Assignment #1

COMP 307 Introduction to Artificial Intelligence

Qiangqiang Li(Aaron)

ID: 300422249

Tutors: Prof. Mengjie Zhang

Dr. Yi Mei

Part 1: Nearest Neighbour Method

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

Reply : where k=1 , the classification accuracy is 90.67%

- ***Iris-setosa : correct***
- ***Iris-setosa : correct***
- ***Iris-setosa : correct***
- ***Iris-setosa : correct***
- ***Iris-setosa : correct***
- ***Iris-setosa : correct***
- ***Iris-setosa : correct***
- ***Iris-setosa : correct***
- ***Iris-setosa : correct***
- ***Iris-setosa : correct***
- ***Iris-setosa : correct***
- ***Iris-setosa : correct***
- ***Iris-setosa : correct***
- ***Iris-setosa : correct***
- ***Iris-setosa : correct***
- ***Iris-setosa : correct***
- ***Iris-setosa : correct***
- ***Iris-setosa : correct***
- ***Iris-setosa : correct***
- ***Iris-setosa : correct***
- ***Iris-setosa : correct***
- ***Iris-setosa : correct***
- ***Iris-setosa : correct***
- ***Iris-setosa : correct***
- ***Iris-setosa : correct***
- ***Iris-versicolor : correct***
- ***Iris-versicolor : correct***
- ***Iris-virginica : incorrect***
- ***Iris-versicolor : correct***
- ***Iris-versicolor : correct***
- ***Iris-versicolor : correct***
- ***Iris-versicolor : correct***
- ***Iris-versicolor : correct***
- ***Iris-virginica : incorrect***
- ***Iris-versicolor : correct***
- ***Iris-versicolor : correct***
- ***Iris-versicolor : correct***
- ***Iris-versicolor : correct***
- ***Iris-versicolor : correct***
- ***Iris-versicolor : correct***
- ***Iris-versicolor : correct***
- ***Iris-versicolor : correct***
- ***Iris-versicolor : correct***
- ***Iris-versicolor :***

correct

- ***Iris-versicolor : correct***
- ***Iris-versicolor : correct***
- ***Iris-versicolor : correct***
- ***Iris-versicolor : correct***
- ***Iris-versicolor : correct***
- ***Iris-versicolor : correct***
- ***Iris-virginica : correct***
- ***Iris-virginica : correct***
- ***Iris-versicolor : incorrect***
- ***Iris-virginica : correct***
- ***Iris-virginica : correct***
- ***Iris-virginica : correct***
- ***Iris-virginica : correct***
- ***Iris-virginica : correct***
- ***Iris-versicolor : incorrect***
- ***Iris-versicolor : incorrect***
- ***Iris-virginica : correct***
- ***Iris-virginica : correct***
- ***Iris-virginica : correct***
- ***Iris-versicolor : incorrect***
- ***Iris-virginica : correct***
- ***Iris-virginica : correct***
- ***Iris-virginica : correct***
- ***Iris-virginica : correct***
- ***Iris-virginica : correct***
- ***Iris-virginica : correct***
- ***Iris-virginica : correct***
- ***Iris-virginica : correct***
- ***Iris-virginica : correct***
- ***Iris-virginica : correct***
- ***Iris-versicolor : incorrect***
- ***when k = 1,the accuracy is 90.67%***

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);

Reply : In K nearest neighbour result, if k=1,accuracy is 90.67%. If k=3, accuracy is 96.00%. So k=3 accuracy is higher than k=1 accuracy. The reason is that , when k=3 it consider more possibility, it looks for the 3 nearest neighbour, and pick up the class may be majority.

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

Reply :KNN-Advantages:the K-Nearest Neighbor (KNN) Classifier is a very simple classifier, which is works well on basic recognition problems. It is flexible to feature the value k can be easily to change, can be handles multi-class cases, and also can do well in practice with enough representative data.

KNN-Disadvantages:

- KNN is not a good method for high dimension spaces data If there are a large number of training examples, the algorithm must compute the distance and sort all the training data at each prediction, so can be slow.

- Another disadvantage of this approach is that the algorithm does not learn anything from the training data, which can result in the algorithm not generalizing well and also not being robust to noisy data. Further, changing *K* can change the resulting predicted class label.

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.

- Step1: Separate data(150 instances)into 5 chunks equally(30 instances each).
- Step2: Use first chunk as test set, the rest 4 chunks as training set. Train classifier using the training set, apply first chunk to the test set. Get accuracy for this classifier.
- Step3: Repeat Step2 4 times, use each chunk as test set, rest chunks as training set.
- Step4:Get all the classification accuracy and calculate the average accuracy.
- Step5:Compare with each other (when k= 1,2,3,4,5…) then selecting the highest classification accuracy is the best solution for this data set.

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.

Reply : Use the K-means method will solve the problem.

- Step1: Set k initial "means" randomly from the data set.
- Step2: Create k clusters by associating every instance with the nearest mean based on a distance measure.
- Step3: Replace the old means with the centroid of each of the k clusters (as the new means).
- Step4: Repeat the above two steps until convergence (no change in each cluster center).

6. (Optional, bonus 5 marks) Implement the clustering method above.

Reply :

the code is in code bag: k-mean.py , and test k= 3.

```
data_list, ranging = data_process("iris.data")
k = 3
cluster_center_list = random.sample(data_list, k)
clustercenter1 = cluster_center_list[0][0]
clustercenter2 = cluster_center_list[1][0]
clustercenter3 = cluster_center_list[2][0]
count = 0
k_mean_cluster(data_list, ranging, clustercenter1, clustercenter2, clustercenter3, count)
```

the report result :

```
 results after 1 times
clustercenter1: [4.902564102564102, 3.056410256410256, 1.8025641025641022,
0.358974358974359]
clustercenter2: [6.338709677419353, 2.909677419354839, 5.016129032258065,
1.7247311827956997]
clustercenter3: [5.322222222222222, 3.794444444444445, 1.5, 0.30000000000000004]
results after 2 times
clustercenter1: [4.8, 3.0300000000000002, 1.6999999999999997, 0.32000000000000006]
clustercenter2: [6.314583333333331, 2.8958333333333335, 4.973958333333335,
1.7031250000000007]
clustercenter3: [5.2625, 3.7166666666666663, 1.4708333333333334, 0.2791666666666667]
results after 3 times
clustercenter1: [4.774074074074075, 2.9888888888888894, 1.7185185185185186,
0.325925925925926]
```

clustercenter2: [6.314583333333331, 2.8958333333333335, 4.973958333333335, 1.7031250000000007]
clustercenter3: [5.237037037037037, 3.681481481481481, 1.4777777777777776, 0.2777777777777778]
results after 4 times
clustercenter1: [4.773076923076923, 2.9730769230769236, 1.723076923076923, 0.3307692307692308]
clustercenter2: [6.314583333333331, 2.8958333333333335, 4.973958333333335, 1.7031250000000007]
clustercenter3: [5.221428571428571, 3.6714285714285713, 1.482142857142857, 0.275]
results after 5 times
clustercenter1: [4.772, 2.956000000000001, 1.7159999999999997, 0.3360000000000001]
clustercenter2: [6.314583333333331, 2.8958333333333335, 4.973958333333335, 1.7031250000000007]
clustercenter3: [5.206896551724138, 3.6620689655172414, 1.4965517241379307, 0.2724137931034483]
results after 6 times
clustercenter1: [4.7625, 2.941666666666667, 1.7291666666666667, 0.34166666666666673]
clustercenter2: [6.314583333333331, 2.8958333333333335, 4.973958333333335, 1.7031250000000007]
clustercenter3: [5.2, 3.65, 1.4933333333333333, 0.2700000000000001]
results after 7 times
clustercenter1: [4.754545454545454, 2.9045454545454548, 1.7454545454545451, 0.33636363636363636]
clustercenter2: [6.314583333333331, 2.8958333333333335, 4.973958333333335, 1.7031250000000007]
clustercenter3: [5.178125, 3.63125, 1.4968749999999997, 0.27812499999999996]
The results after 7 times and after 8 times are same, so the final center is:
clustercenter1: [4.754545454545454, 2.9045454545454548, 1.7454545454545451, 0.33636363636363636]
clustercenter2: [6.314583333333331, 2.8958333333333335, 4.973958333333335, 1.7031250000000007]
clustercenter3: [5.178125, 3.63125, 1.4968749999999997, 0.27812499999999996]

Part2: Decision Tree Learning Algorithm
1. 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.

Reply : the result is below:

```
Reading training data from file hepatitis-training.data
2 categories
16 attributes Reading test data from file hepatitis-test.data
Read 27 instances
Read 16 attributes
[AGE, FEMALE, STEROID, ANTIVIRALS, FATIGUE, MALAISE, ANOREXIA, BIGLIVER, FIRMLIVER,
SPLEENPALPABLE, SPIDERS, ASCITES, VARICES, BILIRUBIN, SGOT, HISTOLOGY]

Desction tree:
The learned decision tree classifier is
  ASCITES = True
    SPIDERS = True
      VARICES = True
```

```
         FIRMLIVER = True
            Class live, prob = 1, /49
         FIRMLIVER = False
            BIGLIVER = True
               STEROID = True
                  Class live, prob = 1, /5
               STEROID = False
                  HISTOLOGY = True
                     SGOT = True
                        Class live, prob = 1, /1
                     SGOT = False
                        Class die, prob = 1, /1
                  HISTOLOGY = False
                     ANTIVIRALS = True
                        Class live, prob = 1, /4
                     ANTIVIRALS = False
                        BILIRUBIN = True
                           Class die, prob = 1, /1
                        BILIRUBIN = False
                           Class live, prob = 1, /1
            BIGLIVER = False
               Class live, prob = 1, /7
      VARICES = False
         Class die, prob = 1, /1
   SPIDERS = False
      FIRMLIVER = True
         SGOT = True
            Class live, prob = 1, /1
         SGOT = False
            FEMALE = True
               Class live, prob = 1, /1
            FEMALE = False
               HISTOLOGY = True
                  Class die, prob = 1, /4
               HISTOLOGY = False
                  ANOREXIA = True
                     Class die, prob = 1, /1
                  ANOREXIA = False
                     Class live, prob = 1, /1
      FIRMLIVER = False
         SGOT = True
            BIGLIVER = True
               SPLEENPALPABLE = True
                  Class live, prob = 1, /4
               SPLEENPALPABLE = False
                  ANOREXIA = True
                     Class die, prob = 1, /2
                  ANOREXIA = False
                     Class live, prob = 1, /1
            BIGLIVER = False
               Class die, prob = 1, /3
         SGOT = False
            Class live, prob = 1, /10
ASCITES = False
   BIGLIVER = True
      STEROID = True
         Class die, prob = 1, /7
      STEROID = False
```

```
        ANOREXIA = True
          Class die, prob = 1, /2
        ANOREXIA = False
          Class live, prob = 1, /2
    BIGLIVER = False
      Class live, prob = 1, /1
```

```
The learned decision tree is applied to test data
Read: 27 instances
Baseline class is: live
live: 20 correct out of 23
die: 2 correct out of 4
Accuracy: 81.48%
Baseline Accuracy: 86.96%
```

Reply : Decision Tree accuracy is 81.48%. Baseline accuracy is 23/27, which is 86.96%. Decision Tree accuracy is higher than Baseline accuracy. Baseline classifier always predicts the most frequent class in the dataset, all test instances are classified as “live” in this data. In this cas, the “live” lable have high frequent class, Baseline accuracy will higher than Decision Tree accuracy. If the data have half each lable, Decision Tree accuracy will be high than Baseline accuracy.

2.Show you working. There is a script split-datafile that takes the name of the full data set (eg, hepatitis), the number of training instances, and a suffix for the filenames, and will construct pairs of training and test files.

Reply : When running the process for 10 times, the average accurate rate are : 78.38%.
The outputs are below:

- run1
- Read: 37 instances
- Baseline class is: live
- live: 30 correct out of 34
- die: 1 correct out of 3
- Accuracy: 83.78%
- run2
- Read: 37 instances
- Baseline class is: live
- live: 30 correct out of 33
- die: 2 correct out of 4
- Accuracy: 86.49%
- run3
- Read: 37 instances
- Baseline class is: live
- live: 26 correct out of 32
- die: 4 correct out of 5
- Accuracy: 81.08%
- run4
- Read: 37 instances
- Baseline class is: live
- live: 22 correct out of 29
- die: 4 correct out of 8
- Accuracy: 70.27%
- run5
- Read: 37 instances
- Baseline class is: live
- live: 27 correct out of 33
- die: 1 correct out of 4
- Accuracy: 75.68%
- run6
- Read: 37 instances
- Baseline class is: live
- live: 21 correct out of 29
- die: 4 correct out of 8
- Accuracy: 67.57%
- run7
- Read: 37 instances
- Baseline class is: live
- live: 30 correct out of 33
- die: 1 correct out of 4
- Accuracy: 83.78%
- run8
- Read: 37 instances
- Baseline class is: live
- live: 24 correct out of 32
- die: 1 correct out of 5
- Accuracy: 67.57%
- run9
- Read: 37 instances
- Baseline class is: live
- live: 26 correct out of 31
- die: 3 correct out of 6
- Accuracy: 78.38%
- run10
- Read: 37 instances
- Baseline class is: live
- live: 26 correct out of 30
- die: 3 correct out of 7
- Accuracy: 78.38%
- Average accuracy: 77.30%

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;

Reply : Pruning is the inverse of splitting.After a tree has been built ,it may be overfitted. The CART algorithm will repeatedly partition data into smaller and smaller subsets until those final subsets are homogeneous in terms of the outcome variable. If this node is irrelevant , then this node can be replaced as a leaf node. Any pair whose elimination yields a satisfactory (small) increase in impurity is eliminated, and the common parent node becomes leaf node Repeat this process.

(b) Why it would reduce accuracy on the training set

Reply : When do pruning for DT, the complexity of DT is reduced. This would reduce the fitness of DT for training data. DT will eliminate some irrelevant nodes. These nodes can split data more detailed . So it would reduce accuracy on the training set.

(c)Why it might improve accuracy on the test set.

Reply : The high accuracy for training data is overfitting. Pruning method is a way to reduce overfitting on the training set. So it would increase accuracy on the test set.

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

Reply : Because when there are three or more classes the algorithm that use to calculate the purity is not accurate. If there are 3 classes, the formula for impurity should be impurity = 3abc/(a+b+c).
If there are two attributes X ,Y, X with label a and b, Y is all label a or b, Both of them do not have label c. impurity(X) = 3a*b*0/(a+b+0)= 0. impurity(Y) = 3a*0*0/(a+0+0)= 0, X and Y have same impurity. However, their impurity should be different obviously. Because when one class do not have any instances used the formula to calculate the output will become 0.

Part3: Perceptron
1. Report on the accuracy of your perceptron. For example, did it find a correct set of weights?

Reply : After 165 times learning, the matched number of image is 100:

```
Accuracy: 1.00
Final weight set is:165
[-8.865635755887599, 0.8474337369372327, 20.763774618976612, -17.744930974260576,
-0.5045649129080587, -10.550508935211262, 15.651592972722764, -3.2112766488644873,
-10.906140413225765, -10.971652523477994, 1.8357651039198695, -7.567232932094947,
14.762280082457941, 8.00210605335111, -16.554612805945197, 1.7215400323407835,
-7.771237778729548, 10.945270695553923, 5.901427457611483, 50.03058998303355,
2.025445860993461, -9.458587527206504, -4.060850837221489, 0.38120423768821243,
-14.783400602869385, 20.422116575582717, -4.970959212425132, 22.221691666273035,
7.437887593650572, -0.5041877586181494, -16.76691554974243, -0.7691334584590157,
-14.781218962662312, 1.4596034657377341, -3.7102183854095143, 17.021489705265907,
-3.162422024337427, -24.443545677347565, 12.642294362932446, -20.81409373410528,
4.992543412176065, -6.14005347120471, 1.1208899598058064, 5.332695185360129,
-28.27851559241673, 0.7111917696952794, -26.06355941320054, 7.422106999961414,
4.830035693274327, 11.670305566414072, 5.303368510932918]
```

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.

Reply :.When we all the training data to build the perceptron and use the training data to evaluate the perceived performance, the result will be too good, which is called overfitting. This will reduce or destroy model generalisation ability. Which will decreasing the accurate rate when doing the tests. although the Accurate rate are up to 100%.

For example, we create 6 addition images for test (test1, test2, test3,test4,test5,test6). Test1 is a typical 0, which is Incorrect. The other test file are the typical X . which is correct..So the result is incorrect. This is overfitting.

The test report:

```
 from file: test1 out: Incorrect!
 from file: test2 out: Correct!
 from file: test3 out: Correct!
 from file: test4 out: Correct!
 from file: test5 out: Correct!
 from file: test6 out: Correct!
```