Knowledge-Based Systems 163 (2019) 830–841

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Multi-step forecasting for big data time series based on ensemble learning

A. Galicia [a], R. Talavera-Llames [a], A. Troncoso [a,*], I. Koprinska [b], F. Martínez-Álvarez [a]

[a] Data Science & Big Data Lab, Universidad Pablo de Olavide, ES-41013 Seville, Spain
[b] School of Information Technologies, University of Sydney, Sydney, NSW, Australia

ARTICLE INFO

Article history:
Received 15 March 2018
Received in revised form 4 October 2018
Accepted 5 October 2018
Available online 12 October 2018

Keywords:
Big data
Ensemble
Electricity time series
Forecasting

ABSTRACT

This paper presents ensemble models for forecasting big data time series. An ensemble composed of three methods (decision tree, gradient boosted trees and random forest) is proposed due to the good results these methods have achieved in previous big data applications. The weights of the ensemble are computed by a weighted least square method. Two strategies related to the weight update are considered, leading to a static or dynamic ensemble model. The predictions for each ensemble member are obtained by dividing the forecasting problem into h forecasting sub-problems, one for each value of the prediction horizon. These sub-problems have been solved using machine learning algorithms from the big data engine Apache Spark, ensuring the scalability of our methodology. The performance of the proposed ensemble models is evaluated on Spanish electricity consumption data for 10 years measured with a 10-minute frequency. The results showed that both the dynamic and static ensembles performed well, outperforming the individual ensemble members they combine. The dynamic ensemble was the most accurate model achieving a MRE of 2%, which is a very promising result for the prediction of big time series. Proposed ensembles are also evaluated using solar power from Australia for two years measured with 30-min frequency. The results are successfully compared with Artificial Neural Network, Pattern Sequence-based Forecasting and Deep Learning, improving their results.

1. Introduction

Advances in technology have led to an increasing generation and storage of massive data in recent years [1,2]. These data need to be efficiently processed in order to extract useful and valuable knowledge. Thus, the development of new tools for dealing with big data has become a critical issue. An essential component of the nature of big data is that information is usually captured over time at different points, resulting in big data time series [3]. This information can be analysed for various purposes: to predict the future values, to establish relations among variables, to detect anomalous values, or to discover patterns.

The main existing frameworks for the massive data processing have been developed thanks to leading technology companies. For example, the MapReduce technology was developed by Google [4]; it divides the input data into small blocks, processes them and then aggregates the output into a single solution. Based on this paradigm, Yahoo! developed an open-source implementation called Hadoop [5] and later Spark was developed by the University of Berkeley in California [6].

Spark maximizes parallelization of data processing in-memory, achieving much faster processing speed than Hadoop. Spark also has specific modules for mining big data, such as the Apache Spark's Machine Learning Library (MLlib) [7]. Although its native language is Scala, it also supports Python, R and Java. Spark is seen as a standard framework for data-intensive computing, and is being used in a wide range of problems such as climate data [8], water system [9], earthquakes [10], entity matching for information integration and data cleansing [11] or energy data in buildings [12]. In addition, studies of the performance of the Spark system are shown in [13] and [14].

This study is framed in the time series forecasting context, with arbitrary time horizon and in a big data environment. The previous work in [15] assessed the performance of MLlib for the forecasting of big data time series. A set of scalable algorithms was studied and adapted for very large time series forecasting. In particular, representative methods of different nature, such as linear regression, decision trees, gradient boosted trees and random forest were analysed. The results reported were promising and, for this reason, we now explore the suitability of combining some of these algorithms into ensembles to forecast big data time series.

* Corresponding author.
E-mail addresses: agalde@alu.upo.es (A. Galicia), rltallla@upo.es (R. Talavera-Llames), atrolor@upo.es (A. Troncoso), irena.koprinska@sydney.edu.au (I. Koprinska), fmaralv@upo.es (F. Martínez-Álvarez).

https://doi.org/10.1016/j.knosys.2018.10.009
0950-7051/

In particular, three of the aforementioned methods (decision trees, gradient boosted trees and random forest) have been used to develop a novel ensemble forecasting algorithm. Linear regression has been discarded because its performance, although acceptable in general terms, was too poor when compared to the other methods. The ensemble approach assigns different weights to every method by using a weighted square least method, which optimizes the contribution of each individual forecast in the combined forecast for a given forecasting time. Therefore, our ensemble is not a typical boosting ensemble, but a weighed voting ensemble as we combine three base models by using a weighed majority vote, where the weights are calculated based on the previous performance of these models using a least squares method. We propose two different strategies for training the ensemble models: static, in which the training set remains the same for each target sample to be forecasted, and dynamic, in which the training set slides forwards and changes for every target sample to be forecasted. The performance of the proposed ensemble algorithm has been evaluated using electricity consumption data from Spain. The ensemble outperforms all three methods it combines when they are used individually. In particular, the dynamic ensemble was the best performing model achieving mean relative errors of about 2%, which is a very competitive results [16].

An important feature of the proposed approach is its ability to deal with prediction horizons of arbitrary length. In this sense, if the prediction horizon is composed of h samples, h independent models are generated and simultaneously processed.

Although we used the MLlib implementations of the three base prediction methods, there were some limitations that we had to overcome. On one hand, the regression techniques available in MLlib do not support more than one step ahead prediction. On the other hand, the RDD (Resilient Distributed Dataset) data structure used by Spark is not designed to guarantee the order of data, which is a critical aspect in time series processing . The handling of these two key limitations is another contribution of this work.

In summary, the main contributions of this work are:

(1) We propose a weighted voting ensemble that uses a least squares method to calculate the weights of the base models.

(2) We develop two versions of this ensemble, static and dynamic.

(3) We show how this ensemble can be evaluated on big data for multi-step ahead forecasting by decomposing the task into multiple prediction problems, which can be solved by the Apache Spark big data engine.

(4) We conduct a comprehensive evaluation using Spanish electricity data for 10 years, measured at 10-min intervals, demonstrating that both ensemble members performed, outperforming the base models they combine, and particularly showing the potential of dynamic ensembles for big data forecasting.

(5) Solar photovoltaic dataset from Australia has been used to compare proposed ensembles with algorithms of different nature, such us deep learning, pattern sequence-based forecasting and artificial neural networks.

Section 2 reviews the literature related to time series forecasting techniques, machine learning for big data and ensemble learning. A theoretical background is included in Section 3, where the multi-step methodology is proposed. Section 4 presents and discusses the results of Spanish electricity data. Section 5 discusses and compares the results of Australian solar data . Finally, Section 6 summarizes the main conclusions.

2. Related work

This section discusses the most relevant related works, focusing on big data. Although short and medium term time series forecasting have been extensively studied in the literature, there are very few works on time series forecasting for big data.

In general, the methods for predicting time series can be classified into classical methods based on Box and Jenkins [17], such as ARIMA and GARCH; and data mining methods, such as Support Vector Machine (SVM), k Nearest Neighbour techniques (kNN) and Artificial Neural Networks (ANN). For a taxonomy of these techniques applied to energy time series forecasting, the reader is referred to [16].

However, due to the high computational cost, the majority of the data mining techniques cannot be applied when big data have to be processed. Therefore, big data mining techniques [18,19] are being developed for distributed computing in order to solve typical tasks as classification, clustering or regression. A brief description of the main advances in this area is given below.

Several MapReduce-based approaches for big data scenarios have been recently provided for classification tasks. The SVM algorithm was recently adapted to the field of high performance computing giving rise to parallel SVMs [20]. Based on Spark, several parallel implementations of the kNN algorithm have been proposed in [21–23]. Also, a MapReduce-based framework focused on instance reduction methods was proposed in [24] to reduce the computational cost and storage requirements of kNN. Deep learning has been also used for industry process planning in [25].

In recent years, increased attention has been paid to big data clustering. A survey on this topic can be found in [26,27]. In the particular field of big data time series, K-means has been successfully applied in [28]. Likewise, the challenging task of determining the optimal number of partitions has been addressed in [29], where scalable approaches to determine the quality of the generated partitions using clustering were proposed.

Very few works have been published on using big data for regression tasks, hence there is much room for improvement. A survey on big data forecasting is presented in [30]. Some regression algorithms based on cloud and big data technologies have been used for earthquake prediction in California [31]. An approach based on kNN to forecast big data time series was introduced in 2016 in [22] and approaches based on deep learning have also been published in 2017 [32]. A scalable fuzzy system for regression is proposed in [33], as the performance of the fuzzy rules depends on the size of the problem.

A variety of ensemble methods have been proposed and successfully used in practical applications [34]. The field of ensembles was developed to improve the accuracy of an automated decision-making system, with the aim of reducing variance. Since then, ensembles have been widely used in different machine learning problems, for prediction and classification, feature selection, missing feature, incremental learning, confidence estimation, error correction, among others.

Polikar [35] provided an overview of ensembles, their properties and how they can be applied to different tasks. Ensemble learning is being used for streaming analysis, a survey can be found in [36]. Ensemble techniques based on threes are the most recurrent topic in the literature for big data. This is mainly due to the easy adaptation of these algorithms for distributed computing. Random Forest has been applied to some specific problems, showing good performance for large datasets [37]. Analogously, regression trees have been constructed using parallel learning with MapReduce technology in a cluster [38].

Hadoop and its machine learning library Mahout have been selected for classification tasks using Random Forest in [39]. Meta classifiers combined automatically in an iterative way have been proposed for the detection of malware in [40]. A classifier ensemble algorithm for multimedia classification was proposed in [41], where a decision tree was used to combine the predictions of the individual ensemble members. However, an extensive analysis of the literature reveals that these methods have not been applied to the prediction of big data time series, and therefore, the work here introduced attempts at filling this gap.

Recently, to alleviate some of the issues associated with big data, Do and Poulet [42] developed a parallel ensemble learning algorithm of random local SVM that was able to perform much better than the standard SVM algorithm.

As a large number of resources is necessary to generate the ensemble, in [43] a new ensemble method that minimizes the usage of resources was proposed. Learning a decision multi-tree instead of a decision tree allows to share the common parts of the components of the ensemble and build a shared ensemble.

After a thorough review of the previously published works, it can be concluded that forecasting of big data time series is an emerging topic that should be further investigated. In particular, very few papers have been published using distributed and highly scalable computing systems. Additionally, not many ensemble methods for big data have been proposed and none of them made use of the Spark benefits. In this paper we aim to address these limitations by proposing and investigating the performance of ensemble method for big data forecasting, that makes use of the Spark engine.

In the next section we describe our methodology for forecasting big data time series using static and dynamic ensemble models.

3. Methodology

In this section, we firstly introduce the three regression methods that we use to generate prediction models for big data time series. Then, we present the proposed ensemble model which combines the individual tree-based regression models to further improve the prediction accuracy. For all regression methods we use their MLlib library implementation, to ensure the scalability of the ensemble model, and therefore, its ability to deal with big data time series.

3.1. Decision tree

Decision Trees (DTs) are a very popular and successful machine learning method, for both classification and regression tasks. Some of the advantages they offer are the following: they are able to model nonlinear relations between the attributes and the target variable, do not require attribute scaling, and the resulting tree (a set of if-then rules) is easy to interpret and use for decision making by the end-user.

A DT is built through a recursive binary partition of the feature space. A node in the tree corresponds to a test for the value of an attribute and a leaf node corresponds to a regression function. When building the tree, at each step the attribute with the highest information gain is selected. A test for its value is used to split the tree, creating a branch for the possible outcomes. The test divides the instances into several subsets, based on the attribute value. The process is repeated recursively for each subset until the stopping condition is satisfied, in which case a leaf node is created. The tree growing stops when there is no split candidate with sufficiently high information gain or when a pre-specified maximum tree depth is reached.

To predict the value for a new instance, we start at the root of the tree and follow the path corresponding to the values of the instance until a leaf node is reached and the prediction is obtained.

MLlib supports DTs for both classification and regression, and can be used with both continuous and discrete attributes.

3.2. Gradient boosted trees

Gradient Boosted Trees (GBT) is an ensemble of DTs. An ensemble method combines the predictions of a set of base models. DT-based ensembles such as GBT have showed high performance in regression and classification tasks [44,45]. GBT creates the tree ensemble iteratively. Thus, the errors made by the first tree are taken into account when adding the second tree and so on. The final prediction is the mean of the predictions of the individual trees.

3.3. Random forests

Random Forest (RF) is also an ensemble of DTs. In contrast to GBT, where the trees are built iteratively, RF trains multiple trees in parallel. Each tree uses a different training set generated by creating a bootstrap sample from the training data. In addition, when selecting the best attribute at each node, only a subset of all attributes available at the node will be considered, instead of all features as in DTs. Thus, RF uses both random instance and feature selection. To make a prediction for a new instance, RF takes the average of the predictions obtained from each tree.

In summary, both GBT and RF use decision trees as a base model, but the training process is different and each of the two methods has its advantages and disadvantages. MLlib supports both GBT and RF.

3.4. Proposed ensemble model

Ensembles of prediction models are one of the most successful methods used in practical applications [34]. An ensemble usually improves the results of the single base models it combines. In this paper we focus on ensembles for time series forecasting, and especially for big data. We propose to combine DT, GBT and RF in an ensemble, due to the good results obtained by each of these algorithms when applied to big data time series forecasting [15].

Instead of combining the predictions by taking the average of the individual predictions, which is the most simple and straightforward combination, we propose to use a weighted average combination, where different weights for each algorithm are computed based on its previous performance.

To calculate the coefficients for the weighted prediction, we minimize the forecasting error on a validation set. Let K be the number of algorithms that form the ensemble model. Let us suppose that the validation set is composed of N instances and h is the prediction horizon. Then, a weighted least squares method is applied to minimize the squared error between the predictions of the K algorithms and the actual values for the N instances of the validation set. Thus, the weights that minimize the difference between the predicted and actual values are obtained by solving the following equation for each jth value of the prediction horizon:

$$\widehat{P}^j \alpha^j = b^j \tag{1}$$

where $\widehat{P}^j$ is a matrix with N rows and K columns containing the prediction for the jth value of the prediction horizon for the validation set for each algorithm, α^j is a vector of K elements corresponding to the weights for the jth value of the prediction horizon for each algorithm, and b^j is a vector composed of the N actual values for the jth value of the prediction horizon for the validation set. The predictions $\widehat{P}^j$ for each algorithm are computed following the methodology described in the Section 3.5.

Once the weights have been obtained by the weighted square least method, we use them to make predictions for the test set. Let us suppose the test set is composed of M instances. Then, the Mxh predictions represented by the matrix $\widehat{P}$ are computed by a linear combination of the predictions for the K algorithms, where the coefficients of the linear function are given by the weights:

$$\widehat{P} = \left[\widehat{Q}^1\alpha^1, \ldots, \widehat{Q}^h\alpha^h\right] \tag{2}$$

where $\widehat{Q}^j$ is a matrix with M rows and K columns containing the prediction for the jth value of the prediction horizon for the test set for each algorithm and α^j are the weights obtained by Eq. (1). Thus, $\widehat{Q}^j\alpha^j$ is a vector of M elements containing the predictions obtained by the ensemble model for the jth value of the prediction horizon for the test set.

From Eq. (2) it also follows that:

$$\widehat{p}_{i,j} = \sum_{l=1}^{K} \alpha_l^j \widehat{p}_{i,l}^j \tag{3}$$

where $\widehat{p}_{i,j}$ is the (i, j)-th element of the matrix $\widehat{P}$, $\widehat{p}_{i,l}^j$ is the (i, l)-th element of the matrix $\widehat{Q}^j$ and α_l^j is the lth element of the vector α^j.

Both Eqs. (2) and (3) represent a static ensemble combination, i.e. the same weights are used to predict all Mxh values of the test set. However, different adaptive strategies can be used to update the weights after a given time interval, and thus to create dynamic ensembles.

Fig. 1 shows a general diagram of the static ensemble model described above to predict a big data time series. Note that the weights are defined by a matrix due to a multi-step prediction problem that we address in this paper.

In the dynamic ensemble model, let R be the updating period for the weights. Then the test set TS can be divided into subsets as follows:

$$TS = \bigcup_{t=1}^{R} TS^t \tag{4}$$

where the subset TS^t is composed of [M/R] instances and $[\cdot]$ denotes the whole number part of the division. Then, the weights are found by solving the following equation:

$$\widehat{P}^j(t)\alpha^j(t) = b^j(t) \quad t = 1, \ldots, R \tag{5}$$

where $P^j(t)$ and $b^j(t)$ are the predictions and actual values for the jth value of the prediction horizon respectively, using the validation set from the training set updated as follows:

$$TRS \leftarrow TRS^t \bigcup \left(\bigcup_{l=1}^{t-1} TS^l \right) \quad t = 1, \ldots, R \tag{6}$$

where TRS^t is the training set by removing the oldest $(t-1)[M/R]$ instances. Note that $(t-1)[M/R]$ is the size of the $\bigcup_{l=1}^{t-1} TS^l$, and therefore, the updating of the training set consists of sliding the training set $(t-1)[M/R]$ instances forward while keeping the same training set size.

Once the weights have been obtained by the weighted square least method from Eq. (5), the predictions for each subset TS^t are computed as:

$$\widehat{P}(t) = \left[\widehat{Q}^1(t)\alpha^1(t), \ldots, \widehat{Q}^h(t)\alpha^h(t) \right] \quad t = 1, \ldots, R \tag{7}$$

where $\widehat{Q}^j(t)$ contains the prediction for the jth value of the prediction horizon for the TS^t test subset for each algorithm and $\alpha^j(t)$ are the weights obtained by Eq. (5).

Fig. 2 presents graphically how the dynamic ensemble model works when the weights are periodically updated, in our case, every 3 months.

3.5. Multi-step forecasting

This section summarizes the forecasting methodology proposed in our previous work [15] for h-steps ahead prediction for big data time series using the MLlib library of Apache Spark.

Problem formulation. Given a time series with previous values up to time t, $[x_1, \ldots, x_t]$, the task is to predict the h next values of the time series, from a window of w past values, as shown in Fig. 3.

This forecasting problem can be formulated as below, where f is the model to be learnt by the forecasting method in the training phase:

$$[x_{t+1}, x_{t+2}, \ldots, x_{t+h}] = f\left(x_t, x_{t-1}, \ldots, x_{t-(w-1)}\right) \tag{8}$$

However, the existing regression techniques in MLlib do not support this multi-step forecasting. Therefore, we split the problem into h forecasting sub-problems as follows:

$$\begin{aligned} x_{t+1} &= f_1\left(x_t, x_{t-1}, \ldots, x_{t-(w-1)}\right) \\ x_{t+2} &= f_2\left(x_t, x_{t-1}, \ldots, x_{t-(w-1)}\right) \\ \vdots &= \qquad\qquad \vdots \\ x_{t+h} &= f_h\left(x_t, x_{t-1}, \ldots, x_{t-(w-1)}\right) \end{aligned} \tag{9}$$

Hence, in each sub-problem we use the same input data with size w but predict a different target value from the forecasting window h. We note that using this formulation, the existing possible relations between the h consecutive values $x_{t+1}, \ldots, x_{t+h}$ are not taken into consideration. An alternative approach would be to use a rolling forecasting where the predictions of the previous values are used as actual values when predicting the next value, e.g. after predicting x_{t+1}, it will be included in the w values used to predict x_{t+2}. However, we found that the rolling forecasting method was less accurate due to the accumulation of the error along the prediction horizon. Hence, we chose the first approach.

We build h different training sets. Each training instance is composed of the w features. The target value for each of the h problems corresponds to a different value of the prediction horizon, as shown in Fig. 3. Thus, we learn h prediction models.

To predict a new instance from the test data, the prediction of the ith value of the prediction horizon is obtained by using the ith model with the corresponding w features from the test set as an input.

This methodology was tested in [15] using four different regression methods from MLlib (linear regression, DT, GBT and RF) demonstrating the suitability of these methods for big data time series forecasting.

In the next section we evaluate the performance of the proposed ensemble models on Spanish electricity consumption data for 10 years measured with a 10-minute frequency.

4. Results for electricity consumption data

In this section, we present and discuss the application of our proposed method for prediction of big electricity consumption time series data. We firstly describe the electricity consumption dataset in Section 4.1. The experimental setting is presented in Section 4.2 and the sensitivity analysis used to select an adequate historical window size is provided in Section 4.3. We present and analyse the results obtained by the different static and dynamic ensemble methods in Section 4.4.

4.1. Dataset description

The time series used in this work is related to the total electrical energy consumption in Spain, from January 1st 2007 at midnight to June 21st 2016 at 11:40 pm. In short, it is a time series of nine and a half years with a high sampling frequency, namely 10 min intervals, including 49,832 measurements in total.

When using the proposed methodology with a prediction horizon of 4 h (h is set to 24 values), the dataset consists of 20,742 instances and 144 attributes, corresponding to 5.70 MiB of storage size. These 144 attributes correspond to a window w of 144 past values (24 h).

For the static ensemble, this dataset is divided into a training set and a test set consisting of 60% and 40% of the data, respectively. The training set has 298,752 measurements; it includes data from January 1st, 2007 at midnight to September 8th, 2012 at 10:30 am. The test set contains the remaining data, namely 199,080 measurements from September 8th, 2012 at 10:40 am to June 21st, 2016 at 11:40 pm.

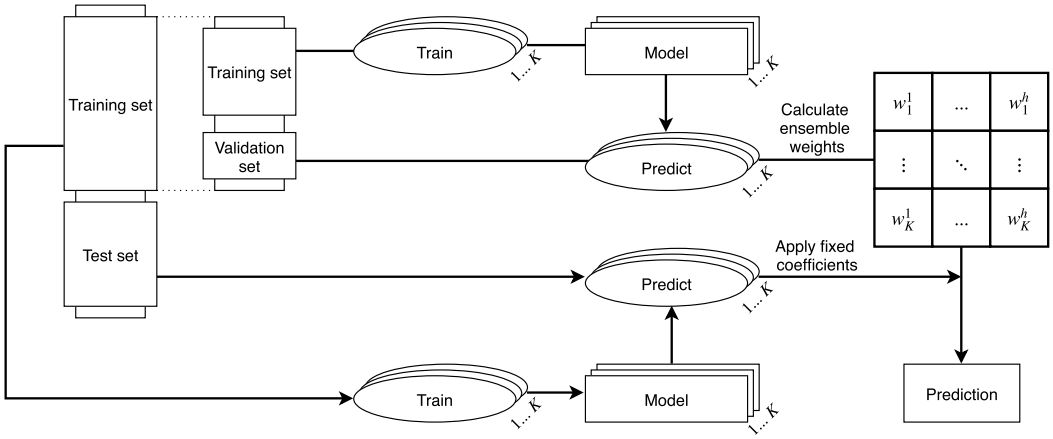


Fig. 1. Workflow of the proposed static ensemble.

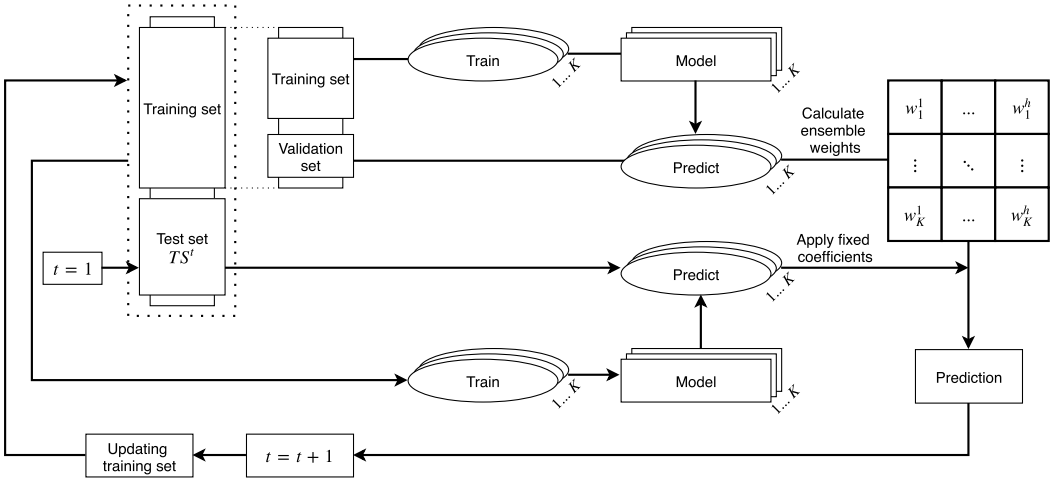


Fig. 2. Workflow of the proposed dynamic ensemble.

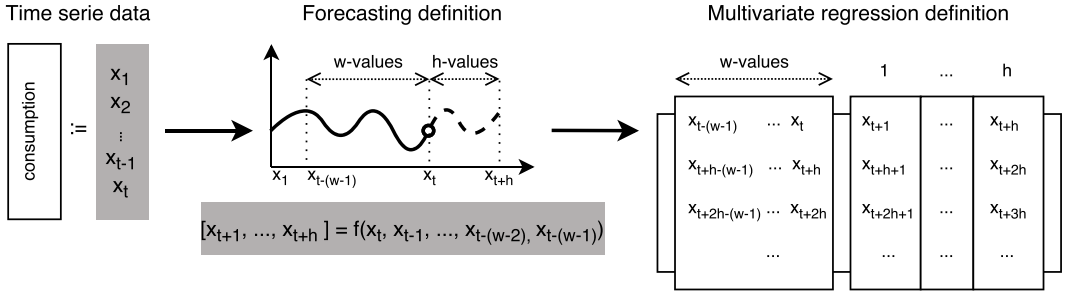


Fig. 3. Illustration of the multivariate problem.

The training set is divided again into a sub-training set –60% used to generate the prediction model for each algorithm , – and a validation set — the remaining 40% used to obtain the weights of the ensemble method .

In the case of dynamic ensemble, the weights of the ensemble are updated every 3 months (every 13,104 predicted values) sliding the training set 13,104 measurements forward while keeping the same training set size. In the same way, the prediction model is updated every 3 months.

4.2. Design of experiments

The experimentation carried out consists of a total of 248 executions, obtaining a total of 5952 prediction models for the time series of electrical consumption in the Spanish electricity market. The experimental setting is summarized below:

1. The size of the window w formed by past values has been set to 24, 48, 72, 96, 120, 144 and 168, corresponding to 4, 8, 12, 16, 20, 24 and 28 h, respectively. Given this number of past values, the goal is to predict the next 24 values.
2. The number of trees and the maximum depth of trees are input parameters in GBT and RF. Both parameters were tested in [15] and the optimal configuration obtained is used in this work. Specifically, a depth of 8 has been used for both algorithms, 5 trees for GBT and 100 trees for RF.
3. The ensemble technique combines DT, GBT and RF. When a dynamic ensemble is applied, the weights are updated every

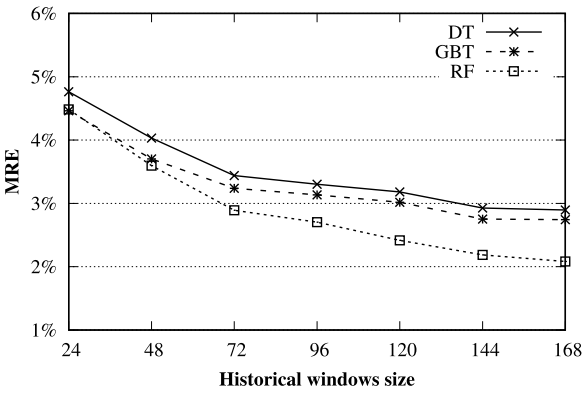


Fig. 4. MRE evolution for different historical window sizes.

3 months. Thus, the dynamic ensemble uses a total of 2304 prediction models.

The mean relative error (MRE) has been used as an evaluation measure to compare the accuracy of the predictions obtained by the different prediction methods. The MRE is defined as follows:

$$MRE = \frac{1}{n}\sum_{i=1}^{n}\frac{|p_i - a_i|}{a_i} \quad (10)$$

where a_i and p_i represent actual and predicted values of the time series, respectively, and n is the number of samples to be predicted.

The experimentation was conducted using high-performance computing resources on the Open Telekom Cloud Platform with five machines, one master and four slave nodes. Each node has 60 GB of main memory and 8 logical cores from an Intel Xeon E5-2658 v3 @ 2.20 GHz processor that has 30 MB L3 cache. The cluster works with Apache Spark 2.1.2 and Hadoop 2.6.

4.3. Sensitivity analysis

This section presents a sensitivity analysis regarding the size of the historical window for the DT, GBT and RF algorithms, which are included in the ensemble model. The analysis consists of a total of 152 executions, obtaining 3648 prediction models in total.

Fig. 4 shows the evolution of MRE on the validation set when increasing the window size for the three proposed methods. For all methods, we can see an improvement in MRE as the size of the window w increases. For the DT algorithm, the optimal configuration was obtained with a window of 168 values, resulting in MRE of 2.90%. For GBT, the optimal model used a window of 168 past values obtaining MRE of 2.74%. Finally, for RF, the smallest error of 2.08% was obtained with a window of 168 past values. However, we can see that increasing the window size from 144 to 168 does not lead to a significant improvement for DT and GBT.

Based on the analysis above, $w = 144$ has been selected for the results shown in the following sections. We note that this window size is not accidental – it represents the values corresponding to the past 24 h –demonstrating the strong stationarity of the time series of electric demand during the day.

4.4. Analysis of results

In this section, we present and discuss the accuracy of the ensemble prediction models – the daily errors along with the worst and best days and the average relative errors of the static and dynamic ensemble models on the test set –comparing them to the errors of the single DT, GBT and RF models.

Table 1
MRE (%) distribution.

Interval	Static	(Agg)	Dynamic	(Agg)
[0,0.5)	0.00	(0)	0.00	(0)
[0.5,1)	0.87	(1)	5.50	(5)
[1,1.5)	13.82	(15)	30.82	(36)
[1.5,2)	28.58	(43)	22.79	(59)
[2,2.5)	23.52	(67)	17.80	(77)
[2.5,3)	16.86	(84)	9.41	(86)
[3,3.5)	6.73	(90)	5.72	(92)
[3.5,4)	3.62	(94)	2.46	(95)
[4,4.5)	2.60	(97)	2.75	(97)
[4.5,5)	1.45	(98)	0.51	(98)
[5,9.5)	1.95	(100)	2.24	(100)

Recall that in the case of DT, GBT, RF and the static ensemble model, we build one prediction model by using 60% of the data as training set. For the dynamic ensemble model, the training set always retains the same size, but the model is updated every 3 months, i.e., every 13,104 values. Thus, the training set is slid forward 13,104 measurements and the weights of the ensemble model are calculated again from the new validation set. Then, a new updated model is obtained to predict the 13,104 next values.

4.4.1. Overall performance

Fig. 5 presents the mean relative error for the static and dynamic ensemble and each individual model they combine, for every hour of the 24 h forecasting horizon. Table 3 also shows the aggregated mean values for each prediction algorithm for the whole test set. From Table 3 we can see that the most accurate prediction model is the dynamic ensemble –it outperforms the static ensemble and all other prediction models . This shows the benefits of dynamically adapting to the changes in the time series when building prediction models.

Within each group (dynamic and static), the ensemble outperforms the individual prediction models it combines. The most accurate individual prediction model is RF, followed by GBT and DT. DTs are single classifiers, so it is expected that they will be outperformed by ensembles of trees such as GBT and RF. RF is performing very well showing the advantage of using two strategies for generating diverse ensemble members — bagging and random feature selection when selecting the best attribute .

Fig. 5 shows that initially (during the first 1–2 h of the forecasting horizon) all methods perform similarly. For hours 3–7, RF and the ensemble show similar performance and start outperforming the other methods, and after that the ensemble method outperforms RF. For the following hours of the forecasting horizon, the ranking of the algorithms is consistent (ensemble, RF, GBT and DT).

From Fig. 5 we can also see that MRE increases as the forecasting horizon increases which is as expected. Thus, the lowest and highest MRE are obtained when the first and last value of the prediction horizon are forecasted, respectively.

4.4.2. Daily performance

To study the daily MRE we group the predictions of each algorithm into groups of 144 values as the measurements are taken every 10 min and we predict 24 h ahead. Fig. 6 shows the histogram of the daily MRE of the test set for the dynamic and static ensemble. The histogram represents the frequency of the daily MRE in different intervals when predicting all days in the test set. We can see that the dynamic ensemble considerably reduces the error in the intervals between 0.5% and 1.0% and 1.0% and 1.5%, where the accuracy is the highest. The impact of this improvement is also noticeable for daily MRE between 1.5% and 3%, due to the low number of days with these average prediction errors for the dynamic ensemble model.

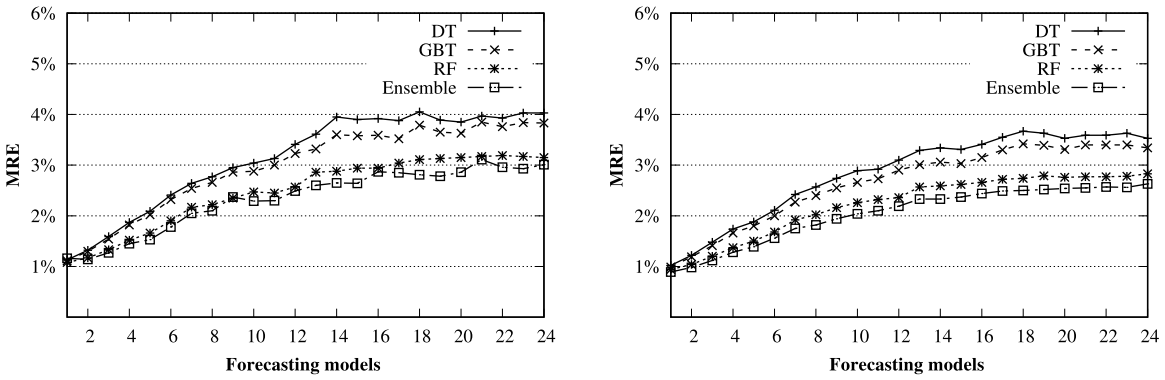


Fig. 5. MRE for each model, for each time point of the prediction horizon.

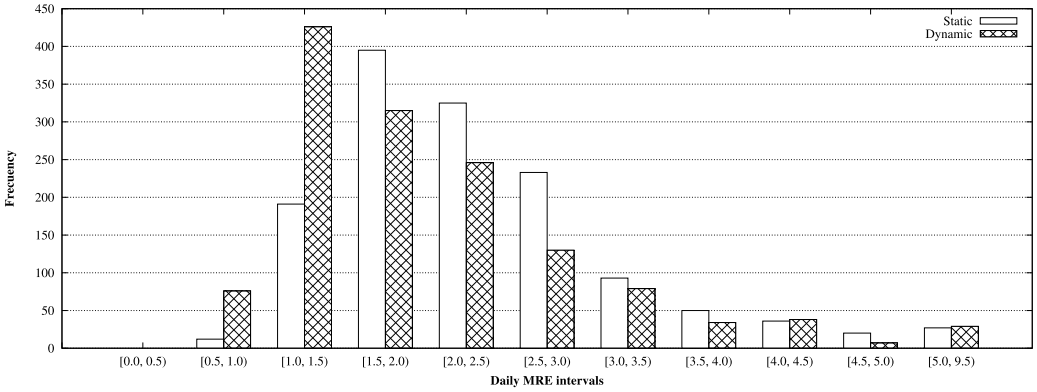


Fig. 6. Histogram of daily errors for static and dynamic ensemble models.

The percentage along with the accumulated number of days for each interval in the histogram is shown in Table 1. As it can be seen, the proposed dynamic ensemble model is stable as the 98% of days have a MRE less than 5% and all days exceeding this threshold correspond to holidays or long weekends as Table 2 shows.

4.4.3. Worst and best days

It is also interesting to study the worst and best predicted days for the different forecasting methods. Table 3 presents the MRE for the best and worst predicted day for DT, GBT, RF and also the static and dynamic ensembles. As it can be observed, the static ensemble model improves the MRE about 25% compared to DT, 21% compared to GBT and 6% compared to RF. In the case of the dynamic ensemble, it achieves a MRE improvement of 28% compared to DT, 23% comparing to GBT and 8% compared to RF. The dynamic ensemble is clearly the best performing model, outperforming the static ensemble with 13%. In addition, it can be noticed that the three ensemble models have similar prediction error variance, which is lower than the variance of the individual methods.

Table 4 compares the static and dynamic ensembles with an ANN that supports the multi-output regression. An ANN configuration with 84 neurons in the hidden layer (the mean of input and output neurons) and a hyperbolic tangent activation function has been selected. It can be seen that the accuracy of ANN is lower than both ensemble models. Compared to the dynamic ensemble, the MRE of ANN for the best predicted day – 2.0199% – is much bigger than the error of the ensemble — 0.7189% . Similarly, the error of the worst predicted day increases from 8.6016% for the ensemble to 17.0503% for ANN.

Fig. 7 shows a graphical representation of the MRE for the test set, and also the MRE corresponding to the days with the best and

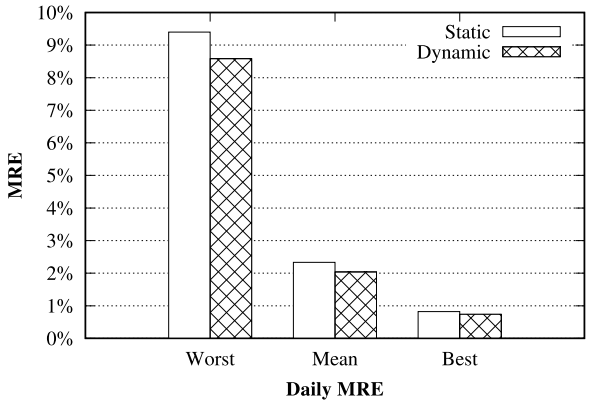


Fig. 7. Comparison of daily MRE of static and dynamic ensemble models.

the worst prediction, for each ensemble model. We can see again that the dynamic ensemble outperforms the static ensemble in all three cases.

Fig. 8 shows how the weight of each individual model in the ensemble is distributed over time. It can be seen that the weights of all three models remain stable for the prediction horizon considered. Moreover, the contribution of the single models ranges from 20% to 40% on average, showing that there is no single dominating model.

Fig. 9 provides information about the hourly predictive performance of the static and dynamic ensemble for the best day. Specifically, Fig. 9(a) shows the actual and predicted electricity demand for the day with the best prediction (MRE of 0.82%), obtained by

Table 2
Worst days for static and dynamic ensemble models.

Static ensemble			Dynamic ensemble		
MRE	Day	Type of Day	MRE	Day	Type of Day
9.32	24/12/13	Christmas Eve	8.60	24/12/13	Christmas Eve
7.71	24/12/12	Christmas Eve	7.18	19/04/14	Easter
7.66	19/04/14	Easter	7.16	30/03/13	Easter
7.40	30/03/13	Easter	7.01	29/03/13	Easter
7.31	24/12/15	Christmas Eve	6.98	24/12/12	Christmas Eve
7.15	31/12/12	New Year's Eve	6.88	31/12/12	New Year's Eve
7.14	24/12/14	Christmas Eve	6.71	24/12/15	Christmas Eve
6.96	31/12/15	New Year's Eve	6.33	31/03/13	Easter
6.56	31/03/13	Easter	6.00	30/04/14	Labour Day
6.41	30/04/13	Labour Day	5.83	30/04/15	Labour Day
6.29	17/04/14	Easter	5.72	01/04/13	Easter
6.29	31/12/13	New Year's Eve	5.58	31/12/15	New Year's Eve
6.00	30/04/15	Labour Day	5.54	31/12/13	New Year's Eve
5.97	29/03/13	Easter	5.54	07/12/14	Immaculate Conception
5.95	21/04/14	Easter	5.48	26/12/12	Christmas Day

Table 3
MRE (mean and variance) on the test set, for the worst and best predicted days.

Model	Algorithm	Worst (%)	Mean (%)	Variance (%)	Best (%)
Static	DT	10.2102	3.1400	0.014	1.2401
	GBT	10.1950	2.9680	0.012	1.2074
	RF	8.8475	2.4838	0.011	0.7621
	Ensemble	**9.3207**	**2.3320**	**0.009**	**0.8230**
Dynamic	DT	9.5022	2.8395	0.015	1.1500
	GBT	9.1633	2.6569	0.014	1.0782
	RF	8.5162	2.2243	0.012	0.6530
	Ensemble	**8.6016**	**2.0362**	**0.010**	**0.7189**

Table 4
Static and dynamic ensembles comparison with ANN.

Method	Worst (%)	Mean (%)	Variance (%)	Best (%)
ANN	17.0503	4.0342	0.136	2.0199
Static ensemble	9.3207	2.3320	**0.009**	0.8230
Dynamic ensemble	**8.6016**	**2.0362**	0.010	**0.7189**

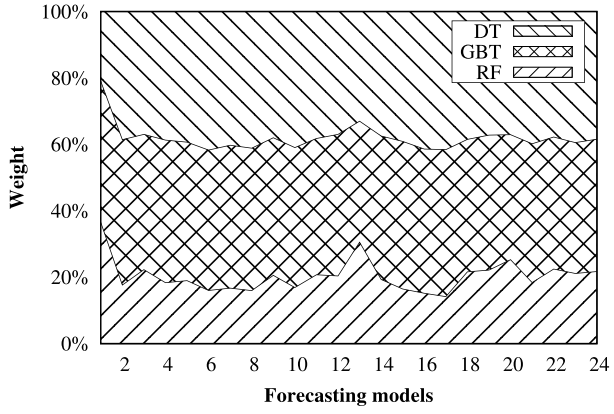


Fig. 8. Static ensemble — weight change during the prediction horizon for the individual models .

the static ensemble model. This day corresponds to the 24 h from Tuesday, August 4th, 2015 at 10:00 pm until Wednesday, August 5th, 2015 at 10:50 pm. Fig. 9(b) shows the same information for the day with the best prediction (MRE of 0.74%), obtained by the dynamic ensemble model. This day, corresponds to the 24 h from Wednesday July 30th, 2014 at 11:00 pm until Thursday July 31st, 2014 at 10:50 pm.

It is also important to analyse the days with worst predictions since they contribute to increasing the average errors. Fig. 10(a) shows the day with the worst prediction obtained with the static ensemble model, resulting in a MRE of 9.32%. This day corresponds to the 24 h from Tuesday December 24th, 2013 at 11:00 pm to Wednesday December 25th at 10:50 pm. Fig. 10(b) shows the day with the worst prediction obtained with the dynamic ensemble model, resulting in MRE of 8.60%, also for the 24 h from Tuesday December 24th, 2013 to Wednesday December 25th. This coincidence is reasonable because December 24th and 25th are special days (Christmas holidays) characterized by more random electricity consumption; they are more different than the previous days, and hence, are more difficult to predict.

In summary, our results showed that both the static and dynamic ensembles performed well and were more accurate than the individual prediction models they combined. The dynamic ensemble was considerably more accurate than the static ensemble, for all predicted days from the test set, achieving an improvement in MRE of about 13%. It also reduced the error of the worst predicted day by 8% and the error of the best predicted day by 13%.

5. Results for solar photovoltaic data

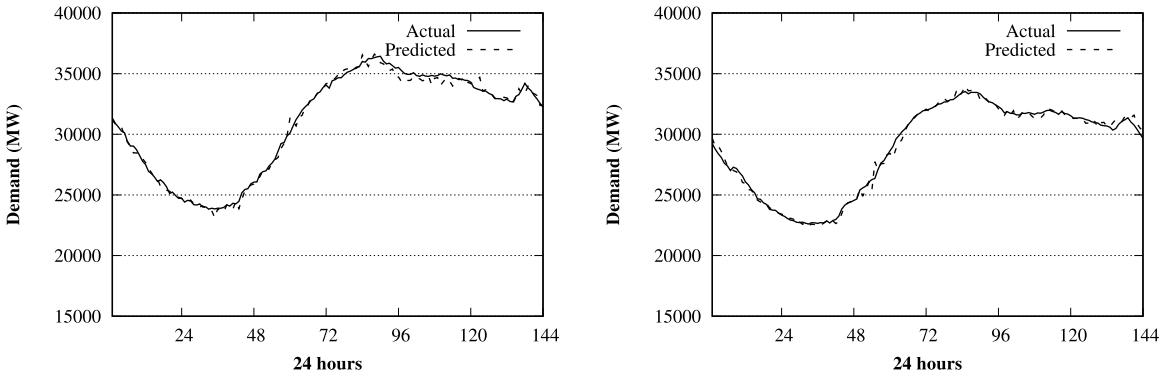
Solar energy is a very promising renewable energy source, which is still underutilized. However, in recent years there has been a considerable increase in production worldwide. Solar energy production depends on weather conditions such as solar radiation, cloud cover, rainfall and temperature. This dependence creates uncertainty where it is important to ensure a reliable supply of electricity, making it difficult to integrate solar energy into electricity markets. Therefore, the ability to predict the solar energy generated is a critical task for stakeholders in the energy sector.

In this section, we present and discuss the application of our proposed method for prediction of solar power data. We firstly describe the dataset in Section 5.1. The experimental setting is presented in Section 5.2. We present and analyse the results obtained by the different static and dynamic ensemble methods in Section 5.3.

5.1. Dataset description

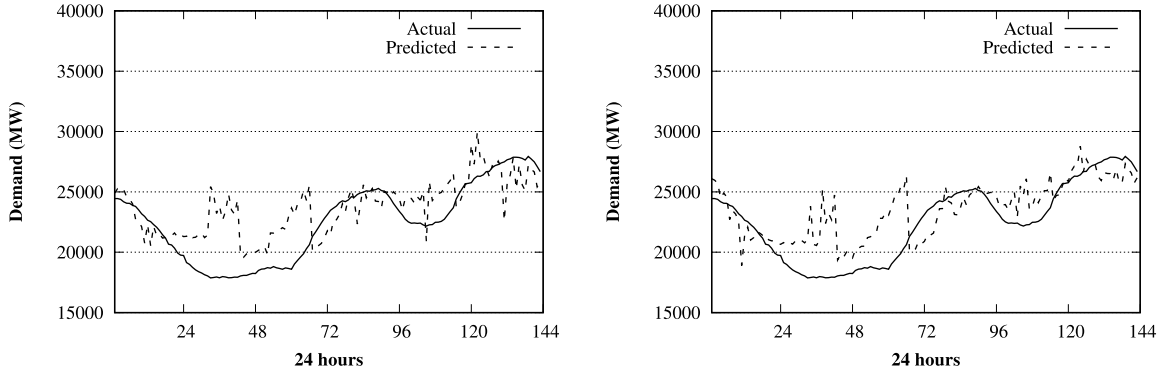
The time series used is related to Australian solar photovoltaic data for two years, from January 1st 2015 to 31 Decemberst 2016. It is a time series with 30 min intervals, where each day has 20 measurements corresponding to the solar day from 7:00 to 17:00.

When using the proposed methodology with a prediction horizon of 10 h (h is set to 20 values), the dataset consists of 730 instances and 20 attributes. These 20 attributes correspond to a window w of 20 past values (10 h). This dataset is divided into a training set and a test set consisting of 70% and 30% of the data, respectively.



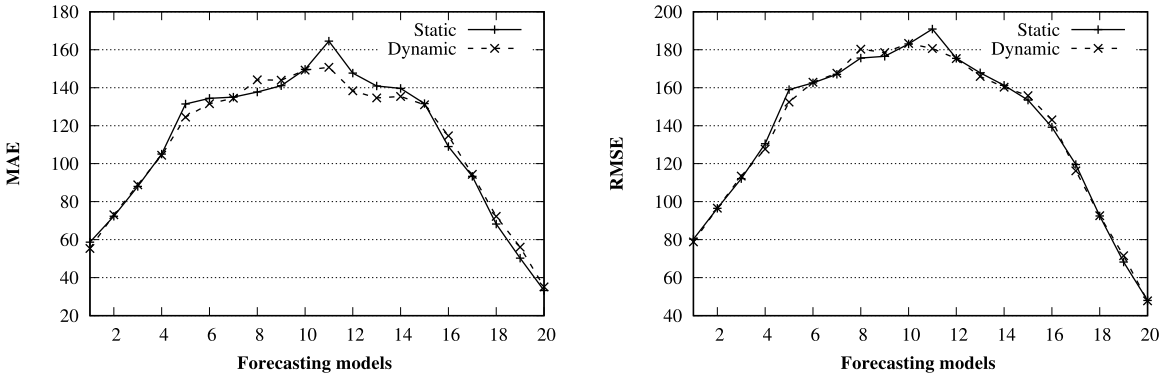
(a) Static ensemble (b) Dynamic ensemble

Fig. 9. Best predicted day — actual and predicted values .



(a) Static ensemble (b) Dynamic ensemble

Fig. 10. Worst predicted day — actual and predicted values .



(a) MAE comparison (b) RMSE comparison

Fig. 11. MAE and MRSE comparison for each time point of the prediction horizon.

The training set is divided again into a sub-training set – 30% used to generate the prediction model for each algorithm –and a validation set—the remaining 30% used to obtain the weights of the ensemble method.

In the case of dynamic ensemble, the weights of the ensemble are updated every two weeks (every 280 predicted values) sliding the training set 280 measurements forward while keeping the same training set size. In the same way, the prediction model is updated every two weeks.

5.2. Design of experiments

Deep learning (DL) has been used to forecast large datasets of solar energy data [46], comparing the performance with two other advanced forecasting methods published in [47]. In particular, Pattern Sequence-based Forecasting (PSF) based on pattern similarity [48] and an Artificial Neural Network (ANN). Static and dynamic ensembles are compared with these algorithms.

According to the referenced work to compare with, the experimental setting is summarized below:

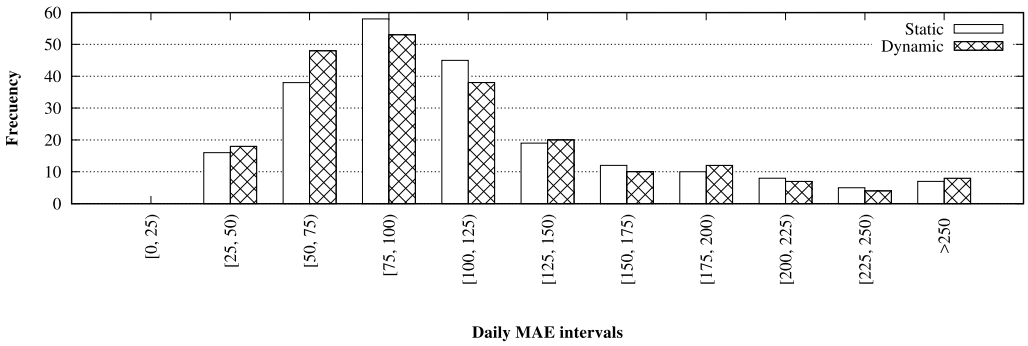


Fig. 12. Histogram of daily errors for static and dynamic ensemble models.

1. The size of the window w is formed by 20 past values, corresponding to 10 h. Given this number of past values, the goal is to predict the next 20 values.
2. The number of trees and the maximum depth of trees are input parameters in GBT and RF. Specifically, a depth of 8 has been used for both algorithms, 5 trees for GBT and 100 trees for RF.
3. The ensemble technique combines DT, GBT and RF. When a dynamic ensemble is applied, the weights are updated every two weeks.

The mean absolute error (MAE) and root mean squared error (RMSE) have been used as evaluation measures to compare the accuracy of the predictions obtained by the different prediction methods. MAE and RMSE are defined as follows:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|p_i - a_i| \tag{11}$$

$$RMSE = \frac{1}{n}\sum_{i=1}^{n}\sqrt{(p_i - a_i)^2} \tag{12}$$

where a_i and p_i represent actual and predicted values of the time series, respectively, and n is the number of samples to be predicted.

5.3. Analysis of results

In this section, we present and discuss the accuracy of the ensemble prediction models – the daily errors along with the worst and best days and the average relative errors of the static and dynamic ensemble models on the test set –comparing them to the errors of the ANN, PSF and DL algorithms.

Recall that in the case of the static ensemble model, we build one prediction model by using 70% of the data as training set. For the dynamic ensemble model, the training set always retains the same size, but the model is updated every two weeks, i.e., every 280 values. Thus, the training set is slid forward 280 measurements and the weights of the ensemble model are calculated again from the new validation set. Then, a new updated model is obtained to predict the 280 next values.

5.3.1. Overall performance

Fig. 11 presents the MAE and RMSE for the static and dynamic ensemble, for every half hour of the 10-hour forecasting horizon. From Table 5 we can see that the most accurate prediction model is the dynamic ensemble. It outperforms the static ensemble and all other prediction models.

Fig. 11 shows that initially (during the first 1–3 h of the forecasting horizon) the accuracy is high. For hours 4–7, prediction is more difficult, because error increases. For the following hours of the forecasting horizon, both methods perform better again.

Table 5
Comparison of the performance of solar energy forecasts.

	Worst		Mean		Best	
Method	MAE	RMSE	MAE	RMSE	MAE	RMSE
ANN	**191.52**	**221.58**	114.64	154.16	58.87	106.88
PSF	252.77	279.12	117.17	147.52	31.72	36.15
DL	206.33	233	114.76	148.98	31.66	41.91
Static ensemble	303.83	353.45	111.59	130.98	**25.93**	**32.95**
Dynamic ensemble	329.60	362.60	**110.62**	**129.46**	26.72	34.04

These result are consistent for both MAE (Fig. 11(a)) and RMSE (Fig. 11(b)), where static and dynamic ensemble behaviours are similar.

5.3.2. Daily performance

To study the daily MAE and MRSE we group the predictions of each algorithm into groups of 20 values as the measurements are taken every 30 min and we predict next day ahead. Fig. 12 shows the histogram of the daily MAE and MRSE of the test set for the dynamic and static ensemble. The histogram represents the frequency of the daily MAE and MRSE in different intervals when predicting all days of the test set. We can see that the dynamic ensemble considerably reduces the error in the MAE intervals between 75 and 125, where the accuracy is the highest.

5.3.3. Worst and best days

It is also interesting to study the worst and best predicted days for the different forecasting methods. Table 5 presents the MAE and MRSE for the best and worst predicted day for ANN, PSF, DL and also the static and dynamic ensembles. As it can observed, the dynamic ensemble achieves the best accuracy by average, and the static ensemble the predicted best day.

Fig. 13 shows a graphical representation of the MAE and RMSE for the test set, and also the MRE corresponding to the days with the best and the worst prediction, for each ensemble model. We can see similar behaviours between dynamic and static ensembles.

It is also important to analyse the days with worst predictions since they contribute to increase the average errors. Fig. 14(a) shows the day with the worst prediction obtained with the static ensemble model, resulting in a MAE of 303.83. Fig. 14(b) shows the day with the worst prediction obtained with the dynamic ensemble model, resulting in MAE of 329.60.

Fig. 15 provides information about the predictive performance of the static and dynamic ensemble for the best day. Specifically, Fig. 15(a) shows the actual and predicted PV data for the day with the best prediction (MAE of 25.93), obtained by the static ensemble model. Fig. 15(b) shows the same information for the day with the best prediction (MAE of 26.72), obtained by the dynamic ensemble model.

The results show that these ensemble methods offer accurate predictions, obtaining better averaged results that the methods

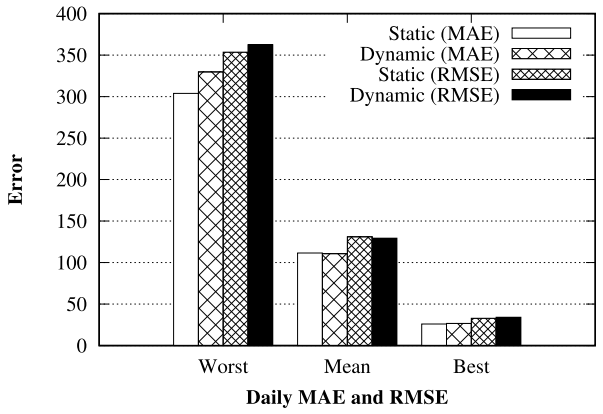


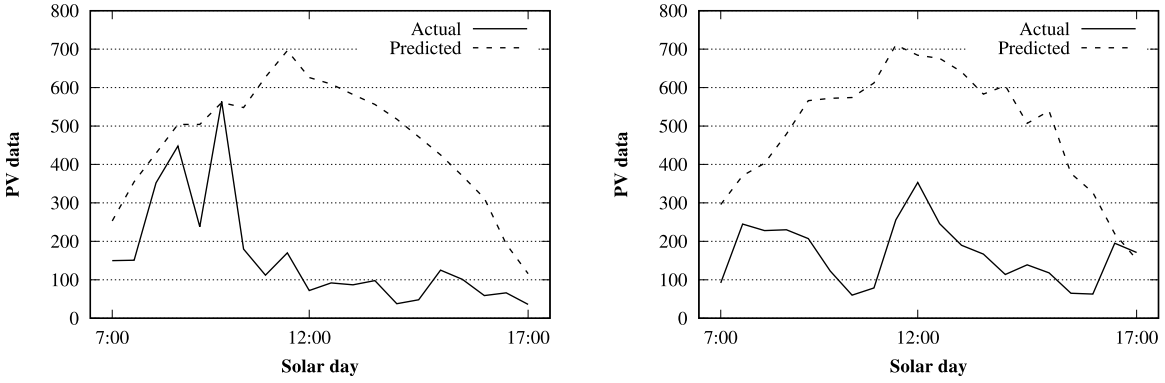
Fig. 13. Comparison of daily MAE and RMSE of static and dynamic ensemble models.

used for comparison. When the day to predict is very atypical, ensembles also have difficulty for obtaining accurate predictions. However, the best predicted days by the ensembles have a higher accuracy than ANN, PSF and DL algorithms. A bigger dataset with more training instances could help the dynamic ensemble.

6. Conclusions

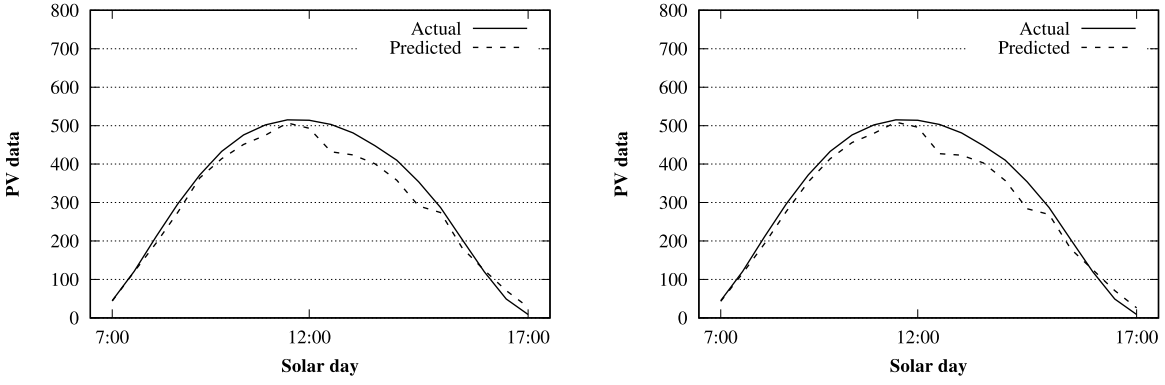
In this paper, we proposed a novel approach based on ensemble learning for predicting big data time series (time series with a high sampling frequency and multi-step forecast horizon). We proposed an ensemble method which computes the weights for each ensemble member using a least square method, assigning higher weights to the more accurate ensemble members based on their past performance. We investigated two strategies for updating the weights, resulting in a dynamic and static ensemble. Although in our case study we have chosen to combine tree-based regression models (DT, GBT and RF), our method can be used to combine other type of prediction models. For the implementation of the prediction algorithms we have used the MLlib library of the Apache Spark framework, to ensure the scalability of our method and its suitability for big data. We conducted a comprehensive evaluation using Spanish electricity consumption data for 10 years consisting of about 500,000 measurements at 10-min intervals. Our results showed that both ensemble methods performed well, outperforming the individual ensemble members they combined, but the dynamic ensemble was the best method. It considerably outperformed the static ensemble, obtaining MRE of 2%. This is very competitive result, showing the viability of the proposed methodology for the prediction of big time series. In addition, Australian solar data have been used to compare static and dynamic ensembles with ANN, PSF and DL, improving the accuracy of these algorithms.

In future work, we plan study the addition of other types of prediction models to the ensemble, which are suitable for big data, in order to increase the diversity among the ensemble members. We will also investigate other machine learning strategies for determining the weights in a dynamic way. We also plan to evaluate the performance of our dynamic ensemble on other big datasets from different sources. Finally, we will develop models for forecasting special days.



(a) Static ensemble (b) Dynamic ensemble

Fig. 14. Worst predicted day—actual and predicted values .



(a) Static ensemble (b) Dynamic ensemble

Fig. 15. Best predicted day—actual and predicted values .

Acknowledgements

The authors would like to thank the Spanish Ministry of Economy and Competitiveness and Junta de Andalucía for the support under projects TIN2017-8888209C2-1-R, TIN2014-55894-C2-R and P12-TIC-1728, respectively. Additionally, the authors want to express their gratitude to the T-Systems Iberia company since all experiments were carried out on its Open Telekom Cloud Platform based on the Open-Stack open source.

References

[1] M. Ge, H. Bangui, B. Buhnova, Big data for internet of things: A survey, Future Gener. Comput. Syst. (2018).
[2] X. Liu, P.S. Nielsen, A hybrid ICT-solution for smart meter data analytics, Energy 115 (2016) 1710–1722.
[3] Y. Sakurai, Y. Matsubara, C. Faloutsos, Mining and forecasting of big time-series data, in: Proceedings of the ACM SIGMOD International Conference on Management of Data, 2015, pp. 919–922.
[4] J. Dean, S. Ghemawat, MapReduce: Simplified Data Processing on Large Clusters.
[5] T. White, Hadoop, the Definitive Guide, O' Really Media, 2012.
[6] M. Hamstra, H. Karau, M. Zaharia, A. Knwinski, P. Wendell, Learning Spark: Lightning-Fast Big Analytics, O' Really Media, 2015.
[7] Machine learning library (mllib) for apache spark, 2016, On-line. http://spark.apache.org/docs/latest/mllib-guide.html.
[8] F. Hu, C. Yang, J.L. Schnase, D.Q. Duffy, M. Xu, M.K. Bowen, T. Lee, W. Song, ClimateSpark: An in-memory distributed computing framework for big climate data analytics, Comput. Geosci. 115 (2018) 154–166.
[9] M.E. Shafiee, Z. Barker, A. Rasekh, Enhancing water system models by integrating big data, Sustai. Cities Soc. 37 (2018) 485–491.
[10] S. Magana-Zook, J.M. Gaylord, D.R. Knapp, D.A. Dodge, S.D. Ruppert, Large-scale seismic waveform quality metric calculation using Hadoop, Comput. Geosci. 94 (2016) 18–30.
[11] D. Gomes-Mestre, C.E. Santos-Pires, D.C. Nascimento, A.R. Monteiro-de Queiroz, V. Borges-Santos, T. Brasileiro-Araujo, An efficient spark-based adaptive windowing for entity matching, J. Syst. Softw. 128 (2017) 1–10.
[12] T. Cerquitelli, Predicting large scale fine grain energy consumption, Energy Proc. 111 (2017) 1079–1088, 8th International Conference on Sustainability in Energy and Buildings, SEB-16, 11-13 2016, Turin, Italy.
[13] X. Zhang, U. Khanal, X. Zhao, S. Ficklin, Making sense of performance in in-memory computing frameworks for scientific data analysis: A case study of the spark system, J. Parallel Distrib. Comput. (2017).
[14] S. Caíno-Lores, A. Lapin, J. Carretero, P. Kropf, Applying big data paradigms to a large scale scientific workflow: Lessons learned and future directions, Future Gener. Comput. Syst. (2018).
[15] A. Galicia, J.F. Torres, F. Martínez-Álvarez, A. Troncoso, Scalable forecasting techniques applied to big electricity time series, in: Proceedings of the 14th International Work-Conference on Artificial Neural Networks, 2017, pp. 165–175.
[16] F. Martínez-Álvarez, A. Troncoso, G. Asencio-Cortés, J.C. Riquelme, A survey on data mining techniques applied to electricity-related time series forecasting, Energies 8 (11) (2015) 13162–13193.
[17] G. Box, G. Jenkins, Time Series Analysis: Forecasting and Control, John Wiley and Sons, 2008.
[18] C.W. Tsai, C.F. Lai, H.C. Chao, A. Vasilakos, Big data analytics: a survey, J. Big Data 2 (1) (2015) 21.
[19] L. Zhou, S. Pan, J. Wang, A.V. Vasilakos, Machine learning on big data: opportunities and challenges, Neurocomputing 237 (2017) 350–361.
[20] G. Cavallaro, M. Riedel, M. Richerzhagen, J.A. Benediktsson, A. Plaza, On understanding big data impacts in remotely sensed image classification using support vector machine methods, IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 8 (2015) 4634–4646.
[21] J.L. Reyes-Ortiz, L. Oneto, D. Anguita, Big data analytics in the cloud: Spark on hadoop vs MPI/OpenMP on Beowulf, Proc. Comput. Sci. 53 (2015) 121–130.
[22] R. Talavera-Llames, R. Pérez-Chacón, M. Martínez-Ballesteros, A. Troncoso, F. Martínez-Álvarez, A nearest neighbours-based algorithm for big time series data forecasting, in: Proceedings of the International Conference on Hybrid Artificial Intelligence Systems, 2016, pp. 174–185.
[23] J. González-López, S. Ventura, A. Cano, Distributed nearest neighbor classification for large-scale multi-label data on spark, Future Gener. Comput. Syst. 87 (2018) 66–82.
[24] I. Triguero, D. Peralta, J. Bacardit, S. García, F. Herrera, MRPR: A MapReduce solution for prototype reduction in big data classification, Neurocomputing 150 (2015) 331–345.
[25] N. Mehdiyev, J. Lahann, A. Emrich, D. Enke, P. Fettke, P. Loos, Time series classification using deep learning for process planning: A case from the process industry, Proc. Comput. Sci. 114 (2017) 242–249.
[26] A. Fahad, N. Alshatri, Z. Tari, A. Alamri, A.Y. Zomaya, I. Khalil, F. Sebti, A. Bouras, A.Y. Zomaya, S. Foufou, A. Bouras, A survey of clustering algorithms for big data: Taxonomy & empirical analysis, IEEE Trans. Emerg. Top. Comput. 5 (3) (2014) 267–279.
[27] H. Asri, H. Mousannif, H.A. Moatassime, Real-time miscarriage prediction with SPARK, Proc. Comput. Sci. 113 (2017) 423–428.
[28] R. Pérez-Chacón, R. Talavera-Llames, F. Martínez-Álvarez, A. Troncoso, Finding electric energy consumption patterns in big time series data, in: Proceedings of the International Conference on Distributed Computing and Artificial Intelligence, 2016, pp. 231–238.
[29] J.M. Luna-Romera, M.M. Martínez-Ballesteros, J. García-Gutiérrez, J.C. Riquelme-Santos, J.C. Riquelme, M.M. Martínez-Ballesteros, J.C. Riquelme, An approach to silhouette and dunn clustering indices applied to big data in spark, in: Progress in Artificial Intelligence, Springer, Cham, 2016, pp. 160–169.
[30] H. Hassani, E.S. Silva, Forecasting with big data: A review, Ann. Data Sci. 1 (2) (2015) 5–19.
[31] G. Asencio-Cortés, A. Morales-Esteban, X. Shang, F. Martínez-Álvarez, Earthquake prediction in California using regression algorithms and cloud-based big data infrastructure, Comput. Geosci. 115 (2018) 198–210.
[32] J.F. Torres, A.M. Fernández, A. Troncoso, F. Martínez-Álvarez, Deep Learning-Based Approach for Time Series Forecasting with Application to Electricity Load, Springer International Publishing, 2017, pp. 203–212.
[33] I. Rodríguez-Fdez, M. Mucientes, A. Bugarín, S-FRULER: Scalable fuzzy rule learning through evolution for regression, Knowl.-Based Syst. 110 (2016) 255–266.
[34] L.I. Kuncheva, Combining Pattern Classifiers: Methods and Algorithms, John Wiley & Sons, 2014.
[35] R. Polikar, Ensemble Learning, Springer US, 2012, pp. 1–34.
[36] B. Krawczyk, L.L. Minku, J. Gama, J. Stefanowski, Michał Woźniak, Ensemble learning for data stream analysis: A survey, Inf. Fusion 37 (2017) 132–156.
[37] L. Li, S. Bagheri, H. Goote, A. Hasan, G. Hazard, Risk adjustement of patient expenditures: A big data analytics approach, in: Proceedings of the IEEE International Conference on Big Data, 2013, pp. 12–14.
[38] B. Panda, J.S. Herbach, S. Basu, R.J. Bayardo, PLANET: massively parallel learning of tree ensembles with MapReduce, in: Proceedings of the Very Large Databases, 2009, pp. 1426–1437.
[39] K. Singh, S.C. Guntuku, A. Thakur, C. Hota, Big data analytics framework for peer-to-peer botnet detection using random forests, Inform. Sci. 278 (2014) 488–497.
[40] J.H. Abawajy, A. Kelarev, M. Chowdhury, Large iterative multitier ensemble classifiers for security of big data, IEEE Trans. Emerg. Top. Comput. 2 (3) (2014) 352–363.
[41] Y. Yan, Q. Zhu, M.L. Shyu, S.C. Chen, A classifier ensemble framework for multimedia big data classification, in: 2016 IEEE 17th International Conference on Information Reuse and Integration (IRI), jul 2016, pp. 615–622.
[42] T. Do, F. Poulet, Random local SVMs for classifying large datasets, in: Proceedings of the International Conference on Future Data and Security Engineering, 2015, pp. 3–15.
[43] V. Estruch, C. Ferri, J. Hernández-Orallo, M.J. Ramírez-Quintana, Shared ensemble learning using multi-trees, in: Proceedings of the 8th Ibero-American Conference on Artificial Intelligence, 2002, pp. 204–213.
[44] A. Torres-Barrán, Á. Alonso, J.R. Dorronsoro, Regression tree ensembles for wind energy and solar radiation prediction, Neurocomputing (2017).
[45] M.A. Hassan, A. Khalil, S. Kaseb, M.A. Kassem, Exploring the potential of tree-based ensemble methods in solar radiation modeling, Appl. Energy 203 (Suppl. C) (2017) 897–916.
[46] J.F. Torres, A. Troncoso, I. Koprinska, Z. Wang, F. Martínez-Álvarez, Deep learning for big data time series forecasting applied to solar power, in: International Joint Conference SOCO'18-CISIS'18-ICEUTE'18, Springer International Publishing, 2019, pp. 123–133.
[47] Z. Wang, I. Koprinska, M. Rana, Solar power forecasting using pattern sequences, in: Artificial Neural Networks and Machine Learning – ICANN 2017, Springer International Publishing, 2017, pp. 486–494.
[48] F. Martínez-Álvarez, A. Troncoso, J.C. Riquelme, J.S. Aguilar Ruiz, Energy time series forecasting based on pattern sequence similarity, IEEE Trans. Knowl. Data Eng. 23 (8) (2011) 1230–1243.