CS 486 AI Assignment 5
Ziyi Zhao
20477924

1. Cross Validation
   Leave-one-out cross-validation predicts each instance while training on all n-1 other instances. This means that the testing set would be either {1 positive, 0 negative} or {0 positive, 1 negative} while the training set would be {99 positive, 100 negative}, {100 positive, 99 negative}. Using majority classifier would always produce different label for training and testing data. For example, if the testing data instance is positive, the training data would be negative. As a result, it would be always scoring zero.

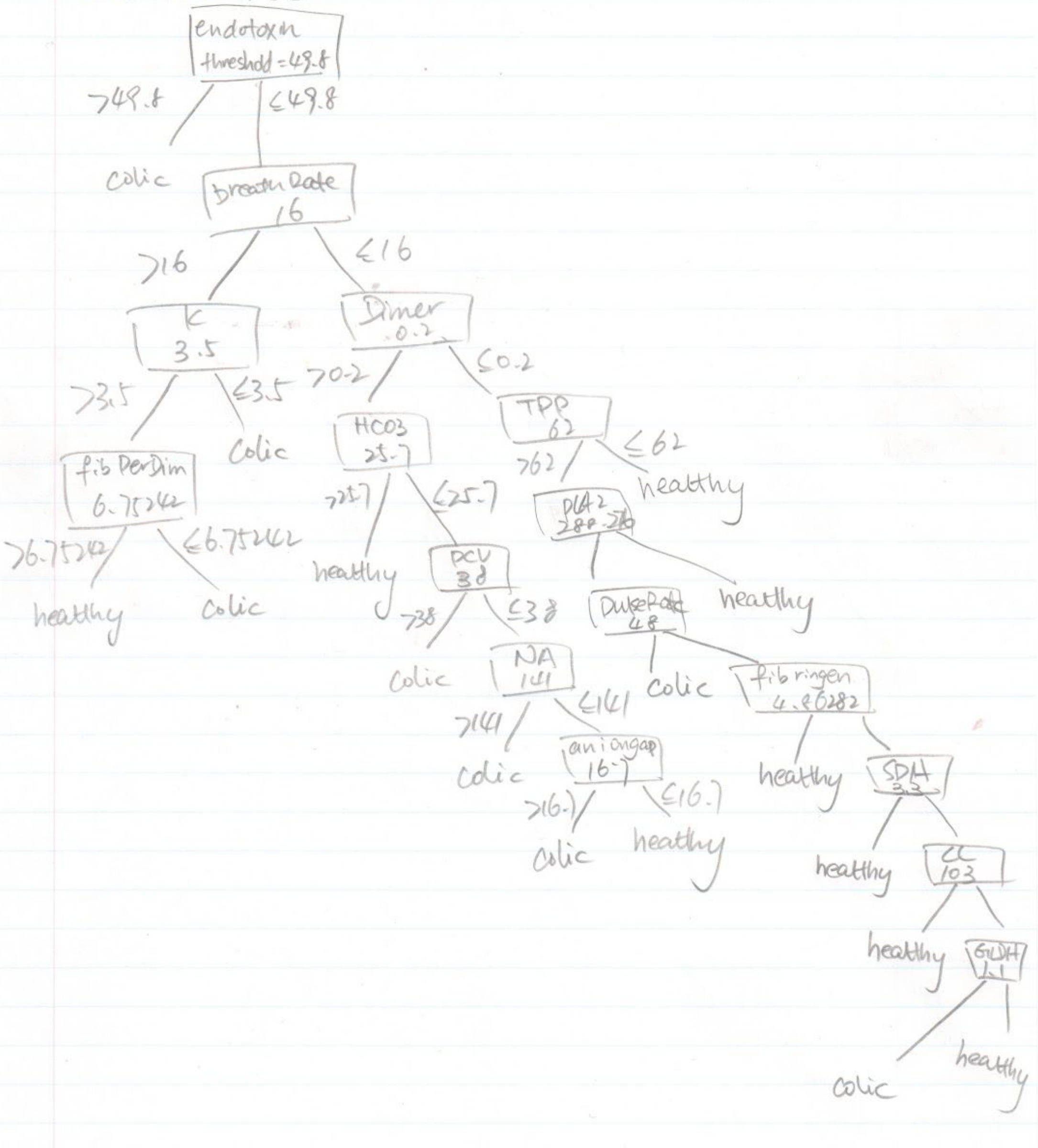
2.

q2) Output of the program:

```
entropy: 0.9182958340544896
split attribute: endotoxin: with thres: 49.8: info gain: 0.44730295866571396
split attribute: breathRate: with thres: 16.0: info gain: 0.3497424768976448
split attribute: k: with thres: 3.5: info gain: 0.5734659864425016
split attribute: fibPerDim: with thres: 6.75242: info gain: 0.9182958340544896
split attribute: dimer: with thres: 0.2: info gain: 0.4728836025705746
split attribute: HCO3: with thres: 25.7: info gain: 0.3673183336217959
split attribute: PCV: with thres: 38.0: info gain: 0.2516291673878229
split attribute: Na: with thres: 141.0: info gain: 0.2295739585136224
split attribute: aniongap: with thres: 16.7: info gain: 0.9182958340544896
split attribute: TPP: with thres: 62.0: info gain: 0.6782009862942735
split attribute: PLA2: with thres: 288.276: info gain: 0.594191422035258
split attribute: pulseRate: with thres: 48.0: info gain: 0.31668908831502096
split attribute: fibrinogen: with thres: 4.86282: info gain: 0.3673183336217959
split attribute: SDH: with thres: 3.3: info gain: 0.2516291673878229
split attribute: Cl: with thres: 103.0: info gain: 0.0
split attribute: GLDH: with thres: 1.1: info gain: 0.0
```

q3)

Q2 Decision Tree



q4) the tree classify training instance correctly with 100% by feeding the training set again into the tree and produce output result. The output result is then compared with the original training set.

Program output:
```
healthy
healthy
healthy
healthy
healthy
healthy
colic
colic
colic
colic
colic
colic
colic
colic
colic
colic
healthy
healthy
healthy
healthy
healthy
healthy
colic
colic
colic
colic
colic
colic
colic
colic
colic
colic
colic
healthy
healthy
healthy
healthy
```

healthy
colic
colic
colic
colic
colic
colic
colic
colic
colic
colic
healthy
healthy
healthy
healthy
healthy
colic
colic
colic
colic
colic
colic
colic
colic
colic
colic
healthy
healthy
healthy
healthy
healthy
colic
colic
colic
colic
colic
colic
colic
colic
colic
colic
healthy
healthy

healthy
healthy
healthy
healthy
colic
colic
colic
colic
colic
colic
colic
colic
healthy
healthy
healthy
healthy
healthy
colic
colic
colic
colic
colic
colic
colic
colic
colic
healthy
healthy
healthy
healthy
colic
colic
colic
colic
colic
colic
colic
colic
colic
healthy
healthy
healthy
colic

```
colic
colic
colic
colic
colic
colic
colic
colic
colic
```

q5) the test instance was feed into the tree after the training set. It also results in 100% correctness by comparing the output similar to q4 above.

Program output:

```
healthy
healthy
healthy
healthy
healthy
colic
colic
colic
colic
colic
colic
colic
colic
```

q6)

The program first calculates the entropy at the very beginning. Then, for each node in the tree, including the root node, i maintained a working list of NodeInfo. With this working list, i calculate the threshold for each attribute. Then, i computed the information gain using the threshold for each attribute and selected the attribute with maximum attribute to split. The information gain is calculated using the formula IG = entropy - remainder (A), where the raminder(A) = -Sum{j->n} P(X=xj) * Sum{i->k} P(Y=yi | X = xj) log2(P(Y=yi | X=xj)). For details, please see the code. After the attribute with maximum info gain is achieved, I splitted the working set into left and right node, where the left node is > threshold and right node is <= threshold. The program then recurses until it finishes building the tree.

3. Code is identical to question 2.

q3) the training instance is correct 100% of the time where i used similar testing mechanism in the above question q4) test instance is correct in about 54% of the time (79/146) using the algorithm i wrote for previous question.

q5) There is a difference between fraction of correct instance between the training and the testing set. This is caused by overfitting. This question does not do any pruning and testing, therefore the tree is exposed to errors. Overfitting defines the finding of patterns in the data where there is no actual pattern, thus it decreased the accuracy of decision tree by more than what is presented on the slide(slide: 10-25%, actual 46%).