Meta-Learning for Stacked Classification

Alexander K. Seewald

Austrian Research Institute for Artificial Intelligence, Schottengasse 3
A-1010 Wien, Austria; alexsee@oefai.at

Abstract. In this paper we describe new experiments with the ensemble learning method Stacking. The central question in these experiments was whether meta-learning methods can be used to accurately predict various aspects of Stacking's behaviour. The resulting contributions of this paper are two-fold: When learning to predict the accuracy of stacked classifiers, we found that the single most important feature is the accuracy of the best base classifier. A simple linear model involving just this feature turns out to be surprisingly accurate. When learning to predict significant differences between Stacking and three common meta-classification methods, we have found simple models, all but one of which are based on single features which can be efficiently computed directly from the dataset. For one of these models, we were able to offer an interpretation. These models may ultimately be used to decide in advance which meta-classification scheme to use on a given dataset, since neither of them is always the best choice. Furthermore, aiming to understand these models can lead to new insights into Stacking's behaviour.

1 Introduction

Meta-learning focusses on predicting the right algorithm for a particular problem based on characteristics of the dataset [3] or based on the performance of other, simpler learning algorithms [6]. Here we are concerned with meta-learning of *meta-classification schemes*. Stacking can be considered the best-known such scheme and was introduced in [11]. We take a more general view of meta-learning and use it to predict two aspects of Stacking's behaviour: accuracy as estimated via ten-fold crossvalidation; and also significant differences vs. other common meta-classification schemes. We use Stacking in the extension proposed in [10].

2 Experimental setup

In our experiments, we used twenty-six datasets from the UCI machine learning repository [2]. Details can be found in [9]. We used Stacking with all of the following seven base classifiers for our experiments, which were chosen in an attempt to maximize diversity. All algorithms were taken from the Waikato Environment for Knowledge Analysis (WEKA[1]), Version 3-1-8.

- DecisionTable: a decision table learner.
- IBk: the IBk instance-based learner using K=1 nearest neighbors.
- J48: a Java port of C4.5 Release 8 [7]
- KernelDensity: a simple kernel density classifier.

[1] The Java source code of WEKA has been made available at www.cs.waikato.ac.nz.

- KStar: the K* instance-based learner [4], using all nearest neighbors.
- MLR: a multi-class learner based on linear regression, which separates each class from all other classes by linear discrimination (*Multi-response Linear Regression*)
- NaiveBayes: the Naive Bayes classifier using kernel density estimation (-K)

We used the following four meta-classification schemes.

- Stacking is the stacking algorithm as implemented in WEKA, which follows [10]. It constructs the meta dataset by adding the entire predicted class probability distribution instead of only the most likely class. We used MLR as the level 1 learner.
- X-Val chooses the best base classifier on each fold by an internal ten-fold CV. This is just the selection by cross-validation we mentioned in the beginning.
- Voting is a straight-forward adaptation of voting for distribution classifiers, i.e. the mean class distribution of all classifiers is calculated. It is the only scheme which does not use an expensive internal cross-validation.
- Grading is an implementation of the grading algorithm evaluated in [8] which uses IBk ($K = 10$) as meta-classifier.

We used seventeen dataset-related features which characterize the dataset, inspired by [3]. A reference implementation is available from the author upon request.

- *Inst*, the number of examples.
- *log(Inst)* which is the natural logarithm of *Inst*.
- *Classes*, the number of classes.
- *Attrs*, the number of attributes (excluding the class)
- *PropNomAttrs*, number of nominal attributes as a proportion of *NumAttrs*.
- *PropContAttrs*, number of numeric attributes as a proportion of *NumAttrs*.
- *PropBinAttrs*, number of binary-valued attributes as a proportion of *NumAttrs*.
- *ClassEntropy*, the entropy of the class attribute.
- *AttrEntropy*, the entropy of all attributes.
- *MutualEntropy*, the mutual entropy of class and attributes.
- *EquivAttrs*, the equivalent number of attributes, $\frac{ClassEntropy}{MutualEntropy}$
- *RelEquivAttrs*, $\frac{EquivAttrs}{Attrs}$
- *S/N*, the signal-to-noise ratio.
- *MeanAbsCorr*, the mean absolute correlation over all pairs of numeric attributes.
- *MeanAbsSkew*, the mean absolute skew of all numeric attributes.
- *MeanAbsKurtosis*, the mean absolute kurtosis of all numeric attributes.
- *defAcc*, the default accuracy, i.e. the proportion of the most common class.

Additionally, we used the accuracies of our seven base-learners as features. We also calculated standard statistical features of this set of seven accuracies. Furthermore, we used the same statistical features over pairwise base classifier κ-statistics[2].

- 7 accuracies, one for each base classifier (*DT, IBk-K1, J48, KD, KStar, MLR, NB-K*)
- 8 statistical features describing the set of accuracy values (*MinAcc, MaxAcc, MeanAcc, StDevAcc, SkewAcc, SkewAcc*^{2}*, KurtosisAcc, relRangeAcc* $= \frac{MaxAcc-MinAcc}{StDevAcc}$)
- Eight statistical features describing the set of all pairwise κ-statistics between base classifiers (*MinK, MaxK, MeanK, StDevK, SkewK, SkewK*^{2}*, KurtosisK, relRangeK*)
- *relMeanAcc*=$\frac{AvgAcc}{defAcc}$, the ratio of average accuracy to default accuracy.

The above features were computed both on predictions estimated from the full data set (training set accuracy and diversity) and on predictions estimated via tenfold crossvalidation. For meta-learning of significant differences, we only used the latter set because it consistently offered better estimates during the first task. This also simplified the experimental evaluation. All statistical differences for meta-learning were computed via a t-Test with α=99%.

[2] 1.0 stands for identical predictions between two learners while 0.0 represents random correlations. A negative value signifies systematic disagreement, see [5].

3 Estimating Stacking's Accuracy

This section is concerned with predicting the accuracy of Stacking. In order to obtain a reasonable estimate, a ten-fold CV was used for accuracy estimation. We first investigated the simplest models: based on only a single feature. Thus, we assumed linear relationships between each feature and the accuracy of our stacked classifier and characterized this relation by statistical correlation coefficients and mean absolute errors (MAE). Afterwards, we considered more complex and non-linear models obtained by various regression algorithms from machine learning.

We computed statistical correlation coefficients and mean absolute errors (MAE) for all our features, always versus the accuracy of the stacked classifiers. Space restrictions prevent us from showing detailed results, which can be found in [9].

Correlations and MAEs were determined for all meta-data (**All**) and also via leave-one-out crossvalidation (**CV**). In the former, this estimate was based on the output of one linear regression model computed from all meta-examples. In the latter case, the estimate was based on twenty-six linear models which were trained using all but one meta-example and tested on the last one. This latter case is a more reliable indicator of model performance on unseen data than the former.

In the case of base-classifier related features, we have an additional dimension: we can estimate the base classifier accuracies on the full dataset (**AllT, CV T**, i.e. training set accuracies) or via tenfold crossvalidation (**All, CV**), yielding two different set of features. Since Stacking uses CV internally, we expect **All** and **CV** to be better predictors for stacked accuracy. This is indeed the case – a single feature, *MaxAcc*, already yields excellent results. However, computing a crossvalidation on the original dataset comes with a non-negligible computational cost. A computational cost reduction by an order of magnitude could be obtained by using training set output to compute our features – which motivates **AllT** and **CV T**. As expected, in this case we get less good but still acceptable results for best single feature, *MeanAcc*.

As should be expected from a high-bias linear model, all base-classifier related features show a graceful degradation from **All** to **CV**. We were surprised to note that this is not always true for the dataset-related features - about half of the features have a negative correlation for **CV** whose absolute value is higher than the positive correlation for **All**. This higher negative correlation can unfortunately not be used to predict stacked accuracy[3] and is always coupled to a large MAE. It seems that a lot of the dataset-related features are not relevant to this task or that a one-dimensional linear model is not appropriate to find a relevant relation.

In order to test how we may improve our results by using multiple features, we resorted to using standard machine-learning approaches for regression on our meta-dataset. We created one meta-dataset with accuracy estimation via training set (*MetaTrain*) and one estimated via tenfold CV (*MetaCV*). The dataset-related features were included in both cases. We evaluated linear regression, LWR (*locally weighted regression*), model trees, regression trees, KStar and IBk instance based learners at the meta-level. Linear regression and model trees proved superior[4]. However,

[3] The maximum negative correlation appears in feature *defAcc* (-0.94; **CV**) This correlation is based on twenty-six different models, one per leave-one-out training fold. All data would have to be used to determine the final regression line, but then this result can no longer be validated and seems certainly too optimistic.

[4] Both were always best by highest correlation and lowest MAE.

we were still unable to find any model which performed better than the best linear model based on a single feature.

Concluding, features derived from classifiers seem to be more relevant in the context of predicting accuracy than those derived directly from the datasets, which was also found in [1]. For example, the formula $StAcc = 1.074 * MaxAcc - 0.082$ predicts Stacking's accuracy with a correlation of 0.96 and a MAE of 0.022. Notice that although it seems at first glance that Stacking performs slightly worse than the best component classifier, this view is biased: *MaxAcc*, i.e. the best base classifier by hindsight, is a less fair comparison than accuracy of X-Val since its decision is based on all available data while X-Val and Stacking only see the training data from the leave-one-out CV, i.e. all but one meta-instance. Notice also that while computing *MaxAcc* leaves us with a lot of data which could be used directly by Stacking, this would only enable us to compute the training set accuracy for Stacking and not the ten-fold cv estimate we used here.

Given our results, it is surprising that other meta-learning approaches have not considered that quite simple models may suffice, but instead rely on complex models whose interpretation may be quite difficult.

4 Meta-Learning of Significant Differences

This section is concerned with predicting significant differences between Stacking and three other meta-classification schemes. For each of Stacking vs. Voting, Stacking vs. Grading and Stacking vs. X-Val, we generated a separate meta-dataset consisting of all dataset-related and classifier-related features[5] followed by a binary class variable, being 1 if Stacking is significantly better than the other scheme and 0 otherwise. In case there is no significant difference, we removed the respective example from the meta-dataset, under the premise that in this case we can consider both variants to be equivalent and thus judge either answer to be correct.

On these meta-datasets, we evaluated a number of standard machine learning algorithms available in WEKA[6] via leave-one-out crossvalidation. We only discuss the best models which in most cases seem to be rather simple and based on single attributes only, hinting that they may be robust. In one case, insight into the workings of both meta-classification schemes suggests an interpretation.

For Stacking vs. Voting, there are twelve datasets without significant differences. After removing them from our meta-dataset, we have fourteen instances, seven with class=1, seven with class=0. The baseline accuracy is thus 50%. Here, IBk is the best meta-learner with an accuracy of 92.86% and a single error for *vote*. A cross-validation using only seven folds produces the exact same result.

When removing the base-classifier dependent features, IBk is still the best classifier with an additional error on *labor*, the smallest dataset. In this case MLR, another high-bias and global learner, is equally good. So we may tentatively conclude that for this meta-dataset, there seems to be no single feature which can predict the significant differences as good as a combination of all features.

For Stacking vs. Grading, there are again twelve datasets on which there are no significant differences. After removing them from our meta-dataset, we have fourteen

[5] Because of the much better results in predicting stacked classifier accuracy and also to simplify our experiments, we only considered those classifier features estimated via CV.

[6] All base learners plus 1R and DecisionStump.

instances whose classes are again equally distributed. Thus the baseline accuracy is also 50%. Here, J48 is the best choice with 92.86% accuracy and only a single error on the smallest dataset, *labor*. The training set model is based on a single attribute, *PropNomAttr*. In all fourteen folds but two there is the same model[7], which also appears as the training set model. In the two other folds, the same attribute appears in the same formula with 0.65 and 0.695652 resp. as value on the right side. It seems that the proportion of nominal attributes plays a role on the performance between Stacking and Grading: in case there about $\frac{2}{3}$ or less of the attributes are nominal, Stacking works significantly better than Grading.

A smaller proportion of nominal attributes makes learning harder for the base-learners, since most of them are better equipped to handle nominal data. Stacking seems to be able to compensate for this, since its meta-level data is independent of the base-level data[8] and is processed by MLR which is among all base learners best equipped to handle numeric data. However, Grading seems to be unable to compensate for this since its meta-level data contains just the base-level attributes. Thus its meta learner IBk can be expected to be susceptible in the same way as the base learners.

For Stacking vs. X-Val, seventeen examples offer no significant differences. Only nine examples remain for our experiments, the baseline accuracy is already 66.7%. Interestingly in this case the best model is from DecisionStump which learns a single J48 node, obtaining 88.9% accuracy, corresponding to a single error on dataset *balance-scale*. It seems J48 is prone to overfitting on this meta-dataset. The training set model[9] is based on *MeanAbsSkew* and appears in seven folds. Once the same model appears with value 0.53 instead of 0.31. Once a model based on $numClasses <= 13 : class = 1$ appears. The same overall accuracy is also obtained in a six-fold cross-validation.

5 Related Research

Up to now there is no research aiming to either predict the accuracy of meta-classification schemes or to predict which meta-classification scheme to use for a given dataset. In this paper we have investigated both tasks and found them to work quite well.

6 Conclusion

In this paper we have investigated the use of machine learning techniques in the context of meta-learning both to predict stacked classifier accuracy and significant differences between Stacking and three other meta-classification schemes. We used both dataset-related and base-classifier related features in our tasks.

In the context of predicting classifier accuracy, we found that classifier-related features, namely some of those derived from accuracy, are excellently suited to this task, as have others, [1, 6]. As feature, the accuracy of the best component classifier in the ensemble is able to predict the accuracy of the stacked classifier quite well. Other meta-learning approaches seem not to take into account that such simple models may be competitive to more complex models, but far much easier to understand.

[7] $IF\ PropNomAttr <= 0.684211\ THEN\ class = 1\ OTHERWISE\ class = 0$

[8] Meta-level data for Stacking = class probability distributions from all base learners.

[9] $IF\ (MeanAbsSkew <= 0.31\ OR\ missing)\ THEN\ class = 0\ OTHERWISE\ class = 1$

In the second part of the paper we investigated the prediction of significant differences between stacking and other meta-classification schemes. In this case we found that features derived directly from the dataset were usually better suited. For the model which predicts significant differences between Grading and Stacking, intimate knowledge of the inner workings of both schemes have enabled us to formulate a tentative explanation of the learned model.

At last we have found that there is no single best meta-classifier for predicting significant differences – a variety of machine learning algorithms had to be evaluated for best results. Although most of our best models were based on single features, it seems that no single learning algorithm is able to find all of them. This hints that pairwise learning problems have quite different properties, which may explain why meta-learning is usually so hard.

Acknowledgements

This research is supported by the Austrian *Fonds zur Förderung der Wissenschaftlichen Forschung (FWF)* under grant no. P12645-INF. This research was also partially supported by the ESPRIT LTR project METAL (26.357). The Austrian Research Institute for Artificial Intelligence is supported by the Austrian Federal Ministry of Education, Science and Culture. We would like to thank Johannes Fürnkranz for valuable comments.

References

1. Bensusan, H., Kalouis, A.: Estimating the Predictive Accuracy of a Classifier. In Proceedings of the twelveth European Conference on Machine Learning (2001), Freiburg, Germany, 25–36. Springer Verlag.
2. Blake, C. L., Merz, C. J: UCI repository of machine learning databases. `http://www.ics.uci.edu/~mlearn/MLRepository.html` (1998). Department of Information and Computer Science, University of California at Irvine, Irvine CA.
3. Brazdil, P. B., Gama, J., & Henery, B. Characterizing the applicability of classification algorithms using meta-level learning. *Proceedings of the 7th European Conference on Machine Learning (ECML-94)* (83–102). Catania, Italy: Springer-Verlag.
4. Cleary, J. G., Trigg, L. E: K*: An instance-based learner using an entropic distance measure. Proc. 12th International Conference on Machine Learning (1995) 108–114, Lake Tahoe, CA.
5. Dietterich, T. G: Ensemble methods in machine learning. In Kittler, J., Roli, F., First International Workshop on Multiple Classifier Systems (2000) 1–15. Springer-Verlag.
6. Pfahringer, B., Bensusan, H., & Giraud-Carrier, C. (2000). Meta-learning by landmarking various learning algorithms. *Proceedings of the 17th International Conference on Machine Learning (ICML-2000)*. Stanford, CA.
7. Quinlan, J. R: C4.5: Programs for Machine Learning (1993). Morgan Kaufmann, San Mateo, CA.
8. Seewald A.K., Fürnkranz J.: An Evaluation of Grading Classifiers, in Hoffmann F. et al. (eds.), Advances in Intelligent Data Analysis, Proc. 4th International Conference, IDA 2001, Springer, 115–124.
9. Seewald A.K.: Meta-Learning for Stacked Classification (ext.vers.). Technical Report, Austrian Research Institute for Artificial Intelligence, Vienna, TR-2002-05, 2002.
10. Ting, K. M., Witten, I. H: Issues in stacked generalization. Journal of Artificial Intelligence Research *10* (1999) 271–289.
11. Wolpert, D. H: Stacked generalization. Neural Networks *5(2)* (1992) 241–260.