COMP301 – Assignment 1 Report

Part 1:

1. Report the class labels of each instance in the test set predicted by the basic nearest neighbor method (where k=1), and the classification accuracy on the test set of the basic nearest neighbor method;

K = 1

```
1
Iris-setosa | Values = 1.4, 0.2, 5.1, 3.5

2
Iris-setosa | Values = 1.4, 0.2, 4.9, 3.0

3
Iris-setosa | Values = 1.3, 0.2, 4.7, 3.2

4
Iris-setosa | Values = 1.5, 0.2, 4.6, 3.1

5
Iris-setosa | Values = 1.4, 0.2, 5.0, 3.6

6
Iris-setosa | Values = 1.7, 0.4, 5.4, 3.9

7
Iris-setosa | Values = 1.4, 0.3, 4.6, 3.4

8
Iris-setosa | Values = 1.5, 0.2, 5.0, 3.4

9
Iris-setosa | Values = 1.4, 0.2, 4.4, 2.9

10
Iris-setosa | Values = 1.5, 0.1, 4.9, 3.1

11
Iris-setosa | Values = 1.5, 0.2, 5.4, 3.7

12
Iris-setosa | Values = 1.6, 0.2, 4.8, 3.4
```

```
13
Iris-setosa | Values = 1.4, 0.1, 4.8, 3.0

14
Iris-setosa | Values = 1.1, 0.1, 4.3, 3.0

15
Iris-setosa | Values = 1.2, 0.2, 5.8, 4.0

16
Iris-setosa | Values = 1.5, 0.4, 5.7, 4.4

17
Iris-setosa | Values = 1.3, 0.4, 5.4, 3.9

18
Iris-setosa | Values = 1.4, 0.3, 5.1, 3.5

19
Iris-setosa | Values = 1.7, 0.3, 5.7, 3.8

20
Iris-setosa | Values = 1.5, 0.3, 5.1, 3.8

21
Iris-setosa | Values = 1.7, 0.2, 5.4, 3.4

22
Iris-setosa | Values = 1.5, 0.4, 5.1, 3.7

23
Iris-setosa | Values = 1.0, 0.2, 4.6, 3.6

24
Iris-setosa | Values = 1.7, 0.5, 5.1, 3.3

25
Iris-setosa | Values = 1.9, 0.2, 4.8, 3.4

26
Iris-versicolor | Values = 4.7, 1.4, 7.0, 3.2

27
Iris-versicolor | Values = 4.5, 1.5, 6.4, 3.2

28
Iris-versicolor | Values = 4.9, 1.5, 6.9, 3.1
```

```
29
Iris-versicolor | Values = 4.0, 1.3, 5.5, 2.3

30
Iris-versicolor | Values = 4.6, 1.5, 6.5, 2.8

31
Iris-versicolor | Values = 4.5, 1.3, 5.7, 2.8

32
Iris-versicolor | Values = 4.7, 1.6, 6.3, 3.3

33
Iris-versicolor | Values = 3.3, 1.0, 4.9, 2.4

34
Iris-versicolor | Values = 4.6, 1.3, 6.6, 2.9

35
Iris-versicolor | Values = 3.9, 1.4, 5.2, 2.7

36
Iris-versicolor | Values = 3.5, 1.0, 5.0, 2.0

37
Iris-versicolor | Values = 4.2, 1.5, 5.9, 3.0

38
Iris-versicolor | Values = 4.0, 1.0, 6.0, 2.2

39
Iris-versicolor | Values = 4.7, 1.4, 6.1, 2.9

40
Iris-versicolor | Values = 3.6, 1.3, 5.6, 2.9

41
Iris-versicolor | Values = 4.4, 1.4, 6.7, 3.1

42
Iris-versicolor | Values = 4.5, 1.5, 5.6, 3.0

43
Iris-versicolor | Values = 4.1, 1.0, 5.8, 2.7

44
Iris-versicolor | Values = 4.5, 1.5, 6.2, 2.2

45
```

```
Iris-versicolor | Values = 3.9, 1.1, 5.6, 2.5

46
Iris-versicolor | Values = 4.8, 1.8, 5.9, 3.2

47
Iris-versicolor | Values = 4.0, 1.3, 6.1, 2.8

48
Iris-versicolor | Values = 4.9, 1.5, 6.3, 2.5

49
Iris-versicolor | Values = 4.7, 1.2, 6.1, 2.8

50
Iris-versicolor | Values = 4.3, 1.3, 6.4, 2.9

51
Iris-virginica | Values = 6.0, 2.5, 6.3, 3.3

52
Iris-virginica | Values = 5.1, 1.9, 5.8, 2.7

53
Iris-virginica | Values = 5.9, 2.1, 7.1, 3.0

54
Iris-virginica | Values = 5.6, 1.8, 6.3, 2.9

55
Iris-virginica | Values = 5.8, 2.2, 6.5, 3.0

56
Iris-virginica | Values = 6.6, 2.1, 7.6, 3.0

57
Iris-virginica | Values = 4.5, 1.7, 4.9, 2.5

58
Iris-virginica | Values = 6.3, 1.8, 7.3, 2.9

59
Iris-virginica | Values = 5.8, 1.8, 6.7, 2.5

60
Iris-virginica | Values = 6.1, 2.5, 7.2, 3.6

61
Iris-virginica | Values = 5.1, 2.0, 6.5, 3.2
```

```
62
Iris-virginica | Values = 5.3, 1.9, 6.4, 2.7

63
Iris-virginica | Values = 5.5, 2.1, 6.8, 3.0

64
Iris-virginica | Values = 5.0, 2.0, 5.7, 2.5

65
Iris-virginica | Values = 5.1, 2.4, 5.8, 2.8

66
Iris-virginica | Values = 5.3, 2.3, 6.4, 3.2

67
Iris-virginica | Values = 5.5, 1.8, 6.5, 3.0

68
Iris-virginica | Values = 6.7, 2.2, 7.7, 3.8

69
Iris-virginica | Values = 6.9, 2.3, 7.7, 2.6

70
Iris-virginica | Values = 5.0, 1.5, 6.0, 2.2

71
Iris-virginica | Values = 5.7, 2.3, 6.9, 3.2

72
Iris-virginica | Values = 4.9, 2.0, 5.6, 2.8

73
Iris-virginica | Values = 6.7, 2.0, 7.7, 2.8

74
Iris-virginica | Values = 4.9, 1.8, 6.3, 2.7

75
Iris-virginica | Values = 5.7, 2.1, 6.7, 3.3

==================RESULTS=================
|
|  CORRECT CLASSIFICATION:  71 / 75
|-----------------------------------------
|  ACCURACY:                        94.67%
|
```

==========

2. Report the classification accuracy on the test set of the k-nearest neighbour method where k=3,and compare and comment on the performance of the two classifiers (k=1 and k=3);

K = 3

```
=================RESULTS================
|
|  CORRECT CLASSIFICATION:  72 / 75
|----------------------------------------
|  ACCURACY:                        96.00%
|
========================================
```

K = 1 Accuracy = 71/75 = 94.67%

K = 3 Accuracy = 72/75 = 96%

The performance for when K = 3 is slightly higher than K = 1, The higher k value means more closest neighbours are compared. This means that when K = 3 there is a “higher resolution” when predicting.

3. Discuss the main advantages and disadvantages of k-Nearest Neighbour method.

The main advantages of K-nearest Neighbor include the following:

- The method is simpler than others and is relatively easy to understand and implement
- Works well if there is a large training set
- It allows a variety of distance criteria, in our case we used Euclidean Distance.
- Almost no assumptions

The main disadvantages of K-nearest Neighbor include the following:

- The algorithm requires a ‘k’ value which needs to be determined first
- The algorithm is slow and costly
- It is sensitive to outliers; this has an effect on the overall accuracy.
- Can’t deal with missing values

4. Assuming that you are asked to apply the k-fold cross validation method for the above problem with k=5, what would you do? State the major steps.

- Split the data into 5 folds
- Use first fold for testing and the rest for training

- Next step is to use the second fold for testing and the other for training. Keep doing this while using the next fold for testing
- Finally average the accuracy.

5. In the above problem, assuming that the class labels are not available in the training set and the test set, and that there are three clusters, which method would you use to group the examples in the data set? State the major steps.

   K-means Clustering

   - Three clusters => K = 3
   - Make K amount of predictions of where the cluster will be
   - For each element in data, assign it to the cluster it is closest to.
   - Move the center of each cluster to be in the middle of the elements that are assigned to that cluster
   - Repeat until no more data moves between clusters

Part 2:

1. You should first apply your program to the hepatitis-training.dat and hepatitis-test.dat files and report the classification accuracy in terms of the fraction of the test instances that it classified correctly. Report the learned decision tree classifier printed by your program. Compare the accuracy of your Decision Tree program to the baseline classifier which always predicts the most frequent class in the dataset, and comment on any difference.

   The baseline classifier of the training list is LIVE. It can be seen that the decision tree outputted 'LIVE' for the greater part.

```
ASCITES = True:
 SPIDERS = True:
  VARICES = True:
   FIRMLIVER = True:
    Class live, Probability = 1.00
   FIRMLIVER = False:
    BIGLIVER = True:
     STEROID = True:
      Class live, Probability = 1.00
     STEROID = False:
      FEMALE = True:
```

```
          Class live, Probability = 1.00
         FEMALE = False:
          ANTIVIRALS = True:
           FATIGUE = True:
            Class die, Probability = 1.00
           FATIGUE = False:
            Class live, Probability = 1.00
          ANTIVIRALS = False:
           Class die, Probability = 1.00
       BIGLIVER = False:
        Class live, Probability = 1.00
     VARICES = False:
      Class die, Probability = 1.00
    SPIDERS = False:
     FIRMLIVER = True:
      AGE = True:
       Class live, Probability = 1.00
      AGE = False:
       SGOT = True:
        Class live, Probability = 1.00
       SGOT = False:
        ANTIVIRALS = True:
         Class die, Probability = 1.00
        ANTIVIRALS = False:
         STEROID = True:
          Class live, Probability = 1.00
         STEROID = False:
          Class die, Probability = 1.00
     FIRMLIVER = False:
      SGOT = True:
       BIGLIVER = True:
        SPLEENPALPABLE = True:
         Class live, Probability = 1.00
        SPLEENPALPABLE = False:
         ANOREXIA = True:
          Class die, Probability = 1.00
         ANOREXIA = False:
          Class live, Probability = 1.00
       BIGLIVER = False:
        Class die, Probability = 1.00
      SGOT = False:
       Class live, Probability = 1.00
   ASCITES = False:
    BIGLIVER = True:
     STEROID = True:
      Class die, Probability = 1.00
     STEROID = False:
      ANOREXIA = True:
       Class die, Probability = 1.00
```

```
      ANOREXIA = False:
       Class live, Probability = 1.00
    BIGLIVER = False:
     Class live, Probability = 1.00

=================RESULTS================
|
|  CORRECT CLASSIFICATION:  21 / 27
|----------------------------------------
|  ACCURACY:                        77.78%
|
========================================
Baseline Classifier = LIVE
```

2. You should then construct 10 other pairs of training/test files, train and test your classifiers on each pair, and calculate the average accuracy of the classifiers over the 10 trials. Show you working. There is a scriptsplit-datafile that takes the name of the full data set (e.g, hepatitis), the number of training instances, and a suffix for the filenames, and will construct pairs of training and test files. For example. /split-datafile hepatitis.dat 100 run1will construct the fileshepatitis-training-run1.datandhepatitis-test-run1.datwith 100and 37 instances respectively. This process needs to run 10 times.

   I ran the split-datafile script 10 times on Windows using the Windows Subsystem for Linux. I used the "./split-datafile hepatitis.dat 100 run1" command to create 10 different files, from run1 to run10. I created a new class to run each of the 10 new files.

```
public class Run10 {
    private double avg = 0;
    public Run10(String directory){
        double average = 0;
        for(int i = 1; i < 11; i++) {
        DecisionTree k = new DecisionTree(directory + "hepatitis-training-run" + i +".dat", directory + "hepatitis-test-run" + i + "
        avg += k.getAccuracyPercent();
        //System.out.println(avg);
        }
        average = (avg/10);
        System.out.printf("Accuracy Average Of 10 Trials = %.2f%% ", average);
    }
}
```

```
Accuracy Average Of 10 Trials = 79.46%
```

3. Pruning" (removing) some of leaves of the decision tree will always make the decision tree less accurate on the training set. Explain (a) How you could prune leaves from the decision tree; (b)Why it would reduce accuracy on the training set, and (c)Why it might improve accuracy on the test set.
   a. Pruning is the inverse of splitting => Any pair whose elimination yields a satisfactory (small) increase in impurity is eliminated, and the common parent node becomes leaf node. This ultimately reduces the size of the final tree. ~From Lecture slides.

b. Pruning would reduce accuracy since it eliminates overfitting. Overfitting allows you to achieve perfect accuracy on training data, this makes the decision tree very specific to the training data making it less useful for other sets of data.
c. It would increase accuracy on the test-set since pruning allows us to mitigate overfitting. It allows us to get a better test-set accuracy on other data sets since there is no overfitting.

4. Explain why the impurity measure is not a good measure if there are three or more classes that the decision tree must distinguish.

   Since the Impurity is the number of occurrences between two classes with a binary value, the measure shows the purity of the node. Since the value is either 0 or 1 this can't be translated to more than two classes. This would require fractions and make the measure much more complicated.

Part 3:

1. Report on the accuracy of your perceptron. For example, did it find a correct set of weights?

   I tested my perceptron's accuracy for 100 'attempts' and got an accuracy range between 88% – 100%(5 trials). After increasing the amount of 'attempts' to 150 the accuracy ranged between 94% - 100%(5 trials). My perceptron reached 100% accuracy multiple times indicating that it found the correct set of weights. It was much more achievable once the 'attempts' are increased to around 150.

2. Explain why evaluating the perceptron's performance on the training data is not a good measure of its effectiveness. You may wish to create additional data to get a better measure. If you do, report on the perceptron's performance on this additional data.

   Like the decision tree, the perceptron is prone to overfitting. Increasing the number of attempts only trains the perceptron to suit the data set even more which increases the accuracy. This is not good since overfitting will cause the perceptron to be less accurate on new unseen tests.