Introduction to Machine Learning

Supervised Learning Exercise (Part 2)

These exercises can be solved in any programming language of your choice. It is assumed you are proficient with programming. It will be helpful if the language has a library to plot graphics.

In the following exercises the objective is to program algorithms that, given examples and an expected output learn to mimic the behavior present in the data.

1. Build a K-NN classifier that is specifically suited for the dataset from previous exercises (Supervised Learning, Part 1, Ex. 3). Given a "training-set" and a test example the classifier should calculate the euclidean distance from the test example to all the elements of the training-set, choose the K closest elements of the training-set and output this examples' classification as the class of the majority of the K closes training-set elements (the K-Nearest Neighbors). Split the data set randomly in two subsets (70% / 30%). Use the bigger subset as the "training-set" and the smaller as the test set. Run all test-examples through the classifier. Compare the scores of K-NN classifiers for K=3, 7 and 11. Repeat 30 times, with different splits for each value of K. Why should K always be an odd number?

2. Using the dataset from previous exercises (Supervised Learning, Part 1, Ex. 3), split the dataset in two according to the value of the first column (all examples where this column has zero value to one subset and all examples where this column has 1 value to another subset). Calculate the entropy of the 3 datasets (the complete dataset and the two subsets).

Remember that a set's (*S*) entropy is calculated by:

$$entropy(S) = -(p+) * log_2(p+) - (p-) * log_2(p-)$$

where *p+* is the percent of positive examples and *p-* the percent of negative examples.

Calculate the gain of the split of *S* by feature *a*:

$$G(S,a) = entropy(S) - \sum_v (|S_v| Entropy(S_v))/|S|$$

where $|S|$ is the number of elements in *S* and S_v is each of the subsets of S when partitioned by the value of *a*. What is the value of G(S,a)? What does it mean in terms of your ability to classify the elements of S?

Do the same for all features of your set. Which is the feature with greatest gain? How can you improve your chances of guessing a random examples' class using this information?

Explain how would you proceed to build a decision tree with this information.