The Weka solution to the 2004 KDD Cup

Bernhard Pfahringer
Department of Computer Science
University of Waikato
Hamilton, New Zealand
bernhard@cs.waikato.ac.nz

ABSTRACT

This short communication describes the Weka solution for the 2004 KDD cup problems, mostly focusing on the bioinformatics problem, where this approach performed best among all submissions. Differences were not significant for the best three submissions, though. The identical setup trained for the physics problem achieved rank nineteen, which is still reasonable.

1. INTRODUCTION

The same final setup achieved overall first place on the protein homology prediction problem, and ranked nineteen on the quantum physics problem. Both problems are described in a lot more detail in the introductory paper in this journal written by the KDD cup organizers. The homology prediction problem was slightly non-standard, as the attributes actually describe the outcomes of various methods trying to estimate the similarity between pairs of protein sequences. Unique identifiers for both sequences were available, and the scoring was done in a one-to-many fashion, collecting all results for one sequence into a bag. This approach can be viewed as a variation of multi-instance learning [1]. Even though an extension for multi-instance learning is available for Weka [2], it was not used as it currently only supports standard multi-instance learning and not this slightly different setting.

2. DATA QUALITY

Bearing in mind that the homology problem is an extremely skewed two-class prediction problem with an apriori probability of 0.88% for the minority class, it may seem amazing that such good results as reported below are possible at all. A possible explanation is the fact that this is almost a meta-learning problem, as the inputs being used for prediction are already outputs of other sophisticated predictors including learning approaches like hidden Markov models. Therefore these attributes are of rather high quality, as can also be seen from Figures 1, 2, and 3 which depict plots of all attributes. One can clearly see some attributes with almost bell-shaped blue bumps representing the majority class, and well-separated long red tails representing the minority class. Attributes 3, 10, and 23 in Figure 3 are typical examples for that pattern. The remaining two figures show similar shapes for many of the other attributes.

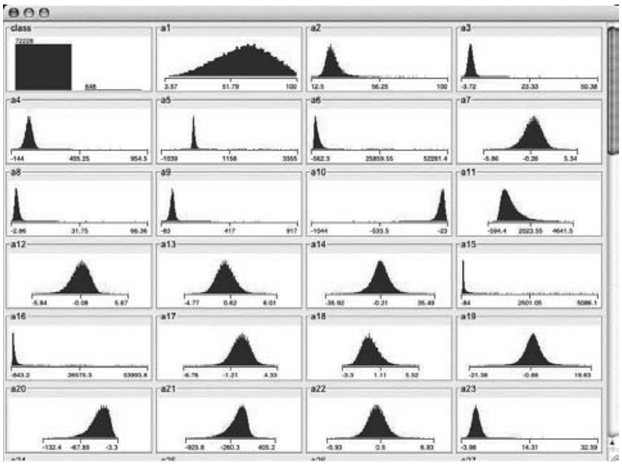

Figure 1: Plots for the first third of the attributes.

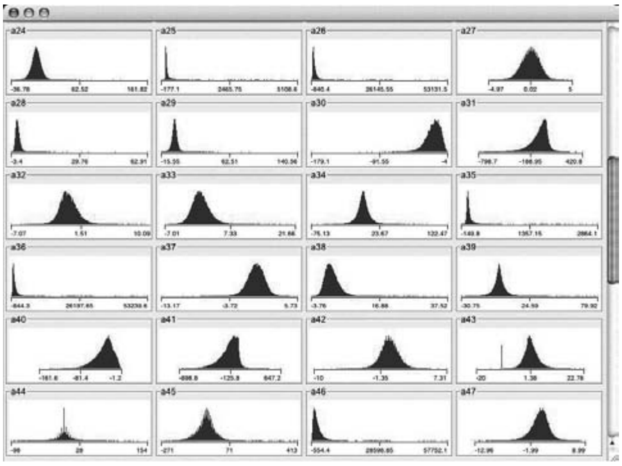

Figure 2: Plots for the second third of the attributes.

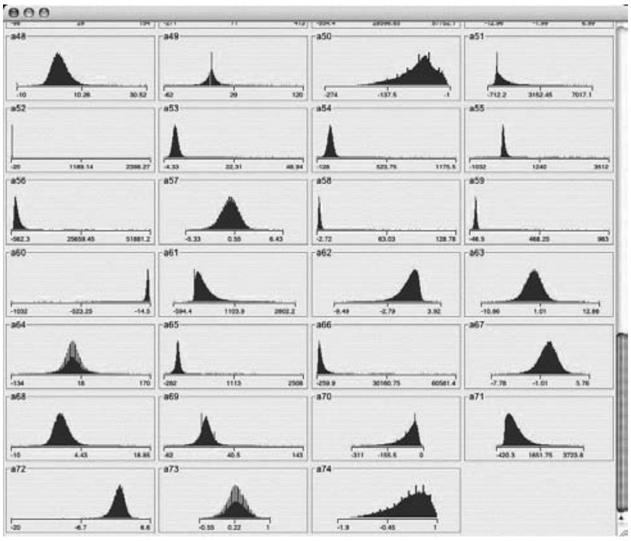

Figure 3: Plots for the last third of the attributes.

3. INITIAL ATTEMPTS

The size of the datasets is just about within the range of datasets that can reasonably be handled by Weka on a current PC with about half a gigabyte of memory. Therefore we decided to employ only two-fold cross-validation instead of the more commonly used ten folds. Given the size of the datasets one can expect reasonable estimates from half the data already. For homology prediction, evaluation was to use the following four performance criteria:

- Top1: the fraction of blocks or bags with a homologous sequence ranked as top1.
- RMSE: the root mean squared error over all predicted probabilities.
- RKL: the average rank of the lowest ranked homologous sequence inside each block or bag
- APR: the average over all blocks or bags of the average precision within each block or bag.

Note that only RMSE is actually sensitive to the absolute magnitude of each prediction, whereas for all other criteria these absolute magnitudes have no influence on the outcome, only the relative ranking of examples that they generate is important.

Unfortunately for this experimental setup Weka currently only supports RMSE directly, and RMSE did not seem to work well as a proxy for the other measures, so standard predictive accuracy was used as a proxy in all the cross-validation experiments.

As Weka supports a large number of different classification algorithms out of the box, all the standard ones were applied and compared in a first round. Not surprisingly, the following three algorithms enjoyed a slight edge over the rest:

- A linear support vector machine with a logistic model fitted to the raw outputs for improved probabilities.
- A boosting ensemble of 10 unpruned decision trees
- 100000 or more random rules [3]

	Top1	RMSE	RKL	APR
Boost10	0.79333	0.03690	500.68	0.68582
linSVM	0.88667	0.03699	64.41	0.82581
10^5 RR	0.89333	0.04142	53.77	0.83733
Ensemble	0.90667	0.03833	52.45	0.84118

Table 1: Performance criteria breakdown

Accuracy-wise, all results looked amazingly good given the extreme skewness of the class distribution. Variations of tuning parameters did not seem to lead to significant improvements for single classifier results. On the contrary, boosting more trees for instance seemed to actually overfit, at least in cross-validations on the training set. Mostly for efficiency reasons, at that stage no elaborate tuning of the support vector machine was attempted. Due to some optimizations in the Weka implementation of support vector machines training and evaluation of linear models is at least one order of magnitude faster than using other polynomial kernels or Gaussian kernels.

As the raw attributes supplied seemed to do such a good job, and as higher-order polynomial kernels seemed to do worse than a linear kernel, no further attribute construction or selection was attempted.

4. FINAL SUBMISSION

As ensembles usually are more robust than single classifiers, and as well predicted probabilities would work well across all the performance criteria described above, a specialized ensemble was constructed for the final submission. The ensemble consisted of the three best classifiers that were identified in the initial round of experiments, as listed above. But as standard boosting usually does not produce good probability estimates, the following non-standard voting procedure was employed for the final ensemble:

```
IF   SVM and RandomRules agree on predicted class
THEN average their predicted probabilities
ELSE use the booster as a tie-breaker and use the
     probability predicted by the winner
     (either SVM or RandomRules)
```

Table 1 shows the performance across the four criteria for the three single classifiers as well as the ensemble. The latter is better for all criteria that depend only on good rankings, but interestingly is outperformed by the booster in terms of RMSE. Without access to the actual test-labels the current hypothesis to explain this counter-intuitive finding is based on the skewness of the class distribution: as boosters quite often predict hard zeros or ones, they may pay a higher penalty for errors, but none for correct predictions, and in this case more than 99% of the predictions are correct. The other two schemes mostly predict close to zero or one, and therefore incur some small penalty in terms of RMSE for almost every example. In terms of the other three criteria boosting is clearly outperformed, which provides some a posteriori justification for the voting scheme described above.

5. POST MORTEM

Since the actual Cup the submission web-site has reopened to allow for further experiments. Going back to the original attempts, I had a second look at support vector machines. For efficiency reasons this time not even two-fold

Rank	1	2	3	4	5	6	7
SVM	a53	a58	a3	a60	a59	a8	a45
RR	a53	a63	a55	a58	a3	a34	a54
DS	a53	a55	a58	a59	a60	a54	a3

Table 2: Attribute rankings

cross-validation was employed, but instead randomized train-test splits with only 10% of the data being used for training were used. This allowed for more extensive parameter tuning which in turn revealed that a slightly higher-order polynomial kernel of $degree \sim 1.5$ as well as a Gaussian kernel of appropriate non-default bandwidth (for Weka this is achieved with the option setting $-G$ 0.44) actually work slightly better than a linear kernel. Also, larger ensemble sizes, i.e. voting more than three classifiers, seems to be more robust, but the differences are rather small on an absolute scale. But given the edge a non-linear kernel enjoys, some attribute engineering might help to further improve the performance of classifiers other than a support vector machine.

From an application point of view, it is always interesting to try to extract some knowledge from the induced classifiers. For both the linear support vector machine as well as the random rules it is straightforward to at least generate attribute rankings, thus revealing the relative importance and merit of each single attribute. Boosted ensembles of trees are harder to inspect and understand, therefore an boosted ensemble of simple decision stumps was generated as a kind of proxy for the full-trees ensemble. Again, the ensemble of stumps provides an attribute ranking. Table 2 lists the top seven attributes for each of the three classifiers. There is considerable agreement on what are the top attributes between the three classifiers. This might be useful information for further data engineering in the domain.

5.1 Physics task

As already mentioned above, the exact same setup applied to the Physics task achieved only rank nineteen. Looking at what the winners for that task have done to be significantly better than all other submissions, it is clear that some kind of data preprocessing was essential for success. This data has a lot of missing values, and for some attributes strong patterns of correlated missing values are present. This suggests that these values are not actually randomly missing, but more like measurements that are not applicable in certain settings, and which can only be represented as "missing" in a rigid attribute-value pair representation. Transforming these attributes, splitting up the data into disjoint subsets, and more extensive tuning of especially the support vector machines might have given better results.

6. FUTURE WORK

There are a few potential directions to approach this problem using Weka which have not been tried so far:

- Optimizing separately for every performance criterion: sofar all classifiers are trained to achieve optimal accuracy with reasonable probabilities, and that seems to achieve good overall performance. Better performance could be expected from direct optimization with regard to each specific criterion. Unfortunately only one is currently supported in Weka, and even though Weka has an excellent modular architecture making it easy to add new classifiers or data preprocessing schemes, this is not the case when it comes to adding new performance criteria.
- Actually exploiting the multi-instance like nature of the problem: again, this is not currently supported in Weka, and not even in the multi-instance learning extension of Weka. One approach tried by other contestants was a kind of nearest neighbor computation over bags.
- Approaching the problem as a "detection of outliers" task: as the number of matches is less than 1%, and as the majority class seems to generate close to perfect normal distribution for a considerable number of attributes (cf. Figure 1), methods for detecting outliers could potentially separate the matches from the non-matches.

7. LESSONS LEARNED

From my point of view, this work has simply reinforced two major lessons which are true for data mining in general:

1. Ensembles are more robust and stable in terms of their predictive performance.
2. Data engineering is essential for success. Luckily, for this year's KDD Cup data, the organizers have already done that by supplying excellent attributes for the homology prediction task.

In terms of Weka the lessons would be that quite a few bits and pieces inside Weka could and should be improved to both show better scaling behavior and also to allow for easier integration of additional functionality other than new classifiers. Still, this competition has shown that from its origin as an educational-only tool Weka has matured into an open-source Java-based tool that is at least competitive for medium-sized data mining tasks.

8. ACKNOWLEDGMENTS

I would like to thank the two organizers of the KDD Cup and their web-support person for providing two challenging and exciting problems, and for running the competition in a very smooth and professional way. Thanks are also due to the numerous Weka supporters for their ongoing efforts in building, extending and improving Weka.

9. REFERENCES

[1] T.G. Dietterich, R.H. Lathrop, and T. Lozano-Perez. *Solving the multiple-instance problem with axis-parallel rectangles.* In: Artificial Intelligence, 89(1-2), pp. 31-71, 1997.

[2] E. Frank and X. Xu. *Applying Propositional Learning Algorithms to Multi-instance data.* Working Paper 06/03, Computer Science, University of Waikato, 2003.

[3] B. Pfahringer, G. Holmes, and C. Wang. *Millions of Random Rules.* In: Workshop on Advances in Inductive Rule Learning, 15th European Conference on Machine Learning (ECML), Pisa, 2004