Alternative Data Mining Techniques for Predicting Tropical Cyclone Intensification

Kyriakos Chatzidimitriou and **Andrew Sutton**
Department of Computer Science
Colorado State University
Fort Collins, CO 80523
email: {kyriakos,sutton}@cs.colostate.edu

Abstract

Predicting the future behavior of tropical cyclones is a problem of great importance for the atmospheric science community with concrete applications. Researchers understand enough about modeling storm systems to predict their track, but forecasting their future intensity remains elusive. In the present study, we formulate tropical storm intensification prediction as a supervised data mining problem; the objective being to produce accurate early warnings with respect to changes in wind speed of a particular storm. We examine two alternative approaches to discover classification rules on current hurricane data: particle swarm optimization and class association rules. Particle swarm optimization employs a population based search method to optimize a rule quality function and discover patterns in the data. Classification with association rules finds sufficiently supported trends in the data and transforms this knowledge into sets of rules. We examine both approaches in detail, present our findings and discuss their impact.

Introduction

Tropical cyclones (TCs), also known as hurricanes or typhoons, even though spectacular natural phenomena, can be extremely catastrophic and deadly (Emanuel 2003). In respect to this fact, for several years researchers have been trying to provide accurate early warnings based on predictions of the behavior of TCs. The overall objective is to heighten awareness in the communities residing along the hurricane's path. Additionally, accurate warnings are of great importance since the costs of an evacuation are huge, ranging from property protection expenses (Mangalindan 1996) to cessation in oil and gas production due to the evacuation of drilling rigs (Considine *et al.* 2004).

In order to achieve precise forecasts, researchers are trying to understand the behavior of tropical cyclones throughout their duration in every possible aspect. Mainly, the field has been divided into three distinct categories:

1. Estimating the probability of TC formation
2. Predicting storm intensity
3. Forecasting hurricane tracks

So far, researchers are capable of forecasting the track with 90% accuracy and up to 72 hours ahead (Aberson 2001), while intensity and formation forecasting accuracy is lagging far behind.

For this reason, in this paper we are going to examine the problem of forecasting the intensification of TCs using alternative data mining (DM) techniques. By alternative we mean techniques other than decision trees or instance-based learners like mining classification rules using particle swarm optimization (PSO) and association rules (AR). These methodologies will be applied to datasets with readings from hurricanes seasons between 1982 and 2003. We have formed the prediction of the intensification as a classification problem where we are trying to predict if the hurricane is going to intensify, abate, or maintain its current wind speed based on 16 independent features. Predictions will be made every 12 hours, for 12 hours ahead, starting at time 0 and finishing after 108 hours.

Our main objective is to produce accurate early warnings with respect to changes in the wind speed. This is especially practical, since the algorithms developed can be thought as a basis for an environmental management system (EMS) that will be capable of providing alerts when tropical storms tend to aggravate. Also, since we are dealing with a real world and very challenging problem, through this study we can provide an evaluation of the performance of these recently developed DM methodologies. Finally, the rules induced from hurricane related measurements can provide certain insight to how the hurricanes behave and help in the analytical formulation of the intrinsic properties of TCs.

The rest of this paper is organized as follows: Section 2 provides a review of the related work on this specific problem. Sections 3 and 4 discuss how to induce classification rules using PSO and using ARs respectively. Section 5 focuses on techniques for aiding the models at hand in order to improve their predictions regarding the behavior of TCs. Finally, the study ends with a comparison of the reported techniques and some concluding remarks and future directions of research.

Related Work

The problem of predicting tropical cyclone intensification has been investigated in both deductive and inductive ways. From the first perspective, researchers are involved with the

derivation of analytical models to forecast the maximum potential intensity (MPI) of hurricanes. That is the maximum intensity, in terms of speed, a tropical cyclone can achieve. In Camp and Montgomery (2000), the two most recent analytical models by Emanuel (1986-1997) and Holland (1997) are compared and their shortcomings are discussed. Unfortunately, there are factors that prevent hurricanes from reaching their MPI. This drawback, along with other reasons that intensity forecasting performs poorly, are discussed in (Wang & Wu 2004). Having this in mind, more accurate predictors should be investigated. Inductive learning and statistical inferencing are other practices to help overcome the above problems.

The best statistically based intensity forecast model is the one by DeMaria and Kaplan (1994; 1999). In their statistical hurricane intensity prediction scheme (SHIPS), the problem is posed as a regression problem, where the linear least squares method is applied to certain predictor variables. Despite of the simplicity of their algorithm, most of their work is focused on selecting appropriate predictors (i.e. independent variables) that would achieve smaller intensity errors. In our paper we use the same variables, which are described in Table 1, but attack the problem as classification using supervised learning. The limitations of classification to provide quantitative predictions can be easily overcome (for example by applying linear regression to the variables of a classification rule), as it is easy to translate regression into classification (e.g. by defining certain thresholds in the output).

Variable	Description
VMAX	Current maximum wind velocity (kt)
PER	The previous 12 hour change in maximum winds (kt)
ADAY	A scaled variable indicating hour far in time from peak of season
SPDX	The eastward component of the storm movement (kt)
PSLV	Vertical depth
VPER	Quadratic term: $VMAX \times PER$
POT	Potential strength
SHRD	The vertical shear of horizontal wind
T200	Temperature at 200 hPa surrounding the storm
EPOS	A measure of how conducive the atmosphere is to cloud formation
RHHI	The relative humidity in the middle of the atmosphere
Z850	A measure of how conducive the atmosphere is to cloud formation
LSHR	SHRD scaled by the sine of the latitude
D200	A measure of upward motion in the upper atmosphere
VSHR	Quadratic term: $VMAX \times SHRD$
POT2	Quadratic term: POT^2

Table 1: *The 16 independent variables used in our prediction models.*

Finally, a recent work (Tang, Yang, & Kafatos 2005) resembles very much what is presented in this paper, but has certain drawbacks. According to their approach, association rules in the form of classification rules are developed for predicting if a TC will intensify or not. Unfortunately, no quantitative results are given in the form of classification error and also no algorithm is described for the derivation of the final classifier. Their objective is more adjusted to find interdependencies between the independent variables and hurricane intensification and also in attribute selection based on the theory that governs tropical storms (they use some different variables than we do).

Swarm Intelligent Rule Discovery

In this section we explore a population-based search method for mining classification rules from the hurricane set.

Searching for Classification Rules

In order to classify data points in the hurricane set we are interested in discovering certain classification rules, or attribute tests, that assign class labels to measurement vectors. The idea is to recognize patterns in the data that may correspond to trends in class membership. For example, we may find that those measurements whose `VMAX` (current maximum wind velocity in knots) attribute lies between certain bounds tend to belong to hurricanes that will intensify in the future. In this case, we would like to say there is a *rule* in the form of a logical implication: $a \leq$ `VMAX` $\leq b \rightarrow$ `intensify`. Correct rules may then be applied to future measurements in order to classify novel storm data.

Classification rules have been used to predict class attributes since their inception with tree induction algorithms originally proposed by Feigenbaum and Simon (1962) as an attempt to simulate the processes by which humans learn concepts. Since this time they have primarily been applied to database applications as a method of classifying new elements based on rules built from a set of training instances. In uncomplicated datasets, relatively simple classification rules have been shown to perform well (Holte 1993).

Part of the attractiveness of this rather high level symbolic approach to concept learning is readability. A small number of high quality rules may prove useful for examining patterns in the dataset. With such devices, we may be able to better visualize the underlying model that produces the data. Unfortunately, the problem of mining classification rules from a potentially infinite domain is extremely difficult, especially in the presence of highly complex data. Tree induction methods like ID3 and C4.5 fall short when there may be interactions among the variables (Giuffrida, Chu, & Hanssens 2000) and computation time may be costly for high dimensional continuous data.

Changing perspectives slightly, we can formulate the problem of discovering classification rules as a multivariate nonlinear function optimization problem. Specifically, we are interested in searching through the infinite space of possible rules, climbing an evaluation surface that corresponds to how well each rule performs at classifying the example data. In this way, evolutionary algorithms have been applied

to search through a rule space and realize accurate class rules (Freitas 1998). However, encoding problems can occur when a genetic algorithm (GA) is required to search through a continuous space (as in the case of the hurricane set: we want to be able to find attribute bounds over the domain of real numbers).

PSO is a population based search inspired by the swarming behavior of insects, birds, and fish. Large populations of such animals are able to move efficiently in aggregation through space if each individual follows two simple rules: fly toward food or follow a neighbor. The individuals will wander *en masse* until one or more find food, after which the entire population will eventually converge on the food. This type of behavior is particularly attractive to computer science researchers interested in effectively moving a population of search points through an abstract space. Kennedy and Eberhart (1995) developed PSO inspired by simulations of aggregate social behaviors in the artificial life field.

Specifically, we randomly scatter the search space with a population of individual search points. Associated with each individual is a position vector $\tilde{x}$, a velocity vector $\tilde{v}$, and a fitness value f. Each individual has a "memory" insofar that it stores the position with the highest fitness it has experienced since the beginning of the algorithm, denoted P_{bst}. Each member of the population is also aware of the fittest individual in its neighborhood, G_{bst} (in this variant of PSO, this neighborhood is equal to the entire population). The algorithm consists of an iterative process in which we continually update the positions of each particle based on its velocity vector and the velocities of each particle based on its P_{bst} value and the G_{bst} value such that particles tend to steer toward the fittest particle in the population and the fittest region of the space they have each visited so far. The state transition equation is thus:

$$\tilde{v}_t = \chi\left(\tilde{v}_{t-1} + r_1\phi_1\left(P_{bst} - \tilde{x}_{t-1}\right) + r_2\phi_2\left(G_{bst} - \tilde{x}_{t-1}\right)\right)$$
$$\tilde{x}_t = \tilde{x}_{t-1} + \tilde{v}_t$$

where χ is the inertial coefficient, ϕ_1 and ϕ_2 are acceleration constants and $r_1, r_2 \sim U(0,1)$. The inertial coefficient adjusts how reactive a particle is to velocity changes, the acceleration constants adjust the factor by which particles tend to fly toward their P_{bst} and G_{bst} positions, and the random variables are used to introduce stochasticity into the transition. This stochasticity is necessary to jostle particles out of an equilibrium that occurs at points between P_{bst} and G_{bst} which are not necessarily optimal (Kennedy & Eberhart 1995).

The algorithm is run for a specified number of iterations, or until all particles are within a hyperspheric convergence radius of each other (Sousa, Silva, & Neves 2004). During each update, the fitness of the particles are evaluated and the P_{bst} and G_{bst} values are updated accordingly. When termination occurs, the fittest particle is returned: the search point that was found to have the highest fitness value.

An advantage PSO has over GAs is its relative finesse with real values. For instance, in a GA, if we encode real value arguments as a double precision floating point number in a standard IEEE 754 mantissa exponent representation, different changes to the bit string may result in disproportionate moves through search space depending on the significance of the bit changed. There are ways of dealing with this in the GA discipline (i.e. Gray codes) but PSO handles real arguments more naturally by representing each argument as an orthogonal direction in search space. Particles moving through this dimension at some velocity correspond to proportional changes in the argument.

To apply PSO to mine classification rules in the hurricane set we formulate the problem as a search for boundary values in each attribute test. Specifically, we want to find an upper and lower bound for each test of attribute x_i that maximizes some quality metric of the rule. There are several considerations that must be taken into account discussed in the following sub section.

Implementation of PSO for Rule Discovery

In order to develop a framework for a swarm based search for classification rules we must take several implementation issues into account. It is important to represent the rules and their evaluations in a compact and cogent manner to produce an efficient, effective algorithm. Inspiration was drawn from Sousa *et al.* (2004) who experimented with PSO to mine rules for databases in the UIUC repository.

Each rule is composed of two parts, the antecedent and the consequent. The antecedent is a conjunction of attribute tests $m(a_i, b_i, x_i)$ where x_i is the tested attribute or feature, and a_i and b_i are the value range boundaries for the test. Specifically, each attribute test $m(a_i, b_i, x_i)$ succeeds if the value of x_i lies between $a_i - b_i$ and $a_i + b_i$, otherwise the test returns false. The particles, then, are responsible for discovering good values for each a_i, b_i for $i = 1, \cdots, d$ where d is the dimensionality of the data space.

A true classification rule rarely contains a test for each attribute; we must employ a mechanism that allows the algorithm to choose the omission of any given test. This is an easy task for the GA domain: simply include an extra bit (gene) that corresponds to whether or not the test is performed (Freitas 1998). PSO, in contrast, does not allow for such adroit bitwise manipulation. A solution (Sousa, Silva, & Neves 2004) is to normalize the data to lie on the interval $[0, t]$ where $t < 1$ is a threshold value. During the course of the search, when one of the test bounds drifts above this interval, the attribute test is omitted (by always evaluating to true). This allows for regions of the rule space in which to explore corresponding to attribute exclusion. If the evaluation of a rule tends to be higher in some such regions, the preference will be for omitting those particular tests.

The consequent of a rule is simply a class label. For this implementation we search for one rule per class at a time (similar to what is termed the Michigan approach in the GA community (Freitas 1998)). We run the search for a specified number of iterations for a rule with one particular consequent, e.g., `intensify`. After the rule is found, we remove the instances of the `intensify` class in the data set that were classified correctly by that rule. The process is repeated until less than 10% of the `intensify` class remains (to avoid overfitting): each time adding the newly discovered rule to the set of rules with that particular consequent. We

then restore the data set and repeat the entire process twice more to find the `abate`, and `steady` rule sets.

Finally, in order to discover good rules, we must have an accurate and efficient evaluation of rule quality. We are interested in finding an evaluation that encompasses both a high number of correctly classified instances and a low number of incorrectly classified instances. To that end we define the rule quality function as the product of the true positive rate and the false negative rate:

$$Q(X) = \frac{TP}{TP + FN} \times \frac{TN}{TN + FP}$$

Therefore, the particles swarm the rule space, optimizing this function $Q(X)$.

Experiments

To test the performance of the PSO algorithm for rule discovery, each 12 hour hurricane data set was normalized on $[0, t]$ and divided into a 66% training and 34% test partitions after randomly permuting the points to prevent temporal bias between partitions. The PSO algorithm was then run on the training partitions for 300 iterations per rule with the following parameters: $\chi = 0.73, \phi_1 = \phi_2 = 2.05, t = 0.7$. The swarm population was set to 25.

The correctness of the discovered rules were then examined by calculating precision, recall, and accuracy on the test partitions for each twelve hour set.

Results

The algorithm successfully found classification rules for each 12 hour set. As shown in Figure 1 recall is extremely high, while precision is low: indicative of rules that are not very "discerning" and tend to over-classify for their consequent class. Accuracy is shown in Figure 2 and is plagued by low values. An interesting trend is the substantial *increase* in accuracy for intensification with time and a corresponding decrease for the steady class. The actual quality of the discovered rules remained low with values ranging between 0 and 0.54.

In each case, two to three rules per class were sufficient to cover 90% of the training set. Search time was less than an hour per rule.

Discussion

The performance, while not impressive, holds certain promise. In this set, the data is noisy and relatively complicated so it is difficult to obtain clear rules that effectively classify with a high accuracy. The rules that it did find were particularly strange and plagued with abnormally high ranges. A possible reason for this is that the particles are allowed to wander through parts of the rule space that may have no local improvement in quality. That is, a particular search point may wander on a plateau for some time allowing its boundary values to grow significantly large without changing rule quality. A solution to this problem would be perhaps to add a penalty to particles that wander too far out of the actual ranges for the data value.

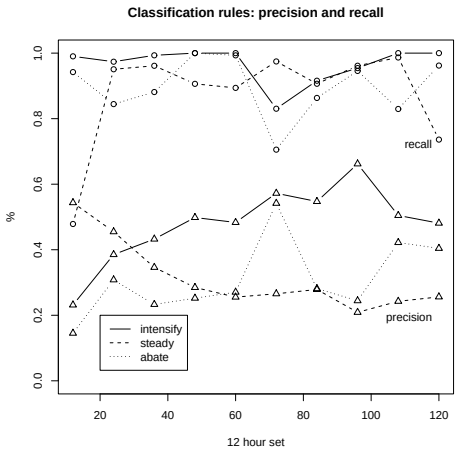

Figure 1: *Precision and recall of the rules per 12 hour set.*

As a classifier, the rule set did not perform as well as standard regression techniques such as linear and neural networks. The best performance of the algorithm was on the 96 hour dataset with an error of 39.93% (see Table 2). Linear and neural net regression achieved an error of 19% to 21% when predicting the sign of velocity change according to Anderson's work on an earlier data set with the same features. Interestingly, on this 96 hour set the rules perform the best while neural networks and linear regression seem to do much better on other sets (Anderson 2004).

Future work in this area would involve applying more direction to the search such as a penalty for out of range values and a more thoroughly tested choice of parameters. Validation should be tested using k-fold cross validation.

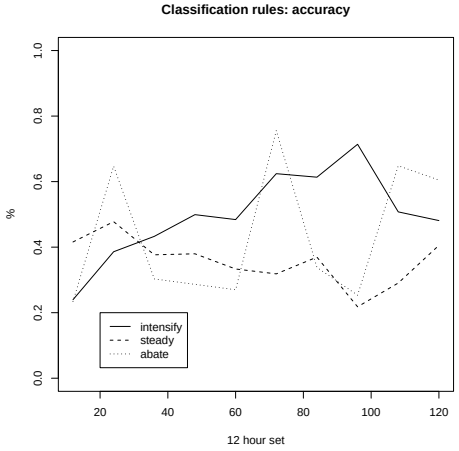

Figure 2: *Accuracy of classification rules per 12 hour set.*

Classification using Association Rules

In this section, we investigate another approach for mining classification rules for the TC intensity prediction, which is based on association rule (AR) discovery. Association rules (Agrawal, Imielinski, & Swami 1993) are mainly used for marketing purposes in order, for example, to attack the problem of market basket analysis. In this context, they are employed to discover associations between products with a variety of applications like creating catalogs, promoting sales, and arranging aisles to name a few.

Class association rules (CARs) were initiated by the work in (Srikant & Agrawal 1996), where ARs began to be applicable in both quantitative and qualitative features, something essential for DM purposes. As a complete algorithm, they were first introduced by Liu et al. (1998). The intuition behind CARs is to keep as the consequent of the ARs the target classes and then discover rules using the same algorithms for AR discovery. The motivations behind implementing CARs for this problem was that the results presented (Liu, Hsu, & Ma 1998) seemed promising with respect to other well known machine learning approaches. Furthermore, classification rules are always more expressive and easier to handle than decision trees and they are mined using other accuracy measures (support, confidence) that can be proven to be more effective than measures like accuracy or information gain. Additionally, they have been recently used for the same problem (Tang, Yang, & Kafatos 2005), but with no report on classification error results.

The high-level description of the algorithm for mining CARs is presented in Figure 3. First of all, the attributes are discretized using an entropy based discretization (Fayad & Irani 1993). Next, the apriori algorithm (Agrawal, Imielinski, & Swami 1994) is applied to mine all the CARs that satisfy certain user specified support and confidence levels. In order to deal with infrequent classes that impose the danger of getting cut-off the rule discovery due to small amount of representation in the dataset the mechanism of multiple minimum supports was implemented (Liu, Ma, & Wong 2001). Each class has a different minimum support based on the equation: $miniSup(C_i) = minSup \times freqDistr(C_i)$

Due to the above formula, frequent classes get higher support level than infrequent, ensuring production of rules for infrequent classes and decrease in the production of rules for frequent classes that could allow overfitting. As a final step, a classifier has to be built from the CARs and a default class must be set, since rules are neither mutually exclusive nor exhaustive. So the algorithm for the construction of a classifier should select a subset of rules, order them and assign a default class in a way that will imply an improvement on the classification error. Since the possible subsets of rules are 2^m, where m is the number of rules, it is infeasible to find an optimal subset. Thus, certain greedy and search approaches have been proposed. In our case we used two of them, the original one (Liu, Hsu, & Ma 1998), which is basically a heuristic approach, and the one used by C4.5 (Quinlan 1993) in order to transform a decision tree into a set of rules and is based on a variant of simulated annealing (SA) (Kirkpatrick, Gelatt, & Vecchi 1983).

The construction of the classifier in the greedy approach

```
1. trainingSet = discretize(trainSet);
3. freqs = findClassFreqs(trainSet);
2. CARs = Apriori(trainSet, freqs);
3. Classifier = buildClassifier(CARs);
```

Figure 3: *The algorithm for mining association rules for classification.*

(CAR+G), simply sorts the rules using their confidence level, breaking ties using their support level. After that it starts picking rules in decreasing order of confidence and adding them to the classifier, finding also a default class for the current classifier. At some point the error will increase, signaling the stop condition for the algorithm. The default class of the last added rule is used. In the other approach (CAR+SA), the classifier building algorithm tries to find subsets of rules for each class, using a variant of SA. Rules are picked randomly and are added to the subsets if they decrease the description length, measured in bits, of the theory (rules) along with their exceptions. A rule that increases the number of bits is added based on a probability, calculated according to the current temperature of the SA procedure. The above approach is based on the minimum description length (MDL) principle (Rissanen 1983), which states that the simplest classifier is the one that needs the minimum number of bits in order to be transferred through a communication channel. After the SA search is terminated, the subsets of rules are sorted according to their false positive rate. Sorting classes by the false positive rate ensures that earlier rules of a certain class will not misclassify to many instances, which could be correctly classified by later classes.

Evolving Class Association Rules

Even though entropy based discretization has proven to be of good quality, it is possible to do better by applying certain algorithms in the original format of the numerical attributes. An example of an approach that could work with continuous valued attributes for rule mining is DM using evolutionary algorithms (Freitas 1998). In this work we present two approaches for evolving CARs, by basically mutating the bounds of each attribute test and driving the quality of the rules using selection in the rule population. The format of the rules is depicted in Figure 4. For every attribute there is a tuple, which defines (a) if the attribute test is active (A) that is, if it is used as an attribute test of the rule (b) the lower (L) and (c) upper (U) bounds of the attribute tests.

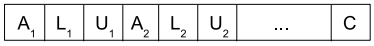

Figure 4: *Rule format for use with the evolutionary algorithm.*

In the first approach the evolution starts with a randomly generated initial rule population, in which the number of active attribute tests is intensified between runs. This simulates

the discovery of k-ruleitems, where k equals the number of antecedents in k runs. For every run we are trying to find rules that satisfy the prespecified support and confidence thresholds. These rules are kept for the later construction of the classifier. Selection is based on the confidence level of the rules. These rules are selected for the next generations, while rules with 0% confidence are discarded and replaced by new random rules. This procedure continues until a certain number of generations is reached. Intense mutation is applied in the lower and upper bounds of the attributes. The mutation operator randomly alters the bounds of the rules, trying to find better quality ones.

In the second approach (CAR+SA+GA), the basic difference is the initial population used. In this case the rule population are the rules of the classifier built using SA. Additionally, the measure that guides the data mining procedure and also the selection procedure is switched from support and confidence to precision times true positive rate:

$$fitness = \frac{TP}{TP + FP} \times \frac{TP}{TP + FN}$$

This formula for fitness has more information than formulas that simply use precision or accuracy by themselves, since it avoids rules that overfit the data or introduce problems with unbalanced classes. Additionally, a more "intelligent" mutation operator is applied, based on the following rules:

1. If the rule is too general, i.e. covers too many instances and a certain percentage of them leads to incorrect classification, then specialize the rule, i.e. shrink the coverage of the rule.
2. If the rule is too specific, i.e. covers too few instances, then generalize the rule, i.e. make the bounds of the attributes more broad.

A discussion about other fitness measures, operators and their implications can be found in (Freitas 1998).

Experiments

The performance of the above four approaches was tested on the hurricane dataset for all the hours. The dataset was divided into 66% for training and 34% for testing. The minimum support level was set at 1% and the confidence level was set to 50%. For evolving the CARs in the randomly generated population, 400 generations were used for each of the 16 runs, starting and maintaining 80 rules per class in the population. For the CAR+SA+GA approach, again 400 generations were used while the number of rules in the initial population varied according to how many rules were in the classifier. The results are reported according to the classification error. The first evolutionary approach produced poor results in the area of 40-55% of classification error in all cases. The errors of the other approaches can be found in Figure 5.

To compare the technique of classification using association rules, we chose also two well known classifiers and compared them with the best developed classifier from our study, CAR+SA+GA and the one induced using PSO. The other two approaches were C4.5 (Quinlan 1993) and Naïve Bayes (Duda & Hart 1973), which were applied to the datasets using the WEKA machine learning tool (Witten & Frank 2000). The results are reported in Table 2.

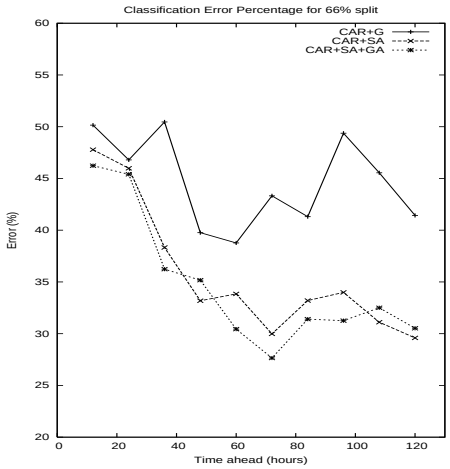

Figure 5: *Classification error between the classification using the different association rules approaches.*

Hour	C4.5	NB	CAR+SA+GA	PSO
12	**42.51**	44.27	46.23	75.64
24	**42.95**	44.47	45.40	59.77
36	36.49	39.92	**36.23**	52.42
48	38.11	**34.52**	35.17	64.86
60	36.22	34.18	**30.44**	51.86
72	31.14	30.75	**27.65**	43.46
84	30.54	**29.67**	31.40	50.00
96	32.25	34.73	**31.26**	39.93
108	**29.72**	30.27	32.50	42.46
120	**28.34**	32.71	30.52	40.43

Table 2: *Comparison with other classifiers. Bold format characterizes the winning approach.*

Discussion

These are to our knowledge the first quantitative results for the hurricane intensification problem when formulated as a classification problem. From the above results, we observe that classification using association rules is indeed a very competitive and comparable approach to other well known techniques. Moreover, the C4.5 style classifier performed extremely better than the greedy one. Even though the greedy one is faster and deterministic in the construction of the classifier, we believe that the additional time and the small variance in the quality of the classifier using SA and the MDL principle paid off by large amounts in the final results. As for the evolution of the CARs, in many cases it boosted their performance by tweaking their bounds. It is still an open question to us whether this tweaking is actually helping with the classifier's error performance or just

overfits the rules based on the training set. It is perhaps the case that with "smarter" operators better quality rules could be found on a constant basis, but this remains to be investigated. The main issue is that it is better to have a mechanism for initializing the rule population with good quality rules than random initialization when trying to evolve classification rules. Both the running time and the quality of the rules improve greatly. This idea could also be applied to the starting position of the particles when using PSO for rule mining.

Aiding the Approaches

Besides having DM algorithms that process the data in a straightforward way, one can preprocess and postprocess the datasets and the derived classifiers by using, for example, feature selection and combining models respectively.

Feature Selection

We used several different feature selection algorithms from the WEKA tool and then applied the selected features to the classifiers. Both ranking (ReliefF, PCA, Information Gain, Gain Ratio) and filtering approaches (Best First, Genetic Search) were tried. In the first approach the attributes are ranked according to a criterion and the user can select among them in this order. In the second approach the algorithms search for a subset of features that minimize the classification error. Unfortunately, the results were not that promising. In most cases we had an improvement of less than 5% in classification error, but in other cases the performance decreased, another indication that the datasets at hand are indeed resistant to pattern discovery. The main problem is that the dataset is already preprocessed to contain variables that improve the error when using regression. Also the information gain criterion discovered that the attributes are not very informative (the best attributes had an information gain of 0.2-0.3 bits and the worst less than 0.01 bits). That explains the large classification error of CARs and C4.5 that have as a main component information gain for discretization and tree building respectively. Table 3 has the results from the previously mentioned approaches, when the 10 best attributes were selected using the ReliefF algorithm. They are compared against the previously best (PB) classification error from Table 2.

Combining Models

DM models are procedures driven by a bias in the form of what someone is trying to discover. So different models can result in differences in how they classify future data, even though the same training set is used. It is often the case that the coalition of models provides better classifiers by minimizing their disadvantages. Having this in mind several schemes like boosting, voting, and bagging have been developed for this purpose (see (Witten & Frank 2000) for an overview).

In relation to our case, it is very easy to aggregate the two classifiers (CARs and PSO rules) using a method that was developed in order to help CARs using other techniques (Liu, Ma, & Wong 2001). Basically, the algorithm has a main classifier and other back up classifiers and tries to evaluate the error performance of the rules in the main classifier with respect to the other classifiers. If their performance is less efficient the primary rules are replaced by the best back up classifier for this specific rule. Since the PSO classifier was weak in predictive accuracy, we used the WEKA package to implement such a cooperation as an example of further experimentation. Additive logistic regression (Friedman, Hastie, & Tibshirani 1998) (Logit Boost in WEKA - LB) was used (without feature selection), having as unit classifiers decision stumps. The results can be found in Table 3. In most of the cases, boosting has given the best performance, even when attribute selection was used for the other classifiers, indicating the power of combining models.

Hour	PB	CAR+SA	C4.5	NB	LB
12	42.51	48.19	40.55	44.27	**36.11**
24	42.95	45.16	42.60	42.72	**37.82**
36	**36.23**	36.36	38.47	41.50	37.28
48	34.52	35.27	37.51	32.73	**31.53**
60	30.44	30.10	34.52	33.50	**29.51**
72	27.65	**25.33**	28.82	28.62	26.88
84	29.67	31.20	30.10	30.10	**26.15**
96	31.26	30.76	**28.53**	35.73	30.27
108	29.72	31.11	**25.00**	30.55	25.27
120	28.34	28.34	27.72	33.95	**26.16**

Table 3: *Comparison of several classifiers using feature selection with the Logit Boost algorithm.*

Conclusion

In general the overall project was very interesting and helped us obtain a deeper understanding of supervised learning algorithms for classification. Moreover, we were introduced to several techniques of machine learning such as particle swarm optimization, association rules, entropy based preprocessing techniques, feature selection methodologies and classifier combination schemes.

As far as the TC intensification problem is concerned, we confirmed previously derived results in the research community that, though a problem of essential value, it is very hard. It was previously attacked as a regression problem or with analytical formulations. This is a new approach that tries to facilitate the solution but encounters the bottlenecks of the previous approaches as well. Among the difficulties is that the classes vary greatly in representation between datasets and within the same dataset; it is often the case that classifiers will find more accurate rules in the most frequent classes since the exemplars can provide more information regarding these classes. Also, such rules could dominate those for the infrequent classes. It is possible that these particular variables cannot represent the state of the hurricane completely and that temporal information could help capture the dynamics of hurricanes.

Overall, the two approaches differ very much with PSO having more degrees of freedom in the search, while CAR performs a more exhaustive search based on certain criteria.

So it is often the case that CAR would find better quality rules, which is reflected in the results reported.

The major shortcoming of the rules found by PSO is that they are not very discerning. The search points need to be more constrained: there seems to be a lack of gradient information in large parts of the space where points are forced to wander on plateaus without making improving moves and producing uselessly large values.

For future work, we would like to investigate importing temporal information in the derived models. Also it would be interesting to compare the derived classification rules with the existing MPI models. It would also be interesting to incorporate more attributes like the ones described in (Tang, Yang, & Kafatos 2005) and let the DM algorithms find more interesting patterns with a bigger collection of features at their disposal.

References

Aberson, S. D. 2001. The ensemble of tropical cyclone track forecasting models in the north atlantic basin (1976-2000). *Bulletin of the American Meteorological Society* 82(9):1895–1904.

Agrawal, R.; Imielinski, T.; and Swami, A. 1993. Mining associations between sets of items in massive databases. In *Proceedings of the ACM-SIGMOD 1993 International Conference on Management of Data*.

Agrawal, R.; Imielinski, T.; and Swami, A. 1994. Fast algorithms for mining association rules. In *Proceedings of the 20th International Conference on Very Large Databases*.

Anderson, C. W. 2004. Predicting storm intensity 12 to 120 hours ahead with linear models and committees of neural networks. Unfinished.

Camp, J. P., and Montgomery, M. T. 2000. Hurricane maximum intensity: Past and present. *Monthly Weather Review* 129(7):1704–1717.

Considine, T. J.; Jablonowski, C.; Posner, B.; and Bishop, C. H. 2004. The value of hurricane forecasts to oil and gas producers in the gulf of mexico. *Journal of Applied Meteorology* 43(9):1270–1281.

DeMaria, M., and Kaplan, J. 1994. A statistical hurricane intensity prediction scheme (ships) for the atlantic basin. *Weather and Forecasting* 9:209–220.

DeMaria, M., and Kaplan, J. 1999. An updated statistical hurricane intensity prediction scheme (ships) for the atlantic and eastern north pacific basin. *Weather and Forecasting* 14:326–337.

Duda, R., and Hart, P. 1973. *Pattern classification and scene analysis*. Wiley.

Emanuel, K. 2003. Tropical cyclones. *Annual Review Earth Planetary Sciences* 31:75–104.

Fayad, U. M., and Irani, K. B. 1993. Multi-interval discretization of continuous-valued attributes for classification learning. In *Proceedings of IJCAI-93*.

Feigenbaum, E. A., and Simon, H. A. 1962. Simulation of human verbal learning behavior. *Communications of ACM* 5(4):223.

Freitas, A. A. 1998. *Data Mining and Knowledge Discovery with Evolutionary Algorithms*. Berlin: Springer-Verlag.

Friedman, J.; Hastie, T.; and Tibshirani, R. 1998. Additive logistic regression: A statistical view of boosting. Technical report, Stanford University.

Giuffrida, G.; Chu, W. W.; and Hanssens, D. M. 2000. Mining classification rules from datasets with large number of many-valued attributes. In C. Zaniolo et al., ed., *Advances in Database Technology - EDBT 2000, 7th International Conference on Extending Database Technology Proceedings*, volume 1777 of *Lecture Notes in Computer Science*, 335–349. Springer.

Holte, R. C. 1993. Very simple classification rules perform well on most commonly used datasets. *Mach. Learn.* 11(1):63–90.

Kennedy, J., and Eberhart, R. 1995. Particle swarm optimization. In *Proceedings of the International Conference on Neural Networks*, 1942–1948.

Kirkpatrick, S.; Gelatt, C. D.; and Vecchi, M. P. 1983. Optimization by simulated annealing. *Science* 220:671–680.

Liu, B.; Hsu, W.; and Ma, Y. 1998. Integrating classification and association rule mining. In *Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining*.

Liu, B.; Ma, Y.; and Wong, C.-K. 2001. Classification using association rules: Weaknesses and enhancements. In V. Kumar et al., ed., *Data mining for scientific applications*.

Mangalindan, M. 1996. How much does it cost to prepare for a hurricane? $670,000 per mile . . . at least. *The Virginian-Pilot*.

Quinlan, R. J. 1993. *C4.5: Programs for machine learning*. Morgan Kaufmann.

Rissanen, J. 1983. A universal prior for integers and estimation by minimum description length. *Annals of Statistics* 11(2):416–431.

Sousa, T.; Silva, A.; and Neves, A. 2004. Particle swarm based data mining algorithms for classification tasks. *Parallel Comput.* 30(5-6):767–783.

Srikant, R., and Agrawal, R. 1996. Mining quantitative association rules in large relational tables. In *Proceedings of the ACM-SIGMOD 1996 Conference on Management of Data*.

Tang, J.; Yang, R.; and Kafatos, M. 2005. Data mining for tropical cyclone intensity prediction. *Sixth Conference on Coastal Atmospheric and Oceanic Prediction and Processes*.

Wang, Y., and Wu, C.-C. 2004. Current understanding of tropical cyclone structure and intensity changes - a review. *Meteorology and Atmospheric Physics* 87:257–278.

Witten, I. H., and Frank, E. 2000. *Data Mining: Practical machine learning tools with Java implementations*. Morgan Kaufmann.