Improving classification of J48 algorithm using bagging,boosting and blending ensemble methods on SONAR dataset using WEKA

Aakash Tiwari, Aditya Prakash

***Abstract*— Many researchers have investigated the technique of combining the predictions of multiple classifiers to produce a single classifier. The resulting classifiers is more accurate than any individual classifier. This paper investigates the ability of ensemble methods to improve the efficiency of basic J48 machine learning algorithm. Ensemble algorithms like Bagging, Boosting and Blending improved the discrimination between sonar signals bounced off a roughly cylindrical rock in the SONAR dataset. The ranking and standard deviation functionalities provided by the WEKA experimenter helps to determine the effectiveness of a classifier model.**

***Index Terms*— WEKA,SONAR,Bagging,Boosting,Blending.**

I. INTRODUCTION

Decision tree is one of the classifying and predicting data mining techniques, belonging to inductive learning and supervised knowledge mining. It can generate easy-to-interpret If-Then decision rule, it has become the most widely
applied technique among numerous classification methods [1]. Decision tree is a tree diagram based method, the node on the top of its tree structure is a root node and nodes in the bottom are leaf nodes. Target class attribute is given to each leaf node.

From root node to every leaf node, there is a path made of multiple internal nodes with attributes. This path generates rule required for classifying unknown data. Moreover, most of decision tree algorithms contain two-stage task, i.e., tree building and tree pruning.

In tree building stage, a decision tree algorithm can use its unique approach (function) to select the best attribute, so as to split training data set. The final situation of this stage will be that data contained in the split training subset belong to only one certain target class. Recursion and repetition upon attribute selecting and set splitting will fulfill the construction of decision tree root node and internal nodes. On the other hand, some special data in training data set may lead to improper branch on decision tree structure, which is called over-fitting. Therefore, after building a decision tree, it has to be pruned to remove improper branches, so as to enhance decision tree model accuracy in predicting new data . Among developed decision tree algorithms, the commonly used ones include ID3 [2], C4.5 [3], CART [4] and CHAID [5]. C4.5 was developed from ID3 (Iterative Dichotomiser 3) algorithm, it uses information theory and inductive learning method to construct decision tree. C4.5 improves ID3, which cannot process continuous numeric problem. J48 is an open source Java implementation of the C4.5 algorithm in the WEKA data mining tool.

II. ENSEMBLE METHODS

1. BOOSTING - Boosting is an ensemble method that starts out with a base classifier that is prepared on the training data. A second classifier is then created behind it to focus on the instances in the training data that the first classifier got wrong. The process continues to add classifiers until a limit is reached in the number of models or accuracy.

2. BAGGING - Bagging (Bootstrap Aggregating) is an ensemble method that creates separate samples of the training dataset and creates a classifier for each sample. The results of these multiple classifiers are then combined (such as averaged or majority voting). The trick is that each sample of the training dataset is different, giving each classifier that is trained, a subtly different focus and perspective on the problem.

3. BLENDING - Blending is an ensemble method where multiple different algorithms are prepared on the training data and a meta classifier is prepared that learns how to take the predictions of each classifier and make accurate predictions on unseen data..

III. WEKA INBUILT ENSEMBLES

A. Boosting

ADABOOST M1 is class for boosting a nominal class classifier using the Adaboost M1[6] method. Only nominal class problems can be tackled. Often dramatically improves performance, but sometimes overfits. AdaBoost M1 is adaptive in the sense that subsequent weak learners are tweaked in favor of those instances misclassified by previous classifiers.

Path - weka.classifiers.meta.AdaBoostM1

STEPS:

1.Click "*Add new…*" in the "*Algorithms*" section.

2.Click the "*Choose*" button.

3.Click "*AdaBoostM1*" under the "*meta*" selection.

Manuscript received September 18, 2014.

Aakash Tiwari, Computer Engineering FRCRCE, Bandra, Mumbai University, India

Aditya Prakash, I.T Engineering, FRCRCE, Bandra, Mumbai University, India

4.Click the "*Choose*" button for the "*classifier*" and select "*J48*" under the "*tree*" section and click the "*choose*" button.
5.Click the "*OK*" button on the "*AdaBoostM1*" configuration.

B. Bagging

BAGGING - Bagging [7] is a class for bagging a classifier to reduce variance. It can do classification and regression depending on the base learner. We can choose here the bag size -- this is saying a bag size of 100%, which is going to sample the training set to get another set the same size, but it's going to sample "with replacement". That means we're going to get different sets of the same size every time we sample, but each set might contain repeats of the original training. Weka has an inbuilt bagging classifier.
Path - weka.classifiers.meta.Bagging

STEPS:
1.Click "*Add new…*" in the "*Algorithms*" section.
2.Click the "*Choose*" button.
3.Click "*Bagging*" under the "meta" selection.
4.Click the "*Choose*" button for the "*classifier*" and select "*J48*" under the "*tree*" section and click the "*choose*" button.
5.Click the "*OK*" button on the "*Bagging*" configuration.

C. Blending

STACKING - Stacking [8] combines several classifiers using the stacking method. It can do classification or regression. You can choose different meta-classifiers here, and the number of stacking folds. We can choose different classifiers; different level-0 classifiers, and a different meta-classifier. In order to create multiple level-0 models, we can specify a meta-classifier as the level-0 model.
Path - weka.classifiers.meta.Stacking.

STEPS:
1.Click "*Add new…*" in the "*Algorithms*" section.
2.Click the "*Choose*" button.
3.Click "*Stacking*" under the "*meta*" selection.
4.Click the "*Choose*" button for the "*metaClassifier* and select "*Logistic*" under the "*function*" section and click the "*choose*" button.
5.Click the value (algorithm name, it's actually a button) for the "*classifiers*".
6.Click "*ZeroR*" and click the "*Delete*" button.
7.Click the "*Choose*" button for the "*classifier*" and select "*J48*" under the "*tree*" section and click the "*Close*" button.
8.Click the "*Choose*" button for the "*classifier*" and select "*IBk*" under the "*lazy*" section and click the "*Close*" button.
9.Click the "*X*" to close the algorithm chooser.
10.Click the "*OK*" button on the "*Bagging*" configuration.

IV. DATASET

This is the data set used by Gorman and Sejnowski [9] in their study of the classification of sonar signals using a neural network. The task is to train a network to discriminate between sonar signals bounced off a metal cylinder and those bounced off a roughly cylindrical rock.

PROBLEM DESCRIPTION: The dataset contains 111 patterns obtained by bouncing sonar signals off a metal cylinder at various angles and under various conditions .It contains 97 patterns obtained from rocks under similar conditions. The transmitted sonar signal is a frequency-modulated chirp, rising in frequency. The data set contains signals obtained from a variety of different aspect angles, spanning 90Mdegrees for the cylinder and 180 degrees for the rock. Each pattern is a set of 60 numbers in the range 0.0 to 1.0. Each number represents the energy within a particular frequency band, integrated over a certain period of time. The integration aperture for higher frequencies occur later in time, since these frequencies are transmitted later during the chirp.

V. EXPERIMENT

The SONAR dataset is loaded in the WEKA experimenter so as to classify it into class. The main goal of this experiment is to increase the efficiency of J48 algorithm using ensemble methods. The ADABOOST M1 is used as a boosting ensemble, BAGGING is used as a bagging ensemble.We will add Stacking with two classifiers (J48 and IBk) and use Logistic Regression as the meta classifier. The J48 and IBk[10] (k-nearest neighbour) are very different algorithms and we want to include algorithms in our blend that are "good" (can make meaningful predictions on the problem) and varied (have a different perspective on the problem and in turn make different useful predictions). Logistic Regression[11] is a good reliable and simple method to learn how to combine the predictions from these two methods and is well suited to this binary classification problem as it produces binary outputs itself. The individual J48 classifier alongwith its ensemble classifiers are run at the same time. The result is analyzed for further research on the ensembles. The Weka Experimenter allows you to design your own experiments of running algorithms on datasets, run the experiments and analyze the results. The first thing we want to know is which algorithm was the best. We can do that by ranking the algorithms by the number of times a given algorithm beat the other algorithms. 1. Click the "*Select*" button for the "*Test base*" and choose "*Ranking*". 2. Now Click the "*Perform test*" button. The ranking table shows the number of statistically significant wins each algorithm has had against all other algorithms on the dataset.

```
Tester:     weka.experiment.PairedCorrectedTTester
Analysing:  Percent_correct
Datasets:   1
Resultsets: 4
Confidence: 0.05 (two tailed)
Sorted by:  -
Date:       7/27/14 4:51 PM


>-<   >   < Resultset
  2   2   0 meta.Stacking '-X 10 -M \"functions.Logistic -R
  0   0   0 meta.AdaBoostM1 '-P 100 -S 1 -I 10 -W trees.J48
 -1   0   1 meta.Bagging '-P 100 -S 1 -I 10 -W trees.J48 --
 -1   0   1 trees.J48 '-C 0.25 -M 2' -217733168393644444
```

Fig 1

Rank	Classifier
1	STACKING
2	BOOSTING
3	BAGGING
4	J48

Fig 2

VI. RESULT

The WEKA experimenter arranges the algorithm and its ensembles based on the percent of correctly classified instances and rank it accordingly. While considering the method which increases the efficiency of j48 method the most, we find that BLENDING does this much efficiently than others. Other ensembles like BOOSTING and BAGGING also increase the efficiency of J48. From Fig 1,2 we can see Blending ensemble(Stacking) comes first in the list followed by Boosting ,Bagging and at last J48 individually. Another parameter which strengthens our result is the percent of correctly classified instances shown by the WEKA experimenter. Again, from Fig 3,4 we see that BLENDING ensemble classifies 86.07% of instances correctly. The other ensemble methods also classifies more instances correctly than individual J48 algorithm.

```
Test output
Tester:     weka.experiment.PairedCorrectedTTester
Analysing:  Percent_correct
Datasets:   1
Resultsets: 4
Confidence: 0.05 (two tailed)
Sorted by:  -
Date:       7/27/14 4:57 PM


Dataset                   (1) trees.J4 | (2) meta. (3) meta. (4) meta
---------------------------------------------------------------------
sonar                   (100)    73.61 |   79.13     77.72     86.07
---------------------------------------------------------------------
                               (v/ /*) |   (0/1/0)   (0/1/0)   (1/0/0


Key:
(1) trees.J48 '-C 0.25 -M 2' -217733168393644444
(2) meta.AdaBoostM1 '-P 100 -S 1 -I 10 -W trees.J48 -- -C 0.25 -M 2'
(3) meta.Bagging '-P 100 -S 1 -I 10 -W trees.J48 -- -C 0.25 -M 2' -51
(4) meta.Stacking '-X 10 -M \"functions.Logistic -R 1.0E-8 -M -1\" -S
```

Fig 3

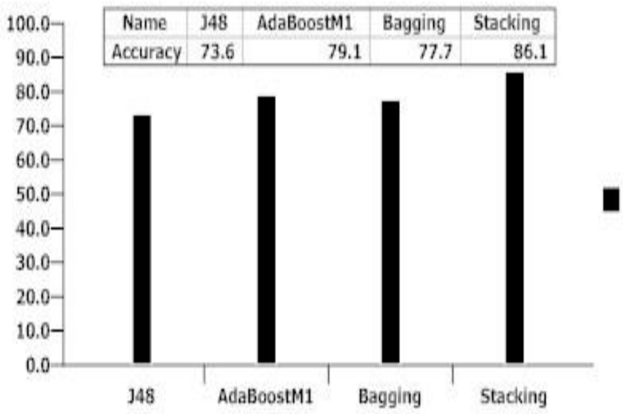

Fig 4

The ranking table(Fig 2) shows the number of statistically significant wins each algorithm has had against all other algorithms on the dataset. A win, means an accuracy that is better than the accuracy of another algorithm and that the difference was statistically significant. The accuracy bar graph(Fig 4) shows us that the efficiency of the J48 algorithm increases with every ensemble method.

VII. CONCLUSION

In this paper we try to increase the efficiency of J48 algorithm on SONAR dataset using different ensembles and we come to conclusion that the ensemble methods are always more efficient than the individual algorithm in this case. For this dataset , Stacking, the blending ensemble works best . The other ensembles are better than individual J48 algorithm. In future we may extend this work by using other ensemble methods on the same dataset.

REFERENCES

[1] 1.J. R. Quinlan, Induction of Decision Trees, Machine Learning, v.1 n.1, p.81-106

[2] "Building Decision Trees with the ID3 Algorithm", by: Andrew Colin, Dr. Dobbs Journal, June 1996

[3] Quinlan,J.R. C4.5:Programs for Machine Learning.Morgan Kaufmann Publishers,1993

[4] Breiman, Leo; Friedman, J. H.; Olshen, R. A.; Stone, C. J. (1984). Classification and regression trees.

[5] Kass, Gordon V.; An Exploratory Technique for Investigating Large Quantities of Categorical Data, Applied Statistics, Vol. 29, No. 2 (1980), pp. 119–127

[6] Yoav Freund, Robert E. Schapire: Experiments with a new boosting algorithm. In: Thirteenth International Conference on Machine Learning, San Francisco, 148-156, 1996.M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989.

[7] Leo Breiman (1996). Bagging predictors. Machine Learning. 24(2):123-140

[8] David H. Wolpert (1992). Stacked generalization. Neural Networks. 5:241-259..

[9] Goran R.P ,Sejnowski ,T.J ,Learned Classification of Sonar Targets using a Massively Parallel Network,IEEE pp 1135-1140

[10] D. Aha, D. Kibler (1991). Instance-based learning algorithms. Machine Learning. 6:37-66.

[11] le Cessie, S., van Houwelingen, J.C. (1992). Ridge Estimators in Logistic Regression. Applied Statistics. 41(1):191-201.