2.2

USING RANDOM FORESTS AND FUZZY LOGIC FOR AUTOMATED STORM TYPE IDENTIFICATION

John K. Williams* and Jennifer Abernethy
National Center for Atmospheric Research, Boulder, Colorado

1. INTRODUCTION

This paper discusses how random forests, ensembles of weakly-correlated decision trees, can be used in concert with fuzzy logic concepts to both classify storm types based on a number of radar-derived storm characteristics and provide a measure of "confidence" in the resulting classifications. The random forest technique provides measures of variable importance and interactions, as well as methods for addressing missing data, suggesting fruitful ways to transform the input data and to structure the final classification algorithm. *N*-fold cross-validation is used as the basis for tuning the algorithm parameters.

2. THE PROBLEM

This paper addresses a problem posed by the 2008 Artificial Intelligence Competition organized by the American Meteorology Society Committee on Artificial Intelligence Applications to Environmental Science and sponsored by Weather Decision Technologies (Lakshmanan et al. 2008). Two datasets were provided by the competition organizers: (1) a training dataset consisting of 1356 records containing numerical measurements of various thunderstorm cluster attributes accompanied by a storm type label specifying whether the storm had been identified by a human as "Not Severe" (NS), "Isolated Supercell" (IS), "Convective Line" (CL) or "Pulse Storm" (PS), and (2) a testing dataset consisting of 1069 feature records but without the storm type label. The goal of the competition was to use statistical analysis, data mining, or machine learning techniques to create a function that would map the thunderstorm cluster attributes to the storm type and to use it to classify the instances in the testing dataset.

This classification problem is drawn from a paper addressing the connection between storm types and the accuracy of operational tornado and severe thunderstorm warnings (Guillot et al. 2008). Because it was not feasible to classify the more than 50,000 storms involved in that study, Guillot et al. trained a decision tree to classify storms automatically, then used those classifications to draw conclusions about warning accuracy. On the dataset prepared for the AI Competition, a sample decision tree provided by the contest organizers for performing this classification produced a multi-category True Skill Score (TSS) of 0.58 on the testing dataset, having the contingency table shown in Table 1. Clearly, greater accuracy would be desirable.

* *Corresponding author address:* John K. Williams, National Center for Atmospheric Research, P.O. Box 3000, Boulder, CO 80307; email: jkwillia@ucar.edu.

Predicted ⇒	NS	IS	CL	PS	TOTAL
True NS	498	0	3	68	569
True IS	2	82	14	33	131
True CL	4	28	41	22	95
True PS	57	24	25	168	274

Table 1: Contingency table for the sample decision tree run on the testing dataset, as provided by the contest organizers. True classes are listed down rows, and predicted classes across columns.

3. PRELIMINARY DATA ANALYSIS

The attributes associated with each instance in the training and testing datasets were derived from "clusters" of thunderstorm identified using a method described in Guillot et al. (2008). These features include the following, in alphabetical order:

AspectRatio (dimensionless) *the ratio of the major to minor axis of an ellips fitted to the storm cluster*

ConvectiveArea (km^2) *the area that is convective*

LatRadius (km) *North-South extent*

LatitudeOfCentroid (degrees) *location of centroid*

LifetimeMESH (mm) *maximum expected hail size over the storm's entire past history*

LifetimePOSH (dimensionless) *peak probability of severe hail over the storm's entire past history*

LonRadius (km) *East-West extent*

LongitudeOfCentroid (degrees) *location of centroid*

LowLvlShear (s^{-1}) *shear closest to the ground as measured by radar*

MESH (mm) *maximum expected hail size*

MaxRef (dBZ) *maximum observed reflectivity*

MaxVIL (kg/m^2) *maximum vertical integrated liquid*

MeanRef (dBZ) *mean reflectivity*

MotionEast ($m\ s^{-1}$) *speed in easterly direction*

MotionSouth ($m\ s^{-1}$) *speed in southerly direction*

OrientationToDueNorth (degrees) *orientation of the major axis of the fitted ellipse*

POSH (dimensionless) *peak probability of severe hail*

Rot120 (s^{-1}) *maximum observed azimuthal shear over the past 120 minutes*

Rot30 (s^{-1}) *maximum observed azimuthal shear over the past 30 minutes*

RowName (dimensionless) *storm ID*

Size (km^2) *storm size*

Speed ($m\ s^{-1}$) *storm speed*

Three of these attributes were not used in the following analysis due to a suspicion that any relationship between them and the storm type in the training dataset was specific to the training days and would not generalize well to the testing dataset; these were LatitudeOfCentroid, LongitudeOfCentroid and RowName. Removing those fields from consideration left 20 candidate predictor variables.

The first step of any statistical analysis or data mining effort is to examine the available data and identify any obvious contamination or inconsistencies. For example, we verified that LifetimeMESH ≥ MESH and Size ≥ ConvectiveArea. However, it was not true, as one might have expected, that Rot120 ≥ Rot30. Furthermore, two values of Rot120 and one value of LowLvlShear were less than zero, and a number of values were flagged as bad or missing, as shown in Table 2. There is no universally best way to handle such questionable or missing values; one approach might be to experiment with several substitution or imputation methods to see which gives the best results. In the present case, the authors chose to set negative values of MaxRef, MeanRef, Rot30, and Rot120 to zero. Moreover, OrientationToDueNorth was rotated somewhat, and missing values of VIL were replaced by

$$\mathrm{MaxVIL} = 0.0071\left(10^{\mathrm{MaxRef}/10}\right)^{4/7} + 0.0200\left(10^{\mathrm{MeanRef}/10}\right)^{4/7} \quad (1)$$

The form of this equation comes from a standard relationship between radar reflectivity and liquid water content, while the coefficients 0.0071 and 0.0200 were found from a least-squares best fit using the instances having good values of MaxVIL, MaxRef and MeanRef.

Another step taken before using the data was to devise additional derived data fields which might be more simply related to the storm type and hence might facilitate creation of a simple, skillful predictive function. These included the following:

fractAreaConvective = ConvectiveArea/Size

delRot = Rot120 – Rot30

delPOSH = LifetimePOSH – POSH

delMESH = LifetimeMESH – MESH

dirMotion *direction of storm velocity vector*

MESHtimesPOSH = MESH × POSH

MESHisbad *boolean 0 or 1 depending on whether MESH is good or bad/missing*

POSHisbad *boolean 0 or 1*

LifetimeMESHisbad *boolean 0 or 1*

LifetimePOSHisbad *boolean 0 or 1*

MaxVILfromRefl *maximum VIL "equivalent" from equation (1)*

LatLonArea = LatRadius × LonRadius

MaxRot = max(Rot30, Rot120)

meanBothRefl = (maxRef + meanRef)/2

meanBothVIL = (MaxVIL + MaxVILfromRefl)/2

These derived variables were selected based on physical intuition and curiosity rather than any systematic methodology, though more principled methods, including a matrix of variable interactions provided by the random forest training procedure might have been used. With these additions, there were 35 candidate predictor variables.

A final note regarding the data: the training dataset was observed to consist of 1356 instances divided into storm types as 526 NS, 222 IS, 208 CL, and 400 PS, for a frequency distribution of (0.388, 0.164, 0.153, 0.295). From the sample decision tree contingency table for the testing set (Table 1), it can be seen that the 1069 instances in the testing set are distributed as 569 NS, 131 IS, 95 CL, and 274 PS, for a frequency distribution of (0.532, 0.123, 0.089, 0.256). Clearly there are relatively more non-severe cases and fewer convective line cases in the testing dataset, which could pose a challenge for generalizing from the training to testing dataset.

Field Name	*Train Set*	*Test Set*
AspectRatio	0	0
ConvectiveArea	0	0
LatRadius	0	0
LifetimeMESH	335	412
LifetimePOSH	802	740
LonRadius	0	0
LowLvlShear	6	15
MESH	390	487
MaxRef	0	2
MaxVIL	83	60
MeanRef	0	2
MotionEast	113	73
MotionSouth	113	73
OrientationToDueNorth	0	0
POSH	910	799
Rot120	23	31
Rot30	4	12
RowName	0	0
Size	0	0
Speed	113	73

Table 2: Number of values labeled as bad or missing (-99900) in the training and testing datasets, respectively.

4. RANDOM FORESTS

The machine learning technique selected for producing the classification algorithm was random forests (Breiman 2001), which one of the authors has found to work well on other datasets (Williams et al. 2007, Cotter et al. 2007). Essentially, random forests are ensembles of weak, weakly-correlated decision trees that "vote" on the correct classification of a given input. Because the decision trees are weak, they minimize the risk of overfitting the training set, a significant and well-known problem with individual decision trees. On the other hand, since the trees are weakly correlated with one another, using a large number of them in an ensemble makes up for the weakness of the individual constituent trees and provides a powerful predictive model.

In constructing each tree of a random forest, a "bagged" training sample is selected by drawing a random subset of *n* instances from the *n*-member training set, with "replacement" after each draw. This means that, on average, each tree is trained on roughly 2/3 of the dataset, including many duplicates. Then, at each node of the tree, a subset of only *m* randomly-selected feature variables are chosen as candidates for splitting, contrasting with the usual decision tree practice of choosing the best split from all the feature variables. Because not all feature variables are used to train each tree, those not used (the so-called "out-of-bag" samples) may be used to evaluate the performance of that tree. This allows the random forest training process to estimate the importance of each variable based on the degradation in classification performance when each variable's values are randomly permuted among training instances. Using this technique, the feature variables may be ranked in order of their importance to the random forest's performance, providing a helpful starting point for performing feature selection.

Once a random forest has been trained, the trees function as an "ensemble of experts" to make predictions. For example, **Error! Reference source not found.** shows a conceptual diagram of a random forest with 100 trees. When a new instance (vector of attributes) is presented to the random forest, each tree will output a classification. These class "votes" are then compiled, and can be used to classify the point based on the consensus "winner", or the vote distribution may be used to assess the confidence of that prediction.

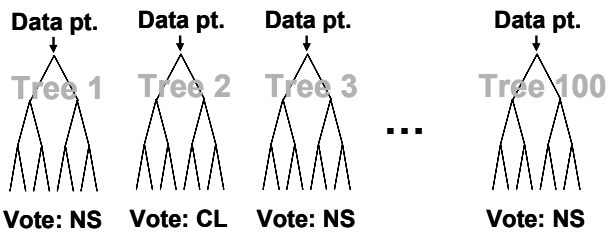

Figure 1: Conceptual diagram of a trained random forest, an ensemble of decision trees that "vote" on the classification of each data point (vector of attributes).

5. TRAINING AND CROSS-VALIDATION

A random forest may be quickly trained to create a predictive algorithm using a training set of labeled instances along with the default choice of *m* and the number of trees, *T*. However, to ensure that the choice of *m* and *T* and the set of variables used are appropriate for the problem, it is helpful to perform tests in which a portion of the training dataset is withheld during the random forest training and then used instead to evaluate the trained random forest's skill. To ensure that this evaluation is meaningful, the random choice of holdout set, the random forest training, and the evaluation should be repeated numerous times, say *N* times, and is therefore called *N*-fold cross-validation.

As noted earlier, the distribution of storm types is different in the training and testing datasets for this problem. Therefore, the authors chose to select the random holdout sets to always have the same distribution of categories as the known distribution in the testing dataset, with the idea that the evaluation would then better reflect the random forest's ability to generalize to that distribution. Using this approach, different values of *m* and *T* were tried using a 20% holdout set, and the random forest's TSS was evaluated. These experiments suggested that the skill improved rapidly as *T* was increased from 10 to 50, then much more slowly with increases to 100 or 200 trees, and it was difficult to establish a statistically significant improvement for 500 trees. Using 200 trees, various values for *m* were tried, and several different around the default number were found to have quite little impact on the skill. Values of $T = 1000$ and $m = 3$ were chosen for the final random forest training run entered into the contest.

The dependence of random forest performance on the set of predictor variables used was also examined; this process is illustrated in a bit more detail. First, using a generous choice of $T = 1000$ to insure accuracy of the importance rank lists, the random forest was run using all the candidate predictor fields to produce a ranked list of variable importance. The variable with the lowest importance rank was then removed, and the process repeated in a "backwards selection" process. For each set of variables, *N*-fold cross-validation was performed using $T = 200$ and a holdout set of 10% of the instances to evaluate the potential skill of a forest trained using those variables. Classifications were obtained from the random forest vote distributions using two different approaches: the simple mode (taking the classification as the class with the most votes, or "winner"), and the mode after adjustment of the class votes for the difference in storm type distributions in the training and test sets. The results are shown in Figure 2 and Figure 3. As expected, the random forest performance initially improved with the increasing number of variables used, though more slowly after about 15 variables; it appears to reach a maximum around 25 variables, and then stays about the same. Note that the second method, which adjusts the votes before taking the mode, does seem to perform somewhat better, with mean and median TSS around

0.67 for 25 variables as opposed to 0.65 for the simple mode. Because one generally wants to use the smallest number of variables that provide good classification to avoid the risk of unnecessary generalization error from an overly complex function, the final random forest training was performed using these 25 variables, listed in order of the random forest "importance" rank from most to least important: meanBothVil, MaxVIL, MaxRef, meanBothRefl, MaxVILfromRefl, MESH, LifetimeMESH, MeanRef, LifetimePOSH, delMESH, LowLvlShear, AspectRatio, MESHisbad, LatRadius, LatLonArea, Size, Rot30, POSH, LonRadius, Speed, OrientationToDueNorth, MotionSouth, Rot120, MotionEast, and ConvectiveArea. The final random forest classification was tried with both the simple mode and the mode of the distribution-adjusted votes. Although the distribution adjustment seemed to work best in cross-validation tests, the simple mode produced a distribution of classes that better matched the known distribution of the test set, and therefore the simple mode classification was submitted to the contest.

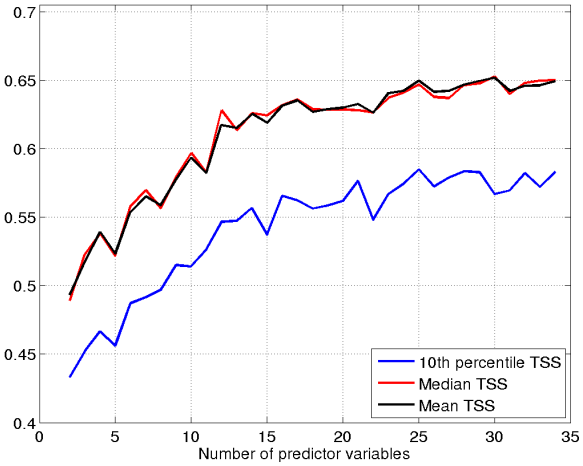

Figure 2: Random forest skill as a function of the number of predictor variables, for simple mode category selection and sets of variables selected as described in the text. The median, mean, and 10th percentile TSS over 120 cross-validation runs are shown. The large difference between the 10th percentile and mean and median TSS is due in large part to the small (10%) random holdout sets used.

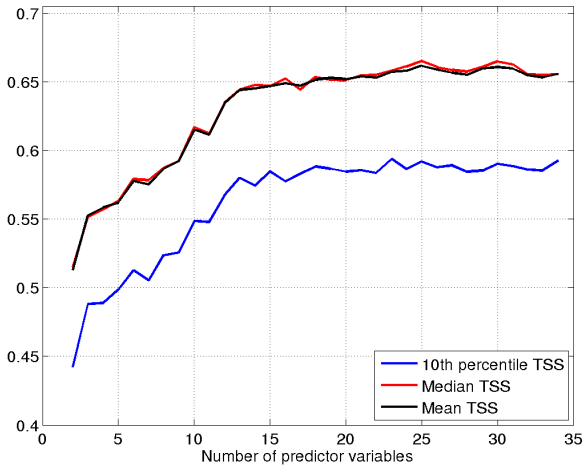

Figure 3: Same as Figure 2, but for "distribution adjusted" votes and 600 cross-validation runs.

6. TESTING DATASET RESULTS

The submitted random forest classifications of the training dataset turned out to have a TSS of 0.638 when compared to the actual storm type labels. This performance is slightly worse than had been suggested by the cross-validation experiments, possibly because the test dataset was produced from separate days with slightly different combinations of conditions, making generalization imperfect.

Following the submission results, the contest organizers made the true dataset classifications public. This allowed a demonstration of how a fuzzy logic approach to interpreting the random forest votes might add value. For each instance, the distribution of votes was used to produce a classification "confidence" c in each mode classification via $c = (4/3)v/T - 1/3$, where v is the number of votes obtained by the winning category. If only the instances in the testing set having confidence above some threshold are considered, the fraction of data declines but the percent classified correctly and the TSS both increase, as depicted in Figure 4. For example, using only the half of the dataset with the highest classification confidences, the TSS rises from 0.64 to 0.77 and the percent correct rises from 77% to 88%. This "confidence" measure of uncertainty in the classification, which is a natural product of the random forest ensemble, could potentially be very useful to downstream applications. Such assessments of confidence are an essential component of many fuzzy logic algorithms.

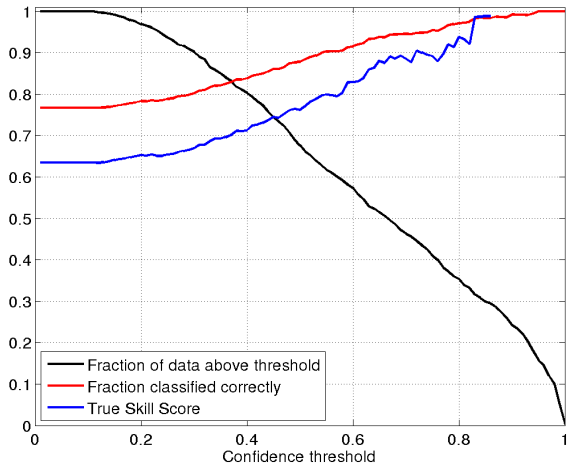

Figure 4: Test set performance of the final trained random forest as a function of classification "confidence" threshold derived from the vote distribution as described in the text.

7. CONCLUSION

This paper has shown how a dataset can be analyzed and a random forest trained to learn a predictive function based on it. The data analysis includes considering how bad or missing data may be handled, and adding derived data fields that may be more simply or robustly related to the classification problem. The parameters used for the random forest training may be selected using *N*-fold cross-validation experiments, though the random forest appears to work remarkably well over a wide variety of choices. A minimal set of predictor variables to be used for training the random forest may be selected in a similar fashion. Finally, a "confidence" in the random forest classification can be easily produced using the vote distribution, and does appear to correlate well with classification accuracy. This confidence value may be useful to interpreting the results or to automated downstream applications.

The authors wish to thank Weather Decision Technologies for sponsoring this competition, and the AMS Committee on Artificial Intelligence Applications to Environmental Science for organizing it.

8. REFERENCES

Breiman, L., 2001: Random forests. *Machine Learning*, 45, 5-32.

Cotter, A., J. K. Williams, R. K. Goodrich and J. Craig, 2007: A Random Forest Turbulence Prediction Algorithm. *AMS Fifth Conference on Artificial Intelligence Applications to Environmental Science*, 1.3.

Guillot, E. M., V. Lakshmanan, T. M. Smith, G. J. Stumpf, D. W. Burgess, and K. L. Elmore, 2008: Tornado and Severe Thunderstorm Warning Forecast Skill and its Relationship to Storm Type. *AMS 24th Conference on International Interactive Information and Processing Systems (IIPS) for Meteorology, Oceanography, and Hydrology*, 4A.3.

Lakshmanan, V. and E. E. Ebert and S. E. Haupt, 2008: The 2008 Artificial Intelligence Competition. *AMS Sixth Conference on Artificial Intelligence Applications to Environmental Science*, 2.1.

Williams, J. K., J. Craig, A. Cotter, and J. K. Wolff, 2007: A hybrid machine learning and fuzzy logic approach to CIT diagnostic development. *AMS Fifth Conference on Artificial Intelligence Applications to Environmental Science*, San Antonio, TX, 1.2.