that the probability of surviving is much higher than the probability of dying, as this is combined with a very unbalanced level of males and females in the dataset it has given a set of results where all of the female patients survived and therefore there is nothing associated with a female patient dying.

Due to the imbalance between males and females in the dataset this could also cause another issue where if a random train test split is made it is possible that no females could be included in a possible training set which would cause errors for prediction. Both issues would have been likely to not exist if a larger sample of females was included but with the observations given it is clear sex should be removed as a predictor.

QUESTION THREE (45 MARKS) - Logistic Regression

Perform the following steps 3.1 Load in the hepatitis data.

```
hep <- read.csv("hepatitis.data")
```

3.2 Turn the response and appropriate explanatories into factors and remove the "Sex" column.

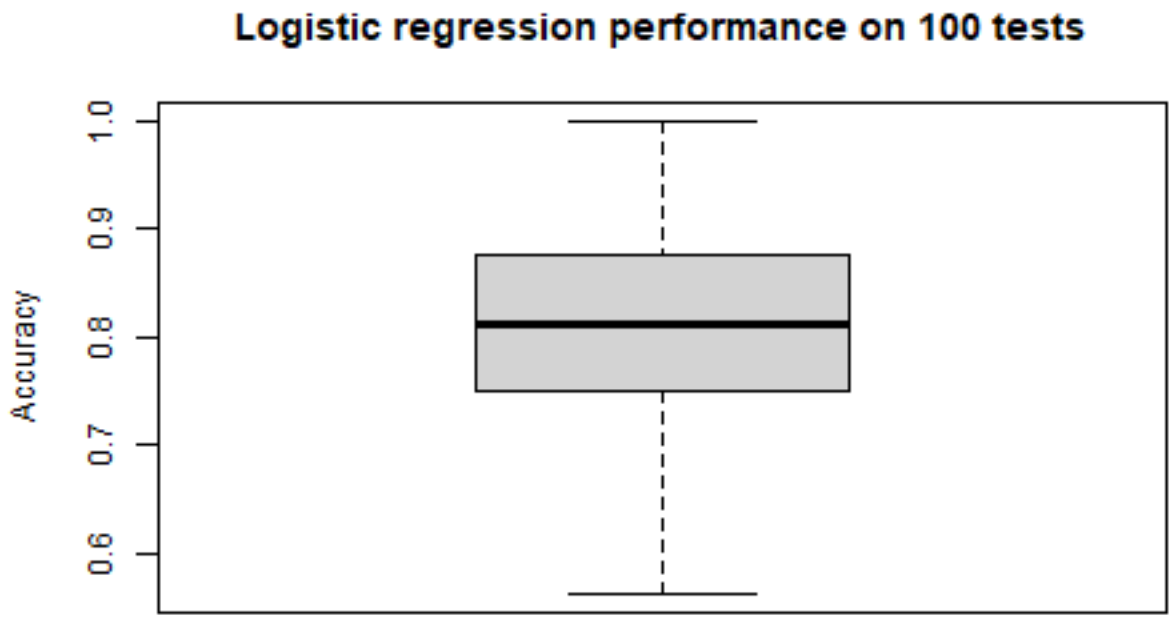
```
hep <- data.factorise(hep, c(1,3:14,20))
hep <- hep[,-3]
```

3.3 Create an imputed dataset for the hepatitis data using the default values.

```
hep.imp <- make.imputations(hep, 1)
```

3.4 Estimate the accuracy of a logistic model using all explanatories (Class ~ .) by running test.logistic(...) 100 times – present the result as a boxplot. Hint: Use replicate(...). Comment on the accuracy versus that stated in the Breiman and Diaconis/Efron paper. Why might your result differ from the published examples?

```
logistic <- replicate(100, test.logistic(hep.imp, Class ~.))
boxplot(logistic, main = "Logistic regression performance on 100 tests", ylab = "Accuracy")
```



After trialing the 100 replications of logistic regression a few times I have found that the average accuracy of the model tends to fall between 80%-90% when using all predictors aside from sex with a 90/10 train test split.

From breiman's method he obtained an average accuracy of 82.6% using this same method except he kept sex as a predictor.

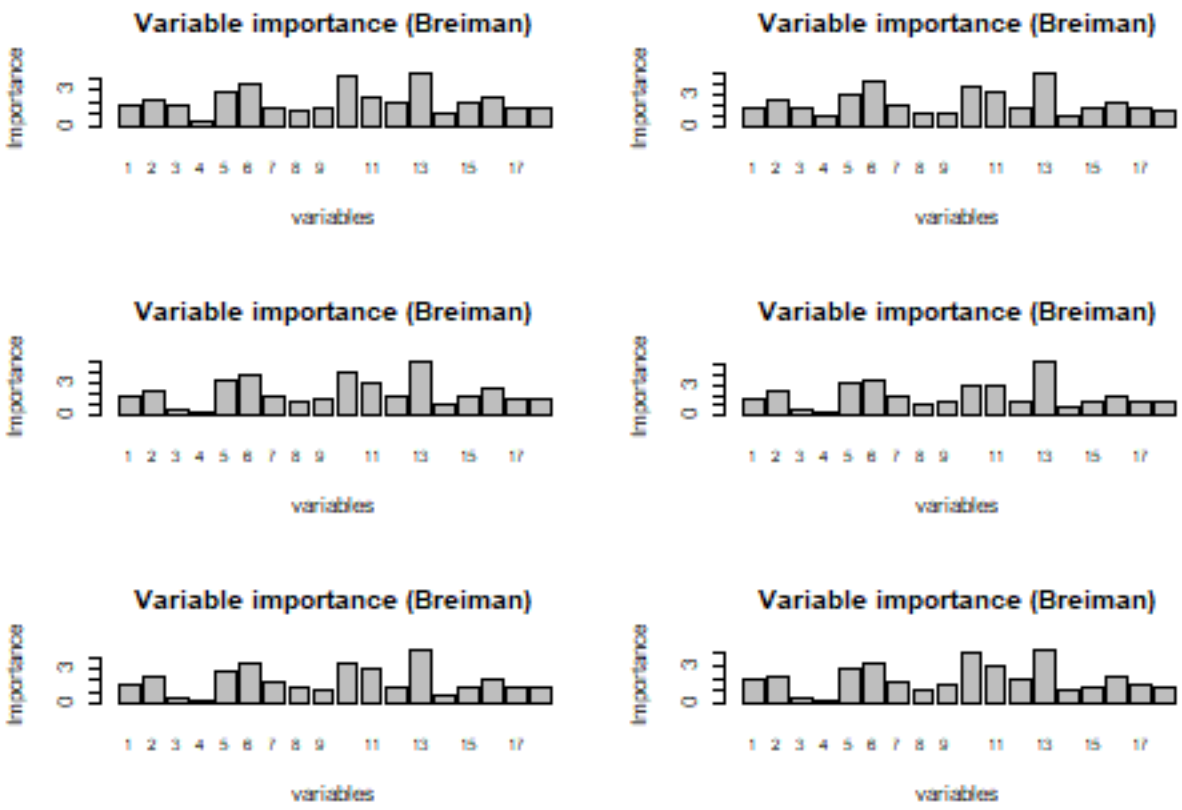
Diaconis and Efron achieved a model with an accuracy of 80%, their method was to perform feature selection by using bootstrapped samples to obtain a prediction for the most important variables which were then used to create a final model with the variables: fatigue, spiders, varices and protime. I believe this may be slightly lower than Breiman's due to less information being modelled but this may also have a lower risk of overfitting. It was also mentioned that in the bootstrap variable selection no variable was selected more than 60% of the time so this could also provide some evidence that none of the variables were important enough to justify the small number of variables used to obtain a final model.

I do not believe there is any significant difference between my results and these as mine fluctuate between 80%-90% between runs. As both methods only report a single result it is unknown if they have the same variation in results, so I believe the main differences in results is likely due to the variation caused by each new train/test split on the data leading to varying coefficients between runs.

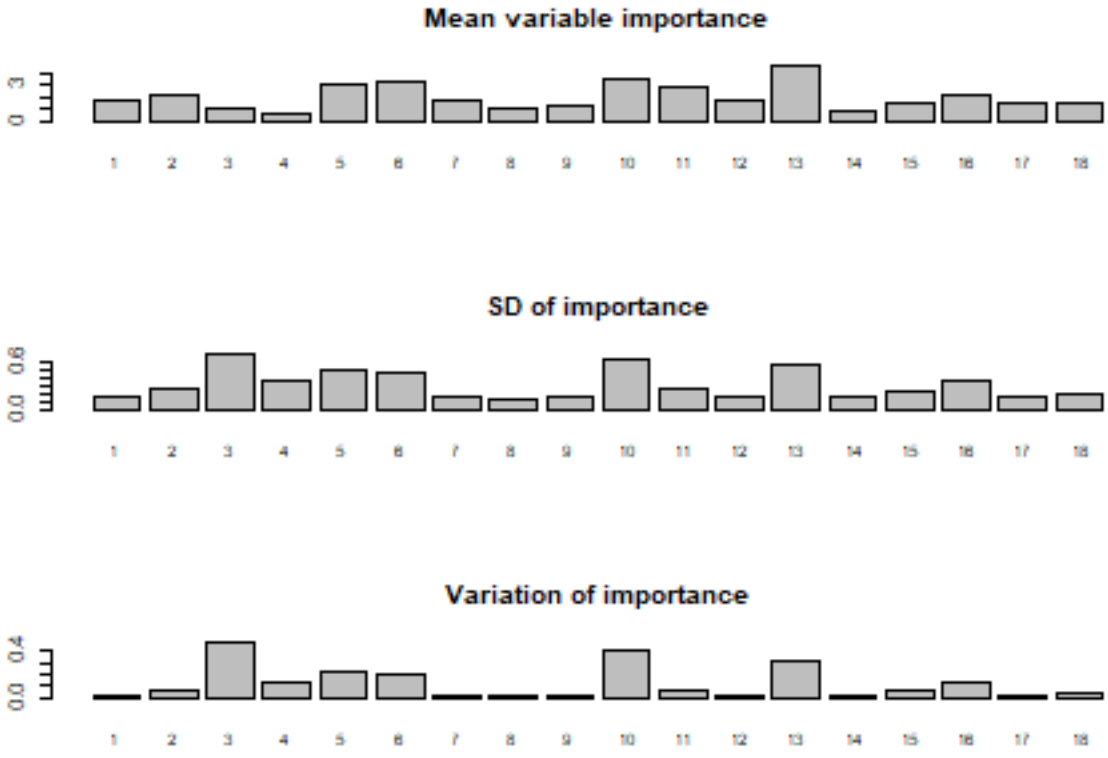
3.5 Breiman suggests that variable importance can be based on examining the absolute value of the coefficients of a variable divided by their standard deviation (over many runs of the model with different training sets). Using glm.coefficients(imp,formula,perc.train=0.9) do 100 runs of this function, produce the variable importance measure stated above, and create a barplot showing variable importance for each explanatory.

How stable is this measure? Rerun the model several times and comment on your observations. What might this suggest about model stability and logistic regression for the hepatitis dataset? Would you be confident in using this approach to assess which variables are important?

```
library(matrixStats)
par(mfrow = c(3,2))
for (var in 1:6) {
  importance <-as.data.frame(abs(replicate(100, glm.coefficients(hep.imp, Class ~.))))
  importance <- importance[-1,]
  importance_sd <- as.data.frame(apply(importance, 1, sd))
  colnames(importance_sd) <-"sd"
  importance <- as.data.frame(rowMeans(importance))
  importance <- as.data.frame(importance$`rowMeans(importance)` / importance_sd$sd)
  colnames(importance) <- "Value"
  barplot(importance$Value, main = "Variable importance (Breiman)",
          xlab = "variables", ylab = "Importance",
          names.arg = rownames(importance), cex.names = 0.7,
          ylim = c(0, max(importance) + 0.2))
}
```



```
overall <- data.frame(ncol = 18)
for (var in 1:50) {
  importance <-as.data.frame(abs(replicate(100, glm.coefficients(hep.imp, Class ~.))))
  importance <- importance[-1,]
  importance_sd <- as.data.frame(apply(importance, 1, sd))
  colnames(importance_sd) <-"sd"
  importance <- as.data.frame(rowMeans(importance))
  importance <- as.data.frame(importance$`rowMeans(importance)` / importance_sd$sd)
  overall <- cbind(overall, importance)
}

overall <- overall[,-1]
mean <- rowMeans(overall)
sd <- rowSds(as.matrix(overall))
vari <- rowVars(as.matrix(overall))
par(mfrow = c(3,1))
barplot(mean, names.arg = rownames(importance), cex.names = 0.7,
        main = "Mean variable importance")
barplot(sd, names.arg = rownames(importance), cex.names = 0.7,
        main = "SD of importance")
barplot(vari, names.arg = rownames(importance), cex.names = 0.7,
        main = "Variation of importance")
```



From running the above code multiple times I have found while Breiman's evaluation of variable importance to be fairly unstable. Across multiple runs I have found the method to have a very high variance for the variables it considers "important". This means that although a variable may appear to be important across

multiple runs there is no evidence to suggest that for a given train test split that variable will be considered important. This also implies that the data within the training split has a large effect on the value of the model coefficients which is typically associated with high variance within the predictors.

This suggests that Breiman's method is heavily dependent on the variables having a low variance which the hepatitis dataset does not appear to meet. So while this method does appear to provide some insight across many runs, the results themselves are very unstable so I would not recommend using this to assess variable importance for a logistic model using the hepatitis dataset.

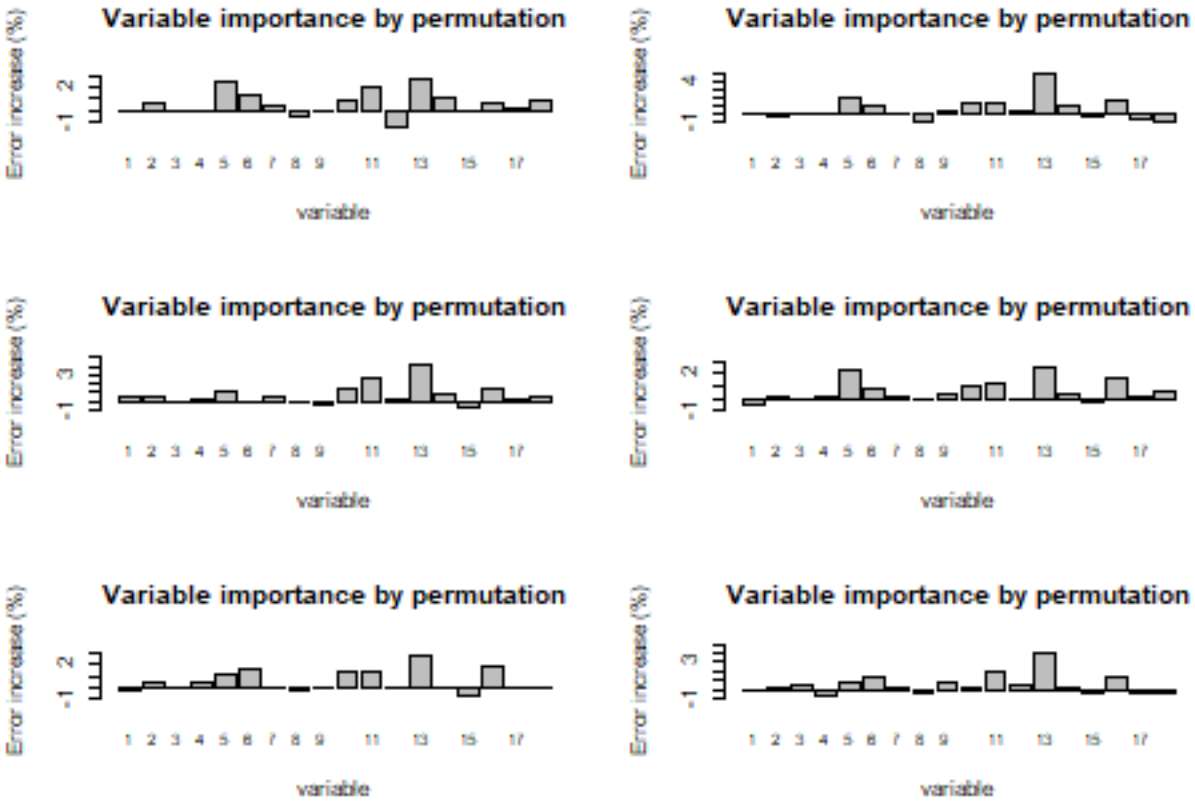
3.6 Implement variable importance using permutation for the logistic model and examine the stability using this approach. HINT: The function definition variable.importance(...) has been provided but is empty. You need to write this function, which uses permutation of one explanatory column to estimate the percentage increase in error when the variable relationship in the data has been removed. There is a function provided (collect.var.imp(..)) that collects up the results for all explanatories.

In your writeup include your function code for variable.importance(...), the resulting visualisation of the model, evidence of stability (or lack of it), and a discussion that addresses:

Is the stability improved over the approach in part 3.5?

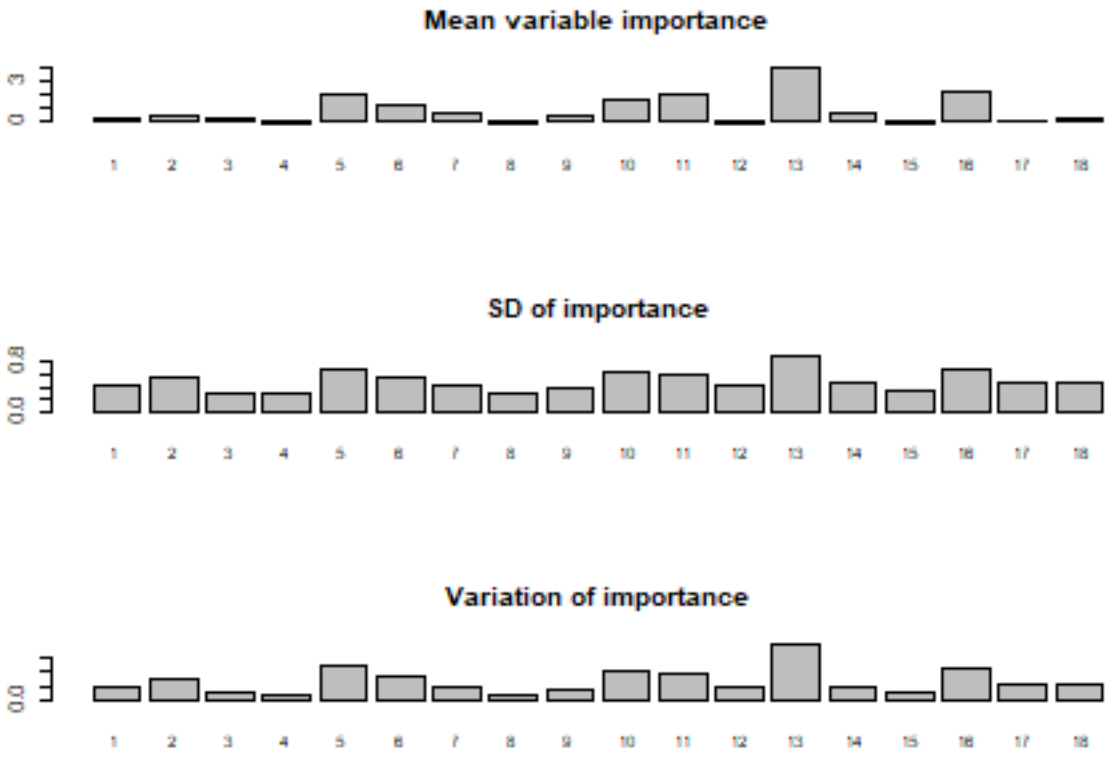
Which variables appear to be important? How do these compare with the provided literature? NOTE: Be aware that you have removed one column so the Breiman paper (which just uses column numbers) needs to be interpreted with care.

```
par(mfrow = c(3,2))
for (var in 1:6) {
   percent <- collect.var.imp(hep.imp, Class ~.)
  percent <- colMeans(percent)
  barplot(percent, names.arg = rownames(importance), cex.names = 0.7,
          ylim = c(-1, max(percent) + 1),
          xlab = "variable", ylab = "Error increase (%)",
          main = "Variable importance by permutation")
}
```




```
overall_new <- data.frame(ncol = 18)
for (var in 1:50) {
```

```
    percent <- collect.var.imp(hep.imp, Class ~.)
  percent <- colMeans(percent)
  overall_new <- cbind(overall_new, percent)
}
overall_new <- overall_new[,-1]
mean <- rowMeans(overall_new)
sd <- rowSds(as.matrix(overall_new))
vari <- rowVars(as.matrix(overall_new))
par(mfrow = c(3,1))
barplot(mean, names.arg = rownames(importance), cex.names = 0.7,
        main = "Mean variable importance")
barplot(sd, names.arg = rownames(importance), cex.names = 0.7,
        main = "SD of importance")
barplot(vari, names.arg = rownames(importance), cex.names = 0.7,
        main = "Variation of importance")
```



Using permutation to find an increase in error appears to provide a much more stable and identifiable measure. Compared to the Breiman's method the variation in the importance measure is lower in relation to the mean value and in the plots there appears to be more separation between important and non important variables. This method of finding more important variables also makes sense as we are testing how important a specific value of a variable when combined with the other predictors so a variable with a higher association to survival will have a significant effect on prediction when permuted.

This method will allow us to build a more stable model as logistic regression performs poorly when there is a large degree of separation between variables so by clearly identifying where this separation exists through permutation we can build a more stable model so I would recommend using this method over Breiman's

variable importance method.

This can be seen in the set of plots for individual runs as the important variables are always important and are clearly separated from the non important variables which tend to have little to no change when permuted. Some variables that are possibly important like Ascites still vary between runs but overall this is much more stable.

From my findings spiders, Bilirubin and Albumin appear to hold a significant importance in a logistic regression model while variables like Malaise and Albumin show signs of potentially being important but still show some noticeable variation in comparison to their importance.

Breiman states that from his method he found Anorexia and spiders to be the two most important variables. Both methods appear to agree on spiders being an important variable, while the permutation method does not disregard anorexia as an important variable but it does have a larger variance between samples in relation to it's overall importance to the logistic model.

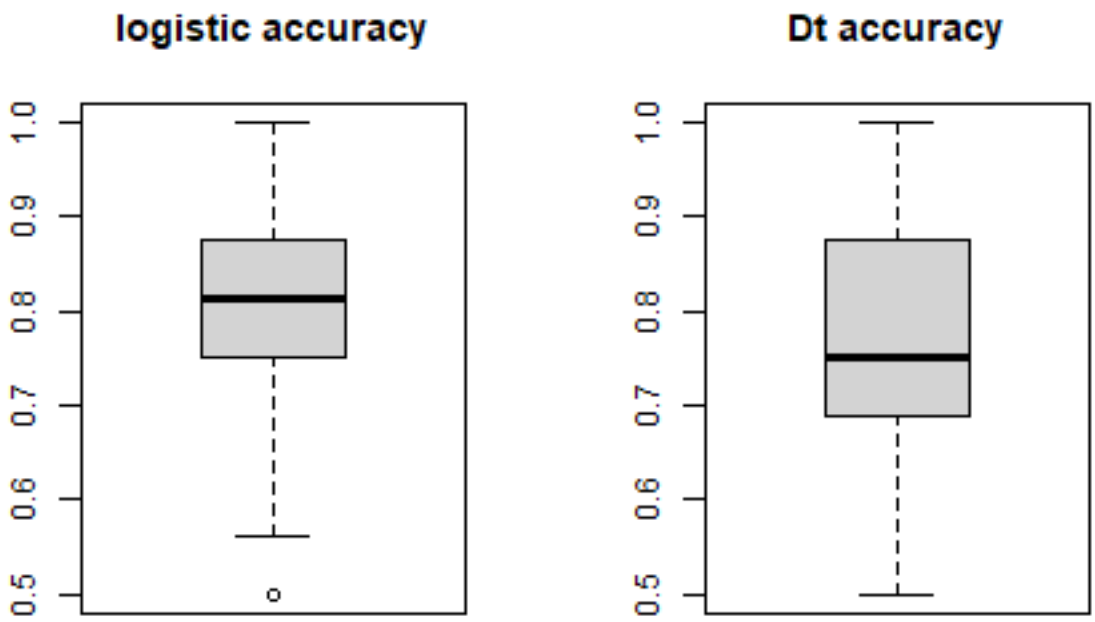
permuation code

```
variable.importance <- function(imp,formula,var.col=2,perc.train=0.9)
{
  #train test data
  dat <- as.matrix(imp)
  ttdata <- get.imp.train.test(dat,perc.train=perc.train)
  train <- ttdata[[1]]
  test <- ttdata[[2]]

  #create standard model
  log.mod <- glm(formula, data=train, family="binomial")
  train.pred <- predict(log.mod,newdata=train,type="response")
  threshold <- fast.threshold(train$Class,train.pred)
  test.pred <- predict(log.mod,newdata=test,type="response")
  pred.vals <- as.factor(ifelse(test.pred >=threshold,2,1))
  conf.mat <- caret::confusionMatrix(data = pred.vals,
                                     reference=test$Class)
  baseline <- (conf.mat$table[1]+conf.mat$table[4])/nrow(test)
  #permutation test
  test[,var.col] <- sample(test[,var.col])
  test.pred <- predict(log.mod,newdata=test,type="response")
  pred.vals <- as.factor(ifelse(test.pred >=threshold,2,1))
  conf.mat <- caret::confusionMatrix(data = pred.vals,
                                     reference=test$Class)
  perm <- (conf.mat$table[1]+conf.mat$table[4])/nrow(test)
  #get difference
  result <- ((baseline - perm) / baseline) * 100
  return(result)
}
```

QUESTION FOUR (20 Marks) – Decision Trees and Random Forests

4.1 Implement a decision tree (using the default parameters) to predict Class using the imputed data (note that you want to predict the response as a "class") and compare this to the logistic model. Include in your writeup the function (code) that does the decision tree and a boxplot showing the resulting error (x100 runs) and compare this to the logistic regression result. You should also compare the 2 distributions and the mean/sd as further evidence of their similarity/difference.

```
num <- sample.int(999, 1)
set.seed(num)

logistic <- replicate(100, test.logistic(hep.imp, Class ~.))
dt <- replicate(100, test.dt(hep.imp, Class ~.))

par(mfrow = c(1,2))
boxplot(logistic, main = "logistic accuracy", ylim = c(0.5,1.0))
boxplot(dt, main = "Dt accuracy", ylim = c(0.5,1.0))
```




```
mean(dt)
```

```
## [1] 0.7725
```

```
mean(logistic)
```

```
## [1] 0.813125
```

```
sd(dt)
```

```
## [1] 0.1078954
```

```
sd(logistic)
```

```
## [1] 0.1146246
```

```
wilcox.test(logistic, dt)
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  logistic and dt
## W = 6232.5, p-value = 0.002212
## alternative hypothesis: true location shift is not equal to 0
```

Comparing the two models after multiple runs I have found that they tend to have a similar accuracy most of the time with the exception of the logistic regression model sometimes having a significant increase in accuracy. I have also noticed that the decision tree model appears to be more stable across a number of runs while the logistic regression model have a higher standard deviation which also varies more than the decision tree between runs. Overall logistic regression has the ability to perform better than the decision tree but this is not consistent as the model is less stable.

```
test.dt <- function(dat, formula, perc.train = 0.9, maxdepth = 30) {
  data <- get.imp.train.test(dat, perc.train=0.9)
  train <- data[[1]]
  test <- data[[2]]
  dt.mod <- build.tree(train, formula, maxdepth=30)
  predictions <- predict.tree(dt.mod, test)
  results <- table(test$Class, predictions)
  return (sum(diag(results))/sum(results))
}
```

4.2 Examine the provided code for a random forest in rf.R– This is a simple implementation: the predict.rf(..) function takes an imputed list of datasets, and builds a random forest and does the prediction and returns the accuracy measure (for direct comparison with previous models).

Give a brief description of how the random forest is created and tested. Run the supplied code and compare the resulting accuracy against the logistic model and single decision tree. Use replicate to run test.rf(...) 100 times with ntrees=1, 5, 10 and 20.

Leave the other parameters at their default value. Does the performance improve? Briefly explain why this model seems superior (does it?) to logistic regression and a single decision tree.

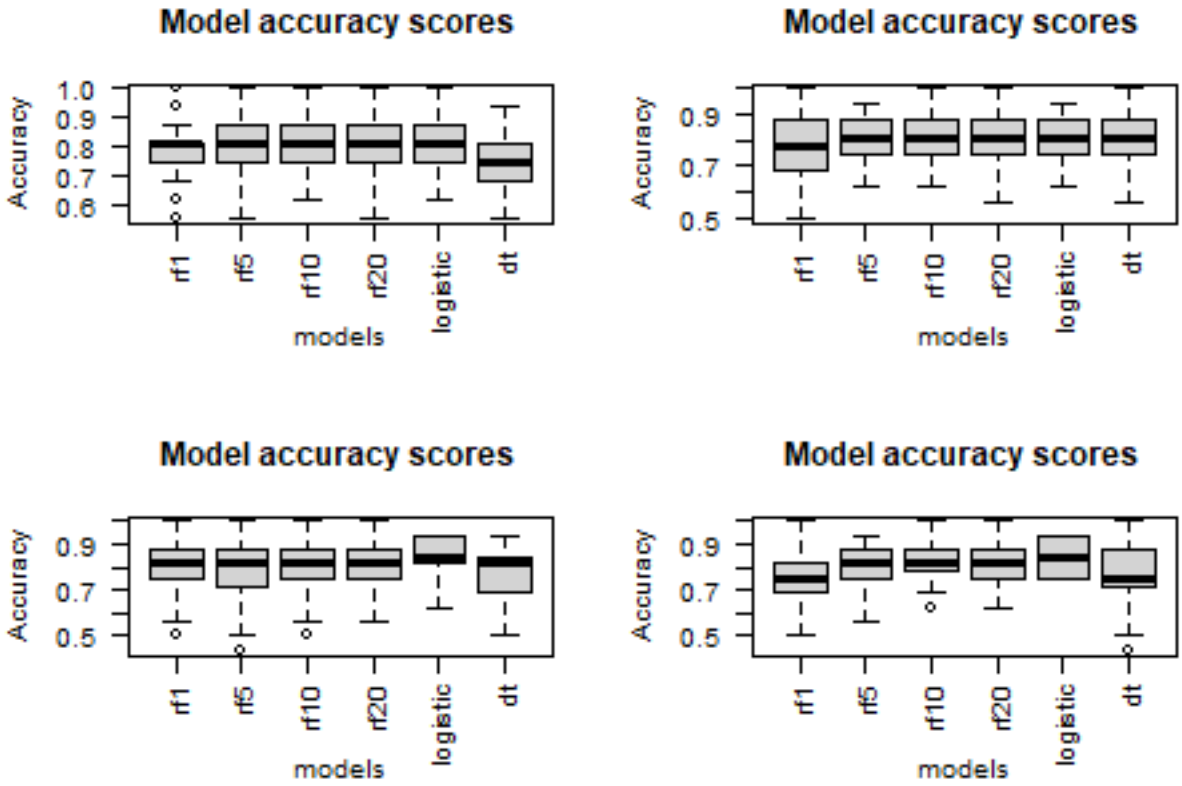
```
par(mfrow = c(2,2))
for (var in 1:4) {
  num <- sample.int(999, 1)
  set.seed(num)
  logistic <- replicate(10, test.logistic(hep.imp, Class ~.))
  dt <- replicate(100, test.dt(hep.imp, Class ~.))
  rf1 <- replicate(100, test.rf(hep.imp, Class ~., ntrees = 1))
  rf5 <- replicate(100, test.rf(hep.imp, Class ~., ntrees = 5))
  rf10 <- replicate(100, test.rf(hep.imp, Class ~., ntrees = 10))
  rf20 <- replicate(100, test.rf(hep.imp, Class ~., ntrees = 20))
  t <- wilcox.test(logistic, rf20)
  cat(t$p.value)
  cat("\n")


  results = data.frame(rf1, rf5, rf10, rf20, logistic, dt)
  boxplot(results, main = "Model accuracy scores", xlab = "models",
          ylab = "Accuracy", names = colnames(results), las = 2)
}
```

```
## 0.7978327
```

```
## 0.5021048
```

```
## 0.5114966
```

```
## 0.3549817
```



A random forest is an ensemble of N decision trees built to make a prediction, as this is a classification problem each tree will make a prediction of either died or lived for each test observation and the most popular option between the trees is selected as the prediction. Each tree bootstraps a new set of training data to ensure that each tree is unique. A subset of K variables is selected for each tree and only these variables are used to build the largest possible tree. No pruning of each tree occurs.

After a few runs of testing all models a number of times I found the models the two best performing models to be logistic regression and a twenty tree random forest. Due to the difference in behavior between the models there are reasons for either to be chosen as a final. When there is a significant difference between the models I found logistic regression to have a higher average accuracy than the random forest more often but for the majority of the time there was no significant difference in model performance and the random forest appears much more stable in these cases, which is expected of an ensemble model as there are much more robust. It is for this reason I would select a random forest model as it is able to obtain more consistent results in prediction which is a worthwhile trade-off.

The random forest model also appears to be a stronger model as in some cases it has a performance increases whereas the only change in the decision tree that is noticeable is the amount of variance across a number of runs.

QUESTION FIVE (15 MARKS) – Write approx. 300 words

Assume you are given a dataset from a loans company that describes the following characteristics of customers including whether they were found to be a good or bad credit risk (the response):

Age of customer; Experience: professional experience in years; Income of customer; Average monthly credit card spending; Mortgage: size of mortgage; Credit Card: No/Yes; Educational level: three categories (undergraduate, graduate, professional); Credit Risk: Good/Bad

You are ultimately interested in building a model that predicts Credit Risk given the other independent (explanatory) variables.

Explain the steps that you would follow to understand the patterns in the dataset, PRIOR to building a model for prediction. Ensure in your description that you include comments on what information each method or visualisation gives, and why this may be important prior to modelling the data.

When performing analysis on a dataset prior to modelling the first thing I would do is to ensure the data is tidy aside from NA values existing. This is required as each variable can only take on one value per observation. Next I would visualise all of the variables to check for any possible non-linear variables that could be transformed. This makes the variables easier to model and will improve their correlation to other variables. Once all the variables are transformed (if needed), I would then deal with any NA values in the dataset most likely through imputation. I would do this after transforming variables as the new imputed values are likely to model better this way.

Once I am happy with the structure of the data I would make a correlation plot of all the variables to test their relationships to each other. This allows to check which variables have strong relationships to credit risk and which predictors may suffer from colinearity issues or potential confounding. Colinearity issues may not possibly be a concern depending on the model used but less stable methods such as regression will suffer from this. If any predictors do appear to have confounding issues I would also visualise these to try and confirm this. For example from the predictors in the credit data without looking at anything I would already be suspicious of Education level being a confounding effect for income and possibly experience.

I would also visualise theories I have on how different variables related to credit risk such as does someones level of education appear to have a significant association with their credit risk. This would allow for a basic prediction to be made on variable importance without running some extensive bootstrapping tests on the data. The last thing I would test is are all levels of each factor relevant to the data, for instance if someone does not already own a credit card does this provide any significant information to how their credit risk might be affected. In this case I would also test if it appears to provide any significant evidence having accounted for monthly credit card spending as customers who do not own a credit card should have a monthly average credit card spending of zero.

Overall through these methods we should get reasonable insight as to the quality of the data and how meaningful the predictors are in association to credit risk.