Ensembles and Their Applications

Matt Dixon
University of Montana
Master of Arts, Mathematical Sciences/Statistics
In-House Project, Dec 2000

DISTRIBUTION STATEMENT A
Approved for Public Release
Distribution Unlimited

20010307 163

HQM01-06-1078

REPORT DOCUMENTATION PAGE	Form Approved OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	12.Feb.01	MAJOR REPORT

4. TITLE AND SUBTITLE
ENSEMBLES AND THEIR APPLICATIONS

5. FUNDING NUMBERS

6. AUTHOR(S)
CAPT DIXON MATTHEW C

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
UNIVERSITY OF MONTANA

8. PERFORMING ORGANIZATION REPORT NUMBER
CI01-46

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
THE DEPARTMENT OF THE AIR FORCE
AFIT/CIA, BLDG 125
2950 P STREET
WPAFB OH 45433

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION AVAILABILITY STATEMENT
Unlimited distribution
In Accordance With AFI 35-205/AFIT Sup 1

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

14. SUBJECT TERMS

15. NUMBER OF PAGES
21

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT

Ensembles and Their Applications

Matt Dixon
Captain, USAF
University of Montana
Master of Arts, Mathematical Sciences/Statistics
In-House Project (21 pages), Dec 2000

Abstract

Classifiers are powerful tools that, given some predictor variables, make a prediction of which category a variable belongs. Linear Discriminant Analysis, k-nearest neighbor, and classification trees are a few of these classifiers. An ensemble attempts to iteratively apply a classifier to a data set to decrease the error rate over a single classifier. Ensembles were the focus of my research. I discussed three accepted, yet still developing, ensembles: Bagging, Arc-Boosting, and Ada-Boosting. Then I demonstrated that they usually decrease the error rate over a single classification tree. I also demonstrated that the k-nearest neighbor classifier rarely benefits from an ensemble. Thorough study of the three ensemble methods led to the exploration of a new ensemble method that proved to reduce error rate on all the sampled data sets, but did not produce competitive results with the other ensemble methods. Most importantly, predictions on the Landsat Imagery Satellite data improve dramatically from all three of the ensembles, when a classification tree is the classifier. Some error rates were bettered by up to 25%, while others saw less significant reductions. However, there was at least a small reduction in error rate for nearly all of the data sets, so long as the single classifier performed slightly better than guessing.

Bibliography

Breiman, L. (1999), 'Using Adaptive Bagging to Debias Regressions', Technical Report 547, February 1999, Statistics Department, University of California at Berkley.

Friedman, J. H. (1996), 'On Bias, Variance, 0/1 – Loss, and the Curse of Dimensionality', Department of Statistics and Stanford Linear Accelerator Center, Stanford University.

Friedman, J., Haste, T., & Tibshirani, R. (1999), 'Additive Logistic Regression: a Statistical View of Boosting, Second revision', Nov 30, 1999.

Opitz, D & Maclin, R. (1999), 'Popular Ensemble Methods: An Empirical Study', *Journal of Artificial Intelligence Research 11*, pp 169-198.

Schapire, R. E. (1999), 'A Brief Introduction to Boosting', *Proceedings of the sixteenth International Joint Conference on Artificial Intelligence, 1999.*

Steele, B. M., 'Land Cover Mapping Using Combination Classifiers', Department of Mathematical Sciences, The University of Montana.

Introduction

Classifiers such as Classification Trees, Linear Discriminant Analysis, and k-nearest neighbor (knn) are very powerful tools for making predictions for test sets with unknown group assignments. With these tools come the ability to make estimates about what group a particular observation belongs to. For example, imagine a scenario where the height and weight of an individual has been measured and recorded, but not the gender. Suppose I want to predict the gender. Since adult males tend to weigh more and be taller than adult females, there should be some weight and height for which if they are more than, they'll be classified as a male and if they are less, they'll be classified as a female. If they are over either height or weight and under the other, I might classify them as either. Clearly some people will be classified incorrectly, but, hopefully, if the correct height and weight thresholds are selected, the amount of people classified incorrectly will be a minimum. Now the question becomes how to select the optimum threshold (or decision rule) for the x variables, height and weight.

The above mentioned classifiers are a few of the methods which attempt to do this. For the purposes of this paper it is not imperative to understand how each classifier works only that they work differently and where one falls short the other may excel. It should be noted that knn works with continuous variables. However, Classification Trees, which will be referred to as a "tree" or "trees", are very useful when we have a lot of categorical variables, but can also be effective with continuous variables. Classifiers are similar to regression in that they attempt to predict an unknown variable, however, regression predicts continuous variables and classifiers predict categorical variables.

Traditionally, by some analysis, the best classifier is chosen for a particular data set and a single classification rule is constructed from a training set (which the dependent group variables are known) to make predictions on a test set with unknown dependent group variables. As mentioned above, even when the best classifier is chosen and a classification rule is constructed, most likely the rule will not perfectly predict the dependent variable of the each observation in the test set. My research focused on a method of reducing the error rate associated with a single classifier. I primarily used the Classification Tree for my research, but some experimentation was done with knn. Within the last ten years, the ensemble ideas have emerged as showing great promise for significantly reducing the error rate of a single classifier.

The primary object of my research is two fold. First, I set out to understand and document how the ensembles work and how well they work. Second, I needed to create functions in S-Plus for each of the methods which would not only enable me to evaluate how well they work, but also have on file for the use on new data sets. As a secondary objective I explored an alternative ensemble.

Motivation

In the spring of 2000 I took a Multivariate Analysis course, in which I studied classifiers. At the same time, Professor Brian Steele, Mathematics Department, University of Montana, was doing analysis on Landsat Imagery Satellite data to predict land cover assignments of specific geographical "quadrants". His work focused on predicting the type of ground cover in a polygon from a satellite photo which measured variables such as altitude, slope, color bands present, etc. Knowing these variables, the

objective was to predict the type of ground cover for each polygon He'd explored knn and LDA and was experimenting with new spatial classifiers. After discussing possibilities, we decided I should learn about ensembles and see if this type data could benefit from ensembles.

Research

My approach to the research was to first find literature on ensembles. After a thorough study of the idea behind ensembles I needed to develop S-Plus code. Ensembles find their roots in machine learning and most of the literature focused on the two group case (often this is referred to as the two class case, and both will be used interchangeably). First, I coded all the ensemble methods in S-Plus for two class scenarios. In this scenario the assigned groups are labeled as ± 1 . This is not only easier to understand but is also easier to code. With the two class code finished, I was able to test some data sets to see if the code was written correctly. Next, I coded the multi class case, which proved to be somewhat more burdensome. With this complete and tested, I was able to run some of the Landsat data sets through it to determine if they could benefit from ensembles. Later, I will discuss the results for all the ensemble methods and the applicable data sets.

Ensembles

To discuss how ensembles work I will give a word description and a mathematical description because one is not complete without the other.

Let me talk in general about ensembles. An ensemble creates multiple classification rules from multiple sub training sets randomly selected from the original training set. Once the classification rules are created, each is applied to a test set. An observation in the test set will have as many group assignments as there are sub training sets. The final group assignment for a particular observation is determined via a specific voting scheme, which depends on the ensemble. For example, imagine that we have a training set (by definition we know the group variable for each observation) which we will use to aid in the prediction of the unknown group assignments in a test set. For this example, we decide to apply a classification tree. From previous experience, or some prediction method, we have an idea of what the error rate will be for this type of data. Nearly always, we can benefit from employing an ensemble method.

I will discuss, in detail, three ensembles that have drawn the attention of several well known statisticians such as Robert Tibshirani, Robert Haste, and Leo Breiman. They are known as Bagging, Arc Boosting, and Ada Boosting. These three methods seem to have become the foundations of other ensembles. Several statisticians have studied deviations of these methods to see if they can be improved upon and to also try and gain insight into *why* the ensembles are successful in reducing error rates. I will also discuss an off-shoot of ada- boosting which was proposed by Professor Brian Steele. Discussing the theory behind why ensembles work is beyond the scope of this paper, however, I will touch briefly on which ensemble methods attempt to reduce what sort of error.

To begin talking about the specific ensembles, I must first define the terms and variables:

n = # observations in the training set, with (x_i, y_i), where x_i is any number of measured variables and y_i is the group assignment of the *i*th observation, $i = 1,\ldots,n$

m = # observations in the test set, with (x_j, y_j), where y_j is unknown , $j = 1,\ldots, m$

t = # of sub-training sets

w is the group assignment {a, b, c, …} for an observation

z is an iteration, where $z \in \{1\ldots t\}$

p_{iz} = probability of *i*th training observation selected for the *z*th sub training set

c_z is the classification rule, of type c classifier, constructed from the *z*th sub training set

Bagging

Bagging is the simplest ensemble and is what I'll begin with. In Bagging, each sub training set is created by bootstrapping the original training set. In other words, each sub training set is created by taking n random sampling, with replacement and equal probability of selection ($p_{1z} = p_{2z} = \ldots = p_{nz}$) , from the original training set. In the sub training sets, there will generally be some observations that are not represented and there will be some that are represented more than once. Once the sub training sets are formed, a separate classification rule is constructed from each sub training set, then each of these classification rules are applied to the test set to develop a vector of group assignments for each test set observation. The final predicted group assignment for an observation is

determined by a simple majority vote. When there is a tie, the assignment is made with a simple random sample from the tied groups.

Notationally, the final group assignment is

$$y_j = \underset{w = a,b,\ldots}{\operatorname{argmax}} \{ \sum_{z=1}^{t} I(y_{jzw}) \} , \quad \text{for } j = 1 \ldots m$$

where,

$I(y_{jzw}) = 1$ if y_{jz} is assigned group w

0 otherwise .

One known reason for why bagging is successful in reducing the error rate of a single classifier is that it reduces the variance of the classifier.

Arc Boosting

Arc Boosting differs only slightly from Bagging in its form, but in its theory the difference is more significant. The format of Arc Boosting (also referred to as Arcing) differs primarily in two ways. First, there is order associated with the sub training sets; we introduce the idea of an iteration, where a new sub training set is randomly selected based on the results from previous iterations. Once the first sub training set is randomly selected and a classification rule is constructed, the rule is applied to the original (entire) training set. Applying the classification rule to the training set with known group assignments, y_i, gives "predicted" group assignments to each observation. Now, compare the predicted to the actual and record which observations were incorrectly predicted. This leads to the second way that Arcing differs from Bagging, the next iteration attempts to focus on the observations that were incorrectly classified in all previous iterations. The initial sub training set is randomly selected with replacement and

equal probabilities of selection, i.e. bootstrapping. However, subsequent sub training sets are randomly selected from the original training set with unequal probabilities of selection. The observations that have been incorrectly classified by previous classification rules will have larger selection probabilities for the next sub training set than those that have been correctly classified. The algorithm for computing the selection probabilities for the *z*th sub training set is

First iteration

$p_{i1} = 1 / n$, for $i = 1 \ldots n$

Subsequent iterations:

Apply c_{z-1} to the entire training set and document for all i which have incorrect group assignments. Let q_i = # of times the ith case has been incorrectly assigned by $c_1 \ldots c_{z-1}$

$$p_{iz} = (1 + q_i^4) / (n + \sum_{i=1}^{n} q_i^4) \text{ , for } i = 1 \ldots n .$$

From this algorithm it can be seen that p_{iz} , the probability of the ith observation being selected for the zth sub training set, is large if the previous classifiers have been strong (correctly assigns the majority of the observations) while the ith observation has continued to be incorrectly assigned. However, p_{iz} will be small in either of the following two situations. First, when the previous classifiers have been weak (misclassified a large portion of the training set observations) or when the ith observation has been correctly assigned for most of the previous iterations. Therefore, as the iterations continue, more focus is given to the hard to classify observations—to create a rule that is better suited to classify them correctly. The easy to classify observations could, in theory, still be classified correctly by a classification rule constructed from a sub training set consisting

mostly of hard to classify observations. This is how Arcing differs fundamentally from Bagging, for it not only attempts to reduce the variance of the classifier, but it also attempts to reduce the bias by focusing on the observations which have been incorrectly assigned.

The final group assignment for an observation is determined by a simple majority vote, the same as in Bagging.

Ada Boosting

Ada Boosting is similar to Arcing in its format. It is different in that it calculates p_{iz} differently and the voting scheme for an observation's final assignment is weighted by how well a classification rule predicted the training set at a given iteration. As with Arcing, Ada Boosting randomly selects a sub training set for each iteration, based on the performance of the previous classification rules. However, the algorithm to compute p_{iz} is different.

First iteration

$p_{i1} = 1 / n$, for $i = 1, \ldots, n$

Subsequent iterations:

Apply c_{z-1} to the entire training set

Standardize $p_{i(z-1)}$ for $i = 1 \ldots n$

Let $\varepsilon_z = \sum_{i=1}^{n} p_{i(z-1)} * I(y_i)$, where

$I(y_i) = 1$, if ith case was incorrectly classified by c_{z-1}

0 , otherwise

for $0 < \varepsilon_z \leq 0.5$

$$p_{iz} = \left| I(y_i) * p_{i(z-1)} * \varepsilon_z^{-1} - p_{i(z-1)} \right| \text{, for } i = 1....n$$

otherwise

$$p_{iz} = 1 / n \text{ , for } i = 1...n$$

Note that the discussion about what will cause p_{iz} to be large or small in Arcing is paralleled with the same principles here. If the zth iteration's classification rule is dramatically inaccurate and causes $\varepsilon_z > 0.5$, then for all incorrectly assigned observations $I(y_i) * p_{i(z-1)} * \varepsilon_z^{-1} < 2 * p_{i(z-1)}$ causing $p_{iz} < p_{i(z-1)}$. If $\varepsilon_z \notin (0,0.5]$, p_{iz} a reset to equal probabilities for all i. However, David Opitz, Computer Science Department, University of Montana, states in his paper Popular Ensemble Methods: an Empirical Study (1999) that he found this to happen in only 5.12% of his results. The final assignment of an observation in the test set is no longer a result of a simple unweighted majority vote, but is a weighted voting scheme defined below.

$$y_j = \underset{w = a,b,...}{\text{argmax}} \{ \sum_{z=1}^{t} \beta_z * I(y_{jzw}) \} \quad \text{for } j = 1...m$$

where,

$$\beta_z = \log[(1 - \varepsilon_z) * \varepsilon_z^{-1}] \text{ , for } 0 < \varepsilon_z \leq 0.5,$$

$$\beta_z = 5 \text{ , for } \varepsilon_z = 0 ,$$

$$\beta_z = .0005 \text{ , for } \varepsilon_z > 0.5 ,$$

and

$I(y_{jzw}) = 1$ if y_{jz} is assigned w

0 otherwise

As defined, ε_z will be large for a weak classification rule and small for a strong classification rule. Therefore, the weight of a vote, $\beta_z = \log[(1 - \varepsilon_z) * \varepsilon_z^{-1}]$, will be

heavier for an iteration with a stronger classification rule and will be lighter for an iteration with a weaker classification rule. It has been said that Ada Boost with Trees is the best off the shelf classifier in the world. (Friedman, Haste, and Tibshirani)

Left Out Boosting

The final ensemble I explored was proposed by Professor Brian Steele. We dubbed this method "Left Out Boosting". The idea of Left Out Boosting is to hopefully put more emphasis on the cases which had absolutely no influence on the *z*th classification rule and if they were classified incorrectly by that rule. Left Out is a manipulation of Ada Boosting, differing only in how the selection probabilities at each iteration are computed. The final voting scheme is the same as Ada Boosting. Each p_{iz} is computed based on the performance of the previous classification rule applied only to the observations that were not selected for the *z-1* sub training set.

First iteration

$p_{i1} = 1 / n$, for $i = 1, \ldots, n$

Subsequent iterations:

Apply c_{z-1} to the **unselected observations** of the original training set

Standardize $p_{i(z-1)}$ for $i = 1 \ldots n$

Let $\varepsilon_z = \sum_{i=1}^{n} p_{i(z-1)} * I(y_i) * I(i)$, where

$I(y_i) = 1$, if *i*th observation was incorrectly classified by c_{z-1}

0 , otherwise

$I(i) = 1$, if *i*th observation was left out of the z-1 sub set

0 , otherwise

for $0 < \varepsilon_z \leq 0.5$

$$p_{iz} = \left| I(i) * I(y_i) * p_{i(z-1)} * \varepsilon_z^{-1} - p_{i(z-1)} \right| \text{ , for } i = 1 \ldots n$$

otherwise

$$p_{iz} = 1 / n \text{ , for } i = 1 \ldots n$$

These formulae are identical to those for Ada Boosting except for the addition of the second indicator variable that delineates between the observations that were selected for the *z-1* iteration and those that were not. The final voting scheme is defined the same as for Ada Boosting.

Results

It has been said that boosting is immune to over fitting. (Friedman 1999 and Schapire 1999) In fact, it isn't until the number of iterations, sub training sets, reaches the order of 10,000 that it appears as though the error rate will slightly increase, while dramatic results are see with as little as 20 to 30 iterations. In fact, the error rate for classification trees and most data sets plateaus by the 30th iteration. (Opitz & Maclin 1999) In discussion of my results, I will first verify that some number between 20 and 30 does appear to be the "magic number". Next, I will look at the Landsat data sets, with 30 iterations (since over fitting is not a concern, I decided to err on the conservative side) and compare two different classifiers, knn (with Euclidean distances) and classification trees. No pruning was done on any of the trees, including the single tree constructed from the original training set for comparison purposes. The minimum node size was set at 10 and the minimum cut size was set at 5.

These numbers are somewhat arbitrary, but for comparison purposes, whether or not they are optimal should be insignificant. In fact, it has been shown that using stumps (only splitting the data once, for two group data sets) with a boosting ensemble will still produce superior results compared to a single classification tree provided that the stumps do at least slightly better than simply guessing. (Friedman 1999) For the nearest neighbor classifier, I let k = 10 because it is commonly used.

The results will be given in three steps. First, results for Bagging, Arc Boosting, and Ada Boosting with a tree will be given where they are applied to three different two-class data sets. Each was done with several different iterations: 15, 30, 50, and 100. These results will demonstrate the potential of the ensembles and also show that by the 30^{th} iteration the minimum error rate is nearly achieved. A brief description of all the data sets will be given. Second, I will demonstrate the potential benefits that can be gained by applying an ensemble, with a tree, to a Landsat data set. Finally, I will provide results for a data set that a classification tree does poorly on. All error rates are based on 4-fold cross validation. The original data set is divided into four randomly selected sets. One at a time the four sets are treated as the test set and the remaining three sets are treated as the training set. Then the four error rates are averaged.

Ionosphere

Data Set:	Ionosphere			
	Error Rates			
Original Tree	0.127			
# of iterations	**15**	**30**	**50**	**100**
ada boosting	0.085	0.068	0.077	0.074
arc boosting	0.074	0.071	0.077	0.066
bagging	0.071	0.08	0.083	0.083
	Error Rates based on 4-fold cross validation			

The ionosphere data set has 34 continuous predictor variables that have to do with some complex electromagnetic signal from 16 different radars. There are no missing values in the 351 observations. The group variable, y, is whether a radar signal was considered "good" or "bad". There are 225 "good" and 126 "bad" observations. The original tree, with parameters described above, has an error rate of 12.7% while all three of the ensemble methods reduce this error rate significantly after only 15 iterations.

FLAG

Data Set:	Flag			
	Error Rates			
Original Tree	0.149			
# of iterations	**15**	**30**	**50**	**100**
ada boosting	0.158	0.132	0.149	0.123
arc boosting	0.123	0.14	0.132	0.123
bagging	0.14	0.096	0.096	0.105
	Error Rates based on 4-fold cross validation			

The flag data set has 20 categorical and 10 continuous predictor variables. The categorical variables are measures of things such as continent, geographical quadrant, and whether a certain element is present in each country's flag. The continuous variables are measures of country's population, country's area, and number of certain things present in their flag. There are no missing values over the 114 observations. The group variable, y, is a country's predominant religion. The original data set had 8 different groups (religions), but for the two class case "Catholic" and "Christian" were combined for 56 observations, while Hindu, Buddhist, and others were combined for 58 observations. The

original tree, with parameters described above, has an error rate of 14.9% while Bagging produced the most significant error rate reduction after 30 iterations.

House Votes 84

Data Set:	house.votes.84			
	Error Rates			
Original Tree	0.041			
# of iterations	**15**	**30**	**50**	**100**
adatree	0.041	0.032	0.039	0.034
arctree	0.039	0.034	0.041	0.034
bagtree	0.041	0.039	0.039	0.039
	Error Rates based on 4-fold cross validation			

The House votes 84 data set has 16 binary categorical predictor variables that are labeled either "yes" or "no" depending on how a Congressional Representative voted on an issue in 1984. There are 288 missing values in 435 observations; this equates to approximately 4% of the values. The missing values were randomly assigned either "yes" or no" with equal probability. The group variable, y, is the representative's party affiliation, either republican (267) or democrat (168). The original tree, with parameters described above, has an error rate of .041%. All of the ensembles showed at least an equivalent error rate to the single tree while some decreased it slightly.

Landsat Data

The Landsat data sets are from satellite images that measure variables on a defined geographical quadrant. Some of the predictor variables are location coordinates, altitude, slope, and specific color bands for a polygon. The dependent variable, or group, is ground cover type.

Data Set 3927

Number of iterations 30

Tree				
Single	Ada	Left Out	Arc	Bagging
0.405	0.317	0.361	0.333	0.353

10-NN				
Single	Ada	Left Out	Arc	Bagging
0.468	0.465	0.486	0.496	0.47

2028 observations, 15 groups, or ground cover types,
Disbursement of the groups (64, 287, 45, 31, 42, 17, 346, 11, 37, 455, 501, 58, 11, 57, 66)

Data Set 4029

Number of iterations 30

Tree				
Single	Ada	Left Out	Arc	Bagging
0.269	0.231	0.251	0.212	0.24

10-NN				
Single	Ada	Left Out	Arc	Bagging
0.391	0.386	0.394	0.402	0.39

2528 observations, 14 groups
Disbursement of groups (277, 657, 18, 207, 52, 108, 442, 143, 24, 38, 112, 25, 281, 144)

Data Set 4128

Number of iterations 30

Tree				
Single	Ada	Left Out	Arc	Bagging
0.454	0.369	0.39	0.363	0.384

10-NN				
Single	Ada	Left Out	Arc	Bagging
0.505	0.495	0.507	0.515	0.493

3007 observations, 15 groups
Disbursement of groups (95, 73, 291, 42, 36, 54, 500, 116, 468, 149, 521, 418, 73, 39, 132)

The single tree and single knn both have large error rates for all three data sets. The single tree has a lower error rate than knn, and also benefits from all of the ensembles. The second classifier, knn, does not appear to benefit from the ensembles at all. Opitz and Maclin (99) suggest that unstable classifiers will benefit more from ensembles than will stable classifiers. A classifier is considered unstable if a small change in the training set produces a large change in the predictions. This may be one reason knn does not realize noticeable reduction in error rates with the ensembles while classification trees do.

Finally, I will demonstrate results on the "Sparrow" data set because single knn does better than a single tree and the single tree performs worse than simply guessing. There are five predictor variables, all of which are continuous and are different measures of the size of an observation, a sparrow. The dependent, or group, variable is whether or not a sparrow survived a specific storm. There are 49 observations with no missing values. Twenty-one sparrows died and 28 survived.

Data Set	Sparrow			
Number of iterations	30			

Tree				
Single	Ada	Left Out	Arc	Bagging
0.612	0.653	0.571	0.694	0.551

10-NN				
Single	Ada	Left Out	Arc	Bagging
0.49	0.531	0.51	0.571	0.49

Noting that the single knn barely does better than guessing and does much better than a single tree, I wanted to see if possibly an ensemble would reduce the error rate of the

single knn. This is clearly not the case. In fact, the best ensemble only does as good as the single knn. It is curious to note that while the single tree performs very poorly, Ada and Arc Boosting both make it even worse while Left Out Boosting and Bagging both decrease the error rate.

Conclusion

Ensembles are relatively new and they are still developing. The three main ensemble methods I focused on were Bagging, Arc Boosting, and Ada Boosting. Some exploration was done with an idea called Left Out Boosting. All four methods seem to always reduce the error rate of a single classification tree, as long as the tree performed better than simply guessing. The amount of reduction is related to the data set. Left Out Boosting with a tree did not seem to be as strong as Ada Boosting, at least for 30 iterations, but it did reduce error rate for all the data sets and on the sparrow data set was the only boosting ensemble which reduced the error rate. While classification trees benefit significantly from an ensemble, it does not appear as though knn does. In fact, for all the data sets I experimented with, ensembles run with knn appeared to be at best only as good as the single knn. It has been shown specifically that the Landsat data sets (satellite imagery to predict land cover) definitely benefit from the slight additional effort of using an ensemble with a classification tree, versus a single tree or knn. In two of the three sampled Landsat data sets, Arc Boosting a tree produced the lowest predicted error rate. As Opitz (99) states "... a Bagging ensemble generally produces a classifier that is more accurate than a standard classifier. Thus one should feel comfortable always

Bagging their decision trees...". At least for the Landsat data, this also appears to be true for boosting.

Follow-on Work

Ensembles are still developing and countless papers have been written discussing their theory, alternative ensembles (logit boost, gentle ada boost, etc) (Tibshirani), regression ensembles, and the list continues. Some new ideas came up while I was working on this project. What if two classifiers were combined at each iteration? What if z iterations were run with one classifier and then z iterations run with a different classifier? These two ideas are different because the first would compute classification rules for both classifiers from the same sub training set and then combine the predictions via some function such as a product of the two posterior probabilities. In the second idea, the classifier which runs on the first 30 iterations would have an impact on the sub training sets that are randomly selected for the second classifier and its associated 30 sub training sets. To gain a more complete understanding of how the ensembles work in general, they should be applied to some simulated data sets. Finally, another interesting project would be to test spatial classifier ensembles.

Bibliography

Breiman, L. (1999), 'Using Adaptive Bagging to Debias Regressions', Technical Report 547, February 1999, Statistics Department, University of California at Berkley.

Friedman, J. H. (1996), 'On Bias, Variance, 0/1 – Loss, and the Curse of Dimensionality', Department of Statistics and Stanford Linear Accelerator Center, Stanford University.

Friedman, J., Haste, T., & Tibshirani, R. (1999), 'Additive Logistic Regression: a Statistical View of Boosting, Second revision', Nov 30, 1999.

Opitz, D & Maclin, R. (1999), 'Popular Ensemble Methods: An Empirical Study', *Journal of Artificial Intelligence Research 11*, pp 169-198.

Schapire, R. E. (1999), 'A Brief Introduction to Boosting', *Proceedings of the sixteenth International Joint Conference on Artificial Intelligence, 1999.*

Steele, B. M., 'Land Cover Mapping Using Combination Classifiers', Department of Mathematical Sciences, The University of Montana.