Applied Intelligence 19, 157–170, 2003

Data Mining for Generating Predictive Models of Local Hydrology

RATTIKORN HEWETT*
School of Electrical Engineering and Computer Science, Washington State University at Vancouver, 14204 NE Salmon Creek Avenue, Vancouver, WA 98686-9600, USA
rhewett@vancouver.wsu.edu

Abstract. The problem of downscaling the effects of global scale climate variability into predictions of local hydrology has important implications for water resource management. Our research aims to identify predictive relationships that can be used to integrate solar and ocean-atmospheric conditions into forecasts of regional water flows. In recent work we have developed an induction technique called *second-order table compression*, in which learning can be viewed as a process that transforms a table consisting of training data into a second-order table (which has sets of atomic values as entries) with fewer rows by merging rows in consistency preserving ways. Here, we apply the second-order table compression technique to generate predictive models of future water inflows of Lake Okeechobee, a primary source of water supply for south Florida. We also describe SORCER, a second-order table compression learning system and compare its performance with three well-established data mining techniques: neural networks, decision tree learning and associational rule mining. SORCER gives more accurate results, on the average, than the other methods with average accuracy between 49% and 56% in the prediction of inflows discretized into four ranges. We discuss the implications of these results and the practical issues in assessing the results from data mining models to guide decision-making.

Keywords: data mining, machine learning, inductive technique, decision tables

1. Introduction

The ability to foresee trends and extreme events in local climates has important economic and social consequences. Effective management of regional water supplies, as well as other human and natural resources, requires decisions that are informed with forecasts of local climates. More accurate forecasts produce better decisions and increased lead times for interventions.

Earth's climate, however, is a complex dynamic system in which accurate predictions are notoriously difficult to achieve. Climate forecasts predict shifts in atmospheric conditions that may persist for months, seasons or centuries. Small shifts in global atmospheric conditions can produce significant changes in the expected means and extremes of meteorological variables for regional climates. Research in climatology has produced significant advances in climate prediction using both dynamic models and empirically induced models based on a vast store of observational data. Such models are now able to predict climate changes in a one to ten year time frame over two to five degree grids over the earth's surface. However, for purposes of regional resource management, predictions on a finer scale are required.

One way to approach the problem is to extend existing coarse grained dynamic models of the global climate to the spatial and temporal scales needed to predict local climate conditions. While these models are valuable, they require a high level of expertise to create, are computationally expensive to run, and may not be accurate enough for resource management decisions. Furthermore, no existing climate model integrates the effects of solar variability into ocean-atmospheric coupled models. A second approach is to apply statistical methods to derive predictive relationships between global climate factors and local climate variability.

*Work was performed when the author was at the Institute for Human and Machine Cognition, University of Florida.

This is exceedingly difficult because of complicated and nonlinear interrelationships between climate factors and the complexity of their relationships, which may violate statistical assumptions, to local hydrology. Both techniques are referred to as *downscaling*. We take another approach to the problem by using data mining techniques to discover predictive relationships that can be used to integrate global scale climate variability including solar and ocean-atmospheric conditions into forecasts of regional water flows.

One of our research objectives is to evaluate the performance of a relatively new induction technique, *(second-order) table compression*, on a complex real-world problem. This paper reports our efforts to construct predictive models of future water inflows of Lake Okeechobee, a primary source of water supply for agriculture and urban areas of south Florida. We apply table compression induction to this problem and compare the results with those obtained from three well-established data mining techniques: neural networks, decision tree learning and associational rule mining. In particular, we use SORCER (**S**econd-**O**rder **R**elation **C**ompression for **E**xtraction of **R**ules), the second-order table compression learning system, and two state-of-the-art data mining systems: C4.5 [1], and CBA [2]. We choose C4.5 and CBA for comparison because they are two of the best performing systems that build predictive models which can be expressed as rules. As a representative decision tree learning technique, C4.5 is one of the most refined and successful data mining systems. It is widely used because of its high accuracy and simplicity [3, 4]. Based on association rule mining techniques [5], CBA (**C**lassification **B**ased on **A**ssociation) is reported to outperform C4.5 in several domains [2]. While the models produced by neural nets [4, 6] are less comprehensible than those produced by the other techniques, neural nets are well known for robustness and the ability to represent complex models. This distinction motivated us to include the neural net approach in our comparisons. In particular, we use the results obtained from a neural net model using a popular back propagation technique [6] as reported in [7, 8].

Since second-order table compression technique is a recent development, we give, in Section 2, a brief overview of the technique and its implementation in SORCER together with the compression algorithms employed in the experiments. Section 3 describes the problem domain and characteristics of the experimental data. Section 4 describes the data preprocessing and methodology used in our experiments. Experimental results and practical issues are discussed in Sections 5 and 6, respectively. Section 7 discusses related work and is followed by a summary of our results in Section 8.

2. Second-Order Table Compression

Second-order table compression is an induction technique that abstracts classification rules from data sets represented as *second-order decision tables*. The theoretical framework presented in [9, 10] defines second-order decision tables to be database relations in which tuples (rows) have *sets* of atomic values as components (entries). In contrast, the rows in traditional first-order tables have atomic values as components. Using sets of values interpreted as disjunctions, the second-order table framework provides compact representations that facilitate efficient management and enhance comprehensibility. A second-order table represents missing data by the empty set, which is interpreted as the (unknown) *null value*, denoted by "–". A second-order table represents the first-order table consisting of rows that can be obtained by taking some row of the second-order table, replacing each of its nonempty entries by one of its elements and replacing the empty set by "–". For example, Fig. 1 shows a first-order decision table, Table 1, with 10 rows, and a five row second-order decision table, Table 2 that represents it. Row 1 of Table 2 contains the same information that is in rows 1 and 2 of Table 1.

Table 1

	Month	*Sunspot*	*Cp*	*Nino3*	*inflow*
1	jan	vweak	low	neutral	dry
2	feb	vweak	low	neutral	dry
3	nov	vweak	low	weak	dry
4	dec	vweak	low	weak	dry
5	jun	weak	high	-	vwet
6	jul	weak	high	-	vwet
7	aug	weak	high	strong	vwet
8	aug	weak	high	vstrong	vwet
9	sep	weak	high	strong	vwet
10	sep	weak	high	vstrong	vwet

Table 2

	Month	*Sunspot*	*Cp*	*Nino3*	*inflow*
1	{jan,feb}	{vweak}	{low}	{neutral}	{dry}
2	{nov,dec}	{vweak}	{low}	{weak}	{dry}
3	{jun,jul}	{weak}	{high}	∅	{vwet}
4	{aug}	{weak}	{high}	{strong,vstrong}	{vwet}
5	{sep}	{weak}	{high}	{strong,vstrong}	{vwet}

Figure 1. First-order vs. second-order decision tables.

The motivation behind the use of the second-order table representation for rule induction is to abstract a predictive model that is relatively easy to understand as well as accurate. Although tabular representations such as decision tables are easy to interpret, sparse and large tables can still be difficult to comprehend. Learning in the second-order table framework can be viewed as decision table compression in which an original table, representing training data, is repeatedly transformed into a table with fewer and more general rules (i.e., rules covering more conditions) by merging rows in ways that preserve consistency with the training data. Unlike most other learning systems, the table compression algorithm attempts to produce the shortest decision table consistent with the training data, a biased justified by Occam's razor [9]. Shorter tables can enhance model comprehensibility. We define terminology and important concepts in Section 2.1 and give an overview of a basic induction scheme in Section 2.2. Section 2.3 describes SORCER and the specific compression algorithms used in this paper. More details on the table compression technique and its theoretical results are presented in [10–12].

2.1. *Preliminaries*

We use the terms table (relation) and row (tuple or rule) to refer to the second-order structures, which we now define. *Rows* are mappings defined on a set of *attributes* (problem variables). The value of row r at attribute A (the A-component of r), denoted $r(A)$, is a subset of A's *domain* (the values which it may assume). A *table* is a set of rows and the *scheme* of a row or a table is the set of attributes on which it is defined. The partial ordering *covers* on the set of all rows (over a fixed scheme) is component-wise set inclusion, i.e., row s is covered by r if $s(A) \subseteq r(A)$ for each attribute A. *Flat rows* are those whose components are either singletons or empty. The *flat extension of table* R is the set of all flat rows covered by at least one row in R. A table S is said to *subsume* table R if the flat extension of R is a subset of the flat extension of S. Two tables are *equivalent* if each subsumes the other. A transformation of a table R into a table S is *equivalence preserving* if R is equivalent to S.

A *decision table* has a scheme consisting of *condition* attributes and a *classification* attribute. It represents a (partial) function that assigns classifications to conditions. The *classification* of a *condition* c (a row whose classification entry is empty) by decision table T, denoted $T(c)$, is the union of the classifications of all rows of T that cover the condition. A *simple condition* is a condition with singleton values for all condition attributes. A decision table is *consistent* if it associates at most one classification to any simple condition. A decision table is *complete* if it classifies (gives a nonempty value to) all simple conditions. A transformation of a table R into a table S is *consistency-preserving* if (1) every simple condition classified by R is given the same classification(s) by S, and (2) for any simple condition c not classified by R, $|S(c)| \leq 1$. Clearly, equivalence preserving operations are consistency preserving.

2.2. *Induction as Table Transformations*

The basic table compression algorithm starts with a flat table of training examples. The table is repeatedly transformed to produce another, usually shorter, table which subsumes and is consistent with the original table but is more general (covers more conditions). At each step, the table is an approximation of the unknown target function. The transformations correspond to a search, through the hypothesis space of second-order tables, for a suitable approximation of the target function. The induction stops when no further "consistency-preserving" transformations are found. Two types of transformations that may be used are: *equivalence-preserving* and *consistency-preserving*. Examples of equivalence preserving transformations include:

- *delete directly covered rule*: remove a row from a table if it is covered by some other row in the table
- *delete redundant rule*: remove a row from a table if it is subsumed by the other rows of the table
- *merge joinable*: replace a pair of rows, which agree on all attributes except one, by their *join* (component-wise union). Such a pair is referred to as a *locally joinable* pair.

An example of a consistency preserving transformation is *merge consistent*: merge a pair of rules whose join does not introduce inconsistency. Such a pair is said to be *consistently joinable*, and their merge may add new conditions, generalizing the table, without creating inconsistency.

To illustrate our table compression technique, consider the training data given in Table 1 of Fig. 1. Table 1 can be represented in a second-order flat table T (by replacing each table entry as a singleton set of the entry). By applying merge joinable to T on Rows i and $i+1$, for odd integers i from 1 to 9, T is compressed into Table 2 in Fig. 1. By applying merge joinable to Rows 4 and

Table 3

	Month	*Sunspot*	*Cp*	*Nino3*	*inflow*
1	{jan,feb}	{vweak}	{low}	{neutral}	{dry}
2	{nov,dec}	{vweak}	{low}	{weak}	{dry}
3	{jun,jul}	{weak}	{high}	∅	{vwet}
4	{aug,sep}	{weak}	{high}	{strong,vstrong}	{vwet}

Table 4

	Month	*Sunspot*	*Cp*	*Nino3*	*inflow*
1	{jan,feb,nov,dec}	{vweak}	{low}	{neutral, weak}	{dry}
2	{jun,jul,aug,sep}	{weak}	{high}	{strong,vstrong}	{vwet}

Figure 2. Examples of transformations on second-order decision tables.

5 of Table 2, Table 2 is further compressed to Table 3 in Fig. 2. Merge joinable is an equivalence-preserving transformation and thus, all the tables generated so far are equivalent, i.e., they contain exactly the same information. Rows 1 and 2 of Table 3 can be consistently merged into Row 1 of Table 4. Similarly, Row 2 of Table 4 is the result of consistently merging rows 3 and 4 of Table 3. Table 4 is more general than Table 3 in that it covers more conditions than Table 3. For example, Row 1 of Table 4 represents the new information: If (*Month* = {nov, dec}) ∧ (*Sunspot* = vweak) ∧ (*Cp* = low) ∧ (*Nino3* = neutral) then (*inflow* = dry), which is not in Table 3 and is not inconsistent with Table 3. On the other hand, the join of Rows 2 and 3 of Table 3 would result in a rule s: If (*Month* = {nov, dec, jun, jul}) ∧ (*Sunspot* = {vweak, weak}) ∧ (*Cp* = {low, high}) ∧ (*Nino3* = weak) then (*inflow* = {dry, vwet}). Rule s includes (covers): If (*Month* = nov) ∧ (*Sunspot* = vweak) ∧ (*Cp* = low) ∧ (*Nino3* = weak) then (*inflow* = vwet), which is inconsistent with Row 2 of Table 3. Thus, Rows 2 and 3 of Table 3 are not consistently joinable. In this example, the 10 rows of Table 1 are compressed into the two rules of Table 4. Table 4 is more general than and is consistent with Table 1.

Since many decision problems involving second-order relations (e.g., determining whether a table covers a tuple) are NP-hard [11], resource constraints (e.g., number of iterations) may be applied during compression for operations that are likely to be prohibitively expensive. Table compression uses an inductive bias that prefers short and dense[1] relations. Heuristics based on domain knowledge, such as ranking of attributes in order of discriminatory power, could also be applied to help select appropriate operation or rows.

2.3. *SORCER and the Algorithms Applied*

A table compression learning system, SORCER, has been implemented in C++ using the C++ Standard Library (STL). SORCER facilitates interactive experimentation and allows the user to define a compression algorithm (using standard transformations) and test procedure by writing a "script" file of commands. Rules are implemented as STL bitsets, and therefore, basic operations on rows have an efficient underlying implementation based on bit vectors. SORCER provides a comprehensive set of operations for manipulating relations which supports induction, querying and checking for consistency and redundancy. Details of computational issues and other theoretical results are discussed in [10–12].

SORCER provides a method to merge consistently joinable rows in an order determined by a "distance" function. Given a distance limit l, rows whose distance is no more than l can be merged repeatedly until each row in the table is more than distance l from every other row. This operation is invoked by the command *merge close consistently joinable l*, where l specifies a distance limit. The operation invoked by *merge all close consistently joinable l* applies the preceding operation repeatedly with distance limits ml for $m = 1, 2, 3, \ldots$, until $ml \geq 1$. The function used in our experiments for the distance between tuples r and s, denoted by $d(r, s)$, is defined by:

$$d(r, s) = \frac{1}{|\mathcal{C}|} \sum_{A \in \mathcal{C}} \frac{(|dom(A)| - |(r \sqcap s)(A)|)g(r, A)g(s, A)}{|dom(A)|^3},$$

where $\mathcal{C}$ is the set of condition attributes, $dom(A)$ is the *domain* of A (i.e., the set of values for A), $r \sqcap s$ is the component-wise intersection of r and s, and $g(t, A) = \min\{|dom(A)|, |dom(A)| + 1 - |t(A)|\}$, for a tuple t and attribute A. (Note that $d(r, s)$ is not a "distance" function in the usual sense, e.g., $d(r, r) \neq 0$.) The use of this distance function favors merging rows with large components that share many values.

For the experiments in this paper, we apply three simple compression algorithms: S1, S2, and S3 described by the commands used in a SORCER script file as shown in Fig. 3. Since the order of training data

Compression S1:	**Compression S2 (*l*):**	**Compression S3 (*p*):**
begin	*begin*	*begin*
Shuffle;	Shuffle;	Shuffle;
Merge consistent	Merge joinable;	Add high probability rows *p*;
end	Merge all close consistently joinable *l*	Merge joinable;
	end	Merge consistent
		end

Figure 3. Three compression algorithms.

may affect the result of compression, for fairness, we shuffle the table before compression. S1 merges pairs of consistently joinable rows until no more consistent joining is possible. S2 first merges locally joinable pairs, until no more such joins are possible, and then merges "close" consistently joinable rows using *merge all close consistently joinable l* (as described above). By applying merge joinable before merge consistent, S2 attempts to give priority to generalization according to the structure of the knowledge partially formed by equivalence preserving transformation of a training data set. S3 is slightly different from S1 and S2. By invoking the operation "add high probability rows p", S3 first adds certain rows whose minimum probability of being correct (on the training data) is p. Specifically, if $\#(X)$ is the number of times that event X occurs in the data set, then *add high probability rows p* causes SORCER to add all rows of the form: $A = a \rightarrow B = b$ such that $\#(A = a \wedge B = b)/\#(B = b) \geq p$. S3 next merges locally joinable pairs, until no more joins are possible, and then merges pairs of consistently joinable rows until no more consistent joining is possible. Unlike S1 and S2, by adding rules based on class distribution ratios, S3 may contain rules that are not consistent with the original training set. The "add high probability rows p" command allows users to add simple rules based on frequency distributions of the attributes. In fact, p is commonly known as the rule "confidence" in association rule mining [5].

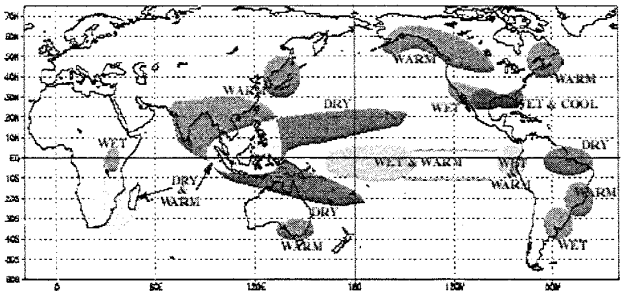
The rule set produced by the algorithm may not be complete. For conditions not covered by the model, a rule in the model is heuristically selected to provide a classification. The heuristics include a preference for rules that (1) cover the query on the most attributes, (2) cover fewer conditions, and (3) give the most common classification appearing in the table. Specifically, to apply the classifier created by the compression algorithm to a test example, the table is searched for a rule that covers the example on all condition components. If such a rule is found, the value of its class component is used, otherwise the first rule covering the test condition on a maximum number of attributes is used. If no rule in the classifier covers the test condition on any attribute, the class which appears most often (in the classifier) is used as the class for the condition.

3. Climate Variability

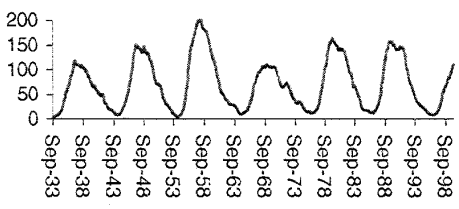
3.1. Ocean-Atmospheric Phenomena and Currents

Florida's climate has its most significant statistical association with the *El Niño-Southern Oscillation* (ENSO)[2] process during the winter months. During El Niño events (La Niña events), greater (less) than normal winter rainfalls are expected in Florida [7]. Figure 4(a) illustrates global effects of El Niño events between December and February (from the National Oceanic Atmospheric Administration). While current understanding points to ENSO phenomena as the underlying cause of climate variability globally and in many regions, the mechanism that bolsters these ENSO driven climate variations are not completely understood. The strength and phase of ENSO can be measured from sea surface pressure anomalies (e.g., SOI—Southern Oscillation Index) and SSTA—sea surface temperature anomalies (e.g., the nino indices: nino1&2, nino3, nino3.4, and nino4).

Abrupt shifts in global climate may also be caused by variations of ocean conveyor currents. In particular, the strength of *Atlantic Ocean Thermohaline Current* (AOTC) may have effects on tropical activity and climate fluctuation. Pacific Ocean SSTAs have the highest correlation to south and central Florida rainfalls (November–April) as compared to rainfalls in other parts of the United States mainland. The *Pacific*



(a) Precipitation anomalies during El Niño in winter



(b) Sunspot cycles

Figure 4. Examples of global scale climate variability.

Decadal Oscillation (PDO) and *North Pacific* (NP) indices that represent the state of northern Pacific Ocean also have an effect. In fact, Florida rainfall, Pacific Basin climate and ocean indices are all correlated.

3.2. *Solar Activity*

Solar activity affects earth's climate on several temporal scales. Figure 4(b) illustrates a periodic pattern of sunspot numbers from 1933 to 1996. In addition to short-term solar eruptive activity and the 11- and 22-year cycles of sunspot activity, there are long-term cycles of 80 to 100 years or longer. Penetration of the solar wind into the earth's magnetic field can produce strong spot heating of the earth's atmosphere and disruption of the zonal weather circulation [13]. As a result, even without appreciable long-term changes in magnetic energy, solar eruptions can cause significant climate fluctuations. Periods of high solar activity have been associated with a probabilistic shift towards wetter conditions in Florida especially during the tropical season. While past studies focused on the sunspot numbers as an indicator of solar activity, there is no reason to believe that sunspot numbers are a better indicator than other measures such as geomagnetic activity. The complexity of the solar-terrestrial and oceanic-atmospheric interaction makes it difficult to apply a traditional statistical approach to forecasting of regional climate.

4. Model Generation

4.1. *Data Sources and Data Preprocessing*

Relevant data are available on-line from various national research centers including the National Oceanic Atmospheric Administration (NOAA), the Atlantic Oceanographic and Meteorology Laboratory,[3] the National Climate Data Center and Climate prediction Center,[4] the National Geophysical Data Center at U. of Colorado at Boulder, and U. of Washington.[5] Estimated monthly lake inflow values were obtained from the United States Army Corps of Engineers [14] and the South Florida Water Management District (SFWMD).

Climate indices that influence Florida include global atmosphere-ocean phenomena (e.g., ENSO as measured by nino indices), ocean currents and oscillations (e.g., AOTC, PDO, Arctic Oscillation, and the Quasi-Biennial Oscillation), and solar variability (e.g., sunspot cycles and geomagnetic activity). For the experiments in this paper, solar activity is characterized by three variables: sunspot number and the Aa and Kp indices, surrogated measures for geomagnetic activity. Sunspot numbers represent slower less abrupt changes of solar energy affecting the earth's climate systems, while the Aa and Kp indices represent solar eruptive activity. According to Willet [13], the Aa-index is the best indicator of solar flare activity. Data sets of climate indices and lake inflows were supplied by SFWMD. Additional derived attributes are also included (e.g., sunspot trend, and smoothing averages of Aa and Kp index over various months to represent solar effect in different time scales). Based on prior knowledge, our domain expert selected 22 attributes (out of 42) that have relatively high correlation with the class attribute values and are likely to be relevant and added three derived attributes (moving averages of accumulated inflows, etc.). For attribute values that required discretization, we divided continuous data into discrete ranges using boundaries determined by our domain expert. Although in this paper we rely on expert knowledge, a variety of techniques for feature selection [15–17] and discretization [18, 19] could be used. SORCER provides operations to project a table onto a subset of attribute from its scheme, and this feature can be used to experiment with different sets of features, e.g., chosen by statistical analysis.

Data collection and preprocessing resulted in a data set containing 1054 rows of monthly data for the period from 4/1912 through 1/2000. The attribute domains range in size from 2 to 12. The class attribute represents accumulated lake inflow over the next six months with four possible values: dry, average, wet and very wet. To obtain a consistent decision table, SORCER resolves inconsistencies in the training data before compression as follows. If a condition is associated with two classes, the example with the class that occurs less frequently for the condition is removed from the data set. If the two classes occur with the same frequency for the condition, the example with the class that occurs less frequently in the entire table is removed. Finally, if the two classes occur with the same frequency in the table, one is chosen randomly for elimination.

4.2. *Methodology*

To avoid overfitting, 10-fold cross-validation, a standard re-sampling accuracy estimation technique, is used [4]. For each 10-fold cross validation, a data set is randomly partitioned into 10 approximately equally

sized subsets (or folds or tests). The induction algorithm is executed 10 times; each time it is trained on the data that is outside one of the subsets and the generated classifier is tested on that subset. The estimated accuracy for each cross-validation test is a random variable that depends on the random partitioning of the data. For this reason, we repeated 10-fold cross-validation 10 times with the data set shuffled into a different order. The estimated *error rate* is the ratio of the number of misclassifications to the number of test examples expressed as a percentage. The *accuracy* is 100 minus the error rate. For comparison to the neural net result, we use a train-and-test approach for comparing results in time series. Train-and-test divides the data set into two parts. The majority part is used for training (and validating) and the rest is for testing. Because the order of data (affected by "shuffling" in the compression algorithms) can affect the compression, we repeated the train-and-test experiments 10 times and obtained an average of the error rates produced by SORCER.

5. Experiments and Results

We present results of models generated by SORCER and compare them to those obtained by C4.5, CBA and the neural net models developed at the SFWMD. We will use NN to refer to the SFWMD's neural net approach and the resulting models. Since C4.5 and CBA are more similar to SORCER than NN in terms of power of model expressiveness and comprehensibility, comparison with NN was performed separately from those with CBA and C4.5.

5.1. *Comparison with C4.5 and CBA*

SORCER applied compression algorithms S1, S2 (with distances $l = 0.5$ and 0.1), and S3 (with probability $p = 0.6$). The values of l and p are determined empirically.[6] Figure 5 shows error rates for SORCER compared to C4.5 (with pruning confidence level set to 25%) and CBA (with 10% minimum support and 80% minimum confidence with rule-pruning and rule limit of 80000).[7] The entry for each experiment is an average error over all ten tests in each ten-fold cross validation. Both C4.5 and SORCER give lower average error rate than CBA by about 4% to 7%. SORCER using S2 (with $l = 0.1$) gives the lowest average error.

Using a one-tailed t-test on the *mean pair difference* statistic [20], SORCER's mean error rate is lower than

Expt.	CBA (pruning)	C4.5 (pruning)	SORCER S1	SORCER S2, l=0.5	SORCER S2, l=0.1	SORCER S3, p=0.6
1	55.37	50.79	50.19	50.19	48.48	50.28
2	55.38	50.04	49.24	48.67	49.53	52.18
3	55.34	49.76	49.15	49.43	47.44	52.66
4	55.37	50.15	51.71	52.28	47.34	51.14
5	55.35	51.85	48.67	48.86	47.63	51.33
6	55.37	51.94	48.58	48.39	47.63	52.66
7	55.35	51.75	47.91	47.06	49.53	51.14
8	55.35	48.41	49.81	49.24	48.48	50.66
9	55.36	50.91	50.47	51.04	49.62	51.61
10	55.35	49.48	48.77	49.05	49.81	50.76
Avg.	**55.36**	**50.51**	**49.45**	**49.42**	**48.55**	**51.44**

Figure 5. Error rates from ten 10-fold cross validations.

Average over ten 10-fold cross validations	C4.5 (pruning)	SORCER (S3, p=0.6)	SORCER (S2, l=0.1)
Error rate on training sets	34.5 ± 1.298	20.2 ± 0.323	0
Error rate on testing sets	50.5± 1.156	51.4 ± 0.826	48.5± 1.002
Error rate of class "dry"	50.2± 7.529	11.6 ± 0.868	25.0± 2.203
Error rate of class "very wet"	53.3± 41.848	71.6 ± 2.764	39.2± 1.743
Rule set size	53.0	186.5	255.2
Tests covered by induced rules	67.5	63.6	28.4
Correctly classified covered test (over all covered test cases)	58.9	58.1	70.61
Correct classification by induced rules	39.7	36.9	20.0
Correct classification by heuristics or defaults	9.8	11.6	31.4

Figure 6. Comparison of C4.5 and SORCER.

that of C4.5 (with $t = -3.824$) whose mean error rate is lower than that of CBA (with $t = -13.279$) at a significance level 0.01 ($t_{0.01} = -2.821$).

Figure 6 compares in greater detail, the average results over ten 10-fold cross validations obtained by C4.5 and by SORCER using algorithms S3 (with $p = 0.6$) and S2 (with $l = 0.1$) (comparable measures from CBA's results are not available). The average size of training and testing sets are 948.6 and 105.4, respectively. For each error rate entry in Fig. 6, the numbers before and after "±" represent the average and the standard deviation of the error rate computed over 10 ten-fold cross validations, respectively. Unlike S3, since the models generated by S2 preserve consistency with the training data set, S2's predictions have a zero error rate on consistent training data sets. Results for S3 are close to those for C4.5 in terms of accuracy and generalization. The average error rates on the testing data by both systems are similar although S2's error rate is slightly lower. However, C4.5's models have better coverage on the test data than SORCER's and the majority of correct classifications is contributed by classification rules rather than default rules. On the other hand, SORCER's heuristics appear to perform quite well for cases not covered by the classification

rules. For the purpose of water management, the ability to predict extreme events is crucial for intervention. In our case, such events are lake inflows that are very wet or dry. S3 does very well on the dry events but performs poorly on the very wet events. However, S2's average errors on these events are better. It predicts the dry events reasonably well with an average error rate of 25%, but is less accurate for very wet events. Compared to S2, C4.5 has higher average error rates (with large variances) on prediction of both extreme events. The difficulty in predicting very wet inflows may be partly due to insufficient data to support training predictors for very wet events.

SORCER's predictive models, on the average, contain larger numbers of rules than C4.5's models. However, SORCER currently does not have a pruning mechanism like C4.5. In this experiment the table is compressed by only about 73.1% (for case S2). This suggests that the data may be too weak to be compressed (or to generalize) to the complexity level of the representation models of SORCER. A similar error rate obtained by C4.5 further suggests that either we have insufficient samples and the data set itself may be an unrepresentative sample, or not all of the attributes, although relevant to the problem, have strong influence on prediction accuracy.

5.2. *SORCER and NN*

Trimble et al. [7, 8] reported the results of a downscaling approach to forecasting inflows to Lake Okeechobee based on artificial neural networks. Their objective was to test if the effects of solar activity could provide useful predictors for accumulative inflows in the next six months. The data set applied was a subset of the data set described in Section 4.1, containing monthly values of seven variables: month, ENSO index, sunspot number, sunspot trend, maximum sunspot number of each cycle, Cp index and AOTC. The data set was divided into a training period from 1933 to 1987 and a testing period from 1988 to 1996. The network was trained using the back-propagation technique [6]. After extensive preprocessing and experiments on various network configurations, they obtained a two-layer architecture with a single hidden layer containing 14 units and an output layer consisting of a single unit. As described in [7], the model explained 65% and 38% of the actual variation with a standard error of estimate of 650 and 670 thousand acre-feet in the training and the testing period, respectively. To better handle the overfitting problem, they later applied an "early stopping" training approach which resulted in predictions that explained 48 percent of the actual variation in both training and testing periods [8].[8] While NN performed reasonably well, neural net models are hard to understand and require much time and expertise to train.

To compare how table compression and neural networks perform in this application, we experiment on the same data set as applied in [7, 8] (i.e., a subset of the overall data set containing 760 rows of monthly values of seven attributes from 1933 to 1996). In this data set, discretization resulted in an inconsistent data set which was originally consistent. SORCER eliminated inconsistent data before compression. The experiment was conducted in two parts:

(1) estimating error rates obtained from SORCER's models in this new data set, and
(2) comparing performance between SORCER and NN on a specific test data as in [7, 8].

For the first part, SORCER applied compression algorithms S1, S2 (with $l = 0.5$ and 0.1), and S3 (with $p = 0.6$). Figure 7(a) shows the average and standard deviation of error rates over ten trials of ten-fold cross validation on the entire data set obtained from each of SORCER's algorithms. The entry in each trial of 10-fold cross validation is an average error over ten tests in the 10-fold cross validation. The average error rates do not vary much by algorithm with values between 43.7 and 44.7. Figure 7(b) compares the results in more detail. On the average, the models generated by S3 are the smallest, provide most coverage on the testing data and have the highest percentage of correct prediction based on the induced rules. This analysis on the entire data set gives us an idea of estimated errors to be expected and indicates that the S3 algorithm may be favorable.

In the second part of our experiment, we use the same training and testing data sets as those used in [7, 8]. First, we select a compression algorithm based on experimental results using only the training data set. Figure 8(a) shows the average and standard deviation of error rates over ten trials of ten-fold cross validation obtained by different algorithms using the training data set. S3 gives a slightly better average error rate with a smaller variance than the rest. The details of the average number of rules in the models, and percentages of coverage and correct predictions by the rules in the compressed tables, are similar to those shown in

Trial	S1	S2, l=0.5	S2, l=0.1	S3, p=0.6
1	42.54	45.37	44.43	44.11
2	45.37	42.54	42.54	44.58
3	43.8	43.49	42.07	43.64
4	44.9	43.49	42.86	44.27
5	41.6	45.21	42.54	46.62
6	43.49	47.41	44.58	45.37
7	44.9	44.74	47.25	45.84
8	45.37	44.58	42.54	44.58
9	43.17	45.37	43.49	43.64
10	42.07	44.74	44.58	44.27
Avg.	**43.72**	**44.69**	**43.69**	**44.69**
Stdev.	**±1.382**	**±1.338**	**±1.568**	**±0.968**

(a)

Average over ten 10-fold cross validations	S1	S2 l= 0.5	S2 l= 0.1	S3 p=0.6
Rule set size	152.1	138.6	137.8	76.44
Tests covered by induced rules	63.44	48.16	46.45	78.34
Correctly classified covered test (over all covered tests)	63.87	71.68	72.49	60.58
Correct classification by induced rules	40.52	34.52	33.67	47.46
Correct classification by heuristics or defaults	15.76	20.78	22.64	7.85

(b)

Figure 7. Error rates from ten ten-fold cross validations on the entire data set using various compression algorithms.

Trial	S1	S2 l=0.5	S2 l=0.1	S3 p=0.6
1	46.32	47.85	46.17	47.55
2	46.78	47.85	49.54	46.63
3	47.85	45.4	47.24	48.62
4	48.16	48.16	47.24	47.7
5	48.93	46.93	47.7	47.55
6	46.78	47.55	49.54	46.32
7	48.31	45.86	47.24	46.78
8	46.17	47.24	47.55	47.85
9	48.01	49.23	47.39	46.63
10	46.17	46.63	48.01	46.93
Avg.	**47.35**	**47.27**	**47.76**	**47.26**
Stdev.	**±1.014**	**±1.125**	**±1.050**	**±0.713**

(a)

Test	Train	Overall
44.44	27.76	30.13
44.44	**27.76**	**30.13**
45.37	27.76	30.26
50.00	27.76	30.92
46.3	27.76	30.39
48.15	27.76	30.66
42.59	27.76	29.87
41.67	27.76	29.74
44.44	27.76	30.13
40.74	27.76	29.61
44.8	**27.8**	**30.2**
±2.836	**±0**	**±0.401**

(b)

Ten trials of train-and-test	Average
Rule set size	80.5
Correctly classified covered test	57.40
Correct classification by induced rules	49.91
Correct classification by heuristics	5.28

(c)

NN	Test	Train	Overall
	46.3	**34.8**	**36.5**

(d)

Figure 8. Error rates from SORCER and NN: (a) ten ten-fold cross validations on the training data set, (b) ten train-and-test trials using S3 ($p = 0.6$), and (c) detailed averages of results in (b), (d) NN's results.

Fig. 7(b). Thus, we selected S3 for model generation from the training data set.

We repeated the experiments for 10 trials. In each trial, SORCER applied algorithm S3 to compress the training data set which had been shuffled into a different random order. SORCER then applied the resulting model (i.e., the compressed table) to the testing data. The error rate (on testing, training and overall periods) in each trial are given in Fig. 8(b). Figure 8(c) shows that on the average the models produced by SORCER contains about 80 rules and the majority of correct predictions was due to predictive rules in the models. Based on the NN's predictions given in numeric values in [7], we used the same discretization of the cumulative inflows as applied in SORCER, Fig. 8(d) shows estimated error rates produced by NN. SORCER's error rates are on the average lower than NN's error rates on testing, training and overall data, although the result on testing data is only slightly lower.

In practice, it is often necessary to decide which of the resulting models (from a number of train-and-test trials) is to be used as a predictor for future events. For this purpose, we may adopt a selection criterion prior to testing the model with the testing data. Suppose our criterion is to select the first model found with the smallest number of rules. Applying this criterion to the ten trials of train-and-test, the model produced in the second trial, shown in bold in Fig. 8(b), is selected (with 79 rules) to be a predictor for the testing period. The error rates from SORCER are lower than NN. Figure 9 compares SORCER's predictions with the actual inflows and NN's predictions on the testing period.

The results show that NN successfully predicted the majority of drier periods of 1988, 1989 and the very wet period of 1994 and 1995 and the return to normal inflows in 1996. However, NN did not adequately predict the 1991 wet period. Like NN, SORCER generally predicted the dry periods very well but under-predicted the very wet period of 1991 and over-predicted some dry periods of 1990. We shall explain the reason why this anomaly occurs in the conclusion section. Note that the training time for SORCER was significantly shorter (seconds vs. hours).

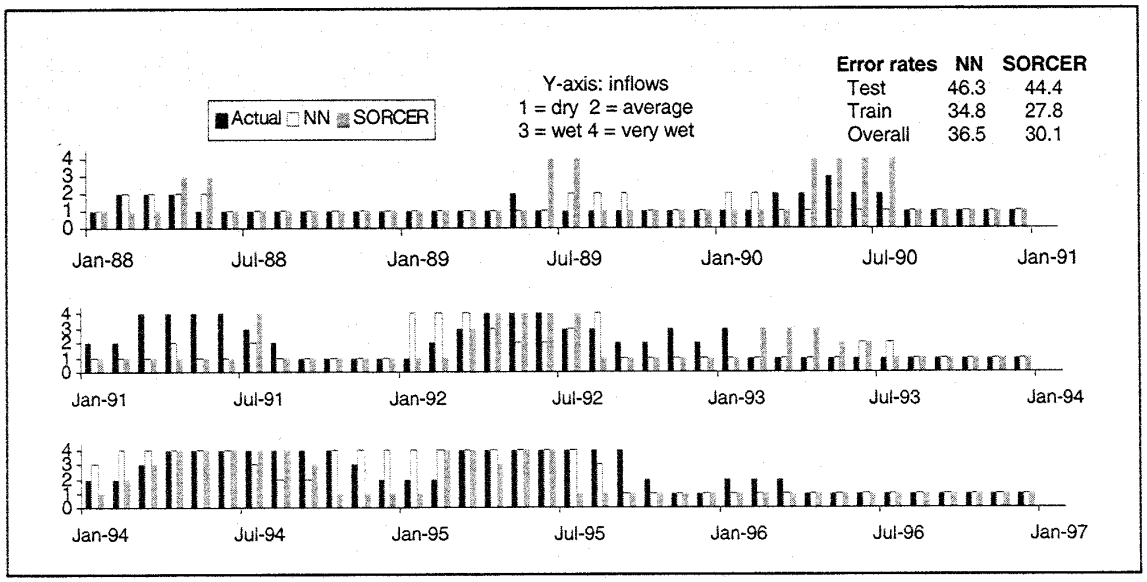


Figure 9. The actual lake inflows compared to the predicted inflows by SORCER and Neural net in the testing period.

6. Practical Analysis

To make resulting models more informative in guiding a decision making process, predictions can be represented in the form of a *confusion matrix*. For example, Fig. 10(a) shows two confusion matrices resulting from NN's and SORCER's predictions as shown in Fig. 9. Each row indicates predictions for a given actual class. For example, for the total of 54 actual inflows of dry events, 46 were predicted correctly by SORCER whereas two, four, and two of them were predicted as average, wet and very wet events, respectively. Similarly, Fig. 10(b) shows two confusion matrices obtained by applying SORCER's and C4.5's models (trained by a data set in Section 5.1 in the period between 1912 and 1989) for prediction in a testing period between 1990 and 1999.

In many domain applications, predictions of particular class values are more important than overall performance. As mentioned earlier, for the purpose of water management, predictions of dry and very wet events are crucial for preparation for intervention. In Fig. 10(a), error rates for dry and very wet events are 14.8% and 47.6% for SORCER and 16.7% and 57.1% for NN, respectively. Thus, both the overall and extreme event error rates from SORCER's models are lower than NN's. On the other hand, Fig. 10(b), SORCER's overall error rate is 59.2%, which is slightly higher than C4.5's 58.3%. However, C4.5's error rates for dry and very wet events, 57.8% and 61.3%,

Predictions

Actual \ SORCER	Dry	Avg	Wet	Vwet
Dry	46	2	4	2
Avg	18	1	1	4
Wet	4	0	2	3
Vwet	8	0	2	11

Predictions

Actual \ NN	Dry	Avg	Wet	Vwet
Dry	45	8	0	1
Avg	15	3	1	5
Wet	3	1	1	4
Vwet	4	5	3	9

(a)

Prediction

Actual \ SORCER	Dry	Avg	Wet	Vwet
Dry	23	8	7	7
Avg	8	8	4	11
Wet	2	4	5	2
Vwet	4	9	5	13

Predictions

Actual \ C4.5	Dry	Avg	Wet	Vwet
Dry	19	15	4	7
Avg	10	18	1	2
Wet	4	6	1	2
Vwet	8	9	2	12

(b)

Figure 10. Confusion matrices from train-and-test experiments.

Actuals \ Predictions	dry	avg	wet	vwet
dry	5	−1	−2	−4
avg	0	2	0	−2
wet	−2	0	4	2
vwet	−4	−2	2	5

Figure 11. An example of a utility function.

are higher than the 48.9% and 58.1% produced by SORCER.

The analysis can be fine tuned further by including considerations of how far the predicted ordinal ranges are from the actual ones instead of measuring performance based on binary outcomes. In order to assess the measurement of the risk in making decisions based on the resulting models, a *utility function* (or cost or risk functions) can be employed [21]. This function is domain specific. Figure 11 shows an example of a possible utility function obtained from our domain expert. The utility matrix indicates that correct predictions of dry and very wet events earn the highest utility score of 5, as do correct predictions of wet events. Predictions of average as dry are better than predictions of dry as average. The worst case is when the model predicts an extreme event when it is actually an extreme event of an opposite type. In general, the matrix reflects the importance of predictions of extreme events and takes account of how far the predicted ordinal ranges are from the actual ones as they can have significant impact on how appropriate interventions have to be prepared in advance. For example, predictions of a very wet event when it is actually dry could lead to an irreversible action, such as water release to the sea, that could cause of lot of damage if drought occurs.

Combining results from Fig. 10(b) and the utility function in Fig. 11, we can compute total utility score $= \sum_{i,j} u_{ij}c_{ij}$, where u_{ij} and c_{ij} are the (i, j)th element of the utility matrix and the confusion matrix, respectively. The total utility scores from SORCER, C4.5, and the baseline performance (by predicting the most likely event or a majority class, which is "dry") are 120, 90 and 75, respectively. Thus, for the given utility function, SORCER's score is over 30% better than C4.5's and 60% better than the baseline's. On the other hand, for the results shown in Fig. 10(a), the utility scores from SORCER, NN, and the baseline performance are 239, 240 and 168, respectively. Thus, SORCER's score is 0.4% lower than NN's, but both are about 40% better than the baseline's score. (Note that the determination of an appropriate utility matrix can be quite involved and subtle.)

7. Related Work

Much work has been done on traditional downscaling methods including statistical and dynamic downscaling techniques [22–24]. Dynamic downscaling has been applied to study climate impacts on water resources using regional climate and hydrology models over the Pacific Northwest and California [25]. Recently, neural networks have been applied to natural system modeling and prediction of vegetation distribution under global climate changes [26, 27]. For purposes of water management, SFWMD has made initial downscaling efforts and applied neural networks to forecasting of inflows to Lake Okeechobee [7, 8]. Since the emphasis of this work is prediction of extreme high and low periods of inflows, predictions of qualified inflows are appealing. Our previous work [28] investigated the applicability of table compression, which required discrete data values. In this paper, we have applied different compression algorithms and extended the data set to include a much larger number of climate indices and compared table compression with state-of-the-art data mining systems.

Decision tree learning, as in C4.5 [1] and ID3 [29], provides a simple but powerful technique for concept learning. It has been applied successfully in various data mining applications. A decision tree model can be viewed as a decision table where each row (or rule) corresponds to a path from the root to the leaf of the tree. However, the difference between the rules generated by decision tree learning and table compression induction is that the former are mutually exclusive, whereas the latter need not be.

Association rule mining [2, 5, 30] is one of the most influential data mining techniques. Each rule has corresponding parameters to signify "support" and "confidence" of the rule based on frequencies of occurrence of the events on the left and right sides of the rule. CBA [2] generates predictive models based on association rules. Unlike SORCER, CBA aims to obtain a complete set of predictive rules.

Other approaches for rule mining include inductive logic programming systems such as the one described in [31] and GOLEM [32]. These systems generate rules in disjunctive normal form like SORCER but they apply inference rules to derive new rules whereas SORCER uses theoretical set-based operations and heuristics to

generalize the rules. The R-MINI system [33] is perhaps most similar to our approach in that both aim to produce a "minimal" or near minimal set of rules that approximates the target function. R-MINI employs a technique based on MINI [34], a heuristic technique for minimizing large Boolean functions, and has shown promising results for data mining applications [35]. However, SORCER is based on a more general representation and uses different operations which are more general and easier to understand. For example, R-MINI ignores rules with missing information (which makes sense in logic synthesis) but SORCER does not since in knowledge discovery, incomplete information can still be useful in identifying patterns.

8. Conclusions

We have described an application of SORCER, a recent rule induction system, the state-of-the-art data mining systems C4.5, CBA and NN to produce models for forecasting lake inflows from a database of climate variability measures. The average accuracies of the inflows (discretized to four values) predicted by the resulting models range, approximately, from 45% to 51%. The similarity in the error rates obtained by these techniques suggests that we have insufficient or unrepresentative samples and that the data set itself is too weak to generalize with representation models of such high complexity. The levels of accuracy obtained may seem low in comparison to other application domains. However, in our expert's view, the results are encouraging considering a 30%–40% improvement of the utility score over the baseline performance and the fact that no regional hydrologic input is included in the predictor.

The average error rate for SORCER is lower than C4.5's which is lower than that of CBA. The differences are statistically significant. Experimenting with SORCER and NN on the same specific training and testing periods, SORCER produces lower error rates than NN in both training and testing data sets. This is surprising since NN has higher representational power, which can represent more complex models. Also, NN is known to tolerate noisy data very well. We believe SORCER tends to perform well on a data set whose underlying regularity involves complex relationships among many problem variables. Rules in second-order tables are more expressive than the simple disjunctive normal forms used in most machine learning systems [12]. Thus, a second-order table provides a rich but comprehensible model to represent relatively complex logical rules compactly. Another advantage of table compression induction is that it retains cases with missing data. Furthermore, in comparison to NN, SORCER requires much less training time (seconds versus days).

Our experiments also include considerations on how far the predicted ordinal ranges are from the actual ones. For the purpose of water management, we focus our attention on prediction of extreme events, i.e., dry and very wet accumulated inflows. We show how the results can be further analyzed to provide users with possible guidelines for decisions by exploiting confusion and utility matrices. In general, second-order table compression performs very well on dry events but tends to perform poorly on very wet events. The training data sets contain many more of instances of dry events than very wet events. We believe this affects the training and produces models that are more accurate in predicting dry events than very wet events. Another possible explanation, as noted by our domain expert, for poor performance of both SORCER and NN in the testing period in the last decade, as shown in Fig. 9 is due to the eruption of Mount Pinatubo in 1990. There is evidence that this eruption greatly affected the global climate for the next few years.

Currently, SORCER uses only simple domain independent heuristics during compression. We plan to continue improving classification accuracy by fine tuning heuristics, incorporating prior domain knowledge, and including data with a richer set of indicators and a longer history. We also believe that coarser ranges in discretization will facilitate better compression which may lead to better accuracy. For example, instead of dividing a year into twelve months, we may divide it into seasons which may better associate to seasonal climate indices such as El Nino.

Acknowledgment

The author would like to thank the referee whose thoughtful suggestions improved the paper. Special thanks to Paul Trimble and his team of SFWMD for their domain expertise and support in this research. John Leuchner implemented SORCER and provided many valuable comments on a previous version of the paper. Marco Carvalho assisted with part of the experiments. Part of this work [36] was presented at the IEEE International Conference on Systems, Man and Cybernetics 2001 (SMC, 2001).

Notes

1. The density of a relation is the size of its minimal flat extension (i.e., the smallest flat relation which covers its flat extension) divided by its length.
2. A coupled ocean-atmosphere phenomenon of large-scale fluctuation of rainfall, ocean temperatures, atmospheric circulation, and air pressure, across the tropical Pacific. It is believed to be the largest climate variability effect observed.
3. http://www.aoml.noaa.gov/phod/
4. http://www.ncdc.noaa.gov
5. http://www.atmos.washington.edu/~mantua/abst.PDO.html
6. SORCER's commands can be used to provide statistics on the training data sets (e.g., the class distribution of each attribute value).
7. Results from other minimum supports (25% and 40%) and confidence supports (60% and 90%) are the same.
8. Also obtained by a communication with the first author of [7].

References

1. J. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann: San Mateo, CA, 1993.
2. B. Liu, W. Hsu, and Y. Ma, "Integrating classification and association rule mining," in *Proceedings of Knowledge Discovery and Data Mining*, New York, USA, 1998, pp. 80–86. Also in http://www.comp.nus.edu.sg/~dm2/result.html.
3. P. Langley and H. Simon, "Applications of machine learning and rule induction," *CACM*, vol. 38, no. 11, pp. 55–64, 1995.
4. T. Mitchell, *Machine Learning*, McGraw-Hill Companies: New York, NY, 1997.
5. R. Agrawal and R. Srikant, "Fast algorithms for mining association rules," in *Proceedings of the 20th International Conference on Very Large Databases*, Santiago, Chile, 1994, pp. 487–499.
6. D. Rumelhart, G.E. Hinton, and R.J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, pp. 533–536, 1986.
7. P. Trimble, S. Everett, and C. Neidrauer, "A refined approach to Lake Okeechobee water management: An application of climate forecasts," Special report, South Florida Water Management District, Florida, 1998.
8. P. Trimble, E. Santee, and C. Neidrauer, "Including the effects of solar activity for more efficient water management: An application of neural networks," in *Proceedings of the Workshop on AI Application in Solar-Terrestrial Physics*, Sweden, 1997.
9. P. Domingos, "The role of Occam's Razor in knowledge discovery," *Data Mining and Knowledge Discovery*, vol. 3, pp. 409–425, 1999.
10. J. Leuchner and R. Hewett, "A formal framework for large decision tables," in *Proceedings of the International Knowledge Retrieval, Use, and Storage for Efficiency Symposium (KRUSE)*, Santa Cruz, USA, 1997, pp. 65–179.
11. R. Hewett and J. Leuchner, "Second-order relations and decision tables," TR 97-27, CSE, Florida Atlantic University, Boca Raton, FL, 1997.
12. R. Hewett and J. Leuchner, "The power of second-order decision tables," in *Proceedings of SIAM International Conference on Data Mining (SDM'02)*, Arlington, USA, 2002, pp. 384–399.
13. H.C. Willet, "Climate responses to variable solar activity-past present and predicted," in *Climate History, Periodicity, Predictability*, edited by R. Michael and Rampino, Van Nostrand Reinhold Company, Inc.: MIT, Cambridge, 1987.
14. United States Army Corps of Engineers, Rules Curves and Key Operating Regulation Manual, Append. D, 1978.
15. K. Kira and L.A. Rendell, "A practical approach to feature selection," in *Proceedings of the Ninth International Conference on Machine Learning*, Aberdeen, Scotland, 1992, pp. 249–256.
16. R. Kohavi and G.H. John, "Wrappers approach," in *Feature Selection for Knowledge Discovery and Data Mining*, edited by H. Liu and H. Motoda, Kluwer Academic Publishers, pp. 33–50, 1998.
17. I. Kononenko, "Estimating attributes: Analysis and extensions of RELIEF," in *Proceedings of the 7th European Conf. on Machine Learning*, Catania, Italy, 1994, pp. 171–182.
18. J. Dougherty, R. Kohavi, and M. Sahami, "Supervised and unsupervised discretization of continuous features," in *Proceedings of the 12th International Conference on Machine Learning*, San Francisco, CA, 1995, pp. 194–202.
19. U. Fayyad and K.B. Irani, "Multi-interval discretization of continuous-valued attributes for classification learning," in *Proceedings of the 13th International Joint Conference on Artificial Intelligence*, Morgan Kaufmann: San Francisco, CA, 1993, pp. 1022–1027.
20. L. Lapin, *Statistics for Modern Business Decisions*, Harcourt Brace Jovanovich, Inc., 1973.
21. S. Weiss and C. Kulikowski, *Computer Systems That Learn*, Morgan Kaufmann: San Francisco, CA, 1991.
22. D.R. Easterling, "Development of regional climate scenarios using a downscaling approach," *Climatic Change*, vol. 41, pp. 615–634, 1999.
23. H. von Storch, E. Zorita, and U. Cubash, "Downscaling of global climate change estimates to regional scales: An application to Iberian rainfall in wintertime," *Journal of Climate*, vol. 6, pp. 1161–1171, 1993.
24. E. Zorita and H. von Storch, "A survey of statistical downscaling techniques," Institute of Hydrophysics, GKSS Forschungszentrum Geesthacht, Germany, 1997.
25. L. Leung, M. Wigmosta, S. Ghan, D. Epstein, and L. Vail, "Application of subgrid orographic precipitation/surface hydrology scheme to a mountain watershed," *Journal of Geophysics Research*, vol. 101, pp. 12803–12817, 1996.
26. E. Chown and T. Dietterich, "A comparison of neural network and process-based models for vegetation distribution under global climate change," Technical Report, Department of Computer Science, Oregon State University, Corvallis, OR, 1997.
27. K. Hsu, H. Gupta, and S. Sorooshian, "Artificial neural network modeling of the rainfall runoff process," *Water Resources Research*, vol. 31, pp. 2517–2530, 1995.
28. R. Hewett, J. Leuchner, and P. Trimble, "Discovering hydrologic forecasting rules for water management using table compression: A preliminary result," in *Proceedings of the 5th International Conference on Computer Science and Informatics (CS&I 2000)*, Atlantic City, USA, 2000, pp. 476–479.
29. J. Quinlan, "Induction of decision trees," *Machine Learning*, vol. 1, no. 1, pp. 81–106, 1986.

30. U. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy (eds.), *Advances in Knowledge Discovery and Data Mining*, AAAI Press/The MIT Press: Menlo Park, CA, 1996.
31. S. Džeroski, "Inductive logic programming and knowledge discovery and databases," in *Advanced in Knowledge Discovery and Data Mining*, edited by U. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, AAAI Press/The MIT Press: Menlo Park, CA, pp. 117–152, 1996.
32. S. Muggleton and C. Feng, "Efficient induction of logic programs," in *Proceedings of the First Conference on Algorithmic Learning Theory*, Tokyo, Japan, 1990, pp. 368–381.
33. S. Hong, "R-MINI: A heuristic algorithm for generating minimal rules from examples," in *Proceedings of the Third Pacific Rim International Conference on Artificial Intelligence*, PRICAI'94, Beijing, China, 1994, pp. 331–337.
34. S. Hong, R. Cain, and D. Ostapko, "MINI: A heuristic approach for logic minimization," *IBM Journal of Research and Development*, pp. 443–458, 1974.
35. C. Apte and S. Hong, "Predicting equity returns from securities data," in *Advanced in Knowledge Discovery and Data Mining*, edited by U. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, AAAI Press/The MIT Press: Menlo Park, CA, pp. 541–560, 1996.
36. R. Hewett, J. Leuchner, and M. Carvalho, "Generating predictive models of regional water flows from global climate history with machine learning," in *Proceedings of IEEE International Conference on Systems, Man, and Cybernetics*, Tucson, USA, 2001, pp. 292–297.

Rattikorn Hewett is an associate professor at the School of Electrical Engineering and Computer Science, Washington State University, Vancouver. She has been a research scientist at the Institute for Human and Machine Cognition, University of West Florida and a postdoctoral fellow at Knowledge Systems Laboratory, Stanford University, and holds a Ph.D. in Computer Science from Iowa State University, M. Eng. Sc. in Computer Science from the University of New South Wales, and B.A. in Pure Mathematics with honors in Statistics from Flinders University. Her research interests include data mining, machine learning, intelligent control agents, model-based reasoning and blackboard systems.