FEDERAL UNIVERSITY OF SANTA CATARINA
JOINVILLE TECHNOLOGY CENTER
MECHATRONICS ENGINEERING B.S.

JESUINO VIEIRA FILHO

COMPARISON OF MACHINE LEARNING METHODS FOR SHORT-TERM URBAN WATER DEMAND FORECASTING IN A COASTAL TOURIST CITY

Joinville
2022

JESUINO VIEIRA FILHO

COMPARISON OF MACHINE LEARNING METHODS FOR SHORT-TERM URBAN WATER DEMAND FORECASTING IN A COASTAL TOURIST CITY

This graduation thesis is presented to fulfill the partial requirement to obtain the title of bachelor in the course of Mechatronics Engineering, at the Joinville Technology Center, from the Federal University of Santa Catarina.

Advisor: Prof. Dr. Pablo Andretta Jaskowiak

Joinville
2022

This research is lovingly dedicated to my grandparent, Natalicio Vieira.
Esta pesquisa é carinhosamente dedicada ao meu avô, Natalicio Vieira.

ACKNOWLEDGEMENTS

Como em qualquer pesquisa acadêmica que resulte na produção de um trabalho de conclusão de curso, na capa não deve haver apenas o nome do pesquisador, mas também os nomes de todos aqueles heróis anônimos, aqueles que, em graus variados, prestaram assistência, incentivo e orientação, e sem os quais eu não teria conseguido. Sou muito grato a todas as pessoas que me deram tanto do seu tempo, amor e energia, a todos que, direta ou indiretamente, influenciaram no meu aprendizado e desenvolvimento como estudante. Ao meu amigo e orientador, Prof. Pablo Andretta Jaskowiak, que teve papel fundamental no meu crescimento pessoal e profissional, e com o qual tive o prazer de trabalhar junto desde o meu segundo semestre na universidade. Você é um grande exemplo para mim. Aos meus pais, Luciana e Jesuino, e ao meu irmão, Luiz Fernando, por todo o apoio, amor e carinho. O incentivo e auxílio de vocês foram fundamentais para eu chegar até aqui, além de que são a minha maior fonte de inspiração e força para seguir em frente. Aos meus irmãos do grupo Bolah, Naruto, Nóia, Vô e Zeca, pelas inúmeras experiências colecionadas e as quais levarei para a vida toda, assim como vocês. Aos meus irmãos do grupo LPs, onde minhas raízes residem, pelos esporádicos, porém inesquecíveis momentos de amizade que pude ter quando retornei para casa. A minha namorada Leticia, pelos momentos de companheirismo e compreensão, principalmente nos últimos meses. Você tornou tudo mais leve. Aos meus amigos da Caverna, do LISHA, e de Aachen, agradeço de coração pela parceria e por se fazerem presentes nas diferentes fases que vivenciei durante a graduação. Por fim, agradeço a Universidade Federal de Santa Catarina por prover estudo gratuito e de qualidade, tenho imenso orgulho de carregar esse nome comigo. Também agradeço a SANEPAR e a SIMEPAR, por fornecerem os dados para este trabalho e assim, incentivar a pesquisa e educação.

Nobody ever figures out what life is all about, and it doesn't matter. Explore the world. Nearly everything is really interesting if you go into it deeply enough.

Richard P. Feynman

ABSTRACT

Proper planning and management of water resources is fundamental to human well-being and contemporary socio-economic development. However, the use of increasing amounts of water has brought a series of problems that compromise its quality and durability. Given the scarcity of this natural resource and its inherent constraints, accurate forecasting of water consumption is imperative for the optimal operation of water collection, treatment, and distribution systems. Thus, the present work aims to study and compare different machine learning methods for predicting daily urban water demand in a Brazilian coastal tourist city. To this extent, four learning algorithms were employed: linear regression (LR), k-nearest neighbors (kNN), support vector regression (SVR) and multilayer perceptron (MLP). Moreover, three configurations of a time series cross-validation approach known as backtesting were considered for each method, two with a sliding window and one with an expanding window, totaling 12 models. They were all subjected to hyperparameter optimization (model selection), and then evaluated using appropriate performance metrics (model evaluation). To train the models, historical data from the city's water distribution system (WDS) was collected, along with additional meteorological and calendar data. These raw data were submitted to exploratory analysis and preprocessing. The empirical results underscore the importance of using nonlinear models to predict short-term water demand. Based on the adopted performance metrics, MLP performed the best, while LR was the worst. SVR and kNN were second and third, respectively. With reference to the three backtesting configurations employed, each learning algorithm had its best model using the expanding window.

Keywords: Water demand. Time series. Machine learning. Backtesting.

RESUMO

O planeamento e gestão adequados dos recursos hídricos são fundamentais para o bem-estar humano e o desenvolvimento socioeconômico contemporâneo. No entanto, a utilização de quantidades crescentes de água trouxe uma série de problemas que comprometem a sua qualidade e durabilidade. Dada a escassez deste recurso natural e os seus constrangimentos inerentes, a previsão precisa do consumo de água é imperativa para o ótimo funcionamento dos sistemas de captação, tratamento e distribuição de água. Assim, o presente trabalho tem como objetivo estudar e comparar diferentes métodos de aprendizado de máquina para prever a demanda diária de água urbana em uma cidade turística litorânea brasileira. Nesse contexto, foram utilizados quatro algoritmos de aprendizado: regressão linear (LR), k-vizinhos mais próximos (kNN), regressão por vetores suporte (SVR) e perceptron multicamadas (MLP). Além disso, três configurações de uma abordagem de validação cruzada para séries temporais conhecida como backtesting foram consideradas para cada método, duas com uma janela deslizante e uma com uma janela de expansão, totalizando 12 modelos. Todos eles foram submetidos a uma otimização de hiperparâmetros (seleção de modelos), e então avaliados por meio de métricas de desempenho apropriadas (avaliação de modelos). Para treinar os modelos, foram coletados dados históricos do sistema de distribuição de água da cidade, além de dados meteorológicos e de calendário adicionais. Estes dados bruto foram submetidos a análise exploratória e pré-processamento. Os resultados empíricos ressaltam a importância da utilização de modelos não lineares para prever a demanda de água no curto prazo. Com base nas métricas de desempenho adotadas, a MLP teve o melhor desempenho, enquanto que o LR foi o pior. SVR e kNN foram o segundo e o terceiro, respectivamente. Com referência às três configurações de backtesting utilizadas, cada algoritmo de aprendizagem teve o seu melhor modelo utilizando a janela de expansão.

Palavras-chave: Demanda de água. Série temporal. Aprendizado de máquina. Backtesting.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

ANFIS Adaptive Neuro-Fuzzy Inference System

ANN Artificial Neural Network

ARIMA Autoregressive Integrated Moving Average

ARMA Autoregressive Moving Average

EDA Exploratory Data Analysis

ELM Extreme Learning Machine

GPR Gaussian Process Regression

kNN k-Nearest Neighbors

LSSVM Least Square Support Vector Machine

LR Linear Regression

ML Machine Learning

MAE Mean Absolute Error

MAPE Mean Absolute Percentage Error

MLP Multilayer Perceptron

RBF Radial Basis Function

RSS Residual Sum of Squares

RMSE Root Mean Square Error

R^2 R-squared

SSA Singular Spectrum Analysis

SVM Support Vector Machine

SVR Support Vector Regression

WDS Water Distribution System

WSS Water Supply System

WTP Water Treatment Plant

TABLE OF CONTENTS

1 INTRODUCTION

Proper management of water resources is fundamental to human well-being and contemporary socio-economic development and, given its importance, numerous civilizations have developed alongside river banks. The use of increasing amounts of water has brought a series of problems that compromise the quality and durability of available water resources. Much of this increase in water demand results from a combination of population growth, economic development and changes in consumption patterns (UN-WATER, 2021), which emphasizes the importance of proper planning and management of water supply systems (WSSs).

An important component in implementing and optimizing effective management programs for WSSs is accurate forecasting of water demand, which plays an important role in the optimal operation of water collection, treatment and distribution systems. Under these circumstances, collecting water consumption data can help monitor the system and also obtain demand forecasts, supporting system operation decisions (SINGAPORE, 2016). The analysis of this data makes it possible to understand the underlying factors that influence water use, optimize the operation of pumps and save energy due to controlled pumping, as well as inform the population about likely peaks in demand and avoid possible periods of water shortages.

However, predicting water demand is a challenging task and the reasons for such complexity arise due to the nature of the available data and the variables that affect water consumption (ARBUES; GARCIA-VALINAS; MARTINEZ-ESPINEIRA, 2003). Among these variables are past water demand (which provides evidence of consumption patterns), current operating conditions, and socioeconomic and meteorological factors (such as relative humidity, air temperature, and precipitation) (DONKOR; ROBERSON, 2014). In order to solve this problem, methods for forecasting future water demand range from a simple estimate of per capita water consumption to machine learning models that take these factors into account.

The present work aims to study and compare water demand forecasting techniques for a coastal tourist city, based on machine learning (ML) methods. The four learning algorithms considered are linear regression (LR), k-nearest neighbors (kNN), support vector regression (SVR) and multilayer perceptron (MLP). In addition, predictions are performed using three backtesting configurations, one with an expanding window and two with a sliding window, totaling 12 final models. Real data from the water distribution system (water consumption data) and from a climate monitoring station (meteorological data) are considered, as well as data referring to holidays and school recess and breaks. Exploratory analysis and preprocessing steps are performed to gain insights and improve data quality before building models.

Due to the large population variation caused by tourism, the application scenario is interesting and challenging. The results emphasize the importance of applying methods capable of modeling the nonlinear relationship between the variables that drive water demand. Overall,

MLP presents the best result considering the performance metrics used, while LR is the worst. Regarding the backtesting settings, all methods performed better using the expanding window.

1.1 GENERAL OBJECTIVE

The main objective of this work is to study and quantitatively compare the performance of different learning algorithms for the task of predicting the daily water demand of a Brazilian coastal city located in the state of Paraná, namely, Guaratuba.

1.2 SPECIFIC OBJECTIVES

The following specific objectives have been established to fulfill the general one:

- Improve data quality through an exploratory analysis and preprocessing;
- Estimate the daily water demand using different machine learning algorithms, namely: linear regression, k-nearest neighbors, support vector regression and multilayer perceptron;
- Compare different backtesting configurations for the evaluation scenario studied.

1.3 OUTLINE

The present work is structured in five chapters and the remainder is organized as follows:

- **Chapter 2 – Background:** the basic concepts of machine learning are presented. Water demand forecasting is introduced and a brief literature review is provided.
- **Chapter 3 – Methodology:** describes data collection and analysis procedures. The chapter also discusses the methodology used to develop and evaluate forecasting models.
- **Chapter 4 – Results and Discussion:** presents the results obtained during the exploratory analysis of the data, preprocessing and, finally, selection and evaluation of the models.
- **Chapter 5 – Conclusions:** this bachelor's thesis is concluded by highlighting the main findings and discussing opportunities for future work.

2 BACKGROUND

In this chapter, the background knowledge necessary for understanding the contents discussed throughout the work are presented. Section 2.1 introduces the basic concepts of machine learning and the algorithms used in this research. Once the reader is familiar with these concepts, Section 2.2 discusses the application of this theory to water demand forecasting, ending with a review of related works used as a reference for the development of the present work.

2.1 MACHINE LEARNING

Machine learning is the field of study that develops algorithms capable of identifying and extracting patterns from data (KELLEHER; TIERNEY, 2018). Driven by the emergence of big data, the speedup in computing power, the massive reduction in the cost of computer memory, and the development of more powerful methods for data analysis and modelling, machine learning is one of today's most rapidly growing technical fields (JORDAN; MITCHELL, 2015). This section provides a concise introduction to the general principles of machine learning that are applied throughout the remainder of this work. A number of examples and definitions are discussed just enough to give the fundamentals without going into the particulars. Those who want a wider perspective are encouraged to consider textbooks with a more comprehensive coverage, such as Mitchell (1997), Bishop & Nasrabadi (2006) and Flach (2012).

2.1.1 Data

The success of a machine learning application is fundamentally dependent on the data. In its most basic form, a datum (singular of data) is an abstraction of a real-world person, object or event (KELLEHER; TIERNEY, 2018). The terms variable, feature, and attribute are often used interchangeably to denote this individual abstraction. A collection of one or more of these attributes establishes an entity (also called a data object, instance, observation), which aims to capture the characteristics of what is being observed (TAN; STEINBACH; KUMAR, 2016). For instance, a book might have the following attributes: title, author, year, publisher, edition, price, etc. Finally, a dataset consists of a collection of entities. Table 1 illustrates a dataset organized as an $n \cdot m$ matrix, where m is the number of entities (rows) and n is the number of attributes (columns). Four books are listed in the dataset, and each book is described by seven attributes: ID, title, author, year, cover, edition and price. One may note that there are different types of attributes. According to Tan, Steinbach & Kumar (2016), the default types are: numeric (quantitative) and categorical (qualitative), each with two subdivisions discussed below.

Numeric attributes refer to any information that can be quantified, counted or measured, and given a numerical value (integer or real). They are subdivided into two groups: interval

and ratio. Interval attributes are measured on a scale with an arbitrary origin, for example, calendar dates. It is appropriate to rank, count, subtract or add interval attributes. However, these measurements don't provide any sense of ratio and arithmetic operations such as multiplication and division are not suitable. On the other hand, ratio variables have a defined origin, which is the starting point or zero of the scale. This implies that multiplication and division operations are appropriate. Temperature is the standard example for distinguishing between interval and ratio attributes. If measured on the Celsius or Fahrenheit scale, it is an interval measurement because a value of zero does not imply zero heat. In contrast, temperature measurement in Kelvins is on a ratio scale: physically speaking, a temperature of $2K$ is twice as high as a temperature of $1K$. This is because $0K$ (absolute zero) is the temperature at which all thermal motion ceases. In Table 1, "year" and "price" are examples of range and ratio, respectively.

Table 1 – An example dataset that represents a collection of classic books.

ID	Title	Author	Year	Cover	Edition	Price
1	Emma	Austen	1815	Paperback	20th	$5.75
2	Dracula	Stoker	1897	Hardback	15th	$12.00
3	Ivanhoe	Scott	1820	Hardback	8th	$25.00
4	Kidnapped	Stevenson	1886	Paperback	11th	$5.00

Source: Kelleher & Tierney (2018)

Categorical data is descriptive in nature, taking values from a finite set, and is expressed in terms of language rather than numerical values. It is subdivided into two groups: nominal and ordinal. Nominal attributes are used for naming or labelling variables and ordering or ranking operations can not be applied to them. In Table 1, "author" and "title" are examples of nominal attributes and although they can be sorted alphabetically, this is a distinct operation from sorting. On the other hand, there is a natural ordering over ordinal data. In Table 1, the "edition" attribute is an example of an ordinal attribute. However, it's noteworthy that this is not a quantitative measure: although the "20th" edition comes after "10th", there is no notion of equal distance between these values and "20th" edition is not twice the "10th". In summary, the numbers are not mathematically measured but are merely assigned as labels for opinions.

Although it is made clear that there are other types, Tan, Steinbach & Kumar (2016) discuss three groups: record data, graph-based data, and ordered data. Here, an emphasis is placed on ordered data. In this case, the attributes have relationships that involve order in time or space, in which the former represents the dataset category used in this work. With respect to this special type of ordered data, each record is a series of measurements taken over time, denoted as time series. It is very important to consider the temporal autocorrelation when working with time series, as this factor can invalidate some operations, such as shuffling the dataset. Figure 1 shows an example of time series, where the observations (y-axis) are plotted against the time (x-axis). This is a slice (only the observations for 2017) of the time series used in this work and represents the amount of water produced in a water treatment plant, measured in m^3.

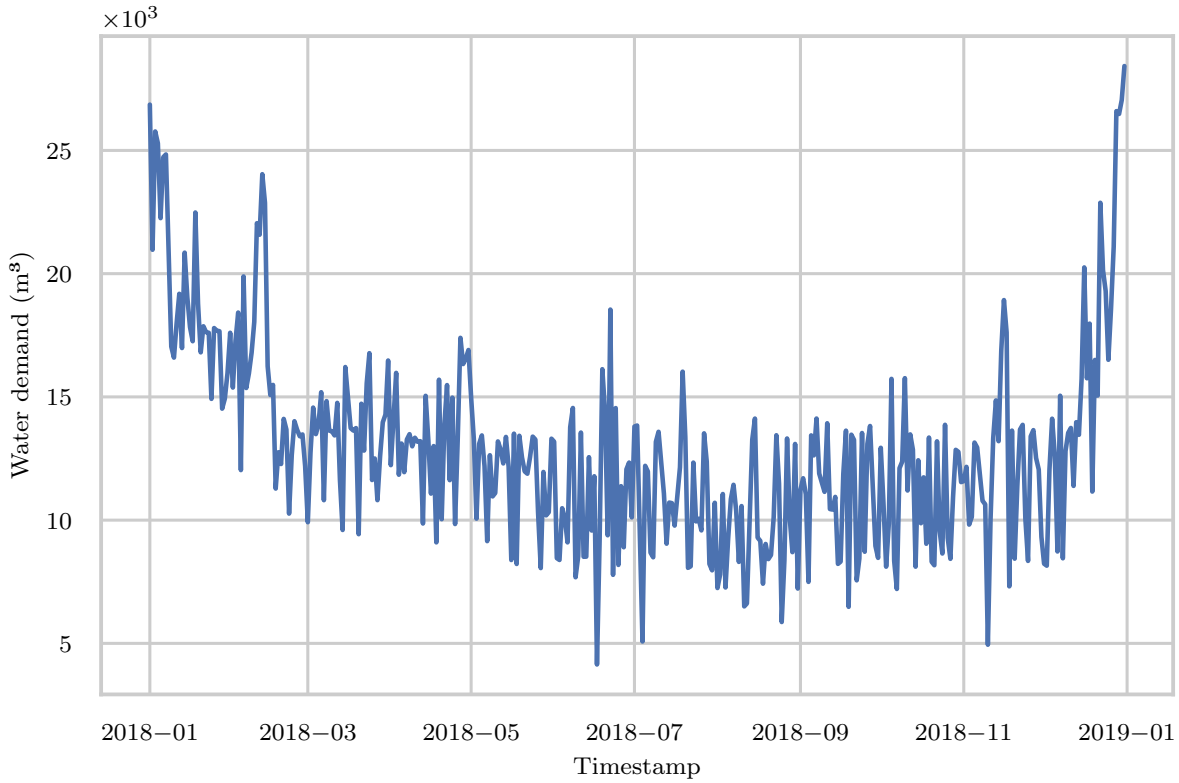


Figure 1 – An example of a time series. This is the daily volume of water produced at the water treatment plants (WTPs) of Guaratuba in 2018.

Thus far, this chapter described the importance of the data, along with some common terms used in the field. As Flach (2012) stated, features are the workhorses of machine learning and determine much of the success of the application. In fact, creating, cleaning, and updating the dataset is the most laborious and time-consuming step of a project (KELLEHER; TIERNEY, 2018). In the real world, data comes from different sources, values or even entire data objects can be missing, and there can be inconsistencies and various data-related issues that are important for a successful analysis. The tools used to address these problems in this work are presented in Section 3.2.1 and Section 3.2.2. The section that follows introduces basic machine learning concepts, starting with how learning algorithms benefit from data.

2.1.2 Basic Concepts

An algorithm is a finite sequence of rigorous and unambiguous instructions executed by computers. Classical algorithms are step-by-step instructions, which implies that with a specific input, one can trace and exactly determine the output (except random algorithms). In contrast, a machine learning algorithm receives an input and possibly a desired output, automatically formulating the rules. The math and logic that support a learning algorithm can update themselves over time (without human intervention) as programming becomes exposed to more data. But what does it mean to learn? According to Mitchell (1997, p. 2), it can be broadly defined as:

> *"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E."*

For instance, a computer program that learns to recognize handwritten words may improve its performance as measured by its ability to correctly classify handwritten words within images, obtained by classifying handwritten words from a dataset with given classifications. One can imagine a wide variety of tasks, experiences, and performance measures. Once the learning problem is well defined, the identification of these characteristics must be clear. In the sections that follow, this formalism will be used as a template to briefly present complex problems, providing examples of how machine learning works and what it can accomplish.

2.1.2.1 Task

All machine learning algorithms use data as input, but what one may want to achieve with them can be different. The abstract representation of the goal to be achieved is the task. Learning algorithms are able to handle tasks that are very difficult to solve with classical ones written by human beings (GOODFELLOW; BENGIO; COURVILLE, 2016). According to Kelleher & Tierney (2018), most projects can be categorized based on the task they are designed for, belonging to one of four general classes: clustering (or segmentation), anomaly (or outlier) detection, association-rule mining and prediction (including classification and regression). The most common among them, which is related to this work, is presented below: prediction.

Prediction aims to estimate the value of an attribute for a given object based on the values of other attributes for that instance. The attribute being predicted is called target, while the others are called inputs and represent the independent variables of the model. Prediction allows for highly accurate guesses about the likely outcomes of a question based on historical data, which can be about all sorts of things – estimating house prices, identifying whether an email is spam, and more. This problem can be described as the mathematical problem of approximating a mapping function f from input variables $\mathbf{X}$ to output variables $\mathbf{y}$. The number of values the output can take indicates whether the task is categorized as classification or regression.

Classification is the task of predicting which of k categories the inputs belong to. It has a finite set of outputs, represented by a discrete label. As shown in Equation (1), the learning algorithm used in the classification task produces a function f that maps its n inputs from real domain $\mathbb{R}^n$ to a range of k possible output (GOODFELLOW; BENGIO; COURVILLE, 2016). The output variables are often called labels or classes. Considering the prediction examples mentioned above, identifying if an email is spam is categorized as a classification task. The output is a numeric code that identifies two classes, spam or not spam.

$$f : \mathbb{R}^n \rightarrow \{1, ..., k\} \tag{1}$$

Sometimes it is necessary to abandon the notion of discrete classes and instead predict a real number. Here is where arises regression, the task of predicting a continuous

quantity such as price or salary. In this case, the learning algorithm produces a function f that maps its n inputs from real domain $\mathbb{R}^n$ to a range defined by the real space $\mathbb{R}$, as shown in Equation (2) (GOODFELLOW; BENGIO; COURVILLE, 2016). Considering the prediction examples mentioned above, estimating house prices is categorized as a regression task. The output is a continuous real-value, and not a fixed set. The main task addressed in this work is regression, with the objective of estimating the water demand in a coastal city.

$$f : \mathbb{R}^n \rightarrow \mathbb{R} \tag{2}$$

There is some overlap between the learning algorithms for classification and regression (GOODFELLOW; BENGIO; COURVILLE, 2016). For instance, a classification algorithm may predict a continuous value, but in the form of a probability (likelihood) for a class label. On the other hand, a regression algorithm may predict a discrete value, but in the form of an integer quantity. Next, its discussed how these algorithms gain experience and learn the function that maps the input values of an instance to the output or target value.

2.1.2.2 Experience

There are different ways in which machine learning algorithms can gain experience, each being convenient for certain tasks and with its own advantages and disadvantages. Before introducing this taxonomy, it is convenient to note what kind of data they take as input: a labeled data or unlabeled data. Unlabeled data does not have meaningful labels associated with it. Typically, it consists of samples that one can obtain relatively easily from the world, such as photos, sensor data, etc. On the other hand, labeled data takes a set of unlabeled data and adds a meaningful label that is somehow informative. For example, labels for the types mentioned above may indicate whether the photo contains an object and whether the sensor measured an expected behavior. However, this process requires a lot of human work to label the data. That being said, most learning algorithms are traditionally divided into two broad categories based on how they gain experience ingesting this data: unsupervised learning and supervised learning.

Unsupervised learning comprises algorithms that learn patterns from unlabeled data. These algorithms aim to discover hidden patterns or clusters of data without the need for human intervention through the time and effort of labeling dataset instances with a target attribute. However, not having a target attribute also means that learning becomes more difficult: the algorithm has a more general task of looking for regularities in the data (KELLEHER; TIERNEY, 2018). Clustering (one of the four classes of tasks mentioned earlier) is the most common unsupervised learning task, where the algorithm looks for clusters that increase both the similarity within the cluster and the diversity between the clusters (XU; WUNSCH, 2005)[1]. Its use can help to identify groups of data characterized by physical causes, which are not clearly observed by humans in a raw dataset, or even impossible considering high-dimensional data.

[1] This is actually one of the clustering definitions.

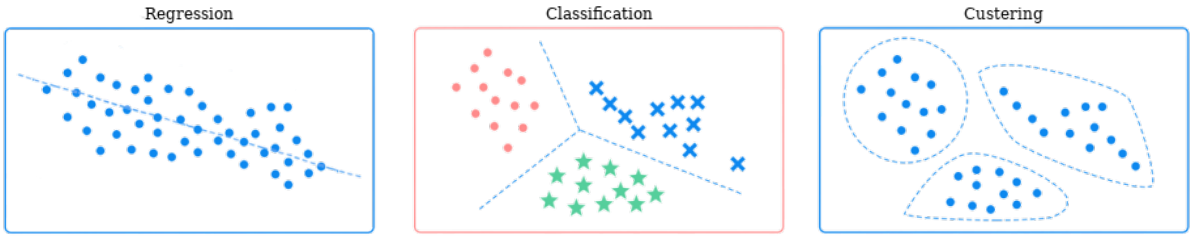


Figure 2 – The main machine learning tasks: regression, classification, and clustering.
Source: adapted from Pradhan (2021)

Supervised learning algorithms work with a labeled dataset. These algorithms obtain a model from input data that has been labeled for a particular output, working as an instructor who shows the system what to do. During the training phase, the algorithm adjusts the model's weights aiming to minimize the error obtained by comparing the labeled output with the predicted one (MURPHY, 2022). So the output of the algorithm is a mathematical approximation of the function that maps the inputs to the output. Classification and regression, the set of prediction tasks discussed in the previous section, are types of problems handled with supervised learning. Accordingly, the present work employs supervised learning algorithms for a regression task. The algorithms considered in this work are presented in Section 2.1.3.

Figure 2 summarizes the three most common machine learning tasks. Regarding how they gain experience, classification and regression use supervised learning methods while clustering uses unsupervised learning methods. Thus, in contrast to the segmentation task, predictive tasks require a labeled training set. To avoid misunderstandings, it should be remembered that in cluster analysis the true number of clusters is usually unknown, while for classification all the classes that one wants to predict are usually known prior to the model construction (there are exceptions, such as the case of open set learning, where there are test samples with classes that are not seen during training).

2.1.2.3 Performance Measure

Thus far, this section has presented a set of common machine learning tasks and described their main types of experience acquisition. The last piece of the puzzle is to have an idea of how well the model is expected to perform on new data. This is done through quantitative measures that are usually particular to the task at hand. Section 3.3.3 presents the ones used to evaluate the models obtained in this work, whereas this section provides an overview of issues and cautions that must be taken when using these metrics to compare models.

Unfortunately, it is very easy to unfairly evaluate ML models and care must be taken to avoid this. In fact, splitting data inappropriately is among the most common pitfalls in machine learning projects (RILEY, 2019). To address this problem, it's necessary to compute the metrics in a test set of data that is separate from the data used for training the model, considering that how well a model performs on the training set is almost meaningless (LONES, 2021). It is also important to ensure that the data in the test set is appropriate, i.e. it must be

representative of the wider population for which the model is aimed at. If these basic premises are neglected, a sufficiently complex model may fully memorize a training set, but fail to capture any generalizable knowledge. Thus, the model becomes useless.

Generalization refers to the model's ability to respond appropriately to new, previously unseen data. This ability to generalize beyond the examples in the training set is the fundamental goal of machine learning and surrounds two central challenges: underfitting and overfitting. (GOODFELLOW; BENGIO; COURVILLE, 2016; KELLEHER; TIERNEY, 2018). Underfitting occurs when the model is not able to obtain a sufficiently low error value in the training set and, consequently, performs poorly on new samples. Overfitting occurs when the training error is much lower than the test error. In this case, the model learns the details of the training data in such a way that it negatively affects performance on new data. Figure 3 illustrates these concepts, the middle plot exemplifies a model with optimal generalization.

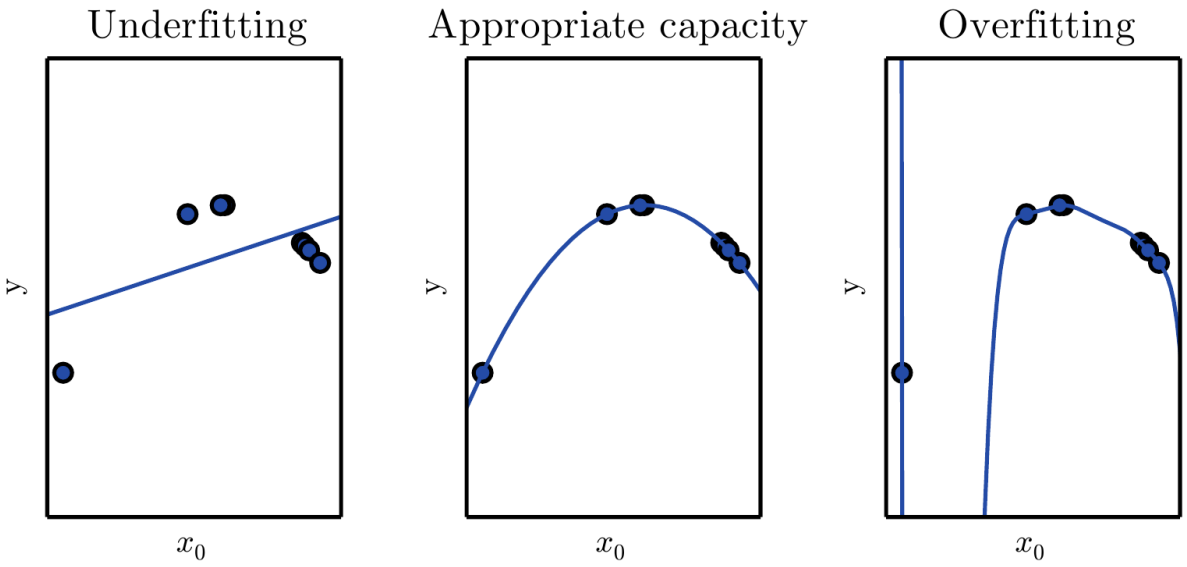


Figure 3 – Under-, optimal- and overfitting in a regression task.
Source: Goodfellow, Bengio & Courville (2016)

In addition, there are different techniques that provide an estimate of a given performance metric, varying according to the task at hand (RASCHKA, 2018). With regard to supervised algorithms, the holdout method is the simplest one: data is split into two sets, one for training and the other for computing the generalization error. Despite not being a rule, usually $2/3$ of data is used to train the model and the $1/3$ remaining for testing. A more robust and also widely used approach is called cross-validation. In this case, different folds (or partitions) of the data are used to train and test a model in different iterations. The performance metric is calculated for each fold and then averaged. As pure cross-validation cannot be used with time series data, a time-based cross-validation procedure known as backtesting is used to evaluate model performance in this work. Section 3.3.1 describes the method in detail.

2.1.3 Supervised Learning Algorithms

Time series forecasting, the objective of this work, can be approached as a regression task and, thus, allows the use of supervised learning methods (BONTEMPI; TAIEB; BORGNE, 2012; BROWNLEE, 2017). Under those circumstances, historical time series observations become the target attribute of the dataset, which is also formed with other input features. Then, the relationship between this set of input variables and the output generates a model that can be used for one-step or multi-step prediction. This section covers the supervised learning algorithms used to address the time series forecasting task of this study, namely: linear regression, k-nearest neighbours, support vector regression and multilayer perceptron.

2.1.3.1 Linear Regression

Linear regression is a widely used method for predicting a real-valued output $y \in \mathbb{R}$ given a vector of n real-valued inputs $\mathbf{x} \in \mathbb{R}^n$ (GOODFELLOW; BENGIO; COURVILLE, 2016; MURPHY, 2022). Its popularity is due to the fact that the model is easy to interpret and fit the data, since the expected value of the output is assumed to be a linear function of the input[2]. In mathematical notation, if $\hat{y}$ is the predicted value, the output is defined as

$$\hat{y} = \mathbf{w}^\intercal \mathbf{x}$$

where $\mathbf{w} \in \mathbb{R}^n$ are known as the weights or regression coefficients learned during training. The method adjusts the coefficients to minimize the cost function between the observed targets in the dataset and the ones predicted by the approximation. The cost function that the linear regression algorithm tries to minimize the residual sum of squares (RSS), given by Equation (3).

$$RSS = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{3}$$

The strategy of fitting a linear function by searching for the regression coefficients that minimize the RSS is known as least squares. For that reason, it is common to refer to a model resulting from this approach as ordinary least squares linear regression. It is worth mentioning that linear regression is an extremely simple and limited learning algorithm that is often used as a baseline for comparing other algorithms (GOODFELLOW; BENGIO; COURVILLE, 2016).

2.1.3.2 k-Nearest Neighbours

The k-nearest neighbors (kNN) is a simple but powerful nonparametric algorithm (also called an instance-based algorithm). Nonparametric approaches make no assumptions for the underlying data distribution, i.e., the model structure is determined from the dataset (MURPHY, 2022). Considering that, this method simply stores the inputs $\mathbf{X}$ and outputs $\mathbf{y}$ of the training set. In an attempt to predict the target value of a new object, the model searches for the nearest

[2] Despite this, it is possible to apply linear regression on transformed data to model nonlinear relationships.

k instances in the stored set and returns the associated regression target. The returned value is the average of the target attribute from these nearest k instances, which can be weighted by the inverse of their distance to the new object, giving more importance to the closest points. The two main parameters in the algorithm are the neighborhood size k and the distance metric.

With reference to k, a small value can lead to overfitting, while a large value can lead to underfitting. Equally important is the notion of distance, a very useful geometric concept in machine learning (FLACH, 2012). If the distance between two instances is small, then they are similar in terms of their attribute values and therefore, the target is expected to receive a similar value. In a Cartesian coordinate system, for example, one can measure the Euclidean distance

$$d(\mathbf{p}, \mathbf{q}) = \sqrt{\sum_{i=1}^{n} (p_i - q_i)^2}$$

which is given by the square root of the sum of the squared distances along each coordinate n. Within this context, the coordinates $\mathbf{p} \in \mathbb{R}^n$ and $\mathbf{q} \in \mathbb{R}^n$ represent the attributes of the object being predicted and those that are part of the training dataset. There are several distances and its choice depends a lot on the dataset. Despite this, attributes might be scaled to a similar range to calculate distance. Considering the Euclidean distance described above, attributes with higher values would have more weight in the distance calculation, affecting the performance of kNN.

2.1.3.3 Support Vector Regression

Initially developed for classification tasks under the name of support vector machine (SVM) and later generalized for regression, support vector regression (SVR) are one of the most elegant prediction methods (CORTES; VAPNIK, 1995; DRUCKER et al., 1996). A brief description of a SVM is helpful to understand its extension for regression. In a classification problem, SVM looks for a hyperplane in the feature space (n-dimensional abstract space where each dataset feature is represented as a point, not including the target variable) in order to correctly classify as many training samples as possible (ZHANG; O'DONNELL, 2020). The hyperplane, also called decision boundary, that best separates the two classes is the one that maximizes the margins between the so-called support vectors. Support vectors are formed by data points closest to the hyperplane and influence its position and orientation. Hence, new samples are classified based on which side of the hyperplane they fall in the n-dimensional feature space.

For regression, instead of finding a hyperplane that widely separates the training samples, the hyperplane is computed to have at most ε deviation from the actual values (SMOLA; SCHÖLKOPF, 2004). SVR doesn't penalize errors as long as they are smaller than ε, giving the flexibility to define how much error is acceptable in the model. Thus, the hyperplane and ε define a region called ε-insensitive tube and the support vectors are those data points that lie on the boundary or outside. All things considered, SVR (used for regression) computes a hyperplane that minimizes the ε-insensitive tube to be as narrow as possible while comprising the most number of training samples, whereas SVM (used for classification) computes a hyperplane that

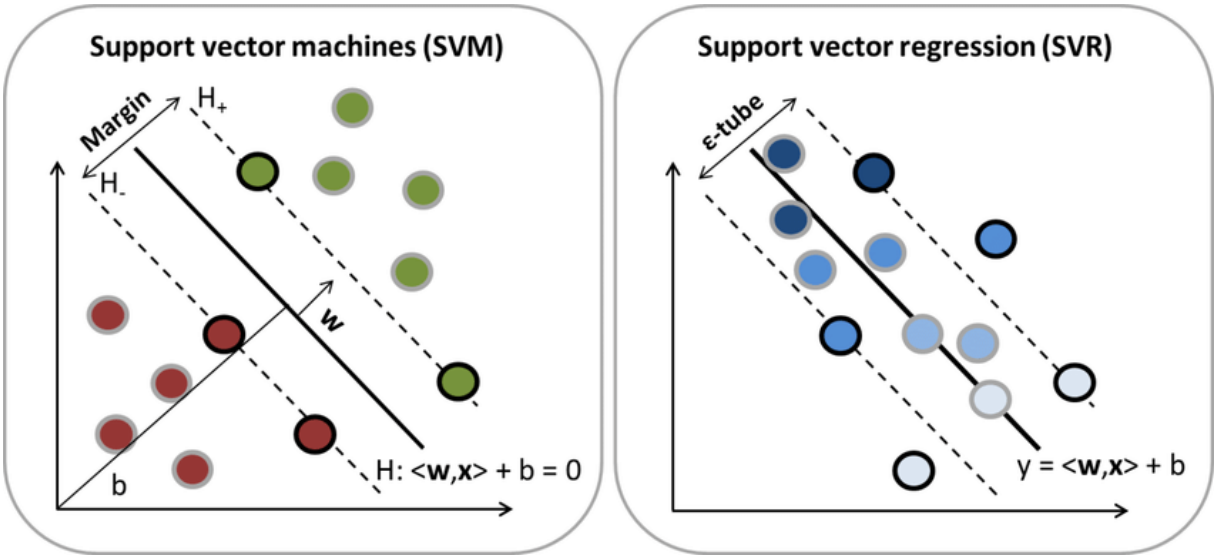


Figure 4 – SVM and SVR modeling. In SVM (left), a hyperplane with maximal margin is constructed to separate two classes (colored green and red, respectively). In SVR (right), the difference between an observed and predicted numerical value is minimized. The gradient from dark to light blue indicates decreasing values. Support vectors for SVM/SVR are indicated by black circles. In SVM, the support vectors are located on the margin, while they may be located outside of the ε-tube in SVR.

Source: Rodríguez-Pérez & Bajorath (2022)

maximizes the margin to be as large as possible while comprising the minimum number of training samples (ZHANG; O'DONNELL, 2020). In both cases, the error term is handled in the constraints of the algorithm. More information on the mathematical formulations that govern the method can be found, for example, in Bishop & Nasrabadi (2006) and Murphy (2022).

Figure 4 illustrates the difference between SVM and SVR. In SVM (left), a hyperplane with maximal margin is constructed to separate two compound classes (colored green and red, respectively). In SVR (right), a hyperplane that helps predict the target value forms an ε-insensitive tube that allows for errors smaller than ε. Support vectors for SVM/SVR are indicated by black circles. This is a simple illustration for didactic purposes, as the relationship between independent and dependent variables is hardly linear. However, it's possible to use kernel functions to transform the original input data into a higher-dimensional space that allows the modeling of nonlinear relationships (BISHOP; NASRABADI, 2006; MURPHY, 2022). This process is known as the "kernel trick" and empowers many linear parametric models. There are a wide variety of kernel functions, such as the radial basis function (RBF).

2.1.3.4 Artificial Neural Network

The artificial neural networks (ANNs), commonly referred to as "neural networks", are a set of algorithms that are inspired by the biological neural networks that constitute the human brain. In terms of information processing, the brain can be seen as a highly complex, nonlinear and parallel computer (HAYKIN, 1998). The computations are done by an extremely interconnected network of neurons, which communicate by sending electric pulses (synapses)

through the neural wiring consisting of axons and dendrites (KROGH, 2008). The pioneering work of McCulloch & Pitts (1943) to create a computational model of a neuron forms the basis of artificial neural network algorithms. The study aimed to understand how the brain could produce highly complex representations using many basic cells that are connected to each other. Nowadays, this mathematical representation is referred to as the McCulloch–Pitts model.

Less than two decades later, Rosenblatt (1958) used this model to propose the first artificial neural network capable to gain experience through a supervised learning process. The system was coined perceptron. The author proved that for any finite set of linearly separable labeled examples, the ANN parameters can be updated iteratively and will converge after a finite number of steps. In other words, the learning algorithm produces parameters that correctly classify all training examples. The proof of convergence of the algorithm is known as the perceptron convergence theorem. This demonstration stimulated interest in the field, which went through ups and downs until more complex networks began to emerge (KROGH, 2008).

Figure 5 shows the model of a neuron, the information-processing unit that forms the basis for designing artificial neural networks. Neuron inputs are represented by **x** and are multiplied by its synaptic weights **w**, which can be positive (excitatory) or negative (inhibitory). The external threshold $w_0 = b$, called bias, is multiplied by a fixed input $x_0 = 1$ and is responsible for shifting the linear combiner from the origin. The weighted inputs are summed to form the induced local field (or activation potential) ν of the neuron. The resulting sum is applied to an activation function ϕ. Mathematically, the neuron output y is described by Equation (4).

$$y = \phi\left(\sum_{i=0}^{n} w_i x_i\right) \tag{4}$$

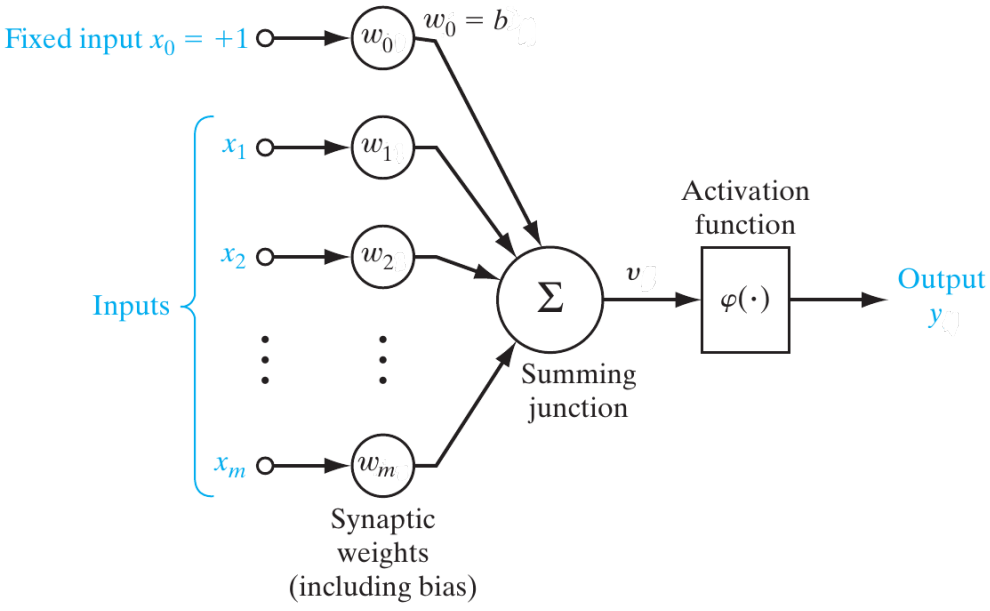


Figure 5 – A nonlinear model of a neuron.
Source: Haykin (1998)

A simple neuron, such as the perceptron, can solve only a very limited class of linearly separable problems. To overcome this limitation, networks with many neurons organized in layers can be used. One of these networks is known as multilayer perceptron (MLP) and consists of at least three layers of nodes: an input layer, one or more hidden layers, and an output layer. Except for the input nodes, each node is a neuron like the one previously described. Figure 6 illustrates the architectural graph of a MLP with two hidden layers. These networks exhibits a high degree of connectivity (generally fully connected, i.e. each node is connected to all nodes of the next layer) and each neuron in the network includes a nonlinear activation function that is differentiable (HAYKIN, 1998). A supervised learning algorithm known as backpropagation guides the training process to determine the network parameters (RUMELHART; HINTON; WILLIAMS, 1986), which takes place in two stages: forward phase and backward phase.

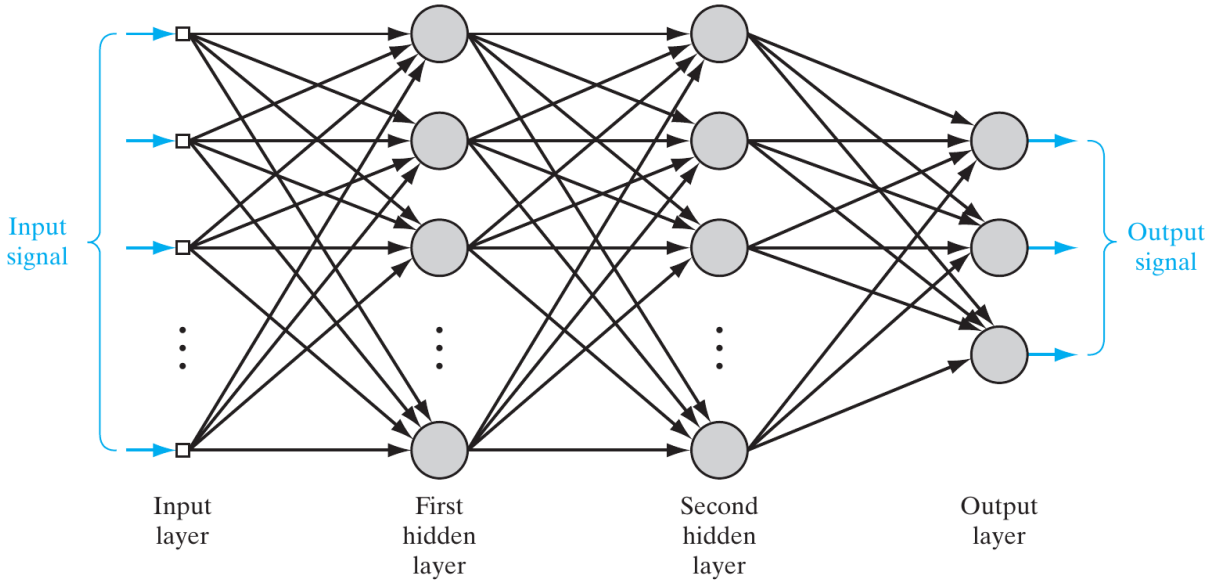


Figure 6 – Architectural graph of a multilayer perceptron with two hidden layers.
Source: Haykin (1998)

The training starts by assigning random weights to each of the connections in the network. In the forward phase, the input signal is propagated layer by layer through the network, until it reaches the output. The synaptic weights remain fixed and have not yet been updated. In the backward phase, the error signal is calculated by comparing the network output with the desired response. The resulting error signal is propagated layer by layer through the network, but in the backward direction (i.e., from the output nodes to the input nodes). Then, successive adjustments are made to the synaptic weights of the network aiming to minimize the error. The algorithm repeats this process iteratively, showing the training instances to the network and updating the weights until it reaches some stopping criteria, such as the maximum number of iterations. For mathematical details about the backpropagation method, see Haykin (1998).

2.2 WATER DEMAND FORECASTING

According to Billings & Jones (2011), water demand is defined as the total volume of water needed to supply customers within a certain period of time. Thus, the accurate forecast of this variable provides the basis for better planning, management and operation of water utilities (GARDINER; HERRINGTON, 1986; BILLINGS; JONES, 2011), making forward-looking information available as they conduct their business. It is useful to distinguish water demand forecasts based on different forecast horizons, considering that they are usually associated with the purpose of the task, the forecasting method that will be used, the variables with the greatest influence and the level of reliability of the predictions. Although there is no single categorization in the literature, a common distinction is long-, medium- and short-term forecasting (GHIASSI; ZIMBRA; SAIDANE, 2008; DONKOR; ROBERSON, 2014).

Table 2 exposes major applications associated with the water demand forecast horizons. As the forecast horizon extends, the forecast error tends to increase, as does the investment required for the applications. Long-term forecasts are developed for forecast horizons greater than a decade, playing an important role in sizing system capacity and the design of new water supply system (WSS). Medium-term forecasts range from years to a decade and are commonly used to promote improvements in the water treatment and distribution system and set water rates. Short-term forecasts, the type considered in this work, include horizons of up to one year and are responsible for supporting the management and optimization of system operations (e.g., optimal performance of pumps and reservoirs), along with budgetary and financial management.

Table 2 – Types of water demand forecasts and their main applications.

Forecast Type	Forecast Horizon	Applications
Long-term	Decades	Sizing system capacity, raw water supply
Medium-term	Years to a decade	Distribution system improvements, investments, setting water rates
Short-term	Hours to a year	Budget and financial management, system operations optimization

Source: adapted from Billings & Jones (2011)

In addition to planning levels and forecast horizons, several factors are considered influential in determining water demand. With respect to this, long-term forecasting may require completely different factors compared to a short-term equivalent (DONKOR; ROBERSON, 2014). For instance, population growth or decline is often associated with the trend (long-term effect) of water demand, while seasonal components (short-term effects) are mainly generated by meteorological data (BILLINGS; JONES, 2011). Other possible influencing factors are socioeconomic, geographic and calendar variables (weekdays, weekends, holidays and special events). Another key point that also needs to be considered is the particularities of each use case. In this work, for example, the forecast is made for a coastal tourist city, which implies an increase in water demand on holidays with high temperatures and summer periods.

All things considered, one can choose between different forecasting approaches to tackle the task at hand. In the context of short-term water demand forecasting, a wide variety

of methods have been proposed. Although there is no single convention, they can be broadly classified into standard statistical methods and learning algorithms (GUO et al., 2018). According to Niknam et al. (2022), traditional time series approaches like autoregressive integrated moving average (ARIMA) are the most popular choice in the literature, whilst the use of machine learning methodologies has grown considerably in the last years. The author states that among the most used ML algorithms are artificial neural network, support vector regression and hybrid models.

2.2.1 Related Work

Adamowski et al. (2012) compared five methods to predict daily urban water demand. The objective was to test a new method based on the coupling of discrete wavelet transforms and ANNs, which was compared with linear regression, nonlinear regression, autoregressive integrated moving average (ARIMA) and traditional artificial neural networks (ANNs). Two meteorological data were considered in the analysis, total daily precipitation and maximum temperature. The ANN wavelet models were found to provide more accurate urban water demand forecasts than the others. The author points out that the ANN and ARIMA models slightly underestimate and linear and nonlinear regression overestimate the high peaks of water demand, while the proposed model (wavelet ANN) provides the closest estimates.

Al-Zahrani & Abo-Monasar (2015) predict the daily water demand for the city of Al-Khobar based on historical water consumption and weather data. Daily minimum, maximum and average humidity and temperature, rainfall intensity, rainfall occurrence and wind speed data were included in the analysis. In order to investigate the impact of each variable, different subsets of the attributes were also considered. The authors created a model by combining an autoregressive moving average (ARMA) with an ANN and evaluating it by comparing it with the independent models. The results show that the combination of traditional time series approaches and ANN can improve predictions. With respect to the attributes, it's reported that temperature is the most important meteorological predictor. Humidity, wind speed and rainfall are also important, but cannot be used alone without temperature. On the other hand, rainfall intensity is the parameter that least contributes to the model's ability to predict water demand.

Kofinas et al. (2016) compared two methods, an adaptive neuro-fuzzy inference system (ANFIS) and an ANN, to forecast daily water demand in a Mediterranean touristic city. The data considered in this work include meteorological (daily mean temperature, daily maximum temperature and daily precipitation), social (daily tourist arrivals) and infrastructure attributes, where the last is an estimation of the leakage level in the network. Comparing the two methods, the authors found that ANFIS gives better results in all considered metrics. Furthermore, the two methods seem to overcome any problems related to the nonlinearity of the predictors.

Vijai & Sivakumar (2018) considered the techniques of linear regression, gaussian process regression (GPR), random forest, least square support vector machine (LSSVM), ANN, extreme learning machine (ELM) and deep learning. The water demand forecasting task was repeated using different time intervals of 1 hour, 12 hours and 24 hours. Categorical variables for

months (1 to 12) and hours (1 to 24) were added to the dataset, which also includes the following parameters: temperature, dew point temperature, relative humidity, wind direction and wind speed. As a result, the ANN model performs better, followed by LSSVM and GPR.

Zubaidi et al. (2020) applies a methodology that includes data preprocessing and two optimization algorithms to tune the hyperparameters of an ANN: backtracking search algorithm and crow search algorithm. The forecast is made for monthly water demand and singular spectrum analysis (SSA) is used to reduce time series noise. The authors conclude that preprocessing improves the data quality and the backtrack search algorithm is more efficient and accurate than the crow search algorithm. Furthermore, the study highlights that the use of algorithms to adjust hyperparameters improves methodology validation and reduces uncertainty.

Finally, Niknam et al. (2022) provides an extensive review of commonly used methods to forecast short-term urban water demand, discusses the impact of exogenous factors on water demand models and gives some future directions for this research area. In a similar review, Groppo, Costa & Libânio (2019) goes beyond forecasting methods and emphasizes the need to pre-process water demand time series to improve the accuracy of results, as previous research has shown that they have chaotic features. These works can serve as a guide for new projects.

3 METHODOLOGY

In this chapter, the methodology adopted to compare different methods for estimating the water demand in Guaratuba is discussed. First, the urban geography of the city and relevant characteristics are introduced in Section 3.1. Historical data from the water distribution system of the city are collected, along with additional meteorological and calendar data. In Section 3.2, the tools used to explore and prepare the raw data are presented. Once the data is preprocessed, it is ready to serve as input to the models. Thus, the techniques adopted for the selection and evaluation of models are discussed in Section 3.3. Three configurations (one expanding window and two sliding window) of a time series cross-validation method known as backtesting are used for each of the four supervised learning algorithms employed: linear regression, k-nearest neighbors, support vector regression and multilayer perceptron. Altogether, 12 models are compared (the results are addressed in Chapter 4). A summary of the different stages and the major tasks involved in each phase of the research is given in Figure 7.

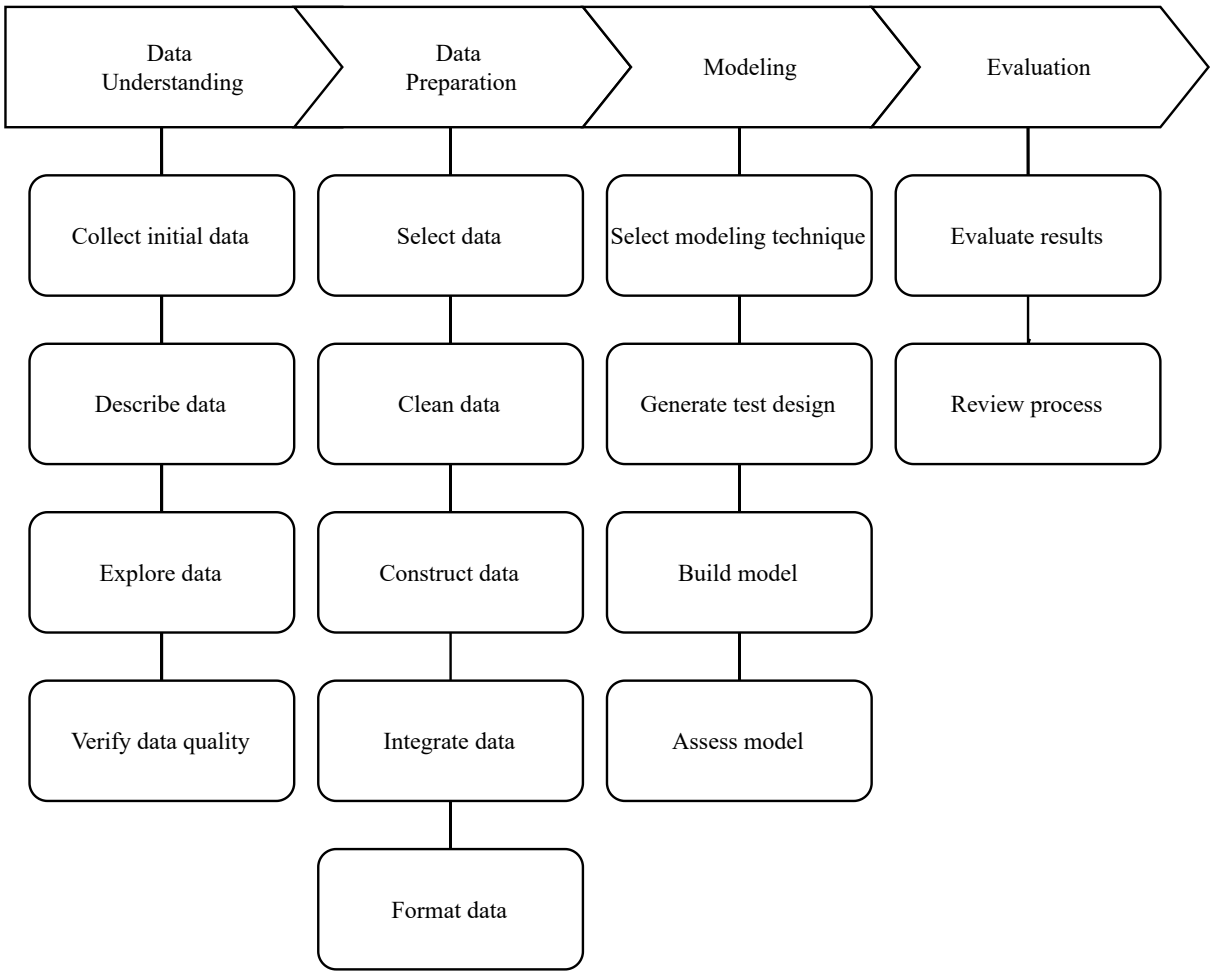


Figure 7 – Different stages and the major tasks involved in each phase of the research.
Source: adapted from Kelleher & Tierney (2018)

It is noteworthy that all software development carried out during this research used the Python programming language, in its version 3.8.13 (ROSSUM; DRAKE, 2009). Linear regression, k-nearest neighbours, support vector regression and multilayer perceptron, the four supervised learning algorithms used are available in the well-known scikit-learn machine learning framework and are trained and validated using its version 1.0.2 (PEDREGOSA et al., 2011).

3.1 STUDY AREA AND DATA

Guaratuba is located on the coast of the state of Paraná, Brazil, as shown in Figure 8. The city has 37.974 inhabitants according to IBGE (2021) estimates, with a large population flow in the summer months (December to March) due to one of its main economic activities: tourism. Consequently, in this period, water consumption patterns are different from those of the resident population, raising the values of per capita dues. (JUNIOR, 2021). For this work, two data related to the city's water distribution system (WDS) were made available for prediction (only one is selected for further analysis), and the additional data collected to model these consumption patterns can be divided into 2 groups: meteorological and calendar data.

The company that provides the water supply and sanitation service to Guaratuba (SANEPAR[1]) provided two types of historical information related to the municipal WDS: the

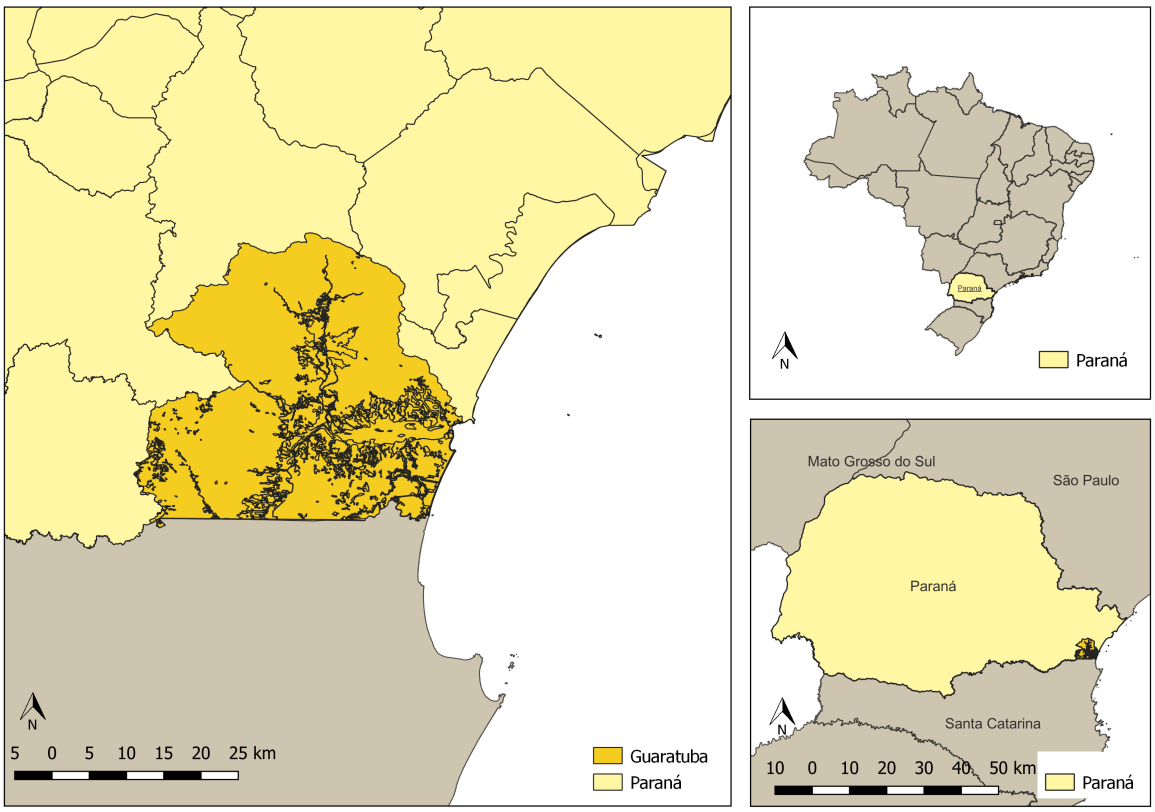


Figure 8 – City of Guaratuba highlighted on Brazil's map.

[1] The SANEPAR website is available at: <https://site.sanepar.com.br/>. Accessed on July 11, 2022.

daily volume of water consumed from the reservoirs; and the daily volume of water produced in the water treatment plants (WTPs). Information for both data is available over 4 years, from 2016 to 2019. It is important to note that only one is used for the prediction task, which aims to estimate the city's water demand. Details about this decision are discussed in Section 3.2.1.

The meteorological data were provided by the company that provides the state with meteorological, hydrological and environmental data (SIMEPAR[2]). According to Billings & Jones (2011), seasonal components in water use are generated mainly by climatic factors, making them critical for scheduling maintenance times for reservoirs, pumps and networks. For this work, the climate data available includes temperature, radiation, relative humidity and precipitation, and was collected from May 2015 to November 2020, with a frequency of 15 minutes. This data represents an ideal meteorological forecast for building the model, since it is only possible to obtain estimates for these values when the model is applied to real life.

Previous research showed that the number of tourists reached 2.597.392 in 2012 (accumulated throughout the year) on the coast of Paraná (JUNIOR, 2021). In an attempt to model this flow of people, data referring to holidays and school recess (classified as calendar data in this work) were collected. Holidays were included for Guaratuba and the two largest nearby cities with significantly large populations (over half a million people), Curitiba and Joinville. This data was collected from a web API (N.A., 2022) which is no longer available[3]. School recess was considered only for the state of Paraná, collected from the website of the state education and sport department (SEED/PR, 2022). In addition to summer and winter school holidays, recesses during the term period were also taken into account.

Table 3 summarizes the raw data considered in this work. Thus far, the data has remained intact. The following chapter goes on to understand and pre-process this data before it is used for further analysis. These two steps are distinct but composed of overlapping subtasks in nature, which are often performed iteratively. In brief, they aim to understand the data, transform it to make it suitable for training, and model features that best represent the underlying problem.

Table 3 – Raw data collected. They can be divided into three broad categories: target variable (i.e., water demand data), meteorological and calendar. For the time series, the frequency with which observations are made is shown.

Raw Data	Category	Frequency	Source
Water produced in the WTPs	Target variable	1 day	SANEPAR
Water consumed from the reservoirs	Target variable	1 day	SANEPAR
Temperature	Meteorological	15 minutes	SIMEPAR
Radiation	Meteorological	15 minutes	SIMEPAR
Relative humidity	Meteorological	15 minutes	SIMEPAR
Precipitation	Meteorological	15 minutes	SIMEPAR
Holidays	Calendar	-	N.A. (2022)
School recess (Paraná)	Calendar	-	SEED/PR (2022)

[2] The SIMEPAR website is available at: <http://www.simepar.br/>. Accessed on July 11, 2022.
[3] The API shows the following message: "Hello, unfortunately we no longer provide the holiday API service".

3.2 DATA UNDERSTANDING AND PREPARATION

The majority of real-world datasets are highly susceptible to missing, inconsistent and noisy values due to their heterogeneous origin. Despite being laborious and time-consuming steps, it is imperative to understand and prepare the raw data in a format suitable for further analysis (TAN; STEINBACH; KUMAR, 2016). Some of the tools used to understand data are briefly introduced in Section 3.2.1, a procedure known as exploratory data analysis. Likewise, techniques applied to data preprocessing are presented in Section 3.2.2. These sections only present the tools and techniques used, and the results are discussed in the next chapter.

3.2.1 Exploratory Data Analysis

Exploratory data analysis (EDA) is the first step before performing any changes to the raw data or developing a statistical model. The main goal of EDA is to gain general insights about the data that will potentially be helpful for further steps in the data analysis process (BERTHOLD et al., 2020). In this section, some of the procedures commonly adopted to achieve this objective are briefly presented and are classified into two types: univariate and multivariate analysis. The topic will not be discussed in-depth, as there is no standard methodology and approaches vary according to the type of data and which questions need to be answered for a better understanding of the problem at hand. Tukey et al. (1977) dedicates an entire book to discuss the topic.

The term univariate analysis refers to the analysis of only one variable, that is, it explores each variable separately. Typically, the goals of this step include checking for abnormal data, missing values, data distribution, and statistics that summarize and provide the gist of information about the sample data. Analysis can be done for both numerical and categorical data using the appropriate tool for each data type. With regards to numerical data, some patterns that can be easily identified with univariate analysis are central tendency (mean, mode and median), dispersion (range, variance), quartiles (interquartile range) and standard deviation. Graphical tools are helpful for visualising these statistics, such as the box plot, which depicts the central tendency, spread and skewness of numerical data through their quartiles.

On the other hand, the term multivariate analysis refers to the simultaneous observation and analysis of two (bivariate analysis) or more variables and is performed to understand the interactions between them. This can be done for both numeric and categorical data, as well as using both types of data in the same analysis. Typical tools include scatter plots, clustering analysis, and correlation measures that indicate whether two variables have any statistical relationship, for example, the Pearson correlation coefficient (PEARSON, 1894).

The Pearson's correlation coefficient is a measure for a linear relationship between two numerical variables **x** and **y**. When applied to a sample, the correlation coefficient is defined as

$$r(\mathbf{x}, \mathbf{y}) = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

where $\bar{x}$ and $\bar{y}$ are the sample mean values of **x** and **y**, respectively. The computation yields values between -1 and 1. The larger the absolute value of the coefficient, the stronger the linear relationship between the two variables. The sign of the coefficient represents the direction of the relationship. Positive coefficients indicate that when the value of one variable increases, the value of the other variable tends to increase. Negative coefficients represent cases when the value of one variable increases, and the value of the other variable tends to decrease.

3.2.2 Preprocessing

Data quality directly affects the learning ability of a machine learning algorithm. Thus, the purpose of preprocessing is to improve the quality of the available data and make them more suitable for further analysis (TAN; STEINBACH; KUMAR, 2016). In this section, some of the common employed preprocessing techniques are briefly presented. Roughly speaking, these techniques fall into two categories: selecting attributes for the analysis or creating/changing attributes. (TAN; STEINBACH; KUMAR, 2016). As for the previous section, the topic will not be discussed in depth considering that there is no standard methodology.

Based on the insights gained through exploratory data analysis, the first step is to select records and resources relevant to the task at hand. This can include creating new features to improve the modelling of the problem, as well as discarding some data that will not be useful for analysis. If the selected data has missing values, this must be taken into account. In fact, it is not uncommon to be missing one or more observations for an object. Depending on the data source, information might not be collected or a sensor might fail to read, for example. There are several strategies for dealing with this problem, and two of them are listed below.

A simple strategy is to eliminate objects with missing values. However, reliable analysis can be difficult or impossible if the proportion of objects removed is significant. Missing data can sometimes be estimated through a process known as imputation. For example, one can use the attribute values of the points closest to the point with the missing value to do the imputation. If the attribute is continuous, the average value of the nearest neighbour's attribute is used. On the other hand, if the attribute is categorical, the most frequently occurring attribute value can be taken. This is the essence of the k-nearest neighbors imputation algorithm, an efficient method to fill in missing data by a value obtained from related cases across the entire set of records.

Another common preprocessing approach for time series is resampling. Typically, data is unevenly spaced over time, requiring resampling for use in some models. Now taking into consideration a dataset with more than one time series, they are likely to be observed at different timestamps if they come from different sources. There are two types of resampling: upsampling and downsampling. The first increases the frequency of samples, such as from minutes to seconds. The latter decreases the frequency of samples, such as from days to months. Resampling can also be used to provide additional data and better model the problem at hand. For instance, one may calculate statistics for a time interval to create new attributes, such as taking the daily mean and the daily standard deviation of the observations collected at a rate of 15 minutes.

In contrast to creating new features, it is sometimes beneficial to reduce the number of dimensions in the dataset. This is the case of dimensionality reduction, where one can search for a subset of the input variables (feature selection) or transform the data from a high-dimensional space to a lower-dimensional space (feature extraction). Although no advanced dimensionality reduction technique is used in this work, this concept is used to combine redundant features and, consequently, reduce the dataset dimension. In extreme cases with thousands or tens of thousands of attributes, data analysis can become significantly more difficult as the number of input variables increases, a phenomenon called the curse of dimensionality (TAN; STEINBACH; KUMAR, 2016). A drawback is, however, the potential loss of interesting detail.

It may also be necessary to transform categorical and numeric attributes of the dataset, depending on the learning algorithm employed. As the methods used in this work require all input variables to be numeric, categorical data must be encoded. A natural encoding for ordinal variables is to convert each label to integer values. For instance, the weekdays (Sunday to Saturday) may be transformed into integers from 0 to 6[4]. In this case the encoded data represents and respects the sequence of labels. Likewise, it is necessary to bring all numerical features to a common scale, a step called feature scaling. The two most common resource scaling techniques are standardization and normalization. Standardization (also called z-score normalization) centres the values around zero with a unit standard deviation. Normalization, the technique used in this work, shifts and resizes the values to be between 0 and 1, as shown in Equation (5).

$$x_{scaled} = \frac{x - min(x)}{max(x) - min(x)} \tag{5}$$

More advanced preprocessing steps include, for example, noise reduction. Tan, Steinbach & Kumar (2016) defines noise as the random component of a measurement error, often used in connection with data that have a spatial or temporal component, such as images and time series, respectively. With respect to time series, it is useful to dissect into systematic and non-systematic components (SHMUELI; JR, 2016). The systematic components (i.e. level, trend and seasonality) characterize the underlying series, which always present some degree of noise (SHMUELI; JR, 2016). These components are commonly considered additive or multiplicative, as shown in Equation (6a) and Equation (6b), respectively. $y(t)$ is interpreted as an observation at the moment t, which is decomposed into the aforementioned components.

$$y(t) = level(t) + trend(t) + seasonality(t) + noise(t) \tag{6a}$$

$$y(t) = level(t) \cdot trend(t) \cdot seasonality(t) \cdot noise(t) \tag{6b}$$

In such cases, signal processing techniques can be adopted to reduce noise and help discover patterns that might be obfuscated. If ignored, the presence of noise can increase model complexity and learning time, and is often considered one of the main reasons for

[4] In the case of cyclic variables, the distance from the last to the first value does not correspond to reality. Another method of encoding a cyclic feature is to perform a sine and cosine transform of the feature.

overfitting (TAN; STEINBACH; KUMAR, 2016). To address this issue, singular spectrum analysis (SSA) is employed to reduce the noise of the target variable. This technique has proved to be very useful and has become a standard tool in the analysis of climatic, meteorological and geophysical time series (ZHIGLJAVSKY; GOLYANDINA, 2020; ZUBAIDI et al., 2020).

The SSA consists of two complementary stages: decomposition and reconstruction. At the first stage, the original series is decomposed into a sum of the independent and interpretable components. Then, in the second stage, the series is reconstructed into its elementary and ideally separable components. Following the description of the algorithm given by Zhigljavsky & Golyandina (2020) and Hassani (2007), the procedure can be divided into four steps, where the first and second belong to decomposition and the third and fourth to reconstruction.

These steps are summarized as follows: (i) embedding, a mapping that transfers the one-dimensional time series into a trajectory matrix; (ii) singular value decomposition, decompose the trajectory matrix and represents it as a sum of elementary matrices; (iii) eigentriple grouping, split the elementary matrices into several groups and sum the matrices within each group; (iv) diagonal averaging, transfers each matrix into a time series. Under those circumstances, the initial time series is decomposed into a sum of reconstructed components, which can be combined into a single time series to obtain the approximate original signal. The only parameters that must be set are the window length L and the number of components r used to group the reconstruct the series. The selection of such values followed the instructions of the authors mentioned above and is presented in Section 4.2, together with the results obtained from its application.

3.3 MODEL SELECTION AND EVALUATION

In this section, the procedures adopted for selecting and evaluating the machine learning models to estimate water demand in Guaratuba is described. As previously mentioned, the learning algorithms considered for the research are linear regression (LR), k-nearest neighbors (kNN), support vector regression (SVR), and multilayer perceptron (MLP). The theory of each algorithm has already been introduced in Section 2.1.3. The simple linear regression method is adopted as a baseline for empirical evaluation and is not intended to produce reasonable predictions, but rather to indicate how difficult it is to make accurate predictions.

Initially, the preprocessed data is split into three sets: train, validation and test. The training set is strictly reserved for training, while the validation set is used to optimize the model hyperparameters (model selection) and the test set to provide an unbiased evaluation of the selected model (model evaluation). To make a reasonable split considering the annual patterns that appear in water demand data, the first two years are reserved for training, the third for validation and the last for testing, as depicted in Figure 9. Having said that, the backtesting technique employed to assess model performance is elaborated in Section 3.3.1. The procedure to find the optimal hyperparameters of each model is described in Section 3.3.2. Finally, the performance metrics considered in the empirical evaluation are presented in Section 3.3.3.

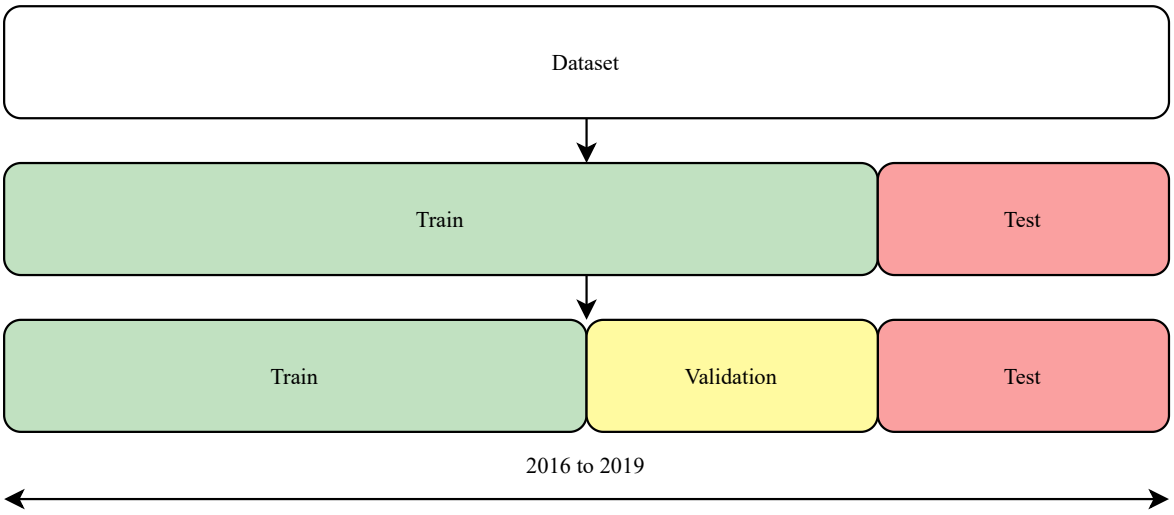


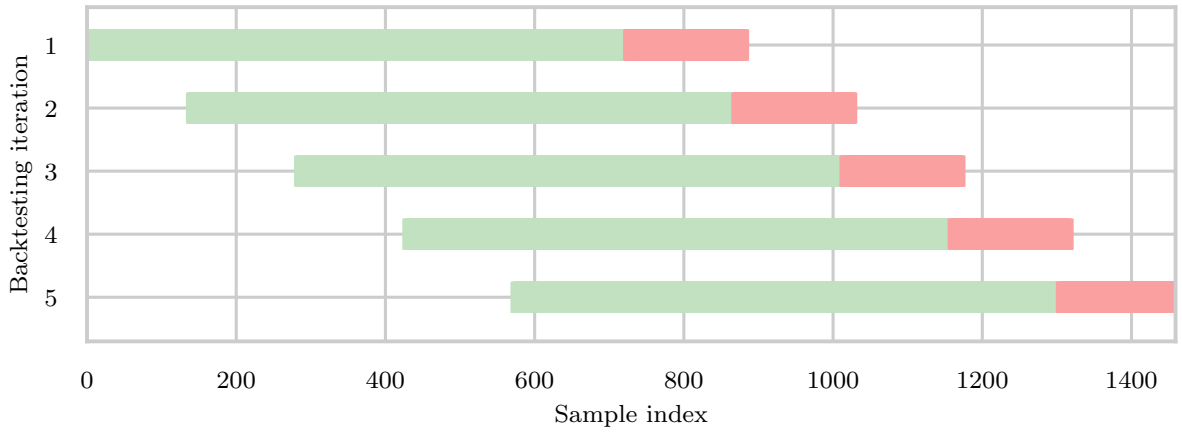
Figure 9 – Train, validation and test split. The first two years are strictly reserved for training. The third year is used for model selection (i.e., hyperparameter tuning). The fourth year is used to evaluate the models selected in the previous step.

3.3.1 Backtesting

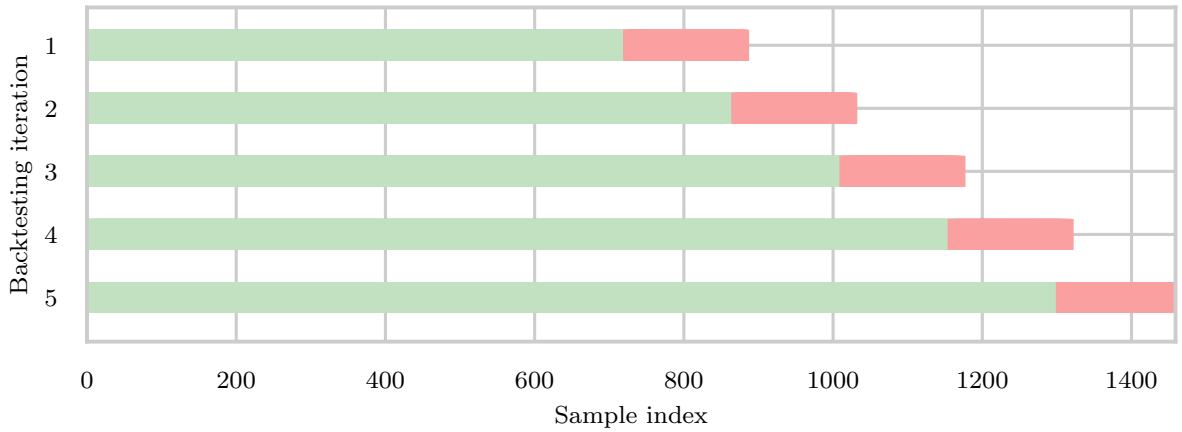
Traditional methods for evaluating machine learning models (e.g., k-fold cross-validation) can not be used directly with time series data, since they assume that observations are independently and identically distributed, i.e. each observation is independent and taken from the same probability distribution (BROWNLEE, 2017). This statement is not valid for time series data, as the temporal dimension of the observations implies that we must respect the temporal order in which the values were recorded. To address this problem, the procedure called backtesting was recently adopted by researchers and companies, although the naming conventions are still being established (UBER, 2019; UBER, 2020; GORDEEV et al., 2020).

Backtesting is the process of evaluating a model in different previous time periods, that is, data that has been collected and stored and can be used to simulate a real application scenario. In this procedure, there are a series of test sets and the corresponding training set consists only of observations that occurred prior to the observations that forms the test set (HYNDMAN; ATHANASOPOULOS, 2021). Thus, the performance metric of interest is calculated by averaging the results for each iteration. The outcome can be used for evaluating a model, as well as for finding optimal parameters and measuring the volatility of predictions over time (GORDEEV et al., 2020). The advantage is a more consistent performance estimate when compared to a single hold out split, applying cross-validation logic and still respecting the temporal order of the data. Backtesting can be categorized into two types: sliding window and expanding window.

The sliding window configuration requires three hyperparameters: training window size, test window size and sliding steps. When an iteration is completed, the training window slides according to the step size, resulting in the exclusion of a step-size data range at the beginning and the addition of a step-size data range at the end of the training set. Then, the prediction for the next test set is generated until the end of the data. Figure 10a illustrates the series of training



(a) Backtesting using a sliding window.



(b) Backtesting using an expanding window.

Figure 10 – Visualization of the backtesting behavior with sliding window (Figure 10a) and expanding window (Figure 10b). The training set is represented in green, while the test set is in red. In this case, both examples have 5 iterations.

and test sets, where the green observations form the training sets, and the red observations form the test sets. With this configuration, the training set has a fixed size and the model is able to forget past patterns that no longer reflect the current data in a new iteration.

The expanding window configuration also requires three hyperparameters: initial training window size, test window size and sliding steps. It is straightforward to note that the only difference from the sliding window is that the training set has a dynamic size (therefore only its initial size is defined), training the model with more data as it becomes available after each iteration. Figure 10b illustrates the series of training and test sets, where the green observations form the training sets, and the red observations form the test sets. While this configuration makes training take longer with each iteration due to adding more data, it can also improve model generalization due to greater diversity (i.e., more training examples) in the data.

In this work, three backtesting configurations are used with each learning algorithm, totaling 12 models. They are primarily employed to estimate model performance both during

Table 4 – The three backtesting configurations adopted to estimate the performance of models, which are performed during model selection (hyperparameter optimization) and in model evaluation. The * symbol indicates that the expanding window configuration uses different initial window sizes depending on the step: 730 during model selection performed on the validation set and 1095 during model evaluation performed on the test set. On the other hand, this value is fixed for the sliding window.

Window Type	Training Window Size	Test Window Size	Sliding Steps
Sliding window	365 days	7 days	7 days
Sliding window	730 days	7 days	7 days
Expanding window	*	7 days	7 days

model selection (performed on the validation set) and model evaluation (performed on the test set). The configurations are summarized in Table 4, consisting of one with an expanding window and two with a sliding window. It is noteworthy that the initial size of the training window for the expanding window configuration is 730 days (the first two years of data) for model selection, while it is 1095 days (the first three years of data) for the model evaluation. That is, the model uses the maximum number of observations available in the step under consideration (model selection or model evaluation) for initial training window size. For the two sliding window configurations, a training window size of one and two years (365 and 730 days, respectively) is used. Considering that the test window size and the sliding steps should make sense for the data in question, 7 days was chosen for both variables, as it is the smallest pattern expected from daily water produced data. These two parameters are the same for all three configurations.

3.3.2 Hyperparameter Optimization

Unlike parameters that the algorithm learns during the training phase, hyperparameters are settings that the user can tune to control the learning process (GOODFELLOW; BENGIO; COURVILLE, 2016). The procedure for finding the optimal set of hyperparameters for a model is called hyperparameter optimization and is performed on the validation set. The optimal configuration of the model is the one that minimizes a predefined loss function, that is, it has the lowest generalization error rate. Among the four methods used in this work, only linear regression does not have hyperparameters, and the others are tuned using the grid search technique.

Grid search is the most basic method of hyperparameter optimization (FEURER; HUTTER, 2019). It executes an exhaustive search through a manually specified subset of the hyperparameter space of a learning algorithm, given by the Cartesian product of these sets. Thus, for each hyperparameter configuration, the backtesting method is applied to the training set, resulting in multiple models and performance estimates. The hyperparameter settings that produced the best results in the backtesting procedure for each of the 12 models are selected and used in the independent test set withheld earlier to evaluate their final performance.

In this work, root mean square error (RMSE) was chosen as the scoring metric to be optimized (the metric is described in Section 3.3.3). Issues that may arise from its use, such as

Table 5 – Search space for the learning algorithms. The symbol † indicates that the values are evenly spaced in logarithmic space. For instance, considering $[1, 1000], 4$ ([start, stop], number of samples), the following values would be sampled: 1, 10, 100 and 1000.

Learning Algorithm	Hyperparameter	Search Space
Linear Regression	–	–
K-Nearest Neighbors	n_neighbors	{3, 5, ..., 29}
	weights	{uniform, distance}
	metric	{euclidean, manhattan}
Support Vector Regression	kernel	{linear, poly, rbf, sigmoid}
	C	[0.001, 100], 10†
	epsilon	[0.001, 0.1], 10†
	gamma	{scale, auto}
Multilayer Perceptron	hidden_layer_sizes	{(11,), (13,), ..., (23,)}
	activation	{tanh, relu}
	solver	{lbfgs, sgd, adam}
	alpha	[0.01, 1.0], 10†
	learning_rate_init	[0.0001, 0.01], 5†

the heavy penalty weight for larger errors, are not covered. The search space for each method is depicted in Table 5. For the numerical hyperparameters, a reasonable range was chosen[5]. Some of them are sampled in logarithmic space, aiming to explore a wider range. Beside three MLP parameters (shuffle=False, early_stopping=True, max_iter=1000), the others not reported use the default values defined by the scikit-learn framework in its version 1.0.2.

3.3.3 Performance Metrics

Performance metrics provide a summary of the capability of the model that performed the predictions. These metrics can be used to monitor and measure the performance of a model during training, model selection, and testing. There are many different performance metrics to choose from, and those adopted in this work to evaluate the performance of the models on the test set are described in this section. Namely, they are: root mean square error, mean absolute error, mean absolute percentage error and r-squared. Regression models have continuous output. Therefore, the metrics are based on calculating some sort of distance between the predicted and the actual value. In the equations below, y_i represents the target value of the ith observation, $\hat{y}_i$ is the value obtained with a forecast model, and n the number of predicted values.

Root mean square error (RMSE) is given by the square root of the average of the squared difference between the target value and the value predicted by the regression model, as shown in Equation (7). The calculation of the square root of the error means that the units of the RMSE are the same as the original units of the target value that is being predicted. Noteworthy,

[5] It is important to highlight that this was not the focus of the research, therefore, the values were established based on the author's knowledge. One could combine a random search and a grid search, taking random samples in space and creating additional grids with a finer resolution where the performance is good.

this metric puts more emphasis on larger absolute errors (SAMMUT; WEBB, 2011).

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{7}$$

Mean absolute error (MAE) is defined as the average of the absolute values of the prediction errors, as shown in Equation (8). Each prediction error is given by the difference between the true value and the predicted value. The error score unit matches the target value unit being predicted. Different from RMSE, the MAE does not give more or less weight to different errors and instead the scores increase linearly with increases in error (BROWNLEE, 2017).

$$MAE = \frac{1}{n}\sum_{i=1}^{n} \mid y_i - \hat{y}_i \mid \tag{8}$$

Mean absolute percentage error (MAPE) is the average of the absolute percentage errors of forecasts, as shown in Equation (9). The percent error is a measure of the discrepancy between the observed value and the true value, being summed without taking into account the sign to calculate the MAPE. This measurement is easy to understand because it gives the error in terms of percentages. Despite this, it has some drawbacks that are discussed in (TOFALLIS, 2015).

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \tag{9}$$

Unlike the error metrics mentioned above, r-squared (R^2), also known as the coefficient of determination, is not an error metric. It is, however, the ratio of the variance of the predicted values to the variance of the true values. Equation (10) shows the R^2 formula, where $\overline{y}$ is the average of all observed values. The R^2 output is an interpretable value that typically ranges from 0 to 1. The closer the score is to 0, the less the variance of the predicted output is associated with the variance of the true values, and this is due to the prediction error. On the other hand, the closer the value is to 1, the greater the similarity between them and, therefore, the error in the variance of the prediction is smaller. R^2 can reach negative values if the model's performance is worst than assuming each prediction equals the mean of the observed values.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \overline{y})^2} \tag{10}$$

Thus far, this work has detailed all the materials and methods used throughout the research. The next chapter brings it all together to present the results. Initially, the results of exploratory data analysis and preprocessing are discussed, establishing the dataset used for the final analysis. Finally, the performance comparison between the 12 considered models is presented and discussed using the metrics discussed in this section.

4 RESULTS AND DISCUSSION

This chapter describes the results obtained in the present work. First, insights obtained during exploratory data analysis (EDA) are presented in Section 4.1. These outcomes are of fundamental importance and support the decisions taken in preprocessing, which are presented in Section 4.2. The optimal hyperparameters found in model selection are described in Section 4.3. Lastly, the performance of the models is compared in Section 4.4.

4.1 EXPLORATORY DATA ANALYSIS

The first data explored were the two candidates for the target variable given by the company that provides the water supply service to Guaratuba (SANEPAR): the daily volume of water consumed in the reservoirs and the daily volume of water produced in the water treatment plants (WTPs). For convenience, the former will be referred to as water consumed and the latter as water produced. Considering the 1461 days that make up the years in which these data were made available (from 2016 to 2019), while the water produced data are complete, 117 observations on water consumed are missing. According to SANEPAR, such problems are related to measurement errors, since the reading equipment is subject to communication failures, generated by power outages and fluctuations, among other factors (JUNIOR, 2021). In addition to missing values, these failures can also result in reading invalid values that do not represent reality. Anomalies like this were also identified in the water consumed data. For instance, 368 observations with values greater than the maximum value of water produced were found.

These values are inconsistent with the volume of water produced in the municipality. In fact, they make it difficult to visualize the underlying structure of the time series, which motivated the application of a filter to remove incoherent observations. As the produced water data does not contain inconsistencies, all data referring to the consumed water that is above or below the maximum and minimum value of produced water, respectively, were removed. This operation resulted in the removal of 369 observations and retrieving this data (e.g., using an imputation method) has become difficult due to the large amount of missing values. After applying the filter, the number of missing observations corresponds to 486, equivalent to 33.26% of the data.

With the remaining data, the Pearson's correlation coefficient was computed to measure the strength of association with water produced. Note that the pairwise correlation for the missing observations is ignored. The results indicate a strong positive linear relationship, with a correlation coefficient of 0.8 over the 4 years available. Figure 11 illustrates the time series that represent the water produced and water consumed, separating each year in a subplot. The observed patterns are similar and the period between August and October 2017 is where the data differs the most. All things considered, water consumption data were discarded from further analysis due to low quality, causing produced water data to be used as the target variable. Note

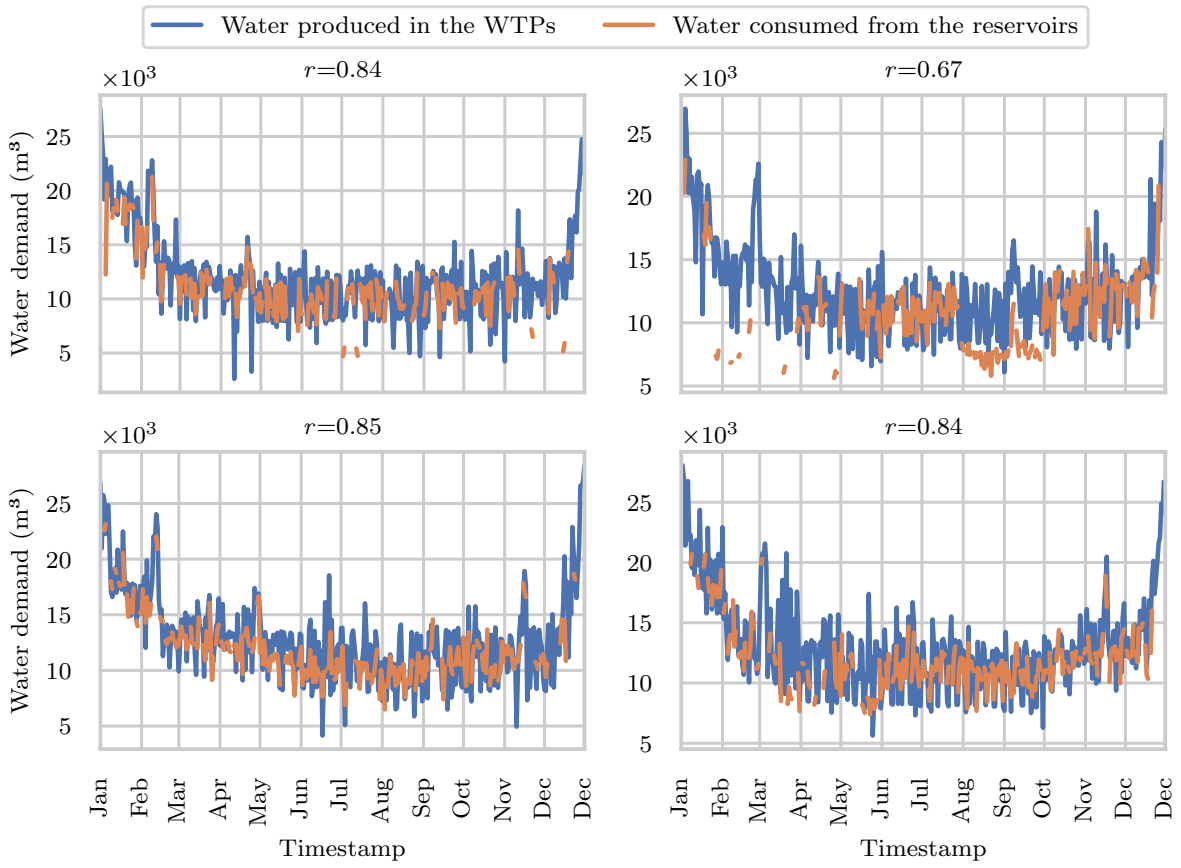


Figure 11 – Water produced in the water treatment plants and water consumed in the reservoirs of Guaratuba during the four years of available data: 2016 (upper left), 2017 (upper right), 2018 (lower left) and 2019 (lower right). Pearson's correlation coefficient between the two variables for each year (assuming the same order as before) is 0.84, 0.67, 0.86 and 0.82, respectively. Over all years the correlation is 0.8.

that the analysis is still valid according to the definition of water demand discussed at the beginning of Section 2.2. As stated by Billings & Jones (2011), the system's water demand and the total water production in a public supply are basically equivalent concepts.

This initial exploration also made it possible to observe significant peaks in water demand during the Carnival period, a pattern that is repeated over the years and deserves special attention to avoid notable errors. These peaks in water consumption can be seen in Figure 11, represented by the sudden increase that occurred in February for the years 2016, 2017 and 2018 and in early March for 2019. These results are not surprising, as Guaratuba is a coastal city that attracts many tourists, especially during festive periods such as Carnival. Following the same reasoning, the time series shows an increase in water demand from December to March due to summer tourism, reaching its maximum at the turn of the year.

Meteorological and calendar data were also subjected to exploratory analysis. In terms of meteorological data, only 5 values are missing. All weather observations (i.e., temperature, radiation, relative humidity and precipitation) from January 5 to 9, 2019 were not present in the dataset. It is likely that the system has stopped working in this period of time, either for

maintenance or due to some technical problem, since the values are missing for all meteorological variables. Regarding calendar data, the similarity between the holidays in Guaratuba, Joinville and Curitiba was investigated using the Jaccard index. The results show a high degree of similarity, with a value of 0.88 between the three. In the next section, the insights obtained from EDA are used to support data preprocessing decisions and obtain the dataset for final analysis.

4.2 PREPROCESSING

The preprocessing results are outlined below. First, new features derived from timestamps are added to the dataset. Next, the changes applied to calendar and meteorological data are discussed. The last preprocessing step involves noise reduction of the water demand data. Thus, the final dataset is obtained and summarized at the end of this section.

Dates and times are rich sources of information that are intrinsic to each observation of the time series data. As a consequence, it is possible to create different features from timestamps that provide strong and ideally simple relationships between inputs and outputs for the learning algorithm to model (BROWNLEE, 2017). In this work, the following attributes were created from timestamps: year, month, day, day of the week (0 to 6), weekend (0 or 1) and season of the year (0 to 3). For the ordinal attributes (day of the week and season of the year), each label is converted to integer values and the encoded data represents the sequence of labels. The binary attribute (weekend) is represented as an integer variable that only accepts the values 0 or 1.

Next, a special feature was created for Carnival, as it was observed that this period has a significant impact on the city's water consumption. In addition to the official public holidays (Monday, Tuesday and Wednesday), the dates of Friday, Saturday and Sunday (before the official days) were also included, covering the entire festive period in which people go to the coast to enjoy the beach. Besides that, it was decided to consider the union of the Guaratuba, Curitiba and Joinville holidays in a single attribute. As noted during EDA, these variables have high similarity and do not add more information to the model if left separate. Generally, analyzing data with a smaller number of dimensions is preferable and tends to avoid the difficulties associated with analyzing high-dimensional data (TAN; STEINBACH; KUMAR, 2016). All holiday-related attributes are binary and represented as an integer variable that only accepts the values 0 or 1.

In the case of meteorological data (i.e., temperature, radiation, relative humidity and precipitation), three steps were carried out: resampling, imputation and scaling. The 15-minute resolution data were resampled on a daily scale, computing two statistics for each variable: daily mean and daily standard deviation. The former informs the central tendency of the values throughout the day and the latter the amount of variability. Following this, the 5 missing observations found during EDA were imputed using k-nearest neighbors (kNN) algorithm. Euclidean distance and $k = 5$ (weighting neighbors by the inverse of their distance) were used to configure the imputer. The last step consists of normalizing the data between 0 and 1.

Aiming to reduce the noise of the water demand time series, singular spectrum analysis

(SSA) was applied. The choice of window length (L) and leading eigentriples used to reconstruct the signal (r) were based on (HASSANI, 2010). To obtain better separability, the author advises taking the window length proportional to a potential periodic component with an integer period that the time series can have. The shortest period of water demand is expected to be one week, that is, 7 days. As it is advisable to choose L reasonably large, but less than half of the total length (which is 730 in this case), $L = 560$ was chosen. The parameter r was selected by examining the weighted correlation (called w-correlation) matrix. In particular, strongly correlated components must be placed in the same group. Figure 12 shows the w-correlations for the 560 components reconstructed in a gray scale from white to black, corresponding to the absolute values of correlations from 0 to 1. Based on this information, we selected the first 108 eigentriples for the reconstruction of the original series and the remainder is interpreted as the beginning of noise.

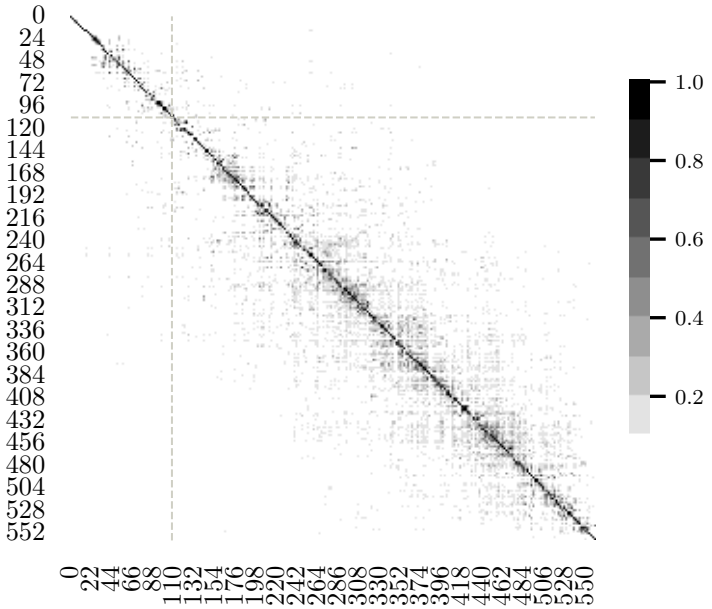


Figure 12 – Matrix of weighted correlations for the 560 water demand components decomposed by the singular spectrum analysis algorithm. The heat map is in a gray scale from white to black, corresponding to the absolute values of correlations from 0 to 1. As a result, the first 108 eigentriples are used for the reconstruction of the original signal and the rest are interpreted as noise. Residuals seem to have no evident structure.

The residuals (i.e., the difference between the original and reconstructed series) seem to have no evident structure – it is close to a normal distribution centered in zero. Thus, in this case the smoothing procedure leads to noise reduction and the smoothed curve describes the signal.

To conclude this section, the attributes that comprises the final dataset used for further analysis are summarized in Table 6. The extra day of the 2016 leap year has been removed so that every year has the same number of observations, totaling 1460 from 2016 to 2019. The dataset has 17 attributes (excluding the target variable), 6 of which are categorical and 11 are numeric. It is worth mentioning again that the categorical attributes were encoded (each label received an integer value) and the numerical ones normalized between 0 and 1.

Table 6 – Attributes that compose the database used for analysis. The dataset has 18 dimensions and the target variable is highlighted in bold.

Attribute	Description	Type
water_produced	Water produced by the WTPs of Guaratuba (m^3)	Ratio
year	Year of the record	Interval
month	Month of the record	Interval
day	Day of the record	Interval
day_of_week	Day of the week of the record	Interval
is_weekend	Indicates whether it is a weekend or not	Nominal
season	Season of the record	Interval
temperature_mean	Daily average temperature (°C)	Interval
temperature_std	Daily standard deviation of temperature (°C)	Ratio
radiation_mean	Daily average radiation	Ratio
radiation_std	Daily standard deviation of radiation	Ratio
relative_humidity_mean	Daily average relative humidity	Ratio
relative_humidity_std	Daily standard deviation of relative humidity	Ratio
precipitation_mean	Daily average precipitation	Ratio
precipitation_std	Daily standard deviation of precipitation	Ratio
is_holiday_ctba_gtba_jve	Indicates whether it is a public holiday	Nominal
is_carnival	Indicates if it is carnival week	Nominal
is_school_recess_pr	Indicates whether it is recess or school break	Nominal

4.3 MODEL SELECTION

Four supervised learning algorithms were adopted in this work, namely, linear regression (LR), k-nearest neighbors (kNN), support vector regression (SVR) and multilayer perceptron (MLP). For each one, three backtesting configurations were employed to evaluate the model's performance, one expanding window and two sliding windows (details are shown in Table 4). Therefore, a total of 12 models were quantitatively compared. The dataset, composed of the attributes of Table 6, was split into train (2016 and 2017), validation (2018) and test (2019). Under those circumstances, the hyperparameters were optimized in the validation set (model selection) and only the selected configuration is evaluated in the test set.

To achieve that, the 12 models were subjected to hyperparameter optimization using an exhaustive grid search. The search space of each method (depicted in Table 5) consists of a single combination for the LR algorithm (that is, this method doesn't have hyperparameters), 56 for the kNN, 800 for the SVR and 2100 for the MLP. The optimal parameters obtained for each learning algorithm are reported in the Table 7. Note that, for most parameters, the optimal set surrounds two minimums of the function being optimized. For reference, the metrics computed on the validation for the selected configurations are shown in Appendix A.

Table 7 – Optimal hyperparameters for the 12 models, determined through grid search.

	Hyperparameter	Backtesting Configuration		
		EW-7D	SW-1Y7D	SW-2Y7D
LR	–	–	–	–
kNN	n_neighbors	9	7	7
	weights	distance	distance	uniform
	metric	manhattan	manhattan	euclidean
SVR	kernel	poly	poly	poly
	C	0.59948	27.82559	0.59948
	epsilon	0.02154	0.001668	0.02154
	gamma	scale	auto	scale
MLP	hidden_layer_sizes	(11,)	(15,)	(19,)
	activation	relu	tanh	relu
	solver	lbfgs	lbfgs	lbfgs
	alpha	0.01668	0.01	0.01
	learning_rate_init	0.0001	0.0001	0.0001

4.4 MODEL EVALUATION

The models with their optimal hyperparameter were evaluated according to the metrics root mean square error (RMSE), mean absolute error (MAE) and mean absolute percentage error (MAPE). Table 8 shows these error metrics computed on test set (model evaluation). The table is separated into three parts, one for each backtesting configuration used. The first refers to the expanding window. The second and third refer to the sliding window, with a training window size of 365 and 730 days, respectively. Each one presents the results for the four learning algorithms in the following order (from top to bottom): LR, kNN, SVR and MLP. Next, the overall results are first analyzed in relation to the best performance of each machine learning algorithm. Subsequently, the effects of each backtesting configuration are discussed.

For each metric, the best performance of each algorithm in the test set is highlighted in bold. Overall, the best performing model is a MLP with a test RMSE of 1709.70 and a test MAE of 1408.97. This means that on average, the absolute difference between the actual and predicted values for daily water demand is 1408.97 m^3. In sequence, the best methods are SVR (RMSE = 1852.52), kNN (RMSE = 1866.11) and LR (RMSE = 2148.36). The great performance of MLP and SVR justify being the two most used machine learning models for forecasting water demand (NIKNAM et al., 2022). On the other hand, despite the great trade-off between performance and complexity, kNN is rarely used in the context of water demand, which leaves room for researchers to develop models using this method. Lastly, perhaps the most serious disadvantage of the LR method is that it cannot model the nonlinear relationship between the drivers of water demand. In fact, this is the only learning algorithm used in this work that cannot model nonlinear relationships between variables.

Table 8 – Performance of the 12 models in the test set. The table is divided into 3 parts with the results for each backtesting configuration adopted: expanding window (EW-7D), one-year size sliding window (SW-1Y7D), and two-year size sliding window (SW-2Y7D). The performance measures are computed for each backtesting iteration and the results are averaged over the rounds to give an estimate of the model's predictive performance. On the right side of each metric is informed its standard deviation. For each metric, the model with the lowest error for each learning algorithm is highlighted in bold.

	Model	RMSE	σ	MAE	σ	MAPE (%)	σ (%)
EW-7D	LR	**2148.36**	929.992	**1832.11**	883.323	15.03	6.507
	KNN	**1866.11**	972.392	**1590.67**	948.388	**12.30**	5.078
	SVR	**1852.52**	647.234	**1525.26**	544.143	**12.12**	4.373
	MLP	**1709.70**	593.104	**1408.97**	515.158	**11.81**	4.707
SW-1Y7D	LR	2403.62	1010.830	2093.66	955.716	16.01	5.430
	KNN	1906.99	969.900	1618.19	954.202	12.47	5.055
	SVR	2022.69	846.400	1685.90	744.767	13.08	4.318
	MLP	2142.58	927.998	1796.94	855.901	14.11	5.841
SW-2Y7D	LR	2178.25	870.311	1870.36	814.996	**14.88**	5.341
	KNN	1968.28	1008.130	1668.98	963.966	12.98	5.370
	SVR	1943.40	783.327	1611.67	648.252	12.67	4.558
	MLP	1927.41	732.005	1620.15	630.532	13.08	4.916

With respect to the three backtesting configurations used, it is clear that models with expanding window achieve better performance. Each learning algorithm had its best model using this configuration (only the MAPE for LR does not). For the results with expanding window (first part of the table), MLP is the method with best performance (RMSE = 1709.70), while LR has the worst (RMSE = 2148.36). Regarding the sliding window with a size of 365 days (second part of the table), kNN is the best (RMSE = 1906.99) and LR the worst (RMSE = 2403.62). In fact, with the exception of kNN, all other methods had the worst performance in this configuration. Finally, the results using a sliding window of size 730 days are presented in the last part of the table. In this scenario, MLP has the best performance (RMSE = 1927.41) and LR has the worst (RMSE = 2178.25), confirming the poor performance of the linear model regardless of the backtesting configuration adopted since it was the worst model in the three.

While it might be intuitive to expect a time series model to improve with more historical data, this is not always the case. Previous research has shown that in some situations the patterns observed in the data can change significantly (e.g. changes in hygiene behavior, with the consequent effect on water consumption), making it preferable to use smaller sized sliding windows for training and forgetting old behavior that no longer reflects the current situation (BATA et al., 2019). Therefore, these results need to be interpreted with caution and are highly dependent on factors intrinsic to the location and the data collected. For example, the time period considered in this work is relatively small (only 4 years) when compared to works such as Adamowski et al. (2012) which considers 9 years of data. The impacts of different backtesting configurations are likely to be more visible in a scenario where data is collected over a longer

period of time. However, most studies in this field have focused only on hold-out strategies and do not consider the limitations of these traditional methods of model evaluation.

A more detailed analysis was carried out to observe the performance of the models in specific months. For this, only the best model of each learning algorithm is considered – that is, the models that use backtesting with the expanding window. The results are shown in Figure 13, where each box plot summarizes the absolute error of each model by month. It is worth noting that this error is calculated in the test set. Interestingly, all models had their worst and best monthly performance in December and September, respectively.

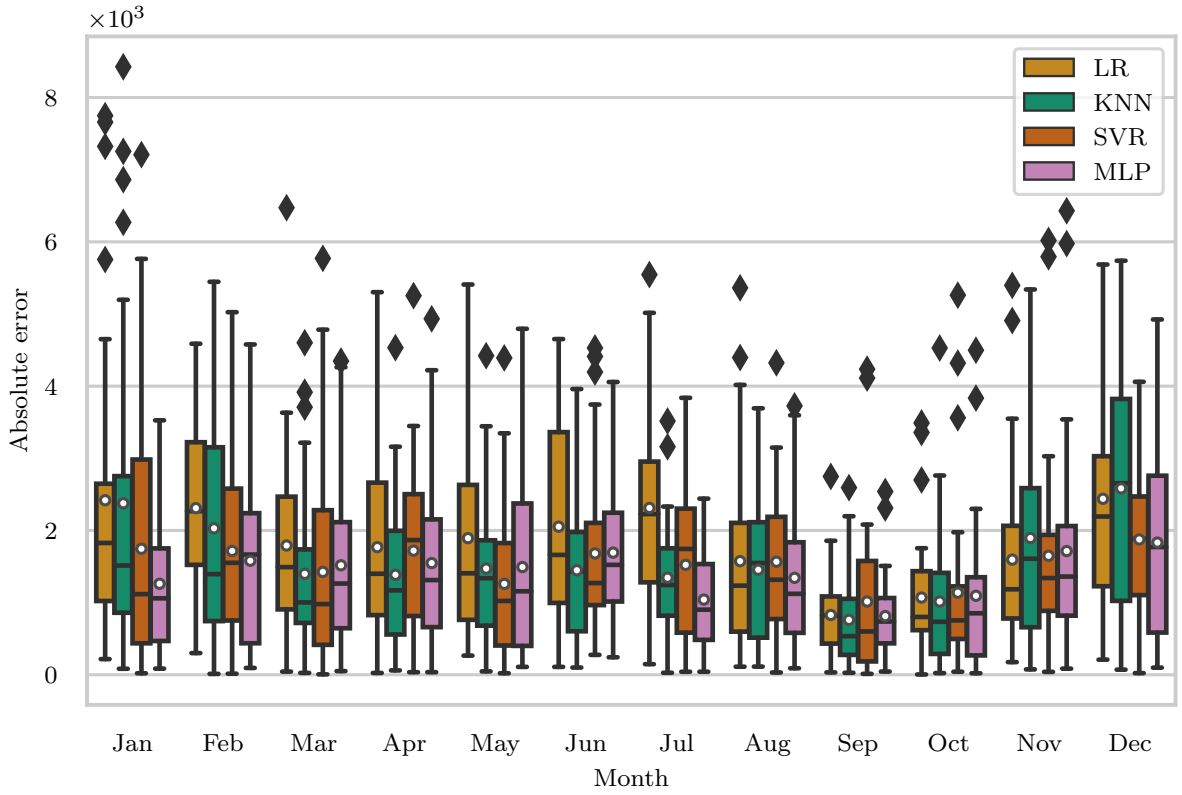


Figure 13 – Box plot of the absolute error (in m^3) for each month of the test set (2019). Only the model with the best performance of each algorithm is considered. The symbol ○ accounts for the mean which, in turn, is the MAE for that month.

Considering the month of December (worst performance), the models produced by the algorithms LR, kNN, SVR and MLP obtained a MAE of 2438.78, 2583.26, 1875.48 and 1831.90, respectively. Although more investigations are needed, it is believed that it is very difficult to model the water demand at this time of year due to the large flow of people that occurs in the city. In addition to December being the month with the worst MAE performance, January is the month with the highest absolute errors (note the outliers in Figure 13), with the exception of the MLP model. In the same order, but for the month of September (best performance), the models obtained a MAE of 829.07, 760.91, 1014.57 and 812.52. Nor were patterns found to explain such performance, which may even be by chance – the behavior of water demand was less chaotic this month. In fact, in addition to September, the October results also perform well overall.

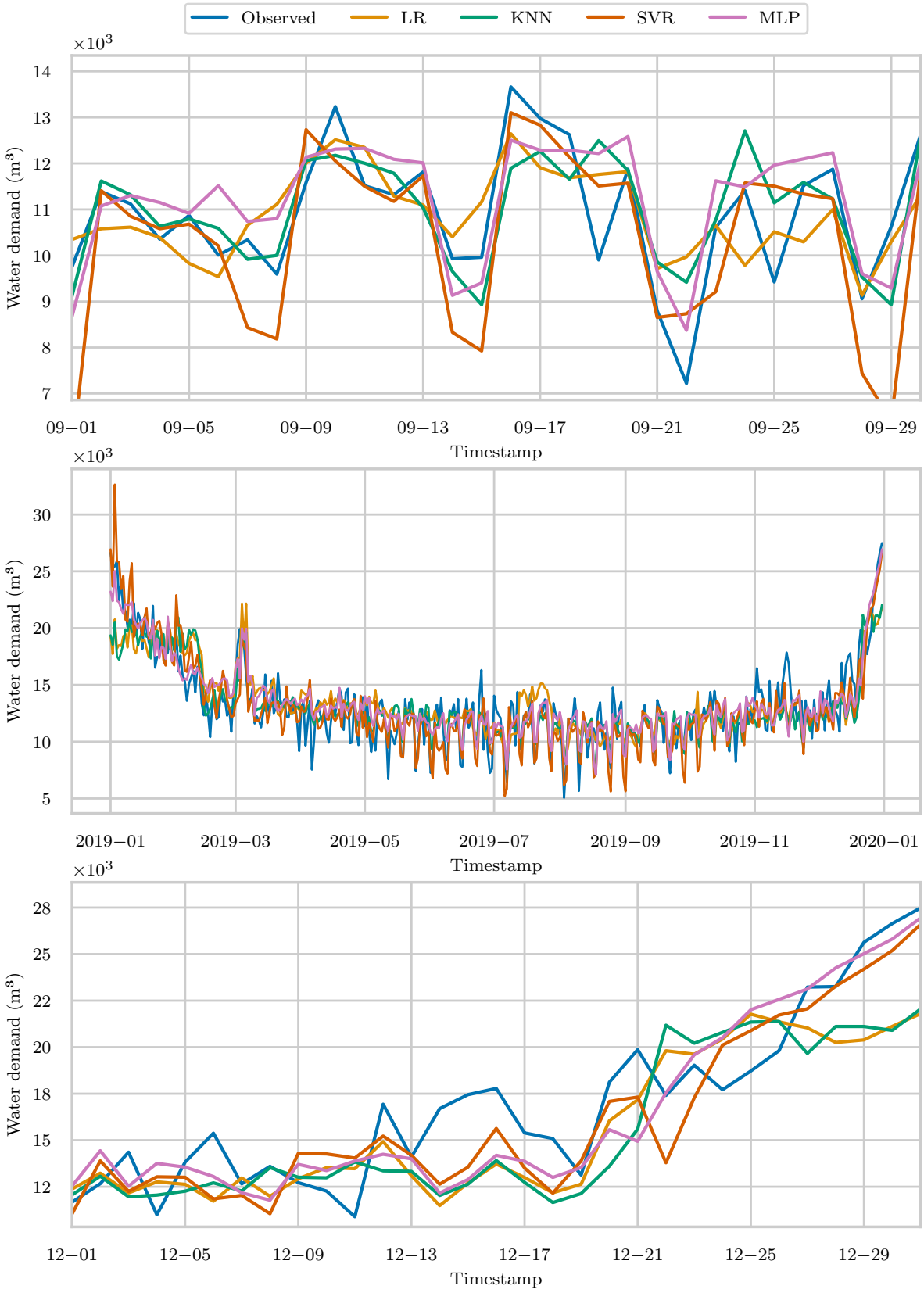


Figure 14 – Daily water demand forecasts for the test set (2019). Only the model with the best performance of each algorithm is considered. At the top, the month in which such models performed best (September) is zoomed in and shown in more detail. At the bottom, the same is done with the worst performing month (December).

To graphically observe the performance of the models in these two months, the predictions are shown in Figure 14. The central part of the image exposes the results of the entire test set, while the upper and lower parts show the results for the months of September and December, respectively, in greater detail. Just as before, these results are only for the best model of each learning algorithm. Analyzing the forecasts for September (best performance), it is possible to clearly notice a seasonality in the water demand data, which has a period of one week and is repeated four times throughout the month. By contrast, these patterns are not observed during the month of December. There is, however, a difficulty of the models to follow the values of the target variable. Approximately during the 13th and 18th, all methods show a significant error. This is also very visible from the 26th, but in this case only for the LR and kNN

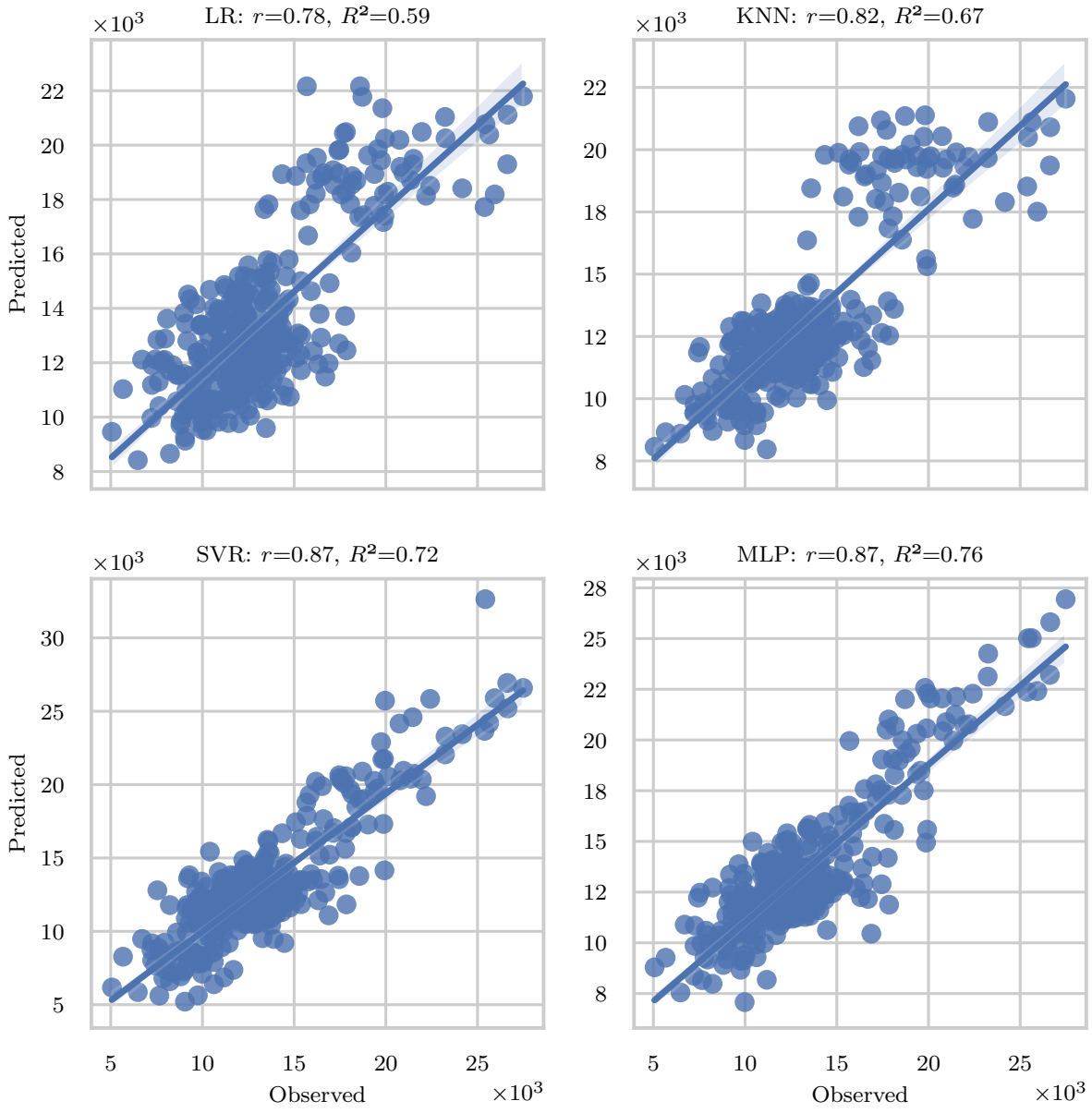


Figure 15 – Scatter plot comparing the observed and predicted water demand for the test set. Only the model with the best performance of each algorithm is considered.

methods. As a tourist coastal city, Guaratuba receives a large number of temporary inhabitants during the summer. This is enhanced in December and January, when schools go into recess and a large part of the population enjoys the warm days on the beaches. In fact, water demand reaches its highest value in the period of the New Year celebrations.

Finally, Figure 15 depicts the scatter plots comparing the observed and predicted urban water demand for the test set. The more the observed and predicted data agree, the more the scatters will be concentrated near the line of identity. The scatters fall on the identity line exactly when the observed and predicted datasets are numerically equal. The results reinforce that the SVR and MLP models have less dispersed estimates, and there a superior predictive capacity, than LR and kNN. Importantly, unlike the error metrics computed for each iteration of the backtesting and average, the R^2 and Pearson correlation coefficient (which is also shown in the title of each image) were computed only once by considering all predictions.

5 CONCLUSIONS

In this work, different machine learning algorithms were studied and quantitatively compared in the task of predicting daily urban water demand. Over the short term, accurate forecasting of water consumption plays an important role in the optimal operation of water collection, treatment, and distribution systems. In particular, the results can be adopted to optimize pump and reservoir operations, as well as inform the population about peaks in demand. The study area of the present research is a Brazilian coastal city called Guaratuba, located in the state of Paraná. Nevertheless, the methodology for building the models can also serve as a template[1] for other cities, given the availability of data and considering their particularities.

Due to the characteristics of a coastal tourist city, the application scenario for Guaratuba is challenging. The reasons for such complexity arise from the nature of the available data and the variables that influence water consumption, in addition to the large population variation experienced by the city in the summer months. To tackle this problem, a historical dataset containing information on meteorological and calendar data was collected, as well as the water demand itself. The observations cover the period from 2016 to 2019. In order to understand and prepare the collected data for further analysis, they were submitted to exploratory analysis and preprocessing. Although these are laborious and time-consuming steps, the quality of the data directly affects the learning ability of a machine learning algorithm. Thus, from the available raw data the attributes that compose the dataset were selected, cleaned and transformed.

In sequence, the experimental setup was built by splitting the dataset into training (2016 and 2017), validation (2018), and test (2019) sets. The models were initially subjected to hyperparameter optimization (model selection), performed on the validation set using the grid search algorithm. Subsequently, the models with their optimal parameters were evaluated on the test set (model evaluation) using the following performance metrics: RMSE, MAE, MAPE, and R^2. To evaluate the generalizability of the models (both during model selection and evaluation), three backtesting configurations (two sliding windows and one expanding window) were employed. With respect to the supervised learning algorithms, LR, kNN, SVR and MLP were adopted. Thus, totaling 12 models, where all went through the same process.

The empirical results underscore the importance of using nonlinear models to predict short-term water demand. Based on the adopted performance metrics, MLP performed the best, while LR was the worst. SVR and kNN were third and fourth, respectively. Besides that, the two main methods (MLP and SVR) provided more reliable estimates of their error compared to the others (kNN and LR), since the computed metrics are less spread out. With reference to the three backtesting configurations employed, each learning algorithm had its best model using the expanding window. All things considered, the models showed satisfactory results.

[1] The code is available on <https://github.com/jesuinovieira/bachelor-thesis>. Accessed on August 1, 2022.

5.1 FUTURE WORK

Possible future research stemming from this study may include: (i) work on better modeling of the attributes that influence water demand. For instance, it would be interesting to add attributes that model the flow of tourists in the city. One can also create time-shifted water consumption features based on autocorrelation, since they are intrinsic to the time series data; (ii) employ a feature selection method in order to get less redundant and misleading data; (iii) finally, with a well-established dataset, it would be possible to explore the application of different algorithms, such as traditional time series forecasting as well as ensemble methods.

BIBLIOGRAPHY

ADAMOWSKI, J. et al. Comparison of multiple linear and nonlinear regression, autoregressive integrated moving average, artificial neural network, and wavelet artificial neural network methods for urban water demand forecasting in montreal, canada. **Water Resources Research**, Wiley Online Library, v. 48, n. 1, 2012.

AL-ZAHRANI, M. A.; ABO-MONASAR, A. Urban residential water demand prediction based on artificial neural networks and time series models. **Water Resources Management**, Springer, v. 29, n. 10, p. 3651–3662, 2015.

ARBUES, F.; GARCIA-VALINAS, M. A.; MARTINEZ-ESPINEIRA, R. Estimation of residential water demand: a state-of-the-art review. **The Journal of Socio-Economics**, v. 32, n. 1, p. 81–102, 2003. ISSN 1053-5357. Available from Internet: <https://www.sciencedirect.com/science/article/pii/S1053535703000052>.

BATA, M. et al. Smart water: Short-term forecasting application in water utilities. University of Windsor, 2019.

BERTHOLD, M. R. et al. Data understanding. In: **Guide to Intelligent Data Science**. [S.l.]: Springer, 2020. p. 33–83.

BILLINGS, R. B.; JONES, C. V. **Forecasting urban water demand**. [S.l.]: American Water Works Association, 2011.

BISHOP, C. M.; NASRABADI, N. M. **Pattern recognition and machine learning**. [S.l.]: Springer, 2006. v. 4.

BONTEMPI, G.; BEN TAIEB, S.; BORGNE, Y.-A. L. Machine learning strategies for time series forecasting. In: SPRINGER. **European business intelligence summer school**. [S.l.], 2012. p. 62–77.

BROWNLEE, J. **Introduction to time series forecasting with python: how to prepare data and develop models to predict the future**. [S.l.]: Machine Learning Mastery, 2017.

CORTES, C.; VAPNIK, V. Support-vector networks. **Machine learning**, Springer, v. 20, n. 3, p. 273–297, 1995.

EMMANUEL A. DONKOR, S. T. A. M. R. S.; J. ALAN ROBERSON, P. Urban water demand forecasting: Review of methods and models. **Journal of Water Resources Planning and Management**, 2014.

DRUCKER, H. et al. Support vector regression machines. **Advances in neural information processing systems**, v. 9, 1996.

FEURER, M.; HUTTER, F. Hyperparameter optimization. In: **Automated machine learning**. [S.l.]: Springer, Cham, 2019. p. 3–33.

FLACH, P. **Machine learning: the art and science of algorithms that make sense of data**. [S.l.]: Cambridge university press, 2012.

GARDINER, V.; HERRINGTON, P. **Water demand forecasting**. [S.l.]: CRC Press, 1986.

GHIASSI, M.; ZIMBRA, D. K.; SAIDANE, H. Urban water demand forecasting with a dynamic artificial neural network model. **Journal of Water Resources Planning and Management**, American Society of Civil Engineers, v. 134, n. 2, p. 138–146, 2008.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. **Deep Learning**. [S.l.]: MIT Press, 2016. <http://www.deeplearningbook.org>.

GORDEEV, D. et al. Backtesting the predictability of covid-19. **arXiv preprint arXiv:2007.11411**, 2020.

SOUZA GROPPO, G. de; COSTA, M. A.; LIBÂNIO, M. Predicting water demand: A review of the methods employed and future possibilities. **Water Supply**, IWA Publishing, v. 19, n. 8, p. 2179–2198, 2019.

GUO, G. et al. Short-term water demand forecast based on deep learning method. **Journal of Water Resources Planning and Management**, v. 144, n. 12, p. 04018076, 2018. Available from Internet: <https://ascelibrary.org/doi/abs/10.1061/%28ASCE%29WR.1943-5452.0000992>.

HASSANI, H. Singular spectrum analysis: methodology and comparison. Cardiff University and Central Bank of the Islamic Republic of Iran, 2007.

HASSANI, H. A brief introduction to singular spectrum analysis. **Optimal decisions in statistics and data analysis**, 2010.

HAYKIN, S. **Neural Networks: A Comprehensive Foundation**. 2nd. ed. USA: Prentice Hall PTR, 1998. ISBN 0132733501.

HYNDMAN, R. J.; ATHANASOPOULOS, G. **Forecasting: principles and practice**. 3rd. ed. OTexts, 2021. Available from Internet: <https://otexts.com/fpp3/>.

IBGE. **Censo Brasileiro de 2021**. 2021. Available from Internet: <https://www.ibge.gov.br/cidades-e-estados/pr/guaratuba.html>.

JORDAN, M. I.; MITCHELL, T. M. Machine learning: Trends, perspectives, and prospects. **Science**, American Association for the Advancement of Science, v. 349, n. 6245, p. 255–260, 2015.

CARVALHO JUNIOR, M. **Consumo e perdas no sistema de abastecimento de água de Guaratuba, Pontal do Paraná e Matinhos (Litoral do Paraná)**. 2021. Bachelor Thesis (Environmental and Sanitary Engineering B.S.), UFPR (Universidade Federal do Paraná), Curitiba, Brazil.

KELLEHER, J. D.; TIERNEY, B. **Data science**. [S.l.]: MIT Press, 2018.

KOFINAS, D. et al. Daily multivariate forecasting of water demand in a touristic island with the use of artificial neural network and adaptive neuro-fuzzy inference system. In: IEEE. **2016 International Workshop on Cyber-physical Systems for Smart Water Networks (CySWater)**. [S.l.], 2016. p. 37–42.

KROGH, A. What are artificial neural networks? **Nature biotechnology**, Nature Publishing Group, v. 26, n. 2, p. 195–197, 2008.

LONES, M. A. How to avoid machine learning pitfalls: a guide for academic researchers. **arXiv preprint arXiv:2108.02497**, 2021.

MCCULLOCH, W. S.; PITTS, W. A logical calculus of the ideas immanent in nervous activity. **The bulletin of mathematical biophysics**, Springer, v. 5, n. 4, p. 115–133, 1943.

MITCHELL, T. M. **Machine Learning**. 1. ed. New York, NY, USA: McGraw-Hill, Inc., 1997. ISBN 0070428077, 9780070428072.

MURPHY, K. P. **Probabilistic machine learning: an introduction**. [S.l.]: MIT press, 2022.

N.A. **API Feriados Municipais e Estaduais**. 2022. Available from Internet: <https://www.calendario.com.br/>.

NIKNAM, A. et al. A critical review of short-term water demand forecasting tools—what method should i use? **Sustainability**, MDPI, v. 14, n. 9, p. 5412, 2022.

PEARSON, K. Contributions to the mathematical theory of evolution. **Philosophical Transactions of the Royal Society of London. A**, JSTOR, v. 185, p. 71–110, 1894.

PEDREGOSA, F. et al. Scikit-learn: Machine learning in Python. **Journal of Machine Learning Research**, v. 12, p. 2825–2830, 2011.

PRADHAN, K. **A Quick Tutorial on Clustering for Data Science Professionals**. 2021. Available from Internet: <https://www.analyticsvidhya.com/blog/2021/11/quick-tutorial-clustering-data-science/>.

RASCHKA, S. Model evaluation, model selection, and algorithm selection in machine learning. **arXiv preprint arXiv:1811.12808**, 2018.

RILEY, P. **Three pitfalls to avoid in machine learning**. [S.l.]: Nature Publishing Group, 2019.

RODRÍGUEZ-PÉREZ, R.; BAJORATH, J. Evolution of support vector machine and regression modeling in chemoinformatics and drug discovery. **Journal of Computer-Aided Molecular Design**, Springer, p. 1–8, 2022.

ROSENBLATT, F. The perceptron: a probabilistic model for information storage and organization in the brain. **Psychological review**, American Psychological Association, v. 65, n. 6, p. 386, 1958.

VAN ROSSUM, G.; DRAKE, F. L. **Python 3 Reference Manual**. Scotts Valley, CA: CreateSpace, 2009. ISBN 1441412697.

RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. Learning representations by back-propagating errors. **nature**, Nature Publishing Group, v. 323, n. 6088, p. 533–536, 1986.

SAMMUT, C.; WEBB, G. I. **Encyclopedia of machine learning**. [S.l.]: Springer Science & Business Media, 2011.

SEED/PR. **Calendário Escolar**. 2022. Available from Internet: <http://www.gestaoescolar.diaadia.pr.gov.br/modules/conteudo/conteudo.php?conteudo=27>.

SHMUELI, G.; LICHTENDAHL JR, K. C. **Practical time series forecasting with r: A hands-on guide**. [S.l.]: Axelrod schnall publishers, 2016.

SINGAPORE, P. U. B. Managing the water distribution network with a smart water grid. **Smart Water**, v. 1, n. 1, p. 4, Jul 2016. ISSN 2198-2619. Available from Internet: <https://doi.org/10.1186/s40713-016-0004-4>.

SMOLA, A. J.; SCHÖLKOPF, B. A tutorial on support vector regression. **Statistics and computing**, Springer, v. 14, n. 3, p. 199–222, 2004.

TAN, P.-N.; STEINBACH, M.; KUMAR, V. **Introduction to data mining**. [S.l.]: Pearson Education India, 2016.

TOFALLIS, C. A better measure of relative prediction accuracy for model selection and model estimation. **Journal of the Operational Research Society**, Springer, v. 66, n. 8, p. 1352–1362, 2015.

TUKEY, J. W. et al. **Exploratory data analysis**. [S.l.]: Reading, MA, 1977. v. 2.

UBER. **Omphalos, uber's parallel and language-extensible time series backtesting tool**. 2019. Available from Internet: <https://eng.uber.com/omphalos/>.

UBER. **Building a backtesting service to measure model performance at uber-scale**. 2020. Available from Internet: <https://eng.uber.com/backtesting-at-scale/>.

UN-WATER. **UN World Water Development Report 2021**. 2021. Available from Internet: <https://www.unwater.org/publications/un-world-water-development-report-2021/>.

VIJAI, P.; SIVAKUMAR, P. B. Performance comparison of techniques for water demand forecasting. **Procedia computer science**, Elsevier, v. 143, p. 258–266, 2018.

XU, R.; WUNSCH, D. Survey of clustering algorithms. **IEEE Transactions on neural networks**, Ieee, v. 16, n. 3, p. 645–678, 2005.

ZHANG, F.; O'DONNELL, L. J. Support vector regression. In: **Machine Learning**. [S.l.]: Elsevier, 2020. p. 123–140.

ZHIGLJAVSKY, A.; GOLYANDINA, N. **Singular Spectrum Analysis for Time Series**. 2nd. ed. [S.l.]: Springer, 2020.

ZUBAIDI, S. L. et al. Urban water demand prediction for a city that suffers from climate change and population growth: Gauteng province case study. **Water (Switzerland)**, v. 12, p. 1–17, 2020. ISSN 20734441. The authors applied an ANN optimized with the Backtracking Search Algorithm (BSA-ANN) to estimate monthly water demand in relation to previous water consumption.

1. Normality and outlier test.
2. Normalization, cleaning and selection of the best model inputs.
3. Mutual Information (MI) technique was used to choose the best explanatory variables (Lag 1 to Lag 4).
4. BSA-ANN results are slightly better than ANN stand-alone.

A VALIDATION SET RESULTS

Table 9 – Performance of the 12 models in the validation set. The table is divided into 3 parts with the results for each backtesting configuration adopted: expanding window (EW-7D), one-year size sliding window (SW-1Y7D), and two-year size sliding window (SW-2Y7D). The performance measures are computed for each backtesting iteration and the results are averaged over the rounds to give an estimate of the model's performance. On the right side of each metric is informed its standard deviation. For each metric, the model with the lowest error for each learning algorithm is highlighted in bold.

	Model	RMSE	σ	MAE	σ	MAPE (%)	σ (%)
EW-7D	LR	1827.16	999.037	1521.70	927.636	12.61	6.965
	KNN	**1739.35**	901.891	1450.26	789.476	**11.41**	4.264
	SVR	**1590.20**	526.033	**1346.94**	479.260	**10.86**	3.503
	MLP	**1551.54**	562.100	1320.07	494.239	11.17	4.958
SW-1Y7D	LR	2216.35	1313.900	1967.68	1289.070	14.85	6.680
	KNN	1758.94	957.670	1469.43	819.429	11.54	4.479
	SVR	1678.06	976.950	1428.82	906.154	11.02	4.623
	MLP	1730.33	696.772	1450.47	602.628	11.56	4.036
SW-2Y7D	LR	**1785.67**	926.404	**1494.10**	861.468	**12.28**	6.137
	KNN	1760.99	832.125	**1448.85**	731.945	11.51	4.489
	SVR	1631.53	594.857	1382.73	534.234	11.21	3.987
	MLP	1555.62	604.619	**1298.03**	493.535	**10.81**	4.610