COMP 135 – Machine Learning – Fall 2016

Empirical/Programming Assignment 1

Due date: Wednesday, 9/28 (by the beginning of class, both paper and electronically)
Please make sure that the code you submit runs on *homework.eecs.tufts.edu*.

1 Introduction

In this assignment you will experiment with the k nearest neighbors algorithm (kNN) and the decision tree learning algorithm, and evaluate kNN's sensitivity to the value of k and the relevance/number of features. You will use the Weka system for decision trees (J48) and will write your own code for kNN.

2 Data

The assignment uses 4 datasets available from the UCI repository[1]. To simplify your experiments we have prepared the data in a convenient form. Each dataset is split into a training portion and test portion for use in experiments. The data is already normalized and features do not require further preprocessing. The processed data files are accessible through the course web page.

3 Your Tasks

3.1 Evaluating Decision Trees

Run the default version of the J48 algorithm in Weka on all datasets and record the test set accuracy. Please note that in all experiments in this assignment we train and test on different portions of the data as prepared in advance. In Weka you can use the options `-t` and `-T` to specify the separate train and test files respectively. You might want to write a short script to perform this task. We will compare the performance of J48 to kNN below.

3.2 Reading Data Files

Write code to read data in data in `arff` format as used in the Weka system. To simplify your coding you may assume that all features are numerical. The class, which is the last column in the data rows is discrete, but it may have more than two values.
In view of implementing kNN and feature selection you should build in a facility to calculate the weighted Euclidean distance between examples: $d(x, y) = \sqrt{\sum_j w_j (x_j - y_j)^2}$. With this scheme, we get the standard distance with $w_j = 1$ for all j but we can also ignore some features by assigning $w_j = 0$. Note that distance calculation should not use the class label.

3.3 Implementing kNN

Implement your own version of k nearest neighbors. Your procedure should take three parameters corresponding to train set, test set, and the value of k. The search for neighbors can be done using linear time search, i.e., you need not worry about the computational improvements discussed in class.

[1] See http://archive.ics.uci.edu/ml/datasets.html

3.4 Evaluating kNN with respect to k

Run kNN on all datasets with values of k from 1 to 25 and record the test set accuracy. For each dataset, plot the accuracy of kNN as a function of k. On the same plot add the performance of J48 (as a flat line as it does not depend on k).
In your report include these plots and a short discussion. How do the two algorithms compare? and how does the performance of J48 vary with k?

3.5 Feature Selection for kNN

kNN's performance may degrade when the data has many irrelevant features. In this part you will implement and test a simple strategy to get around this. The idea in this assignment is to sort features by their information gain and select the top n features according to this ranking. Since our features are numerical we need to define an appropriate notion of information gain.
Please use the following scheme. First, divide the feature range into 5 parts, each of which includes a fifth of the examples. You can do this by sorting the examples according to this feature's value and picking the right boundaries. Then replace the numerical values with 5 corresponding categorical values. This gives a discrete feature which has 5 values. With this in mind you can evaluate the information gain for the feature.

Write code to perform this calculation and then rank the features by information gain. Once this is done evaluate kNN with $k = 5$ using the top n features for n from 1 to the total number of features and record the test set accuracy. Note that it should be easy to do this evaluation using the facility for weighted distance by assigning a weight of zero to features that are not being used in a particular run. For each dataset, plot the accuracy of kNN as a function of n. On the same plot add the performance of J48 (as a flat line since we only train it once using all the features).
In your report include these plots and a short discussion. How does the performance of kNN vary with n? and how does this affect the relationship between the two algorithms (for $k = 5$)? Does it look feasible that we can automatically choose appropriate values for k and n for each dataset?

3.6 For Extra Credit

[This part is not required; if you wish do it for extra fun and some extra credit]. The work above looks at trends on the test set for the effect of k and n. To pick such values we need an independent method that does not use the test set. Then we can evaluate kNN with the selected k and n only once on the test set.
In this part you can evaluate the following scheme or another scheme of your choice. First, split the train set into two parts - call them train and validation. Then redo the work of Section 3.5 to select n (using $k = 1$ or $k = 5$) but evaluating on the validation set. Once n is selected, redo the work of Section 3.4 to select k. Once this is done "train" kNN on the entire train/validation set using the specific values of k and n and test on the test set. Compare this value with the accuracy of J48. Write a short report on your method and findings.

4 Submitting your assignment

- You should submit the following items both electronically and in hardcopy:
  (1) All your code for data processing, learning algorithms, test program, and the experiments. Please write clear code and document it as needed.

  As stated above please make sure that your code runs on *homework.eecs.tufts.edu*. Please include a README file with instructions how to compile and run your code to reproduce the results of experiments. If this is nontrivial please include a script to run your code.
  (2) A short report with the results and plots as requested and a discussion with your observations from these plots. Please make sure to address the questions posed above in your discussion.

- **Please submit a hardcopy** in class.

- **Please submit electronically using provide** by 4:30 (class time). Put all the files from the previous item into a zip or tar archive (no RAR please). For example call it `myfile.zip`. Then submit using `provide comp135 pp1 myfile.zip`.

Your assignment will be graded based on the **code**, its **clarity, documentation and correctness**, the **presentation** of the results/plots, and their **discussion**.