Lab 2: Classification and Overfitting

Kurt Driessens, with the main concept supplied by Evgueni N. Smirnov
kurt.driessens@maastrichtuniversity.nl

June 26th, 2019

In this lab you will get some initial experience with a number of classification algorithms, namely decision trees and support vector machines and how to adapt their expressivity to the data at hand. You will get an introduction to the data-mining tool Weka and learn how to start interpreting its output.

1. Introduction

Given a classification problem, you need to have data that represents the problem, and of course a machine-learning environment that contains the algorithms capable of learning models from that data. In this lab you will study two well-known classification problems. You will try to find classification models for these problems using decision trees and support vector machines. The algorithms to learn these models are available in Weka, a machine-learning environment developed by the machine learning research group in Waikato, New Zealand. You will study the explorer part of Weka to learn how to call different algorithms, how to evaluate the adequacy of the learned models, and how to tune a learning algorithm to avoid under- or over-fitting.

2. Classification Problems

In this lab you are expected to build classification models for two classification problems:

- Labor-negotiation problem, and
- Soybean classification problem.

The data files for the two problems come with the Weka installation.

3. Environment

As stated above to build the desired classification models you will use Weka. Weka is a machine-learning environment that contains a large collection of machine-learning algorithms for solving real-world prediction problems. The algorithms can either be applied directly or called from your own Java code. Weka contains tools for data pre-processing, classification, regression, clustering, association rules, and visualization. It is also well-suited for developing new machine learning schemes. Weka is open source software issued under the GNU General Public License.

4. Algorithms

To build the classifiers you will use four different learning algorithms provided in Weka:

1. `zeroR` is a majority/average predictor. It assigns to each instance the classification of the majority class found in the dataset.
2. `oneR` is one-level decision tree learner. The only option of the algorithm is `bucket size` that specifies the minimum number of instances covered by each leaf of the resulting decision tree.
3. J4.8 (C4.5) is a decision-tree learner.

4. SMO is an implementation of the Support Vector Machine algorithm. (SMO stands for Sequential Minimal Optimization, and represents one way of solving the SVM constraint optimization problem.

5. Lab Tasks

A. Study the two classification problems (given in section 2) and their data files (especially how they are formatted for Weka)

B. Study the Explorer part of Weka. You will work in the "Classify" tab.

C. Apply all the zeroR, oneR and J48 learning algorithms on the two data files using the default settings. Determine the accuracy rates of the resulting classifiers using the training set and 10-fold cross validation. Is there a difference? Can you explain why? Analyze the resulting classifiers from a comprehensibility point of view.

D. Experiment with J4.8 and *error pre-pruning*. To build a decision tree using pre-pruning first set the option `unpruned` to `True`. Then, experiment with pre-pruning by changing the option minNumObj from 0 to the size of the datasets (use some step). (The option minNumObj determines the min number of training instances in the leaf nodes of the decision trees.) Estimate the accuracy rates of the resulting decision trees using the training set and 10-fold cross validation. Plot the accuracy rates based on the training set and 10-fold cross validation for minNumObj from 0 to the size of the datasets. Identify the regions of underfitting, optimality, and overfitting. Compare the accuracy rates obtained with those of J4.8 from the previous task. Please explain the results.

E. Experiment with J4.8 and *reduced error pruning*[1]. To build a decision tree using reduced error pruning you have to use the option `reducedErrorPruning` of J4.8. Experiment with reduced error pruning by changing the option `numFolds` from 2 to the size of the datasets (The option `numFolds` determines the amount of data used for reduced-error pruning. One fold is used for pruning, the rest for growing the tree.). Estimate the accuracy rates of the resulting decision trees using the training set and 10-fold cross validation. Plot the accuracy rates based on the training set and 10-fold cross validation for `numFolds` from 1 to the size of the datasets. Identify the regions of underfitting, optimality, and overfitting. Compare the accuracy rates obtained with those of J4.8 from the previous tasks.

F. Experiment with SMO and the kernel it uses. Change the kernel to the RBF kernel and see if you can get SMO to overfit or underfit by changing the width of the kernel. Note: the 'l' parameter from the slides is called 'gamma' in Weka. How do the accuracies compare to those obtained with the decision tree algorithm? If you have time left over, you can also experiment with the linear and polynomial kernels of different degrees and see if they also tend to overfit.

[1] Please do not forget to turn the option `unpruned` to `False`.