Empirical model assessment and uncertainty quantification

Philip Jonathan

1 January 2020

Abstract

The objective of this document is to provide a concise introduction to considerations for empirical assessment of predictive models. This includes an overview of existing relevant research from the statistics literature, and recommendations regarding the use of resampling methods for model assessment. The document is intended to provide context for the construction of an organisation's technical assurance and standards framework for data science.

1 Introduction

The impact of ML (Machine Learning) and AI (Artificial Intelligence) is potentially revolutionary. To successfully unlock this potential, trustworthy ML/AI systems must satisfy the same rigorous quality assurance principles and methods as a human expert or senior human decision-maker (Crawford et al. 2019, St. Clair et al. 2019), particularly for safety-critical systems (Eldevik 2018).

AI-enabled digital assets require assurance of development and deployment processes as well as product assurance of the digital asset itself. Mistakes in early deployments of AI solutions are almost inevitable, since the discipline is in its infancy; the "DEP (design and engineering practices)" guidelines for reliable AI-enabled digital deployments have not been written (Jordan 2018). For the foreseeable future, it appears that IA (Intelligence Augmentation: ML, search engines and natural language processing, machine vision) will be the main areas of development and deployment in many organisations.

Empirical models from data science and ML are, and will continue to be, key elements of ML/AI solutions. Therefore, reliable procedures for model assurance, followed carefully, will be as critical in future as they are now. All predictions from an empirical model, based on statistical and ML from data, must be made (a) to specified accuracy and precision, (b) for any operating conditions within a clearly-defined operating domain, (c) at all times within a specified period of future operation, and (d) with little or no human intervention. Rigorous technical assurance of empirical models before deployment, and while deployed, is therefore critical.

ML/AI deployments need ongoing technical assurance, post deployment. Not ad-hoc patches and updates, but a thorough ongoing "inspection and maintenance" process of actively reviewing data inputs, performance checking and re-training. This process must be specified at the concept stage before models are built, and includes (e) enhanced enterprise IT governance frameworks to include ML/AI, covering budgets, people, and data, (f) clear business use cases (who will use it, what for, what data it is going to consume, what decisions it will make?), (g) Standard Operating Processes (SOPs) for detection of ML/AI non-conformance (poor, inconsistent, dangerous predictions), including key role of "human backstop" for all ML/AI deployments, (h) chain of accountability (who monitors, who raises alarm, who performs corrective actions, who is ultimately responsible for ML/AI performance?), and (i) assessment of regulatory impact of use of ML/AI (company responsibilities under data or industry regulations, reporting requirements, alignment to industry rules).

Careful assessment of empirical predictive model performance and uncertainty quantification has a long history. There are areas of industrial activity where use of empirical models to aid decision making is well established (e.g. history matching and uncertainty quantification in reservoir engineering; design of experiments for new chemicals, fuels and lubricants; spatio-temporal modelling of the physical environment). The anticipated scope for data science solutions in a multinational organisation is huge, and many of the application domains are not traditionally associated with empirical modelling. Therefore, the potential pitfalls of empirical modelling in these areas may be less clear to data scientists and domain experts. The need to emphasise careful and ethical practice in data science has probably never been greater (Hastie et al. 2008, Lehr and Ohm 2017, Adams and Cohen 2018, RSS 2019, Mateen and Sonabend 2019).

1.1 General Principles

Recent years have seen huge improvements in the speed and scale of data connectivity and processing which underpin the current expectations for data science and AI. However, the underlying mathematics of empirical modelling has not changed. Calculations can be performed dramatically more quickly and with huge amounts of data, but the basics of good empirical modelling remain the same. Empirical models in data science and ML need above all to give accurate predictions. This means we need to build (or estimate or train) the model so that its performance on *future unseen data* is good. This performance is sometimes called the *generalisation error* or *generalisation performance*.

To estimate generalisation performance, we need to specify a *loss function* (or (negative) utility or metric) which quantifies model performance for any prediction, and also adopt a procedure (called *cross-validation*) to generate observations of "future unseen" data. Note that model assessment is only possible in a relative sense, via comparison of generalisation errors for different models (possibly of different complexity). Since a model is built using a data set, the predictive performance of the model may be sensitive to characteristics of the data set caused by one or two influential observations in the data set. These characteristics may be "chance" occurrences, not reflecting the actual structure of the underlying process that generated the data set. We need to assess how big these effects might be. A procedure called *bootstrapping* can be used to quantify prediction uncertainty for individual predictions, and for models. Moreover, it is sometimes useful to use a procedure called *randomised permutation testing* to assess the quality of a modelling approach, and compare modelling approaches across application domains.

Many "off the shelf" prediction methods used in data science assume that the observations made of a system are (in some sense) "independent" and "random". In reality, this is often not the case. When dependence of system outputs (responses) or inputs (covariates, features) is suspected, the best approach to empirical modelling involves the specification of a statistical model, making the structure and characteristics of the system explicit. The task is then to estimate the parameters of the statistical model, and demonstrate the adequacy of model performance compared to competitor models. Bayesian inference provides the natural framework for parameter estimation, allowing prior beliefs about parameters to be combined is observational data to update those beliefs.

Full model specification and Bayesian inference may not always be feasible however. Nevertheless, cross-validation, bootstrapping and randomised permutation testing provide "wrapper" *re-sampling procedures* that can be used fairly in conjunction with *any* predictive modelling procedure. In particular, these resampling techniques can be adjusted (e.g. by *blocking*) to accommodate the effects of dependence. Cross-validation, bootstrapping and randomised permutation testing therefore provide a fair data-driven basis for building, assessing and improving empirical models. They also provide a basis for *fair comparison* of different models, and the selection of preferred model(s).

1.2 Rules of thumb

In practical application, the following rules of thumb are useful starting points to ensure reasonable model building and assessment. They are default approaches to specifying the loss function, assessing generalisation error, and developing models with stable predictive performance. More details are given in the body of the report, with references to the external literature mainly Section 5.

When it is possible to write down a statistical model with appropriate distributional forms to be estimated using the data set, a full Bayesian analysis should be the preferred approach. It is recognised that this may not be feasible in many situations. When a full Bayesian analysis is not feasible, a standard prediction (e.g. classification or regression) analysis can be performed using "off the shelf" ML tools.

- Use the context of the application to decide the appropriate loss function.
  - For a *classification* model with class labels as point outputs, use classification error as loss function.
  - For a *regression* model with real-valued point outputs, use RMSE to estimate expected loss.
  - For a model with *probabilistic outputs*, use negative log-density as loss function.
- Use cross-validation to evaluate generalisation performance in terms of expected loss.
  - k-fold cross-validation with $k = 10$ is a good default approach.
  - Use cluster analysis of model inputs and outputs to identify a reasonable partition of the data set into k subsets or *blocks*, especially when dependence is presence (see time-series below).
- Use bootstrapping to quantify the uncertainty of each prediction (useful for probabilistic predictors also).
- Use bootstrapping to create a pseudo-probabilistic *ensemble* predictor from a point predictor (the model will be less influenced by individual observations in the data set used to build it, which can help build a more stable predictive model).
  - Then assess the model using negative log-density loss.
  - More sophisticated resampling procedures, involving all of blocking, bootstrapping and cross-validation have been shown empirically to be generally useful for predictive modelling with small samples, providing more stable models, estimates of generalisation performance and predictions.
- Use randomised permutation testing to estimate our confidence in the model strategy.
- Assessment of time-series and dynamic forecasting models (involving systems evolving slowly in time) requires more careful consideration than the typical situation (involving, roughly speaking, "independent occurrences" of inputs and outputs).
  - The main principle is now to use "block cross-validation" and "block bootstrapping" for dependent occurrences (in place of the equivalent procedures for independent occurrences).
  - The time-series data set should be partitioned into contiguous blocks of approximately equal length. The block length in time should be chosen so as to balance two requirements: (a) that "within-block" dependence should be much higher that "between-block" dependence, and (b) that the number of blocks is relatively large (around $k = 10$). In this sense, the blocks can then be viewed as a set of k approximately independent "super-observations" (even though the individual data points are dependent).

  - Block cross-validation and block bootstrapping are then always performed using super-observations or blocks of data (in place of individual observations).
  - Note the large specific academic literature on time-series and dynamic model forecasting, with and without covariates (e.g. West and Harrison 1997, Makridakis et al. 1998, Brockwell and Davis 2016).

- For prediction using other types of dependent data (e.g. observations over space, showing spatial dependence), "block cross-validation" and "block bootstrapping" should also be used, along similar lines to those outlined above for time-series. Blocking strategies can be extended to encompass e.g. dependence over both space and time.

1.3 Other useful resources

For those looking for a good text book, Kuhn and Johnson [2013] provides a basic introduction to applied empirical modelling (with associated algorithms for the "R" language, and numerous applications). Hastie et al. [2001] provides a model mathematical presentation on statistical learning. Organisations such as Kaggle provide thousands of data sets and illustrative case studies for predictive model development, some in the form of competitions. Assessment of generalisation performance is typically performed for point predictions. It is difficult to draw general conclusions regarding the relative merits of different approaches to predictive modelling from these competitions in general, since there are often "no clear winners". Indeed is likely that optimal model choice will be problem specific, especially given a challenging model structure or dependent data (see Section 4). Websites such as Medium (Data Science) provide many illustrative articles on predictive modelling, including elementary advice on assessment of predictive performance; these tend to focus on straightforward cases, avoiding issues such as dependence. Approaches to automated ML appear promising, provided they are used wisely. For example, the AutoML package offers opportunity for limited automatic data pre-processing (e.g. imputation, standardisation, feature selection), automatic model estimation for different model forms with hyper-parameter tuning, and estimating best overall model (typically ensemble) model performance. Environments such as MLaut (Kazakov and Kiraly [2019]) also allow fair comparison of different ML approaches.

2 Estimating generalisation performance

To estimate the generalisation performance of a model, we need to specify two quantities: (1) a loss function $c(y, \hat{y})$ which quantifies model performance (in terms of prediction $\hat{y}$ at future observation y), and (2) the probability density $p_F(y)$ of the future unseen observation y. This section gives an informal overview of how this is done; a fuller mathematical treatment is given in Section 4.

2.1 Specifying the loss function c for different prediction types

Specifying the loss function c is generally straight-forward. For example, if the model outputs a real-valued *point prediction* y^* at future observation y, a suitable loss function might be the *squared difference* $(y - y^*)^2$ or the absolute difference $|y - y^*|$. For classifier models, the most common choice of loss function is *classification error*: for each pair of actual and predicted classes y and y^*, the loss function $c(y, y^*) = 0$ if classification is correct, and $c(y, y^*) = 1$ otherwise. When there are more than two classes present, the *confusion matrix* provides a better breakdown of classification by class. It might be that some mis-classifications are more costly than others, and are to be avoided if possible; in this case, more sophisticated measures of classification error can be used as loss functions.

Sometimes the model output is a probability density p^* (rather than a single number y^*). A suitable loss function for these so-called *probabilistic predictions* is the *negative log density* $-\log(p^*(y))$ evaluated at the true value y of the future unseen observation.

For point predictions, the generalisation error is estimated as the *expected loss* using $L = \int c(y, y^*)p_F(y)dy$. For probabilistic predictions, the generalisation error is estimated using $L = \int c(y, p^*)p_F(y)dy$. The smaller the generalisation error, the better the predictor.

2.2 Cross-validation: a good way of approximating p_F

Estimating the probability density p_F of future unseen data is tricky. There are many possible approaches, the ideas underpinning some of which are outlined in Section 5, but none of these is ideal. The issue is that any *model-based* estimate for p_F is dependent on the assumptions made to estimate the model: this makes fair comparison of models impossible (unless we are prepared to accept that one model is to be preferred over all others for estimation of p_F, which defeats the object of the model comparison!). For this reason, methods which avoid directly estimating p_F are popular: the basic approach now is to split the data set used for modelling into two subsets. One subset is used for model building, and the other subset is assumed to represent a random sample from p_F so that the generalisation error can be evaluated. Exactly *how* to split the data is a very important topic: doing this naively can lead to poor models with poor generalisation performance. Later in this report we suggest an approach to clever sample-splitting which allows more reliable estimation of generalisation performance. Note in particular that it's generally a bad idea to use the whole of the data set to both build the model to also to assess it. This is because the estimated model (built using the whole data set) might "simply remember" the features of the data set (including noise features associated with individual points in the data set), without adequately characterising the dependence structure required for good prediction. Sometimes we say that the model has too much "self-influence". Such a model will therefore tend to describe observations in the data set well, but predict other future unseen data poorly. This problem is sometimes known as *over-fitting*: an over-fitting model has good *descriptive* performance of the data set used to build it, but poor *predictive* performance on future unseen data.

Cross-validation involves splitting (or partitioning) the data set into a training subset, used to build the model, and a testing subset, assumed independent of the training subset, used to assess the model. A common cross-validation technique is k-fold cross-validation, where the data set is partitioned into k subsets (see Algorithm 1). The value of k needs to be specified by the analyst; values of 5 and 10 are popular. A total of k different sub-models is then built. In turn, each of these uses all the data set *except* the kth subset for building, and then uses the kth subset only for model assessment. That is, for each value of k, one of the k subsets is used as the test subset and the other $k-1$ subsets form the training subset. The generalisation error of the full model is estimated by summing the generalisation errors over all k sub-models. In cross-validation, every observation in the data set is used exactly once in a test set, and exactly $k-1$ times in training sets for sub-models, so that all observations are used in a "balanced" way. Re-using the data like this attempts to always use as much of the data set as reasonably possible for model building, whilst ensuring that a wide range of test observations are available, which are "effectively independent" of the training sets, helping to provide good estimation of generalisation error. Deciding on a reasonable strategy to partition the data set into k subsets is critical, particularly when the data set contains dependent observations (which is almost always the case, to some extent at least). *Blocking* by input and output using *cluster analysis* is one possibility. *Blocking* in time is another, especially when system inputs and outputs vary slowly in time.

For point predictions, and data set $\{y_i\}_{i=1}^n$, we have $c(y_i, y_i^*) = (y_i - y_i^*)^2$ for the squared difference loss function. Here, y_i is the ith observation in the data set, and y_i^* is the predicted value of y_i evaluated in the cross-validation procedure (using a training subset which *does not* contain y_i). Also, $p_F(y) = 1/n$ in the cross-validation procedure, since each observation is assumed to occur with equal probability in future. Hence, the expected loss L can be written in the form $\sum_{i=1}^n (y_i - y_i^*)^2 \times (1/n)$, which is simply the (cross-validated) mean square error of prediction. In practice, the square-root of this quantity (RMSE, root mean square error) is usually quoted since it has the same scale as the data y.

Cross-validation became popular when the added computational complexity of estimating k sub-models was no longer a burden to the analyst. Before cross-validation, the conventional approach to model building and assessment was to partition the data into two parts, using an approach sometimes today referred to as hold-out. The data set is simply partitioned into a model building subset and a test subset. The model building subset is used to estimate the model (sometimes involving further partitions, e.g. for estimation of tuning parameters), and the testing subset used to estimate generalisation error. Again, the way in which the data partition is made can affect the quality of estimation of the model, and of its generalisation error. Compared with k-fold cross-validation, hold-out involves estimating just one extra sub-model as opposed to k, and may therefore be a simple, useful "poor person's" approach when nothing better is possible. A technique such as maximum dissimilarity could be used to select a training subset representative of the full data set in this case. Multiple different hold-out analyses might be combined. However, in general, cross-validation should be preferred.

The density p_F (for future unseen data) should be defined carefully. When the original data set is considered representative of future unseen observations, the k-fold cross-validation procedure defined above is appropriate. However, when the original data is considered not to be representative of *future* observations, the training set used should best reflect our beliefs about the nature of p_F.

Cross-validation can also be helpful in estimating the values of model parameters within the model building (as opposed to the model assessment) phase (see Algorithm 2). It is important not to confuse the uses of cross-validation for model assessment and for model building. When cross-validation is needed for model building, it can also be used for model assessment, by adopting a *nested* cross-validation strategy, in which an *inner* cross-validation loop is used as part of model building to estimate model paramaters, and an independent *outer* cross-validation loop is used to assess generalisation performance for the estimated model.

An illustration of the application of cross-validation for quantification of generalisation performance is given in Section 7.

```
  Input: Data set $D = (X, Y)$ of $n$ observations, model form $M$
1 Partition $D$ into $k$ groups at random (or using cluster analysis of $X$ and $Y$) referenced by index set $\mathcal{J}$ with elements
    $j \in (1, 2, ..., k)$. Label the groups $D_1, D_2, ..., D_k$ (with analogous notation for partitions of $X$ and $Y$);
2 for $j = 1, 2, ..., k$ do
3     Fit the model to the data set $(X_{\mathcal{J} \neq j}, Y_{\mathcal{J} \neq j})$ to obtain estimated model $M^{(j)}$;
4     Generate a prediction $Y^*_{\mathcal{J}=j}$ from $M^{(j)}(X_{\mathcal{J}=j})$;
5 end
  Output: Estimated generalisation error $L = (1/n) \sum_j c(Y_{\mathcal{J}=j}, Y^*_{\mathcal{J}=j})$
```

Algorithm 1: Pseudo-code for estimation of generalisation error using k-fold cross-validation.

3 Estimating uncertainty for individual predictions and models

3.1 Bootstrapping

Bootstrapping is a data set resampling procedure. A bootstrap resample $\{y_i^b\}_{i=1}^n$ is a "mimic" of the original data set $\{y_i\}_{i=1}^n$, in which each bootstrap observation y_i^b is just a random draw (with replacement) of one observation from $\{y_i\}_{i=1}^n$. This means that the bootstrap resample may contain some of the original observations multiple times, and others not at all. Moreover,

```
Input: Data set $D = (X, Y)$ of $n$ observations, model form $M_\lambda$ for tuning parameter $\lambda$.
1  Partition $D$ into $k$ groups at random (or using cluster analysis of $X$ and $Y$) referenced by index set $\mathcal{J}$ with elements
   $j \in (1, 2, ..., k)$. Label the groups $D_1, D_2, ..., D_k$ (with analogous notation for partitions of $X$ and $Y$);
2  Define a set $\{\lambda_\ell\}_{\ell=1}^{n_\ell}$ of possible values for $\lambda$ to consider;
3  for $\ell = 1, 2, ..., n_\ell$ do
4      for $j = 1, 2, ..., k$ do
5          Fit the model to the data set $(X_{\mathcal{J}\neq j}, Y_{\mathcal{J}\neq j})$ and tuning parameter $\lambda_\ell$ to obtain estimated model $M^{(j,\ell)}$;
6          Generate a prediction $Y^*_{\mathcal{J}=j,\ell}$ from $M^{(j,\ell)}(X_{\mathcal{J}=j})$;
7      end
8      Estimate generalisation error $L(\lambda_\ell) = (1/n)\sum_j c(Y_\mathcal{J}, Y^*_{\mathcal{J}=j,\ell})$;
9  end
10 Select value of $\lambda^*$ of $\lambda_\ell$ which minimises generalisation error $L(\lambda_\ell)$;
Output: Estimated optimal tuning parameter $\lambda^*$
```

Algorithm 2: Pseudo-code for estimation of optimal tuning parameter using k-fold cross-validation.

the bootstrap resample only contains observations from the original data set. Bootstrap analysis (for any modelling task) then involves repeating that task for each of a large number of bootstrap resamples, as well as on the original data set.

Bootstrap resampling provides a quantification of prediction uncertainty for point prediction. To achieve this at prediction point y, we simply estimate y^{b*} for m bootstrap resamples (with $m \geq 250$), and use the m values of y^{b*} to estimate an uncertainty interval for the prediction. The same procedure can be used for probabilistic predictions of course.

Moreover, bootstrapping can convert a model which outputs point predictions into one which outputs probabilistic predictions. To achieve this at prediction point y, we simply estimate y^{b*} for m bootstrap resamples, and use the m values of y^{b*} to construct an empirical density p^* for prediction at y. We can then use the negative log-density loss function to assess model performance. This can help build a more stable predictive model, in the sense that it is less influenced by individual observations in the data set used to build it.

Bootstrapping is appealing in that it can provide a rational quantification of uncertainty for any inference. Bayesian inference provides the same capability, but in general Bayesian inference is a more complex task than bootstrap re-sampling. However, most would argue that Bayesian inference provides a more consistent inference, and should therefore be preferred when feasible.

```
Input: Data set $D = (X, Y)$ of $n$ observations, test input $\tilde{X}$, model specification $M$
1  for $j = 1, 2, ..., m$ do
2      Generate bootstrap index set $\mathcal{J}_j$ of size $n$ by sampling randomly with replacement from index set $(1, 2, ..., n)$;
3      Fit the model to the data set $(X_{\mathcal{J}_j}, Y_{\mathcal{J}_j})$ to obtain estimated model $M^{(j)}$;
4      Generate a prediction $\tilde{Y}^*_j$ from $M^{(j)}(\tilde{X})$;
5  end
Output: Predictions $(\tilde{Y}^*_1, \tilde{Y}^*_2, ..., \tilde{Y}^*_m)$ at test input $\tilde{X}$
```

Algorithm 3: Pseudo-code for generating bootstrap re-sampled predictions from a model M.

Algorithm 3 may be embedded within Algorithm 1 to provide estimation of generalisation error using pseudo-probabilitistic bootstrap predictions. An illustration of the application of bootstrapping for quantification of predictive uncertainty is given in Section 7.

3.2 Randomised permutation testing

A randomised permutation test is a type of statistical significance test. We use the randomised permutation test to estimate how unusual the value of generalisation error (using the original data set) is, compared to generalisation errors generated using data sets in which the values of the responses y (in the original data set) have been randomly permuted between observations. We would expect the generalisation error for models based on randomised response data to be high because there should be no basis for good prediction in general; we might expect that the actual generalisation error is relatively small. We can quantify this effect by estimating a *p-value*, namely the probability that a model based on a randomised permutation of the original responses would generate lower generalisation error than the model based on the actual data set. If the p-value is greater than 0.05 say, we might be suspicious of the quality of the model or the method of model assessment, since they can relatively frequently generate low generalisation errors for randomised data, potentially suggesting over-fitting.

More formally, a permutation test estimates the distribution of any test statistic under the null hypothesis by calculating all possible values of the test statistic under rearrangements of the labels (classes, responses) on the observed data points. If the labels are exchangeable under the null hypothesis, then the resulting tests yield exact significance levels.

To illustrate this further, consider a data set containing $n = 50$ observations on $p = 10^6$ covariates, i.e. many more inputs than observations. . In this situation, because of the "ill-posed" nature of the data set, it is possible just by chance that good predictive models for randomised responses might be found, which may not be based on any physical relationship between covariates and response. The distribution of generalisation error using randomly permuted responses allows us to quantify how likely this, and to compare this to generalisation error estimated using the actual original data set. In a similar fashion, if our model assessment strategy were in some sense biased, the randomised permutation test would indicate this, since the mean value of generalisation error under randomised data would be lower than typically expected.

Like bootstrap re-sampling, randomised permutation testing can be used quite generally to diagnose statistics (such as generalisation error) in empirical modelling. An illustration of the application of randomised permutation testing is given in Section 7.

```
  Input: Data set $D = (X, Y)$ of $n$ observations, model specification $M$
1 Estimate the generalisation performance $L$ of model $M$ based on original data set $(X, Y)$;
2 for $j = 1, 2, ..., m$ do
3 |   Generate random permutation index set $\mathcal{J}_j$ from the index set $(1, 2, ..., n)$;
4 |   Estimate generalisation performance $L_j$ of model $M^{(j)}$ based on randomised-response data set $(X, Y_{\mathcal{J}_j})$;
5 end
  Output: Generalisation performance $L$ for original dat set, empirical distribution of generalisation performance from
          randomised responses, and p-value.
```

Algorithm 4: Pseudo-code for generating empirical density of generalisation performance from randomised permutations of responses, from model M.

4 Prediction with dependent data

4.1 "Classic" time-series forecasting (with no covariates)

There is a very large literature on time-series modelling, and also of time-series competitions where different authors compete to produce algorithms with the best predictive performance for time-dependent data. This literature tends to be restricted to situations where no covariates are available (i.e. the "classic" time-series forecasting problem). A possible statistical model might be represented by the ARIMA equation

$$\left(1 - \sum_{i=1}^{p} \alpha_i \Delta^i\right) (1 - \Delta)^d Y_t = \delta + \left(1 + \sum_{j=1}^{q} \theta_j \Delta^j\right) \epsilon_t$$

where Δ is the lag operator, $\{\alpha_i\}$ are auto-regressive parameters, $\{\theta_j\}$ are moving-average parameters, and $\{\epsilon_t\}$ are error random variables with independent standard Gaussian distributions. δ is the drift term. The integers p, q and d define the order of the model. This family of models includes autoregressive (AR), moving average (MA), autoregressive-moving average (ARMA) as well as autoregressive-integrated-moving average (ARIMA). Further extensions are possible, e.g. for the including of covariates (ARIMAX) or fractional lag operators (ARFIMA).

Makridakis et al. [2018a] compares some statistical time-series models with those based on ML for more than 1000 time-series data sets in the so-called M3 Competition. "Machine learning" algorithms including Multi-Layer Perceptron (MLP), Bayesian Neural Network (BNN), Radial Basis Functions (RBF), Generalized Regression Neural Networks (GRNN, also called kernel regression), k-Nearest Neighbour regression (KNN), CART regression trees (CART), Support Vector Regression (SVR), and Gaussian Processes (GP). In general, traditional time-series methods yield better predictive performance, and are computationally less demanding. The authors stress the need for careful methods for assessment of empirical time-series forecasting methods. Makridakis et al. [2018b, 2020] extend the comparisons to the order of 10^5 time-series and 60 forecasting methods, and note that (a) algorithms that combine different approaches in general do better than algorithms which rely on one approach only (c.f. ensemble models, and Bayesian model averaging), (b) algorithms which merge statistical and ML approaches appear to do better than those using either statistical or ML approaches, (c) algorithms based on ML alone perform particularly poorly.

4.2 Time-series prediction and forecasting (with covariates)

In many cases of practical importance, the data set for analysis consists of dependent draws of covariates, and conditionally of responses. There are many forms for a possible data-generating process; a simple example might be

$$\begin{aligned}
\text{Observation equation} \quad &: \quad y_t = g(x_t, \boldsymbol{\beta}) + \epsilon_t \\
\text{Observation error evolution} \quad &: \quad \epsilon_t = \alpha_\epsilon \epsilon_{t-1} + \nu_{\epsilon,t}, \quad \nu_{\epsilon,t} \sim N(0, \sigma_\epsilon^2) \\
\text{System function} \quad &: \quad g(x, \boldsymbol{\beta}) = \sum_{k=0}^{p} x^k \beta_k \\
\text{Covariate evolution} \quad &: \quad x_t = \alpha_x x_{t-1} + \nu_{x,t}, \quad \nu_{x,t} \sim N(0, \sigma_x^2) \\
\text{Conditional density} \quad &: \quad f(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{\beta}, \alpha_\epsilon, \alpha_x)
\end{aligned}$$

In this illustration, both the covariate x and the observation error ϵ evolve as autoregressive AR(1) processes. The system function is a polynomial in the covariate, observed with error. Clearly, in this case, the wisest approach to estimation is to exploit this model form within a Bayesian inference. However, when the model form isn't known, it is still in general possible to use more generic "off the shelf" statistical and ML prediction tools to generate good predictive models, provided that we treat the time-series dependence present appropriately. Rigorous attempts to compare different data science or ML approaches to time-series prediction with covariates are not well reported in the literature.

There is empirical and theoretical evidence that block bootstrap cross-validation provides a reasonable basis for model selection; Hall et al. [1995] provides a good starting point for relevant literature. A possible implementation is given in Algorithm 5.

```
Input: Data set $D = (X, Y)$ of $n$ observations, model form $M$. Specify number of cross-validation groups $k$ and
       number of bootstrap resamples $n_m$
1 Partition $D$ into $k$ time-contiguous groups referenced by index set $\mathcal{J}$ with elements $j \in (1, 2, ..., k)$. Label the groups
    $D_1, D_2, ..., D_k$ (with analogous notation for partitions of $X$ and $Y$);
2 for $j = 1, 2, ..., k$ do
3     Isolate the training subset the subset $(X_{\mathcal{J}\neq j}, Y_{\mathcal{J}\neq j})$;
4     for $m = 1, 2, ..., n_m$ do
5         Block bootstrap resample the training subset $(X_{\mathcal{J}\neq j}, Y_{\mathcal{J}\neq j})$, always keeping time-contiguous blocks intact,
            obtaining set $(X^{(m)}_{\mathcal{J}\neq j}, Y^{(m)}_{\mathcal{J}\neq j})$;
6         Fit the model to the data set $(X^{(m)}_{\mathcal{J}\neq j}, Y^{(m)}_{\mathcal{J}\neq j})$ to obtain estimated model $M^{(j,m)}$;
7         Generate a prediction $Y^{*(m)}_{\mathcal{J}=j}$ from $M^{(j,m)}(X_{\mathcal{J}=j})$;
8     end
9 end
Output: Estimated generalisation error $c = \sum_j \sum_m c(Y_{\mathcal{J}=j}, Y^{*(m)}_{\mathcal{J}=j})$.
```

Algorithm 5: Pseudo-code for block bootstrap cross-validated assessment of predictive performance for dependent data.

5 Mathematical background

In this section we provide some more detail on underpinning arguments supporting the recommended methods for model assessment. The guiding principle for model assessment is validation and/or falsification by *actual data*, using a common quantitative procedure to assess predictive performance. The assessment procedure must be *model agnostic* (and independent of specifics of any of the models or modelling frameworks considered). Assuming a *Bayesian perspective* is useful and natural for discussion of model assessment for all models and modelling frameworks.

We extend the notation somewhat to provide more precise discussion. Consider a prediction problem with covariate(s) x and response y. The available sample of n observations is denoted by $D = \{(x_i, y_i)\}_{i=1}^n$, and future observations by $(\tilde{x}, \tilde{y})$. We want to make inferences about observable quantities like $\tilde{x}$ and $\tilde{y}$.

We estimate expected predictive performance given three key quantities. These are (1) a specific loss function (c, to compare model predictions with actual values), (2) a model p for future observations, and (3) a specific prediction scenario (e.g. are covariates random?). We then compare a set of models against each other according to the values of their expected loss, and may choose a preferred model as that with lowest expected loss.

Expected loss and generalisation performance

In principle, the expected predictive performance of a model M is evaluated as $L = \int c(\tilde{y}, a_M(\tilde{x}))p(\tilde{x}, \tilde{y})d\tilde{x}d\tilde{y}$ where $p(\tilde{x}, \tilde{y})$ is the joint distribution of *future* observations, a_M is the optimal action from model M corresponding to future design point $\tilde{x}$ with response $\tilde{y}$, and c is the loss function. In practice, (1) we don't know $a_M(\tilde{x})$ and must estimate it using D, and (2) we also don't know $p(\tilde{x}, \tilde{y})$ and must estimate it using D. Further, it is usual to estimate loss conditional on a value of $\tilde{x}$. In this case, noting that $L = \int c(\tilde{y}, a_M(\tilde{x}))p(\tilde{y}|\tilde{x})p(\tilde{x})d\tilde{y}d\tilde{x}$, the expected conditional loss becomes $L(\tilde{x}) = \int c(\tilde{y}, a_M(\tilde{x}))p(\tilde{y}|\tilde{x})d\tilde{y}$.

The choice of loss function is arbitrary, but as highlighted in Section 1 there are essentially two important types, corresponding to so-called point and probabilistic prediction. For point prediction, $a_M(\tilde{x})$ is a number (of the same dimension as $\tilde{y}$), and a least squares loss function is typical (such that $c(\tilde{y}, a_M(\tilde{x})) = (\tilde{y} - a_M(\tilde{x}))^2$).

For probabilistic prediction, $a_M(\bullet|\tilde{x})$ is a probability density, and the negative log-action loss function $c(\tilde{y}, a_M|\tilde{x}) = -\log(a_M(\tilde{y}|\tilde{x}))$ is typical; with this loss function, the optimal a_M is given by $p(\tilde{y}|\tilde{x}, D, M)$ under model M and data D; the loss function is therefore referred to as the negative log-density loss. To estimate expected conditional loss $L(\tilde{x}) = \int c(\tilde{y}, a_M(\tilde{x}))p(\tilde{y}|\tilde{x})d\tilde{y}$ and probabilistic prediction, we first estimate $a_M(\tilde{y}|\tilde{x}) = p(\tilde{y}|\tilde{x}, D, M)$ using a Bayesian inference (noting that other approaches are possible). To achieve this, we need to specify (1) a model $p(y|x, \theta, M)$ for observations given conditional model M with parameters θ, and (2) a prior distribution $p(\theta|M)$. Then we estimate the posterior distribution of parameters $p(\theta|D, M)$, and use it to estimate $p(\tilde{y}|\tilde{x}, D, M)$ as $\int p(\tilde{y}|\tilde{x}, \theta, M)p(\theta|D, M)d\theta$.

We also don't know $p(\tilde{y}|\tilde{x})$, the actual conditional data generating distribution. So we also have to try to estimate it. There are various options, which are discussed next.

$\mathcal{M}$-closed, $\mathcal{M}$-complete and $\mathcal{M}$-open views

One approach is to estimate $p(\tilde{y}|\tilde{x})$ as $p(\tilde{y}|\tilde{x}, D, M^\dagger)$ using Bayesian inference, where model $M^\dagger$ is a reference or "best possible" model (and $M^\dagger$ is different to M in general). How to find $M^\dagger$ is an important question! One approach is to specify a set of candidate models $\{M_\ell\}_{\ell=1}^{n_M}$, and estimate $M^\dagger$ as a Bayesian Model Average over the n_M models. This is termed a "$\mathcal{M}$-closed" approach in the literature, since only the n_M models can contribute to specification of $M^\dagger$. The approach requires prior specification of probabilities for each of the n_M models. The actual estimate is then $p_{BMA}(\tilde{y}|\tilde{x}, D) = \sum_{\ell=1}^{n_M} p(\tilde{y}|\tilde{x}, D, M_\ell)p(M_\ell|D)$, where predictive distributions $p(\tilde{y}|\tilde{x}, D, M_\ell)$ and $p(M_\ell|D)$ are themselves estimated using Bayesian inference, as $p(\tilde{y}|\tilde{x}, D, M_\ell) = \int p(\tilde{y}|\tilde{x}, \theta, M_\ell)p(\theta|D, M_\ell)d\theta$, and $p(M_\ell|D) = \prod_i p(y_i|x_i, M_\ell)p(M_\ell)/\prod_i p(y_i|x_i)$, where $p(y_i|x_i, M_\ell) = \int p(y_i|x_i, \theta, M_\ell)p(\theta|M_\ell)d\theta$. The prior might be informed by relative physical plausibility of different model forms: models which are identifiable and interpretable might be rewarded with large prior probabilities. Typically however, we might a-priori set all models equally probable.

If we are confident that at least some of the models $\{M_\ell\}_{\ell=1}^{n_M}$ are performing well, then the $\mathcal{M}$-closed approach might be reasonable. When we are not confident that any of the n_M models is reasonable, applying Bayesian model averaging over the models is not reasonable.

Sometimes it might be possible to define the space of all possible models in terms of a basis representation (c.f. any continuous periodic function with bounded variation can be represented using Fourier series). In this situation, for an infinite set of possible models represented by some basis representation, the modelling approach might be described as *$\mathcal{M}$-complete.*

Another approach, recommended for general application, particularly when models are not performing well, is to refuse to try to estimate $p(\tilde{x}, \tilde{y})$ or $p(\tilde{y}|\tilde{x})$ at all, and instead use cross-validation or similar to simulate "random" samples from $p(\tilde{x}, \tilde{y})$. This approach is termed a "$\mathcal{M}$-open" approach in the literature, since no attempt is made to specify a model for $p(\tilde{x}, \tilde{y})$ or $p(\tilde{y}|\tilde{x})$. For convenience, we write the original data set D in terms of the set of observations X of covariates and Y of responses. Further, we partition the data set D into k blocks $\{D_{\mathcal{J}=j}\}_{j=1}^k$ using index set $\mathcal{J}$, and similarly partition X and Y. Using the notation $D_{\mathcal{J}\neq j}$ to refer to the original sample with the j^{th} block removed, the expected loss corresponding to a model M is evaluated (for the case of probabilistic prediction) using $L(M|\{D_{\mathcal{J}=j}\}) = \sum_{j=1}^k -\log(p(Y_{\mathcal{J}=j}|X_{\mathcal{J}=j}, D_{\mathcal{J}\neq j}, M)) = -\log(\prod_{j=1}^k p(Y_{\mathcal{J}=j}|X_{\mathcal{J}=j}, D_{\mathcal{J}\neq j}, M))$. The argument $\prod_{j=1}^k p(Y_{\mathcal{J}=j}|X_{\mathcal{J}=j}, D_{\mathcal{J}\neq j}, M)$ of the last expression can be interpreted as a "predictive likelihood" for the partitioned sample under model M.

Model selection

As long as model diagnostics and generalisation error are acceptable, there is no need for model selection in general. Use of an ensemble of models rather than a single model is fine if the ensemble gives good performance. There is further no need to perform procedures such as variable selection in general if a larger model gives good performance. The only role for model selection is to reduce the complexity of data gathering and computation required for inference. Having said this, parsimonious models are often more stable with respect to generalisation error especially when $p(\tilde{x}, \tilde{y})$ is poorly estimated (Hand and Vinciotti 2003, Hand 2006, 2012), and might therefore be rationally preferred. If desired, we can evaluate "posterior" densities for each of a number of different models $\{M_\ell\}_{\ell=1}^{n_M}$, at least some of which are performing reasonably, using the $\mathcal{M}$-open framework. We write $p(M_\ell|\{D_{\mathcal{J}=j}\}) = \prod_{j=1}^k p(Y_{\mathcal{J}=j}|X_{\mathcal{J}=j}, D_{\mathcal{J}\neq j}, M_\ell)p(M_\ell)/\prod_{j=1}^k p(Y_{\mathcal{J}=j}|X_{\mathcal{J}=j}, D_{\mathcal{J}\neq j})$. Here, terms $p(Y_{\mathcal{J}=j}|X_{\mathcal{J}=j}, D_{\mathcal{J}\neq j})$ in the demoninator might be evaluated as $\sum_\ell p(Y_{\mathcal{J}=j}|X_{\mathcal{J}=j}, D_{\mathcal{J}\neq j}, M_\ell)p(M_\ell|D_{\mathcal{J}\neq j})$ (and the terms in the last expression are themselves evaluated as explained under $\mathcal{M}$-closed above). Therefore, in the case that priors $p(M_\ell)$ are all set equal, comparison of models in terms of expected loss $L(M_\ell|\{D_{\mathcal{J}=j}\})$ is equivalent to model comparison using logarithms of posterior probabilities $p(M_\ell|\{D_{\mathcal{J}=j}\})$ of the models, evaluated using the "predictive likelihood" under the k-fold cross-validation scheme.

How best to partition D for k-fold cross-validation?

In general, we wish to estimate expected predictive performance over the true joint distribution of $\tilde{x}$ and $\tilde{y}$ (rather than conditional on a particular value of $\tilde{x}$). That is, we wish to evaluate $\int c(\tilde{y}, a_M(\tilde{x}))p(\tilde{x}, \tilde{y})d\tilde{x}d\tilde{y}$. To achieve this requires a joint model $p(\tilde{x}, \tilde{y})$ for future observations (and recall that there are typically a large number of covariates $\tilde{x}$, not just one). In this situation, it is typical to adopt the $\mathcal{M}$-open view, and use an assessment procedure based on cross-validation.

In simple terms, the partition of sample D into $\{D_{\mathcal{J}=j}\}_{j=1}^k$ should be performed in such a way that the dependence between each hold-out block $X_{\mathcal{J}=j}$ and the remainder $X_{\mathcal{J}\neq j}$ of the original sample of covariate values is representative of the dependence between any new prediction point(s) $\tilde{x}$ and the full sample X of covariates. There are two approaches which might be useful.

In the absence of further knowledge, it is best to partition such that each block $D_{\mathcal{J}=j}$ contains similar values of covariates. By construction, therefore, each covariate block $X_{\mathcal{J}=j}$ will be relatively dissimilar to (or show less dependence with) the remainder $X_{\mathcal{J}\neq j}$; we might achieve this partitioning using a (e.g. k-means) cluster analysis. Such covariate-based partitioning is useful when we expect the prediction point(s) $\tilde{x}$ to be quite different to the original set of values of covariates in X.

In the (probably less common) case that the prediction point(s) $\tilde{x}$ are expected to be similar to the original set of values X of covariates, a random partitioning of D into $\{D_{\mathcal{J}=j}\}_{j=1}^k$ might be more appropriate. If the original sample X of covariates describes the design space for prediction well and relatively *densely* (in the sense that observations in X are as close to each other as they are to any prediction point $\tilde{x}$), random partitioning is appropriate.

Properties of loss functions

The optimal action $a(\tilde{x})$ using squared difference loss $(\tilde{y} - a(\tilde{x}))^2$ for point prediction is the posterior predictive mean $E[\tilde{y}|\tilde{x}]$. Informally, this can be seen from differentiation the expression $L(\tilde{x}) = \int(\tilde{y} - a(\tilde{x}))^2 p(\tilde{y}|\tilde{x})d\tilde{y}$ with respect to a and setting the answer to zero: the optimal action $a^*(\tilde{x})$ is then $\int \tilde{y}p(\tilde{y}|\tilde{x})d\tilde{y} = E[\tilde{y}|\tilde{x}]$. When we use $p(\tilde{y}|\tilde{x}, D, M)$ to estimate $p(\tilde{y}|\tilde{x})$ using data set D and model M, the optimal action $a^*_M(\tilde{x})$ under model M and data D is $E[\tilde{y}|\tilde{x}, D, M]$.

For probabilistic prediction, $a(\bullet|\tilde{x})$ is a probability density, and the negative log-action loss function $c(\tilde{y}, a(\tilde{x})) = -\log(a(\tilde{y}|\tilde{x}))$ is typical. Negative log-action loss can be shown to be closely related to fundamental quantities in statistical learning, including entropy and the Kullback-Leibler divergence. With this loss function, the optimal action $a^*(\tilde{y}|\tilde{x})$ is given by $p(\tilde{y}|\tilde{x})$. This clear since $\int \log(q(y))p(y)dy \leq \int \log(p(y))p(y)dy$ for probability densities p and q; the loss function is therefore referred to as the negative log-density loss. When we use $p(\tilde{y}|\tilde{x}, D, M)$ to estimate $p(\tilde{y}|\tilde{x})$ using data set D and model M, the optimal action under model M and data D is $p(\tilde{y}|\tilde{x}, D, M)$.

Using bootstrapping if necessary, any predictive model can be considered to be probabilistic. Its (conditional) generalisation error is therefore given by $L(\tilde{x}) = -\int \log(p^*(\tilde{y}|\tilde{x}))p(\tilde{y}|\tilde{x})d\tilde{y}$. The best possible predictive model would of course be the data-generating model itself for future unseen data, that is $p^*(\tilde{y}|\tilde{x}) = p(\tilde{y}|\tilde{x})$. Therefore the minimum generalisation error would be $L = -\int \log(p(\tilde{y}|\tilde{x}))p(\tilde{y}|\tilde{x})dy$. This is exactly the entropy of $p(\bullet|\tilde{x})$, which plays a fundamental role in many fields e.g. statistical physics and information theory. The relative generalisation error for the estimated model is then $-\int \log(p^*(\tilde{y}|\tilde{x}))p(\tilde{y}|\tilde{x})d\tilde{y} + \int \log(p(\tilde{y}|\tilde{x}))p(\tilde{y}|\tilde{x})d\tilde{y} = \int p(\tilde{y}|\tilde{x})\frac{\log(p(\tilde{y}|\tilde{x}))}{p^*(\tilde{y}|\tilde{x})}d\tilde{y}$. This quantity is known as the Kullback-Leibler divergence of $p^*(\bullet|\tilde{x})$ from $p(\bullet|\tilde{x})$, another fundamental quantity for comparing two distributions.

In certain circumstances, negative log-density loss for probabilistic prediction is closely related to squared difference loss for point prediction. In many applications it may be reasonable to approximate $p^*(\bullet|\tilde{x})$ using a Gaussian. If we are further happy to assume that the variance $\sigma^2(\tilde{x})$ of the Gaussian is approximately constant, then $-\log(p^*(\tilde{y}|\tilde{x})) = \frac{1}{2}\log(2\pi\sigma^2(\tilde{x})) + \frac{(\tilde{y}-\mu(\tilde{x}))^2}{2\sigma^2(\tilde{x})}$, where conditional mean $\mu(\tilde{x}) = \int \tilde{y}p^*(\tilde{y}|\tilde{x})d\tilde{y} = E(\tilde{y}|\tilde{x})$ is the optimal action under squared loss for point prediction. That is, generalisation error $-\log(p^*(y|\tilde{x}))$ for Gaussian probabilistic prediction is equivalent to generalisation error $(\tilde{y} - E(\tilde{y}|\tilde{x}))^2$ for point predictions with squared difference loss.

A quick review of the academic literature

There is a large literature on empirical predictive modelling. The literature on the fundamentals of model assessment is considerably more limited. Chapter 7 of Hastie et al. [2008] provides an excellent basic introduction on model assessment.

The need for careful assessment of predictive performance has been recognised for a long time; McKay [1995] provides an excellent review of fundamental concepts. Hand and Vinciotti [2003] shows that in many prediction problems, predictive accuracy matters more in some parts of the data space than in others, and it is appropriate to aim for greater model effectiveness in those regions. If the relevant parts of the space depend on the use to which the model is to be put, then the best model will depend also on this intended use. Hand [2012] reviews the assessment of classification rules, emphasising the need to choose a loss function to match the analysis objectives. Hand [2006] argues, in a classification setting, that model assessment often fails to take into account important aspects of real applications, so that the apparent superiority of more sophisticated methods may be something of an illusion. In particular, simple methods typically yield performance almost as good as more sophisticated methods, to the extent that the difference in performance may be swamped by other sources of uncertainty that generally are not considered in the classical prediction paradigm. Kuffner and Young [2018] argue for careful consideration of conditionality in predictive inference.

Cross-validation has also been used for many years; one of the earliest works on cross-validation is Stone [1974]. It has long been recognised that cross-validation, together with other resampling procedures, can provide a means to assess the predictive performance of predictive models in a non-parametric manner, avoiding dubious or unjustified distributional assumptions. For instance, Stone and Jonathan [1993, 1994] provide a critique of a variety of statistical techniques of actual or potential use in quantitative structure-activity relationship (QSAR) studies and related fields of empirical prediction, exploring the statistical thinking that is needed to underpin those techniques. Emphasis is placed on "exchangeability" as an alternative to unrealistic statistical modelling, and the use of cross-validation to limit self-deception in the use of any particular technique. Arlot and Celisse [2010] provide an excellent survey of cross-validation, with guidelines for choosing the best cross-validation procedure according to the particular features of the problem in hand. Sugiyama [2015] recommends cross-validation as a method of choice for model selection and assessment of generalisation error in reinforcement learning. Raschka [2018] reviews model evaluation and selection in ML, emphasising the role of cross-validation and bootstrapping. Molinaro et al. [2005] provides a comparison of resampling methods for estimation of prediction error. Vehtari et al. [2017] promote the use of cross-validation for estimating pointwise out-of-sample prediction accuracy from a fitted Bayesian model using the log-likelihood evaluated at the posterior simulations of the parameter values. They suggest a Pareto-smoothed importance sampling approach for efficient computation (and R package loo).

Davison and Hinkley [1997] is an obvious reference point for the bootstrap; Hall [2003] provides an interesting context to the development of bootstrap methods. Efron [1983] compares different approaches to cross-validation and bootstrapping for the assessment of prediction error in two-group classification; in practical situations the different methods can give considerably different answers. Cross-validation gives a nearly unbiased estimates, but often with high variability, especially for small sample sizes. The bootstrap gives an estimate with low variability, but sometimes with large negative bias, particularly in highly overfitted situations. A double bootstrap procedure corrects the bias of the ordinary bootstrap without increasing the MSE of estimation. A randomised bootstrap results in substantially lower MSE. Efron and Tibshirani [1986] provides a review of bootstrap methods. Stine [1985] suggests a bootstrap method for estimation of prediction intervals. Tsamardinos et al. [2018] propose an efficient approach, called bootstrap bias corrected cross-validation, to estimate prediction error. Jonathan et al. [1996] use randomised permutation testing to assess the performance of discrimant analysis. In an unsupervised context, Vitale et al. [2017] use randomised permutation testing to estimate the number of components to retain in principal components analysis.

There is a large literature on the application of bootstrapping to estimate bias and variance from dependent data, relevant for estimation of generalisation error for time-series prediction. The work of Carlstein [1986], Hall et al. [1995], Hall and Jing [1996], Garcia-Soidan and Hall [1997], Fukuchi [1999] and Goncalves and White [2004] are noteworthy. Hall et al. [1995] provides recommendations for the choice of block size using dependent data.

Many authors combine cross-validation and bootstrapping to leverage both techniques for improved estimation of predictive performance, and for improved estimation of prediction uncertainty. Of course, bootstrap cross-validation is fundamental to many existing techniques for empirical inference such as random forests; wrapper functions for bootstrap cross-validation in conjunction with standard tools (e.g. locally-weighted regression) are available in the CRAN library for the R statistical modelling environment. Dietterich [1998] recommended the use of a very simple bootstrap cross-validation scheme to assess the relative performance of learning algorithms. Fu et al. [2005] present a bootstrap cross-validation method for evaluation of predictive performance, show to perform better than cross-validation (in isolation) for small samples. Chin [2010] uses bootstrap cross-validation for partial least squares model selection. Cavenaugh [2019] demonstrates that bootstrap cross-validation provides a useful basis for model selection.

Fernandez-Delgado et al. [2014] argue that in general, a large variety of different classification methods is not needed, and that a random forest classifier provides good predictive performance in a large study involving 179 classifiers and 121 data sets. Kazakov and Kiraly [2019] provide the ML automation toolbox (MLaut), which automates large-scale evaluation and benchmarking of ML algorithms on a large number of datasets, and confirm many of the findings of Fernandez-Delgado et al. [2014]: for general-purpose supervised learning, the best performing algorithms are ensembles of trees and kernel-based algorithms; neural networks (including deep learning) do not perform as well.

Vehtari and Ojanen [2012] provide an excellent review of Bayesian predictive model assessment, selection and related methods, emphasising estimation of expected utility in predicting future data, which has been recently enhanced by Gressmann et al. [2018]. Quiñonero-Candela et al. [2006] review different probabilistic predictors, discuss loss functions appropriate for classification and regression problems, and the evaluation of predictive performance. Many authors have attempted to combine fundamental theoretical understanding with empirical modelling for improved predictive skill. This would appear to be the best approach when sufficient resources are available to tackle a problem fully. Kennedy and O'Hagan [2001] described a probabilistic approach (associated with terms such as "emulation", "history matching", "proxy modelling", "uncertainty quantification") widely used in many fields, including conventionally in reservoir modelling for the oil and gas industries. Karpatne et al. [2017] provide recent illustrations in the external literature. The work of authors such as Craig, Goldstein, O'Hagan, Vernon and co-workers in the development statistical models to characterise otherwise intractably complex systems might be regarded as best-in-class, a beacon for data scientists seeking to use empirical models rigorously on a large scale (Kennedy and O'Hagan 2001, Craig et al. 2001, Vernon et al. 2010,Vernon et al. 2014).

6 Conclusions

6.1 Long-term

Recent decades have seen major progress in "Intelligence Augmentation" (IA), using data and computation to augment human intelligence and creativity (e.g. the web search engine, machine vision, NLP). While IA could involve higher-level "reasoning", currently it usually only involves algorithmic pattern-matching useful to human interpretation. The "Internet of Things"(IoT) or "Intelligent Infrastructure" (II) envisions a network of computation, data and physical entities to make human environments more supportive, interesting and safe. Current efforts typically address the problem of establishing basic communications and data streams. However, the real challenge is the establishment of algorithms able to interpret these data streams usefully at a much higher "human" level of abstraction. The tools needed to build future global inference and decision-making systems), blending computer science with statistics, and taking into account the environment and human utilities, are in their infancy. A new engineering discipline needs to emerge, encompassing ideas such as "information", "algorithm", "data", "uncertainty","computing","inference", "optimization", "human-interface" and "human utility". The building blocks of this discipline have begun to emerge in recent decades, but the principles for putting these blocks together safely and reliably for routine deployments have not yet emerged. We have yet to learn from the mistakes of previous deployments, since these have been so few in number. In manufacturing language, we are missing the DEP (design and engineering practice) guidelines for ML- and AI-assisted deployments. But these will emerge in due course as lessons are learned from current and near-future deployments across society.

Jordan [2018]: "AI is the mantra of the current era. The phrase is intoned by technologists, academicians, journalists and venture capitalists alike. As with many phrases that cross over from technical academic fields into general circulation, there is significant misunderstanding accompanying the use of the phrase. But this is not the classical case of the public not understanding the scientists — here the scientists are often as befuddled as the public. The idea that our era is somehow seeing the emergence of an intelligence in silicon that rivals our own entertains all of us — enthralling us and frightening us in equal measure. And, unfortunately, it distracts us."

6.2 Medium-term

Even over the next few years, we might anticipate that a multinational organisation needs to support thousands of digital deployments, each relying on data-driven ML or empirical models. The number of deployments might be expected to grow considerably over time thereafter. Rigorous design and implementation procedures should obviously be followed for each deployment. Because of the empirical nature of the models, it is reasonable to expect that model recalibration (or other modification) would be necessary at least occasionally, even for models (e.g. based on reinforcement learning) which include active exploration of the model environment for empirical learning. To achieve this reasonably using limited "model inspection and maintenance" resources, requires optimal scheduling of model "maintenance" operations. This in turn requires regular "inspection" of model performance (in terms of generalisation error) for all deployments using the techniques outlined in this report. It might be anticipated that all model inspection studies would be conducted following specific standard procedures (depending on the type of model), and that the results of all inspections would be held in a central repository of "model management information" with which to monitor ML-driven deployments organisation-wide. Over time, this repository would itself become a valuable source of historical performance data, for deployments of different model types to different environments and lines of business. The repository would provide the basis for targeting future resources on improved system measurement and data, model research and development, and investments in improved system "hard engineering". It would also provide the basis for optimal "prior" model selection given application characteristics alone.

6.3 Short-term

Best practice for technical assurance of deployed statistical, ML and data science solutions is yet to emerge, since the number of deployments is relatively limited. As best practice evolves, we must be extremely careful to assure current and near-future deployments are assessed as rigorously as possible using the knowledge and experience we have.

Deployments of "off the shelf" statistical and ML algorithms need to work well in-situ in future. It is critical therefore to test models in circumstances likely to represent future operating conditions fully.

Software deployments must of course satisfy standards for industrial software solutions, but this is not enough. They must also satisfy standards for statistical assurance: all empirical predictions must be made (a) to specified accuracy and precision, (b) for any operating conditions within a clearly-defined operating domain, (c) for a specified periods of future operation, and (d) with a clear codified policy for the extent of human intervention for model assessment and recalibration during the operating window.

Data resampling techniques such as cross-validation, bootstrapping and randomised permutation testing, potentially in conjunction with other model assessment tools, provide a rational basis for quantification of generalisation error, estimation of prediction uncertainty, and quantification of statistical significance. All of these quantities are important for selection of the optimal model prior to deployment, and for subsequent "inspection and maintenance" of deployed models.

7 Practical application

The purpose of this section is to illustrate some of the resampling techniques discussed earlier in the report in simple applications. No attempt is made to be exhaustive in our comparisons.

7.1 Non-time-series case

We start by generating synthetic data sets consisting of a single response y and a single covariate x generate, according to

$$\begin{aligned}\text{Observation equation} \quad &: \quad y = g(x, \boldsymbol{\beta}) + \epsilon \\ \text{System function} \quad &: \quad g(x, \boldsymbol{\beta}) = \sum_{k=0}^{p} x^k \beta_k\end{aligned}$$

where $p = p_0 = 2$ (i.e. a quadratic model) for data generation; this model corresponds to a simple form of that given in Section 4.2, expect that no time-series dependence is present. The data sets have n observations, with $\epsilon \sim N(0, \sigma^2)$, $x \sim U(-3, 3)$ and $\beta_0 = 1, \beta_1 = 2, \beta_2 = -2$. The values of n and σ used are given in the Figure 2. Figure 1 illustrates a typical data set. We consider fitting six polynomial models of orders $p = 0$ (a constant) to $p = 5$ (quintic). We estimate the value of p which minimises generalisation error evaluated using cross-validation, loss functions for point and probabilistic prediction, combined with either covariate- or random-blocking of data for cross-validation.

Figure 2 shows the RMSE from cross-validated point predictions corresponding to covariate-blocking (left) and random-blocking (right). Coloured bars represent different combinations of n, σ as shown in the figure label. For every colour, the covariate-blocked RMSE is minimised at the correct quadratic ($p = p_0 = 2$) model. This suggests that covariate-blocked cross-validation is a reasonable resampling technique to identify model order. In contrast, random-blocking correctly identified that the model order p is at least 2, but all more complex models appear equally good. It is likely that this occurs, because the parameter estimates for polynomial terms of the form x^k with $k > 2$ are estimated to be small with random-blocking. In this sense, covariate blocking is seen to be a more exacting form of cross-validation.

Figure 3 illustrates the performance of covariate- and random-blocked cross-validated pseudo-probabilistic prediction, with bootstrapping used to estimate pseudo-probabilistic predictions from point predictions per bootstrap resample. Inferences are most clear when presented in terms of the posterior probabilities of different model orders (on right hand side). Again, covariate-blocked cross-validation (top right) provides better estimation of model order than random-blocking (bottom right).

Figure 4 is the analogue of Figure 3, except that it is based on full probabilistic prediction. That is, the full joint likelihood of withheld observations is used to assess predictive performance. Again, comparing top and bottom panels of Figure 3; covariate-blocked cross-validation provides the best indication of model order. The left- and right-hand panels are equivalent representations of predictive performance in terms of negative log likelihood (left) and posterior model probability (right). Figure 5 is similar to Figures 3 and 4, except that now point-wise probabilistic prediction (i.e. using the product of marginal likelihoods, rather than the joint likelihood) is used for model assessment. Results are very similar to those in Figure 3, indicating that bootstrapping provides a very good approximation to pointwise probabilistic Bayesian prediction.

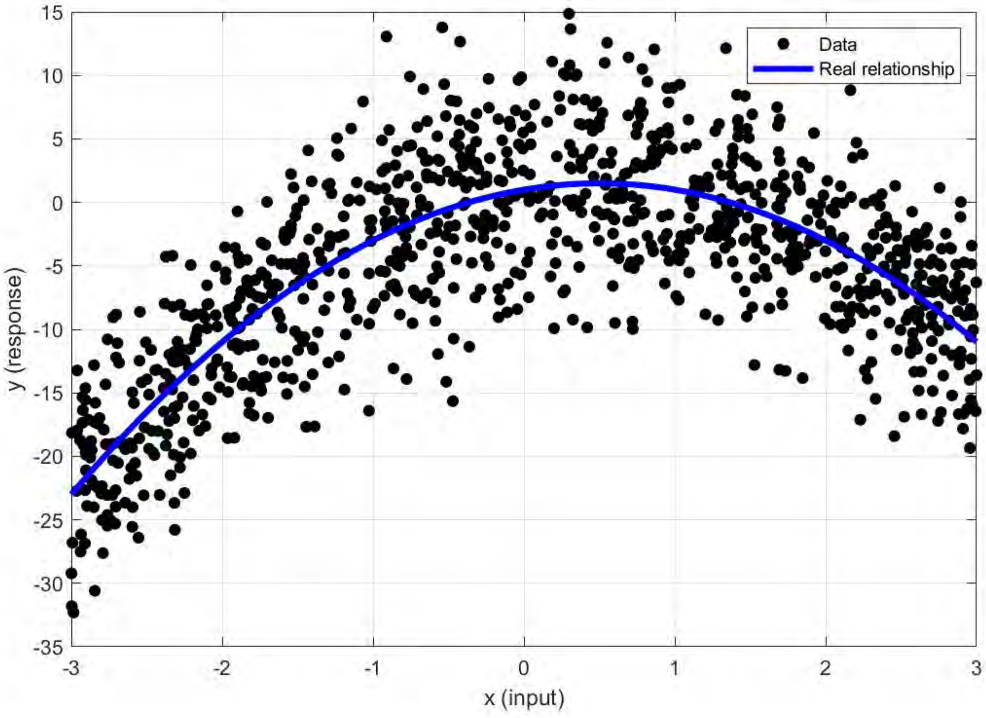

Figure 1: Typical sample (black dots) and true $y(x)$ relationship (blue).

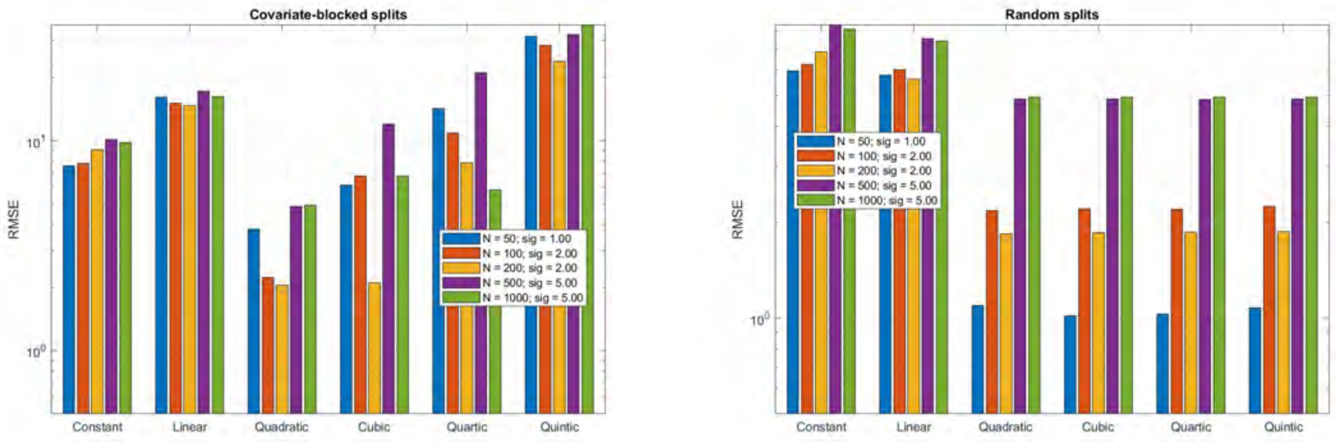

Figure 2: RMSE from point prediction using covariate blocking (left) and random blocking (right), for different model orders, sample sizes and data generation error standard deviations.

Figure 6 compares estimated models from full (joint) probabilistic inference (a, left) using the whole sample for training, and (b, right) using covariate-blocked cross-validated analysis. The latter approach correctly estimates that $p = p_0 = 2$ is the correct model. Predictions from the "whole sample" inference are also reasonable for $p \geq 2$, because the parameter values for coefficients β_k of x^k for $k > 2$ are estimated to be approximately zero.

Figure 7 illustrates the outcome of randomised permutation testing of covariate-blocked cross-validated RMSE for different model orders. Paraphrasing the discussion of Section 3.2, the purpose of the test is to compare the performance of model building (using RMSE) for the true responses (vertical red lines) with the distribution of RMSE for models based on randomised permutations of responses. We would expect that, if a specific model was useful, it would give a lower RMSE for the true responses than for randomised permutations (that is, the red line in Figure 7 sits left of the blue RMSE distribution). Note that randomised permutation testing also has a precise statistical interpretation in terms of exchangeability and non-parametric hypothesis testing. Figure 7 shows that randomised permutations of responses provide considerably better predictive models for model orders $p = 0$ (constant) and $p = 1$ (linear); this is to be expected, since randomising the responses reduces the quadratic trend present between y and x in the true sample (see Figure 1). Conversely, for model order $p \geq 3$, RMSE corresponding to the true responses lies in the tail of the distribution of RMSE from randomised responses, which is promising. But there is still approximately 6% chance of obtaining a better RMSE using randomised responses - which is not so good! We might then estimate that the chance that the actual RMSE was obtained "by chance" is approximately 6%. In marked contrast, RMSE for actual responses for the quadratic model $p = 2$ is clearly much lower than those based on models for randomised responses, indicating that the quadratic model provides the most satisfactory inference in this case.

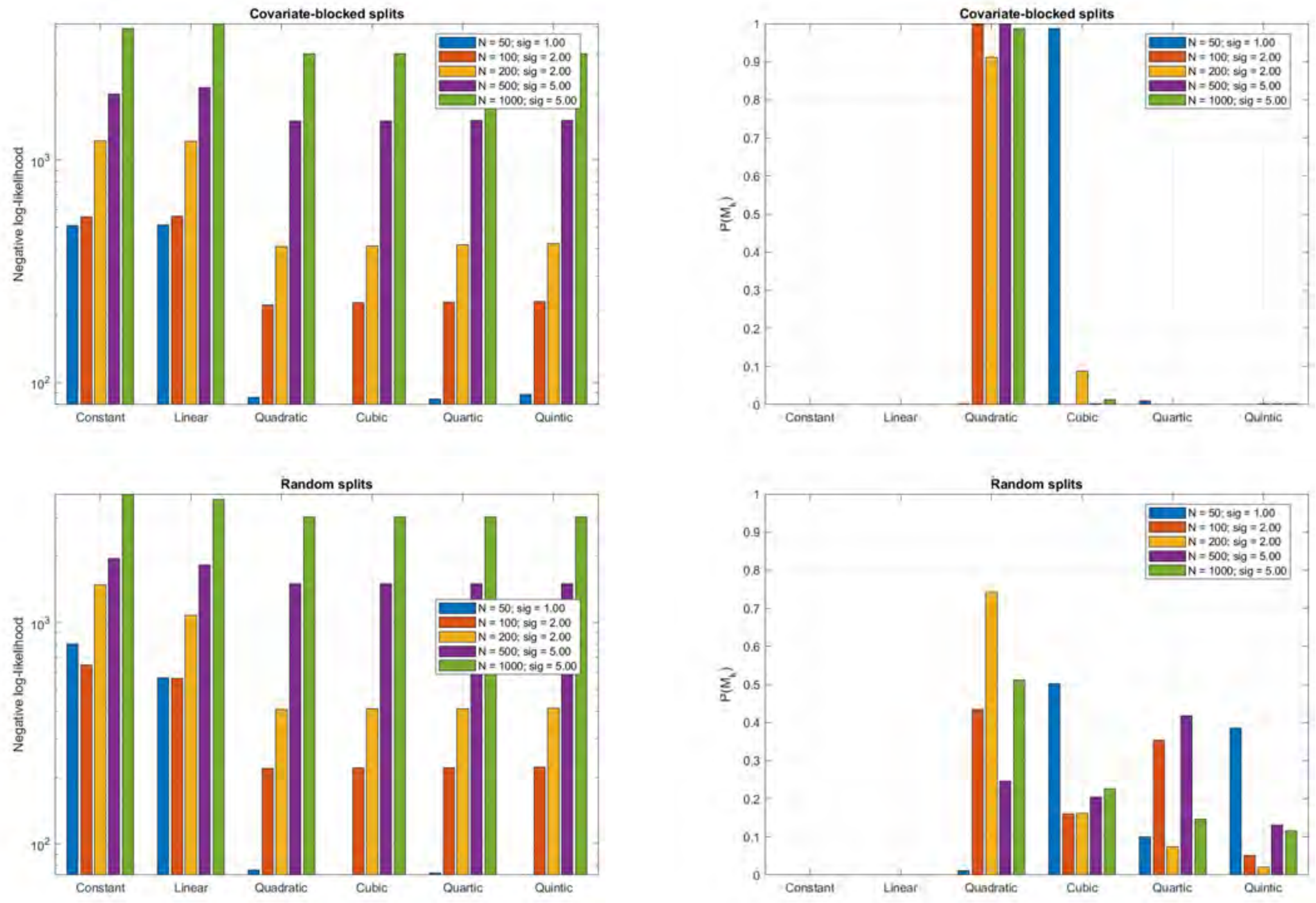

Figure 3: Joint negative log-density (left) and joint posterior probability mass (right) from **pseudo-probabilistic** prediction (using a bootstrapped point prediction). Covariate blocking (top) and random blocking (bottom), for different model orders, sample sizes and data generation error standard deviations.

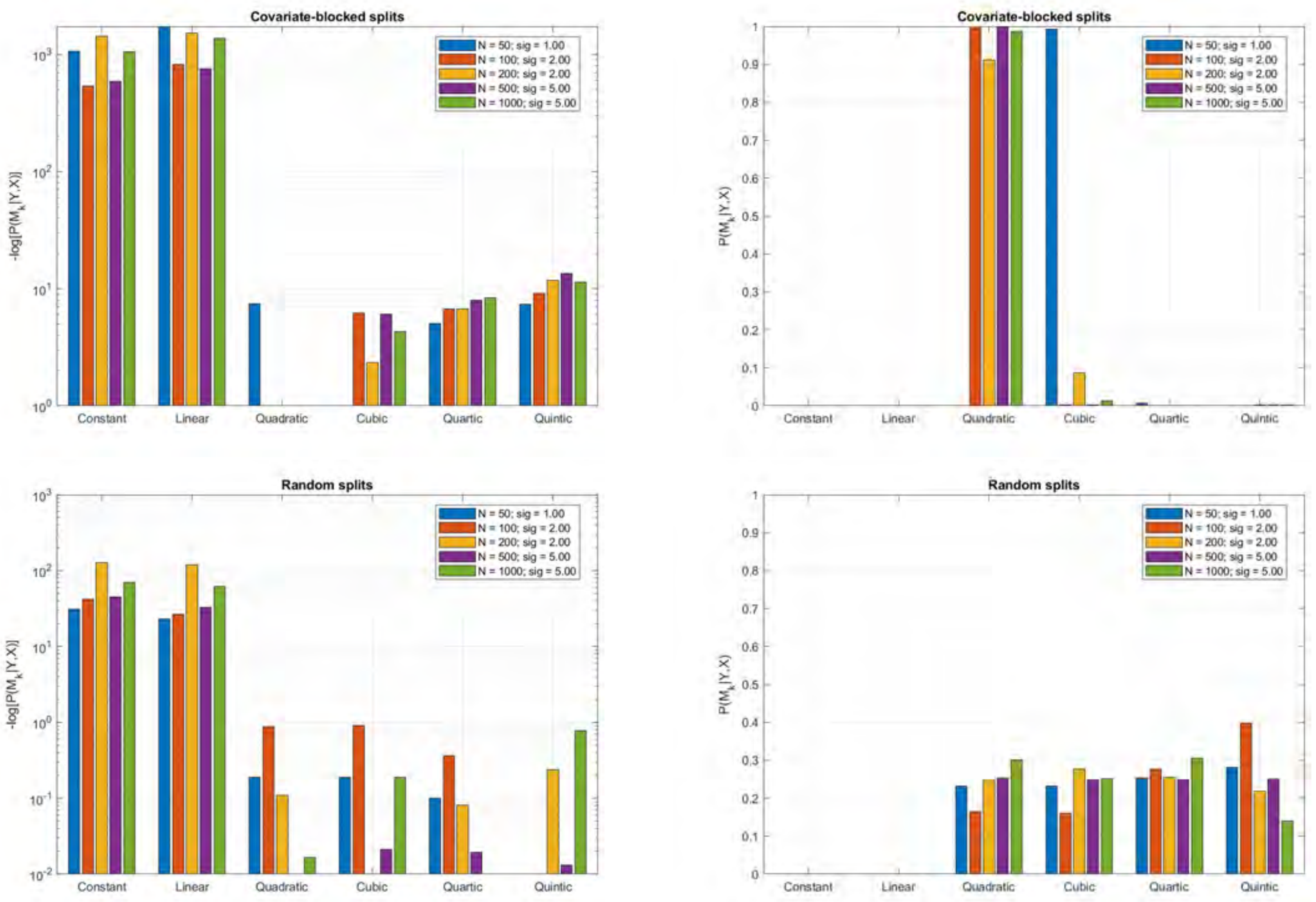

Figure 4: Joint negative log-density (left) and joint posterior probability mass (right) for different models, from full **probabilistic** prediction. Covariate blocking (top) and random blocking (bottom), for different model orders, sample sizes and data generation error standard deviations.

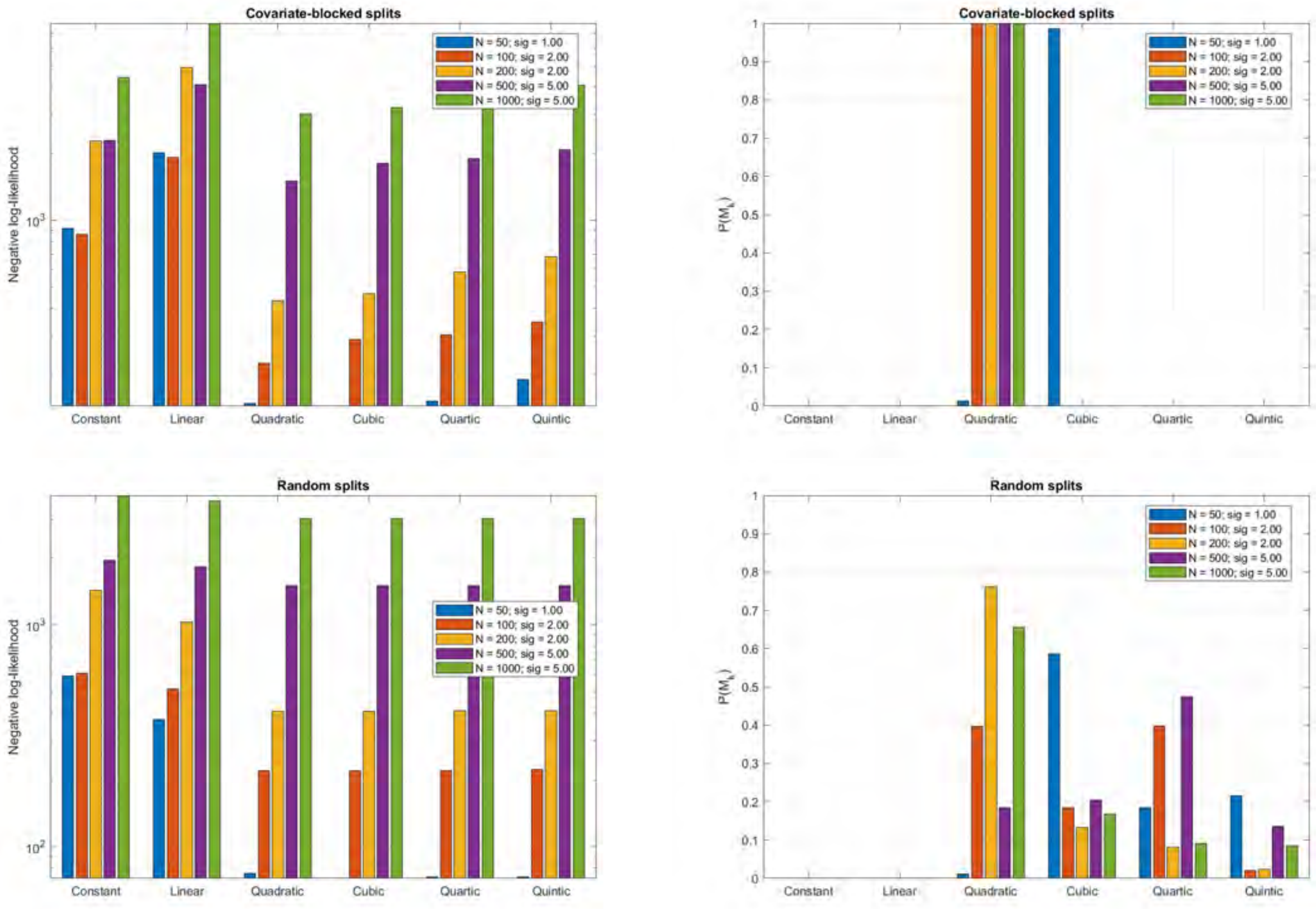

Figure 5: Sum of marginal negative log-densities (left) and sum of marginal posterior probability masses (right) from pseudo-probabilistic prediction (using a bootstrapped point prediction). Covariate blocking (top) and random blocking (bottom), for different model orders, sample sizes and data generation error standard deviations.

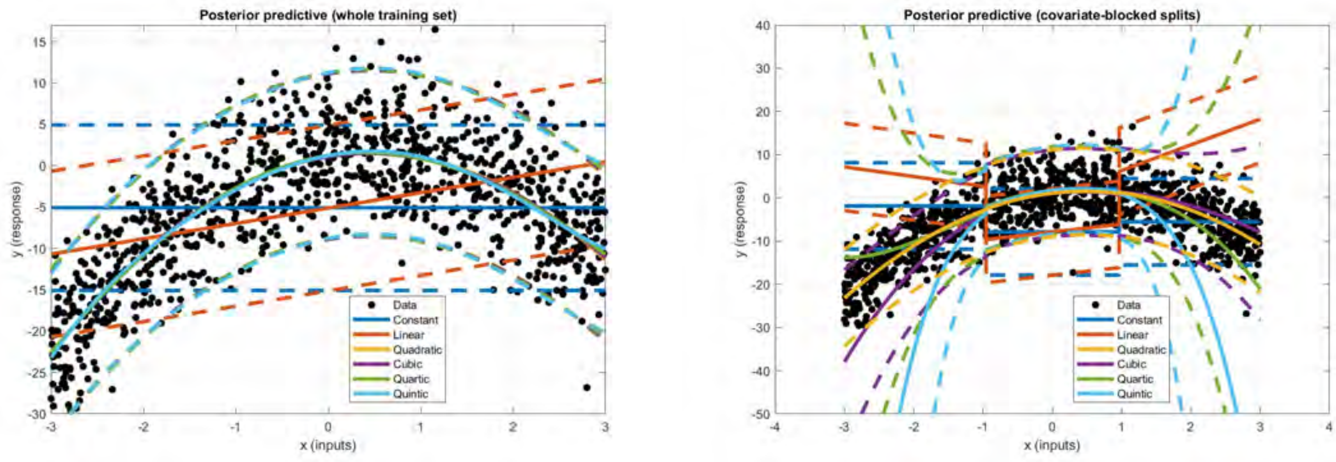

Figure 6: Left: predictive means and ±2 predictive standard deviations, from probabilistic prediction using whole training set for **in-sample** fit. Right: predictive means and ±2 predictive standard deviations, from probabilistic prediction using **covariate-blocked cross-validation**, for different model orders, sample sizes and data generation error standard deviation = 5.

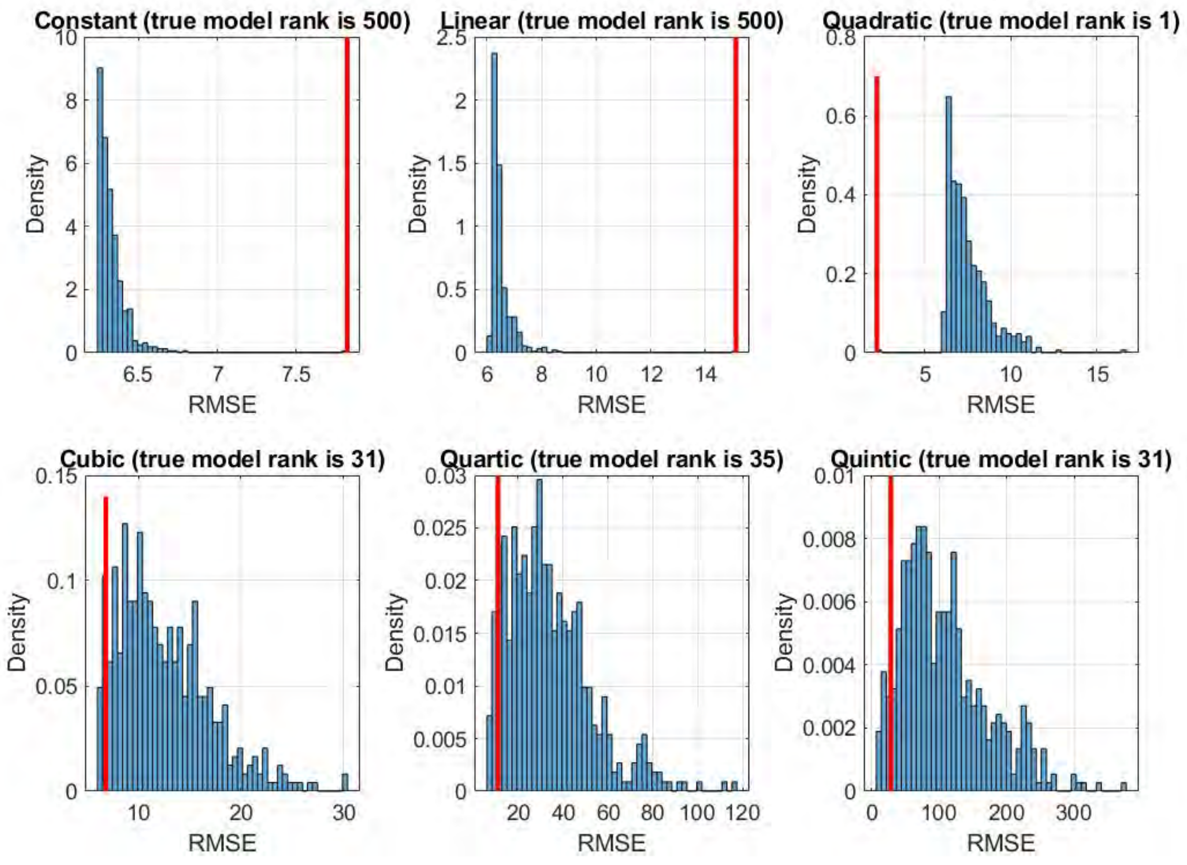

Figure 7: Randomised permutation distribution (based on 500 randomised analyses) of covariate-blocked cross-validated RMSE for different model orders, in the case of point prediction using a sample size of 100 and observation error standard deviation of 2. The vertical red line shows the corresponding RMSE for the true allocation of responses. For the true (quadratic) model, the RMSE estimate is approximately equal to its true value of 2. Moreover, the "p-value" associated with the quadratic fit is the minimum over all model orders considered.

7.2 Time-series case

When the data set corresponds to a time-series, time blocking for cross-validation is a suitable alternative to covariate blocking, although the latter may also still be useful. Figures 8-10 illustrate model assessment using point and probabilistic prediction for the time-series model described in Section 4.2. For this model, the full joint conditional density $f(y_t, y_{t+1}, ..., y_{t+k}|x_t, x_{t+1}, ..., x_{t+k}, \alpha_x, \alpha_\epsilon, \sigma_x, \sigma_\epsilon)$ corresponding to a withheld interval of time-series is available in closed form. It is therefore possible to perform full joint probabilistic prediction in principle. An outline of the calculation of the joint conditional density is given in Appendix 2. It is also possible to perform point predictions using estimated model parameters.

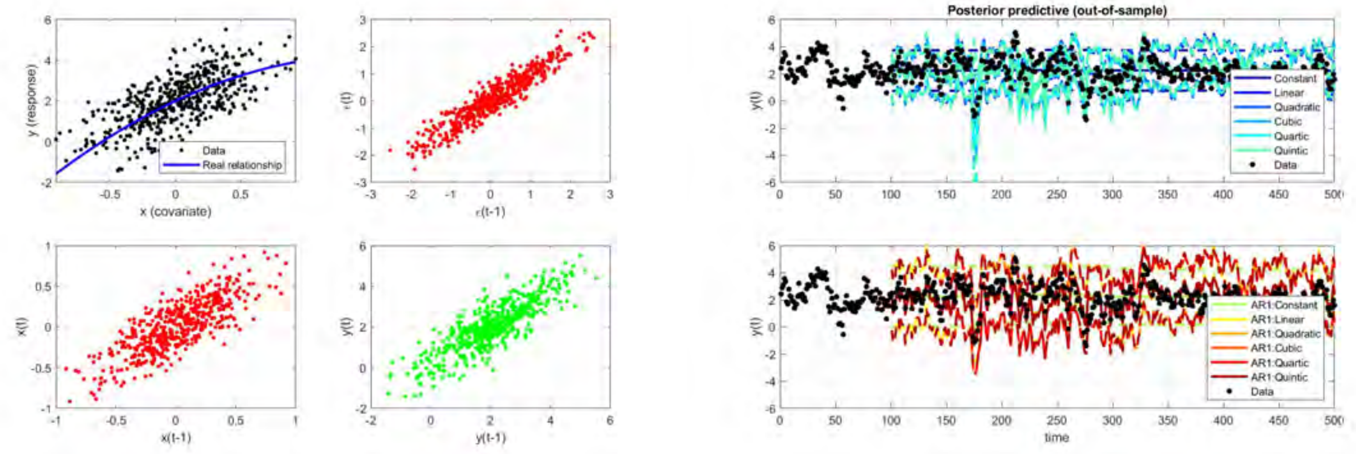

Figure 8: Time-series prediction: Left: scatter plots of illustrating the serial- and cross-dependence present. Right: time-series plot of observed (black dots) and time-blocked cross-validated predictions (with ± two standard deviation predictive error bands) from the models listed in the legends. Predictions accounting for the true time-series structure of the data are considerably less certain than those which ignore it.

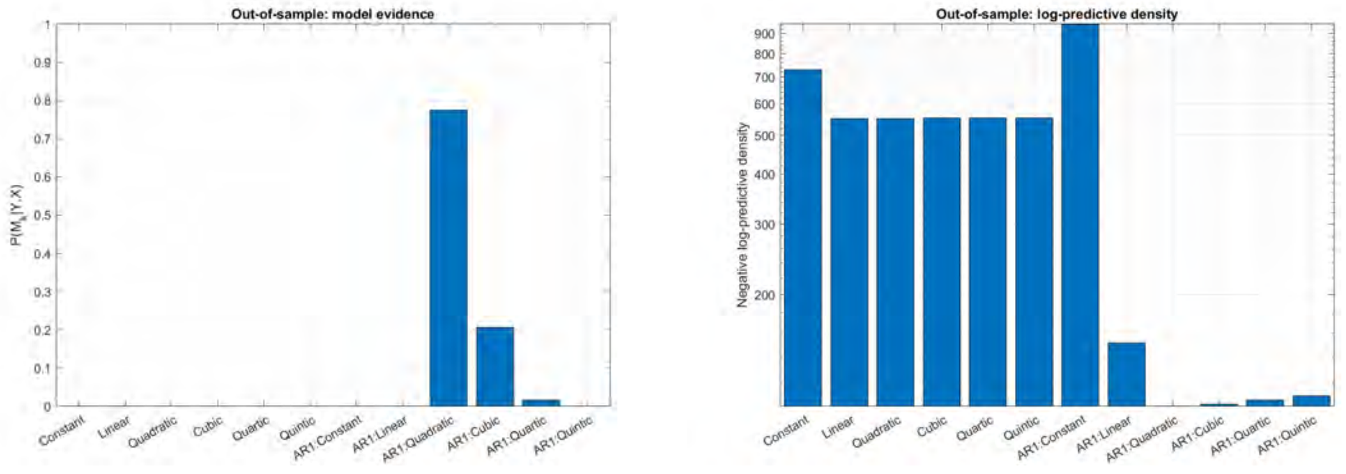

Figure 9: Time-series prediction. Left: out-of-sample model evidence (for full joint likelihood of each model) using time-blocked cross-validation; Right: corresponding negative predictive log likelihood.

Figure 8 illustrates the sample of time-series data, illustrating the serial dependence in all of x, ϵ and y obvious; the strong correlation between x and y is also clear. Figure 8 also shows predictions from different models, in terms of time-blocked out-of-sample model evidence (from full probabilistic prediction) for two types of model. Models on the top-right make the assumption of no time-series dependence in the data (i.e. that $\sigma_\epsilon = \sigma_x = 0$ in the equations in Section 4.2). Models on the bottom-right (prefixed "AR") assume the correct time-series model form. It can be seen that prediction uncertainty from models ignoring the serial correlation present in the data (top-right) appears to be smaller than for a more-appropriate time-series model (bottom-right). In fact, the prediction intervals on the bottom-right more accurately reflect true uncertainty in prediction; this can be demonstrated by "counting" the number of black dots inside and outside the prediction interval. The models on the top-right provide overly-optimistic (and inaccurate) estimates for prediction uncertainty.

Figure 9 shows that full probabilistic prediction identifies that time-series dependence is present, and estimates the correct model order. Only using full probabilistic prediction can the time-series dependence of responses be estimated correctly. However, the right-hand plot suggests, for models which ignore the time-series dependence, that time-blocking can help identify the correct model order if no better approach is available.

Figure 10 demonstrates that point prediction using time-blocked cross-validation can also assist in identifying model order, regardless of assumptions made about serial correlation (although the linear model is identified as optimal, not the quadratic for this realisation of data.

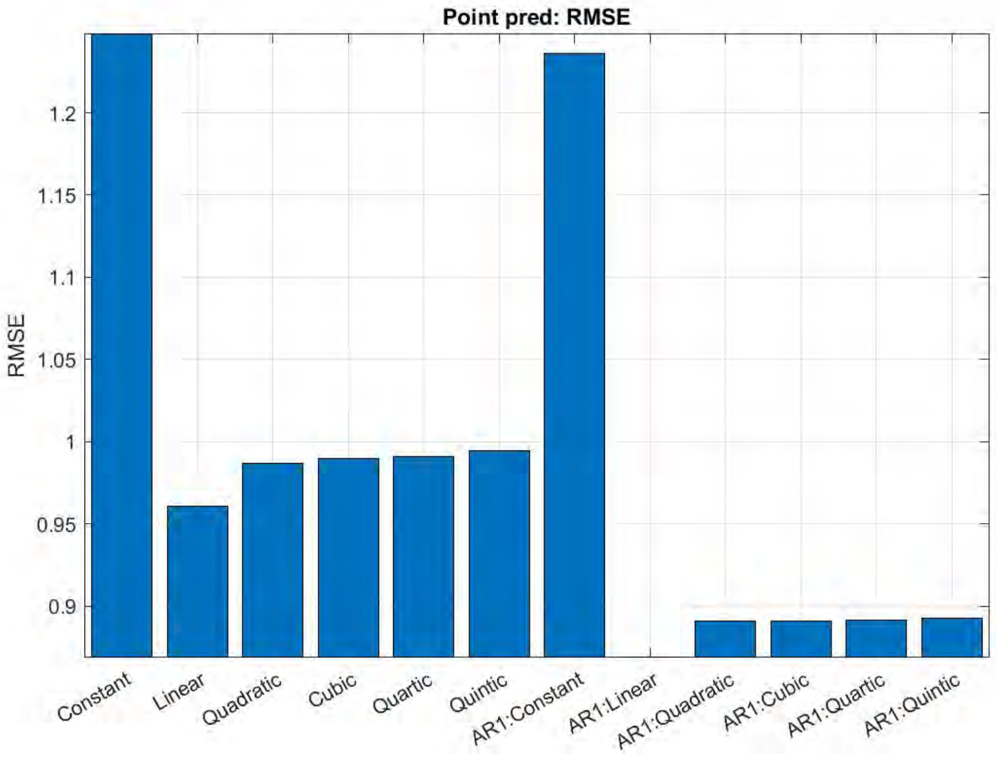

Figure 10: Time-series prediction. RMSE for point predictions using time-blocked cross-validation.

8 Acknowledgement

Sincere thanks to Matthew Jones and Emma Ross of Shell in Amsterdam for assistance in preparing an early version of this document.

References

N. Adams and E. Cohen, editors. *Statistical Data Science.* WORLD SCIENTIFIC (EUROPE), 2018.

S. Arlot and A. Celisse. A survey of cross-validation procedures for model selection. *Statist. Surv.*, 4:40–79, 2010.

AutoML. URL `www.automl.org`.

P. J. Brockwell and R. A. Davis. *Introduction to time series and forecasting.* Springer-Verlag, 2016.

E. Carlstein. The use of subseries values for estimating the variance of a general statistic from a stationary sequence. *Ann. Stat.*, 14(2):590–606, 1986.

J. S. Cavenaugh. Bootstrap cross-validation improves model selection in pharmacometrics, 2019.

W. W. Chin. *Bootstrap Cross-Validation Indices for PLS Path Model Assessment*, pages 83–97. Edts. Esposito Vinzi, V. and Chin, W. W. and Henseler, J. and Wang, H., Springer Berlin Heidelberg, 2010.

P S Craig, M Goldstein, J C Rougier, and A H Seheult. Bayesian forecasting for complex systems using computer simulators. *J. Am. Statist. Soc.*, 96(454):717–729, 2001.

K. Crawford, R. Dobbe, T. Dryer, G. Fried, B. Green, E. Kaziunas, A. Kak, V. Mathur, E. McElroy, A. N. Sanchez, D. Raji, J. L. Rankin, R Richardson, J. Schultz, S. M. West, and M. Whittaker. *AI Now 2019 Report.* AI Now Institute, New York, 2019. URL `ainowinstitute.org/AI_Now_2019_Report.html`.

A. C. Davison and D. A. Hinkley. *Bootstrap Methods and their Application (Cambridge Series in Statistical and Probabilistic Mathematics).* Cambridge University Press, Cambridge, UK, 1997.

T. G. Dietterich. Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Comput.*, 10:1895–1923, 1998.

B Efron. Estimating the error rate of a prediction rule: Improvement on cross-validation. *J. Am. Statist. Soc.*, 78:316–331, 1983.

B. Efron and R. Tibshirani. Bootstrap methods for standard errors, confidence intervals, and other measures of statistical accuracy. *Statist. Sci.*, 1:54–75, 1986.

S. Eldevik. *Position paper: AI + Safety: Safety implications for Artificial Intelligence. Why we need to combine causal- and data-driven models.* DNV GL AS, Norway, 2018.

M. Fernandez-Delgado, E. Cernadas, S. Barro, and D. Amorim. Do we need hundreds of classifiers to solve real world classification problems? *J. Mach. Learn. Res.*, 15:3133?3181, 2014.

W. J. Fu, R. J. Carroll, and S. Wang. Estimating misclassification error with small samples via bootstrap cross-validation. *Bioinformatics*, 21:1979?1986, 2005.

J. I. Fukuchi. Subsampling and model selection in time series analysis. *Biometrika*, 1999.

P. H. Garcia-Soidan and P. Hall. On Sample Reuse Methods for Spatial Data. *Biometrics*, 1997.

S. Goncalves and H. White. Maximum likelihood and the bootstrap for nonlinear dynamic models. *J. Econom.*, 119:199–219, 2004.

F. Gressmann, F. J. Király, B. A. Mateen, and H. Oberhauser. Probabilistic supervised learning. *ArXiv*, abs/1801.00753, 2018.

P. Hall. A Short Prehistory of the Bootstrap, 2003.

P. Hall and B. Jing. On Sample Reuse Methods for Dependent Data. *J. Roy. Statist. Soc. C*, 1996.

Peter Hall, Joel L . Horowitz, and Bing-Yi Jing Jing. On Blocking Rules for the Bootstrap with Dependent Data. 82(3): 561–574, 1995.

D. J. Hand. Classifier technology and the illusion of progress. *Statist. Sci.*, 21:1–14, 2006.

D. J. Hand. Assessing the performance of classification methods. *Int. Stat. Rev.*, 80:400–414, 2012.

D. J. Hand and V. Vinciotti. Local versus global models for classification problems. *The American Statistician*, 57:124–131, 2003.

T. Hastie, R. Tibshirani, and J. Friedman. *The elements of statistical learning. Data mining, inference and prediction.* Springer, 2001.

T. Hastie, R. Tibshirani, and J. Friedman. *The elements of statistical learning. Data mining, inference and prediction (second edition).* Springer, 2008.

P. Jonathan, W. V. McCarthy, and A. M. I. Roberts. Discriminant analysis with singular covariance matrices. A method incorporating cross-validation and efficient randomized permutation tests. *J. Chemometr.*, 10:189–213, 1996.

M. I. Jordan. *Artificial Intelligence - The Revolution Hasn't Happened Yet.* Medium, 2018. URL `medium.com/@mijordan3/artificial-intelligence-the-revolution-hasnt-happened-yet-5e1d5812e1e7`.

Kaggle. URL `kaggle.com`.

A. Karpatne, G. Atluri, J. H. Faghmous, M. Steinbach, A. Banerjee, A. Ganguly, S. Shekhar, N. Samatova, and V. Kumar. Theory-guided data science: A new paradigm for scientific discovery from data. *IEEE Transactions on Knowledge and Data Engineering*, 29:2318–2331, 2017.

V. Kazakov and F. J. Kiraly. Machine learning automation toolbox (MLaut). *arXiv:1901.03678*, 2019.

M. C. Kennedy and A. O'Hagan. Bayesian calibration of computer models. *J. Roy. Statist. Soc. B*, 63:425–464, 2001.

T. A. Kuffner and G. A. Young. Principled statistical inference in data science. In Niall Adams and Edward Cohen, editors, *Statistical Data Science*, pages 21–36. WORLD SCIENTIFIC (EUROPE), 2018.

M. Kuhn and K. Johnson. *Applied predictive modeling.* Springer, 2013.

D. Lehr and P. Ohm. Playing with the Data: What Legal Scholars Should Learn About Machine Learning. *U C Davis Law Review*, 51, 2017. URL `lawreview.law.ucdavis.edu/issues/51/2/symposium/51-2_Lehr_Ohm.pdf`.

S. Makridakis, E. Spiliotis, and V. Assimakopoulos. Statistical and machine learning forecasting methods: Concerns and ways forward. *PLOS ONE*, 13:1–26, 2018a.

S. Makridakis, E. Spiliotis, and V. Assimakopoulos. The M4 competition: Results, findings, conclusion and way forward. *Int. J. Forecast.*, 34:802 – 808, 2018b.

S. Makridakis, E. Spiliotis, and V. Assimakopoulos. The M4 competition: 100,000 time series and 61 forecasting methods. *Int. J. Forecast.*, 36:54 – 74, 2020.

S. G. Makridakis, S. C. Wheelwright, and R. J. Hyndman. *Forecasting: methods and applications.* John Wiley & Sons, 1998.

B. Mateen and R Sonabend. All I want for Christmas is rigorous validation of predictive models to prevent hasty generalisations. In *Significance*, pages 20–24. Royal Statistics Society & Wiley, December 2019.

M. D. McKay. Evaluating prediction Uncertainty. Technical report, Los Alamos National Laboratory NUREG/CR-6311, LA-12915-MS, 1995.

Medium (Data Science). URL medium.com/topic/data-science.

A. M. Molinaro, R. Simon, and R. M. Pfeiffer. Prediction error estimation: a comparison of resampling methods. *Bioinformatics*, 21:3301–3307, 2005.

J. Quiñonero-Candela, C. E. Rasmussen, F. Sinz, O. Bousquet, and B. Schölkopf. Evaluating predictive uncertainty challenge. In J. Quiñonero-Candela, I. Dagan, B. Magnini, and F. d'Alché Buc, editors, *Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Tectual Entailment*, pages 1–27, 2006.

S. Raschka. Model evaluation, model selection, and algorithm selection in machine learning. *ArXiv*, abs/1811.12808, 2018.

RSS. *A Guide for Ethical Data Science*. Royal Statistical Society, 2019.

A. L. St. Clair, O. Smogeli, A. Odegardstuen, J. A. Glomsrud, S. Eldevik, and C. Nadeau. *Position paper: Trustworthy Industrial AI Systems*. DNV GL AS, Norway, 2019.

R. A. Stine. Bootstrap prediction intervals for regression. *J. Am. Statist. Soc.*, 80:1026–1031, 1985.

M. Stone. Cross-validatory choice and assessment of statistical predictions. *J. Roy. Statist. Soc.*, 36:111–147, 1974.

M. Stone and P. Jonathan. Statistical thinking and technique for QSAR and related studies. Part I : General theory. *J. Chemometr.*, 7:455–475, 1993.

M. Stone and P. Jonathan. Statistical thinking and technique for QSAR and related studies. Part II : Specific methods. *J. Chemometr.*, 8:1–20, 1994.

M. Sugiyama. *Statistical Reinforcement Learning: Modern Machine Learning Approaches*. Chapman and Hall/CRC, 2015.

I. Tsamardinos, E. Greasidou, M. Tsagris, and G. Borboudakis. Bootstrapping the out-of-sample predictions for efficient and accurate cross-validation. *Mach. Learn.*, 107:1895–1922, 2018.

A. Vehtari and J. Ojanen. A survey of Bayesian predictive methods for model assessment, selection and comparison. *Statist. Surv.*, 6:142–228, 2012.

A. Vehtari, A. Gelman, and J. Gabry. Practical Bayesian model evaluation using leave-one-out cross-validation and WAIC. *Statistics and Computing*, 27:1413–1432, 2017.

I. Vernon, M. Goldstein, and R. G. Bower. Galaxy formation: a Bayesian uncertainty analysis. *Bayesian Analysis*, 5:619–669, 2010.

I. Vernon, M. Goldstein, and R. Bower. Galaxy formation: Bayesian history matching for the observable universe. *Statistical Science*, 29(1):81–90, 2014.

R. Vitale, J. A. Westerhuis, T. Naes, A. K. Smilde, O. E. de Noord, and A. Ferrer. Selecting the number of factors in principal component analysis by permutation testing. numerical and practical aspects. *J. Chemometr.*, 31(12):e2937, 2017.

M. West and J. Harrison. *Bayesian forecasting and dynamic models*. Springer, 1997.

Appendix 1 : Illustration of estimation of joint conditional density for probabilistic prediction of time-series

Using the model form in Section 4.2, the joint conditional density $f(y_t, y_{t+1}, ..., y_{t+k} | x_t, x_{t+1}, ..., x_{t+k}, \alpha_x, \alpha_\epsilon, \sigma_x, \sigma_\epsilon)$ can be evaluated relatively easily. Note that all random variables are Gaussian-distributed, and hence evaluation of means and covariances is sufficient to specify the full joint conditional density. We also assume that model parameters are known for ease of explanation. When not known, they would be estimated by Bayesian inference. The equations below would then be replaced by the corresponding equations for posterior predictive distributions.

$$\begin{aligned} y_t|x_t &= g(x_t) + \epsilon_t \quad \text{gives} \\ E(y_t|x_t) &= g(x_t) \quad \text{and} \\ E(y_t|x_t)^2 &= g^2(x_t) + \sigma_\epsilon^2/(1-\alpha_\epsilon^2) \quad \text{so that} \\ \text{var}[y_t|x_t] &= E(y_t|x_t)^2 - (E(y_t|x_t))^2 = \sigma_\epsilon^2/(1-\alpha_\epsilon^2). \end{aligned}$$

Further

$$\begin{aligned} E(y_t, y_{t+1}|x_t, x_{t+1}) &= g(x_t)g(x_{t+1}) + \alpha_\epsilon \sigma_\epsilon^2/(1-\alpha_\epsilon^2) \quad \text{from which} \\ \text{cov}(y_t, y_{t+1}|x_t, x_{t+1}) &= E(y_t, y_{t+1}|x_t, x_{t+1}) - E(y_t|x_t)E(y_{t+1}|x_{t+1}) = \alpha_\epsilon \sigma_\epsilon^2/(1-\alpha_\epsilon^2). \end{aligned}$$

Hence

$$y_t, y_{t+1} | x_t, x_{t+1} \sim N\left(\left[\begin{array}{c} g(x_t) \\ g(x_{t+1}) \end{array} \right], \frac{\sigma_\epsilon^2}{1-\alpha_\epsilon^2} \left[\begin{array}{cc} 1 & \alpha_\epsilon \\ \alpha_\epsilon & 1 \end{array} \right] \right),$$

and by analogy

$$y_t, y_{t+1}, ..., y_{t+k} | x_t, x_{t+1}, ..., x_{t+k} \sim N\left(\left[\begin{array}{c} g(x_t) \\ g(x_{t+1}) \\ . \\ . \\ . \\ g(x_{t+k}) \end{array} \right], \frac{\sigma_\epsilon^2}{1-\alpha_\epsilon^2} \left[\begin{array}{cccc} 1 & \alpha_\epsilon & ... & \alpha_\epsilon^k \\ \alpha_\epsilon & 1 & ... & \alpha_\epsilon^{k-1} \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \\ \alpha_\epsilon^k & \alpha_\epsilon^{k-1} & ... & 1 \end{array} \right] \right).$$

The joint conditional likelihood for any interval of $k+1$ observations can now be evaluated, allowing the estimation of generalisation error.