

Autoregressive Tree Models for Time-Series Analysis

*C. Meek, D.M. Chickering, and D. Heckerman**

1 Introduction

The analysis and modeling of time-series data is an important area of research for many communities. In this paper, our goal is to identify models for continuous-valued time-series data that are useful for data mining in that they (1) can be learned efficiently from data, (2) support accurate predictions, and (3) are easy to interpret. To these ends, we describe an interpretable class of models that we call AutoRegressive Tree models, or ART models, that are a generalization of standard autoregressive (AR) models. We describe learning methods for ART models and compare these methods to those for alternative models. Our experiments, performed on 2,494 time-series data sets from the International Institute of Forecasters, demonstrate that ART models provide superior predictive accuracy. We concentrate on the problem of modeling the evolution of values of a continuous variable over time; that is, we model a univariate time series. The generalization to multivariate time-series analysis is straightforward and is discussed in Section 6.

Roughly speaking, we construct ART models as follows. First, we use a standard “windowing” transformation of a time-series data set into a set of cases suitable for a regression analysis, where the “predictor variables” and “target variable” in the analysis correspond to the preceding values and current value, respectively, in the time series. This transformation is often used when constructing autoregressive models. Then, we use the transformed data set to learn a decision tree for the target variable. This decision tree has linear regressions at its leaves, thus, producing a piecewise-linear auto-regression model. We use a Bayesian technique to learn the structure and parameters of the decision tree.

Although there has been much research on learning decision trees with linear regressions at the leaves (see Section 2), we know of no other work in which

* {meek|dmax|heckerma}@microsoft.com

these models have been applied to time-series analysis. In addition, our work is distinguished from previous work in that we (1) use a Bayesian approach for model selection, and (2) demonstrate that our approach (which incorporates the ART model) produces predictions that are more accurate than other approaches suitable for data mining.

2 Autoregressive Trees

In this section, we describe a class of models called autoregressive tree (ART) models as well as several other types of models, some closely related to ART models.

We begin by introducing notation and nomenclature. We denote a temporal sequence of variables by $Y = (Y_1, Y_2, \dots, Y_T)$. Time-series data is a sequence of values for these variables denoted by $y = (y_1, y_2, \dots, y_T)$. In this paper, we restrict consideration to time series models that are probabilistic, stationary, and p -order Markov ($p \geq 0$). That is, we concentrate on models of the form

$$p(y_t|y_1, \dots, y_{t-1}, \theta) = f(y_t|y_{t-p}, \dots, y_{t-1}, \theta), \quad p < t \leq T \quad (1)$$

where $f(\cdot|\cdot, \theta)$ is a family of conditional probability distributions that represents the functional form of the model and θ are the model parameters. Stationarity means that the dependence of y_t on the preceding variables does not change with time. The p -order Markov assumption means that, given the previous p observations, y_t is independent of the remaining previous observations. Note that the function $f(y_t|y_{t-p}, \dots, y_{t-1}, \theta)$ is often called a *regression* where Y_t is the *target variable* and $(Y_{t-p}, \dots, Y_{t-1})$ are the *regressor variables*.

2.1 ART Models

The most common type of model used for time-series analysis is the linear autoregressive (AR) model (e.g., [9]). A *linear autoregressive model of length p , denoted $AR(p)$* , is described by Equation 1 in which $f(y_t|y_{t-p}, \dots, y_{t-1}, \theta)$ is a linear regression:

$$f(y_t|y_{t-p}, \dots, y_{t-1}, \theta) = N(m + \sum_{j=1}^p b_j y_{t-j}, \sigma^2) \quad (2)$$

where $N(\mu, \sigma^2)$ is a normal distribution with mean μ and variance σ^2 , and $\theta = (m, b_1, \dots, b_p, \sigma^2)$ are the model parameters (e.g., [6] page 55).

An *autoregressive tree (ART) model* is a piecewise linear autoregressive model in which the boundaries are defined by a decision tree, and the leaves of the decision tree contain linear autoregressive models. Figure 1 is an example of an ART model in which there are three regions defined using the variable Y_{t-1} and each leaf contains an AR(1) model described by the equation at the leaf.

In this paper, we further specialize to a subset of ART models that we call *autoregressive tree models of length p , denoted $ART(p)$* . An $ART(p)$ model is an ART model in which each leaf of the decision tree contains an AR(p) model, and the split variables for the decision tree are chosen from among the previous p variables

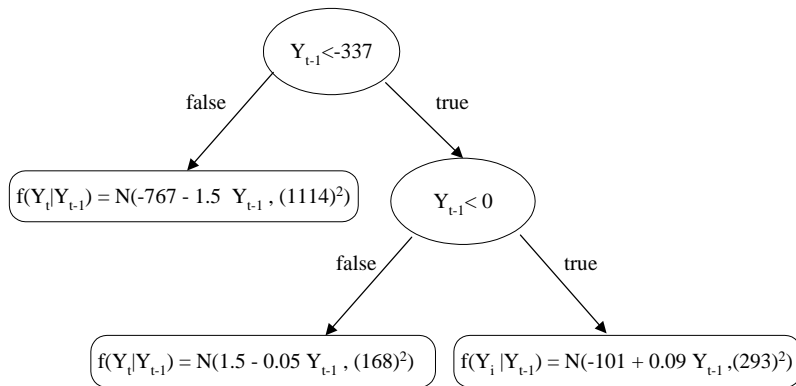


Figure 1. An AR tree.

in the time series. We concentrate on these models so that we can compare more directly with AR models—that is, more easily assess the benefits of allowing for a piecewise linear rather than linear model.

In $\text{ART}(p)$ models, each non-leaf node in a decision tree has associated with it a Boolean formula that is a function of the p variables Y_{t-p}, \dots, Y_{t-1} . For example, the root node of the ART model in Figure 1 tests whether $Y_{t-1} < -337$. We associate with each *edge* in the tree the formula (the negation of the formula) for its parent node if the label on the edge is “true” (“false”). We associate with each leaf l_i an indicator function, ϕ_i , that returns 1 when the conjunction of all the formulas associated with the edges along the path from the root node to the leaf l_i are true, and 0 otherwise. For example, the indicator function associated with the middle leaf in Figure 1 returns 1 when $(Y_{t-1} < -337) \wedge (Y_{t-1} \geq 0)$, and 0 otherwise. Then, an $\text{ART}(p)$ model is defined by Equation 1 where

$$f(y_t | y_{t-p}, \dots, y_{t-1}, \theta) = \prod_{i=1}^L f_i(y_t | y_{t-p}, \dots, y_{t-1}, \theta_i)^{\phi_i} = \prod_{i=1}^L N(m_i + \sum_{j=1}^p b_{ij} y_{t-j}, \sigma_i^2)^{\phi_i}, \quad (3)$$

where L is the number of leaves, $\theta = (\theta_1, \dots, \theta_L)$, and $\theta_i = (m_i, b_{i1}, \dots, b_{ip}, \sigma_i^2)$, are the model parameters for the linear regression at leaf l_i , $i = 1, \dots, L$.

$\text{ART}(p)$ (and ART) models are generalizations of AR models because an $\text{ART}(p)$ model with a decision tree having only a single leaf is an $\text{AR}(p)$ model. $\text{ART}(p)$ models are more powerful than AR models because they can model non-linear relationships in time-series data. Furthermore, $\text{ART}(p)$ models can represent *periodic* time-series data. This fact follows from the analysis by Tong [16] of a family of piece-wise linear models that are a special case of ART models. To illustrate the potential benefits of an ART model as compared to an AR model, consider the time series data (one of the 2,494 data sets described in Section 4) shown in Figure 2. Here, the data is shown as a scatter plot of y_t versus y_{t-1} . Also shown on this plot are $\text{AR}(1)$ and $\text{ART}(1)$ models learned from this data. We can see that the ART model provides a piecewise linear approximation that fits the data more closely than

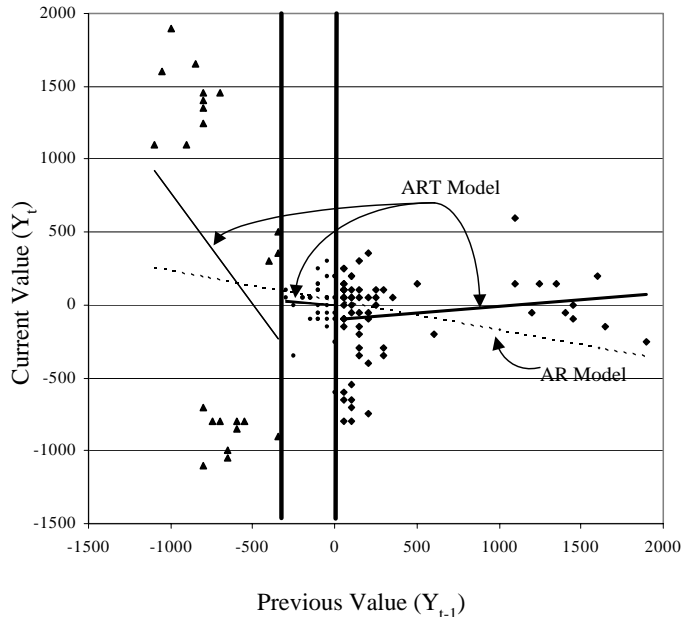


Figure 2. Scatter plot of time-series data and $AR(1)$ and $ART(1)$ models.

does the AR model. ART models that include additional predictor variables can provide further benefits over AR models, although it is difficult to illustrate these benefits with a simple (two-dimensional) figure.

In Section 3, we shall discuss computationally efficient methods for learning $ART(p)$ models from data. In Section 5, we show that the resulting models yield accurate forecasts. Finally, because ART models are simple piece-wise linear predictors, they are easy to interpret. These three properties make ART models particularly suitable for data mining.

2.2 Other Non-linear Autoregressive Models

There are a wide variety of alternative non-linear modeling techniques that have been applied to time-series analysis (see, e.g., [8]). In fact, as we describe below, one can transform the time-series problem into a regression problem and apply any non-linear or non-parametric regression technique.

The most closely related families of models to ART models are the self-exciting threshold autoregressive models (SETAR) of Tong [16] and the adaptive smooth threshold regressive models (ASTAR) of Lewis, Ray, and Stevens [14]. Both the SETAR and ASTAR models, as well as ART models, are piece-wise linear models. Described in terms of a decision tree representation, the SETAR models are limited to a single split variable. Because of this limitation we do not further consider SETAR models in this paper. The ASTAR models are obtained by the application

of the well-known multiple adaptive regression splines (MARS) system [7] to time-series data. ART models are primarily distinguished from the ASTAR models in two ways (1) the error estimates for an ART model can differ between each of the pieces of the piece-wise linear model, and (2) the ART models allow there to be discontinuities between the piece-wise linear models.

Another common technique for non-linear time-series analysis is the use of neural networks. Although these models can lead to reasonable predictive performance, they are difficult to interpret and can be expensive to learn. Thus, these models are less useful for data mining and we do not consider them further in this paper.

3 Learning and Forecasting with ART Models

In this section, we show how to learn the structure and parameters of ART models from observed time-series data and how to use an ART model for forecasting. In Section 3.1, we describe a general Bayesian approach for learning models from data. In Section 3.2, we describe how to calculate a Bayesian score for choosing an ART model structure. In Section 3.3, we describe search techniques that can be used in conjunction with our Bayesian score to identify good models from data. Finally, in Section 3.4, we describe how to forecast with an ART model.

3.1 Learning Framework

Let us consider a general Bayesian approach to learning. In this approach, we have a collection of alternative model structures s_1, \dots, s_S having unknown model parameters $\theta_{s_1}, \dots, \theta_{s_S}$, respectively, and we express our uncertainty about the structure and parameters by placing probability distributions on them: $p(s)$ and $p(\theta_s|s)$ —we omit the structure subscript s from θ when clear from context. We then use Bayes’ rule in conjunction with the data d to infer posterior distributions over these quantities: $p(s|d)$ and $p(\theta|d, s)$. In the most general case, we then make predictions by averaging over these distributions. This approach, however, is typically computationally intractable. Consequently, we use a Bayesian-model selection approach in which we choose the structure s that has the highest posterior probability $p(s|d)$, and make predictions according to $p(\theta|d, s)$ for that s .

The key quantity in this Bayesian approach is the posterior probability of model structure $p(s|d)$. By Bayes’ rule, this posterior probability is given by $p(s|d) = p(s)p(d|s)/p(d)$. Because $p(d)$ is a constant we can use the product $p(s)p(d|s)$ to choose the best model. We shall refer to this product as the *Bayesian score* for the model. The first term in this score is simply the structure prior. The second term $p(d|s)$ is called the *marginal likelihood* and is equal to $\int p(d|\theta, s) p(\theta|s) d\theta$, the likelihood of the data $p(d|\theta, s)$ averaged over the uncertainty in θ .

It is interesting to note that, when used for model selection, the marginal likelihood balances the fit of the model structure to data with the complexity of the model. One way to understand this fact is to note that, when the number of cases

N is large, the marginal likelihood can be approximated by

$$p(d|\hat{\theta}_s, s) - \frac{|\theta|}{2} \log N$$

where $\hat{\theta}_s$ is the maximum-likelihood estimator of the data (see, e.g., [10]) for model s . The first quantity in this expression represents the degree to which the model fits the data, which increases as the model complexity increases. The second quantity, in contrast, penalizes model complexity.

Now, let us turn to the application of the Bayesian approach to learning a stationary, p -order Markov time-series model. According to Equation 1, the likelihood of the data is

$$p(y_{p+1}, \dots, y_T | y_1, \dots, y_p, \theta, s) = \prod_{t=p+1}^T f(y_t | y_{t-p}, \dots, y_{t-1}, \theta, s) \quad (4)$$

In this equation, we have included s to emphasize that we are learning model parameters *and* model structure. Note that, in writing this likelihood, we have omitted the first p observations, because a p -order Markov model can not predict these observations.

Given this likelihood, we can proceed with learning as described in the previous paragraphs, placing priors on model structures and model parameters and using Bayes' rule. In the following section, we discuss this approach in detail for the ART model. In the remainder of this section, we describe a standard "windowing" approach in the literature that reduces our problem to the problem of learning an ordinary linear regression models (e.g., [8]).

At the heart of the windowing approach is a transformation of the single sequence $y = (y_1, \dots, y_T)$ to a set of cases x^1, \dots, x^{T-p} . The transformation is given by $x^i = (x_1^i, \dots, x_{p+1}^i)$, for $1 < i < T - p$, where $x_j^i = y_{i+j-1}$. We call this transformed data set the *length p transformation* of the time-series data set. As an example, consider the sequence $y = (1, 3, 2, 4)$. Then, the length-2 transformation is $x^1 = (1, 3), x^2 = (3, 2), x^3 = (2, 4)$, and the length-3 transformation is $x^1 = (1, 2, 3), x^2 = (2, 3, 4)$.

Given this transformation, we can rewrite the likelihood of the model in Equation 4 as follows:

$$p(y_{p+1}, \dots, y_T | y_1, \dots, y_p, \theta, s) = \prod_{t=p+1}^T f(x_{p+1}^t | x_1^t, \dots, x_p^t, \theta, s) \quad (5)$$

This likelihood is precisely the likelihood for an ordinary regression model with target variable X_{p+1} and regressor variables X_1, \dots, X_p . Thus, we can learn stationary, p -order Markov model time-series using any ordinary regression technique, including decision trees.

Finally, we note that this general approach of Bayesian model selection using $p(s|d)$ to choose the model structure (including the order of the Markov process p) has one complication. Namely, when we are selecting among p -order Markov

models having different values of p , the number of product terms in the likelihood of Equation 5 (or Equation 4) will vary, making comparisons difficult. One approach to overcome this complication is to choose a small maximum value p' of p for consideration, and including only those terms for $t \geq p'$ in the product. Another approach is to divide the marginal likelihood of a model by the number of cases used to compute the marginal likelihood. The latter approach, which we use in our experiments, is justified by the prequential interpretation of the marginal likelihood as described by Dawid [5].

3.2 The Bayesian Score for an ART Model

Now let us consider the Bayesian score for an ART(p) model. To facilitate efficient computation, it is desirable to have model scores that can be computed in closed form and factor according to the structure of the decision tree (e.g., [2]). It is this desire that leads us to make the following two assumptions. One, the a priori likelihood of a model structure s is given by

$$p(s) = \kappa^{|\theta|}$$

where $0 < \kappa \leq 1$ and $|\theta|$ is the number of model parameters. In our experiments, we use a fixed value $\kappa = 0.1$, a value we have found to work well for many other domains. Two, the parameters $\theta_1, \dots, \theta_L$ —the parameters associated with the leaves of the decision tree—are mutually independent. Together, these assumptions imply

$$\text{score}(s) = \prod_{i=1}^L \text{LeafScore}(l_i) \quad (6)$$

where

$$\text{LeafScore}(l_i) = \kappa^{p+2} \int \prod_{x^t \text{ at } l_i} f_i(x_{p+1}^t | x_1^t, \dots, x_p^t, \theta_i, s) p(\theta_i | s) d\theta_i \quad (7)$$

where f_i is the Normal distribution corresponding to the linear regression at leaf l_i , as described in Equation 3. $\text{LeafScore}(l_i)$ is the product of (1) the prior probability of the leaf-component of the structure (there are $p+2$ parameters at each leaf) and (2) the marginal likelihood of the data that falls to the leaf.

The remaining ingredient for the Bayesian score is the parameter prior. In this work, we use the traditional conjugate prior for a linear regression. Namely, we assume that θ_i has a normal-gamma prior (e.g., [1]). In the Appendix, we derive the formulas for $\text{LeafScore}(l_i)$ given this prior. We note that this score can be computed in closed form and has a computational complexity of $O(p^3 + p^2 C_i)$, where C_i is the number of cases that fall to leaf l_i .

3.3 Searching for ART-Model Structure

In this section, we describe two approaches for learning the structure of ART models that utilize a model-structure score such as the one described in the previous section.

The first learning method is for learning the structure for an ART(p) model when we are given p ; the second method is for learning the structure for an ART model when we are not given p . When we describe our results in Section 5, we call the result of the first method an *ART model for known p* and the result of the second method an *ART model for unknown p* . Because AR models are a special case of an ART model we can also learn AR models with *known p* and *unknown p* . Each of these four methods is used in our experiments.

Our method for learning an ART model with known p —that is, for learning the split variables and split values for a decision tree when the possible split variables are limited to the previous p time periods and the leaves contain a regression on p regressors—is as follows. We use a greedy search algorithm that employs the operator *split-leaf*. The operator is applied to a leaf of a decision tree and takes two arguments: a variable to split on and the value of the variable. For instance, the decision tree in Figure 1 was obtained by the application of $\text{split-leaf}(Y_{i-1}, -337)$ to the single leaf of an empty decision tree (the root) followed by the application of $\text{split}(Y_{i-1}, 0)$ to the right child of the decision tree resulting from the first split. When applying the split-leaf operator to leaf l_i , we restrict attention to potentially splitting on seven values of each predictor variable; these values are the boundaries of eight equiprobable contiguous regions of a normal distribution estimated from the restricted data set at the leaf for the predictor variable (see [3] for a justification of the choice of eight regions). The initial decision tree is a decision tree with a single leaf, that is, no splits. We grow a decision tree by iteratively applying the split-leaf operator to each leaf for each variable, and choosing the split that results in the largest (non-negative) increase in model score. If no split on any leaf yields a higher score, then the search is terminated.

The greedy procedure is computationally tractable. Recall that a single evaluation of a split-leaf operator applied to leaf l_i has computational complexity $O(p^3 + p^2C_i)$, where C_i is the number of cases that fall to leaf l_i . In addition, for each leaf, we search among p potential split variables and search among (e.g.) $k = 8$ possible split points. Also, because the splits are binary, the number of leaf nodes that are evaluated for expansion is always less than twice the number of leaves in the resulting tree. Thus, because $C_i < T$, the overall learning time grows as $O(kL(p^4 + p^3T))$, where L is the number of leaves. As with other decision-tree learning algorithms, the learning time is a function of the size of the tree. Typically, as one increases the size of a data set, the size of the learned tree grows and thus the time to learn does not necessarily grow linearly in the size of the data set. Despite this potential super-linear scaling, we find empirically that decision-tree algorithms scale almost linearly for large data sets.

We learn an ART model with unknown p by repeatedly using the method for learning an ART model with known p . In particular, we learn an ART(i) model for each $0 \leq i \leq p_{max}$, and choose the model with the highest Bayesian score.

3.4 Forecasting with ART Models

We now consider the problem of using ART models to forecast. Given a sequence of observations, the task of forecasting is to calculate the distributions for future

observations in the sequence. We distinguish between two important types of forecasting: (1) *one-step forecasting* and (2) *multi-step forecasting*. In our evaluation of predictive accuracy, we use one-step forecasting.

In one-step forecasting, we are interested in predicting y_{T+1} given y_1, \dots, y_T are known. For this situation, the posterior distribution for the variable Y_{T+1} is a function of a single leaf node in the tree. In particular, using the conjugate priors we have described, each leaf in the tree has a conditional t-distribution for this variable. In our experiments, due to limitations in our implementation, we do not use the appropriate t-distribution to compute these log-likelihoods. Instead, we use the normal distribution $f_i(y_t|y_{t-p}, \dots, y_{t-1}, \theta_i)$ (described in Equation 3) evaluated at the value of θ_i that is most likely given the data—the *maximum a posteriori (MAP) value*:

$$\tilde{\theta}_i = \operatorname{argmax} \prod_{x^t \text{ at } l_i} f_i(x_{p+1}^t | x_1^t, \dots, x_p^t, \theta_i, s) p(\theta_i | s) \quad (8)$$

A detailed expression for $\tilde{\theta}_i$ is given in the Appendix.

In multi-step forecasting, we are interested in predicting values for variables at multiple future time steps. When forecasting more than one step into the future, a simple lookup is not possible due to the non-linearities in the AR tree. For example, we may have the ART model from Figure 1, and wish to predict Y_4 , Y_5 , and Y_6 when we know the values for only Y_1 and Y_2 . The prediction for Y_4 does not correspond to a single leaf because we do not know the value of Y_3 . In such situations, one can apply a computationally efficient Monte Carlo approach such as forward or logic sampling [12]. In this approach, one samples y_{T+1} given y_1, \dots, y_T , then y_{T+2} given y_1, \dots, y_{T+1} , and so on, using either the appropriate t-distribution or MAP (normal) distribution. These samples are then used to estimate quantities of interest such as the expected values and variances for variables at future time steps.

4 Evaluation

In this section, we provide an empirical evaluation of our methods and several alternative methods. To evaluate the methods, we use the 2,494 largest time-series data sets from the M3 competition available from the International Institute of Forecasters (<http://forecasting.cwru.edu/Data/m3comp/m3comp.html>). The time series vary in length from 23 to 126 periods with both the median and mean lengths larger than 60 periods. The data sets are from a variety of sources including industrial production data, micro and macro economic data, and finance data. We eliminate data sets smaller than 23 periods to allow for a fair comparison with one of the methods (MARS) which (quite reasonably) could not be applied to smaller data sets.

For each of the data sets, we create nine data sets using the length p transformation (described in Section 3.1) with p varying from zero to eight. Each of these transformed data sets is centered and standardized before modeling; that is, for each variable we subtract the mean value and divide by the standard deviation.

We divide each data set into a training set, used as input to the learning method, and a holdout set, used to evaluate the models. The holdout set contains the cases corresponding to the last five observations in the sequence.

We evaluate the quality of a learned model by computing the *sequential predictive score* for the holdout data set corresponding to the training data from which the model was learned. The sequential predictive score for a model is the average log-likelihood of the cases in the holdout set, where the log-likelihood of a case is the log probability that the model assigns to the observed value of the target variable given the observed predictor values. This log-likelihood computation is simply a one-step forecast. Finally, to evaluate the quality of a learning method, we compute the average of the sequential predictive scores for each of the 2,494 learned models produced by the learning method. Note that the use of the log-likelihood to measure performance simultaneously evaluates both the accuracy of the estimate and the accuracy of the uncertainty of the estimate.

In our experiments, we compare the following learning methods: the Salford Systems implementation of Friedman’s method for learning MARS (ASTAR) models [7], an implementation of a simple regression-tree learning algorithm described by Chickering et al. [3], and the methods described in Section 3 for learning AR and ART models.

We apply the learning methods to transformed data sets using nine different transformation lengths ($0 \leq p \leq 8$). Thus, for each data set, we learn nine MARS models, nine regression-tree models, nine AR models with known p , and nine ART models with known p . We additionally learn eight AR and ART models with unknown p ($0 < p_{max} \leq 8$).

5 Results

The results of our experiments are summarized in Figure 3. In this figure, we plot the average sequential predictive scores for the models learned by the candidate learning methods. The p^{th} point for a learning method is the average sequential predictive score for the 2,494 models learned by that method when we restrict the maximum dependency to length p . Note that the results of learning AR and ART models with known p will have a dependency length of exactly p . We emphasize the results for AR models with known p and for ART models with known p by connecting the points with a curve. We do not plot the results of the learning methods applied to the length-zero data transformations. The result of using an AR model on this transformed data obtains an average sequential predictive score of -9.03. The results of the other models are nearly identical.

The poorest performing models are the simple regression trees. The fact that ART models perform significantly better indicates that allowing non-trivial linear regression at the leaves of the decision tree is important when modeling a time series.

The MARS method performs significantly better than the simple regression tree but worse than the other methods. One possible explanation of this difference is that the forecasts for the target variable made by both AR(p) and ART(p) models

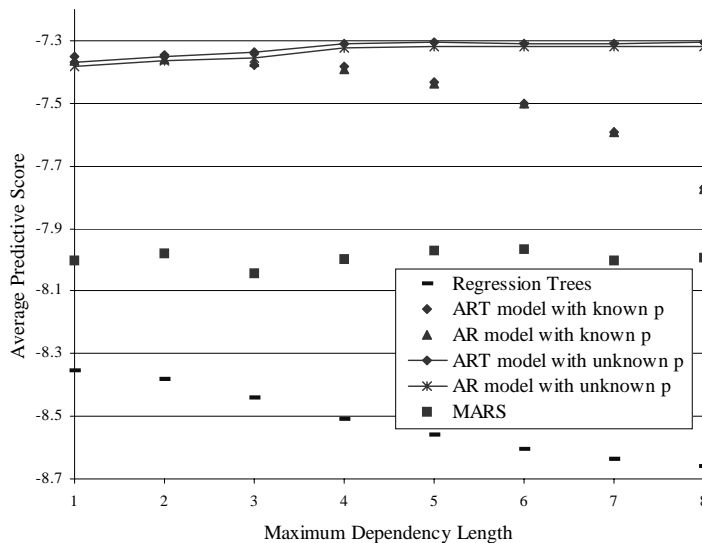


Figure 3. *Comparison of methods.*

necessarily depend on all of the p previous predictor variables. In contrast, the forecasts made by MARS models often eliminated some of these predictor variables. In fact, in many of the data sets, the backward step-wise elimination stage of the MARS method (see Friedman, 1991) eliminated all dependence on the predictor variables. Thus, $AR(p)$ and $ART(p)$ place more emphasis on recent time periods for prediction, which may have led to better predictions. Perhaps one could improve the performance of the MARS system by biasing the search and selection of knots towards recent time periods.

In Figure 4, we concentrate on the performance of the AR and ART models that are learned with known and unknown p . The fact that both the AR and ART curves with known p (corresponding to model selection for p) increase roughly monotonically is evidence that our approach to model selection is working. Additionally, the comparison of the respective “known p ” and “unknown p ” points for the AR and ART models indicate the benefits of model selection. Namely, as we include more models we are able to improve our predictive score—choosing models that provide good fit with without overfitting. Nonetheless, this benefit tapers off as p_{max} increases. Note that there is a large increase in performance for both AR and ART models when allowing dependencies at a lag of four. We believe this jump is due, in part, to the fact that roughly one fourth of our data sets are quarterly (e.g. quarterly sales results for a company).

In Figure 4, the curve for ART model with unknown p dominates the curve for AR model with unknown p indicating that the additional flexibility of ART models to model non-linear dependencies leads to better generalization. While the differences in predictive accuracy for various p_{max} are not as large as the differences between other methods, the improvement in performance is statistically significant.

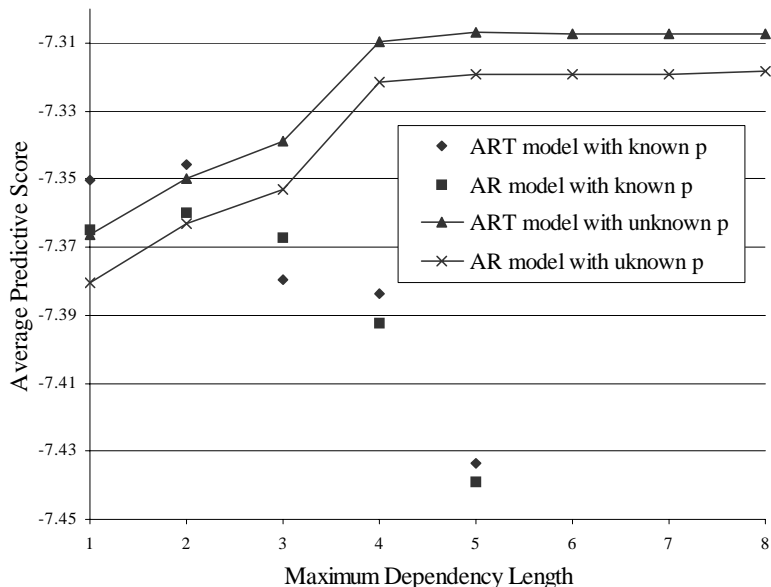


Figure 4. *Effectiveness of model selection and comparison of AR and ART models.*

The p-values for all values of p_{max} are less than 0.001 for the sign test and smaller for the Wilcoxon signed-rank test.

Recall that an ART model with no splits is an AR model; thus, the learning procedure for ART models can return an AR model. We note that in only 5% of the data sets did the resulting ART model have at least one split. The fact that a large fraction of ART models have no splits is due, in part, to the fact that the data sets are fairly short time series. This explanation is supported by the observation that the number of splits chosen by our method is correlated with the length of the time series. We suspect that the benefits of ART models with respect to AR models will increase when applied to longer time series.

6 Summary and Future Work

Our experimental results show that ART models are a useful family of models for time-series analysis and our Bayesian approach to learning ART models can identify predictively accurate models. In addition, coupled with our method for model selection, ART models provide a tractable family of models for modeling time-series data that improve on the standard AR models. Also, ART models provide a simple interpretable alternative to other non-linear techniques such as neural networks. Of course, there is more work to be done in understanding the benefits of ART models and in improving their performance on time-series prediction tasks.

In this paper, we have focused on our Bayesian approach to learning ART

models. Because AR trees are simply decision trees applied to time-series analysis, any system that learns decision trees with linear regressions at the leaves can be used to build an AR tree. Several such systems exist including the RETIS system [13] and the M5 system [15]. These two systems differ in that the RETIS system uses all of the features in the linear regression and uses the linear regression to evaluate the model during the greedy construction of the decision tree, whereas the M5 adds the linear regression while pruning the decision tree and can potentially prune features from the linear-regression. Another approach for learning decision trees with linear regressions at the leaves is described by Chipman, George, and McCulloch [4]. Their Bayesian approach uses a Markov Chain Monte Carlo technique to explore the space of alternative models. It would be useful to compare our approach with alternative methods (including alternative methods for learning decision trees) for the purpose of time-series analysis and other regression problems.

In this paper, we have focused on ART(p) models with the following two restrictions: (1) the split variables are chosen from among the p previous variables and (2) the leaves contain AR(p) models. Our method for learning ART(p) models can be adapted to remove these restrictions. First, we can expand the set of possible covariates for our split-leaf operator to include additional covariates (even discrete covariates, e.g., month in a daily time series). Second, we can include a search over subsets of regressors. It would be useful to investigate the benefits of these and other methods for relaxing the restrictions.

ART models and our method for learning them can also be adapted to the analysis of multivariate time series (e.g. sales results from more than one company). Multivariate time series can be handled in two ways: (1) learn decision trees with multivariate regressions in the leaves, or (2) learn a separate ART model for each time series allowing for regressors from all time series. We suspect that the second approach would be better, but further evaluation is needed.

We note that, by learning a single tree with MAP parameter values, our ART-model forecasting procedure ignores both structure and parameter uncertainty. Structure uncertainty (including uncertainty about split points) can be taken into account using full or partial model averaging (e.g., [1]), but the computational cost is often prohibitive. As we have discussed, parameter uncertainty can be taken into account in a computationally tractable manner using t-distributions. It would be useful to understand the relative importance of handling these two components of uncertainty.

Finally, one important aspect of modeling time series is handling seasonality in data. Seasonality can be handled using ART models by explicitly allowing or including relevant regressor variables in the linear regressions at the leaves. Alternatively one can use standard techniques to de-season the data prior to the construction of an ART model. Further experimentation is needed to characterize the performance of these alternatives.

Bibliography

- [1] J. Bernardo and A. Smith. *Bayesian Theory*. John Wiley and Sons, New York, 1994.
- [2] D. Chickering, D. Heckerman, and C. Meek. A Bayesian approach to learning Bayesian networks with local structure. In *Proceedings of Thirteenth Conference on Uncertainty in Artificial Intelligence*, Providence, RI. Morgan Kaufmann, August 1997.
- [3] D. Chickering, C. Meek, and R. Rounthwaite. Efficient determination of dynamic split points in a decision tree. In *Proceedings of the 2001 IEEE International Conference on Data Mining*, pages 91–98, San Jose, CA, November 2001. IEEE Computer Society.
- [4] H. Chipman, E. George, and R. McCulloch. Bayesian treed models, to appear in Machine Learning. Technical Report http://bevo2.bus.utexas.edu/GeorgeE/Research_papers/treed-models.pdf, University of Waterloo, 2001.
- [5] P. Dawid. Statistical theory. The prequential approach (with discussion). *Journal of the Royal Statistical Society A*, 147:178–292, 1984.
- [6] M. DeGroot. *Optimal Statistical Decisions*. McGraw-Hill, New York, 1970.
- [7] J. Friedman. Multivariate adaptive regression splines. *Annals of Statistics*, 19, 1991.
- [8] N. Gershenfeld and A. Weigend. The future of time series: learning and understanding. In *Time series prediction*, pages 1–70. AddisonWesley, New York, 1994.
- [9] J. Hamilton. *Time Series Analysis*. Princeton University Press, Princeton, New Jersey, 1994.
- [10] D. Haughton. On the choice of a model to fit data from an exponential family. *Annals of Statistics*, 16:342–355, 1988.
- [11] D. Heckerman and D. Geiger. Learning Bayesian networks: A unification for discrete and Gaussian domains. In *Proceedings of Eleventh Conference on Uncertainty in Artificial Intelligence*, Montreal, QU, pages 274–284. Morgan Kaufmann, August 1995. See also Technical Report TR-95-16, Microsoft Research, Redmond, WA, February 1995.
- [12] M. Henrion. Propagation of uncertainty by probabilistic logic sampling in Bayes' networks. In *Proceedings of the Second Workshop on Uncertainty in Artificial Intelligence*, Philadelphia, PA. Association for Uncertainty in Artificial Intelligence, Mountain View, CA, August 1986. Also in Kanal, L. and Lemmer, J., editors, *Uncertainty in Artificial Intelligence 2*, pages 149–164. North-Holland, New York, 1988.
- [13] A. Karalic. Employing linear regression in regression tree leaves. In *Proceedings of ECAI-92*, pages 440–441. John Wiley & Sons, 1992.
- [14] P. Lewis, B. Ray, and J. Stevens. Modeling time series by using mars. In *Time series prediction*, pages 297–318. AddisonWesley, New York, 1994.
- [15] J.R. Quinlan. Learning with continuous classes. In *Proceedings of of the 5th Australian Joint Conference on Artificial Intelligence*, pages 343–348. World Scientific, 1992.
- [16] H. Tong. *Threshold models in Nonlinear Time Series Analysis*. Springer-Verlag, New York, 1983.

Appendix

We derive formulas for $\text{LeafScore}(l_i)$ given by Equation 7, and for $\tilde{\theta}_i$, the MAP parameters for the linear regression at leaf l_i given by Equation 8.

Heckerman and Geiger [11]—herein HG—make the following assumptions for a set of observations $d = (x^1, \dots, x^N)$ where each $x^t = (x_1^t, \dots, x_{p+1}^t)$ is an observation over variables $X = (X_1, \dots, X_{p+1})$: (1) the likelihood of the data for a given model structure s is $\prod_{i=1}^N p(x_1^t, \dots, x_{p+1}^t | \mu, W, s)$ where each term is a multivariate-Normal distribution with unknown mean vector μ and precision matrix W , (2) $p(W | s)$ is a Wishart distribution, and (3) $p(\mu | W, s)$ is a multivariate-normal distribution. Under these assumptions, it follows that the relationship between X_{p+1} and X_1, \dots, X_p is the linear regression:

$$p(x_{p+1}^t | x_1^t, \dots, x_p^t, \theta, s) = N(m + \sum_{j=1}^p b_j x_j, \sigma^2), \quad t = 1, \dots, N \quad (9)$$

where,

$$m = \mu_{p+1} - \sum_{i=1}^p b_i \mu_i, \quad b_j = \sum_{i=1}^p (W^{-1})_{p+1,i} (((W^{-1})^{p \times p})^{-1})_{ij}, \quad \sigma^2 = 1/W_{p+1,p+1}. \quad (10)$$

In Equation 10, we use vector-matrix notation in which v_i denotes the i^{th} element of vector v , M_{ij} denotes the element in the i^{th} row and j^{th} column of matrix M , and $M^{p \times p}$ denotes the upper $p \times p$ sub-matrix of M . HG's assumptions also imply that $\theta = (m, b_1, \dots, b_p, \sigma^2)$ has a normal-gamma distribution. Thus, when we identify the cases in d with those that fall to leaf l_i and identify θ in Equation 9 with θ_i in the main body of the text, we find that HG's assumptions imply the conditions leading to the expressions for $\text{LeafScore}(l_i)$ in Equation 7 and $\tilde{\theta}_i$ in Equation 8. Thus, we use the HG framework to derive these quantities.

Following their approach, let $p(\mu | W, s)$ be a multivariate-normal distribution with mean μ_0 and precision matrix $\alpha_\mu W$ ($\alpha_\mu > 0$), and $p(W | s)$ be a Wishart distribution with α_W degrees of freedom ($\alpha_W > p$) and positive-definite precision matrix W_0 . Then, the MAP parameter values—those that maximize the probability of d given θ and s —are given by

$$\tilde{\mu} = \frac{\alpha_\mu \mu_0 + N \bar{\mu}_N}{\alpha_\mu + N} \quad \text{and} \quad \tilde{W}^{-1} = \frac{1}{\alpha_W + N - (p + 1)} W_N$$

where

$$W_N = W_0 + S_N + \frac{\alpha_\mu N}{\alpha_\mu + N} (\mu_0 - \bar{\mu}_N)(\mu_0 - \bar{\mu}_N)' \quad (11)$$

In these and subsequent equations, we use vector v to denote a column vector and v' to denote the transpose of v (a row vector). The terms $\bar{\mu}_N$ and S_N are the sample mean and scatter matrix, respectively, given by

$$\bar{\mu}_N = \frac{1}{N} \sum_{t=1}^N x^t \quad \text{and} \quad S_N = \sum_{t=1}^N (x^t - \bar{\mu}_N)(x^t - \bar{\mu}_N)' \quad (12)$$

We obtain the MAP values for $\theta = (m, b_1, \dots, b_p, \sigma)$ by transforming these expressions for $\tilde{\mu}$ and \tilde{W}^{-1} according to the mapping in Equation 10.

Given HG's assumptions, it also follows that the marginal likelihood is given by

$$p(d|s) = \pi^{-(p+1)N/2} \left(\frac{\alpha_\mu}{\alpha_\mu + N} \right)^{(p+1)/2} \frac{c(p+1, \alpha_W + N)}{c(p+1, \alpha_W)} |W_0|^{\frac{\alpha_W}{2}} |W_N|^{-\frac{\alpha_W + N}{2}} \quad (13)$$

where

$$c(l, \alpha) = \prod_{i=1}^l \Gamma \left(\frac{\alpha + 1 - i}{2} \right). \quad (14)$$

In addition, (X_1, \dots, X_p) has a (p -dimensional) multivariate-normal distribution with unknown mean and precision, which we shall denote μ^- and W^- , respectively. Furthermore, $p(\mu^- | W^-, s)$ has a multivariate-normal distribution with mean μ_0^- (the first p entries of μ) and precision matrix $\alpha_\mu W^-$, and $p(W^- | s)$ is a Wishart distribution with $\alpha_W - 1$ degrees of freedom and precision matrix W_0^- such that $(W_0^-)^{-1}$ is equal to the upper $p \times p$ sub-matrix of $(W_0)^{-1}$. Thus, if we let d^- be the data d restricted to the variables (X_1, \dots, X_p) , then the marginal likelihood $p(d^- | s)$ is given by the p -dimensional version of Equation 13, with μ_0 , W_0 , and α_W , replaced by μ_0^- , W_0^- , and $\alpha_W - 1$, respectively. Finally, the (conditional) marginal likelihood is given by

$$\int \prod_{t=1}^N p(x_{p+1}^t | x_1^t, \dots, x_p^t, \theta, s) p(\theta | s) d\theta_i = \frac{p(d|s)}{p(d^-|s)}. \quad (15)$$

Substituting the expression for $p(d|s)$ given by Equation 13 and the analogous expression for $p(d^-|s)$ into Equation 15, we obtain a formula for the marginal-likelihood component of LeafScore(l_i).

In our experiments, we use $\mu_0 = 0$, $W_0 = I$ (the identity matrix), $\alpha_\mu = 1$, and $\alpha_W = p + 2$ applied to data that has been standardized to have a mean of zero and a standard deviation of one.