Prequential Analysis of Complex Data with Adaptive Model Reselection[†]

Jennifer Clarke[1*] and Bertrand Clarke[1,2,3]

[1] *Department of Epidemiology and Public Health, University of Miami, Miami, FL 33136, USA*

[2] *Department of Medicine, University of Miami, Miami, FL 33136, USA*

[3] *Center for Computational Science, University of Miami, Miami, FL 33136, USA*

Received 16 September 2008; revised 10 July 2009; accepted 20 July 2009
DOI:10.1002/sam.10052
Published online 14 October 2009 in Wiley InterScience (www.interscience.wiley.com).

Abstract: In Prequential analysis, an inference method is viewed as a forecasting system, and the quality of the inference method is based on the quality of its predictions. This is an alternative approach to more traditional statistical methods that focus on the inference of parameters of the data generating distribution. In this paper, we introduce adaptive combined average predictors (ACAPs) for the Prequential analysis of complex data. That is, we use convex combinations of two different model averages to form a predictor at each time step in a sequence. A novel feature of our strategy is that the models in each average are re-chosen adaptively at each time step. To assess the complexity of a given data set, we introduce measures of data complexity for continuous response data. We validate our measures in several simulated contexts prior to using them in real data examples. The performance of ACAPs is compared with the performances of predictors based on stacking or likelihood weighted averaging in several model classes and in both simulated and real data sets. Our results suggest that ACAPs achieve a better trade off between model list bias and model list variability in cases where the data is very complex. This implies that the choices of model class and averaging method should be guided by a concept of complexity matching, i.e. the analysis of a complex data set may require a more complex model class and averaging strategy than the analysis of a simpler data set. We propose that complexity matching is akin to a bias–variance tradeoff in statistical modeling. © 2009 Wiley Periodicals, Inc. Statistical Analysis and Data Mining 2: 274–290, 2009

Keywords: model uncertainty; model selection; predictive optimality; Prequential analysis; Bayes model averaging; stacking; complexity

1. INTRODUCTION

When the true model F is unknown, a collection of candidate models, say, $f_1, f_2, \ldots, f_m$, can often be identified which would be useful to consider. We hope that at least one such f_i will be close to the true model F. At the same time, the f_is should be distinguishable from each other and span a neighborhood that contains F [1]. Given a data set, the ideal would be to find the f_i closest to F. Selection criteria such as AIC, BIC, or PRESS, among others, have been used to good effect [2]. Indeed, if we are satisfied that the f_i selected approximates the true model well enough, then using the selected model is defensible.

However, many authors have expressed concerns about classical model selection procedures. Several authors have argued that the uncertainty implicit in selecting a model is of primary importance; see Refs. [3–5]. Not only has model uncertainty relative to the list $f_1, f_2, \ldots, f_m$ been downplayed but also the uncertainty in forming the list itself has been ignored. Methods to account for these uncertainties have been proposed in the literature; these include Bayesian model averaging (BMA), ensemble learning [6,7], and weightings based on the bootstrap [5]. Two such techniques are relevant to this work, namely, stacking and likelihood weighted averaging (LWA).

As a brief synopsis of stacking and LWA, consider the usual 'signal plus noise' regression model of the form $Y = F(X) + \epsilon$ where ϵs are i.i.d. unimodal with mean zero and F is the unknown regression function. Suppose we have a sequence of outcomes $Y_1, Y_2, \ldots, Y_n$ to be predicted by the use of models $f_1, f_2, \ldots, f_m$. In stacking, the model coefficients are chosen to minimize the sum of squared prediction errors between the Y_i and the linear combination of predictors from the models $f_1, f_2, \ldots, f_m$ formed by

Correspondence to: Jennifer Clarke (JClarke@med.miami.edu)

[†]This work was partially supported by National Institutes of Health Grant 5K25CA111636.

a cross-validation criterion [8,9]. In contrast, BMA puts a prior on the models, as well as assigning priors within each model, and weights the models by their posterior probabilities; see Ref. [10]. In our research, we place a uniform prior on the models in the model space because the uniform prior has different support from time step to time step. As the posterior probabilities are proportional to the likelihood values, we call this procedure as LWA rather than BMA. Note that we are re-choosing the model list at each time step in response to residual errors. This means that we are treating the models as actions and updating the Bayes decision problem that the Bayes predictor is solving.

Unfortunately, using a weighted sum of models does not automatically account for model uncertainty because model list uncertainty has not been assessed. We address model list uncertainty by including it in the formation of our predictors. Our predictive procedure involves taking an average of averages, i.e. making predictions sequentially where at each time step the prediction is an average of a predictor based on stacking and a predictor based on model averaging. We call predictors generated by our procedure as ACAPs because our predictions are made sequentially, our predictors are adaptive, and variation due to model list reselection is implicit in the sequence of prediction errors our method generates. The motivating ideas behind ACAPs are that an extra layer of averaging will lead to better predictions, particularly in scenarios with complex data, and that improved prediction can be achieved by including the uncertainty in the model list in the predictive procedure, i.e. optimizing over a larger space as we optimize over model terms as well as model parameters.

The rationale for combining stacking and LWA is that the stacking predictor tends to have a lower predictive error than LWA in the presence of moderate-to-large model mis-specification, whereas the efficiency of LWA allows it to outperform stacking predictively when model mis-specification is negligibly small; see Ref. [11]. An alternative heuristic is that a convex combination of a set of candidate models achieves the minimum relative entropy; see Ref. [12]. The performance of an ACAP can be evaluated by its cumulative predictive error (CPE).

Out-of-sample prediction is done in the obvious way. For a given sequential data set, apply our procedure to it; this will give an out-of-sample prediction for each data point. For a given batch of data, choose m orderings of the data and apply our procedure to each of them. This will give m predictions for each data point, which can be averaged to give an overall prediction that can be regarded as independent of the ordering. In other words, the output of our procedure at time n is exactly the predictor one would use for time $n + 1$, if one existed. We comment further on this in Section 6.

Provided the models chosen are reasonable, including more and more models in a convex combination should, in the limit, give better performance in terms of bias, although not necessarily in terms of variance. Analogously, a convex combination of predictors should give better performance as one includes more predictors as long as the extra predictors reduce bias enough to compensate for the concomitant increase in variance. Thus, averaging two different averages should give better prediction than either average alone, especially in problems where the true model is very complex. This explains how ACAPs, with an extra layer of averaging and model space searching, improve the tradeoff between bias and variance for complex data, thereby giving better predictive performance than the individual component averages.

Unlike the perceptive work of Domingos [13] which evaluates bias and variance relative to an average over predictors called a main prediction, we do not offer a variance/bias/noise decomposition for our results. Instead, we interpret our results in terms of complexity, admitting this concept is not well defined outside of algorithmic or information-theoretic contexts. Here, we define complexity in terms of characteristics of the response given a set of predictor values, an idea that has been explored in the pattern recognition literature [14]. The characteristics of interest are obtained from the distance matrix D whose elements are the pairwise distances between the response values. The three complexity measures are the skewness of D (Eq. (9)), the ratio of positive to negative correlations (Eq. (10)), and dimension as defined by principal components analysis (PCA) (Eq. (11)). These ideas are well established in statistical theory, e.g. PCA dimension is an example of intrinsic dimension, an idea relevant to dimension reduction [15,16].

In our examples with high complexity data sets, we consider three model classes for prediction, namely, linear models, generalized additive models (GAMs), and recursive partitioning models. These classes were chosen to represent a range of model complexity and mathematical form. The selection of model terms is an effort to balance breadth from random selection of new terms and parsimony from model reduction to only those terms which provide good prediction.

The structure of this paper is as follows. In the next section, we give a brief introduction to Prequential analysis as a framework for statistical inference. We then introduce our ACAP approach to Prequential analysis of complex data in Section 3, where we present our technique and demonstrate ACAP performance on simulated data. In Section 4, we describe measures of data complexity that we have developed and demonstrate their behavior on simulated data. We present the results of the ACAP analyses of several complex data sets in Section 5. These results demonstrate that in a full optimization over model classes and averaging strategies ACAPs achieve the best predictive results. Section 6 contains a discussion of our results and relates our conclusions to the concept of complexity matching and its relevance to statistical modeling.

2. PREQUENTIAL ANALYSIS

The Prequential setting is a natural context for investigating complex data because sequential prediction allows information to be extracted from data that could not be extracted as readily if the data were treated as an undivided batch. This is seen in Ref. [17] in which using the information in the sequence of residuals to evaluate risk, in finite sample i.i.d. linear regression settings, predictively outperforms asymptotically optimal Bayesian batch analysis. The basic object of the Prequential treatment of data is a predictor function denoted by $\hat{Y}_{t+1}$, formed from all information available up to time t, which makes a testable statement about an occurrence at time $t+1$; see Refs. [18,19]. Predictor sequences $\langle \hat{Y}_t \rangle$ may give probabilities, outcomes, decisions, or other quantities whose performance can be evaluated at each time step. The great benefit of sequential prediction is that the choice and performance of a predictor sequence up to time t is information that can be used to choose a predictor for time $t+1$.

Despite variations among authors, every treatment of the Prequential approach includes at least three components:

(1) Predictor sequence: At each time step, a Forecaster, F, is required to issue a prediction for $t+1$ using all information available up to time t (in particular, all past data) and F's performance will be evaluated by how well $\hat{Y}_{t+1}$ matches Y_{t+1}, the correct answer at time $t+1$.

(2) Prequential principle: This is usually stated as the evaluation of F's forecasting strategy should depend only on the actual forecasts issued. See Refs. [18,19] for further variations on this.

(3) Updating: Upon receipt of the outcome at $t+1$, F is permitted to reformulate elements of the forecasting strategy in view of the new information. This may be simple or elaborate; see Ref. [20] or [21], for examples.

Evaluation of Prequential schemes in theory has been extensive, including calibration [22], efficiency [23,24], model fit [25,26], and comparative performance to Bayes methods [17]. The general formulation in terms of loss functions is amenable to a worst case analysis; see Ref. [27]. This is similar in spirit to coding in the worst case scenario [28–30] and to more recent work on mixture strategies and oracle inequalities [31,32].

To a substantial extent, the Prequential approach can be regarded as the automation of residual analysis in linear regression, but done predictively. At each time step, a model space must be chosen, a model must be selected from it, the coefficients in the model must be estimated, and the resulting estimated model must be used to form a prediction for the next time step. Thus, there is information in the sequence of residuals, the estimated parameters, and the chosen models. In particular, these sequences indicate how rapidly the complexity of modeling can increase (or decrease) with n and what sort of model deficiencies are hardest or most important to correct. This analysis is conditional on having a fixed sequence of the data; if the data has no natural sequence an analysis can be conducted on multiple permutations of the data and results averaged over permutations.

3. ADAPTIVE COMBINED AVERAGE PREDICTORS

ACAPs form a method for Prequential analysis in which model lists and model averages are updated adaptively. This level of adaptation is advantageous in complex modeling scenarios.

3.1. Formal Description of Predictors

Let $\mathcal{E}$ be an ensemble of terms so that all the models we intend to consider can be written as a linear combination of elements in $\mathcal{E}$. As in regression, the parameters are taken to be the coefficients on the terms. We denote a generic model as f, or f_i when it is in a list of models, and as F when it is taken as true. We denote lists of models formed from $\mathcal{E}$ generically by $\mathcal{M}$, or $\mathcal{M}_t$ to indicate dependence on the tth time step. For m models, $\mathcal{M}_t = \{f_1^t, f_2^t, \ldots, f_m^t\}$. In general, $\mathcal{M}_t \not\subset \mathcal{M}_{t+1}$ and $\mathcal{M}_{t+1} \not\subset \mathcal{M}_t$.

Given a model list and a set of weights λ_k, we can form the model average predictor

$$\hat{Y}_t = \sum_{k=1}^{m} \lambda_k Y_t^k, \tag{1}$$

where Y_t^k is the predictor derived from the kth model in $\mathcal{M}_t$, denoted f_k^t. A predictor is characterized by the model list, the weights on the models, and the estimates of the parameters in the models. The weights λ_k, like the parameters, must be estimated from the data sequence. The performance of a predictor can be assessed by its prediction error, or CPE, at each time step.

One of the novelties of our approach is the use of model averaging procedures in a sequential setting. These procedures are often regarded as too complex to allow the re-use of parts of earlier iterations. Nevertheless, at a given time, our procedure does use the previous model list in determining the current model list. Our work shows that the computational burden is not as high as may be feared; see Table 2.

Stacking is motivated by cross-validation; it uses weights λ_k for the f_is that achieve

$$\text{error}_{\text{CV}} = \min_{w_1,\ldots,w_K} \sum_{i=1}^{t} \left(y_i - \sum_{f \epsilon \mathcal{M}_t} w_f f(\mathbf{x}_\text{i}) \right)^2, \quad (2)$$

given a list $\mathcal{M}_t$ at time t. Following Wolpert [8] or Breiman [9], the variant we use here does not weight the fs directly but uses a sort of leave-k-out linear regression. Specifically, let $\hat{f}^{-j}(x)$ denote the predictor from a model f evaluated at x where the coefficients in f are estimated using the past data except the data in the jth hold out set, denoted $\mathcal{D}_j$. The stacking coefficients result from minimizing

$$\text{error}_{\text{S}} = \sum_{j=1}^{5} \sum_{i \in \mathcal{D}_j} \left(y_i - \sum_{f \epsilon \mathcal{M}_t} w_f \hat{f}^{-i}(\mathbf{x}_\text{i}) \right)^2, \quad (3)$$

over the w_fs; see Ref. [33]. There are various classes of weights w_f over which to optimize, leading to different stacking coefficients; see Ref. [11]. We have used the most general optimization, i.e. we do not ensure the optimal w_fs are positive or that they sum to 1. We made this choice because the narrower the class of w_fs one uses, the more stacking resembles LWA, obviating the point in averaging the two procedures. [We caution that some of our computations (not shown here) indicate that convex combinations tend to outperform non-convex combinations predictively. So, it is possible that imposing the convexity constraint in the stacking optimization would lead to a better version of stacking.]

The other model average, LWA, is Bayesian. Let $\boldsymbol{Z}$ denote all the data, i.e. $\boldsymbol{Z} = (\boldsymbol{Y}, \boldsymbol{X})$ where $\boldsymbol{Y} = (Y_1, Y_2, \ldots, Y_t)$ and $\boldsymbol{X} = (X_1, X_2, \ldots, X_t)$. The posterior distribution of Y_{t+1} given $\boldsymbol{Z}$ is $P(Y_{t+1}|\boldsymbol{Z}) = \sum_{i=1}^{K} P(Y_{t+1}|f_i^t)$ $P(f_i^t|\boldsymbol{Z})$, which has posterior mean $E(Y_{t+1}|\boldsymbol{Z}) = \sum_{i=1}^{K}$ $E(Y_{t+1}|f_i^t)$ $P(f_i^t|\boldsymbol{Z})$. This is a weighted average of the predictors derived from the individual models with weights given by the posterior probability of each model.

To have a uniform procedure for comparing the model classes, we approximated $E(Y_{t+1}|\boldsymbol{Z})$ by the sum of $\hat{f}_i^t(Z)w(f_i|Z)$ where $\hat{f}_i^t(Z)$ is a predictor of Y_{t+1} based on model i and $w(f_i|Z)$ the absolute value of the model deviance (−2× log-likelihood), where the deviances have been scaled to sum to 1 across models. By the use of $\hat{f}_i^t(Z)$ we have avoided having to assign priors within models. In the linear models, case earlier computations (data not shown) showed that using the posterior mean under normal within-model priors or the least-squares estimator made little difference relative to other sources of error. More generally, if we wanted to be strictly Bayesian, it is unclear how to do compatible within-model prior specification across model classes. Our use of model deviance ensures that our across model prior specifications are broadly consistent. As stated, we have chosen a uniform prior over the models in the model space, so the posterior weights are proportional to the model likelihoods.

To fix notation, suppose $\hat{Y}_{t+1}^{\text{S}}$ is the predictor from the stacking model list $\mathcal{M}_t^{\text{S}}$ when the λ_ks come from the stacking optimization and $\hat{Y}_{t+1}^{\text{LWA}}$ is the predictor from the LWA model list $\mathcal{M}_t^{\text{LWA}}$ when the λ_ks are posterior weights.

It remains to combine a stacking predictor and a LWA predictor to form an ACAP. We define the ACAP to be the weighted average of these two, that is

$$\hat{Y}_{t+1} = \alpha_t \hat{Y}_{t+1}^{\text{S}}(x_t) + (1 - \alpha_t)\hat{Y}_{t+1}^{\text{LWA}}(x_t), \quad (4)$$

where $0 \leq \alpha_t \leq 1$. For each time step, an optimal α_t can be obtained from the previously observed data by a least-squares criterion. To form the predictor for the $t + 1$ time step, we update α_{t-1} to

$$\alpha_t = \arg\min_{\alpha} \sum_{i=t-n'}^{t} \left(Y_i - [\alpha \hat{Y}_i^{\text{S}} + (1-\alpha)\hat{Y}_i^{LWA}]\right)^2. \quad (5)$$

The minimization in Eq. (5) can be done in closed form leading us to choose

$$\alpha_t = \frac{(\hat{\mathbf{y}}^{\text{LWA}} - \hat{\mathbf{y}}^{\text{S}})^T(\mathbf{y} - \hat{\mathbf{y}}^{\text{S}})}{(\hat{\mathbf{y}}^{\text{LWA}} - \hat{\mathbf{y}}^{\text{S}})^T(\hat{\mathbf{y}}^{\text{LWA}} - \hat{\mathbf{y}}^{\text{S}})}, \quad (6)$$

where $\mathbf{y}$, $\hat{\mathbf{y}}^{\text{LWA}}$, and $\hat{\mathbf{y}}^{\text{S}}$ are the data vector, LWA predictions, and stacking predictions for time steps $[t - n', \ldots, t]$, respectively. When $\alpha < 0$ or $\alpha > 1$, we set it to 0 or 1, respectively.

In updating α_t, we use only the most recent n' data points, here the last $n' = 20$, although the other parameters in the predictors are permitted to depend on all the accumulated data. Finding $\hat{\alpha}$s by using only the most recent data points gave better performance than using all of the accumulated data (results not shown). We conjecture that these choices worked well because they were consistent with the different convergence rates between parameters and model selection for the two model averages. It has been implied that LWA converges to the point in its support closest to the true distribution in relative entropy, and convergence in the discrete case is exponentially fast [34]. However, the coefficients in a stacking average do not make use of the information in the likelihood and hence stacking averages converge more slowly than LWAs. Otherwise put a single parameter estimate converges to its limit often at rate $O(1/\sqrt{n})$ as data accumulates, and this is much faster (in terms of the sample size n) than a model list can converge to an 'optimal' model list. Limiting the amount of data used to estimate the α_ts avoids the possibility that a difference in convergence rates will let one convergence dominate the other.

Our examples are based on the evaluation of stacking, LWA, and ACAP by their CPE in settings of high data complexity. For a generic predictor $\hat{Y}$ at time i evaluated at x_i, the error is

$$\mathrm{CPE}(\mathcal{M}_{t+1}) = \sum_{i=1}^{t+1} \left(\hat{Y}_i(x_i) - Y_i\right)^2. \tag{7}$$

In Eq. (7), we have re-indexed so that $i = 0$ corresponds to the last time step of burn-in. We found it necessary to use a minimal burn-in data subset to obtain preliminary estimates for the model weights and the parameters in the models. An initial model list is specified, and the models on the list are fit to the burn-in observations (here, the first 30 observations); the sequential process begins with the first observation after burn-in.

3.2. Updating Model Lists

Suppose we begin with a model list $\mathcal{M}_t$ of cardinality K at time t. We want to update $\mathcal{M}_t$ by the use of $x_1, \ldots, x_t$ and $y_1, \ldots, y_t$ to form a new model list $\mathcal{M}_{t+1}$ also of cardinality K; we will use $\mathcal{M}_{t+1}$ to form a predictor $\hat{Y}_{t+1}$ which we evaluate at x_{t+1} to predict Y_{t+1}. We use the same updating procedure for both the stacking and LWA model lists and then combine the two for the ACAP.

We begin by choosing an initial model list by selecting terms at random to be included in each model; the ensemble of possible terms includes single terms (x_i), two-way interaction terms ($x_i x_j$), and squared terms (x_i^2). Both LWA and stacking begin with the same initial list. Our procedure has two stages. First, we add models that we think will improve prediction. Second, we remove models whose contribution to good prediction is minimal.

Our procedure constructs three candidate models for addition to the list by a random search around a 'midpoint of the model list', which represents its 'center', and then tests whether one of the models on the list should be replaced by a candidate. A term is included in the 'central' model if the majority of the models in $\mathcal{M}_t$ include the term, i.e. the 'central' model m_t^{c} is formed by a majority vote on the terms in $\mathcal{M}_t$. Next we consider any model formed by m_t^{c} plus one additional term; the model with the lowest AIC [35] is the first candidate. We use AIC as defined by

$$\mathrm{AIC}(m) = -2L(m) + 2q, \tag{8}$$

where $L(m)$ is the log-likelihood of model m and q the number of degrees of freedom in m. For GAMs, the value of q in Eq. (8) is the effective degrees of freedom of the fitted model [36]. We comment that in linear models, AIC has been shown to be asymptotically equivalent to leave-one-out cross-validation [37] and is consistent as a model selection procedure for linear models if the dimension of the true model increases with n at an appropriate rate [38]. For trees and GAMs, these do not hold in general. GAMs have unavoidable bias for functions that are not additive; estimating the splits in a tree model rests on a cost complexity pruning whose consistency and predictive properties have not been elucidated, cf., Rao [39] and Nobel [40].

Next we consider the collection of models formed by removing a term from m_t^{c}; the model in this collection with the lowest AIC is the second candidate. The third candidate is formed by a random selection of terms where the probability of selecting a term for inclusion is the proportion of terms which appear in m_t^{c}. Of these three candidate models, the model with the lowest AIC is the new candidate model m_t^{new}.

To decide whether or not to replace one of the models in $\mathcal{M}_t$ by m_t^{new}, we perform five-fold cross-validation on $\{x_1, \ldots, x_t, y_1, \ldots, y_t\}$ with each possible sublist of $\{\mathcal{M}_t, m_t^{\mathrm{new}}\}$ of cardinality K. The sublist with the minimum cross-validation error is defined as $\mathcal{M}_{t+1}$; the omitted model is discarded. We take the convex combination of these two values as the ACAP prediction as in Eqs. (4) and (5).

Each model list (LWA or stacking) contained five distinct models; each model was represented by a list of terms from $\mathcal{E}$. The models on the lists were constructed by the model fitting procedures and averaged using either LWA or stacking weights. This choice of five models per list was a balance between the desire to include many possible models and the results of previous work which suggest that the formation of distinct models becomes difficult as the number of models increases because of limited sample size.

A drawback of this procedure is that output model lists can depend on the choice of initial model list. We have chosen to randomize over the initial choice of model list in each experimental run (see Section 5) and average over the predictions. Our method also includes a random search at each time step, an effective way to avoid oversensitivity to initial conditions. Our conclusions would be broadly the same if each of the models were replaced by individual fixed models [41].

3.3. Simulation Results: ACAP

We have compared the performance of LWA, stacking, and ACAP on simulated data with models chosen from three classes, namely, linear models, GAMs [42,43], and recursive partitioning models (trees) [44]. These choices of model classes will allow a comparison of parametric and non-parametric approaches, both statistical and rule-based, and has been used previously in the literature [45].

The data sets were generated from a process of the form $Y_t = F(X_t) + \epsilon,\ i = 1, \dots, N$, where $N = 100$ and the X_ts are independent penta-variate normals with mean zero and variance the identity matrix. The error terms are i.i.d. $N(0, 1/4)$. A total of 100 such data sets were generated from each of 15 different functions F constructed by combining the base terms $F_1 = -|x_1|^{(3/2)}$, $F_2 = |x_1|^{(7/2)}$, $F_3 = x_5|x_1|^{(1/2)}$, and $F_4 = \text{sgn}(x_1)|x_1|^{(1/2)}$. Each function was assigned to a class: class 1 consisted of all single base term Fs, class 2 consisted of all 6 two-term Fs, class 3 had 4 three-term Fs, and class 4 had 1 four-term F.

Consider the representation of a model in a given model list as a collection of terms. In the linear model case, these terms represent the predictors in the model and can be used *as is*. These terms cannot be used in the corresponding GAM as is; instead the GAM contains cubic smoothing spline predictors [46], one for each term. The GAM has an identity link for the mean and a log link for the standard deviation; this model is an example of a GAM for location, scale, and shape [43]. The model fitting is achieved via the GAMLSS software [47]. As for recursive partitioning models, there are many variations of the basic tree algorithm from different choices of splitting rules and stopping rules to different statistical models for Y in a given terminal node [44,48,49]. As our interest is model class comparison and not the intricacies of trees, we chose the well-established algorithm for tree development provided in the R `tree` package [50]. In this algorithm, the terms in the model list for a given model are not included directly in the tree model. Instead the tree algorithm selects the terms from the list (and the split points) which provide the best fit. As a result terms in the model list may or may not appear in the resulting tree.

Summary barplots of the results and some representative line plots are presented in Fig. 1. ACAP outperforms stacking, in CPE, and provides results better or equivalent to those of LWA. Figure 1(right) contains plots for one function from each class; these plots are representative of cases where the results of all three methods are similar (upper left subplot), stacking outperforms LWA (upper right subplot), and LWA outperforms stacking (lower left subplot).

4. DATA COMPLEXITY

We have presented an approach to the Prequential analysis of complex data. But how do we define ‘complex’? In general, a system is complex if it consists of elements or components that are difficult to distinguish or whose interdependencies are hard to follow [14]. More complex systems have more components and more interdependencies, so complexity can be characterized by variability, i.e. the number of elements and their size and shape, as well as dependency between components.

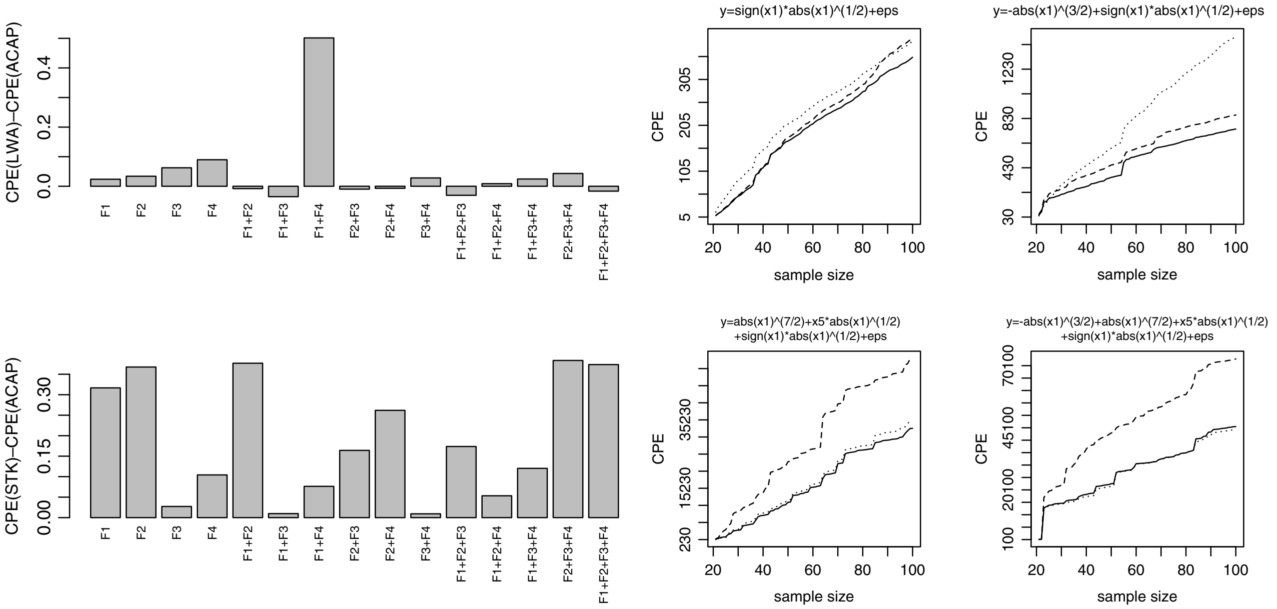

Fig. 1 CPE results for simulated data from 15 functions from four classes (left). The horizontal axis is function; the vertical axes are CPE(LWA)-CPE(ACAP) and CPE(stacking)-CPE(ACAP), calculated separately for each function (right). CPE results for one function from each class. The horizontal axis is sample size; the vertical axes are CPE. LWA results are dotted lines, stacking results dashed lines, and ACAP results solid lines. CPE results are averaged over 100 runs.

The number of elements and their dependencies are represented by the set of available examples from the system, i.e. the data set. This implies that the complexity of a regression (or classification) task depends on both the choice of variables and the sample size. The outcome of interest may have some hidden structure that is only partially reflected in the set of predictors. The structure of the outcome may be evident from a small set, or require a large set, of predictors and observations. The link between complexity and variable selection is one of the principles behind dimension reduction techniques: see Refs. [51–53].

Note that a data set may be complex from a modeling perspective for reasons independent of dimension and sample size. For example, the distribution of the response may be ambiguous either intrinsically or due to inadequate feature measurements [54]. Similarly, the response may have a form not considered by existing modeling techniques, so that the limitation is the modeling approach. Reports on data complexity in classification contexts can be found in the literature [14,55,56] although similar reports in regression contexts are rare [57].

The assessments of complexity we propose are derived from dissimilarity matrices D; see Section 4.1. Similar metrics were initially developed for pattern recognition problems [14,58]; they are fully non-parametric in that they make no strong distributional assumptions. The main assumption required here is that if these assessments are regarded as random variables having a distribution derived from the true distribution, then D must have at least three moments.

We comment that there is also an organized theory for complexity based on minimum description length or minimum message length [59–61]. We have not used this formulation because it depends heavily on assigning codes to classes of distributions.

4.1. Measures of Complexity

Let $D(Y, Y)$ be a $n \times n$ dissimilarity matrix, where $Y = \{y_1, y_2, \ldots, y_n\}$ is a response vector of size n and each observation is represented by a vector of predictor values x_i of length p. The (i, j)th element of D, denoted $D(y_i, y_j)$ or d_{ij}, is the dissimilarity between responses i and j. $D(y_i, Y)$ is the vector of pairwise dissimilarities between y_i and each observation in Y. We define dissimilarity in terms of Gower distance [62] or Minkowski distance of order 2 (L_2 distance) [63].

The complexity measures we use here are a function of D only. This arises because we first cluster the observations in X, i.e. cluster observations based on the similarity of their predictor values. Complexity is calculated based on the responses for the observations in each cluster and then a weighted sum is taken over clusters (with weights proportional to cluster size) to yield a complexity measure for a data set. In other words, we form a non-parametric model for Y in terms of X and then look at the complexity of Y conditional on the clusters. We chose the k-means algorithm [64] for clustering because it is well established in the literature and relatively simple. In our examples, we set $k = 5$ as a trade off between too small clusters for small n and too large clusters for large n.

Our first measure is based on skewness and assesses directly the distribution of the pairwise dissimilarities between the response values. The *skewness* measure for any two objects i and j is

$$C_{\text{sk}} = E\left[\frac{d_{ij} - E(d_{ij})}{\sqrt{E(d_{ij} - E(d_{ij}))^2}}\right]^3. \tag{9}$$

If the data generator is complex and the data set is small, adding a new observation can generate many large dissimilarities and few small ones. As a result, the distribution of dissimilarities will have a peak at small values and a long tail in the direction of large values. Eventually, adding new objects will generate only small dissimilarities. So, with an increase in the cardinality of Y skewness will grow but eventually converge or stabilize. This stabilization will occur at smaller sample sizes for simpler problems. Simple problems should converge to higher skewness values than complex problems, as simpler problems will have a smaller mean dissimilarity.

Our second measure is based on correlation and captures the idea that similar responses show similar dissimilarities to other responses and is, thereby, positively correlated. The *correlation* measure is defined as the ratio of the average of positive correlations to the average of the absolute values of the negative correlations between the columns (or rows) of D, i.e.

$$C_\rho = \frac{1/(n^2 - n)\sum_{i,i \neq j} \rho_+(D(y_i, Y), D(y_j, Y))}{1 + 1/(n^2 - n)\sum_{i,i \neq j} \mid \rho_-(D(y_i, Y), D(y_j, Y)) \mid}. \tag{10}$$

For a well-sampled data generator, this measure will be large, as new observations will be similar to existing observations, and will increase slightly when new observations are added. As with skewness, correlation will grow as the cardinality of Y increases but it will converge at a smaller sample size and to a higher value for simpler problems then for more complex problems.

Our final measure is based on PCA and assesses the similarities between responses, where each response is represented by its vector of pairwise dissimilarities to other responses. The *PCA* measure of D is defined as

$$C_{\text{pca}} = \frac{n_\alpha}{n}, \tag{11}$$

where n_α is the smallest integer such that $\sum_{k=1}^{n_\alpha} \lambda_{(k)} / \sum_{k=1}^{n} \lambda_k \geq 0.95$ where $\{\lambda_1, \lambda_2, \ldots, \lambda_n\}$ are the eigenvalues of D and $\lambda_{(k)}$ is the kth largest eigenvalue of D. The faster the value of PCA drops and converges with increasing sample size, the smaller the intrinsic dimension of the dissimilarity space representation. Intrinsic dimension has long been recognized in the literature as an important measure of data set complexity [16,65–68] and our use of the PCA metric reflects this. We expect the PCA metric to converge at a smaller sample size and to a smaller value for simpler problems relative to more complex problems.

4.2. Simulation Results: Complexity

We evaluated our complexity measures on simulated data sets to determine whether they capture data set complexity across a range of outcome distributions, sample sizes, and number of predictors. The sample size ($n = 300$) and dimensions ($p = 2, 10, 50$) were selected for ease of comparison with the results from our real data examples in Section 5. As the results for varying dimension were qualitatively the same, we discuss only the results for $p = 10$. Our simulation data sets are described below.

(1) Normal: The response $\mathbf{Y} = \mathbf{X}\psi + \epsilon$ where $\mathbf{X}$ contains n observations of length $p(\{x_1, x_2, \ldots, x_n\}$; $x_{ij} \sim N(0, 1)$ for $i = 1, \ldots, n$, $j = 1, \ldots, p$, $\psi = 1$ for $j = 1, \ldots, p$, and $\epsilon_{\mathbf{i}} \sim N(0, 0.5)$ for $i = 1, \ldots, n$. This is a common regression context.

(2) ARMA: The same model as (1) except $\epsilon_i \sim$ ARMA (2,2) for $i = 1, \ldots, n$, with random normal innovations with sd $= 0.5$. This scenario should be slightly less complex than 1 because the error has more structure.

(3) Uniform: The model for $\mathbf{X}$ is the same as in (1). However ϵ and ψ are discarded and $y_i \sim U(-5, 5)$ for $i = 1, \ldots, n$. As the response is independent of the predictors and has high entropy, we expect this scenario to be of higher complexity than scenarios 1 and 2.

(4) Nonlinear unbiased: The response $\mathbf{Y} = \mathbf{X}^3\psi + \epsilon$ where $\mathbf{X}$, ψ, and ϵ_i are as in 1. As $x_{ij} \sim N(0, 1)$ for $i = 1, \ldots, n$, this nonlinear scenario is compressed relative to scenario 1, e.g. compressed predictors and a compressed response. Clustering is done on $\mathbf{X}^3$, not $\mathbf{X}$. We expect this scenario to be the least complex.

(5) Nonlinear bias #1: This scenario is the same as in (4) except clustering is done on $\mathbf{X}$, not $\mathbf{X}^3$. The data is compressed but the predictors used for clustering are not compressed, which introduces bias into the modeling. This scenario should be at least as complex as scenario 4.

(6) Nonlinear bias #2: This scenario is the same as in (4) except clustering is done on $\mathbf{X}^3$, not $\mathbf{X}$. The data is not compressed but the predictors used for clustering are compressed, which introduces bias into the modeling but in a direction opposite to that in 5. This scenario should be at least as complex as scenario 4.

The parameter values used in these simulations were determined by choosing values which seemed intuitively simple/complex. A total of 30 data sets were generated for each scenario described above, and the complexity of a scenario was defined for each measure as the average value of the measure over data sets.

The results of our simulations are shown in Figs. 2 and 3. In each figure, the subplots in the right column display a portion of the region shown in the left column. All measures report decreasing complexity as sample size increases. Skewness and correlation decrease to a non-zero asymptote, i.e. a limiting complexity. However, PCA dimension decreases to an asymptote at zero, so the maximum value of this measure (and its rate of convergence to zero) is taken as an indication of complexity. The results in Fig. 2 confirm that, as expected, the uniform scenario is the most complex while the nonlinear unbiased scenario is the least complex. The ARMA scenario is slightly less complex than the normal scenario because the noise has relatively more structure.

Scenarios 4–6 are plotted in Fig. 3 as they represent cases with and without modeling bias; biased scenarios should have higher complexity. In the scenario of clustering linear predictors but generating data from nonlinear predictors, the added bias leads to less complexity by skewness but more complexity by correlation and PCA dimension. In contrast, clustering nonlinear predictors but generating data from linear predictors leads to more complexity by skewness and correlation but no change in complexity by PCA dimension (the PCA results of these two scenarios overlap and cannot be distinguished).

Clearly no one measure is able to capture all aspects of complexity. Although all three measures agree in scenarios of complexity based primarily on variability (scenarios 1–4), complexity based primarily on bias is more difficult to capture. Some directions of bias are detected by skewness and correlation (scenario 6) while others are detected by correlation and PCA dimension (scenario 5). In essence, it is the trio of values which indicate the nature of the complexity, primarily in terms of the relative level of variability and the direction of bias.

5. EXAMPLES

Given the behavior of the complexity measures in simulated scenarios, we are able to use them to assess data

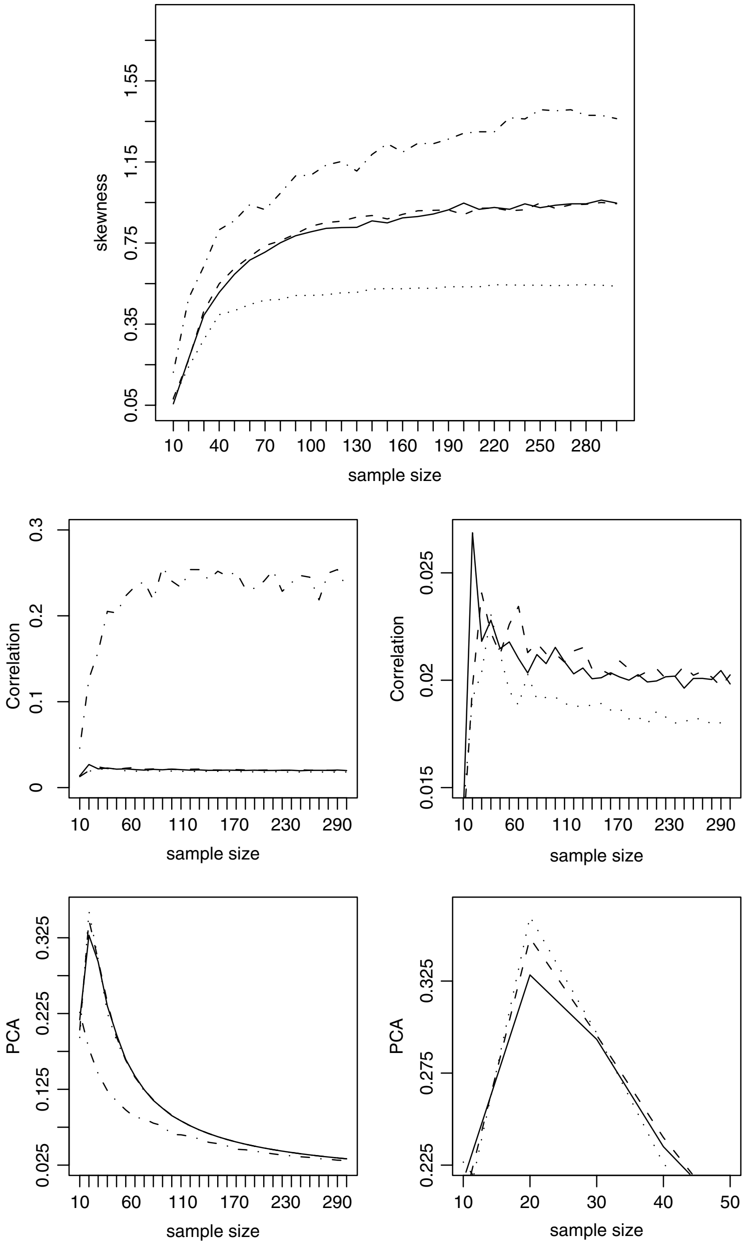

Fig. 2 Complexity for simulated data with respect to variance, $p = 10$. Complexity values for scenarios 1–4 are plotted for each measure. The line types correspond to the different scenarios: solid, normal; dashed, ARMA; dotted, uniform; and dotdash, nonlinear unbiased.

set complexity in real data examples. These examples were chosen to compare the predictors from LWA, stacking, and ACAP methods by their CPE (Eq. (7)). We compare linear models, GAMs, and trees on three data sets:

(1) *Friedman*: This regression problem has ten independent variables uniformly distributed on the interval [0, 1]. The output y is defined as

$$y = 10\sin(\pi x_1 x_2) + 20(x_3 - 0.5)^2 + 10x_4 + 5x_5 + \epsilon \quad (12)$$

where $\epsilon \sim N(0, 1)$. The data is available in the R mlbench package [69,70].

(2) *CompActiv*: This database contains records of various computer performance measures used to predict the fraction of time that central processing units (CPUs) run in user mode. We chose to model the smaller version of this data set, containing 15 of the original 24 predictors. This data set is available from the Delve project website [71].

(3) *Concrete*: The actual concrete compressive strength (MPa) for a given mixture under a specific age (days)

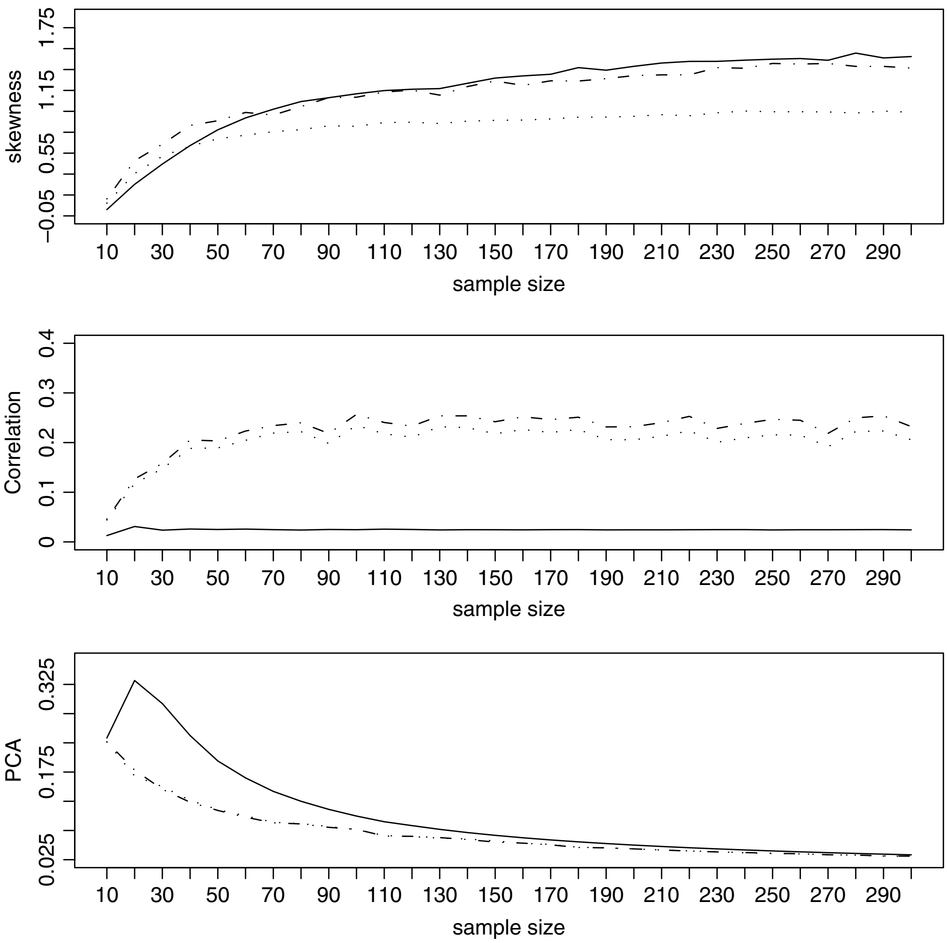

Fig. 3 Complexity for simulated data with respect to bias, $p = 10$. Complexity values for scenarios 5–7 are plotted for each measure. The line types correspond to the different simulation scenarios: dotdash, nonlinear unbiased; solid, nonlinear bias # 1; and dotted, nonlinear bias #2.

> was determined in a laboratory. The data set contains nine quantitative attributes on 1030 records, eight input attributes and one output attribute (compressive strength). These data were provided by the author of the original study [72] to the UCI Machine Learning Repository [73].

We chose an ensemble of terms $\mathcal{E}$ consisting of all single terms, two-way interaction terms, and squared terms; the cardinality of $\mathcal{E}$ is $2p + \sum_{i=1}^{p-1} i$. The specific experimental runs which we perform with each data set are described in Table 1.

5.1. Complexity Results

To examine the level of complexity of our example data sets, we calculated the measures of complexity described in Section 4 and graphed the results in Fig. 4. The Friedman data has complexity similar to that of uniform noise, i.e. high variability reflected in low skewness, low correlation, and high PCA dimension. It has larger (less complex) skewness values but higher (more complex) correlation and PCA dimension values (as in scenario nonlinear bias #2 in Section 4.2), an indication of some level of bias from the predictors not included in the model. The CompActiv data follows the complexity patterns of the nonlinear bias #2 simulation for skewness, but is closest in complexity

Table 1. Experimental runs.

Data set	N	Burn-in	Number of runs
Friedman	80	30	250
	150	30	50
	300	50	25
Concrete	200	30	20
CompActiv	70	32	25
	200	32	25
	300	32	25

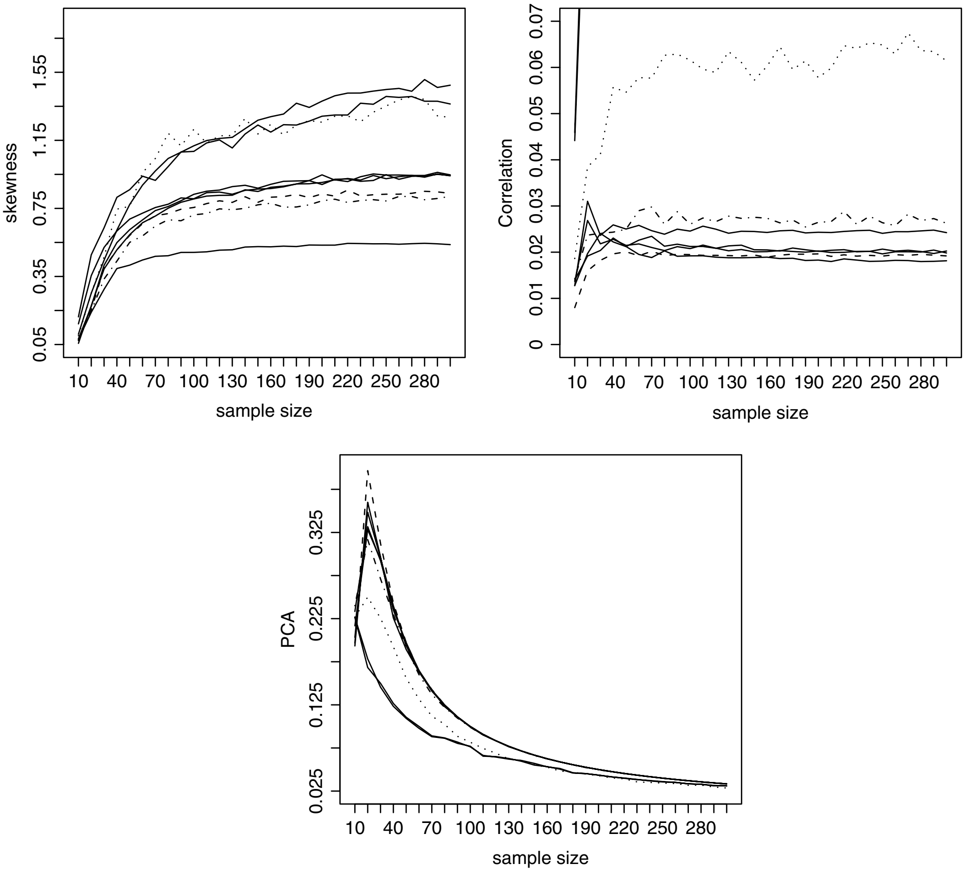

Fig. 4 Complexity results for example data. Rows from top to bottom are skewness, correlation, and PCA complexity measures. The line types vary with different example data and simulation scenarios: solid, simulated; dashed, Friedman; dotted, CompActiv; and dashdot, Concrete .

to uniform noise according to correlation and PCA dimension. This indicates a case of high variability and some bias incurred by missing important predictors. The Concrete data has more complexity by skewness than the simulated scenarios (except for uniform noise) but resembles nonlinear bias #1 in terms of correlation and PCA dimension. This indicates high variability and some bias with respect to the predictors. The resemblance to nonlinear bias (see Section 4.2) is as expected; concrete compressive strength is a highly nonlinear function of age and ingredients [72].

The complexity measures provide a framework for interpreting the complexity of these examples in terms of variability and bias. It is interesting that the complexity values of the real data examples can be interpreted through the simulation results as some mixture of high variance and moderate to high bias.

5.2. Predictive Results

We performed the experiments described in Table 1 with the burn-in length chosen to provide enough data for initial parameter estimation. For each run, a different random seed was used either for data generation (Friedman data) or to select a subset of the entire data set for analysis (CompActiv and Concrete). Our interest is primarily in the limiting behavior of the predictors. As we are not attempting function estimation or approximation, only minimal predictive error, we omit discussion of the specific model lists or model weights. We simply note that model classes like LMs or GAMs are typically biased although their predictions may outperform some classes like trees that can provide asymptotically exact approximations.

5.2.1. Results for the Friedman data

We performed computations with the Friedman data at two different choices of length of burn-in, data set size, and number of runs (see Table 1). Our results for $n = 200$ are summarized in Fig. 5; the results at other settings were qualitatively similar.

The best results are for ACAP for GAMs, with CPE at time step 300 a little over 1000, followed by stacking with GAMs with CPE at time step 300 near 1200. In aggregate, GAMs perform best followed by trees (with minimum CPE

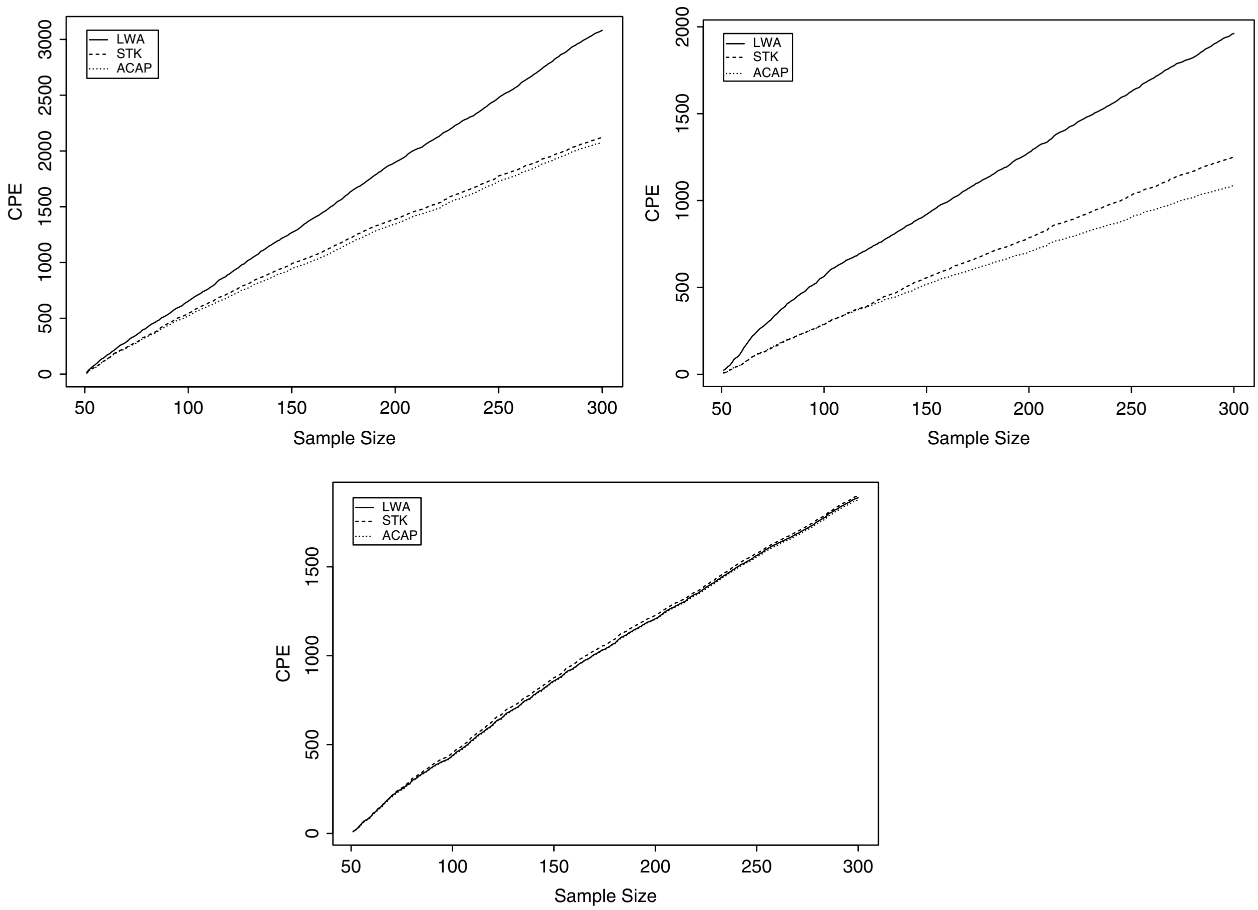

Fig. 5 CPE over time for Friedman data. Upper left panel is for LMs, upper right for GAMs, lower for trees. Note that the ranges on the vertical axes are 3000, 2000, and 1500, respectively.

with ACAP near 1900) and LMs (with minimum CPE with ACAP near 2100).

We posit that the Friedman data is of high complexity yet not nonlinear, so GAMs perform best along with a prediction strategy of high complexity, i.e. ACAPs. The next best approaches combine a high complexity model with any averaging strategy; as trees may be slightly too complex for this data, the averaging strategy has little impact. It may also be that there is not enough data for the trees to discriminate among the averaging methods. The observation that trees may be too complex is consistent with our observation in Section 5.1 that the Friedman data has the lowest relative complexity of our example data sets. Overall, LMs perform worse as they are of the lowest complexity, although their best performance is with ACAPs.

It seems reasonable to suggest the complexity of the GAMs matches the complexity of the Friedman data. The searches involved in forming the modeling averages are efforts to incorporate the complexity of the Friedman function within the complexity of the model averaging; hence ACAP yields the best results with LMs and GAMs. The trees provide adequate complexity for the data and hence all averaging strategies perform equally well.

Occasional jumps in the sequential values of the cumulative prediction error regularly occur (as in the case of trees in Fig. 5); we attribute these to the fact that the algorithm can add or omit terms discontinuously. That is, rather than having a coefficient smoothly go to zero or having a coefficient smoothly move away from zero, it is possible that at one time step a term is present, or absent, and at a later time step it is removed, or included suddenly. In either case, a jump may result, after which the error increases slowly as the parameters converge to better values.

5.2.2. *Results for the CompActiv data*

Figure 6 shows the average CPE values of all three model classes and averaging strategies for two samples sizes, $n = 70$ and $n = 300$. The results for $n = 200$ are omitted because they mirror the results for $n = 300$.

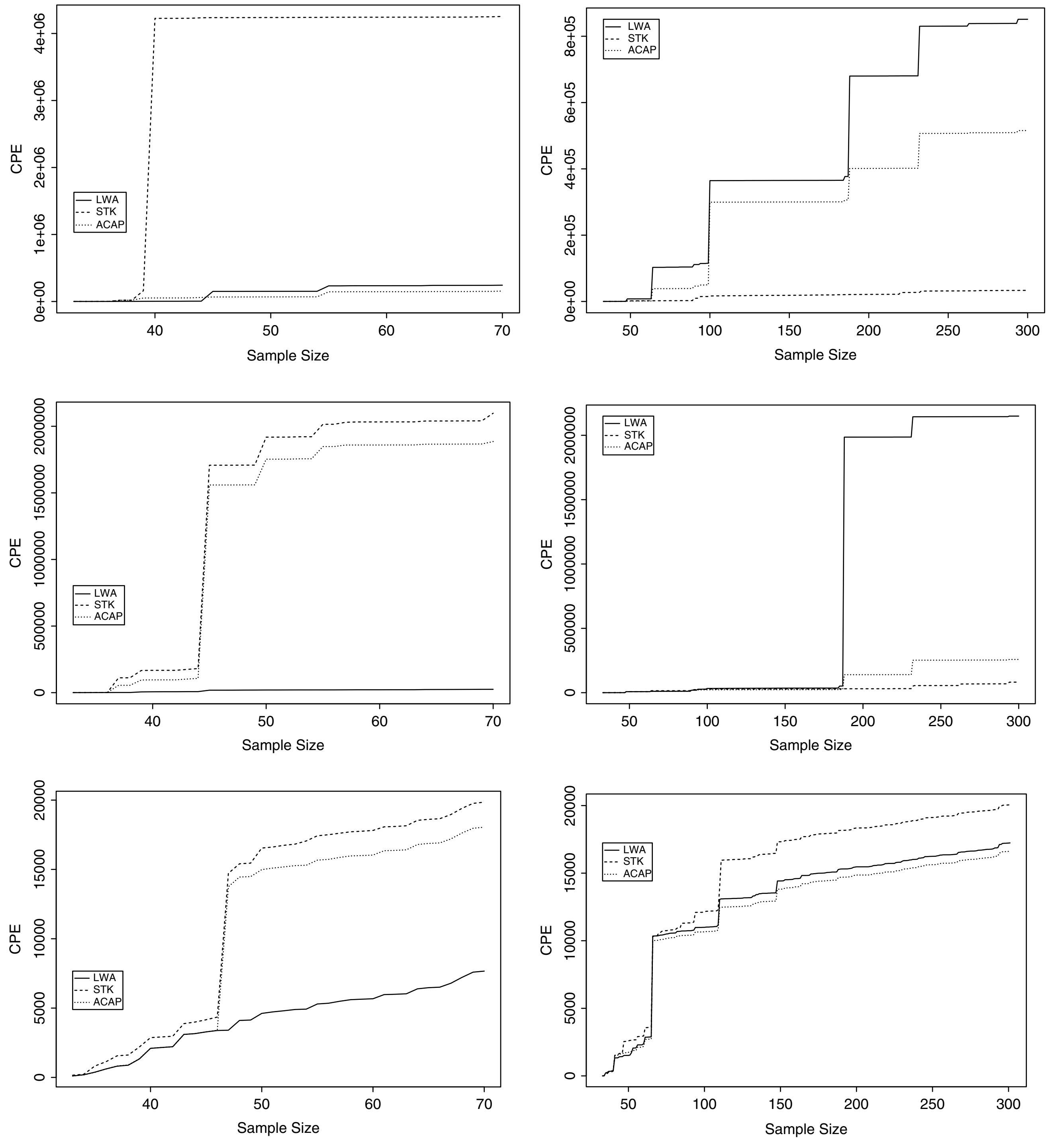

Fig. 6 CPE over time for CompActiv Data: the left-hand panels are for $n = 70$; the right-hand panels for $n = 300$. The top row is for LMs, the middle row for GAMs, and the bottom row for trees. The vertical axes are up to 800 000, 200 000, and 20 000, respectively.

The results at $n = 70$ provide a snapshot of preconvergent predictive performance. Initially, when sample size is small relative to data complexity so little information is available, all three models favor LWAs. As sample size increases, the performance of LWA deteriorates while both stacking and ACAPs show improved performance. At $n = 300$, it is seen that the best results are for trees with ACAP and LWA (with CPE near 16 000); the next best results are for trees with stacking (CPE near 20 000). All other results are considerably worse, e.g. a CPE near 33 000 for LMs with stacking, and a CPE near 82 000 for GAMs with stacking.

We have demonstrated previously that the data are highly complex. It is reasonable that the most complex model class, trees, provides all top three results when enough data is available for model estimation, and why the most complicated averaging strategy, ACAP, does best with the trees. If either the model class or model average is of a slightly lower complexity, then the performance of the approach is correspondingly degraded. For trees, the choice of α_t in the ACAP, which is near 0.7, indicates dependence on stacking. By converging more slowly, it is as if stacking refrains from giving a definite conclusion about the model until it is quite confident of its choice. The 0.3 weight on LWA is the ACAPs way of providing a smaller variance without harming the gradual reduction of bias as the stacking average finds a better and better predictor.

As a reasonable heuristic, the present calculations suggest that the complexity gap between trees and GAMs is much larger than the corresponding gap between GAMs and LMs. That is, the trees are much more complicated compared with GAMs than GAMs are to LMs. In contrast, the complexity gaps between the different model averaging strategies seem much smaller. Although ACAPs are more complex than stacking (because they involve a more extensive search over models), and stacking is more complex than LWA (because its coefficients are not restricted by the likelihood), these gaps seem roughly equal. They contribute less to the overall complexity of the method than does the choice of model class. This may be why given a model class of appropriate complexity (trees), ACAPs achieve the best balance between bias and variance. However, given a model class of clearly inadequate complexity (LMs or GAMs), none of the strategies perform well. The mismatch between the complexity of the data and the complexity (or mathematical form) of the model class (as shown in the high level of error) is a hurdle too large for any averaging strategy to overcome.

5.2.3. *Results for the Concrete data*

Recall from Section 5.1 that this data set has a highly nonlinear response, so it is no surprise that tree models perform considerably better than either GAMs or LMs (Fig. 7). In the simplest model class, stacking provides poor performance and ACAPs perform best but not significantly better than LWAs. The results with GAMs are similar but the advantage of ACAPs over LWAs increases with the more complex model class. Trees provide the best performance; in this class stacking outperforms LWA but ACAP outperforms LWA. It is interesting to note that ACAPs perform best across levels of model uncertainty; as uncertainty decreases (model/data complexity matching improves) the improvement of ACAPs over Bayes increases while the improvement of ACAPs over stacking decreases.

With respect to computational expense, the GAM models were the most expensive, followed by the trees and then by the linear models. Table 2 displays run times (in minutes) on a Windows XP laptop with a 2.3 Mhz processor and 4 GB of RAM for the Concrete data set in R with $n = 100, 200, 300$ and $p = 4, 8, 16$ (dimensions were created or removed by adding cross-terms or removing randomly selected predictors). Once LWA and stacking have been applied to a particular data set at a particular time point, ACAP is a simple averaging of the results of each method. The ability of ACAP to scale to high dimensions or sample sizes depends on the ability of the underlying model fitting procedures to scale. LM and trees appear to scale well while GAMs are more limited.

6. DISCUSSION

The modeling of data sets of high complexity requires not only an appropriate choice of model class but also a sophisticated method for model averaging. We have demonstrated that data set complexity can be measured and that these measures reflect both the variability and bias in a given statistical scenario. Real problems encountered in actual statistical practice are generally messy [74] and a rigid statistical approach can lead to overfitting and lack of validation. Through the use of examples we have shown that both a prequential approach and model averaging (ACAPs) can provide a relative advantage in such situations.

As noted in Section 1, if a prediction is desired for a new observation (at time $n + 1$), the ‘final’ model lists/weights/etc. at time n would be used to construct the prediction. In the case of simultaneous predictions at m new predictors, we would use this same procedure m times with each new predictor being treated as the observation at time $n + 1$. With respect to data that has no natural ordering, we order the entire data set for ACAPs but we must compensate for this artificial ordering by using multiple permutations of the data. This allows us to evaluate predictive performance independently of any artificial ordering.

We suggest that the settings in which LWA or stacking alone are genuinely best are important special cases, but not in general representative of the situation typically confronted by a practitioner. Oversimplifying for the sake of clarity, LWA works best when the true model is simple or the practitioner has good pre-experimental information about candidate models. Stacking works best when the true model is very complicated and no single model list can be proposed with confidence. Our method works best in the typical case that partial information is available pre-experimentally. The model list is neither so small that bias

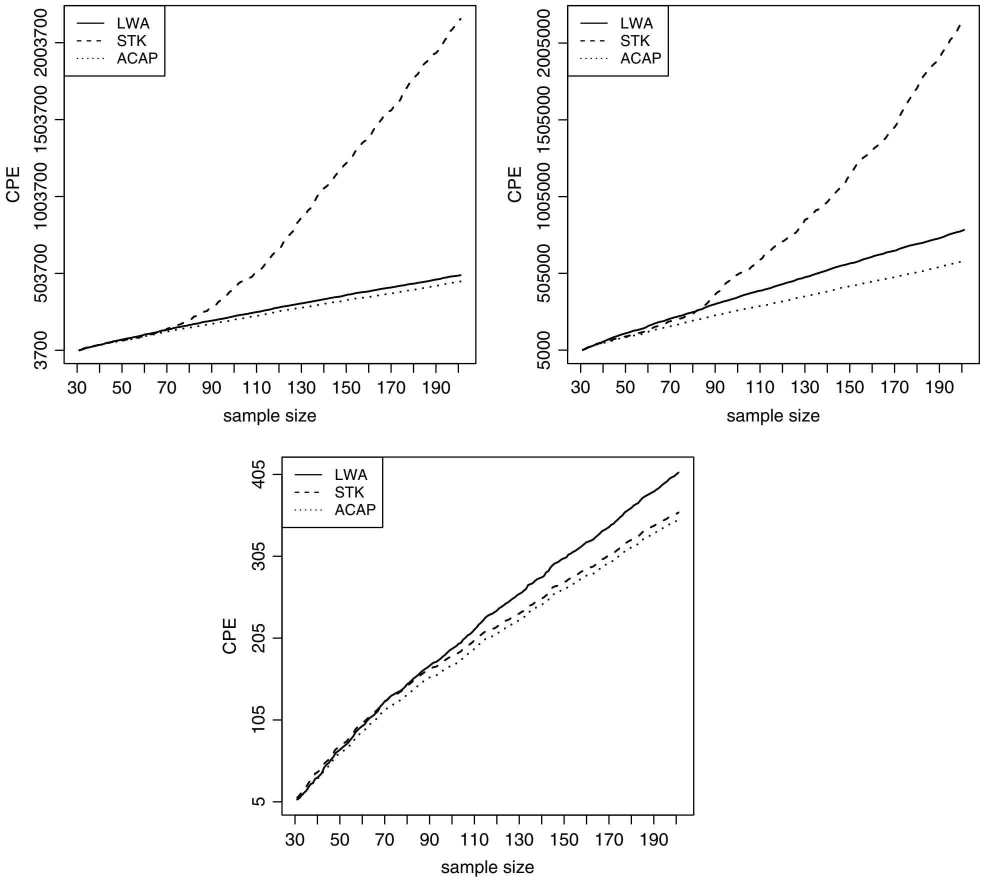

Fig. 7 CPE over time for Concrete data. Upper left panel is for LMs, upper right for GAMs, and lower for trees. Note that the ranges on the vertical axes are 2 003 700, 2 005 000, and 405, respectively.

Table 2. Run times (minutes).

	LM			GAM			Tree		
n/*p*	4	8	16	4	8	16	4	8	16
100	0.49	0.54	0.63	9.65	13.4	15.5	1.07	1.42	1.99
200	1.42	1.60	2.91	25.07	27.4	37.9	2.85	4.91	5.66
300	2.54	3.02	3.38	48.8	57.0	62.7	4.79	5.91	8.46

is likely to be a problem nor so large that model variability will swamp the information in the data. Thus, when the information available to a practitioner is enough to make a complex problem manageable, ACAPs will typically be a good choice.

ACAP may outperform LWA and stacking partially because its emphasis on using recent data makes it more adaptive. The ACAP can cycle around a small set of models that are good for prediction without settling on any of them too quickly. That is, the ACAP may more accurately encapsulate the true model uncertainty and the ability of the ensemble to approximate the true model. We comment that careful restriction of the model space to plausible models, as was achieved in our work by limiting our model list size, is important for avoiding problems with dilution [75] or, in the LWA case, giving weights that only reflect prior information. In particular, a good model list that is too large relative to the available data will tend to have too many weights of reasonable models close to zero. This can give excessive variability, and hence poor predictions, purely from the list itself.

Our results can be conceptualized as a demonstration of a principle of complexity matching. For ease of exposition, suppose that problems of prediction are segregated into classes based on their complexity (low, medium, and high). We suggest that the complexity of the modeling task, i.e. the complexity of the data generator and the sample size, should determine the complexity of the modeling strategy, i.e. the model class and model averaging. Although crude and imprecise, write

$$\mathcal{C}(\text{problem}) \sim \mathcal{C}(\text{model}) + \mathcal{C}(\text{averaging}) \qquad (13)$$

to indicate the comparison between the actual complexity of the problem on the left and the combined complexity of model class and model averaging on the right. Complexity matching is the principle that an appropriate statistical approach will satisfy this equation. Complexity affects the balance or tradeoff between bias and variance; however, complexity matching differs from the variance/bias tradeoff because the goal is not to minimize the complexity but to represent it accurately.

Our interpretations are consistent with the notions of M-Closed, M-Complete, and M-Open; see Ref. [76]. Essentially, LWA is the uniquely right answer in the M-Closed case because it achieves the (Bayes) optimality criterion in the decision problem. Once the prediction problem is not as well represented by the decision problem, so that the predictor must first find the correct decision space in which to optimize, the optimality of Bayes need not hold. Hence, in the sequential setting, one is approximating an M-Complete problem by sequentially improving an M-Closed problem. The model list reselection we have built into our approach is intended to speed the learning of the decision problem; it is our way of using the residuals to update the decision problem sequentially. Our use of stacking or ACAPs thus corresponds to an enlargement of the action space of the decision problem, probably necessary to ensure that the updated decision problem will be rich enough. More generally, ACAPs can be regarded as overcomplete predictors and this may be most appropriate for the M-Open context.

REFERENCES

[1] S. Morgenthaler and J. Tukey, Configural Polysampling: A Route to Practical Robustness. New York, John Wiley & Sons, 1991.

[2] Y. Yang, Can the strengths of aic and bic be shared? a conflict between model identification and regression estimation, Biometrika 92 (2005), 937–950.

[3] D. Draper, Assessment and propagation of model uncertainty (with discussion), J Roy Stat Soc: Series B 57 (1995), 45–70.

[4] C. Chatfield, Model uncertainty, data mining and statistical inference (with discussion), J Roy Stat Soc: Series A 158 (1996), 419–466.

[5] L. Breiman, Heuristics of instability and stabilization in model selection, Ann Stat 24 (1996), 2350–2383.

[6] J. Wichard, C. Merkwirth, and M. Ogorzalek, Building ensembles with heterogeneous models. 7th Course of the International School on Neural Nets IIASS, Salerno, Italy, 22–28 September 2002,. Available from http://citeseer.ist.psu.edu/wichard03building.html..

[7] D. Wolpert, Stacked generalization, Neural Netw 5 (1992), 241–259.

[8] J. Wichard, Model selection in an ensemble framework, Neural Networks 2006: Proceedings of the International Joint Conference on Neural Networks (IJCNN), IEEE, Piscataway, NJ, 2006, 2187–2192.

[9] L. Breiman, Stacked regression, Machine Learning 24 (1996), 49–64.

[10] J. Hoeting, D. Madigan, A. Raftery, and C. Volinsky, Bayesian model averaging: a tutorial, Stat Sci 14 (1999), 382–417.

[11] B. Clarke, Comparing bayes and stacking when model misspecification cannot be ignored, J Machine Learning Res 4 (2003), 683–712.

[12] J. Aitchison, Goodness of prediction fit, Biometrika 62 (1975), 547–554.

[13] P. Domingos, Bayesian averaging of classifiers and the overfitting problem, In Proceedings of the Seventeenth International Conference on Machine Learning, Stanford, CA: Morgan Kaufmann, 2000, 223–230.

[14] R. Duin and E. Pękalska, Object representation, sample size and dataset complexity In Data Complexity in Pattern Recognition, M. Basu and T. Ho, eds. New York. Springer-Verlag, 2006, 25–47.

[15] D. Donoho and C. Grimes, Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data, Proc Nat Acad Sci USA 100 (2003), 5591–5596.

[16] E. Levina and P. J. Bickel, Maximum likelihood estimation of intrinsic dimension In NIPS: Advances in Neural Information Processing Systems 17, L. K. Saul, Y. Weiss, and L. Bottou, eds. MIT Press, 2004, 167–189.

[17] H. Wong and B. Clarke, Improvement over bayes prediction in small samples in the presence of model uncertainty, Can J Stat 32 (2004), 269–284.

[18] A. P. Dawid, Statistical theory: The prequential approach (with discussion), J Roy Stat Soc: Series A 147 (1984), 278–292.

[19] A. P. Dawid and V. G. Vovk, Prequential probability: Principles and properties, Beroulli 5 (1997), 125–162.

[20] D. Bessler and J. Kling, Prequential analysis of cattle prices, Appl Stat 39 (1990), 95–106.

[21] D. Modha and E. Masry, Prequential and cross-validated mixture regression estimation, Machine Learning 33 (1998), 5–39.

[22] A. P. Dawid, Calibration-based empirical probability, Ann Stat 13 (1985), 1251–1273.

[23] K. Skouras and A. P. Dawid, On efficient point prediction systems, J Roy Stat Soc: Series A 60 (1998), 765–780.

[24] K. Skouras and A. P. Dawid, On efficient probability forecasting systems, Biometrika 86 (1999), 765–784.

[25] F. Seillier-Moiseiwitsch, T. J. Sweeting, and A. P. Dawid, Prequential tests of model fit, Scan J Stat 19 (1992), 45–60.

[26] F. Seillier-Moiseiwitsch and P. Dawid, On testing the validity of sequential probability forecasts, J Am Stat Assoc 88 (1993), 355–359.

[27] D. Foster, Prediction in the worst case, Ann Stat 19 (1991), 1084–1090.

[28] D. Haussler and A. Barron, How well do bayes methods work for on-line prediction of $\{+1, -1\}$ values? Proceedings of the Third NEC Symposium on Computation and Cognition, SIAM, 1992, 74–100.

[29] D. Haussler and M. Opper, Mutual information, metric entropy and cumulative relative entropy risk, Ann Stat 25 (1997), 2451–2492.

[30] Q. Xie and A. Barron, Asymptotic minimax regret for data compression, IEEE Trans Inf Theory 46 (2000), 431–445.

[31] Y. Yang, Adaptive regression by mixing, J Am Stat Assoc 96 (2001), 574–588.

[32] Z. Chen and Y. Yang, Assessing forecast accuracy measures, Department of Statistics, Iowa State University, Ames, IA, Technical Report, Preprint #04-10, 2004.

[33] T. Hastie, R. Tibshirani, and J. Friedman, The Elements of Statistical Learning: Data Mining, Inference, and Prediction. New York, Springer-Verlag, 2001.

[34] R. Berk, Limiting behavior of posterior distributions when the model is incorrect, Ann Math Stat 37 (1966), 51–58.

[35] H. Akaike, Information theory and an extension of the maximum likelihood principle, In Second International Symposium on Information Theory, B. N. Petrov and F. Csake, eds. Budapest, Akademiai Kiado, 1973, 267–281.

[36] S. Wood, Generalized Additive Models: An Introduction with R. Boca Raton, Chapman and Hall/CRC Press, 2006.

[37] J. Shao, An asymptotic theory for linear model selection, Stat Sin 7 (1997), 221–264.

[38] R. Shibata, An optimal selection of regression variables, Biometrika 68 (1979), 45–54.

[39] J. S. Rao, Bootstrap choice of cost complexity for better subset selection, Stat Sin 9 (1999), 273–287.

[40] A. Nobel, Analysis of a complexity-based pruning scheme for classification trees, IEEE Trans Inf Theory 48 (2002), 2362–2368.

[41] A. Juditsky and A. Nemirovski, Functional aggregation for nonparametric regression, Ann Stat 28 (2000), 681–712.

[42] T. Hastie and R. J. Tibshirani, Generalized additive models, Stat Sci 1 (1986), 297–318.

[43] R. A. Rigby and D. M. Stasinopoulos, Generalized additive models for location, scale and shape, Appl Stat 54 (2005), 507–554.

[44] L. Breiman, J. Friedman, R. Olshen, and C. Stone, Classification and Regression Trees. Belmont, Wadsworth Press, 1984.

[45] C. Conversano, R. Siciliano, and F. Mola, Supervised classifier combination through generalized additive multi-model, In Multiple Classifier Systems: First International Workshop, MCS 2000, Lecture Notes in Computer Science 1857, J. Kittler and F. Roli, eds. New York, Springer-Verlag, 2000, 167–189.

[46] C. de Boor, A Practical Guide to Splines. New York, Springer-Verlag, 1978.

[47] D. M. Stasinopoulos and R. Rigby, The GAMLSS Package: Generalized Additive Models for Location, Scale and Shape, Version 1.6-0, 2007.

[48] H. Zhang and B. Singer, Recursive Partitioning in the Health Sciences, Statistics for Biology and Health 12, New York, NY, Springer Verlag, 1999.

[49] W.-Y. Loh, Classification and regression tree methods, In Encyclopedia of Statistics in Quality and Reliability, F. Ruggeri, R. Kenett, and F. W. Faltin, eds. Chichester, John Wiley and Sons, 2007, 315–323.

[50] B. D. Ripley, The R tree Package: Classification and Regression Trees, Version 1.0–26, 2007.

[51] R. Holbrey, Dimension reduction algorithms for data mining and visualization, 2006,. http://www.comp.leeds.ac.uk/richardh/astro/index.html..

[52] B. Li and S. Wang, On directional regression for dimension reduction, J Am Stat Assoc 102 (2007), 997–1008.

[53] D. Warton, Penalized normal likelihood and ridge regularization of correlation and covariance matrices, J Am Stat Assoc 103 (2008), 340–349.

[54] T. Ho and M. Basu, Complexity measures of supervised classification problems, IEEE Trans Pattern Anal Mach Intell 24 (2002), 289–300.

[55] R. Baumgartner and R. Somorjai, Data complexity assessment in undersampled classification of high-dimensional biomedical data, Pattern Recognit Lett 27 (2006), 1383–1389.

[56] L. Li and Y. Abu-Mostafa, Data complexity in machine learning, Department of Computer Science, California Institute of Technology, Pasadena, CA, Technical Report Caltech CSTR:2006.004, 2006.

[57] D. Wolpert and W. Macready, Using self-dissimilarity to quantify complexity, Complexity 12 (2007), 77–85.

[58] D. Coleby and A. Duffy, Analysis of techniques to compare complex data sets, COMPEL 21 (2002), 540–553.

[59] J. Rissanen, Modeling by shortest data description, Automatica 14 (1978), 465–471.

[60] C. Wallace and D. M. Boulton, An information measure for classification, Comput J 11 (1968), 185–194.

[61] A. Barron and T. Cover, Minimum complexity density estimation, IEEE Trans Inf Theory 37 (1991), 1034–1054.

[62] J. Gower, A general coefficient of similarity and some of its properties, Biometrics 27 (1971), 623–637.

[63] P. E. Black, Minkowski distance, In Dictionary of Algorithms and Data Structures [online], P. E. Black, ed. U.S. National Institute of Standards and Technology, 31 May 2006,. Available from: http://www.nist.gov/dads/HTML/rootedtree.html. Accessed 17 July 2008.

[64] J. Hartigan and M. Wong, A k-means clustering algorithm, Appl Stat 28 (1979), 100–108.

[65] K. Fukunaga, Intrinsic dimensionality extraction, In Classification, Pattern Recognition and Reduction of Dimensionality: Volume 2 of Handbook of Statistics, P. Krishnaiah and L. L. N. Kanal, eds. Amsterdam, North Holland, 1982, 347–360.

[66] P. Grassberger and I. Procaccia, Measuring the strangeness of strange attractors, Physica D9 (1983), 189–208.

[67] V. Pestov, An axiomatic approach to intrinsic dimension of a dataset, Neural Netw 21 (2008), 204–213.

[68] X. Wang and J. S. Marron, A scale-based approach to finding effective dimensionality in manifold learning, Electron J Stat 2 (2008), 127–148.

[69] R Development Core Team, R: A Language and Environment for Statistical Computing. Vienna, Austria, R Foundation for Statistical Computing, 2007,. ISBN 3-900051-07-0: http://www.R-project.org..

[70] F. Leisch and E. Dimitriadou, The mlbench Package: Machine Learning Benchmark Problems, Version 1.1–2, 2006.

[71] Delve Development Group. Delve: Data for evaluating learning in valid experiments, 2006,. http:// www.cs. toronto. edu/∼ delve/.

[72] I.-C. Yeh, Modeling of strength of high performance concrete using artificial neural networks, Cem Concr Res 28 (1998), 1797–1808.

[73] A. Asuncion and D. Newman, UCI machine learning repository, 2007.

[74] L. Peck, R. Haugh and A. Goodman, Statistical Case Studies: A Collaboration between Academe and Industry. Philadelphia, Society for Industrial Mathematics, 1987.

[75] E. George, The variable selection problem, J Am Stat Assoc 95 (2000), 1304–1308.

[76] J. M. Bernardo and A. F. M. Smith, Bayesian Theory, Chichester England, John Wiley and Sons, 1994.