Short-term water demand forecasting using machine learning techniques

A. Antunes, A. Andrade-Campos, A. Sardinha-Lourenço and M. S. Oliveira

ABSTRACT

Nowadays, a large number of water utilities still manage their operation on the instant water demand of the network, meaning that the use of the equipment is conditioned by the immediate water necessity. The water reservoirs of the networks are filled using pumps that start working when the water level reaches a specified minimum, stopping when it reaches a maximum level. Shifting the focus to water management based on future demand allows use of the equipment when energy is cheaper, taking advantage of the electricity tariff in action, thus bringing significant financial savings over time. Short-term water demand forecasting is a crucial step to support decision making regarding the equipment operation management. For this purpose, forecasting methodologies are analyzed and implemented. Several machine learning methods, such as neural networks, random forests, support vector machines and k-nearest neighbors, are evaluated using real data from two Portuguese water utilities. Moreover, the influence of factors such as weather, seasonality, amount of data used in training and forecast window is also analysed. A weighted parallel strategy that gathers the advantages of the different machine learning techniques is suggested. The results are validated and compared with those achieved by autoregressive integrated moving average (ARIMA) also using benchmarks.

Key words | machine learning, short-term, water demand forecast, water supply systems, water utilities, weighted parallel strategy

A. Antunes
A. Andrade-Campos (corresponding author)
A. Sardinha-Lourenço
Department of Mechanical Engineering, Centre for Mechanical Technology and Automation (TEMA), GRIDS Research Group,
University of Aveiro,
Campus Universitário de Santiago, 3810-193 Aveiro,
Portugal
E-mail: *gilac@ua.pt*

M. S. Oliveira
Department of Economics, Management and Industrial Engineering,
University of Aveiro,
Campus Universitário de Santiago, 3810-193 Aveiro,
Portugal
and
INOVA, Water Management Company,
Zona Industrial,
3064-909 Cantanhede,
Portugal

INTRODUCTION

The cost efficiency and operation of water supply systems (WSS) can be improved by taking advantage of the electricity tariff, favoring its operation when the electricity is cheaper and avoiding the more expensive periods. By predicting the water demand in the short-term (24–48 h) the pumping schedule can be planned to operate at the cheapest periods, always guaranteeing the availability of minimum water in the tanks (Bunn & Reynolds 2009; Coelho 2016).

Water demand forecast depends on some phenomena, such as the inertia of the system, seasonality, type and number of clients, and the external factors that affect the consumption. Short-term (daily) forecasting has significant importance, allowing an efficient management of the water existent in the storage tanks and of the equipment associated with it. Long-term (annual) forecasting is essential in the water network design phase. Mala-Jetmarova *et al.* (2017) present an extensive literature review on the field of optimization of water distribution systems.

Artificial intelligence is a field of knowledge dedicated to developing ways to make machines and computers mimic human intelligence and behavior. In machine learning, three main types of problems arise: supervised learning, unsupervised learning (Bishop 2006), and reinforced learning (Mitchell 1997). The first deals with datasets composed of inputs and outputs. For any paired input–output, the machine must determine how they relate to each

doi: 10.2166/hydro.2018.163

another, allowing it to estimate the probable output for a new untrained input. The second deals with problems in which the datasets used in the training process are composed only by inputs. The machine's task is to find the common features between examples and categorize them. As for the latter, it consists of a group of problems with no dataset. The computer generates its own dataset by running examples and evaluating the results.

Several works, such as Cembrano *et al.* (2000), Salomons *et al.* (2007) and Kang *et al.* (2014), analyze the use of water consumption forecasting to improve the operation of WSSs. They concluded that demand forecasting and deriving the operation schedule from the results can lead to a cost reduction ranging between 18 and 55%. Although a cost reduction higher than 18% is not always guaranteed, these works show that a poor forecast is better than none at all, provided the decision maker takes it into consideration.

Recent studies, such as Alvisi *et al.* (2007), Bakker *et al.* (2013), Candelieri *et al.* (2014a) and Shabani *et al.* (2018), made their forecasts using hybrid methods which are decomposed in two steps: in the first, the data is analyzed as a whole and the different patterns (clusters) are identified – unsupervised learning. In the second, an algorithm is applied to each identified cluster to produce reliable predictions. Similarly, Candelieri *et al.* (2014b) use the same two-step strategy to detect water leaks.

In their paper, Candelieri *et al.* (2014a) describe a method to forecast water demand in the city of Milan, Italy. Their method is also divided into two steps: (1) identifying patterns in the water consumption data and (2) predicting the water demand of the network for the next 24-*t* hours based on the first *t* hours of any given day. To identify the patterns in the data, cosine similarity techniques were used on all the time series calculated for each 24-hour division of the data. They found six distinct clusters: three relative to periods of year ('Spring-Summer', 'Fall-Winter' and 'Summer-break'), combined with two day-type clusters ('working-days' and 'holidays-weekends'). The second step deals with the forecasting of the water demand for any given day, based on the consumption observed in the first hours of that day. Comparing the measured consumption of the first *t* hours of the day with the data contained in the identified clusters, and using a series of support vector regression (SVR) models previously trained, the output is the predicted water demand for the remaining hours of the day. There is a different SVR for each combination of cluster-hour of the day. A more recent study involving the two-step approach (clustering + forecasting) used smart meters at an individual level (Candelieri 2017). The use of these meters proved its usefulness at the network's operation optimization through demand forecast as well as specific and traceable anomaly detection such as fraud or smart meter faults.

Parallel studies (Herrera *et al.* 2010; de Lima *et al.* 2016; Ghiassi *et al.* 2016; Peña-Guzman *et al.* 2016; Tiwari *et al.* 2016) have been made with the purpose of testing different forecasting methods and comparing their accuracy. Generically, it is shown that machine learning techniques (e.g. SVR and artificial neural networks (ANN)) have higher accuracy than non-machine-learning approaches (e.g. time-series). Herrera *et al.* (2010) tested a group of forecasting algorithms on a dataset corresponding to a city in south-eastern Spain. They concluded that the methods can be ranked considering their accuracy as follows: heuristic model, ANN, random forest, projection pursuit regression, multi-variate adaptive regression splines, and SVR. Furthermore, comparison tests have also been made with different configurations of each method.

De Lima *et al.* (2016) studied three forecasting methods: exponential smoothing (ES), seasonal autoregressive integrated moving average (SARIMA) and ANN. Additionally, they applied those methods to data from 10 cities in Paraná, Brazil. Then, 14 combinations of the results were evaluated to assess the best. They concluded that more complex methods do not mean better results, since ES was found to be the best method in five cities, SARIMA in four cities and ANN in just one city. The best model in each city showed values of mean absolute percentage error (MAPE) of less than 4%.

Tiwari *et al.* (2016) compared six models, consisting of two methods: extreme learning machine (ELM – a derivation of neural networks where the weights of the hidden neurons are randomly assigned or inherited from their predecessors) and ANN with three different implementations: traditional, wavelet analysis and bootstrap. These methods used three years of water demand and climate registries of a network in Calgary, Alberta, Canada. The ELM and ANN methods achieved similar results and showed no significant improvement when using the bootstrap method. However, significant improvements were observed when the wavelet analysis was applied to both ELM and ANN.

Peña-Guzman et al. (2016) applied support vector machines to a real network located in Bogotá, Colombia, and used previously observed water consumption, number of users and the value billed for monthly consumption data in their forecasts. They analyzed six residential sub-networks, one commercial sub-network and one industrial sub-network. Except for one residential sub-network, all the others showed a root mean square error (RMSE) of <2% and coefficient of determination $R^2 > 0.9$. Moreover, they found that the least squares support vector machine used achieved better performance than the feedforward neural network backpropagation (FNN-BP) tested for comparison.

Ghiassi et al. (2016) used three machine learning methods (dynamic artificial neural network (DANN), focused time-delay neural network and *k*-nearest neighbors (KNN)) to forecast the urban water demand in Tehran, Iran, for three time horizons: four weeks, six months and two years, using respectively daily, weekly and monthly time steps. Their methods used the daily water production and monthly water consumption data between March 2003 and April 2009 provided by the Tehran Water and Wastewater Company. They tested two methods for the daily forecasts, where they studied the impact of partitioning the weekdays into weekends and non-weekends. They found that the best results were achieved when this partitioning was not considered. They also tested the monthly forecast taking into consideration seasonality (high and low seasons) and observed a positive impact of this decision. Generically, the three developed methods were considered to provide good results in the three time scales, with a slightly better performance by the DANN.

Brentan et al. (2017) developed a hybrid method in which they make a base prediction using SVR, followed by the application of an adaptive Fourier series (AFS) to improve the previous forecast. This method was validated using the dataset of a water utility in Franca, Brazil. They used previously observed consumption and weather data (rain, temperature, humidity and wind velocity) in the process. They also considered the yearly seasonality (on a monthly basis) and the difference between weekends, non-weekends and holidays. The comparison between the developed hybrid method and the basic SVR method proved that applying the AFS resulted in a much better forecast: RMSE from 4.767 to 1.318 L/s, mean absolute error (MAE) from 12.91 to 3.45% and coefficient of determination R^2 from 0.745 to 0.974.

Shabani et al. (2016) used phase space reconstruction to derive the proper lag time (found to be three months) to be used in their genetic expression programming (GEP) method, which is aimed at predicting the average water demand for the entire next month. In the dataset considered, they found a high correlation between the water demand, the temperature and hotel occupancy, which seems to reflect the seasonality in the case being studied. The population of the city and the rainfall did not show a high correlation with the water demand forecast. The GEP with the best performance was then compared with SVR with different kernel functions – radial, linear and polynomial – and the polynomial was found to be the best, not only among the SVM but also among all the methods evaluated. The results were validated using data referent to the City of Kelowna, British Columbia, Canada.

In their work, Moutadid & Adamowski (2017) used combinations of water demand data (1999–2010), maximum daily temperature and daily total precipitation referent to the city of Montreal, Canada, and forecast the water demand with one and three days of lead time. ANN, SVR, ELM and the traditional multiple linear regression models were evaluated, with the ELM presenting the best performance independent of the lead time. They also observed that an increase in the lead time means a worse forecasting, even though this diminishing in performance is considered as not drastic by the authors.

Haque et al. (2017) showed an innovative regression method named independent component regression (ICR) and applied it to medium-term (monthly) water demand forecasting in Aquidauano, Brazil. For comparison, they also calculated forecasts using multiple linear regression and principal component regression. They used monthly history data of maximum temperature, relative humidity, number of water customers and water consumption. The results showed that even though the R^2 of the ICR method was lower, the other evaluation parameters proved its high performance. An overestimation tendency of the ICR method was observed.

Rodríguez-Galiano & Villarín-Clavería (2016) applied regression trees as a water demand forecasting technique. The model used socio-demographic data such as age of the population, cadastral value and size of the buildings, and derivatives of these. In total, 15 variables were used as input vector in the training process. The domestic water consumption history was used as the output target vector in the training process. They evaluated the RMSE when using *n* variables with more impact in the forecasts and observed that when using only one variable (household size [inhab./household]) the RMSE = 26.91 L/y, and using the 15 input variables the RMSE = 18.89 L/y. As a consequence of using the *n* most important input variables, they also observed that the last input variables used had little impact on the forecasts. The RMSE calculated when using only the six most important variables was 18.96 L/y. Considering more variables results in an insignificant improvement in the results, with a higher computational cost. The tests were performed using data relative to a WSS in Sevilla, Spain.

Mellios *et al.* (2015) developed an artificial neural-fuzzy inference system (ANFIS) to forecast the daily water demand in the Greek touristic island of Skiathos. They used daily water pumping history, daily mean and high temperature, daily precipitation, daily wind speed and monthly arrivals by air and sea for a two-year period (2011–2012) for training and 2013 for testing. From the 32 networks evaluated, the best presented the following results: $R^2 = 0.916$, RMSE = 192.99 m^3 and MAPE = 8.1%.

In their work, Suh & Ham (2016) used backpropagation (BP)-ANN to forecast the water demand in buildings of four cities in South Korea. Using climate, geographic, and morphologic input variables and average monthly water consumption as output in the training process, they could predict the monthly water consumption with a MAPE of 19.6% and RMSE of 98.11 m^3/y.

Seo *et al.* (2015) used three wavelet decomposition methods to assess their ability to predict the water level of a dam. They used the ANN and ANFIS methods and their decomposed variants WANN and WANFIS and validated the results using real daily water level data for the Andong Dam in South Korea. The results showed that the ANFIS methods are generally better than the ANN. Furthermore, the application of the wavelet decomposition resulted in significantly better results, and the best method is identified as WANFIS7-sym10 – input set 7 with Symmlet-10 wavelet decomposition.

Adamowski & Karapataki (2010) observed that for peak urban water demand forecasting, and in the two datasets used (networks in Nicosia, Cyprus), the accuracy of the learning algorithms can be ranked as follows: multiple linear regression, resilient BP-ANN, conjugate gradient Powell-Beale ANN, Levenberg-Marquardt ANN.

Some studies have been made on the influence of the weather as input data, such as Rodríguez-Galiano & Villarín-Clavería (2016) and Adamowski & Karapataki (2010). They all concluded that the weather influences the water demand, mainly on domestic and agricultural levels. Nonetheless, opposing conclusions have been presented concerning the impact on the forecast of either the quantity of rain or the occurrence of rain (Adamowski & Karapataki 2010). Although the use of weather data as input was shown as a performance enhancer of the methods used, it is also well understood by the community that the difficulties of implementation of such methods do not always pay off for the additional effort. Bakker *et al.* (2013) developed a method that takes into consideration the weather effect, even though they do not use any weather data input. In this study, they also showed that a shorter time interval helps to model critical times of the day (early morning), but results in a smaller overall accuracy.

Machine learning techniques have also been used to predict malfunctions of the equipment and locate leakages. Candelieri *et al.* (2014b) used spectral clustering and SVR techniques with the aid of a simulated water supply network. They achieved a reliability of 98% for pressure and flow variables and leak locations. When applied to real cases (Milan and Timisoara), this technique achieved a reliability larger than 90%.

DEVELOPING A MACHINE LEARNING WATER DEMAND FORECASTING MODEL

According to previous studies made in the field, no particular machine learning model is the most adequate for every water demand forecasting problem. However, it is expected that some methods will present better predictions than others for different datasets. Developing a flexible, transversal and accurate algorithm involves studying a variety of methods applied to multiple databases.

Choosing the input data features

The selection of the right input features for the training stage is critical. In this context, each feature represents an input variable that affects the outcome of the forecast. Some examples are the water demand history, temperature or rain occurrence. Note that the temperature observed at the instant t and the temperature observed at the instant t-1 can be two features both referents to the same instant t. The scientific community has made several studies concerning the most adequate features for water demand forecast and it can be generically concluded that, besides the historic data, the weather and seasonality have the strongest impact on the results.

In this context, seasonality must be considered at several levels and is to some extent correlated with the weather data. The sporadic seasonal events, such as Christmas, Easter or even a major sports event, can also be considered. The seasonality can be implemented in the algorithm in two different ways:

1. The periodicity of the data directly affects the number of machines (predictive models) used in the forecast and, consequentially, the amount of data used in the training of each machine. For example, considering a periodicity of 24 hours means the algorithm will train 24 machines and make forecasts for a 24-hour interval. The different periodicities to consider have different applications and may have implications for the accuracy of the forecasts.
2. Identifying and separating different patterns in the data and using each of them as independent datasets. This is known as clustering and can be achieved using machine learning algorithms. However, for the purpose of this study, it was decided to use brute-force clustering, where the algorithm is explicitly told how to separate the different patterns. By using this approach (an approach is the group of features applied to each model; one approach might be using 70 water demand features, and another approach might be using 14 water demand and one temperature feature), it is assumed that different periods have different typical behaviors and therefore must be predicted based solely on those of the same pattern. Additionally, sporadic events may be considered as one of these clusters. In this paper, the clusters considered are: (i) weekday, including the days between Monday and Friday; and (ii) weekend, including Saturdays, Sundays and the Portuguese official holidays.

As for weather features, the works made on the subject concluded that temperature, rain amount and rain occurrence have the largest impact on the training stage. For this reason, when available, these registries are considered as features for the forecast. Nonetheless, models (a model is a well-defined forecasting method configured and trained) with no weather features are also tested. Regarding this matter, it is considered that the water demand forecast depends on the previous water consumption observations paired with the predicted weather conditions. For example, the forecast for the hour t of tomorrow depends on the water consumption observed at the hour t of today and the temperature forecast for hour t of tomorrow. The weather variables are not predicted, as they are usually available from external sources, and are not the object of this work.

The last consideration is related to the amount of data that is used in the process. In this work, the aim is to use as much available data as possible. If two years of records are available, it is not advised to use any less than those two years of data. However, the water consumption observed two years ago does not have a direct impact on tomorrow's demand. All the data must be used during the training, but only a part of that has a direct influence on each step of the process and, consequently, on the forecasting. The amount of data used is updated in each training step but maintains the same size (see Figure 2). This implies that the water consumption for the day D_{+1} is predicted considering the features registered in the days D_{-n} to D_{-1}, and each of those days is a sample. In this context, a sample represents a moment of observation, each consisting of the features used by the machine (in training and predicting). Each sample is a vector of the features' values observed at any given moment. Additionally, the number of weather features considered must also be tested, and it is not mandatory that it equals the number of demand observations considered. In practice, tests are made considering only the weather forecast for 1 day or weather forecasts for 14 days (starting from the forecasted day, towards the previous days).

Clustering

In unsupervised learning, a common problem is to identify and correctly classify a certain number of identical classes (clusters) present in the data (Bishop 2006). At first, the number of clusters, k, is usually unknown. Finding it can be achieved by running the clustering algorithm for different values of k and evaluating the results for each value. The optimal number of clusters is the one that optimizes a determined criterion, such as the variance between observations in the same cluster. If for $k+1$ clusters the criterion used has no significant improvement, then k is usually a good guess for the number of identical classes in the data.

If k classes are thought to exist, the clustering algorithm will randomly allocate k points as cluster centers. Then each point of the data is compared with the existing cluster centers and is assigned to the one that is more similar to itself (i.e. has smaller distance). For the next iteration, the new cluster centers are re-calculated as the average of all the points that were assigned to them. A variation of this algorithm calculates the new cluster centers when each point is assigned to any of them.

Similarity and distance are slightly different concepts, although they both try to translate the correlation between two vectors in a numerical way. Different formulations of these concepts usually mean different results. Altogether, the number of clusters and how they are found, the formulation of the distance and the clustering algorithm implementation itself are mutually affected, and different combinations of them will produce different results.

Here, clustering analysis can be used to verify seasonality and other relations between input data.

Forecasting techniques

The water demand forecast can be seen as a regression problem, and many machine learning methods have been studied in the past. Based on these two statements, the following methods arise as candidates for solving the problem: ANN, random forest regression, KNN and SVR.

Artificial neural networks

Inspired by the biological neural networks, these networks process the information through a series of perceptrons that, because of their interconnections, will give a certain importance to different parts of the input information (Mitchell 1997; Bishop 2006). From the simpler to the more complex, they all are made of (i) similar smaller units (perceptrons) that behave in the same way as the others in the same network and (ii) connections (synapses) that define how the perceptrons interact with each other. Neural networks can have multiple inputs and outputs and can be applied to either classification or regression problems.

On the smaller scale (input-synapse-perceptron-output), the synapses connect the inputs to the perceptron, multiplying them by a weight $\mathbf{w}_i^k$, where $\mathbf{w}^k$ is the set of weights in the layer k. The perceptron can be described as a small machine that processes the information it is given. It takes the information given by each synapse connected to it, $\mathbf{x}_i\mathbf{w}_i$, and proceeds to their sum. To the result is applied an activation function, introducing nonlinearity, and the result of this operation is the output of the perceptron. In each layer of the network, all inputs must be connected to all the perceptrons. If a certain input is not relevant to the calculations, the algorithm will find a small weight for that synapse.

Backpropagation is the most commonly used learning algorithm, and it calculates each weight based on the difference between the output of each iteration and the target value observed. Other learning algorithms are described in Kingma & Ba (2015) and Liu & Nocedal (1989).

The design of a neural network can be divided into two steps. The first deals with the network's morphology, i.e. the number of layers and the number of neurons in each of them, as well as the activation function applied at each neuron. A collection of neural network architectures and its description is presented by van Veen (2016). There, the feed forward neural networks (FFNN) are described as simple and practical and are used in this work for ease of implementation. FFNN are trained neural networks in which the new information travels in from the input to the output. Svozil *et al.* (1997) discuss some advantages of this method, namely its learning process being autonomous from the user, its application in non-linear problems, the resistance to noise and the fact that each input set generates a trained model fully adapted and adequate to that same problem, preserving the idea that no two problems should have the same solution. They reported the slow convergence and unpredictability associated with difficult interpretation of results associated with this method as the major disadvantages.

Using a non-linear activation function means that the output of a neuron cannot be expressed as a linear combination of its inputs. Without this step, every neuron would output a combination of its inputs and, in the end, the solution would itself be a combination of the initial inputs. A neural network without non-linear activation functions is a neural network that can be simplified to a single layer. For simple problems that can be described with a linear model, the generic need for activation function is fulfilled using the identity function $f(\mathrm{x}) = \mathrm{x}$. The rectifier function is defined by:

$$\mathrm{f}(x) = \begin{cases} 0, & x < 0 \\ x, & x \geq 0 \end{cases} \tag{1}$$

and assures the non-linearity with low computational effort. It assures that the output of the perceptron has a positive infinite range. Somewhat similar to each other, the logistic and the hyperbolic tangent functions, respectively:

$$\mathrm{f(x)} = \frac{1}{1 + \mathrm{e}^{-\mathrm{x}}} \text{ and } \mathrm{f}(x) = \tanh(x), \tag{2}$$

assure that very large positive or negative numbers are approximated to the same value (1 if x is a large positive and 0 (logistic) or -1 (tanh) if x is a large negative). It also ensures that around $x = 0$, $\mathrm{f}(x)$ is approximately a linear function.

The second consideration relates to the training process occurring in the neural network, embracing the learning algorithm and learning rate. Gradient descent is an optimization method often used in machine learning problems, where it is used to find a minimum of the cost function $\mathrm{f}(\boldsymbol{\theta})$. Until convergence it iteratively calculates:

$$\theta^{i} = \theta^{i-1} - \alpha \nabla \mathrm{f}(\theta^{i-1}) \tag{3}$$

where $\boldsymbol{\theta}^{i}$ are the model fitting parameters found at the ith iteration, α is the learning rate and ∇ represents the gradient operator. The use of a constant learning rate carries two possible unwanted outcomes. Too small and it converges unnecessarily slowly; too large and it may fail to converge. Choosing the learning rate is frequently carried out by trial and error. Alternatively, one can use an adaptive learning rate as an attempt to avoid these issues. The adaptive moment estimation (Adam) (Kingma & Ba 2015) method is an adaptation to the gradient descent, as it recalculates the learning rate at each iteration, applying exponentially decaying average of previously observed gradients of first and second order. Another possible way to bypass the disadvantages of stochastic gradient descent (SGD) is to use a second-order optimization method such as the Broyden–Fletcher–Goldfarb–Shanno (BFGS) algorithm described in Liu & Nocedal (1989), where the vector $\boldsymbol{\theta}^{i}$ is calculated based on the hessian of $\mathrm{f}(\boldsymbol{\theta})$. In each step, the learning rate is updated such that its value assures that $\mathrm{f}(\boldsymbol{\theta}^{i+1})/\mathrm{f}(\boldsymbol{\theta}^{i})$ is smaller than a stipulated value. The BFGS algorithm generically guarantees that fewer iterations are required, but because of the heavy calculations associated with the hessian matrix, those take more computational time than SGD. Limited memory BFGS (LBFGS) is an implementation of BFGS designed to overcome this issue (Byrd *et al.* 1995).

Random forest regression

A regression tree is a method that was first developed to solve classification problems. It was later adapted to regression problems, although it does not provide a continuous output space. When given a new sample, the model will proceed to a series of comparisons starting from the root of the tree and following the path that respects all comparisons made. When it reaches a node with no splits, the process finishes and the output is calculated according to that node (Bishop 2006).

The method starts by mapping each sample into an N-dimensional space, where N is the number of features thought to influence the phenomenon. Then, along each direction, the data is split in a way that minimizes the mean square error in each of the two newly generated regions. This process stops if a certain error criterion is satisfied or when the maximum depth of the tree is reached. The deeper the tree, the better the fitting, thus opening the possibility for overfitting. The tree is built in such a way that by navigating through it, all training samples are correctly classified.

A random forest is a collection of random trees, the output of the model being the average of the individual outputs of each tree. In this case, each individual tree takes a portion of the data, resulting in slightly different trees. This technique is used as a way to minimize noise-induced errors.

Instance-based learning and K-nearest neighbors

Unlike other methods, the instance-based methods are based on storing training examples (Mitchell 1997). When evaluating a new instance, these methods compare its information with one of the examples stored in its memory, outputting a value close to the most similar instances studied. Because it does not have a global target function, this type of algorithm needs to evaluate each new instance, making it a slower solution if multiple instances are given at short intervals. For the same reason, the algorithm needs no initial training other than memorization, making it easier to implement. Due to its implementation, the target function is replaced by simpler local target functions. Instance-based algorithms can have multiple forms, such as k-nearest neighbor, locally weighted regression or radial basis function networks, being applicable to classification and regression problems.

The KNN algorithm makes a forecast by making a weighted average of the k vectors most similar to the input vector currently being assessed. Given the input vector $\mathbf{x}^i$, the distance between it and each vector present in memory is calculated (Bishop 2006). The Minkowski distance (The SciPy Community 2017) can be used to calculate the Manhattan distance and the Euclidean distance. The distance function is important in this method not only to find the nearest vectors, but also to find the weights later assigned to them. The output of the algorithm is a weighted average of the k vectors found, where the weights are usually proportional to the distance to the input vector. Alternatively, using uniform weights means that each of the k vectors is assumed to have an equal impact on the outcome. This method is simple and has fast training, and for this reason it is usually one of the first methods tested when studying a machine learning problem. However, other methods often surpass this method when accuracy is more important than simplicity.

Support vector regression

Support vector machines (SVM) are classification algorithms that apply a non-linear transformation to the input (Bishop 2006). Therefore, the SVM transforms the space where the two classes are only separable by a non-straight line into a new space where it is now possible to separate the classes using a straight line, also called a hyperplane for higher-dimension problems. The desired transformation function is called kernel, and it takes an n-dimensional input and gives an $(n+1)$-dimensional output, where the two classes will presumably be linearly separable. Several types of kernel function can be used, such as circular, spherical, linear, polynomial or hyperbolic, just to name a few.

Multiple class SVM can be achieved by a list of adapted methods, either by finding a new single objective function or by running a binary classification SVM for each identified class using the remaining classes as negative examples (one-versus-the-rest) (Bishop 2006).

For regression problems, SVR can be used. The idea is similar to that of SVM, using a kernel function to transform a non-linear into a linear dataset, where the equivalent of a maximum margin hyperplane is calculated. When a new vector $\mathbf{x}$ is used as input, the output is calculated by computing $\mathrm{f}(\mathbf{x})$. Immediately a disadvantage over other machine learning arises. While other methods can have multiple outputs, SVR only allows single outputs.

Given a dataset, it is possible that more than one hyperplane correctly classifies all data points, but the desired solution is the hyperplane that is equally distant from both classes, providing a better generalization, necessary for new data. After the hyperplane $\mathrm{f}(\mathbf{x})$ is found, a classification problem is tackled by computing f($\mathbf{x}$) followed by:

$$\begin{cases} \mathbf{x} \in \mathrm{A}, & \text{if } \mathrm{f}(\mathbf{x}) > +1 \\ \mathbf{x} \in \mathrm{B}, & \text{if } \mathrm{f}(\mathbf{x}) < -1 \end{cases} \tag{4}$$

where A and B are the two classes. More generically, in a regression problem, the output is simply given by f($\mathbf{x}$) for each new instance $\mathbf{x}$ being evaluated.

Creating the model

Although the best model is generally thought to be the one that presents smaller differences between the forecast and the observation, that is not always the case. Often the datasets used for the training of the algorithm contain measurement errors, noise, or random unpredictable occurrences. On the WSS, leakages, sporadic events or urban fires are some examples. Adjusting the model to fit these events will result in forecast failure. The sample error of an overfit model is smaller than that of a more general model, but the true error tends to get smaller on a generic model. To avoid this issue, known as overfitting, one can use strategies such as early stopping of the learning process or using separate sets of data for training and testing (Mitchell 1997).

The methods used to predict water consumption usually consider a previous period of about two years. Holidays certainly have a high impact on the water demand of the network, but a two-year registry designed to avoid overfitting by simply eliminating outliers will fail to predict those events. The experience and sensibility of the engineer are crucial when designing the model.

Selection and configuration of the models

As shown in the literature, different techniques are better suited for different systems. For this reason, it is not expected to find a solution that perfectly fits all datasets, or to find the perfect solution for each methodology. However, it is expected to find which model configurations being tested present the best results. The present strategy contains a large set of models varying only one parameter between them in order to evaluate the influence of each parameter. Only the parameters that are expected to have the largest impact on the definition of the model are considered.

A neural network is essentially defined by the shape of the network itself (number of neurons and number of layers), activation function and learning algorithm. Nine shapes of networks will be tested, being those combinations of three numbers of layers (2, 5, 8) and three numbers of neurons per layer (10, 25, 75). Three activation functions will be studied: identity, logistic sigmoid and rectified linear unit. The learning algorithms to be tested are SGD, LBFGS and Adam.

The SVR machines are highly dependent on the kernel they use. Radial based function (RBF), linear and polynomial (second degree) are the kernels tested. Two values of tolerance (0.01 and 0.001) for stopping each learning iteration are also tested.

The KNN method is strongly defined by the number of neighbors considered relevant to the calculations and by how the weights are calculated. Tests with three numbers of neighbors (2, 5, 8) and two weight functions (uniform and distance) are considered. The uniform weight assigns the same weight to the *k* neighbors considered, while the distance weight function gives a weight proportionally inverse to the Euclidean distance between each neighbor and the data.

The random forest method can be modeled by the number of trees in the forest and the minimal number of samples required at each split. With this in mind, six combinations of three numbers of trees (2, 8, 15) and two numbers of samples required to split (2, 8) are analyzed.

In total, 99 model configurations are tested, being 81 ANN, six SVR, six KNN and six random forest regressor (RFR).

Implementing the models

The Scikit-learn 0.18.1 (Pedregosa *et al.* 2011) library for Python 3 (Python Software Foundation (US) 2017) allows the creation of machine learning models in a simple and efficient way. Other libraries from the SciPy Community (2017) environment ease the data manipulation (NumPy and pandas) and the data visualization (Matplotlib). The overall algorithm is schematically represented in Figure 1. It starts by reading the data file. Then, it applies a filtering routine that eliminates outliers, missing values and normalizes the data, and rearranges the data in a three-dimensional matrix. Although other shapes for this matrix would be

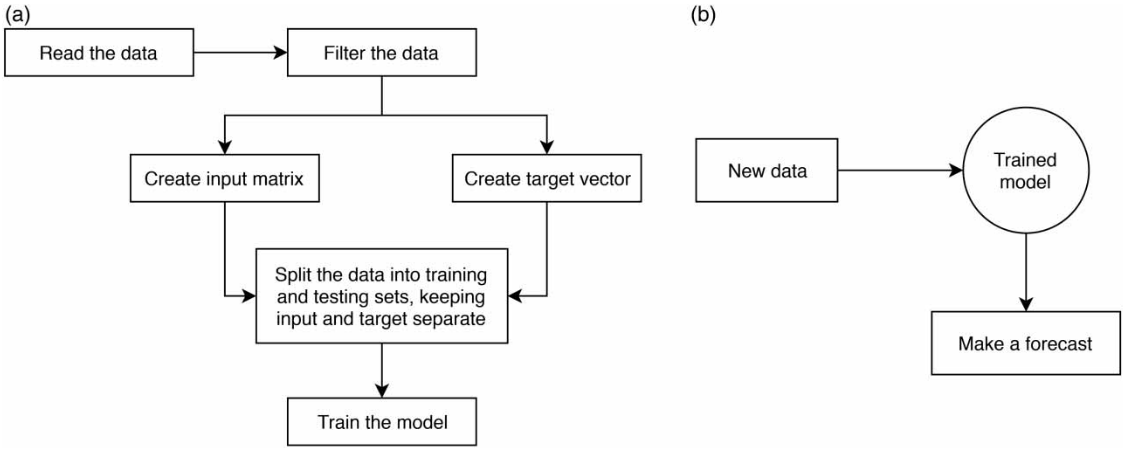

Figure 1 | Schematic representation of the machine-learning algorithm used: (a) training the model and (b) forecasting procedure.

possible, this presents an easier visualization of the data matrix. In the three-dimensional matrix represented in Figure 2, each horizontal plane represents an hour of the periodicity considered, where each line represents a sample in the training data and each column represents a feature. If the periodicity is of 24 h, a sample is 1 day; however, if the periodicity is of 168 h, a sample is a week.

In addition to the input data matrix, a target matrix is created. This target matrix only has one column as features – the observed consumption and weather data – when existent. The algorithm then divides the available data into training and testing sets, reserving the last split for testing. Cross-validation is not used because the data is time sensitive and altering the order of the data would affect the training and the results. A number of machines equal to the height of the matrix (periodicity) is created and trained using the two-dimensional matrix corresponding to the *T*th plane of the larger matrix as input data and its corresponding target vector in the training set. This way, the *T*th machine predicts the *T*th hour of the periodicity considered and stores its value in the *T*th element of a one-dimensional 'Prediction' vector. Finally, the program compares the forecast result (for the period being studied) with the testing data for the *D* days in the testing set.

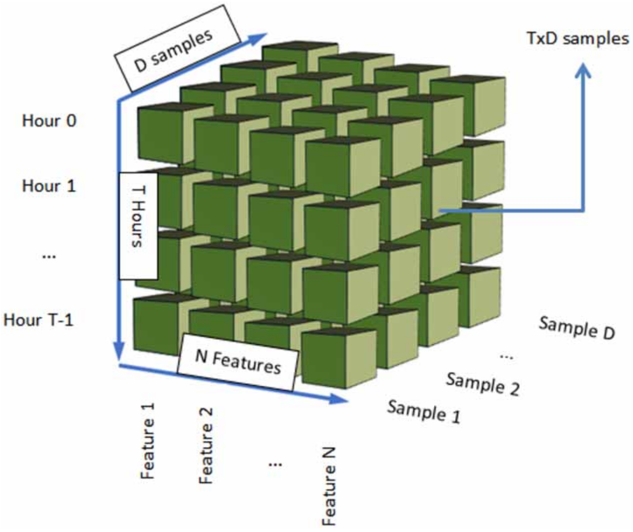

Figure 2 | Representation of the data matrix used in the machine learning models. Each sample has *N* features. Adapted from Qian & Pan (2017).

Evaluating the performance of the models

Even though the main goal of machine learning algorithms is similar in this work – to mimic a real system – there are multiple performance criteria. While some methods are better at finding the overall pattern, ignoring occasional peaks, others can give a better understanding of sporadic events. When measuring the performance of a statistical experiment such as a demand forecast, there are two major dimensions: (1) how it describes the general tendency of the phenomenon and (2) how it behaves when it encounters possible random outliers and noise. In machine learning, the definition of the model and its parameters/weights is dependent on the performance criteria chosen for the training stage. Different criteria result in different model weights.

For the purpose of this paper, when fitting a model, the RMSE between the observations and the estimations is minimized. The RMSE values are affected by the normalization of the data, being presented dimensionless.

The performance of the models was evaluated using the standard statistical metrics RMSE and MAPE. However, other metrics can be useful. The mean error (ME), also called mean bias, can give an idea of whether the forecasts are above or under the objective. The coefficient of determination R^2 is a measure of how the difference between the observations and the forecasts relates to the difference between the observations and their average. It can be interpreted as the likelihood that new values are going to be correctly predicted (The Pennsylvania State University 2017).

Ensemble of forecasting strategies

The advantages and disadvantages of each presented forecasting technique were previously discussed. The advantages of each technique can be combined through a hybrid strategy of parallel methods, diluting the disadvantages. In this work, a weighted parallel strategy is suggested.

Considering that some methods tend to overestimate the demand and others tend to underestimate it, one can improve the forecast by calculating a weighted average of the various forecasts previously made (i.e. one for each model tested). The weight of each model must reflect the robustness of the forecast made. Attributing the weights to the desired models is not just a matter of evaluating the errors and assigning a higher weight to those with smaller errors, but of combining over- and under-estimations in a way that diminishes the error. A thorough analysis must be carried out to understand the models that over- and underestimate the forecast and its magnitude.

The methodology hereby proposed includes several steps. The first step includes the training and validation made by the models from the previously discussed set. The performance of each model is considered as the average of the results found for each individual day in the validation set. In the next step, $n_{\mathrm{ME}+}$ models with positive mean error and $n_{\mathrm{ME}-}$ models with negative mean error are chosen, and $n_{\mathrm{ME}} = n_{\mathrm{ME}-} + n_{\mathrm{ME}+}$. In each case, the chosen models are those that present the best RMSE, MAPE% and R^2, respectively. This methodology presents a new forecast, based solely on a combination of the forecasts previously made. Each of the selected models has an associated weight proportionally inverse to the absolute value of its ME and that balances the number of $n_{\mathrm{ME}+}$ and $n_{\mathrm{ME}-}$ models. Therefore, the models with lower errors have a higher impact on the forecast. For that, each weight w_i is the inverse of the ME given by its corresponding model. The final forecast is given by:

$$\mathrm{F} = \sum_{i=1}^{n_{ME}} (w_i \mathrm{F}_i), \tag{5}$$

where:

$$w_i = \frac{|1/ME_i|^2 a_i}{\sum_{\mathrm{i}=1}^{n_{ME}} (|1/ME_i|^2 a_i)} \text{ with } \sum_{\mathrm{i}=1}^{n_{ME}} w_i = 1, \tag{6}$$

and:

$$a_i = \begin{cases} \dfrac{\sum_{\mathrm{i}=1}^{n_{ME+}} |1/ME_i|}{\sum_{\mathrm{i}=1}^{n_{ME-}} |1/ME_i|} & \mathit{if}\ ME_i < 0 \\ 1 & \mathit{if}\ ME_i \geq 0 \end{cases} \tag{7}$$

The advantage of this efficient weighted parallel forecasting strategy is its simplicity and low computational cost. The computational effort of this strategy is very low because it makes use of the already made calculations of the training-validation of the individual models.

VALIDATION OF THE MACHINE LEARNING ALGORITHMS DEVELOPED USING BENCHMARKS

To effectively use the developed algorithm for predicting water consumption, one must make sure the algorithm achieves its goal, and with advantages compared with other algorithms. By comparing the results achieved by the developed algorithm with those presented by other algorithms, one can assess the viability, applicability and quality of the forecasting program algorithm. The method considered for comparison is the autoregressive integrated moving average (ARIMA), recommended for time series problems.

The analysis of the algorithms can also be made using benchmarks (Hyndman 2010). Researchers and developers frequently use the same benchmarks, for reasons such as availability and ease of comparison of results.

The evaluation of the developed models is carried out in two consecutive steps. First, each sample of the testing set data is evaluated, and second, the average for each metric is calculated considering the entire testing set. The metrics evaluated are the ME, MAE, RMSE, MAPE% and R^2, but only the RMSE (used in the fitting process), MAPE% and R^2 are shown. To assess the performance of each methodology, the results for the best model in each family (RFR, KNN, SVR and ANN) considering both metrics are compared with those of the ARIMA methodology.

Sinusoidal periodic function

Consider a hypothetical network with a daily water demand pattern that repeats itself infinitely. This pattern has a valley during the night, a peak in the morning and another in the evening. Moreover, random noise and weekly and monthly seasonalities are added, changing the pattern for each day. The noise was added to transform the analytical data into more realistic observations and because the tests made resulted in identical and very precise results. In fact, most models achieved a perfect forecast, validating the implementation of the algorithms. The function used in this process is defined as:

$$\mathrm{Q}(t) = 50 + \frac{\sum_{i=1}^{5} \mathrm{f}_i(t)}{5} + \mathrm{GAUSSIAN}(0, 2) \quad (8)$$

where:

$$\mathrm{f}_i(\mathrm{t}) = \alpha_i \sin\left(\beta_i + \frac{2\,\pi}{\gamma_i}\mathrm{t}\right) \quad (9)$$

$\alpha = \{30, 30, 30, 20, 10\}$; $\beta = \{3, -3, 10, -3, 0\}$; $\gamma = \{12, 24, 24, 168, 744\}$ and GAUSSIAN(0,2) represents a Gaussian noise with mean 0 and standard deviation 2. The functions $\mathrm{f}_1(t)$, $\mathrm{f}_2(t)$ and $\mathrm{f}_3(t)$ model a daily behavior, since they have periods of 12, 24 and 24 h, respectively. The functions $\mathrm{f}_4(t)$ and $\mathrm{f}_5(t)$ represent weekly and monthly tendencies, defined by their periods of 168 and 744 h. One should note that the data was generated once and then was used across all the models tested. The dataset is the same in the different tests. The hourly data observed in the last 2 days is shown in Figure 3.

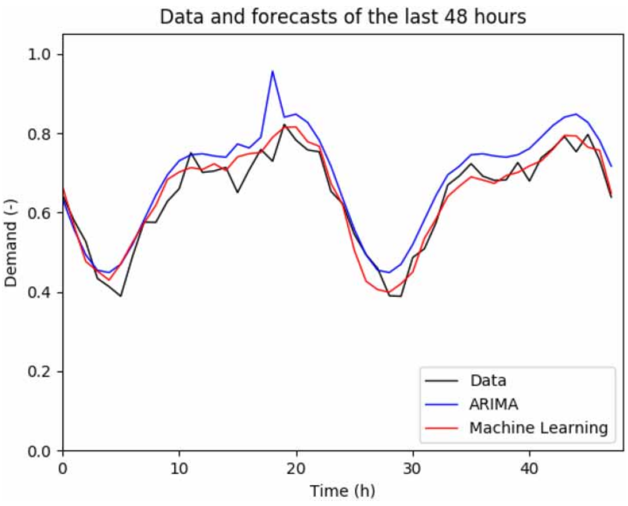

Figure 3 | Sinusoidal periodic function benchmark. Data generated for a water consumption observed in the last 2 days. Forecasts given by the ARIMA and KNN ($N = 8$ Euclidean) models.

Using the sinusoidal periodic function data, the performance of the 99 designed models was evaluated. Between the periodicity thought to rule the phenomenon and the past observations thought to influence the forecast, the influence of these two parameters is analyzed, fixing one of them and varying the other. Fixing the periodicity at 24 h, tests were made considering 3, 14 and 70 past observations. Defining the past observations at 14 samples, tests were made considering periodicities of 12, 24, 48 and 168 hours. The tests made without considering noise show perfect forecasts (RMSE = MAPE = 0 and $R^2 = 1$) for the RFR, KNN and SVR methods in every approach, except when using clustering (where the best model, ANN(identity(2×10) sgd), obtained RMSE = 0.0241). For the case when using Gaussian noise, the best models per approach are presented in Table 1. It is noticeable that the different approaches tested present similar results for all metrics shown, with the exception of the coefficient of determination when using 12-hour periods in the forecasts. This

Table 1 | Forecasting errors of the best model found with each approach tested for the sinusoidal periodic function benchmark

Periodicity (h)	Features	Model	RMSE (−)	MAPE (%)	R^2
24	Demand (3)	ANN(identity(2 × 25) lbfgs)	0.0392	4.7678	0.9086
24	Demand (14)	KNN($N = 8$ Euclidean)	0.0304	3.6355	0.9417
24	Demand (70)	KNN($N = 8$ Euclidean)	0.0284	3.4653	0.9476
12	Demand (14)	KNN($N = 8$ Euclidean)	0.0323	3.9408	0.6677
48	Demand (14)	KNN ($N = 8$ uniform)	0.0315	3.8536	0.9411
168	Demand (14)	KNN ($N = 8$ Euclidean)	0.0300	3.6170	0.9510
24	Demand, using clustering (14)	ANN(identity(2 × 10) sgd)	0.0408	4.8732	0.8919

behavior can be explained by comparing two consecutive periods generated by the function. The consumption patterns observed in the first 12 h and in the last 12 h of any given day are obviously different. Using the data of the first half of the day to predict the second half is not advised, as often stated by the literature and reinforced by the results hereby presented. The performance of the models improves when the number of features increases. However, the same cannot be said about the time scale of the forecasts. In fact, the lowest RMSE is found when using 70 water demand features with a 24 h periodicity. It is also observable that the KNN models are usually the best choice.

In Table 2, the overall best results of each family of models (RFR, SVR, KNN and ANN) and for the ARIMA are presented, along with the best three models, considering the best approach found in Table 1. The results obtained by the best machine learning models in any method are satisfactory, with their RMSE at least 32% lower than the one observed with the ARIMA. However, as proved by the results obtained by the worst model (a neural network using SGD as learning algorithm and the logistic activation function), the use of machine learning does not guarantee good results. Figure 3 presents the results obtained with the ARIMA and KNN ($N = 8$ Euclidean) models.

Table 2 | Models that achieved the (i) best RMSE per family, (ii) the overall best three RMSE, (iii) the overall worst RMSE, and (iv) the ARIMA results, applied to the sinusoidal periodic function benchmark using 70 × 24 h demand samples

Model	RMSE (−)	MAPE (%)	R^2
RFR ($N = 8$ $n = 2$)	0.0305	3.7452	0.9400
KNN($N = 8$ Euclidean)	0.0284	3.4653	0.9476
SVR(linear t = 0.001)	0.0352	4.3049	0.9210
ANN(relu(5 × 25) lbfgs)	0.0306	3.7358	0.9392
KNN($N = 8$ Euclidean)	0.0284	3.4653	0.9476
KNN($N = 8$ uniform)	0.0285	3.4647	0.9475
KNN($N = 5$ Euclidean)	0.0290	3.4713	0.9452
ANN(logistic(5 × 10) sgd)	0.1319	17.8044	−59.4768
ARIMA	0.0523	6.7217	0.8149

Cars benchmark

This benchmark is based on a dataset used in three Artificial Neural Network and Computational Intelligence Forecasting Competitions, held between 2009 and 2010 by the Lancaster University Management School. The database consists of a collection of traffic data, including highways, subways, flights, shipping imports, and railways. The entire dataset is presented in four parts of 1,735 instances plus five parts of 895 instances, but a quick analysis shows that these do not represent a pure sequence of data. For this reason, only one part with 1,735 is used. This means that only 72 days are available to test the machine learning models presented. The amount of data available brings an extra difficulty, derived from the small number of iterations during training.

The dataset used by this benchmark has the peculiarity of having just three months of registries, which is not usually advised due to issues related to incomplete or short training. For the same reason, this dataset does not have sufficient data to allow the study of clustering-based forecasts, or to study the approaches involving weekly periodicity or 70 past observations.

Table 3 | Forecasting errors of the best model found with each approach tested for the cars benchmark

Periodicity (h)	Features	Model	RMSE (−)	MAPE (%)	R^2
24	Demand (3)	KNN ($N = 2$ Euclidean)	0.1150	35.0550	0.5199
24	Demand (14)	KNN ($N = 2$ Euclidean)	0.0922	36.4973	0.6185
12	Demand (14)	KNN ($N = 2$ Euclidean)	0.0553	25.5196	0.8108
48	Demand (14)	KNN ($N = 2$ Euclidean)	0.1475	59.9765	0.4100

The results achieved by the best models for each approach are presented in Table 3. The consistency of the KNN methodology can be observed, in particular when it is configured with two neighbors and the Euclidean distance weight function, since this model is found to give the best results in every approach. It is also shown that a 12 h time window for training and forecasting offers the best performance considering any of the metrics presented.

In Figure 4 it can be observed that the data presents a very atypical behavior. This represents a great difficulty when training the models, as each new sample can potentially bring more noise with no contribution to the process, and for the prediction phase, as the new data has a high probability of being something the models have not previously been confronted with.

Table 4 also presents the results of the ARIMA method for comparison purposes. Except for the SVR, every other methodology presents at least one model that is better than the ARIMA considering any metric. As for the SVR, it gets particularly bad results in this benchmark, with its best model presenting errors about twice as bad as the best models in the other methodologies. Overall, the two best models are the KNN with Euclidean distance weight function. Note that the difference between the best and the second best models is much more accentuated than that between the second and the third best models.

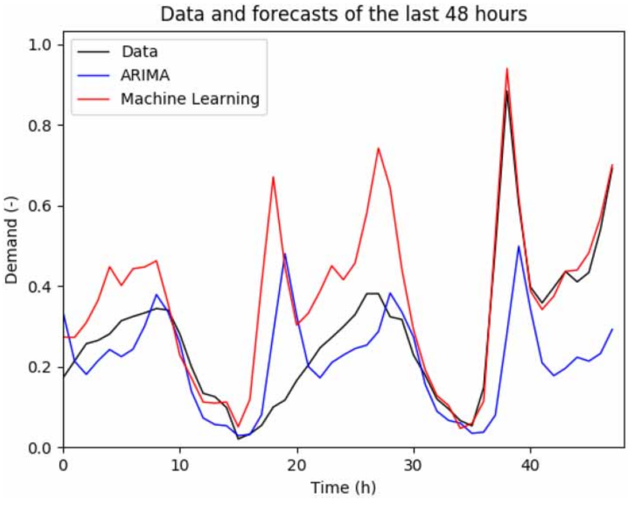

Figure 4 | Cars benchmark. Data observed in the last 2 days and the forecasts given by the ARIMA and KNN ($N = 2$ Euclidean) models.

Even though the SVR methodology did not achieve the expectations, one can conclude that using machine learning techniques proves to outperform the ARIMA in this benchmark.

Air quality benchmark

The air quality benchmark contains the data collected by equipment that measured the quality of the air at regular intervals of 1 hour in an Italian city. In total, 9,358 (389.91 days) measurements were registered. The data considered in the calculations was, however, reduced to ensure that it has a length divisible by the periodicity considered in each test (9,336 registries used for the periodicity of 24 h). When

Table 4 | Models that achieved (i) the best RMSE per family, (ii) the overall best three RMSE and (iii) the overall worst RMSE, and (iv) the ARIMA results, applied to the cars benchmark, using 14 × 12 h demand samples

Model	RMSE (−)	MAPE (%)	R^2
RFR($N = 8$ $n = 2$)	0.0769	35.5161	0.6288
KNN($N = 2$ Euclidean)	0.0553	25.5197	0.8108
SVR(linear t = 0.001)	0.1264	68.2466	0.0076
ANN(relu(2 × 75) lbfgs)	0.0637	29.3742	0.7841
KNN($N = 2$ Euclidean)	0.0398	25.5197	0.8108
KNN($N = 5$ Euclidean)	0.0637	29.0592	0.7623
ANN(relu(2 × 75) lbfgs)	0.0637	29.3742	0.7841
ANN(identity(5 × 75) adam)	0.3128	147.5146	−0.4758
ARIMA	0.1255	48.2089	0.2121

predicting water demand, the method proposed considers a maximum of two types of features (past demand and a meteorological variable). For this reason, for the benchmark tests using the air quality dataset, only two out of the 14 types of features available were selected: true hourly averaged NOx concentration in ppb (reference analyzer) and temperature in °C. The average NOx concentration feature was chosen because its range is similar to that of a typical water demand in cubic meters. The data presents some values of −200, specifically used to avoid implementation errors associated with missing values. However, knowing that these are outliers, they are submitted to a filtering routine described below under 'Sources of data'. For availability reasons, this is also the only benchmark that considers meteorological features, bringing it closer to the real applications intended for the developed methodology and program.

For this benchmark the available data allowed more tests to be made, including tests using weather features, which have not been evaluated previously. Therefore, adding to the tests presented in the first benchmark, two tests using temperature features were also made. Fourteen temperature features were considered for one test and just one temperature feature was considered for the other. Both consider a periodicity of 24 h and 14 demand features.

Table 5 confirms that using a periodicity of 12 hours brings the best results. It also shows that for this benchmark's database, the best methods are neural networks. The use of weather features did not bring an improvement in the performance, and the increase of the periodicity clearly improves the R^2, but not the RMSE or the MAPE%.

The best models in each method are shown in Table 6. This table allows us to conclude that for this benchmark the R^2 achieved are particularly low. This suggests that training the models with the objective of maximizing R^2 would probably bring better overall results, supported by the fact that some models produce forecasts with a much better R^2 with little prejudice to the RMSE (third and fourth entries in Table 5). However, the RMSE results are approximately those observed previously. In this benchmark, it is notable that the range between the best and the worst models' results is less than 40% of the worst RMSE. As before, machine learning methods proved to find better solutions than the ARIMA. Figure 5 illustrates the best machine learning model (ANN identity(8 × 10) lbfgs) in comparison with the ARIMA for the air quality benchmark. The ARIMA model shows a tendency to overestimate the real demand.

Table 6 | Models that achieved the (i) best RMSE per family, (ii) best three RMSE overall and (iii) worst RMSE overall, and (iv) ARIMA results, applied to the air quality benchmark, using 14 × 12 h demand features

Model	RMSE (−)	MAPE (%)	R^2
RFR($N = 5$ $n = 8$)	0.0800	47.4656	−0.2735
KNN($N = 8$ Euclidean)	0.0772	45.8370	−0.0914
SVR(rbf t = 0.01)	0.0680	40.1903	−0.5240
ANN(identity(8 × 10) lbfgs)	0.0655	35.8837	−0.3904
ANN(identity(8 × 10) lbfgs)	0.0655	35.8837	−0.3904
ANN(identity(8 × 25) lbfgs)	0.0666	37.2693	−0.9076
ANN(identity(2 × 10) lbfgs)	0.0672	37.5903	−0.5068
ANN(logistic(8 × 75) adam)	0.1082	60.2676	−2.9964
ARIMA	0.0919	54.5420	−1.2770

Table 5 | Forecasting errors of the best model found with each approach tested for the air quality dataset

Periodicity (h)	Features	Model	RMSE (−)	MAPE (%)	R^2
24	Demand (3)	ANN(identity(2 × 75) sgd)	0.0705	34.9673	−3.4301
24	Demand (14)	ANN(identity(2 × 25) adam)	0.0723	35.4680	−0.0730
24	Demand (70)	ANN(identity(8 × 10) lbfgs)	0.0664	44.9123	0.0571
12	Demand (14)	ANN(identity(8 × 10) lbfgs)	0.0655	35.8837	−0.3904
48	Demand (14)	ANN(identity(8 × 25) lbfgs)	0.0884	51.4168	0.1180
168	Demand (14)	ANN(relu(2 × 10) adam)	0.0770	30.4780	0.5630
24	Demand (14), Temperature (14)	SVR(linear t = 0.01)	0.1157	37.7114	0.0338
24	Demand (14), Temperature (1)	ANN(relu(2 × 10) adam)	0.1134	36.0354	−0.3135

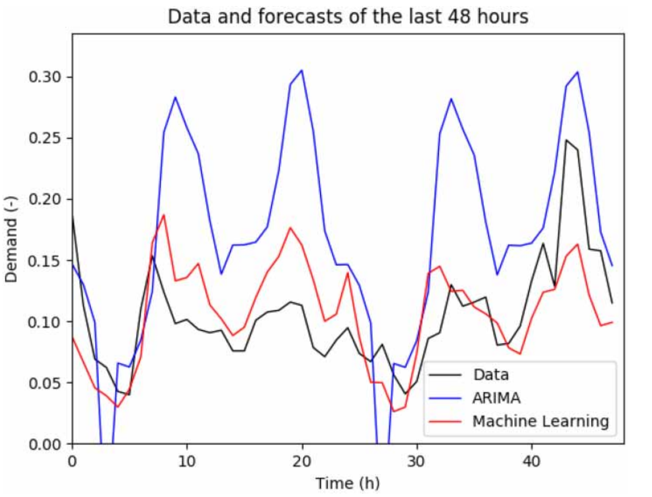

Figure 5 | Air quality benchmark. Data observed in the last 2 days and the forecasts given by the ARIMA and ANN(identity(8 × 10) lbfgs) models.

The benchmark tests allowed the conclusion that the developed algorithms, specifically the machine learning strategies, are capable of producing predictions based on the previous observations and existing patterns in the data. In most cases, the machine learning methods can produce more accurate forecasts than ARIMA, which is a standard method often used in forecasting. However, for different datasets, the best forecasts are often produced by different models. For this reason, it is always important to test different methods and models when a new database is being analyzed.

APPLYING THE MACHINE LEARNING ALGORITHMS TO WATER SUPPLY SYSTEMS

The models previously described are applied to three databases provided by two Portuguese water utilities. Both companies store their data in similar ways. The cumulative amount of water that passes through any node of its network is saved, meaning the water demand in a determined period is the difference between the cumulative data observed at the extremities of that interval. The data was provided in raw, requiring a filtering step.

Sources of data

The first water utility – Water Utility 1 – is located in the north part of Portugal and is responsible for the water collection, treatment and distribution in an area of more than 2,500 km^2 serving more than 1.5 million people. This company provided data concerning four points of its network, but due to the errors found, only two are used in this work – WD2 and WD4. Visual representations of the WD2 and WD4 data (the last 48 hours) can be seen in Figures 6 and 7, respectively. The WD2 and WD4 datasets correspond to dates between 21/09/2012 @ 00:00 and 05/07/2013 @ 23:00 in an hourly frequency (total: 6,912

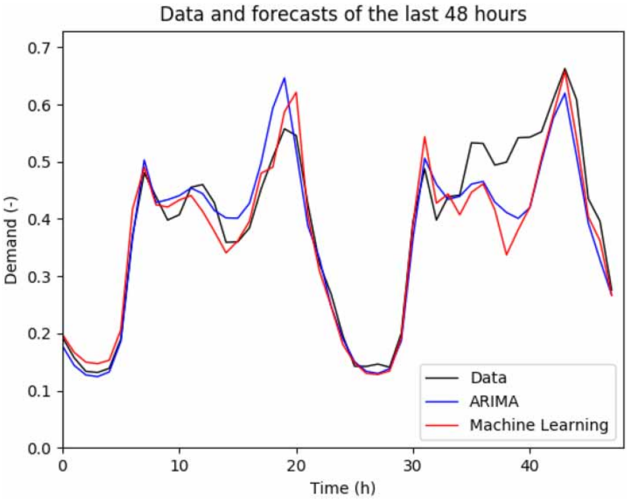

Figure 6 | WD2 data of Water Utility 1. Normalized water consumption observed in the last 2 days. Forecasts given by the ARIMA and ANN(identity(8 × 10) lbfgs) models.

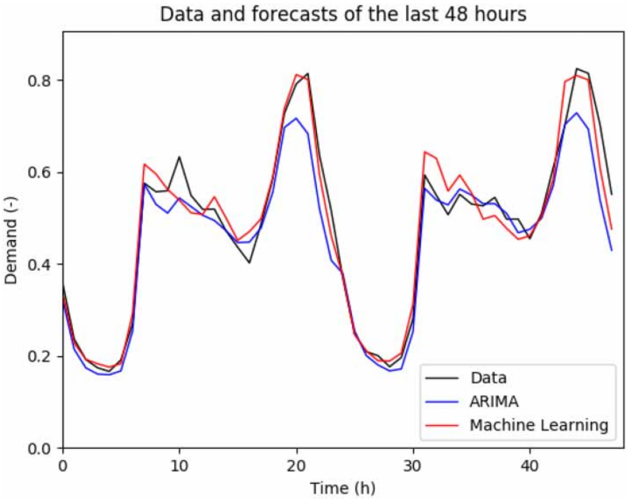

Figure 7 | WD4 data of Water Utility 1. Normalized water consumption observed in the last 2 days. Forecasts given by the ARIMA and ANN(relu(8 × 25) lbfgs) models.

observations). From those, 96.875% were used in training (6,696 observations, 279 days) and 3.125% were used in testing (216 observations, 9 days). After the filtering process, the WD2 and WD4 datasets were normalized from the ranges [0; 34,67] and [0; 97,71] to the interval [0, 1] (m^3/h).

The second real data comes from a water utility located in central Portugal – Water Utility 2 – responsible for supplying water to over 20,000 customers. This company provided data referent to its entire network but with an evident lack of data in some points. In other points of the network, the existent data shows excessive errors. For this reason, only the data of one point of the network (consumption for the Ançã region) will be considered to train the models. The last 48 h of this dataset are represented in Figure 8. The collected data, from 1 September 2015 to 18 December 2016, was provided in flow rates measured in time intervals of 1 hour (total: 8,448 observations). All the 15 months were used. The dataset was divided into 96.875% for training and 3.125% for validation/testing, corresponding to 8,184 observations (341 days) for training and 264 (11 days) for validation/testing. The Water Utility 2 dataset was normalized from the range [0; 106] to the interval [0, 1] (m^3/h).

The observation of the data given by both companies allows a few problems to be identified regarding either the presence of outliers or the absence of data. All the values that did not fall in the range between the average and a margin of 3× the standard deviation were considered as outliers. Therefore, all values that lay above or under the boundaries were assigned to the outliers. When no values were found at any given instant, the algorithm assigned the overall average to those instances. A second iteration of this process is applied to reduce the impact of the errors detected before the first iteration. After the correction of all outliers and missing values, a normalization is applied, having as reference the corresponding maximum value. Therefore, each variable becomes dimensionless and consequently has the same relative importance.

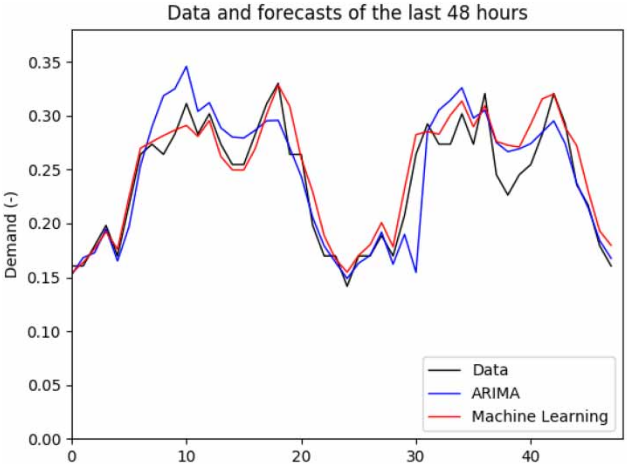

Figure 8 | Water Utility 2 data. Normalized water consumption observed in the last 2 days. Forecasts given by the ARIMA and ANN(identity(2 × 10) lbfgs) models.

Results for Water Utility 1

The 99 designed models were evaluated according to 11 approaches, including one considering clustering, two considering temperature history and two considering rain occurrence history.

The results obtained by the best model for each periodicity and features approach are presented in Table 7. The 12 h periodicity gives the best RMSE (the objective function) by a small margin but also presents the worst R^2 and the third worst MAPE%. However, because the fitting process is performed using the RMSE, this metric must be considered when comparing the models' performance. Therefore, the best model is the ANN(identity(8 × 10) lbfgs) with a periodicity of 12 h and 14 demand features. Increasing the periodicity or decreasing the number of features in the forecasts worsens their RMSE results. Although it is predictable that the number of features increases the quality of the machine learning model, it was expected that a periodicity of 24 h would produce a better forecast than the 12 h. However, observing the data represented in Figure 6, it can be seen that this ANN model can predict the water demands quite well, with a smaller error than the ARIMA method. Concerning the weather features, using temperature or rain occurrence features presents the same results for the three error measures, suggesting a high correlation between the temperature and the occurrence of rain in any specific period. However, using fewer weather features seems to bring benefits to the forecasts when considering the RMSE. When considering the MAPE error, the use of weather input is beneficial, the models that consider one feature of weather being the best. Consequently, in these results, it is not clear if it is advisable or not to use weather features. The best models are

Table 7 | Forecasting errors of the best model found with each approach using the WD2 database of Water Utility 1

Periodicity (h)	Features	Model	RMSE (−)	MAPE (%)	R^2
24	Demand (3)	ANN(identity(8 × 10) lbfgs)	0.0669	11.1843	0.7582
24	Demand (14)	ANN(relu(2 × 75) lbfgs)	0.0554	10.0918	0.8534
24	Demand (70)	ANN(identity(8 × 25) lbfgs)	0.0561	9.7884	0.8609
12	Demand (14)	ANN(identity(8 × 10) lbfgs)	0.0532	10.8467	0.7331
48	Demand (14)	ANN(identity(8 × 10) lbfgs)	0.0605	10.7018	0.8322
168	Demand (14)	ANN(relu(2 × 25) adam)	0.0590	12.0670	0.8700
24	Demand, using clustering (14)	ANN(identity(5 × 10) lbfgs)	0.0624	11.2067	0.8040
24	Demand (14), Temperature (14)	ANN(relu(5 × 10) lbfgs)	0.0684	10.2677	0.8564
24	Demand (14), Temperature (1)	ANN(relu(5 × 10) lbfgs)	0.0678	9.5399	0.8567
24	Demand (14), Rain Occurrence (14)	ANN(relu(5 × 10) lbfgs)	0.0684	10.2677	0.8564
24	Demand (14), Rain Occurrence (1)	ANN(relu(5 × 10) lbfgs)	0.0678	9.5399	0.8567

neural networks with the LBFGS learning algorithm with identity or rectified linear unit activation functions.

In Table 8 the best results obtained by each method are presented, considering no weather features and 14 registries of past water demand with a 12 h periodicity. A deeper look considering all metrics reveals that the KNN models' performance has a clear tendency toward improving with the number of neighbors. The results show no significant difference between the weight functions tested, although the results are slightly better when using the Euclidean distance. The SVR method is less dependent on the tolerance used, since changing that parameter has an insignificant impact on any metric, across all approaches. The selection of the kernel appears to be specific to each approach, since no particular kernel is consistently the best solution. At the same time, no particular kernel presents particularly bad results. Nonetheless, the kernel has a larger importance in the forecasts than the tolerance. Concerning the RFR models, expanding the size of the forest (number of trees) has a positive impact on the quality of the forecasts. The same can be said about the number of required samples at each split. RFR ($N = 8$ $n = 8$) is the best RFR in most approaches. The worst 12 ANN models use SGD, and of those, nine use the logistic activation function. The 12 best ANN models use the LBFGS learning algorithm and none of them uses the logistic activation function. Therefore, it is advised not to use the SGD and logistic in comparison to the LBFGS. The shape of the network has a smaller importance in the outcome of the forecasts, but smaller networks seem to result in better performance. All methods presented better forecasts than the ARIMA (considering RMSE). Figure 6 presents the forecasts made by the ARIMA and ANN(identity(8 × 10) lbfgs) models in this dataset.

Table 8 | Models that achieved the (i) best RMSE per family, (ii) best three RMSE overall and (iii) worst RMSE overall, and (iv) ARIMA results, applied to the WD2 database of Water Utility 1, using 14 × 12 h demand features

Model	RMSE (−)	MAPE (%)	R^2
RFR($N = 8$ $n = 8$)	0.0594	11.1305	0.6854
KNN($N = 8$ uniform)	0.0556	10.8876	0.7150
SVR(linear t = 0.01)	0.0543	11.4832	0.6552
ANN(identity(8 × 10) lbfgs)	0.0532	10.8467	0.7331
ANN(identity(8 × 10) lbfgs)	0.0532	10.8467	0.7331
ANN(identity(2 × 25) lbfgs)	0.0533	10.9579	0.7194
ANN(identity(8 × 25) lbfgs)	0.0536	10.6897	0.7220
ANN(logistic(5 × 10) sgd)	0.1677	48.3851	−832.8060
ARIMA	0.0644	10.8487	0.8390

When forecasting the water demand in the second subnetwork (Table 9), the best results are found for a periodicity of 24 h and 14 demand features. This result is expected, although it is different from the result of the previous subnetwork, indicating a model dependence of the case study. In this case, the use of 14 weather features seems to be better than the case of using just one, but worse than the case that does not use this feature when comparing using the RMSE. If the R^2 criterion is taken into

Table 9 | Forecasting errors of the best model found with each approach using the WD4 database of Water Utility 1

Periodicity (h)	Features	Model	RMSE (–)	MAPE (%)	R^2
24	Demand (3)	ANN(identity(5 × 25) sgd)	0.0762	12.7028	0.8226
24	Demand (14)	ANN(relu(8 × 25) lbfgs)	0.0473	7.8511	0.9229
24	Demand (70)	ANN(identity(2 × 25) lbfgs)	0.0546	8.9986	0.9080
12	Demand (14)	ANN(identity(8 × 10) lbfgs)	0.0490	8.5004	0.8143
48	Demand (14)	ANN(relu(5 × 25) lbfgs)	0.0575	9.5678	0.8995
168	Demand (14)	ANN(identity(2 × 25) lbfgs)	0.0610	10.2320	0.8980
24	Demand, using clustering (14)	ANN(identity(8 × 10) sgd)	0.0590	10.1124	0.8937
24	Demand (14), Temperature (14)	ANN(identity(8 × 10) lbfgs)	0.0523	7.8589	0.9271
24	Demand (14), Temperature (1)	ANN(relu(2 × 10) lbfgs)	0.0530	8.0717	0.9240
24	Demand (14), Rain Occurrence (14)	ANN(identity(8 × 10) lbfgs)	0.0523	7.8589	0.9271
24	Demand (14), Rain Occurrence (1)	ANN(relu(2 × 10) lbfgs)	0.0530	8.0717	0.9240

account, the result using 14 weather features is the best. This fact highlights the influence of the selected error measures. The ANN models continue to present the best performances.

Analyzing the individual models, one can confirm the tendency previously observed. The best models found with this dataset show slightly better results than those found using WD2. By comparing Table 10 with Table 8, one can also observe that the best models seem independent of the dataset used. Namely, using the Euclidean weight function in KNN models with eight neighbors, the rectifier or identity activation functions combined with LBFGS learning algorithm in ANN models and eight estimators with eight samples in each split in RFR models consistently presents good forecasts. For this case, the ARIMA presents slightly better RMSE than machine learning methods. However, the best ANN presents a much better MAPE% and R^2 with little prejudice of the RMSE. Figure 7 shows the forecasts made by ANN(relu(8 × 25) lbfgs) and the ARIMA models.

Table 10 | Models that achieved the (i) best RMSE per family, (ii) best three RMSE overall and (iii) worst RMSE overall, and (iv) ARIMA results, applied to the WD4 database of Water Utility 1, using 14 × 24 h demand features

Model	RMSE (–)	MAPE (%)	R^2
RFR($N = 5$ $n = 2$)	0.0623	9.6819	0.8799
KNN($N = 8$ Euclidean)	0.0681	10.4101	0.8548
SVR(rbf t = 0.001)	0.0574	9.2291	0.8814
ANN(relu(8 × 25) lbfgs)	0.0473	7.8511	0.9229
ANN(relu(8 × 25) lbfgs)	0.0473	7.8511	0.9229
ANN(relu(5 × 10) lbfgs)	0.0474	8.0949	0.9293
ANN(relu(5 × 25) lbfgs)	0.0483	8.4529	0.9101
ANN(relu(5 × 10) sgd)	0.2359	40.1780	−79.0492
ARIMA	0.0417	8.5338	0.8659

Results for Water Utility 2

A similar analysis can be made for the second dataset. Generically, the forecasting errors RMSE and MAPE% found for this dataset are better than those found for Water Utility 1, while the R^2 drops, as seen in Table 11 in comparison to Tables 7 and 9. The use of 14 samples of 24 h presents the best RMSE and MAPE% results. The use of clusters in the forecasts does not bring better forecasts, whichever the dataset, but occasionally results in a better correlation between the forecasts and the observations. The use of similar days to train the models results in a more correctly identified pattern, but also results in fewer examples available for training, possibly resulting in fewer iterations and incomplete training.

The best models of each family of methods are presented in Table 12. Surprisingly, the SVR methods did not present identical results to those obtained previously. However, note that the results obtained by the different methods have a smaller range than those observed using the previous datasets. The best model and the ARIMA's forecasts are represented in Figure 8.

Table 11 | Forecasting errors of the best model found with each approach using the Water Utility 2 database

Periodicity (h)	Features	Model	RMSE (−)	MAPE (%)	R^2
24	Demand (3)	KNN($N = 8$ uniform)	0.0315	9.3134	0.7067
24	Demand (14)	ANN(identity(2×10) lbfgs)	0.0228	7.0477	0.7532
24	Demand (70)	ANN(identity(5×75) lbfgs)	0.0268	8.6002	0.6626
12	Demand (14)	ANN(identity(2×25) lbfgs)	0.0242	7.8026	0.6314
48	Demand (14)	ANN(relu(2×10) lbfgs)	0.0276	8.2274	0.7314
168	Demand (14)	ANN(relu(2×10) lbfgs)	0.0340	10.117	0.6920
24	Demand, using clustering (14)	ANN(identity(8×10) lbfgs)	0.0266	7.9235	0.7420

Table 12 | Models that achieved the (i) best RMSE per family, (ii) best three RMSE overall and (iii) worst RMSE overall, and (iv) ARIMA results, applied to the Water Utility 2 database, using 14 × 24 h demand features

Model	RMSE (−)	MAPE (%)	R^2
RFR($N = 8$ $n = 8$)	0.0264	8.1820	0.6849
KNN($N = 8$ Euclidean)	0.0273	8.2914	0.6450
SVR(linear t = 0.001)	0.0578	23.9645	−0.5905
ANN(identity(2×10) lbfgs)	0.0228	7.0477	0.7532
ANN(identity(2×10) lbfgs)	0.0228	7.0477	0.7532
ANN(identity(5×25) lbfgs)	0.0239	7.3169	0.7338
ANN(identity(5×10) lbfgs)	0.0242	7.4216	0.7385
ANN(logistic(5×75) adam)	0.0814	32.2616	−1.1143
ARIMA	0.0452	8.6048	0.5784

Results for the weighted parallel forecasting (WPF) strategy

Considering the results previously discussed and those found in the literature, one can assess which model configurations and forecasting techniques might present the best results. Instead of designing a model presumed to accomplish good results across different databases, it is possible to conceive a pool of models and approaches, the combination of which outperforms each individual model. The analysis made so far shows that the best models should respect the following criteria:

- 24 h forecast window;
- Use ~2 weeks of previous water demand observations as input;
- When configuring ANN:
  - ○ LBFGS learning algorithm;
  - ○ Rectifier or identity activation function;
  - ○ Small networks;
- When configuring KNN:
  - ○ Euclidean weight function;
  - ○ Eight neighbors;
- When configuring RFR:
  - ○ Eight trees per forest;
  - ○ Eight or more samples at each split.

The new pool of models being tested is composed of four RFR, three KNN, three SVR and 12 ANN. The RFR models have five or eight trees per forest and two or eight minimum samples per split. The KNN models use the Euclidean distance weight function for seven, eight or nine neighbors (refining the previous numbers of neighbors tested). The SVR uses the three kernels tested so far, with the tolerance of 0.01. The ANN uses the LBFGS learning algorithm with identity or rectifier activation functions, with the 10 or 25 neurons distributed by two, five or eight layers.

For the WPF strategy evaluation, the WD2 and WD4 datasets correspond to dates between 21/09/2012 @ 00:00 and 31/07/2013 @ 23:00 in an hourly frequency (total: 7,536 observations). From those, 95.541% were used in training (7,200 observations, 300 days), 2.229% were used in validating (168 observations, 7 days) and another 2.229% were used in testing (168 observations, 7 days).

The Water Utility 2 dataset (consumption in the Ançã region) corresponds to dates between 01/01/2015 @ 00:00 and 31/12/2015 @ 23:00 in an hourly frequency (total: 8,760 observations). From those, 96.164% were used in training (8,424 observations, 351 days), 1.912% were used in validating (168 observations, 7 days) and another 1.912% were used in testing (168 observations, 7 days).

Table 13 lists the results obtained by the models whose output is used as a part of the weighted average forecast in

Table 13 | Results obtained using the weighted parallel forecasting (WPF) methodology when applied to Water Utilities 1 (WD2 and WD4) and 2 (Ancã)

			Validation set					**Test set**			
Water utility	**Methodology**	**Weight**	**ME (−)**	**RMSE (−)**	**MAPE (%)**	**R^2**	***Verification Wi * ME***	**ME (−)**	**RMSE (−)**	**MAPE (%)**	**R^2**
1 (WD4)	SVR(poly tol = 0.01)	0.2614	−0.0031	0.0566	8.1080	0.9189	*−0.0008*	0.0206	0.0936	20.8734	0.6241
	SVR(poly tol = 0.01)	0.2614	−0.0031	0.0566	8.1080	0.9189	*−0.0008*	0.0206	0.0936	20.8734	0.6241
	SVR(poly tol = 0.01)	0.2614	−0.0031	0.0566	8.1080	0.9189	*−0.0008*	0.0206	0.0936	20.8734	0.6241
	ANN(identity(10, 10, 10, 10, 10) lbfgs)	0.0589	0.0127	0.0516	8.2754	0.9379	*0.0007*	−0.0113	0.0627	8.8844	0.8757
	ANN(identity(10, 10, 10, 10, 10) lbfgs)	0.0589	0.0127	0.0516	8.2754	0.9379	*0.0007*	−0.0113	0.0627	8.8844	0.8757
	ANN(identity(25, 25, 25, 25, 25) lbfgs)	0.0980	0.0099	0.0524	7.5083	0.9364	*0.0010*	0.004	0.0607	8.7404	0.8863
	WPF	1.0000	0.0001	0.0534	7.7827	0.9279	**0.0000**	0.0153	0.0817	17.6447	0.7234
1 (WD2)	ANN(relu(25, 25, 25, 25, 25) lbfgs)	0.0884	−0.0051	0.0649	10.9174	0.8864	*−0.0005*	0.011	0.0727	9.6134	0.8537
	ANN(relu(25, 25, 25, 25, 25) lbfgs)	0.0884	−0.0051	0.0649	10.9174	0.8864	*−0.0005*	0.011	0.0727	9.6134	0.8537
	ANN(relu(25, 25, 25, 25, 25) lbfgs)	0.0884	−0.0051	0.0649	10.9174	0.8864	*−0.0005*	0.011	0.0727	9.6134	0.8537
	ANN(identity(25, 25) lbfgs)	0.2449	0.0019	0.0674	10.7613	0.8793	*0.0005*	−0.0099	0.0781	11.852	0.7586
	ANN(identity(25, 25) lbfgs)	0.2449	0.0019	0.0674	10.7613	0.8793	*0.0005*	−0.0099	0.0781	11.852	0.7586
	ANN(identity(25, 25) lbfgs)	0.2449	0.0019	0.0674	10.7613	0.8793	*0.0005*	−0.0099	0.0781	11.852	0.7586
	WPF	1.0000	−0.0001	0.0657	10.5017	0.8839	**0.0000**	− **0.0044**	0.0741	10.5901	0.8024
2 (ANCÃ)	KNN($N = 7$ weight = distance)	0.0577	−0.0060	0.0289	4.9720	0.9143	*−0.0003*	−0.0131	0.0649	10.1623	0.5099
	KNN($N = 7$ weight = distance)	0.0577	−0.0060	0.0289	4.9720	0.9143	*−0.0003*	−0.0131	0.0649	10.1623	0.5099
	KNN($N = 7$ weight = distance)	0.0577	−0.0060	0.0289	4.9720	0.9143	*−0.0003*	−0.0131	0.0649	10.1623	0.5099
	ANN(identity(10, 10) lbfgs)	0.1850	0.0016	0.0301	5.3043	0.9093	*0.0003*	−0.0026	0.0526	8.4021	0.556
	ANN(identity(10, 10) lbfgs)	0.1850	0.0016	0.0301	5.3043	0.9093	*0.0003*	−0.0026	0.0526	8.4021	0.556
	RFR($N = 5$ $n = 2$)	0.4569	0.0010	0.0319	5.1740	0.9086	*0.0005*	−0.004	0.0504	8.1469	0.553
	WPF	1.0000	−0.0003	0.0274	4.6724	0.9260	**0.0000**	−0.0051	0.0523	8.2916	0.5777

The results obtained by the models used for the weighted average are also presented.

the two datasets and their corresponding weights. The weights are calculated during the validation of the models. The forecasting errors listed in Table 13 were found in the validation and testing phase. It should be noticed that the testing set of Water Utility 2 is a difficult set to forecast because it includes the Christmas season.

For the validation set, the proposed parallel methodology results in a significant improvement in ME for all cases. However, this strategy presents a slight decrease in MAPE% and R^2 for the WD2. This fact is expected taking into account that ME was the metric used for the calculation of the weights. The parallel strategy also improved the forecasting results in the testing set for WD2 considering the ME metric. However, for the other examples, the same was not observed.

Although the WPF methodology is interesting and can improve the forecast of a water time series, it has some limitations and drawbacks. For the case of WD4, only the SVR(poly tol = 0.01) model, which obtained poor forecasts for the testing set (MAPE = ~20%), presented ME < 0. Therefore, this model was automatically used, and the results obtained with the WPF were affected by this poor forecast. Generically, it is safe to use the presented parallel methodology, assuming the models used in the parallel computations satisfy a set of pre-requisites relative to their expected performance.

Figure 9 represents a 24 h demand forecast made by the WPF, ANN(relu(2 × 10) lbfgs) and RFR($N = 5$ $n = 2$) models for 2 days of the WD2 in Water Utility 1. Here, it can be observed that the WPF forecast is always a weighted interpolation of the available techniques, being a balanced solution.

Figure 10 represents the demand of the last days of the testing set of the WD4 in Water Utility 1, and the respective

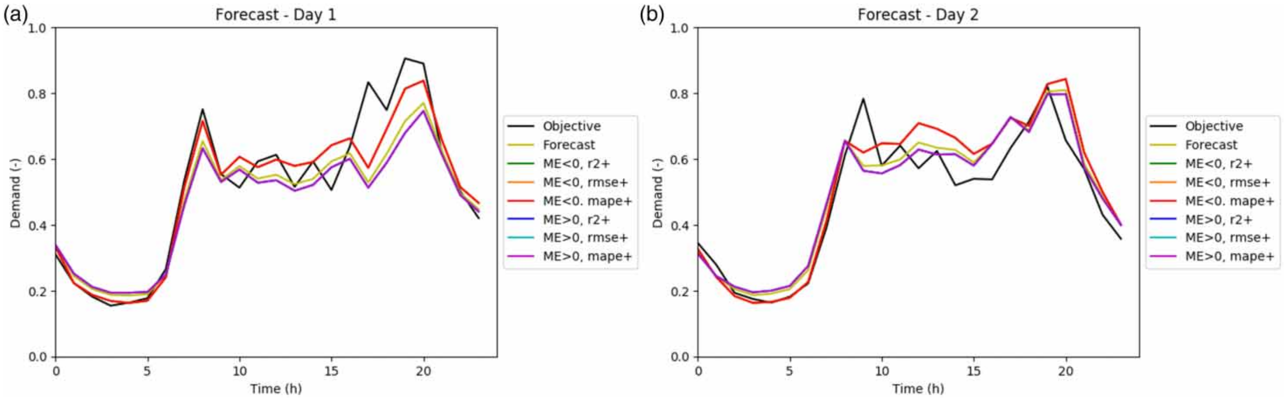

Figure 9 | WD2 database of Water Utility 1. Water consumption observed in the last 2 days. Forecasts given by the WPF, ANN(relu(2 × 10) lbfgs) and RFR($N = 5$ $n = 2$) models.

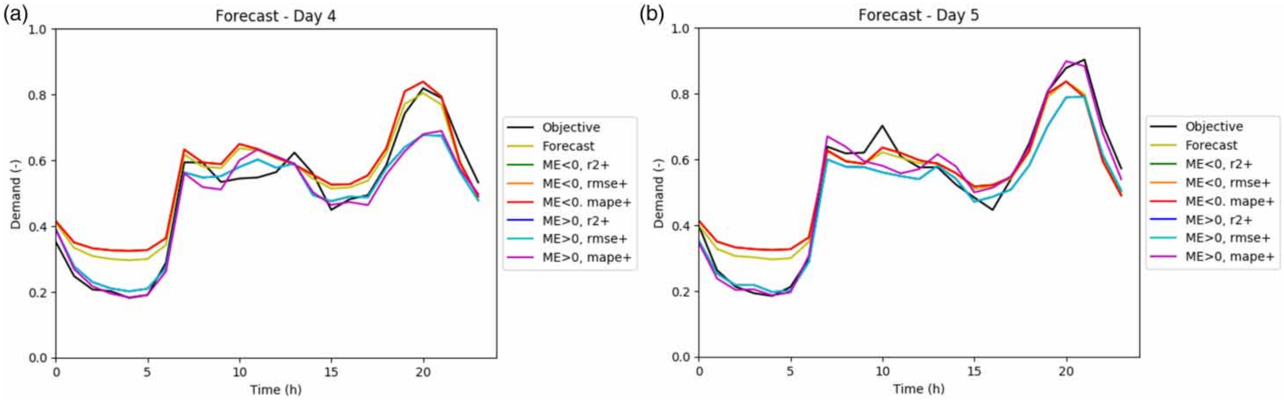

Figure 10 | WD4 database of Water Utility 1. Water consumption observed in the last 2 days. Forecasts given by the WPF, ANN(relu(5 × 25) lbfgs) and ANN(relu(2 × 25) lbfgs) models.

forecasts using the WPF, ANN(relu(5×25) lbfgs) and ANN(relu(2×25) lbfgs) models. The overestimated forecasted values of the $ME < 0$ selected model (the SVR) are responsible for the bad forecast of the WPF method in the 0–5 h period of day 4. However, they are also responsible for the good forecast for the 17–23 h period, where the $ME > 0$ models behave worse. Therefore, the WPF method (indicated as 'Forecast' in Figure 10) is a compromise between the best models, being a balance between the $ME < 0$ and $ME > 0$ best models.

CONCLUSIONS

Developing and applying forecasting algorithms may result in a cost reduction of 18% or more (Cembrano *et al.* 2000; Salomons *et al.* 2007; Kang *et al.* 2014). This work presents machine learning water demand forecasting models capable of producing accurate predictions when compared with traditional strategies. It was found to be reliable when applied to water demand real data, provided there were no significant anomalies of the data used during training. The error metrics here discussed support the evidence that the forecasts made are similar to the real observation, independently of the time of day.

Nonetheless, some remarks on the use of the presented algorithms arise. Although it was found that the same group of models consistently gives the best results, it is not guaranteed that for new data those models will maintain their performance. When applying the algorithm in different datasets, a large set of models must be trained in order to infer the most appropriate models. If applied to real cases where new data is constantly being acquired, it is important that the models are retrained on a regular basis. Note that in the latter case, the introduction of new data could mean that the accuracy of the models that were previously found to be the most adequate for that specific network is affected. Consequently, the suggested periodic retraining must include the larger set of models. Additionally, the proposed weighted parallel forecasting strategy proved its usefulness and can be a good compromise when using several models.

It should also be noted that the evaluation of the forecasting techniques is highly dependent on the metric used. The technique that shows the lowest RMSE is not necessarily the one that presents the lowest MAPE or the R^2 closest to 1. Consequently, the metric used for both training, testing and comparison of techniques should be wisely chosen. In this work, the RMSE metric was selected. However, the results of MAPE and R^2 were also discussed.

According to the tests made, machine learning methods should be chosen over traditional time series analysis. Although the ARIMA often provides results better than those achieved by some machine learning models, most of the time there is at least one machine learning model that outperforms ARIMA (about 18% in RMSE and 8% in MAPE%). Therefore, both strategies should be tested in order to assess their real value in the case being studied.

ACKNOWLEDGEMENTS

The authors gratefully acknowledge the financial support of the Portuguese Foundation for Science and Technology (FCT) under the project UID/EMS/00481/2013-FCT under CENTRO-01-0145-FEDER-022083.

REFERENCES

Adamowski, J. & Karapataki, C. 2010 Comparison of multivariative regression and artificial neural networks for peak urban water-demand forecasting: evaluation of different ANN learning algorithms. *J. Hydrol. Eng.* **15**, 729–743.

Alvisi, S., Franchini, M. & Marinelli, A. 2007 A short-term, pattern-based model for water-demand forecasting. *J. Hydroinform.* **9** (1), 39–50.

Bakker, M., Vreeburg, J. H., van Schagen, K. M. & Rietveld, L. C. 2013 A fully adaptive forecasting model for short-term drinking water demand. *Environ. Model. Softw.* **48**, 141–151.

Bishop, C. M. 2006 *Pattern Recognition and Machine Learning.* Springer, New York.

Brentan, B. M., Luvizotto Jr., E., Herrera, M., Izquierdo, J. & Pérez-García, R. 2017 Hybrid regression model for near real-time urban water demand forecasting. *J. Comput. Appl. Math.* **309**, 532–541.

Bunn, S. M. & Reynolds, L. 2009 The energy-efficient benefits of pump-scheduling optimization for potable water supplies. *IBM J. Res. Dev.* **53** (3), 5:1–5:13.

Byrd, R. H., Lu, P., Nocedal, J. & Zhu, C. 1995 A limited memory algorithm for bound constrained optimization. *SIAM J. Sci. Comput.* **16**, 1190–1208.

Candelieri, A. 2017 Clustering and support vector regression for water demand forecasting and anomaly detection. *Water* **9** (3), 224.

Candelieri, A., Conti, D., Cappellini, D. & Archetti, F. 2014a Urban Water Demand Characterization and Short-Term Forecasting – The ICeWater Project Approach. In: *International Conference on Hydroinformatics*, New York, USA.

Candelieri, A., Soldi, D., Conti, D. & Archetti, F. 2014b Analytical leakages localization in water distribution networks through clustering and support vector machines. The Icewater Approach. *Proc. Eng.* **89**, 1080–1088.

Cembrano, G., Wells, G., Quevedo, J., Pérez, R. & Argelaguet, R. 2000 Optimal control of a water distribution network in a supervisory control system. *Control Eng. Pract.* **8** (10), 1177–1188.

Coelho, B. 2016 *Energy Efficiency of Water Supply Systems Using Optimization Techniques and Micro-Hydroturbines*. PhD thesis, University of Aveiro, Aveiro, Portugal.

de Lima, J. D., Adamczuk, G. O., Trentin, M. G., Batistus, D. R. & Pozza, C. B. 2016 A study of the performance of individual techniques and their combinations to forecast urban water demand. *Rev. Espac.* **37** (22), 5–28.

Ghiassi, M., Fa'al, F. & Abrishamchi, A. 2016 Large metropolitan water demand forecasting using DAN2, FTDNN, and KNN models: a case study of the city of Tehran, Iran. *Urban Water J.* **14** (6), 655–659.

Haque, M. M., de Souza, A. & Rahman, A. 2017 Water demand modelling using independent component regression technique. *Water Resour. Manage.* **31** (1), 299–312.

Herrera, M., Torgo, L., Izquierdo, J. & Pérez-Garcia, R. 2010 Predictive models for forecasting hourly urban water demand. *J. Hydrol.* **387**, 141–150.

Hyndman, R. J. 2010 *Hyndsight Blog*. Available from: https://robjhyndman.com/hyndsight/benchmarks/ (accessed 15 June 2017).

Kang, H.-S., Kim, H., Lee, J., Lee, I., Kwak, B.-Y. & Im, H. 2014 Optimization of pumping schedule based on water demand forecasting using combined model of autoregressive integrated moving average and exponential smoothing. *Water Sci. Technol. Water Supply* **15** (1), 188–195.

Kingma, D. P. & Ba, J. 2015 Adam: A Method for Stochastic Optimization. In: *3rd International Conference for Learning Representations*, San Diego.

Liu, D. C. & Nocedal, J. 1989 On the limited memory BFGS method for large scale optimization. *Math. Program.* **45** (1), 503–528.

Mala-Jetmarova, H., Sultanova, N. & Savic, D. 2017 Lost in optimisation of water distribution systems? A literature review of system operation. *Environ. Model. Softw.* **93**, 209–254.

Mellios, N., Kofinas, D., Papageorgiou, E. & Laspidou, C. 2015 A Multivariative Analysis of the Daily Water Demand of Skiathos Island, Greece, Implementing the Artificial Neuro-Fuzzy Inference System (ANFIS). In: *E-proceedings of the 36th IAHR World Congress*, The Hague, The Netherlands.

Mitchell, T. M. 1997 *Machine Learning*. McGraw-Hill, Boston, USA.

Moutadid, S. & Adamowski, J. 2017 Using extreme learning machines for short-term urban water demand forecasting. *Urban Water J.* **14** (6), 360–368.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O. & Duchesnay, É. 2011 Scikit-learn: machine learning in Python. *J. Mach. Learn. Res.* **12**, 2825–2830.

Peña-Guzman, C., Melgarejo, J. & Prats, D. 2016 Forecasting water demand in residential, commercial, and industrial zones in Bogotá, Colombia, using least-squares support vector machines. *Math. Probl. Eng.* **2016**, 1–10. http://dx.doi.org/10.1155/2016/5712347.

Python Software Foundation 2017 *The Python Language Reference*. Available from: https://docs.python.org/3/reference/index.html (accessed 14 June 2017).

Qian, Q. & Pan, J. 2017 *A Creative Visualization of OLAP Cuboids*. Available from: www.ebaytechblog.com/2017/05/09/a-creative-visualization-of-olap-cuboids/ (accessed 7 March 2017).

Rodríguez-Galiano, V. & Villarín-Clavería, M. C. 2016 Regression Trees for Modelling Water Demand in Sevilla City, Spain. In: *Proceedings of the Geostatistics and Machine Learning. Applications in Climate and Environmental Sciences Conference*, Belgrade, pp. 18–20.

Salomons, E., Goryashko, A., Shamir, U., Rao, Z. & Alvisi, S. 2007 Optimizing the operation of the Haifa-A water-distribution network. *J. Hydroinform.* **9** (1), 51–64.

Seo, Y., Kim, S., Kisi, O. & Singh, V. P. 2015 Daily water level forecasting using wavelet decomposition and artificial intelligence techniques. *J. Hydrol.* **250**, 224–243.

Shabani, S., Yousefi, P., Adamowski, J. & Naser, G. 2016 Intelligent soft computing models in water demand forecasting. In: *Water Stress in Plants* (I. M. M. Rahman, Z. A. Begum & H. Hasegawa, eds). InTech, Croatia, pp. 99–117.

Shabani, S., Candelieri, A., Archetti, F. & Naser, G. 2018 Gene expression programming coupled with unsupervised learning: a two-stage learning process in multi-scale, short-term water demand forecasts. *Water* **10** (2), 142.

Suh, D. & Ham, S. 2016 A water demand forecasting model using BPNN for residential building. *Contemp. Eng. Sci.* **9** (1), 1–10.

Svozil, D., Kvasnicka, V. & Pospíchal, J. 1997 Introduction to multi-layer feed-forward neural networks. *Chemometr. Intell. Lab. Syst.* **39**, 43–62.

The Pennsylvania State University, PennState Eberly College of Science 2017 *The Coefficient of Determination, R-Squared*. Available from: https://onlinecourses.science.psu.edu/stat501/node/255 (accessed 7 July 2017).

The SciPy Community 2017 *SciPy Reference Guide*. Available from: https://docs.scipy.org/doc/scipy/reference/index.html (accessed 8 July 2017).

Tiwari, M., Adamowski, J. & Adamowski, K. 2016 Water demand forecasting using extreme learning machines. *J. Water Land Dev.* **25** (1–3), 37–52.

Veen, F. V. 2016 *The Neural Network Zoo*. Available from: www.asimovinstitute.org/neural-network-zoo/ (accessed 22 June 2017).

First received 6 December 2017; accepted in revised form 30 July 2018. Available online 21 August 2018