Extreme Learning Machine for Missing Data using Multiple Imputations

Dušan Sovilj[a,b,*], Emil Eirola[a], Yoan Miche[b], Kaj-Mikael Björk[a], Rui Nian[c], Anton Akusok[d], Amaury Lendasse[a,d]

[a] *Arcada University of Applied Sciences, 00550 Helsinki, Finland*
[b] *Aalto University School of Science, FI-00076, Finland*
[c] *Ocean University of China, 266003 Qingdao, China*
[d] *The University of Iowa, Iowa City, IA 52242-1527, USA*

Abstract

In the paper, we examine the general regression problem under the missing data scenario. In order to provide reliable estimates for the regression function (approximation), a novel methodology based on Gaussian Mixture Model and Extreme Learning Machine is developed. Gaussian Mixture Model is used to model the data distribution which is adapted to handle missing values, while Extreme Learning Machine enables to devise a *multiple imputation* strategy for final estimation. With multiple imputation and ensemble approach over many Extreme Learning Machines, final estimation is improved over the mean imputation performed only once to complete the data. The proposed methodology has longer running times compared to simple methods, but the overall increase in accuracy justifies this trade-off.

Keywords: Extreme Learning Machine, missing data, multiple imputation, gaussian mixture model, mixture of gaussians, conditional distribution

1. Introduction

Recurring problem in many scientific domains is the accurate prediction or forecast for unknown and/or future instances. This issue is addressed by assuming that there exist the underlying mechanism that generates the available data, and then building a model that provides good enough approximation for that same mechanism. Finally, any kind of inference is based on the constructed model assuming all the necessary information is taken into account. The task of making predictions, for example, daily temperature or retail sales for some specific time period, is considered a *regression* problem or estimation of the regression function. Another issue becoming more prevalent in Machine Learning domain is related to the missing data in databases encountered in many research areas [1, 2, 3, 4]. This issue has huge impact on both the learning algorithms and the subsequent inference procedures. If this issue is not treated correctly, any kind of inference results in severely biased and inaccurate estimates.

In the paper, we are interested with regression problems of the form

$$y_i = f(\mathbf{x}_i) + \epsilon_i \tag{1}$$

in the presence of missing data where $(\mathbf{X}; \mathbf{Y}) = \{(\mathbf{x}_i; y_i)\}_{i=1}^{N}$ are data samples with $\mathbf{x}_i$ consisting of d explanatory features or variables, y_i the target variable and ϵ_i the noise term. The usual assumption behind the noise term is that it follows a Gaussian distribution with zero mean and known variance $\epsilon \sim \mathcal{N}(0, \sigma^2)$. The regression problem is to find a model $\mathcal{M}$ that is a close approximation to the true underlying function f.

*Corresponding author.
Email addresses: dusan.sovilj@aalto.fi (Dušan Sovilj), emil.eirola@arcada.fi (Emil Eirola), yoan.miche@aalto.fi (Yoan Miche), kaj-mikeal.bjork@arcada.fi (Kaj-Mikael Björk), anton-akusok@uiowa.edu (Anton Akusok), amaury-lendasse@uiowa.edu (Amaury Lendasse)

The case explored in this paper is when samples or observations $\mathbf{X}$ contain unobserved (unknown) variables, that is, the values are missing for certain observation and features. Values could be missing for a variety of reasons depending on the source of the data, including measurement error, device malfunction, operator failure, and many others. On the other hand, many modelling methods assume that data contain a fixed number of samples lying in a fixed feature space. Presence of missing values prevents these methods to be applied directly to the data. Simple *ad hoc* solutions to incomplete data include completely removing samples containing unobserved values, mean imputation with the mean computed on available values, replacement from correlated variables, substitution based on prior information (such as regional codes) and others. In order to provide more reliable inference after the modelling stage, and suitable strategy must be employed.

Besides simple ad hoc procedures, there are several paradigms for dealing with missing data used in conjunction with machine learning methods [5] and these include:

- *Conditional mean imputation* approach which is optimal in terms of minimising the mean squared error of the imputed values, but suffers from biased statistics of the data. For instance, estimates of variance or distance are negatively biased.

- *Random draw imputation* that is more appropriate for generating a representative instance of a fully imputed data set. However, the imputations can be highly variable with respect to any single value to be accurate.

- *Multiple imputation*. This setup draws several representative imputations of the data, analyses each set separately, and combines the results to form an overall estimate with uncertainty taken into account [6]. This approach can result in unbiased and accurate estimates after a sufficiently high number of draws, but it is not always straightforward to determine the posterior distribution to draw from [7]. In the context of Machine Learning, repeating the analysis several times is however impractical as training and analysing a sophisticated model tends to be computationally expensive.

The conceptually simplest approach to dealing with incomplete data is to fill in the missing values before commencing any further analysis. Many methods have been suggested for imputation with the intent to appropriately conform to the distribution of the data. These include imputation by nearest neighbours [8], or the improved incomplete-case k-NN imputation [9]. An alternative approach is to study the input density indirectly through conditional distributions, by fully conditional specification [10]. However, the uncertainty of the imputed values is often not explicitly modelled in most imputation methods, and hence ignored in the further analysis, potentially leading to biased results.

Having an appropriate model to take into consideration missing data has several advantages. First, with any kind of imputation, many learning algorithms can be directly applied to imputed data, such as neural networks, Gaussian processes and density estimation methods. Second, having a specific model designed to tackle missing values allows to take into consideration the variability of imputed values, and thus, the variance of the final estimation the practitioner is interested about.

Finite mixture models are a powerful modelling tool with a wide array of applications. Of considerate importance is the Gaussian Mixture Model (GMM), also known as Mixture of Gaussians, which has been studied extensively to describe distributions of a data set. This model provides a suitable estimation of the underlying data density distribution as GMM is a universal approximator [11]. This enables GMM to model any kind of continuous densities to arbitrary precision, and has been employed for a variety of problems in vision [12, 13], language identification [14], speech [15, 16] and image [17, 18] processing. The parameters of GMM are obtained via maximum likelihood (ML) estimation by the Expectation-Maximisation (EM) algorithm [19]. EM algorithm is a general purpose algorithm for finding the ML solution with latent variables or incomplete data and does not require any derivatives of the likelihood function. GMM has been extended to accommodate missing values in data sets [20, 21] which has seen some resurgence in recent years [22, 23, 24].

In this paper, we are considering regression estimation in the presence of missing data. First, mixture of Gaussians is applied to original data with missing values. Second, a large number of imputations is performed, that is, a multiple imputation approach is adopted. After all newly formed data sets are available,

a suitable regression model is build. As the number of draws can be large and the data sets can often contain huge number of samples, a fast (in terms of training speed) and accurate model should be used. The choice is on Extreme Learning Machine (ELM) as it satisfies both criteria. In the case of difficult data, where substantial number of imputed data sets is required, ELM acts a good model as fast computational models are more viable than the alternative gradient-based neural networks or kernel methods.

Gaussian Mixture Model has been used to train neural networks in the presence of missing data [25] with the average gradient computed for the relevant parameters by using conditional distribution for the missing values. The method is designed to handle training of networks with back-propagation and is not applicable to other Machine Learning methods. Extreme Learning Machine has also been adapted to handle missing values [26, 27] with both approaches estimating distances between samples that are subsequently used for the RBF kernel in the hidden layer. One advantage of that approach is circumventing estimation of all the missing values and focusing only on providing required information for the methods based on distances, such as Support Vector Machines or k-nearest neighbours. However, the method only returns expected pairwise distances that are then employed by the ELM for regression. The downside is that other activations functions have to be ignored, and the imputation is done once by the conditional mean. Although conditional mean imputation provides improved results over simple ad hod solutions, it neglects the variability introduced by the underlying Gaussian Mixture Model.

The rest of the paper is organised as follows: Section 2 explains the overall approach in more detail focusing on the main points in the methodology. Two main components of the approach, namely mixture of Gaussians for missing data and Extreme Learning Machine are explained in Sections 3 and 4 respectively. Section 5 showcases the results between two types of imputation – conditional mean and multiple imputation, combined with two different modelling strategies. Finally, summarising remarks are given in Section 6.

2. Methodology

The overall approach consist of four consecutive stages:

1. Fitting the Gaussian Mixture Model on a data set with missing values.
2. Generating new data sets via multiple imputation based on the Gaussian Mixture Model from the first stage.
3. Building Extreme Learning Machine for each generated data set in the second stage.
4. Combining all the Extreme Learning Machines to provide final estimates.

2.1. Gaussian Mixture Model Fitting

In the first stage, a Gaussian Mixture Model Γ is fitted to the data with missing values. Since the data contains missing values, straightforward application of the EM algorithm is not possible and certain adjustments are necessary for both E and M-steps. In the E-step, conditional expectations with respect to known values in samples are used to obtain means and covariances for the missing values. In the M-step, the conditional mean fills the missing parts (per sample imputation) in order to compute GMM component means. The covariance matrices for each component are similarly adjusted taking into account covariances for the missing parts. The details required to carry out these corrections are explained in Section 3.1.

Besides the internal parameters of the model in the form of mixing coefficients, means and covariance matrices for each Gaussian component, $k = 1, \ldots, K$, one parameter that has to be validated is the number of components K. Validation step is important in order to prevent the model from overfitting and many model selection criteria have been proposed to combat this problem. In GMM case, adding a single component to the mixture can result in a large increase in the number of free parameters which is quadratic in the number of dimensions. Two most popular criteria for model selection are Akaike's Information Criterion (AIC) [28] and Bayesian Information Criterion (BIC) [29]. AIC usually tends to select overly complex models [30], which is the reason BIC is usually favoured when model selection needs to be performed and it is adopted in our method.

2.2. Multiple Imputation for Missing Values

Conditional mean imputation remains one of the prevalent solutions to the problem of missing data offering better estimation than simple ad hoc procedures [27]. However, it provides insufficient information for further inference as the whole underlying distribution is condensed into a single value. For this reason, multiple imputation considers many replacements for the unobserved values, and subsequent inference is based on combining inferences across all imputed versions. This way uncertainty about the missing values is taken into account.

In the second stage of the methodology, many data sets are drawn from GMM model Γ obtained in the first phase. The imputation is done per sample $\mathbf{x}_i$ that contains unobserved values. The conditional distribution for missing variables $p(\mathbf{x}_i^{\text{miss}} \,|\, \mathbf{x}_i^{\text{obs}})$ (conditioned on observed variables) is again a *Gaussian Mixture Model* with adjusted components based on the parameters of Γ. Drawing from the adjusted Gaussian Mixture Model is straightforward, but it shows that conditional mean imputation can be severely biased if the conditional distribution $p(\mathbf{x}_i^{\text{miss}} \,|\, \mathbf{x}_i^{\text{obs}})$ is multimodal.

The filling in is done per sample until all the samples are processed. This complete sweep produces one new data set, and the procedure is repeated for prespecified number of times, say V. The end results of this stage is V new data sets $\{\mathbf{X}^v\}_{v=1}^V$. Looking from a different perspective, for each missing value there are a total of V imputations which are then collected to form new data sets. The next step is to train the models on these complete data sets.

2.3. Training Extreme Learning Machines

Extreme Learning Machine is a novel type of neural network that does not have any iterative stages in the learning phase. The network has a single hidden layer, where the input weights coming from the input variables are *randomly initialised*. The model draws its fast training times based on this initialisation while the output weights connecting the hidden layer and the output layer are obtained by solving a linear system. Since the result of multiple imputation is a collection of data sets $\{\mathbf{X}^v\}_{v=1}^V$, it is important to have a fast model able to tackle such scenario.

The final steps involves training ELMs for each data set $\mathbf{X}^v$ which gives us a total of V models $\mathcal{M}_v$. Prediction for a fresh sample $\mathbf{x}_{\text{new}}$ is then combined across all models as an average over their respective predictions, that is,

$$\hat{y}_{\text{new}} = \frac{1}{V} \sum_{v=1}^{V} \mathcal{M}_v(\mathbf{x}_{\text{new}}) \tag{2}$$

where $\mathcal{M}_v(\mathbf{x}_{\text{new}})$ is the output of a model $\mathcal{M}_v$ for a sample $\mathbf{x}_{\text{new}}$. This way, estimation of $\hat{y}_{\text{new}}$ properly reflects sampling variability due to incomplete data.

2.4. Two Modelling Strategies

The proposed methodology where there are V models suggests an ensembling or combining strategy which has been researched within ELM community [31, 32, 33, 34]. The ensembling strategy provides both empirical success over selection methods [35] and explicitly takes into account model selection uncertainty [36]. Assigning model weights involves yet another step requiring additional computational resources. In the case of missing data, averaging over all estimations presents a natural choice to combining as given by Eq. (2) which implies that all the model weights are equal.

Two strategies are tested in the experiments. The first strategy involves generating the input weights $\mathbf{W}$ only once. This initialisation is then used for all the V data sets giving V models. However, due to the nature of the ELM and a linear relationship between hidden and output layer, prediction for a fresh sample no longer requires having all the models in memory since the sum can enter the model itself. This means that *only one model* is produced after the training stage with the output weights being an average of the output weights of all the models $\mathcal{M}_v$. The ELM obtained in this manner takes into account uncertainty about missing values, and is different from a single ELM that has the *same initialisation* with the weights $\mathbf{W}$.

Second strategy can be considered true combining approach, as V models are generated, one for each data set $\mathbf{X}^v$. Each model has different initialisation of the input weights, and the actual prediction is done as given by Eq. (2).

3. Mixture of Gaussians for Missing Data

The goal of Expectation-Maximisation algorithm [19] is to find the maximum likelihood solution to the case where there are unknown latent variables added to the data. In the case of Gaussian Mixture Model's with K components, the criterion to be optimised is the log-likelihood

$$\log \mathcal{L}(\theta) = \log p(\mathbf{X} \,|\, \theta) = \sum_{i=1}^{N} \log \left(\sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_i \,|\, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right), \tag{3}$$

where $\mathcal{N}(\mathbf{x} \,|\, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ is the probability density function of the multivariate normal distribution, and $\theta = \{\pi_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_{k=1}^{K}$ is the set of parameters to be determined. The E-step consists of finding the posterior probabilities for the latent variables, while in the M-step new values for means $\boldsymbol{\mu}_k$, covariances $\boldsymbol{\Sigma}_k$ and mixing coefficients π_k are recomputed based on the probabilities from the E-step. All the parameters θ are initialised to some suitable values (with the constraints $0 < \pi_k < 1$ and $\sum_{k=1}^{K} \pi_k = 1$), and then E and M-steps are alternated until convergence either in the log-likelihood or in the parameter values. For the EM algorithm, we are only considering the input or feature space $\mathbf{X}$, ignoring the target vector $\mathbf{Y}$ altogether.

3.1. Extension for the Missing Data

When dealing with missing data, it is important to establish what kind of missing-data mechanism is appropriate for the problem. In the paper, we are assuming that a missing value represents a value which is defined and exists, but the reason for its missingness is unknown. This assumption corresponds to Missing-at-Random mechanism [5], where the event of a measurement missing is independent from the value it would take conditioned on the observed values.

The standard EM algorithm for mixture of Gaussians has been extended to handle missing data [20, 21]. The input data $\mathbf{X}$ is a set of observations $\{\mathbf{x}_i\}_{i=1}^{N}$ where for each sample there exist an index set $O_i \subseteq \{1, \ldots, d\}$ covering the variables with known values. For every index set O_i, there is a corresponding complement set M_i indexing missing values in the sample $\mathbf{x}_i$. In the case with missing values, the observed log-likelihood can be written as

$$\log \mathcal{L}(\theta) = \log p(\mathbf{X}^O \,|\, \theta) = \sum_{i=1}^{N} \log \left(\sum_{k=1}^{K} \pi_k \mathcal{N}\left(\mathbf{x}_i^{O_i} \,|\, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\right) \right) \tag{4}$$

where $\mathbf{X}^O = \{\mathbf{x}_i^{O_i}\}_{i=1}^{N}$ and $\mathcal{N}(\mathbf{x}_i^{O_i} \,|\, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ is used for the *marginal* multivariate normal distribution probability density of the observed values of $\mathbf{x}_i$.

To account for the missing data, certain additional expectations need to computed in the EM algorithm. These are conditional expectations to compute missing components of a sample with respect to each Gaussian component k, and their conditional covariance matrices, i.e., $\tilde{\boldsymbol{\mu}}_{ik}^{M_i} = \mathrm{E}[\mathbf{x}_i^{M_i} \,|\, \mathbf{x}_i^{O_i}]$, and $\tilde{\boldsymbol{\Sigma}}_{ik}^{MM_i} = \mathrm{Var}[\mathbf{x}_i^{M_i} \,|\, \mathbf{x}_i^{O_i}]$, where the mean and covariance are calculated under the assumption that $\mathbf{x}_i$ originates from the kth Gaussian. For convenience, we also define corresponding imputed data vectors $\tilde{\mathbf{x}}_{ik}$ and full covariance matrices $\tilde{\boldsymbol{\Sigma}}_{ik}$ which are padded with zeros for the known components. Then the E-step is:

$$t_{ik} = \frac{\pi_k \mathcal{N}(\mathbf{x}_i^{O_i} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}_i^{O_i} \mid \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}, \tag{5}$$

$$\tilde{\boldsymbol{\mu}}_{ik}^{M_i} = \boldsymbol{\mu}_k^{M_i} + \boldsymbol{\Sigma}_k^{MO_i}(\boldsymbol{\Sigma}_k^{OO_i})^{-1}(\mathbf{x}_i^{O_i} - \boldsymbol{\mu}_k^{O_i}), \qquad \tilde{\mathbf{x}}_{ik} = \begin{pmatrix} \mathbf{x}_i^{O_i} \\ \tilde{\boldsymbol{\mu}}_{ik}^{M_i} \end{pmatrix}, \tag{6}$$

$$\tilde{\boldsymbol{\Sigma}}_{ik}^{MM_i} = \boldsymbol{\Sigma}_k^{MM_i} - \boldsymbol{\Sigma}_k^{MO_i}(\boldsymbol{\Sigma}_k^{OO_i})^{-1}\boldsymbol{\Sigma}_k^{OM_i}, \qquad \tilde{\boldsymbol{\Sigma}}_{ik} = \begin{pmatrix} \mathbf{0}^{OO_i} & \mathbf{0}^{OM_i} \\ \mathbf{0}^{MO_i} & \tilde{\boldsymbol{\Sigma}}_{ik}^{MM_i} \end{pmatrix}. \tag{7}$$

The notation $\boldsymbol{\mu}_k^{M_i}$ refers to using only the elements from the vector $\boldsymbol{\mu}_k$ specified by the index set M_i, and similarly for $\mathbf{x}_i^{O_i}$, etc. For matrices, $\boldsymbol{\Sigma}_k^{MO_i}$ refers to elements in the rows specified by M_i and columns by O_i. The expressions for the parameters in Eqs. (6) and (7) originate from the observation that the conditional distribution of the missing components also follows a multivariate normal distribution with these parameters [37, Thm. 2.5.1].

The M-step is slightly altered to reflect that we are dealing with missing data. Component means are estimated based on the imputed data vectors $\tilde{\mathbf{x}}_{ik}$ and the covariance matrix estimates require an additional term concerning covariances of imputed values:

$$\boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{i=1}^{N} t_{ik} \tilde{\mathbf{x}}_{ik} \tag{8}$$

$$\boldsymbol{\Sigma}_k = \frac{1}{N_k} \sum_{i=1}^{N} t_{ik} \left[(\tilde{\mathbf{x}}_{ik} - \boldsymbol{\mu}_k)(\tilde{\mathbf{x}}_{ik} - \boldsymbol{\mu}_k)^T + \tilde{\boldsymbol{\Sigma}}_{ik} \right] \tag{9}$$

$$\pi_k = \frac{N_k}{N}. \tag{10}$$

An efficient method to evaluate required matrix inverse operations in the Eqs. (6) and (7) is to use the sweep operator [5]. These steps are explained in more detail in [26]. In the same paper, a case of high dimensional spaces is also studied with the proposed solution based on high-dimensional data clustering [38]. The problem in high-dimensional data is reliable estimation of the parameters as it becomes difficult to fit Gaussians mixture model.

3.2. Sampling from Conditional GMM

The conditional distribution of the missing values of a sample with respect to a Gaussian mixture model also follows a Gaussian mixture distribution. The parameters defining this distribution correspond to the parameters calculated in the E-step in Eqs. (5)–(7). Specifically, the distribution of the missing values of a sample $\mathbf{x}_i$ is a Gaussian mixture of K components with means

$$\tilde{\boldsymbol{\mu}}_{ik}^{M_i} = \boldsymbol{\mu}_k^{M_i} + \boldsymbol{\Sigma}_k^{MO_i}(\boldsymbol{\Sigma}_k^{OO_i})^{-1}(\mathbf{x}_i^{O_i} - \boldsymbol{\mu}_k^{O_i}) \tag{11}$$

and covariances

$$\tilde{\boldsymbol{\Sigma}}_{ik}^{MM_i} = \boldsymbol{\Sigma}_k^{MM_i} - \boldsymbol{\Sigma}_k^{MO_i}(\boldsymbol{\Sigma}_k^{OO_i})^{-1}\boldsymbol{\Sigma}_k^{OM_i}. \tag{12}$$

The mixing coefficients correspond to the probabilities of the sample to originate from each component

$$t_{ik} = \frac{\pi_k \mathcal{N}(\mathbf{x}_i^{O_i} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}_i^{O_i} \mid \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}. \tag{13}$$

The conditional mean imputation of the missing values is then realised as the weighted average of the component centres:

$$\tilde{\mathbf{x}}_i^{M_i} = \sum_{k=1}^{K} t_{ik} \tilde{\boldsymbol{\mu}}_{ik}^{M_i} \,. \tag{14}$$

Sampling from the conditional distribution is accomplished by first fixing the component k (drawing from the categorical distribution defined by probabilities t_{ik}), drawing $|M_i|$ independent standard normal variables into a vector $\mathbf{z}$, and using the Cholesky factor L of the covariance matrix $\tilde{\boldsymbol{\Sigma}}_{ik}^{MM_i}$ corresponding to component k. A representative sample is then generated by

$$\tilde{\boldsymbol{\mu}}_{ik}^{M_i} + L\mathbf{z} \,. \tag{15}$$

3.3. Initialisation of Parameters

The parameters $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$ are initialised in the following way. The means $\boldsymbol{\mu}_k$ are chosen randomly from the observed data, favouring samples without missing values if possible. If the data does not have enough complete samples, some of the centres are set to incomplete samples where the missing values are replaced by the sample means. The covariances $\boldsymbol{\Sigma}_k$ are initialised with the sample covariance of the data ignoring the samples with missing values.

3.4. Selecting Number of Components K

The number of components is selected according to the Bayesian Information Criterion (BIC), also known as Schwarz's criterion [29]. BIC is useful when model selection aims at identifying the true model in the candidate set, as it is a consistent criterion, that is, it correctly picks the true models as the number of samples goes to infinity. For likelihood based models, BIC can be easily computed with the following formula:

$$\mathrm{BIC} = -2 \log \mathcal{L}(\theta) + P \log N \tag{16}$$

where P is the number of free parameters. In the case of full covariance matrix for each component k, there are in total $P = Kd + K - 1 + Kd(d+1)/2$ parameters to estimate: Kd for the means, $Kd(d+1)/2$ for the covariance matrices and $K-1$ for the mixing coefficients. As the number of dimensions d increases, the number of parameters quickly tends to become larger then available samples making the BIC criterion invalid. One possibility to circumvent this issue is by imposing restrictions on the structure of covariance matrices making the model less powerful.

A simple and commonly applied method to determine appropriate number of components is the following: start with a single component $K = 1$, learn the model with this single component Γ_1, then set $K = 2$ and check whether the newly fitted Γ_2 is more suitable than the model with $K = 1$. If this is the case, keep increasing K until the criterion (BIC in our case) no longer improves.

4. Extreme Learning Machine

Extreme Learning Machine (ELM) [39, 40] presents a novel technique for training a neural network that has been applied to variety of cases [41, 42, 43, 44]. ELM belongs to a family of single-hidden layer feedforward networks (SLFN) which considerably reduces the training time. These networks are particularly appealing due to their *universal approximation* capability, meaning that any continuous function f can be approximated with desired level of accuracy [45]. To reach that accuracy requires a suitable learning algorithm that adjusts or tunes all the network parameters. The most well known algorithm for training these networks is the back-propagation algorithm [46] which is an iterative procedure based on the gradient of the error function. The novelty that distinguishes ELM is that certain network parameters need not be tuned, and instead can be randomly generated.

Let us consider a data set with N samples in d dimensional space, i.e., $\{(\mathbf{x}_i; y_i)\}_{i=1}^{N} \in \mathbb{R}^d \times \mathbb{R}$. A SLFN models the data sample $\mathbf{x}_i$ with

$$\sum_{j=1}^{L} \beta_j g_j(\mathbf{w}_j \cdot \mathbf{x}_i + b_j), \quad i = 1, \ldots, N \tag{17}$$

where $\mathbf{w}_j$ are the input weights, b_j the bias, operation $\mathbf{w}_j \cdot \mathbf{x}_i$ is the inner product and g_j the activation function for the jth neuron in the hidden layer. β_j is the output weight coming from the jth neuron to the output neuron, while L is the total number of neurons in the hidden layer. In this paper, we only consider the case with a single output, that is, the target y_i is a scalar value. The above formula can be easily generalised to the multivariate case with a vector of outputs $\mathbf{y}_i$. For the univariate case, the hidden layer output weights can be collected in a vector as $\boldsymbol{\beta} = [\beta_1, \ldots, \beta_L]^T$.

That SLFN can approximate the data with zero error means that the output of a network is exactly the desired target value y_i across all samples, that is,

$$\sum_{i=j}^{L} \beta_j g_j(\mathbf{w}_j \cdot \mathbf{x}_i + b_j) = y_i, \quad i = 1, \ldots, N \tag{18}$$

Eq. (18) can be written in more compact form as

$$\mathbf{H}\boldsymbol{\beta} = \mathbf{y} \tag{19}$$

where

$$\mathbf{H} = \begin{bmatrix} g_1(\mathbf{w}_1 \cdot \mathbf{x}_1 + b_1) & \cdots & g_L(\mathbf{w}_L \cdot \mathbf{x}_1 + b_L) \\ \vdots & \ddots & \vdots \\ g_1(\mathbf{w}_1 \cdot \mathbf{x}_N + b_1) & \cdots & g_L(\mathbf{w}_L \cdot \mathbf{x}_N + b_L) \end{bmatrix}_{N \times V}, \tag{20}$$

$$\boldsymbol{\beta} = \begin{bmatrix} \beta_1 \\ \vdots \\ \beta_L \end{bmatrix} \quad \text{and} \quad \mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_L \end{bmatrix}. \tag{21}$$

$\mathbf{H}$ is the hidden layer output matrix or feature mapping of the SLFN. The jth column of $\mathbf{H}$ is the output of jth hidden neuron for all the data samples, while the ith row of the matrix is the sample $\mathbf{x}_i$ put through all the neuron, that is, a transformed sample in the new feature space $\mathbb{R}^L$. Training SLFN requires tuning all the parameters of the network – $\mathbf{w}_j$, b_j and β_j, $i = j, \ldots, L$.

The novelty that ELM brings to learning of SLFN is that network parameters $\mathbf{w}_j$ and b_j need not be tuned at all – they can be randomly generated before encountering the data and kept fixed throughout the learning stage. This does not prevent the ELM from losing its universal approximation capability, provided that activation functions $g_j(\mathbf{x})$ used in the hidden layer follow certain mild conditions [47, 48]. The probability distribution for $\{(\mathbf{w}_j, b_j)\}_{j=1}^{N}$ can be *any continuous* probability distribution from any interval on $\mathbb{R}^d \times \mathbb{R}$. By dropping the tuning of input weights for the hidden layer, the only remaining adjustable parameters are the output weights $\boldsymbol{\beta}$. Given that relation between hidden layer matrix $\mathbf{H}$ and output is $\mathbf{y}$ is linear, the solution to Eq. (19) is obtained with ordinary least-squares approach, that is, the goal is to find $\boldsymbol{\beta}$ for the following minimisation problem

$$\min_{\boldsymbol{\beta}} \|\mathbf{y} - \mathbf{H}\boldsymbol{\beta}\|^2. \tag{22}$$

As $\mathbf{H}$ might be non-square (in the case $L < N$), the solution is to use Moore-Penrose generalised inverse (pseudo-inverse) $\mathbf{H}^\dagger$ of matrix $\mathbf{H}$ which gives the solution as

$$\boldsymbol{\beta} = \mathbf{H}^\dagger \mathbf{y} = (\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{y}. \tag{23}$$

This provides a direct analytic solution to the learning process which in some cases reduces computational time by a large margin compared to iterative approaches [47, 49].

The complete basic ELM algorithm can be summarised in three steps:

1. Randomly generate hidden node parameters w_j and b_j, $j = 1, \ldots, L$.
2. Compute the hidden layer matrix $\mathbf{H}$
3. Compute the output weights $\boldsymbol{\beta} = \mathbf{H}^{\dagger}\mathbf{y}$.

In order to provide more stable solution to the minimisation problem Eq. (22), a regularisation term C is added to the diagonal of $\mathbf{H}^T\mathbf{H}$. The computation of output weights $\boldsymbol{\beta}$ is now

$$\boldsymbol{\beta} = (\mathbf{H}^T\mathbf{H} + C\mathbf{I})^{-1}\mathbf{H}^T\mathbf{y} \tag{24}$$

which is a solution to a regularised least-squares problem

$$\min_{\boldsymbol{\beta}} \|\mathbf{y} - \mathbf{H}\boldsymbol{\beta}\|^2 + C\|\boldsymbol{\beta}\|^2 \tag{25}$$

which in statistics is know as ridge regression [50]. This modified version of the basic ELM is used in the experiments. The parameter C can be cross-validated to achieve better results for a data set at hand. In our experiments, we are skipping this validation phase in order to speed up the execution time and we are using a small value of $C = 10^{-6}$ to prevent numerical instabilities in the computation of $(\mathbf{H}^T\mathbf{H} + C\mathbf{I})^{-1}$.

4.1. Selecting Number of Neurons

One potential pitfall with the ELM model is the appropriate complexity or the number of neurons in the hidden layer to capture the overall variations in data. This issue is also known as *model structure selection*. Two general approaches have been proposed to tackle this issue: selection methods [51, 49] and model ensembling [33, 34]. Selection methods are focused on picking *the best model* from a set of candidate models where each of the models is of different complexity. The notion of the best model among the candidate set is based on a specific criterion which usually involves having an estimate of the networks performance (either training error plus some penalty term or a validation error). Both AIC and BIC can be again used for this purpose. The second strategy, ensemble modelling, considers many networks structures with the aim of having model weights to discard poor models (assigning them zero weights) and giving similar weights for models with similar performance.

The proposed method in this paper closely follows the ensemble approach. Instead of considering different models on the same data, in our method there is an ensemble of the models with the same complexity (same number of neurons) trained on imputed data sets $\mathbf{X}^v$. Different from the ensemble where the weights for each model are learned based on some performance criterion, in this paper we are simply taking an average over all the models, that is, all the model weights are equal. Surprisingly, carefully choosing the complexity can be alleviated with the proposed approach.

5. Experiments

The effectiveness of the proposed approach is tested on several data sets taken from Machine Learning repositories. Since all the data sets used do not have any missing values, the real case scenario is simulated by removing some portion of the data before the whole methodology is applied. Values in the data set are removed at random with a fixed probability until a prespecified number of instances are discarded. In the experiments, we are only focused on supervised regression task, but the approach can easily be extended to classification tasks (binary and multiclass). Although it is possible to use outputs $\mathbf{Y}$ as another feature to help estimate missing values in the input samples $\mathbf{X}$, they are left out during the first stage when fitting the mixture of Gaussians.

Table 1 shows all the data sets used in the experiments. Data sets are taken from two repositories for Machine Learning related tasks: the UCI Machine Learning repository [52] and the LIACC regression repository [53].

Table 1: Data sets used in the experiments. N indicates the total number of samples in the data set and d denotes the number of features.

Name	N	d
Abalone	4177	8
Bank_8FM	4500	8
Boston housing	506	13
Machine CPU	209	6
Stocks	950	9
Wine quality (red)	1599	11

Table 2: Number of neurons in the ELM used for each data set.

Name	min L	step size	max L
Abalone	100	100	1000
Bank_8FM	200	200	2000
Boston housing	25	25	300
Machine CPU	10	10	120
Stocks	50	50	500
Wine quality (red)	100	100	1000

The comparison is done between two strategies for missing values imputation: conditional mean imputation (CM) and multiple imputation (MI). Both are based on the same fitted Gaussian Mixture Model which is the first step in the methodology. In the CM case, each missing value is replaced by the conditional mean value given by Eq. (14) which gives a data set $\mathbf{X}^{\text{cm}}$. On the other hand, for the MI scenario a total of $V = 1000$ new data sets $\{\mathbf{X}^v\}_{v=1}^V$ are generated using Eq. (15) that are subsequently used to train the ELM models. The criterion for comparison is squared error risk which is estimated as an average of 10 Monte-Carlo runs on a test set. That is, the data set containing missing values is first split into training and test parts. The training part contains two-thirds of the samples, while the remaining third belongs to the test set. Since we are adopting this approach, it is possible to have missing values in the test set, and if this is the case, they are replaced by their true values in order to be able to compute required risk.

Once the splitting is done, the variables are standardised to zero mean and unit variance since the ELM is sensitive to the range of the variables. The test set is then modified according to the statistics (mean and variance) obtained on a training set.

5.1. ELM initialisation

For the ELM, only sigmoid activation function is used for all the neurons. The number of neurons is varied from low to high enough where some overfitting might occur. For all the data sets, the maximum tested number of neurons is close to the available samples for training. For example, for "'Wine quality (red)" data which has just over 1000 samples for training, the maximum number of neurons used in the training stage is 1000. Table 2 summarises the tested structures for each data set.

5.2. Execution Time

Table 3 summarises the running time of three main parts of the methodology (GMM fitting, generating data sets and ELM training) for the tested data sets under MI strategy. The results are for one specific set of experiments where the number of missing values is set to 10%, the ELM networks are fixed to 100 neurons and there are 1000 generated data sets. The experiments are done in MATLAB 2014b environment running on Intel Xeon E31230 @ 3.20GHz and 8Gb of memory (with 4 computational threads running in parallel). The vast majority of time (more than 98% of total running time) is spent on finding the parameters of the GMM which in turns depends on the number of missing values in the data, the convergence speed of the EM algorithm, the number of initialisations of the parameters (tries or restarts for the EM), number of samples,

Table 3: Execution time (in seconds) of the proposed methodology for the case with 10% missing values, network with 100 neurons and there are 1000 imputations. The last column is the number of components of the GMM model (averaged over 10 Monte-Carlo runs).

data set	GMM fitting	generating samples	training ELMs	components
Abalone	2631.8	1.63	8.84	4.2
Bank_8FM	1168.3	1.73	9.51	3.6
Housing	83.1	0.11	1.53	1.25
Machine CPU	62.3	0.05	1.18	3.775
Stocks	648.3	0.64	2.06	10.7
Wine (red)	618.2	0.66	3.95	4.7

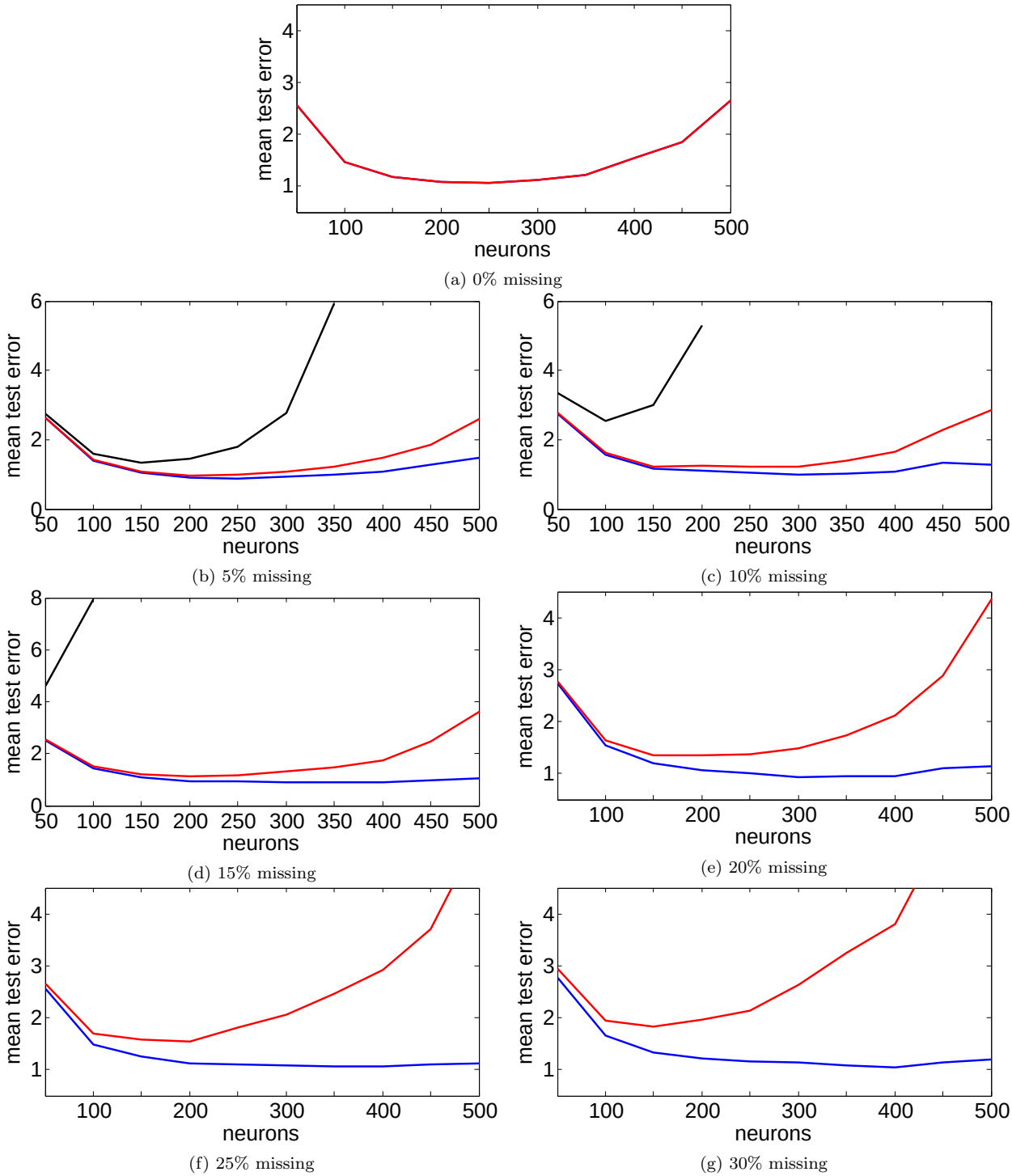
dimensionality of data and the "optimal" number of components. As explained in Section 3.4, the number of components K keeps increasing as the BIC keeps decreasing. For each value of K, there are 10 repetitions of the EM algorithm in order to find several local maxima and return the best one. The quickest step is the imputation part for the complete data since sampling from the GMM is straightforward.

5.3. Performance: Single Initialisation

The first set of experiments is focused on a single initialisation of the input weights of the ELM. The complete setup (initialisation of weights, training the network, and testing) is executed 50 times and the results are averaged to accommodate for the random initialisation of the network parameters. There are two averaging phases necessary for risk estimation: for the random initialisation of the network and for the splitting of the data into training and test parts.

Figure 1 shows the results of a single initialisation of the ELM for two imputation strategies. As explained in Section 2.4, the latter case also produces one model which has same memory footprint as the former case of imputation. The results are presented for different percentages of missing values in the data (considering all Nd values): 5%, 10%, 15%, 20%, 25% and 30%, and for different number of neurons. The result for no missing values is also given to showcase the effect of overfitting and the effect of missing parts on risk estimation. As a comparison, in the figure the results for a simple strategy of removing samples with missing values are also presented. With this simple strategy, the overall performance quickly deteriorates as the number of samples becomes scarce. The number of available samples for Stocks data is the following 633, 392, 249, 140, 90, 47 and 29 for the cases of 5%, 10%, 15%, 20%, 25% and 30% of missing values respectively. With lack of samples, it becomes difficult to train more complex ELMs, which is another drawback of this simple strategy. In Figure 1e, only a model with 50 neurons is trained which gives mean test error of 10.3 which is larger by a factor of 3 than both conditional mean and multiple imputation strategies. The same trend (quick decrease in mean test error) is observed for all other tested data sets (figures not shown).

The case with no missing values shows that ELM is prone to overfitting when the number of neurons approaches the number of available samples, that is, there is an "optimal" model for the data. On the other hand, multiple imputation approach can drastically improve upon conditional mean imputation in both respects: 1) when the number of missing values increases, and 2) when the complexity of the networks increases. This pattern where MI strategy does not suffer too much as the number of neurons increases is also present for all other data sets. An interesting thing to notice is that the proposed approach based on MI does not suffer as the complexity increases when data contains high percentage of missing values. With MI approach, the model with 500 neurons remains competitive with lower complexity models for cases of 20%, 25% and 30% of missing data. The main reason is that the MI strategy has an inherent averaging mechanism (from the imputations) which helps reduce the error and prevents overfitting. This can be thought of as another ensembling approach, whilst not on models (as there is only one initialisation), but on data sets. This effectively *removes the model structure selection phase* and can potentially save considerable amount of time. Figure 2 shows the comparison between the most complex network versus the best performing network to further demonstrate this effect. The best performing network is taken to be the one with *the lowest value*,



(a) 0% missing

(b) 5% missing

(c) 10% missing

(d) 15% missing

(e) 20% missing

(f) 25% missing

(g) 30% missing

Figure 1: Average test mean squared error for Stocks data set as the number of missing data increases. Blue line indicates multiple imputation scenario (MI), red line signifies conditional mean imputation (CM) and black line is simple removal strategy.

that is, it is chosen *once the testing is done*. Although this cannot be done in practice (choosing a model on unseen data), it is shown only to provide an insight that choosing the most complex network stays very close to the optimal network structure, and in some cases it is actually better than the CM strategy (Boston housing and Stocks data).

5.4. Performance: Ensemble Modelling

For convenience, denote with $\mathcal{M}(\mathbf{W}, \mathbf{X})$ an ELM model whose input weights are initialised with values given by matrix $\mathbf{W}$ and then trained on a data set $\mathbf{X}$. In the case of single initialisation of input weights for ELM, the number of models trained depends on the desired number of generated data sets, which in our case is $V = 1000$. Even though the end results is a single ELM in multiple imputation case, training stage still involves finding solutions to V linear systems. A more suitable comparison is between the *ensemble* of different models for both CM and MI strategies. In this case, for the CM strategy there are different V initialised models that are trained on the mean imputed data set $\mathbf{X}^{\text{cm}}$. This provides us with V models $\mathcal{M}(\mathbf{W}_v, \mathbf{X}^{\text{cm}})$, $v = 1, \ldots, V$. For the MI strategy, the models with the same initialisation as in CM strategy are trained on the respective V generated data sets $\mathbf{X}^v$ which results in models $\mathcal{M}(\mathbf{W}_v, \mathbf{X}^v)$. Finally, for both approaches, the final prediction for the test set is simply an average of predictions across all models which is given by Eq. (2). The final predictions are given by

$$\hat{y}^{\text{cm}}_{\text{new}} = \frac{1}{V}\sum_{v=1}^{V} \mathcal{M}(\mathbf{W}_v, \mathbf{X}^{\text{cm}}; \mathbf{x}_{\text{new}}), \qquad \hat{y}^{\text{mi}}_{\text{new}} = \frac{1}{V}\sum_{v=1}^{V} \mathcal{M}(\mathbf{W}_v, \mathbf{X}^{v}; \mathbf{x}_{\text{new}}) \tag{26}$$

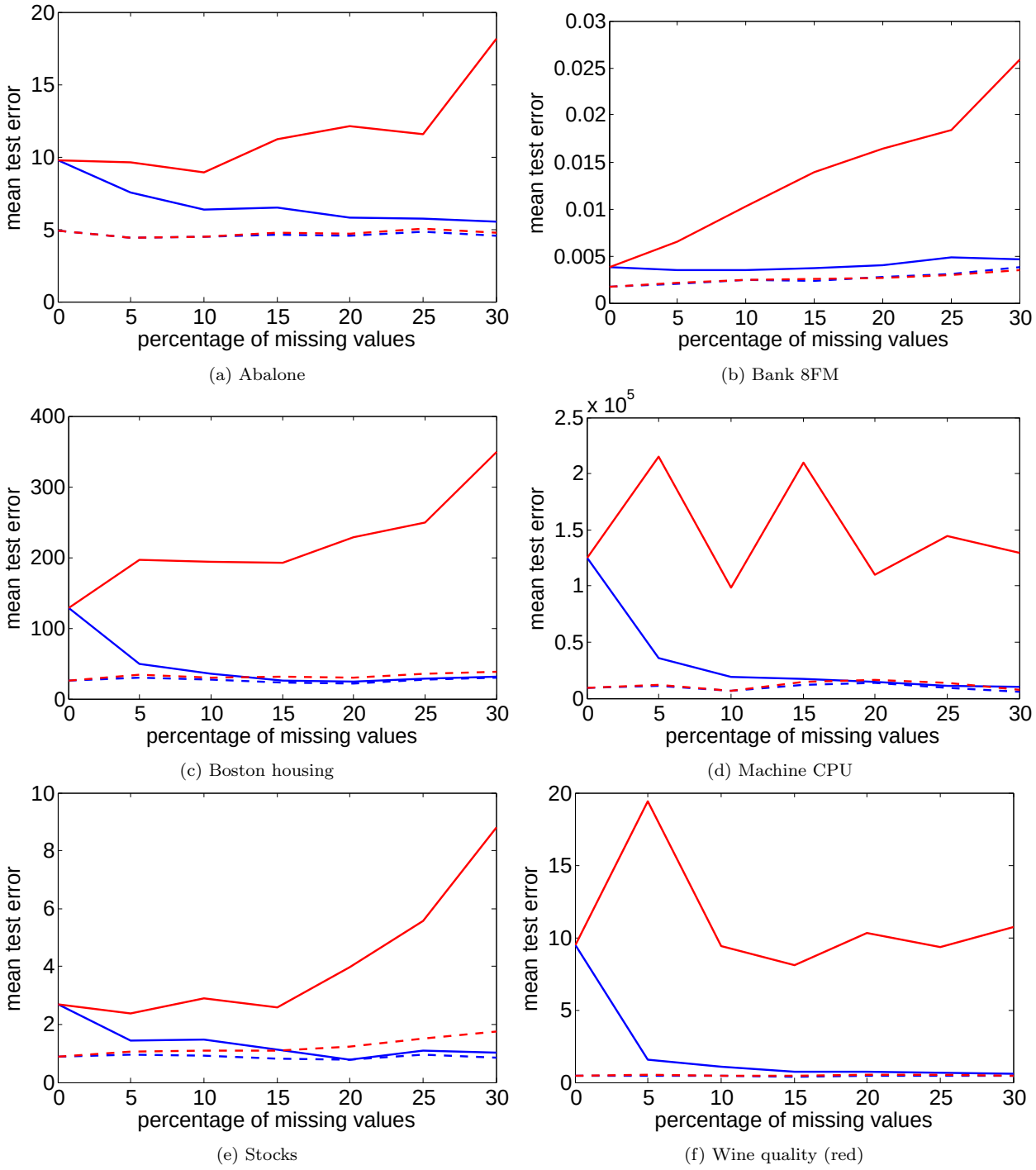
for CM and MI strategies respectively. Figure 3 shows that training multiple models is definitely beneficial compared to a single initialised ELM. In both cases, ensembling on either multiple imputed data sets or on a date set with conditional mean imputation provides better predictions than an approach with single initialisation. For Abalone, it is interesting to see that MI strategy with a single initialised ELM is better than ensemble with CM strategy. The results shown are for the models with maximum complexity without any regularisation.

Finally, Figure 4 shows how the most complex network performs compared to the best performing model on a test set (following the same reasoning as in single initialisation case). In several cases, the ensemble of the most complex networks is able to reach the performance of the "optimal" ensemble (Boston housing, Machine CPU and Stocks), while remaining competitive for Abalone and Wine quality data. For Bank data, there is a large gap between the proposed approach and the best performing result. However, the result stay close to the ensemble of the most complex networks for mean imputed data. It is important to stress that these ensembles on the most complex networks do not require any regularisation steps, circumventing the tedious and long validation process.

6. Conclusion

In this paper, the task of accurate prediction is tackled on a data containing missing values. The complete methodology consists of four steps with simple and known methods. Mixture of Gaussians is employed to model the underlying distribution of the data, while Extreme Learning Machine enables multiple imputation approach to be executed on a reasonable scale. Adjustments for Expectation-Maximisation algorithm for Gaussian Mixture Model are given in order to tackle the missing values in the data, alongside the required updates for conditional Gaussian Mixture Model needed to sample new data sets.

The combination of GMM and ELM allows adopting the multiple imputation approach to missing data, which has shown to be superior in almost all tested cases over the method based on conditional mean imputation. Having a distribution to reflect uncertainty in the data due to missing values can be beneficial over simple ad hoc methods, but can still be severely biased as the experiments have shown. In order to ensure stable and reliable predictions, a sufficient number of draws is required to properly represent the underlying data distribution. In such scenario with potentially high number of data sets, ELM is a suitable model due to its fast training times and good generalisation properties.



(a) Abalone

(b) Bank 8FM

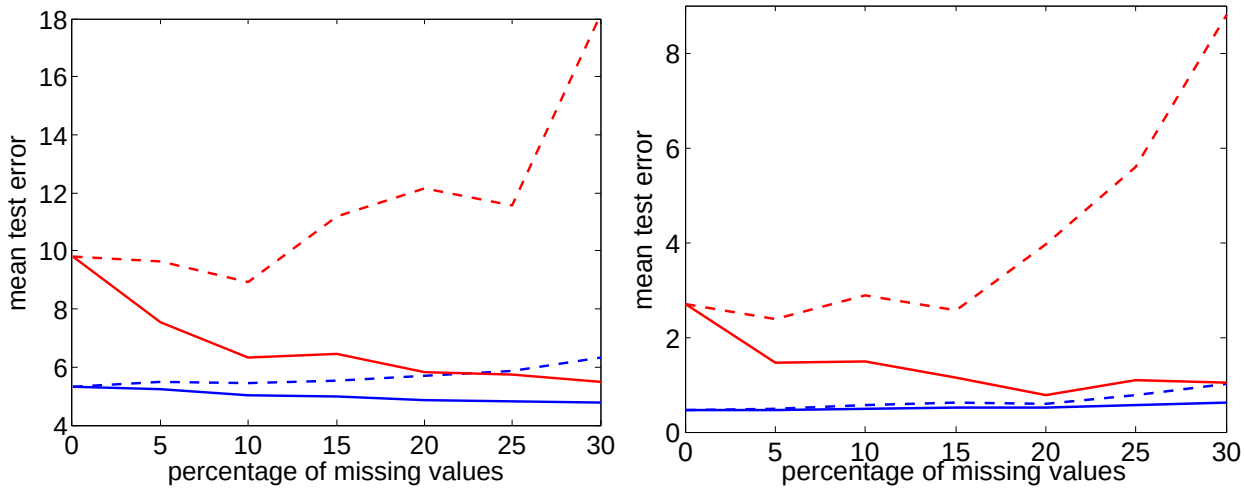
(c) Boston housing

(d) Machine CPU

(e) Stocks

(f) Wine quality (red)

Figure 2: Average test mean squared error for all tested data sets with respect to the number of missing values. Blue colour is MI strategy and red colour represents CM strategy. Solid lines represent the most complex networks while dashed lines are the best performing networks on a test set.



(a) Abalone (b) Stocks

Figure 3: Average test mean squared error for Abalone and Stocks data sets. Blue colour indicates ensembling approach and red colour single initialisation of ELM. Solid lines represent MI strategy while dashed lines are CM strategy. The graphs correspond to the most complex networks.

The disadvantage of applying the multiple imputation procedure is a notable increase in computational time. This can be seen as a trade-off between time and accuracy compared to alternative methods of handling the missing values. Ignoring all incomplete samples is a poor solution, and the difference in accuracy is considerable already at low proportions of missing values, as shown by the experiments. For larger fractions of missing data, it is necessary to apply an appropriate procedure which avoids discarding partially known samples, and multiple imputation provides a practical approach.

An interesting side-effect of having more missing values in the data is the potential removal of model structure selection procedure. The models with the highest complexity have competitive results with the optimal models on majority of data sets. This suggests that validation procedures based on training/validation errors can simply be replaced by a model with enough neurons in the hidden layer of the ELM.

References

[1] A. R. T. Donders, G. J. van der Heijden, T. Stijnen, K. G. Moons, Review: A gentle introduction to imputation of missing values, Journal of Clinical Epidemiology 59 (10) (2006) 1087–1091.

[2] A. Sorjamaa, A. Lendasse, Y. Cornet, E. Deleersnijder, An improved methodology for filling missing values in spatiotemporal climate data set, Computational Geosciences 14 (2010) 55–64.

[3] A. N. Baraldi, C. K. Enders, An introduction to modern missing data analyses, Journal of School Psychology 48 (1) (2010) 5–37.

[4] P. D. Allison, Missing data: Quantitative applications in the social sciences, British Journal of Mathematical and Statistical Psychology 55 (1) (2002) 193–196.

[5] R. J. A. Little, D. B. Rubin, Statistical Analysis with Missing Data, 2nd Edition, Wiley-Interscience, 2002.

[6] D. B. Rubin, Multiple Imputation for Nonresponse in Surveys, Wiley, 1987.

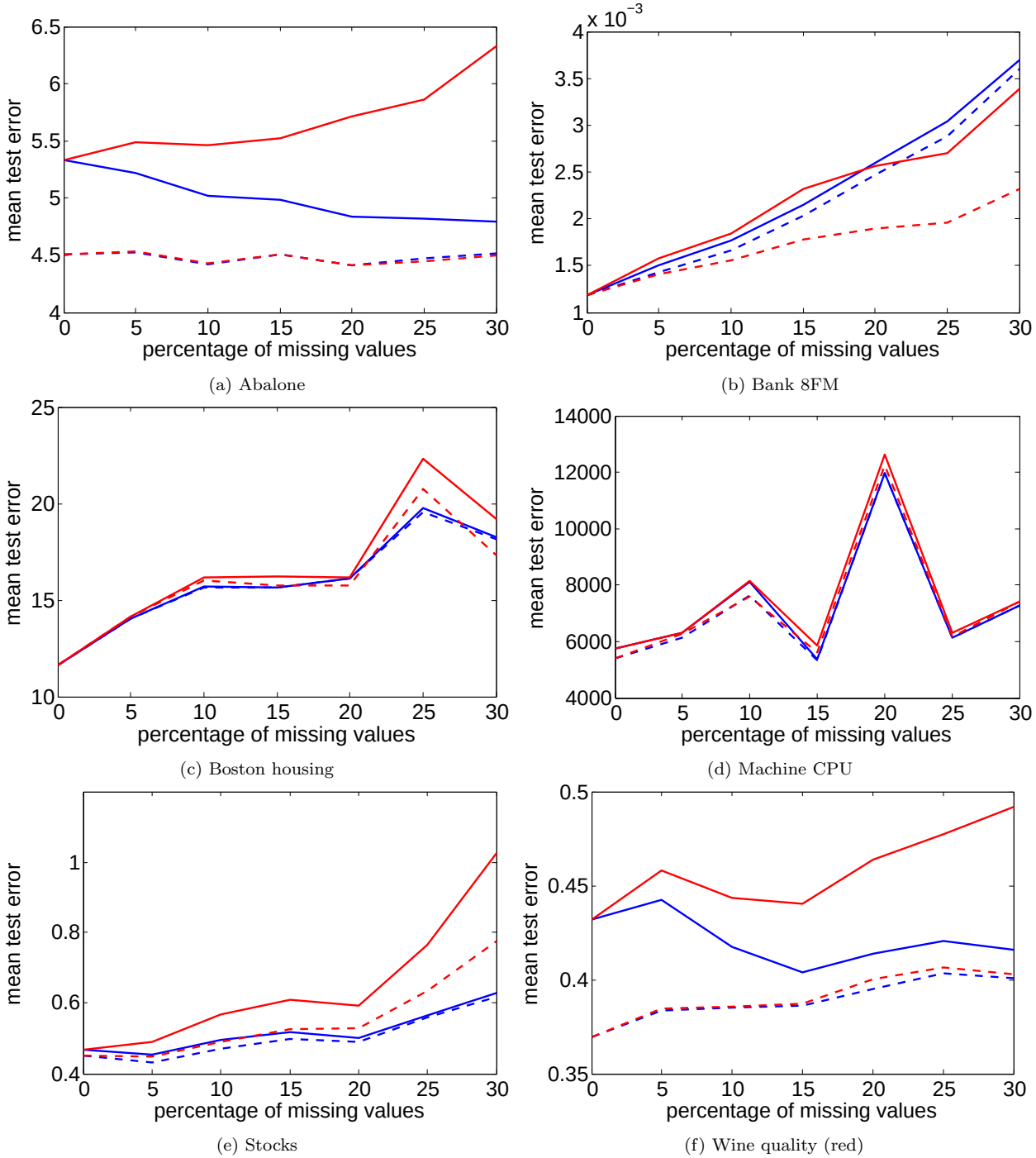
[7] C. K. Enders, Applied Missing Data Analysis, Methodology in the Social Sciences, Guilford Press, 2010.

[8] E. R. Hruschka, E. R. Hruschka Jr., N. F. F. Ebecken, Evaluating a nearest-neighbor method to substitute continuous missing values, in: AI 2003: Advances in Artificial Intelligence, Vol. 2903 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2003, pp. 723–734.

[9] J. Van Hulse, T. M. Khoshgoftaar, Incomplete-case nearest neighbor imputation in software measurement data, in: Proceedings of 2007 IEEE International Conference on Information Reuse and Integration (IRI 2007), Las Vegas, NV, USA, 2007, pp. 630–637.

[10] S. Van Buuren, J. P. Brand, C. G. Groothuis-Oudshoorn, D. B. Rubin, Fully conditional specification in multivariate imputation, Journal of Statistical Computation and Simulation 76 (12) (2006) 1049–1064.

[11] D. Titterington, A. Smith, U. Makov, Statistical Analysis of Finite Mixture Distributions, Wiley, New York, 1985.



(a) Abalone

(b) Bank 8FM

(c) Boston housing

(d) Machine CPU

(e) Stocks

(f) Wine quality (red)

Figure 4: Average test mean squared error for all tested data sets with respect to the number of missing values. Blue colour is MI strategy and red colour represent CM strategy. Solid lines represent the most complex networks while dashed lines are the best performing networks on a test set. The results are for the ensemble on 1000 trained models.

[12] Z. Zivkovic, Improved adaptive gaussian mixture model for background subtraction, in: Proceedings of 17th International Conference on Pattern Recognition (ICPR 2004), Vol. 2, Cambridge, UK, 2004, pp. 28–31.
[13] P. KaewTraKulPong, R. Bowden, An improved adaptive background mixture model for real-time tracking with shadow detection, in: Video-Based Surveillance Systems, Springer US, 2002, pp. 135–144.
[14] P. A. Torres-Carrasquillo, D. A. Reynolds, J. Deller Jr, Language identification using gaussian mixture model tokenization, in: Proceedings of 2002 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2002), Vol. 1, Orlando, FL, USA, 2002, pp. 757–760.
[15] D. A. Reynolds, R. C. Rose, Robust text-independent speaker identification using gaussian mixture speaker models, IEEE Transactions on Speech and Audio Processing 3 (1) (1995) 72–83.
[16] D. Povey, L. Burget, M. Agarwal, P. Akyazi, F. Kai, A. Ghoshal, O. Glembek, N. Goel, M. Karafiát, A. Rastrow, et al., The subspace gaussian mixture model: A structured model for speech recognition, Computer Speech & Language 25 (2) (2011) 404–439.
[17] H. Greenspan, A. Ruf, J. Goldberger, Constrained gaussian mixture model framework for automatic segmentation of mr brain images, IEEE Transactions on Medical Imaging 25 (9) (2006) 1233–1245.
[18] M. Ait Kerroum, A. Hammouch, D. Aboutajdine, Textural feature selection by joint mutual information based on gaussian mixture model for multispectral image classification, Pattern Recognition Letters 31 (10) (2010) 1168–1174.
[19] A. P. Dempster, N. M. Laird, D. B. Rubin, Maximum likelihood from incomplete data via the em algorithm, Journal of the Royal Statistical Society, Series B 39 (1) (1977) 1–37.
[20] Z. Ghahramani, M. Jordan, Learning from incomplete data, Tech. rep., Lab Memo No. 1509, CBCL Paper No. 108, MIT AI Lab (1995).
[21] L. Hunt, M. Jorgensen, Mixture model clustering for mixed data with missing information, Computational Statistics & Data Analysis 41 (3–4) (2003) 429–440.
[22] T. I. Lin, J. C. Lee, H. J. Ho, On fast supervised learning for normal mixture models with missing information, Pattern Recognition 39 (6) (2006) 1177–1187.
[23] Inference from multiple imputation for missing data using mixtures of normals, Statistical Methodology 7 (3) (2010) 351–365.
[24] O. Delalleau, A. C. Courville, Y. Bengio, Efficient em training of gaussian mixtures with missing data, CoRR abs/1209.0521, http://arxiv.org/abs/1209.0521.
[25] V. Tresp, S. Ahmad, R. Neuneier, Training neural networks with deficient data, in: Proceedings of 7th Conference on Neural Information Processing Systems (NIPS 1993), Vol. 6 of Advances in Neural Information Processing Systems, Pasadena, CA, USA, 1993, pp. 128–135.
[26] E. Eirola, A. Lendasse, V. Vandewalle, C. Biernacki, Mixture of gaussians for distance estimation with missing data, Neurocomputing 131 (2014) 32–42.
[27] Q. Yu, Y. Miche, E. Eirola, M. van Heeswijk, E. Sverin, A. Lendasse, Regularized extreme learning machine for regression with missing data, Neurocomputing 102 (2013) 45–51.
[28] H. Akaike, A new look at the statistical model identification, IEEE Transactions on Automatic Control 19 (6) (1974) 716–723.
[29] G. Schwarz, Estimating the dimension of a model, The Annals of Statistics 6 (2) (1978) 461–464.
[30] C. M. Hurvich, C.-L. Tsai, Regression and time series model selection in small samples, Biometrika 76 (2) (1989) 297–307.
[31] Y. Miche, E. Eirola, P. Bas, O. Simula, C. Jutten, A. Lendasse, M. Verleysen, Ensemble modeling with a constrained linear system of leave-one-out outputs, in: Proceedings of 18th European Symposium on Artificial Neural Networks (ESANN 2010), Computational Intelligence and Machine Learning, Bruges, Belgium, 2010, pp. 19–24.
[32] M. van Heeswijk, Y. Miche, E. Oja, A. Lendasse, Gpu-accelerated and parallelized elm ensembles for large-scale regression, Neurocomputing 74 (16) (2011) 2430–2437.
[33] M. van Heeswijk, Y. Miche, T. Lindh-Knuutila, P. Hilbers, T. Honkela, E. Oja, A. Lendasse, Adaptive ensemble models of extreme learning machines for time series prediction, in: Proceedings of 19th International Conference on Artificial Neural Networks (ICANN 2009), Vol. 5769 of Lecture Notes in Computer Science, Cyprus, 2009, pp. 305–314.
[34] D. Sovilj, A. Lendasse, O. Simula, Extending extreme learning machine with combination layer, in: Proceedings of 12th International Work-Conference on Artificial Neural Networks (IWANN 2013), Vol. 7902 of Lecture Notes in Computer Science, Puerto de la Cruz, Tenerife, Spain, 2013, pp. 417–426.
[35] L. Breiman, Stacked regressions, Machine Learning 24 (1) (1996) 49–64.
[36] D. Draper, Assessment and propagation of model uncertainty, Journal of the Royal Statistical Society, Series B 57 (1) (1995) 45–97.
[37] T. W. Anderson, An Introduction to Multivariate Statistical Analysis, 3rd Edition, Wiley-Interscience, 2003.
[38] C. Bouveyron, S. Girard, C. Schmid, High-dimensional data clustering, Computational Statistics & Data Analysis 52 (1) (2007) 502–519.
[39] G.-B. Huang, Q.-Y. Zhu, C.-K. Siew, Extreme learning machine: A new learning scheme of feedforward neural networks, in: Proceedings of 2004 IEEE International Joint Conference on Neural Networks (IJCNN 2004), Vol. 2, Budapest, Hungary, 2004, pp. 985–990.
[40] G.-B. Huang, An insight into extreme learning machines: Random neurons, random features and kernels, Cognitive Computation 6 (3) (2014) 376–390.
[41] A. Sorjamaa, Y. Miche, R. Weiss, A. Lendasse, Long-term prediction of time series using nne-based projection and op-elm, in: Proceedings of 2008 IEEE World Conference on Computational Intelligence (WCCI 2008), Hong Kong, 2008, pp. 2675–2681.
[42] C. Cheng, W. P. Tay, G.-B. Huang, Extreme learning machines for intrusion detection, in: Proceedings of 2012 Interna-

tional Joint Conference on Neural Networks (IJCNN 2012), Brisbane, Australia, 2012, pp. 1–8.
[43] R. Minhas, A. Baradarani, S. Seifzadeh, Q. M. Jonathan Wu, Human action recognition using extreme learning machine based on visual vocabularies, Neurocomputing 73 (10–12) (2010) 1906–1917.
[44] F. Chen, T. Ou, Sales forecasting system based on gray extreme learning machine with taguchi method in retail industry, Expert Systems with Applications 38 (3) (2011) 1336–1345.
[45] K. Hornik, M. Stinchcombe, H. White, Multilayer feedforward networks are universal approximators, Neural Networks 2 (5) (1989) 359–366.
[46] D. Rumelhart, G. Hinton, R. Williams, Learning representations by back-propagation errors, Nature 323 (1986) 533–536.
[47] Z. Q.-Y. Huang, G.-B., C.-K. Siew, Extreme learning machine: Theory and applications, Neurocomputing 70 (2006) 489–501.
[48] Z. Q.-Y. Huang, G.-B., C.-K. Siew, Universal approximation using incremental constructive feedforward networks with random hidden nodes, IEEE Transactions on Neural Networks 17 (4) (2006) 879–892.
[49] Y. Miche, A. Sorjamaa, P. Bas, O. Simula, C. Jutten, A. Lendasse, Op-elm: Optimally-pruned extreme learning machine, IEEE Transactions on Neural Networks 21 (1) (2010) 158–162.
[50] A. Hoerl, R. Kennard, Ridge regressoin: Biased estimate for nonorthogonal problems, Technometrics 12 (1) (1970) 55–67.
[51] Y. Lan, Y. C. Soh, G.-B. Huang, Constructive hidden nodes selection of extreme learning machine for regression, Neurocomputing 73 (16–18) (2010) 3191–3199.
[52] A. Frank, A. Asuncion, UCI machine learning repository, `http://archive.ics.uci.edu/ml` (2012).
[53] L. Torgo, LIACC regression data sets, `http://www.dcc.fc.up.pt/~ltorgo/Regression/DataSets.html`.