Jason Rapsinski
CS 373 HW2
10/26/2020

1.

$$Entropy(D) = \frac{-3}{6} * \log_2\left(\frac{3}{6}\right) + \frac{-3}{6} * \log_2\left(\frac{3}{6}\right) = \frac{-1}{2} * -1 + \frac{-1}{2} * -1 = 1$$

i. Assign missing value with the most common one

$$Entropy(isAlone = 0) = \frac{-1}{2} * \log_2\left(\frac{1}{2}\right) + \frac{-1}{2} * \log_2\left(\frac{1}{2}\right) = \frac{-1}{2} * -1 + \frac{-1}{2} * -1 = 1$$

$$Entropy(isAlone = 1) = \frac{-2}{4} * \log_2\left(\frac{2}{4}\right) + \frac{-2}{4} * \log_2\left(\frac{2}{4}\right) = \frac{-1}{2} * -1 + \frac{-1}{2} * -1 = 1$$

$$Information\ Gain(isAlone) = 1 - \left(\frac{2}{6} * 1 + \frac{4}{6} * 1\right) = 0$$

ii. Assign missing value with the most common one

$$Entropy(isAlone = 0) = \frac{-1}{2} * \log_2\left(\frac{1}{2}\right) + \frac{-1}{2} * \log_2\left(\frac{1}{2}\right) = \frac{-1}{2} * -1 + \frac{-1}{2} * -1 = 1$$

$$Entropy(isAlone = 1) = \frac{-2}{4} * \log_2\left(\frac{2}{4}\right) + \frac{-2}{4} * \log_2\left(\frac{2}{4}\right) = \frac{-1}{2} * -1 + \frac{-1}{2} * -1 = 1$$

$$Information\ Gain(isAlone) = 1 - \left(\frac{2}{6} * 1 + \frac{4}{6} * 1\right) = 0$$

iii. Discard missing values

$$Entropy(isAlone = 0) = \frac{-1}{2} * \log_2\left(\frac{1}{2}\right) + \frac{-1}{2} * \log_2\left(\frac{1}{2}\right) = \frac{-1}{2} * -1 + \frac{-1}{2} * -1 = 1$$

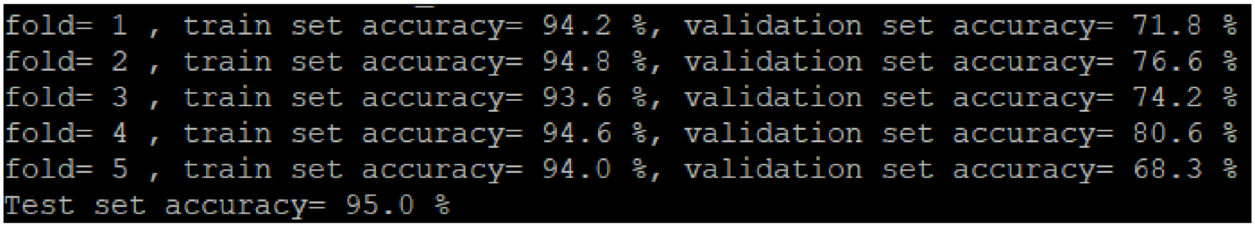
$$Entropy(isAlone = 1) = \frac{-2}{3} * \log_2\left(\frac{2}{3}\right) + \frac{-1}{3} * \log_2\left(\frac{1}{3}\right) = 0.389975 + 0.528321 = 0.918296$$

$$Information\ Gain(isAlone) = 1 - \left(\frac{2}{5} * 1 + \frac{3}{5} * 0.918296\right) = 0.0490224$$

2. If there are no limitations to the decision tree building process, the model it produces is just a memorization of the training data. Trying to prune the tree using the training data would not make any changes, because there are no differences that could be made to the tree to increase accuracy on the training data. Thus, introducing a new data set to validate the tree can reduce the overfitting problems of decision trees.

3.

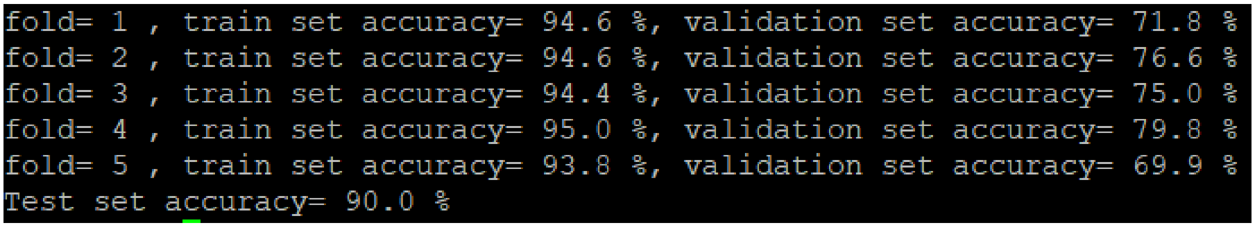
i. Results for assigning missing values the mode or most common value



```
fold= 1 , train set accuracy= 94.2 %, validation set accuracy= 71.8 %
fold= 2 , train set accuracy= 94.8 %, validation set accuracy= 76.6 %
fold= 3 , train set accuracy= 93.6 %, validation set accuracy= 74.2 %
fold= 4 , train set accuracy= 94.6 %, validation set accuracy= 80.6 %
fold= 5 , train set accuracy= 94.0 %, validation set accuracy= 68.3 %
Test set accuracy= 95.0 %
```

Average Training Set Accuracy: 94.24% | Average Validation Set Accuracy: 74.3%

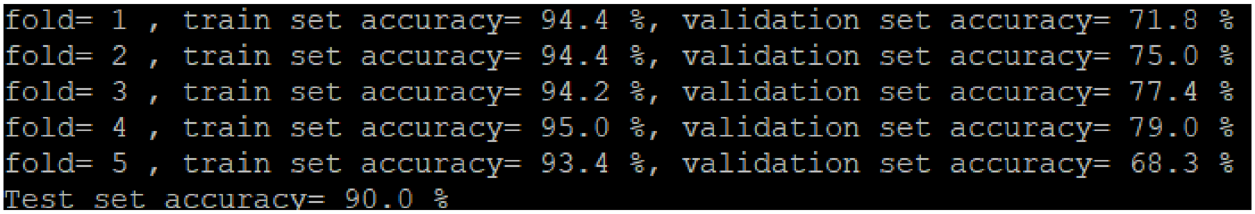
ii. Results for assigning missing values the median value



```
fold= 1 , train set accuracy= 94.6 %, validation set accuracy= 71.8 %
fold= 2 , train set accuracy= 94.6 %, validation set accuracy= 76.6 %
fold= 3 , train set accuracy= 94.4 %, validation set accuracy= 75.0 %
fold= 4 , train set accuracy= 95.0 %, validation set accuracy= 79.8 %
fold= 5 , train set accuracy= 93.8 %, validation set accuracy= 69.9 %
Test set accuracy= 90.0 %
```

Average Training Set Accuracy: 94.48% | Average Validation Set Accuracy: 74.62%

iii. Results for discarding rows with missing values

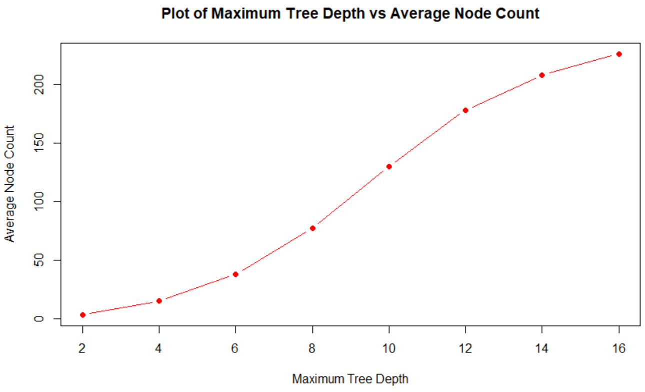


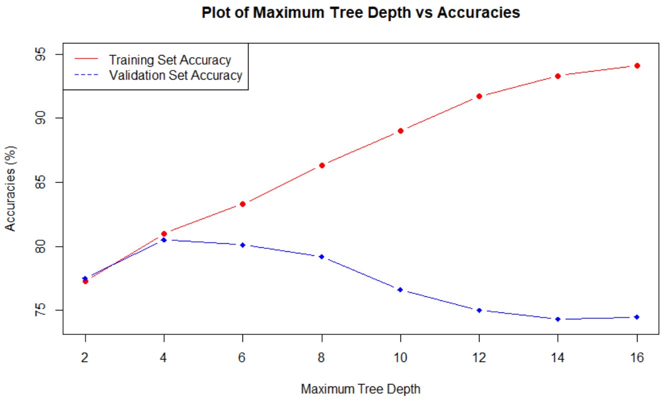
```
fold= 1 , train set accuracy= 94.4 %, validation set accuracy= 71.8 %
fold= 2 , train set accuracy= 94.4 %, validation set accuracy= 75.0 %
fold= 3 , train set accuracy= 94.2 %, validation set accuracy= 77.4 %
fold= 4 , train set accuracy= 95.0 %, validation set accuracy= 79.0 %
fold= 5 , train set accuracy= 93.4 %, validation set accuracy= 68.3 %
Test set accuracy= 90.0 %
```

Average Training Set Accuracy: 94.28% | Average Validation Set Accuracy: 74.3%

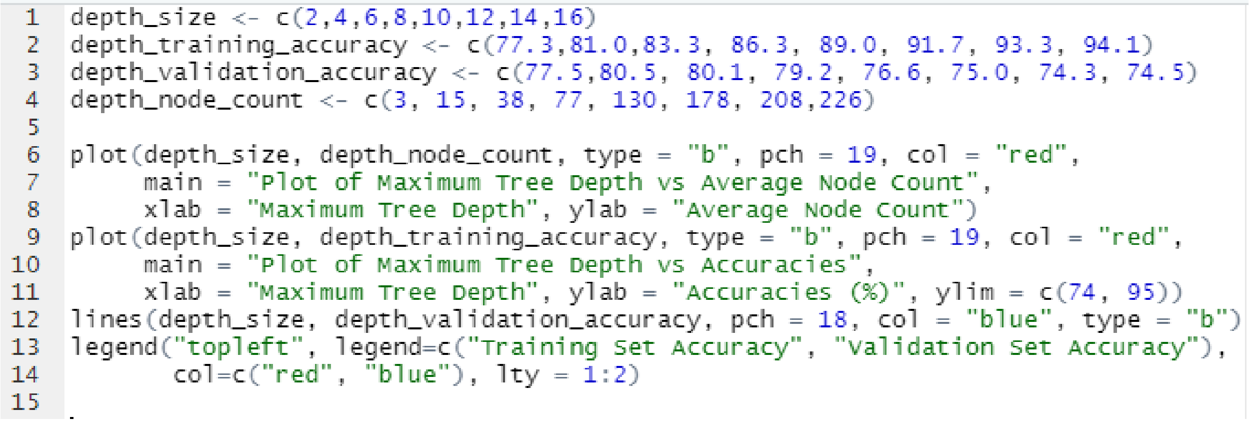
In general, the different handling methods for missing values did not have a considerable impact on the training and validation set accuracies. This is because rows that contain a missing value are infrequent and as such are not numerous enough to make changes to the functioning of the decision tree.

4. The maximum tree depth with the best cross validation accuracy was 4 for this specific situation. I performed my programming over PuTTy, so I was unable to use plotting libraries in python. To produce the graphs seen below, I generated node counts, training set accuracies, and validation set accuracies, recorded them, and used Rstudio to generate the graphs



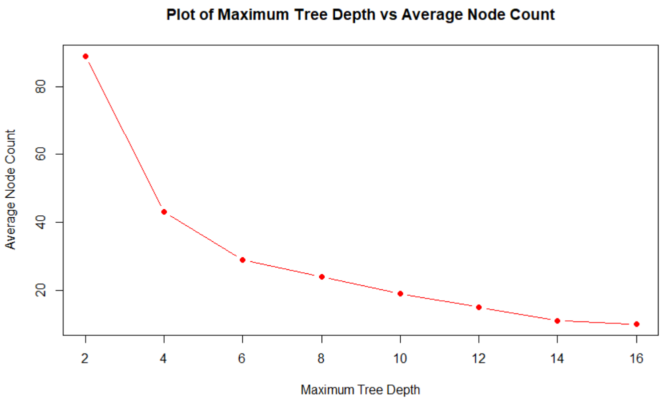


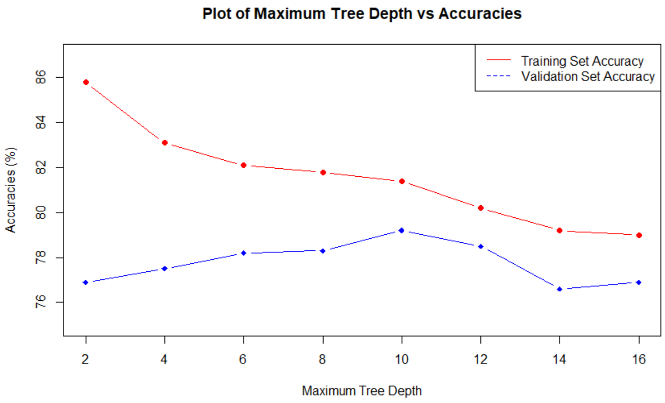
RStudio Code



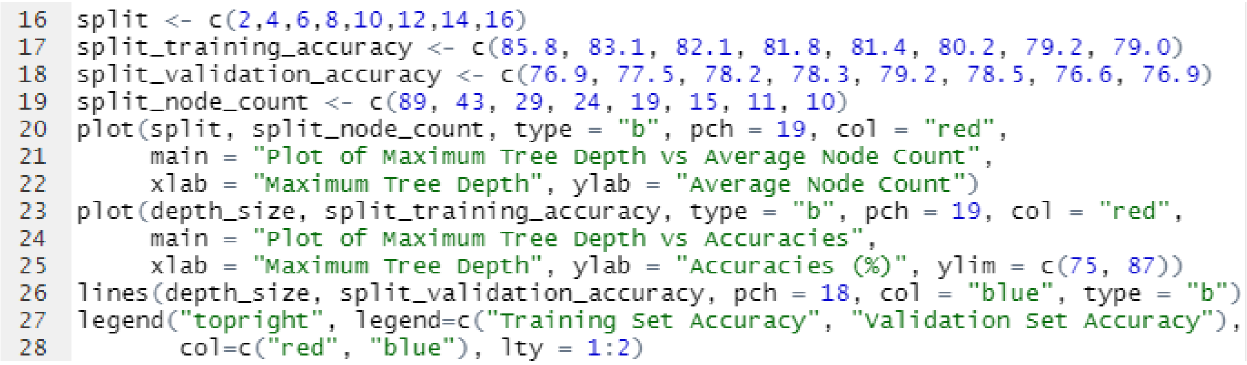
```
depth_size <- c(2,4,6,8,10,12,14,16)
depth_training_accuracy <- c(77.3,81.0,83.3, 86.3, 89.0, 91.7, 93.3, 94.1)
depth_validation_accuracy <- c(77.5,80.5, 80.1, 79.2, 76.6, 75.0, 74.3, 74.5)
depth_node_count <- c(3, 15, 38, 77, 130, 178, 208,226)

plot(depth_size, depth_node_count, type = "b", pch = 19, col = "red",
     main = "Plot of Maximum Tree Depth vs Average Node Count",
     xlab = "Maximum Tree Depth", ylab = "Average Node Count")
plot(depth_size, depth_training_accuracy, type = "b", pch = 19, col = "red",
     main = "Plot of Maximum Tree Depth vs Accuracies",
     xlab = "Maximum Tree Depth", ylab = "Accuracies (%)", ylim = c(74, 95))
lines(depth_size, depth_validation_accuracy, pch = 18, col = "blue", type = "b")
legend("topleft", legend=c("Training Set Accuracy", "Validation Set Accuracy"),
         col=c("red", "blue"), lty = 1:2)

```

5. The split size which resulted in the highest validation set accuracy was 10 for this specific situation. I performed my programming over PuTTy, so I was unable to use plotting libraries in python. To produce the graphs seen below, I generated node counts, training set accuracies, and validation set accuracies, recorded them, and used Rstudio to generate the graphs.


Plot of Maximum Tree Depth vs Average Node Count


Plot of Maximum Tree Depth vs Accuracies

RStudio Code



```r
split <- c(2,4,6,8,10,12,14,16)
split_training_accuracy <- c(85.8, 83.1, 82.1, 81.8, 81.4, 80.2, 79.2, 79.0)
split_validation_accuracy <- c(76.9, 77.5, 78.2, 78.3, 79.2, 78.5, 76.6, 76.9)
split_node_count <- c(89, 43, 29, 24, 19, 15, 11, 10)
plot(split, split_node_count, type = "b", pch = 19, col = "red",
     main = "Plot of Maximum Tree Depth vs Average Node Count",
     xlab = "Maximum Tree Depth", ylab = "Average Node Count")
plot(depth_size, split_training_accuracy, type = "b", pch = 19, col = "red",
     main = "Plot of Maximum Tree Depth vs Accuracies",
     xlab = "Maximum Tree Depth", ylab = "Accuracies (%)", ylim = c(75, 87))
lines(depth_size, split_validation_accuracy, pch = 18, col = "blue", type = "b")
legend("topright", legend=c("Training Set Accuracy", "Validation Set Accuracy"),
       col=c("red", "blue"), lty = 1:2)
```