COMP 307 Assignment 1

Yongbo Yu ID: 300390526

Part 1 (KNN) :

Q1:

With using k=1 the predicted label is shown below:

```
setosa Correct        setosa Correct        setosa Correct        setosa Correct
setosa Correct        setosa Correct        setosa Correct
setosa Correct        setosa Correct        setosa Correct
setosa Correct        setosa Correct        setosa Correct
setosa Correct        setosa Correct        setosa Correct
setosa Correct        setosa Correct        setosa Correct
versicolor    Correct      versicolor    Correct      virginica     InCorrect
versicolor    Correct      versicolor    Correct      versicolor    Correct
virginica     InCorrect    versicolor    Correct      versicolor    Correct
versicolor    Correct      versicolor    Correct      versicolor    Correct
versicolor    Correct      versicolor    Correct      versicolor    Correct
versicolor    Correct      versicolor    Correct      versicolor    Correct
versicolor    Correct      virginica     Correct      virginica     Correct
virginica     Correct      virginica     Correct      virginica     Correct
virginica     Correct      versicolor    InCorrect    versicolor    InCorrect
virginica     Correct      virginica     Correct      versicolor    InCorrect
virginica     Correct      virginica     Correct      virginica     Correct
virginica     Correct      virginica     Correct      virginica     Correct
virginica     Correct      versicolor    InCorrect
```

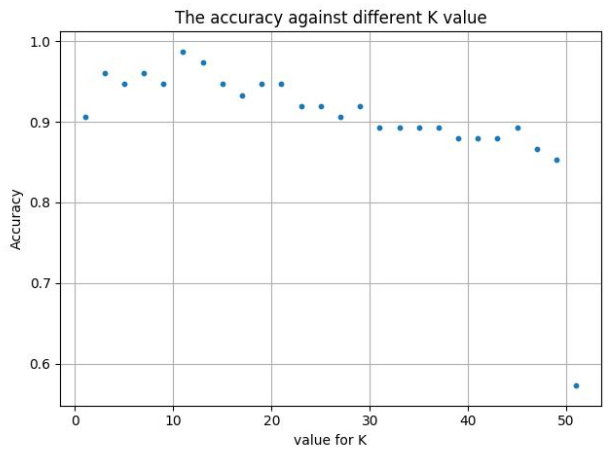
The accuracy for test set is: 0.9066666666666666

Question 2:

For K=3, the test accuracy increased to 96% which is higher than the accuracy for k=1 (0.9067). Which means by select K=3 can improve the performance of KNN on this certain dataset. Therefore for this certain dataset we can say, KNN can perform better than just Nearest Neighbour.

The reason of the performance of k=3 is better is it not only compare with the one nearest neighbour but comparing to the top 3 nearest neighbour, and choose the most frequency class among this three neighbours. By choosing the most frequency, the probability of wrong classification is much lower than just pick one nearest neighbour and assign the label of the neighbour to the instance.

The figure is shown below is the accuracy against different value of K:



Question 3:

Pros:

1. KNN is one simple algorithm, it is easy to implement, and it can easily to apply to lots different cases. More specifically, KNN can have a great performance on a low dimensional data.

2. KNN is a lazy learning technique, it has no learning process, which can leads to a low computation cost.

3.Robust to noisy training data (especially using a weighted distance)

Cons:

1. Need to determine value of parameter K.
2. It can not distinguish the importance of each feature. It only use all the features to compute the euclidean distances.
3. It may requires a high computing cost when dealing with a large dataset which contains high dimensionality. Because compute the euclidean distance need to sum up the distance for each feature between each instance.

Question 4:

Step 1: Separate all the instances(150 instances) in the dataset into 5 subsets.(30 instances each)

Step 2: Use one of the subsets as test set and treat the rest as training sets. And apply the test data to training data with KNN method. And it will provide the accuracy of this classifier.

Step 3: For each subset, we treat it as test set once, and using the rest subsets as training and repeat step2 4 times. After all the subsets have been used as test set, we can finally have five accuracy, and take the mean value of this accuracy as the accuracy for KNN on this dataset.

Question 5:

I would use K-means cluster method where K =3. (3-means cluster)

Step 1:

Set K =3, and initialise three clusters randomly.

Step 2:

Calculate the distance between the instance to each cluster, and associate the nearest mean to the instance. Repeat the this procedure for each instance in the dataset.

Step 3:

Replace the old means with the centroid of each of the 3 clusters as the new means

Step 4: Repeat the Step2 and Step3 until no change in each cluster center.

Question 6:

3-means clusters implemented.

Part 2:

Question 1:

The trained tree is :

```
ASCITES = True

   SPIDERS = True

      VARICES = True

         FIRMLIVER = True

            Class live, prob = 1, instance_left = 12

         FIRMLIVER = False

            BIGLIVER = True

               STEROID = True

                  Class live, prob = 1, instance_left = 2

               STEROID = False

                  SGOT = True

                     Class live, prob = 1, instance_left = 1

                  SGOT = False

                     HISTOLOGY = True
                  Class die, prob = 1, instance_left = 1
                HISTOLOGY = False
                  MALAISE = True
                    Class live, prob = 1, instance_left = 1
                  MALAISE = False
                    Class die, prob = 1, instance_left = 1
          BIGLIVER = False
            Class live, prob = 1, instance_left = 3
      VARICES = False
        Class die, prob = 1, instance_left = 1
    SPIDERS = False
      FIRMLIVER = True
        SGOT = True
          Class live, prob = 1, instance_left = 1
        SGOT = False
          FEMALE = True
            Class live, prob = 1, instance_left = 1
          FEMALE = False
            HISTOLOGY = True
              Class die, prob = 1, instance_left = 3
            HISTOLOGY = False
              Class die, prob = 1, instance_left = 1
      FIRMLIVER = False
```

```
            SGOT = True

                BIGLIVER = True

                    SPLEENPALPABLE = True

                        Class live, prob = 1, instance_left = 2

                    SPLEENPALPABLE = False

                        Class die, prob = 1, instance_left = 2

                BIGLIVER = False

                    Class die, prob = 1, instance_left = 1

            SGOT = False

                Class live, prob = 1, instance_left = 6

ASCITES = False

    BIGLIVER = True

        STEROID = True

            Class die, prob = 1, instance_left = 6

        STEROID = False

            Class die, prob = 1, instance_left = 2

    BIGLIVER = False

        Class live, prob = 1, instance_left = 1
```

The tree construct by the Decision Tree is shown above

The accuracy of the baseline is 85.185%. In this dataset, the baseline model always predict live for the test instances. Because in the training set the live class is the dominate class.
The accuracy of the Decision Tree is 81.48%
The accuracy of the baseline is slightly higher than Decision Tree. This is because the training set is highly imbalanced and the test set is also a highly imbalanced set, and they have the same bias (Live), therefore baseline model have a slightly better performance on this dataset. But if we change the test set to a balanced dataset or a most instances are ‘die’

class, the performance of the baseline model will be very poor. The baseline model is way too overfitting to this certain dataset.

Question 2:

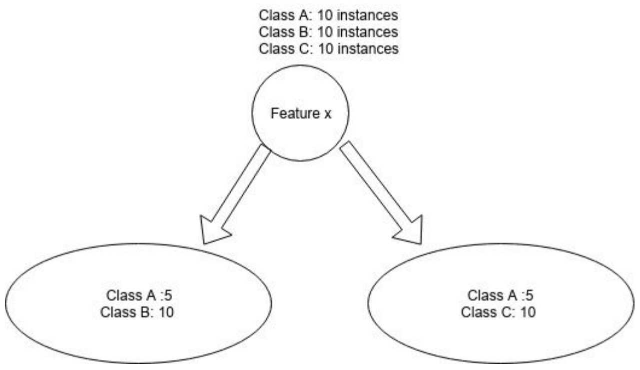
```
The index of the dataset pair we are using is : 1
Read: 37 instances
Baseline class is: live
Accuracy: 64.86%
The index of the dataset pair we are using is : 2
Read: 37 instances
Baseline class is: live
Accuracy: 86.49%
The index of the dataset pair we are using is : 3
Read: 37 instances
Baseline class is: live
Accuracy: 86.49%
The index of the dataset pair we are using is : 4
Read: 37 instances
Baseline class is: live
Accuracy: 72.97%
The index of the dataset pair we are using is : 5
Read: 37 instances
Baseline class is: live
Accuracy: 78.38%
The index of the dataset pair we are using is : 6
Read: 37 instances
Baseline class is: live
Accuracy: 67.57%
The index of the dataset pair we are using is : 7
Read: 37 instances
Baseline class is: live
Accuracy: 81.08%
The index of the dataset pair we are using is : 8
Read: 37 instances
Baseline class is: live
Accuracy: 70.27%
The index of the dataset pair we are using is : 9
Read: 37 instances
Baseline class is: live
Accuracy: 75.68%
The index of the dataset pair we are using is : 10
Read: 37 instances
Baseline class is: live
Accuracy: 81.08%
The average of these 10 training-test pairs is :  76.48648648648648
```

Question 3:
(a):
Pruning is the inverse of splitting.
There are two common kind of pruning techniques, one is top down and the other one is bottom up. The top down pruning will traverse nodes and trim subtrees starting at the root. The bottom up will start at the leaf nodes.
In order to reduce the complexity of the decision tree or reduce the computation cost, we can prune any pair whose elimination yields a satisfactory or only a small increase in impurity is eliminated. And also the change the common parent node to a leaf node.

b:
By doing pruning we reduce the complexity of the Decision tree. It could also reduce the fitness of the DT for the training data. Without pruning, DT can be very complex but it can dig out all the details to classify the training data. Therefore without pruning the training accuracy will be higher than apply pruning.

c :
Without pruning can have a higher training rate, but it may be overfitting, which leads to a poor performance on the test data. Pruning is a useful technique to reduce overfitting hence it can increase the accuracy on the test set.

Question 4 :
As we discussed in the lecture, the formula for calculating the impurity is not very suitable for multi-classes(more than 2 classes.)



Using the formula to calculate the impurity :
Impurity = 0.5(weight) *((5/15) (probability for class A in the left node) * (10/15) (probability for class B in the right node) * 0/15(probability for class C in left node)) + 0.5 * ((5/15) + (0/15))
From the calculation we can see the impurity is zero now, but from the graph we can clearly see that it not 100% pure. Therefore the formula may not suitable for a dataset which has more than 2 classes.

As our lecture mentioned, I also did some research, for a dataset which contains more than 2 classes we can based on gini-impurity instead of based on the impurity calculated by the formula we are taught in the class.

Part 3:

1. In this algorithm I set m = 250, the accuracy I got is 100% Therefore it do find the correct set of weight. The number of cycle it converge is 222 (seed(307))

   Final-weight-matrix:

```
 [-25.679369005297755, -12.581381580241818, -32.59630828133717,
-32.11770922384291, -4.207725928853199, 9.023057775167054,
-4.511733898353011, 17.426147683607635, 34.906077970345656,
-29.34547687805869, 17.251492027990533, -1.0021385897737467,
-4.251247191715965, -5.645924539815258, 18.07217083808407,
1.9643561312950801, 1.5862549437249989, 11.312423569903354,
2.9291007520243926, 4.534380422619169, 10.638408887313716,
1.854175342114447, -0.6213398092081981, 37.56376775992486,
7.490564649533134, 21.408695455274778, -1.8989336837463355,
12.698889957999823, -16.550444058956927, 10.697886409831884,
-19.017110165117558, 17.428150507744242, -15.327617414185305,
-0.27988082767180167, 20.355899169281336, 10.901798932558108,
-3.475459467681569, -0.3237816106597915, 27.753827753085115,
-1.2702240155130222, -2.6540628215365807, -8.378778846104918,
-14.063300194451664, 1.7960763488011615, 14.329622651870977,
-2.6941363749060994, -27.83187115750613, -1.5871715988300297,
-4.0109318670068355, -22.293557390454204, -1.4574000977863284]
```

2:

Because having a high training accuracy does not mean the model is perfect, the high training accuracy may caused by overfitting. Therefore we should use some test, or we can use the technique K-cross-validation as I mentioned above to reduce overfitting, or we can use a validation set which is discussed in the lecture. Therefore we should not only care about the training accuracy but both the validation accuracy(test accuracy) and the training accuracy are important. I do tried some data I created, with feeding a typical “X” image, the model can classify it correctly, but once feed the model with some “X” which is not very obvious, it will give an accuracy around 50%. Hence, I think this model is overfitting.