

Homework 4 Written Portion

due November 17, 2005 at 1pm

Question 1: k-Nearest-Neighbor [10]

In this question we look at the influence of distance measures on k-Nearest-Neighbor. You will see that nearest-neighbor can be very sensitive to the distance measures. Let's assume in two-dimensional space we measure distances in two ways:

Distance measure 1: the standard Euclidean distance, which for points $a=(x_a, y_a)$ and $b=(x_b, y_b)$ is $d_1(a,b) = \sqrt{(x_a-x_b)^2 + (y_a-y_b)^2}$

Distance measure 2: we weigh the y dimension by 10, which gives us a new distance measure $d_2(a,b) = \sqrt{(x_a-x_b)^2 + (10*y_a-10*y_b)^2}$. Equivalently, think of the space as a piece of rubber. Grab it at top and bottom, and pull the y-axis 10 times longer. $d_2()$ is equivalent to the standard Euclidean distance in this stretched space.

With these two distance measures, for the same set of points, some points might have a different 1-nearest-neighbor. For example in the figure below, the left panel shows three points a, b, c in the original space (using measure d_1). a's 1NN is c, b's 1NN is a, c's 1NN is a. The right panel shows the stretched space (equivalent to using measure d_2). Now a's 1NN has changed to b, while b's and c's 1NN remains unchanged.



a) Can you find a configuration of 4 points a, b, c, d, such that when we switch from distance measure d_1 to d_2 , **none** of the points will change their 1NN? Without loss of generality, **we fix a at coordinate (0,0)**. None of a, b, c, d should overlap. For any point, there must be one unique 1NN (i.e. no ties). If yes, write down the coordinates of b, c, d, AND briefly justify your answer. You may draw a figure if it helps you explain your answer. [2]

b) Can you find a configuration where exactly **1 point** will change its 1NN? Everything else is the same as above. [2]

c) How about where exactly **2 points** will change their 1NN? [2]

d) How about where exactly **3 points** will change their 1NN? [2]

e) How about where all **4 points** will change their 1NN? [2]

Question 2: Cross Validation [8]

In this question we investigate the pathology of Leave-One-Out Cross Validation (LOOCV). Let the dataset contain 100 **one-dimensional** points. There are two classes. 50 of the points are in class 1, and the other 50 in class 2. The points do not overlap. Leave-One-Out Cross Validation is similar to N-Fold cross validation, but where the split between training and testing set is all datapoints except one are used as the training set and the one held out value is used as the test set. This is done for each datapoint.

- a) Can you think of a situation where the LOOCV accuracy on this dataset, with 1NN classifier, is 0%? Briefly explain. [4]
- b) Can you think of a situation where the LOOCV accuracy on this dataset is 100% with 1NN classifier, but 0% with 3NN classifier? Briefly explain. [4]

Question 3: Classification rule [12]

In this question we will learn what the correct way to make predictions is.

- a) Let there be a biased coin with $p(\text{head})=0.6$ and $p(\text{tail})=0.4$. For each coin toss, if we always predict 'head', what is our average accuracy? [2]
- b) If for each coin toss above, we generate a random number x between 0 and 1 using a computer, and predict 'head' if $x \leq 0.6$, 'tail' if $x > 0.6$, what is our average accuracy? [3]
- c) In general for a k -sided biased die with probabilities (p_1, \dots, p_k) for sides 1 ... k , for each die roll, what should we predict? What is our average accuracy? [3]
- d) If for each die roll above, we predict 1 with probability p_1 , predict 2 with probability p_2 , ..., what is our average accuracy? [2]
- e) Briefly explain the common problem of b), d). [2]

Question 4: Decision tree [12]

After hearing decision trees from you at a cocktail party, your friend John thinks there is a bug in decision tree building algorithm. John points out that a node should stop splitting whenever all candidate questions have zero information gain there. (For this problem, all questions are of the form "What's the value of a feature?")

- a) Do you agree with John? If you agree, briefly justify your answer. If you do not agree, design a dataset by filling in the binary (values in T or F) features A, B below to prove John wrong. Justify your answer. [4]

A	B	Outcome Y
		T
		T
		F
		F

b) John is also not convinced that there exists a better tree than the one built by choosing the best question (highest information gain) at each node.

i) Convince John by designing a dataset with three binary features A, B, C below. [2]

A	B	C	Outcome Y
			T
			T
			F
			F

ii) Show the height-2 (each path is at most root – one internal node – leaf) decision tree built by choosing the best question at each node. [2]

iii) Compute the information gain of each candidate question at each non-leaf node (show your work). Compute the training accuracy of your tree. [2]

iv) Show another height-2 decision tree with better training accuracy. [2]

Question 5: Ensemble learning [7]

Consider an ensemble learning algorithm that uses simple majority voting among M learned hypotheses. Suppose that each hypothesis has error ϵ and that the errors made by each hypothesis are independent of the others'.

a) Calculate a formula for the error of the ensemble algorithm in terms of M and ϵ , and evaluate it for the cases where $M=5, 11$, and 21 and $\epsilon=0.1, 0.2$, and 0.4 . [5]

b) If the independence assumption is removed, is it possible for the ensemble error to be worse than ϵ ? Justify your answer. [2]

Question 6: Program Portion [40]

Building Decision Trees in Java

For this problem you will implement in Java a version of the decision-tree induction algorithm of Fig 18.5. Specifically, we'll assume that all features are *discrete valued*.

We have provided a sample dataset to help in program creation and for model evaluation. Download it from the course website. This dataset is based on mushroom records drawn from the Audubon Society Field Guide to North American Mushrooms. The task is: given a description of a sample mushroom classify it as edible or poisonous. More information about the dataset and about other datasets can be found at the UC-Irvine archive of machine learning datasets, <http://www.ics.uci.edu/~mlearn/MLRepository.html>. During program creation you will want to divide this dataset into two disjoint sets, a training set and a testing set.

Be aware that we will test your program on other datasets.

Create a *HW4* class, whose calling convention is as follows:

```
java HW4 <printTreeFlag> <trainsetFilename> <testsetFilename>
printTreeFlag is 1 or 0.
    1 - print tree
    0 - don't print tree
```

trainsetFilename - the file containing the training dataset
testsetFilename - the file containing the test dataset

Here is what the program should do:

1. Use the TRAINING SET of examples to build a decision tree. You must use Information Gain from section 18.3 of the course text to find the best attribute.
2. Print out the induced decision tree if printTreeFlag is 1 (using simple, indented ASCII text**).
3. Classify each example in the TESTING SET using the induced tree, printing out example names, predicted classification, and the actual classification. Finally print out the FRACTION that were correctly classified (accuracy). If a test example has values for attributes that were not seen during training, then when you classify the example, if you reach a node that needs a value for the attribute, use the majority class of the examples from the training set that reached that node.

Dataset Format

All data files will contain comma separated discrete features. White space should be ignored. The first element on each line is the example's name. The second element is the classification feature. The remaining elements are the input features that can be used as internal nodes of the decision tree. Lines beginning with a double slash are comments and should be ignored by your program. Shown below is an example data file. The example lines intentionally use different amounts of white space. Your program should be able to handle that. But each example will always have the same number of features.

```
//example data file
//classification of mushrooms into edible=e or poisonous=p
//input features are:
//cap-shape: bell=b,conical=c,convex=x,flat=f,knobbed=k,sunken=s
//odor: almond=a,anise=l,creosote=c,fishy=y,foul=f,musty=m,none=n,pungent=p,spicy=s
//etc.
Ex1, e, b, a, e, z, f, n, n, n, k
Ex2, p, c, a, f, z, k, g, n, n, l
Ex3, p, s, y, a, f, k, n, n, y, l
//etc.
```

Program portion to Turn In

Your file must be called HW4.java. Copy it along with any other .java files necessary to compile and run into ~cs540-2/handin/<login name>/hw4.

Question 7: Evaluation of Model [11]

Now that you have a working program we would like you to evaluate how well this program performs on the given mushroom dataset using 3-fold cross validation. ***Randomly*** divide the dataset into 3 disjoint sets: set1, set2, and set3. To perform 3-fold cross validation you will run your program 3 times, training on a pair of datasets (concatenated together) and testing on the remaining set. The setup should be like this:

```
train: set1, set2      test: set3
train: set2, set3      test: set1
train: set1, set3      test: set2
```

a) Report the accuracy of each run, as well as the average accuracy and the standard deviation. [3]

b) Now run your program again on the entire dataset used for the training set AND the testing set. Report your accuracy. [1]

c) This implementation of Decision Trees will, if possible, split datasets until every example in the training data is labeled correctly. Explain why for some datasets this will not optimize future performance. Suggest one possible method to modify this algorithm and handle this issue. [4]

d) Create a confusion matrix for each run, and create a confusion matrix for the total of all runs by summing up the squares from each run. [3]

*An easy way to divide a dataset into random disjoint sets is to attach a random number to each example, sort the examples, and then split it into disjoint sets.

**A simple way to print out a decision tree would look like the following:

```
att1=b
  att2=a
    class=p
  att2=l
    class=e
att1=c
  class=e
att1=x
  att2=a
    class=e
  att2=l
    class=p
```