COMPX521-22A Assignment 2

Ensembles of Random Ferns

Due date: Friday, 17 June, 5pm

Random ferns (Ozuysal, Calonder, Lepetit & Fua, 2009) are classification models that were initially designed for image data. They can be viewed as a form of semi-naive Bayesian classifier. Just like in naive Bayes, a simple generative model is used for each class, and Bayes' theorem is used to obtain class probability estimates based on the per-class models. The difference is that features are not assumed to be completely conditionally independent as in naive Bayes. Rather, the full set of N predictor attributes is randomly shuffled, split into M sets of disjoint subsets of attributes of size $S = N/M$, and only these subsets are assumed to be conditionally independent from each other. For each of the individual M subsets of attributes—let's refer to the kth of them as F_k—a joint probability distribution table is estimated from the training data, using the Laplace correction to avoid the zero-frequency problem (i.e., starting all counting of event frequencies for the joint probability table from 1 instead of 0).

Assume $P(F_k|C = c_i)$, is the (class-conditional) joint probability distribution for the kth subset of attributes, for class c_i, which we can estimate from the data for class c_i by simple counting. Then, assuming (conditional) independence of the subsets F_k, we can write the conditional probability of observing an instance X given class c_i as

$$P(X|C = c_i) = \prod_{k=1}^{M} P(F_k|C = c_i).$$

For each class c_i, this can be estimated based on the corresponding collection of (class-conditional) joint probability distribution tables that have been established for that class. By using these estimates of $P(X|C = c_i)$, along with estimates of the class prior probabilities $P(C = c_i)$, we can then use Bayes' theorem to get posterior probabilities $P(C|X = x)$ for a given test instance x.

We can use the $P(C|X = x)$ to classify an instance x, but these probability estimates will depend on how the set of attributes is shuffled before it is partitioned into subsets. An obvious way to attempt to eliminate the dependency on the particular random partitioning used, is to train multiple random fern models, for different random permutations of the set of attributes, and average the final class probability estimates $P(C|X = x)$ obtained.

Your job in this assignment is to implement a `RandomFerns` classifier in WEKA, for nominal predictor attributes. (The original paper considers binary features only, but it is straightforward to compute joint probability tables for attributes with more than two values.) The implementation must extend the `RandomizableClassifier` class in WEKA, so that a seed for the random number generator can be specified by the user. This also means that an ensemble of random ferns based on different seed values can be generated automatically using the `RandomCommittee` "meta" classifier in WEKA. The implementation must also provide an additional command-line parameter `-size` that allows the user to specify the number of attributes S used for each of the M "ferns". (Note that if N is not divisible by M, then the last fern must be based on the remainder of the set of attributes.)

Benchmark datasets for the experiments will be available on the Linux file system at `/home/ml/521/assignment2` (and also, for your convenience, via `https://www.cs.waikato.ac.nz/ml/521/assignment2/`). Apply 10 runs of 10-fold cross-validation in WEKA's `Experimenter` interface to estimate predictive performance in terms of the percentage of correct classification on each of the available datasets. Generate results for $S = 2, 3, 4$. For each size, generate results for 10, 50, and 100 ensemble members in a `RandomCommittee`. Compare performance to that obtained using `NaiveBayes` and `RandomForest` in WEKA.

When analysing the experimental results in WEKA's `Experimenter`, make sure that you pay particular attention to the statistical significance of the observed differences in performance. As you know, WEKA indicates those using the symbols * and v. As you also know, by default, a paired corrected resampled t-test (Nadeau & Bengio, 2003) will be performed for significance testing based on the 2×100 performance estimates for the two scheme-dataset pairs involved.

Once the experiments are done, write a report on your findings. Please submit your report as a PDF file. Make a `.zip` or `.tgz` archive with your code (sources and compiled classes) and the report and submit it on Moodle. Once extracted, it must be possible to run your implementations by adding the top level of the extracted directory structure to the Java `CLASSPATH` (assuming the main WEKA 3.8 `weka.jar` file is already in the `CLASSPATH`).

Grades will be based on the quality of the report and the program. The report should be approximately three pages long in standard LaTeX report format. It should contain an introduction, a section explaining the algorithm that was implemented, a section with results and a corresponding discussion, and some conclusions. Relevant references to the literature should also be included (e.g., the paper describing the algorithm, a reference for WEKA, etc.). Results should be presented as tables or graphs when appropriate, with corresponding captions and references in the text that discusses them.

Make sure you start the assignment early. There will be no extensions, except for documented medical reasons.

References

Nadeau, C. & Bengio, Y. (2003). Inference for the generalization error. *Machine Learning*, *52(3)*, 239–281.

Ozuysal, M., Calonder, M., Lepetit, V. & Fua, P. (2009). Fast keypoint recognition using random ferns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *32(3)*, 448–461.