***20CS6037***

Instructor: Anca Ralescu  
***Machine Learning***   
***Fall Semester 2017***

**Assignment 3**

**Assigned 10/5/2017**

**Due 10/15/2017**

**11:59PM**

**40 points**

**In a comment section at the top of your program list all the team members.**

This assignment is asking you to implement ID3 and Naïve Bayes classifiers on the Iris Data Set (<https://en.wikipedia.org/wiki/Iris_flower_data_set>).

As I had already mentione din class, the Iris Data Set contains 150 data points (observations) of three types of the iris flower, classified into three categories, versicolor, virginica, and setosa. Each observation is a vector with 4 components, with the following respective meanings – petal length, petal width, sepal length, sepal width. There is some overlap between virginica and versicolor, but no overlap between these and setosa (see the plots on the pairs of dimensions C(4, 2) = 6 plots).

To make things easier, you are asked to recognize setosa vs the combined virginica and versicolor. This way, thereis a linear separation between the classes.

You are to implement in Matlab the following (NOTE: You are to actually write your own implementation and NOT USE MATLAB, or any other language package):

1. The basic ID3 algorithm
2. The Naïve Bayes classifier

For computing probabilities (needed in each of the two parts) use Matlab hist function, which outputs the frequency of values in a histogram bin.

Discretize by using the bin center (rounded to the nearest integer) as the discrete value of an attribute.

I will discuss Bayes in class….

See below ideas for ID3:

First load the iris data set using the following MATLAB instruction. See example below:

>> load iris.dat

>> who

Your variables are:

iris

**>> size(iris)**

ans =

150 5

**>> iris(1,:)**

ans =

51 35 14 2 1

**>> iris(10,:)**

ans =

49 31 15 1 1

**>> iris(100,:)**

ans =

57 28 41 13 2

**>> iris(140,:)**

ans =

69 31 54 21 3

**[iris(1:5,:); iris(51:55,:); iris(101:105,:)]**

ans =

51 35 14 2 1

49 30 14 2 1

47 32 13 2 1

46 31 15 2 1

50 36 14 2 1

70 32 47 14 2

64 32 45 15 2

69 31 49 15 2

55 23 40 13 2

65 28 46 15 2

63 33 60 25 3

58 27 51 19 3

71 30 59 21 3

63 29 56 18 3

65 30 58 22 3

The attributes are continously valued. To discretize the attribute values you can use hist Matlab function. For the iris data set, invoke hist on the first column as follows.

**>> [n,x]=hist(iris(:,1)) % default number of bins is 10**

n =

9 23 14 27 22 20 18 6 5 6

% n is the frequency of values in each bin

x =

Columns 1 through 7

4.4800 4.8400 5.2000 5.5600 5.9200 6.2800 6.6400

Columns 8 through 10

7.0000 7.3600 7.7200 % x holds the bin centers

**>> sum(n)** % check that all the data are assigned to a bin.

ans =

150

% number of bins n=12

**>> [n,x]=hist(iris(:,1), 12)**

n =

9 13 23 14 21 15 20 15 8 5 2 5

x =

Columns 1 through 7

4.4500 4.7500 5.0500 5.3500 5.6500 5.9500 6.2500

Columns 8 through 12

6.5500 6.8500 7.1500 7.4500 7.7500

+++++++++++++++++++++++++++++++++++

% number of bins n=15

**>> [n,x]=hist(iris(:,1), 15)**

n =

Columns 1 through 12

5 6 21 13 14 14 10 16 16 15 7 2

Columns 13 through 15

5 1 5

x =

Columns 1 through 7

4.4200 4.6600 4.9000 5.1400 5.3800 5.6200 5.8600

Columns 8 through 14

6.1000 6.3400 6.5800 6.8200 7.0600 7.3000 7.5400

Column 15

7.7800

+++++++++++++++++++++++++++++++++++++++++++++++++++

% n = 5 bins

**[n,x]=hist(iris(:,1), 5)**

n =

32 41 42 24 11

x =

4.6600 5.3800 6.1000 6.8200 7.5400

++++++++++++++++++++++++++++++++++++++++++++++++++

>> % how to compute entropy

>> % P = [p1, ..., pn] is prob distribution

>> % log2P is a vector of log2 values: log2p1, ..., log2pn

>> % p1 x log2 p1 + ... + pn log2 pn is the dot product of these

>> % two vectors

>> % example:

>> n

n =

32 41 42 24 11

>> % form p:

>> p = n/sum(n)

p =

0.2133 0.2733 0.2800 0.1600 0.0733

>> %check that it is a prob

>> sum(p)

ans =

1

>> % form logs

>> logp = log2(p)

logp =

-2.2288 -1.8713 -1.8365 -2.6439 -3.7694

>> % Compute the entropy: - dot product

**>> Entropy= -logp\*p' or Entropy = -sum(logp .\* p)**

Entropy =

2.2006

Run your program for bin numbers varying from 5 to 20 incrementing by 5. For each choice of the bins number, run your algorithm 10 times for different training-test data set pairs obtained at random.

Record **max accuracy, min accuracy**, and **average accuracy** over these 10 runs. Plot on the same figure (with different colors) these accuracy values as a function of the number of bins. Turn in, as usual on blackboard.

Turn in your program and sample runs.