Assignments for Course T-61.6020, part I

February 12, 2008

In each assignment you should implement the algorithm according to the instruction. For each algorithm there will be a stub either in Matlab or Python, although you don't have to use it. The stubs and the datasets can be obtained from the homepage of the course. The missing pieces of code in the stubs are marked with XXX.

In your report you should explain the algorithm (in roughly 1 page), report the results you achieve. Attach the source code and the commands you used for getting the answers.

The deadline for these homeworks is **26.3**.

1 Id3

Implement **id3.py** and try it with

```
python id3.py id3_train_data.dat id3_train_class.dat \
              id3_purge_data.dat id3_purge_class.dat
```

Use Reduced Error Pruning (Section 2.2 in [Qui87]) for pruning. What tree did you receive after the purge?

2 Naïve / Chow-Liu Tree

Implement **naive.m**, a mixture of bernoulli variables, and try it on **numbers.mat**. Note that **naive.m** takes *twist* as a parameter. The idea is to handle problems with zero probabilities. In our case, the estimate for the margin i (in some component) is then

$$\frac{twist + \sum_x x_i}{2 \times twist + N},$$

where the sum goes over x in the component and N is the number of elements in the component. Use $twist = 1$ as a parameter.

Implement **mutual_entropy.m** and **cltree.m** and try it on **numbers.mat** and **numbers_cor.mat**. Again we have bayesian twist as

$$\frac{2 \times twist + \sum_x x_i}{4 \times twist + N}$$

for map-estimate of the margin i and

$$\frac{twist + \sum_x x_j x_i}{4 \times twist + N}$$

for the map-estimate of the co-occurrence of the variables i and j.

Note that numbers datafile class variables start with 0 whereas the stub assumes that the class variables start with 1. To fix this add 1 into train_y and test_y variables.

3 k-NN

Implement **knn.m** and experiment it on **two_circles.mat** at points

$$\{0,0\}, \{-3,0\}, \{3,0\}, \{0,-3\}, \{0,3\}.$$

Use $K = 5$, the number of points used for classification, and $KM = 50$, the number of points used for learning the metric. Set also $eps = 0$ and test different $iter$ values, including $iter = 0$. Report the classification and the metric S. Why S behaves the way it behaves?

4 AdaBoost

Implement **adaboost.m**, Adaboost algorithm using perceptron components. Try the algorithm on **ada_data.mat**, (train data). Use **classify.m** to test the prediction error for the training and the test data. Vary cnt, the number of components from 1 to 1000. How the training/test error behave?

5 EM

Implement **em_bernoulli.m**, EM-algorithm for bernoulli process and try it on **bernoulli.mat** with $K = 2$. Implement **em_gaussian.m** and try it on **three_gaussian.mat** with $K = 3$.

References

[Qui87] J. Ross Quinlan. Simplifying decision trees. *International Journal of Man-Machine Studies*, 3(27):221–234, 1987.