Universidade de Aveiro
Ano 2022

MIGUEL CRUZ MATOS

SISTEMA DE GESTÃO DE ENERGIA BASEADO EM AI/ML PARA COMUNIDADES DE ENERGIA RENOVÁVEL

Universidade de Aveiro
Ano 2022

MIGUEL CRUZ MATOS

SISTEMA DE GESTÃO DE ENERGIA BASEADO EM AI/ML PARA COMUNIDADES DE ENERGIA RENOVÁVEL

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Informática, realizada sob a orientação científica do Doutor José Alberto Gouveia Fonseca, Professor Associado do Departamento de Eletrónica, Telecomunicações e Informática, da Universidade de Aveiro, do Doutor Paulo Jorge de Campos Bartolomeu, Professor Auxiliar do Departamento de Eletrónica, Telecomunicações e Informática, da Universidade de Aveiro, e do Doutor João Miguel Pereira de Almeida, Investigador do Instituto de Telecomunicações.

Este trabalho é financiado pelo Fundo Europeu de Desenvolvimento Regional (FEDER), através do Programa Operacional Regional de Lisboa (POR LISBOA 2020) e do Programa Operacional Regional do Centro (CENTRO 2020) do Portugal 2020 [Projeto COMSOLVE com o nº 047019 (CENTRO-01-0247-FEDER-047019)];

Dedico este trabalho à minha família, que nunca deixou de me apoiar.

o júri

presidente

Prof. Doutora Pétia Georgieva Georgieva
professora associado da Universidade de Aveiro

Prof. Doutor Pedro Nuno da Silva Faria
investigador auxiliar do Instituto Superior de Engenharia do Porto

Prof. Doutor Paulo Jorge de Campos Bartolomeu
professor auxiliar em regime laboral da Universidade de Aveiro

agradecimentos

Em primeiro lugar, não posso deixar de agradecer aos meus orientadores, pela paciência e apoio que sempre me deram durante a realização deste trabalho. Mas, principalmente, quero agradecer à minha família, aos meus pais e à minha irmã, que sempre acreditaram em mim e nas minhas capacidades.

palavras-chave

Machine learning, inteligência artificial, comunidades de energia renovável

resumo

A necessidade de reorganização do mercado de energia, com o objetivo de reduzir o consumo de energia de fontes não renováveis, levou à criação de Comunidades de Energia Renovável, que permitem que os seus membros partilhem a sua energia produzida e armazenada entre si. O presente trabalho propõe um estudo sobre um sistema de gestão desta comunidade, usando técnicas de AI/ML para a previsão do consumo de eletricidade. Prevê-se que, com a utilização destas técnicas, o sistema de gestão conseguirá diminuir o preço da fatura de eletricidade da comunidade, ou a redução do consumo de energia proveniente da rede de distribuição.

keywords Machine learning, artificial intelligence, renewable energy communities

abstract The need for a reorganization of the energy market, with the goal of reducing the energy consumption from non-renewable sources, led to the creation of Renewable Energy Communities, which allow their members to share their produced and stored energy among themselves. The present work proposes a study of a management system of this community, using AI/ML techniques for the energy consumption forecast. It is predicted that, with the use of these techniques, the management system will be able to decrease the electricity bill of the community, or the reduction of energy consumption from the distribution grid.

Index

1. Introduction

The need for change in the energy sector, with the goal of reducing the electric consumption from non-renewable sources, prompted the emergence of a new paradigm based on the role of prosumers, which are energy consumers with capacity to produce and store electricity. Recently, a new operation of electricity markets using *peer-to-peer* (P2P) architectures was proposed, allowing the prosumers to directly share their produced or stored energy with other consumers, therefore reducing the usage of "non-green" energy. These P2P markets focus on a *bottom-up* perspective and in an architecture centred on the consumer, allowing the prosumers to organize themselves collectively and cooperate in the operational management of energy, composing a Renewable Energy Community (REC). This reorganization of the electricity markets, with a decentralized management and based in cooperative principles, has the potential to promote the empowerment of the prosumers, while also pleasing their energy consumption preferences, such as renewable energy sources, CO2 gas emissions and the proximity to the energy production place of origin. In this P2P architecture, all peers (prosumers) cooperate with their available resources to produce, trade, or distribute an asset or a service. The P2P and community concepts have been applied under the principle of cooperative economy, easing the transaction amongst all economic agents.

The goal of this thesis is the investigation and implementation of prediction models of consumption of electricity by members of a REC, as well as the evolution of the buy and sell prices of energy in the electricity market. Based on the estimations, it will be possible for the management entity of the community, or each individual member, to perform a more efficient management of the produced, consumed, and stored energy. Machine Learning algorithms will perform the task of obtaining these estimations as trustworthy as possible, while learning existing patterns in a time series, such as the differences between the production and consumption of energy during summer and winter, or between day and night, weekdays, and weekends, etc. From the developed models, it will be possible to optimize the energetic balance of the community, evaluating in real-time the current price and the future estimation of when to buy, sell the electricity to the distribution network and making decisions on storage, buying energy for future usage, or selling the produced excess.

2. State of the Art

This project concerns mainly with Machine Learning (ML) and how it can be used to predict the consumption and production of electricity by members of a REC. Therefore, in this chapter, ML is explained, while also diving into some of its methods, comparing them in how they may or may not benefit this project.

2.1. Machine Learning

Machine Learning (ML) is a field of Artificial Intelligence (AI) that concerns with the question of how to construct computer algorithms that automatically improve with experience [1]. ML has the goal of making predictions or finding patterns without being specifically programmed to do so, finding them because they were learnt with past data. Typically, a ML model is built using a range of techniques and methods that tune their parameters to reduce errors and fit the data as it is made available to them. This process is known as *training*.

The ML models can be trained with distinct types of data, labelled and unlabelled. **Labelled data** have both input and output parameters, where the output are the values to predict, given the input parameters. For example, for a use case where we would try to predict a house price using ML, a dataset with several houses and their features, such as dimensions, location, number floors, and price, can be used as a labelled dataset, where the price is the output parameter, and the other features are the input parameters. Usually, a labelled dataset is obtained with human labour, which can, therefore, be more expensive to obtain. **Unlabelled data**, on the other hand, have no meaningful tags or distinctive features, i.e., there is no specific value to predict. For example, a dataset with users and the movies they liked has no output parameter, yet it can be used to group users according to the similarity of their tastes and used for a recommendation algorithm [2].

There are three main techniques in ML: **Supervised Learning**, which is trained with labelled data and mainly categorized as *Classification*, where the input produces a discrete output, and *Regression*, where the input produces a continuous output [3]; **Unsupervised Learning**, trained with unlabelled data and usually categorized as *Clustering*, the process of joining items into subgroups based on their similarities. This opposes Supervised Learning, since the goal is not to make predictions, but rather finding statistical patterns in data [4]; and **Reinforcement Learning**, where the algorithms act as agents that are rewarded or punished, depending on their behaviour, therefore learning through trial and error [3].

2.1.1. Artificial Neural Networks

An ANN is a ML algorithm that was created as a loose mathematical representation of neurons on a biological brain and how they worked. According to the scientific knowledge of the time of its creation, neurons could be represented as a set of nodes, each connected to other nodes, transmitting signals to their connections. In an ANN, this "signal" is a real number, and the output of each neuron is computed by some non-linear function of the sum of its inputs. Neurons and their connections (edges) typically have a weight that adjusts as learning proceeds. The weight increases or decreases the strength of the signal at a connection [5].

A. Multilayer Perceptron (MLP)

MLP is known as the simplest type of feedforward ANN. The MLP neural network has a layer of input neurons, where each can be mapped to a feature. Each neuron will then send its single output to the group of neurons of which it is connected to, in another layer of neurons

(hidden layers). This process is repeated until it reaches the final layer, the output layer, where the task will be accomplished, such as recognizing an object in an image [6]. Figure 1documents an example of an MLP Neural Network, with n nodes on the input layer, one hidden layer with k nodes and a final layer with a single output.

To find the output of the neuron, first we must take the weighted sum of all the inputs, weighted by the weights of the connections from the inputs to the neuron. We add a bias term to this sum. This weighted sum is sometimes called activation. This weighted sum is then passed through an activation function to produce the output [6].

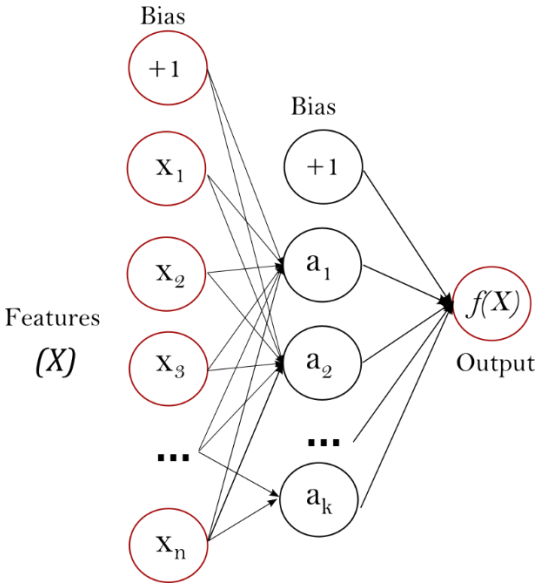

Figure 1 – MLP Neural Network example (image obtained from [7]).

B. Long Short-Term Memory Network (LSTM)

Long short-term memory (LSTM) is an artificial recurrent neural network (RNN). A RNN is a special type of ANN, which differs from them because each neuron can have feedback connections to store representations of recent input events in form of activations [8]. Consequently, the output of a neuron will have an impact on its future output. This is ideal for sequential data, such as speech, video, or data in form of time series. With RNN architectures, however, it can be difficult to solve problems that require learning long-term temporal dependencies. This is because of the vanishing gradient problem, where the gradient of the loss function decays exponentially with time.

A LSTM unit is composed of a **memory cell**, which can maintain information in memory for extended periods of time, an **input gate**, an **output gate**, and a **forget gate**. LSTM, unlike RNNs, which overwrite their content at each time-step, can decide whether to keep the existing memory via the introduced gates. These gates, therefore, control the information that enters the memory, that is forgotten, and that leaves as the output. Intuitively, if the LSTM unit detects an important feature from an input sequence at early stage, it easily carries this information (the existence of the feature) over a long distance, hence, capturing potential long-distance dependencies [9].

C. Adaptive neuro fuzzy inference system

An adaptive neuro-fuzzy inference system or adaptive network-based fuzzy inference system (ANFIS) is an artificial neural network that is based on Takagi–Sugeno fuzzy inference system.

The technique was developed in the early 1990s. Since it integrates both neural networks and fuzzy logic principles, it has potential to capture the benefits of both in a single framework. Its inference system corresponds to a set of fuzzy IF–THEN rules that have learning capability to approximate nonlinear functions. Hence, ANFIS is a universal estimator. For using the ANFIS in a more efficient and optimal way, one can use the best parameters obtained by genetic algorithm. It has uses in intelligent situational aware energy management system [10].

2.1.2. Random Forest

A random forest is a machine learning technique that is used to solve regression and classification problems, which consists of many decision trees, generating a "forest," then trained by the algorithm. A Decision Tree is an intuitive technique that uses a tree-like model to draw decisions and their consequences. A simple example can be seen in Figure 2. A Random Forest builds several decision trees in randomly selected subspaces of the feature space [11] , i.e., a Random Forest contains several unrelated decision trees, each producing different outcomes. This method prevents individual error, since the output will not be dependent of a single tree, but all the trees' outputs. The outcome of a random forest can be the most common output of the different decision trees (see Figure 3).

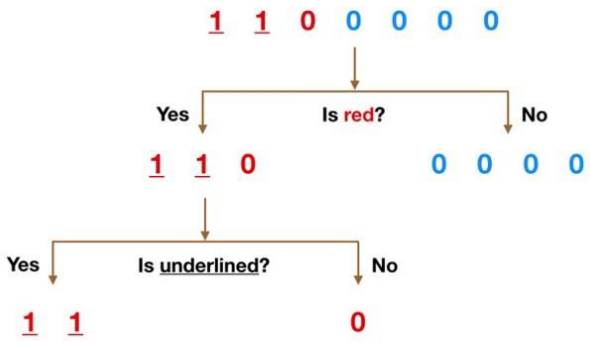

Figure 2 – Decision Tree example (image obtained from [12]).

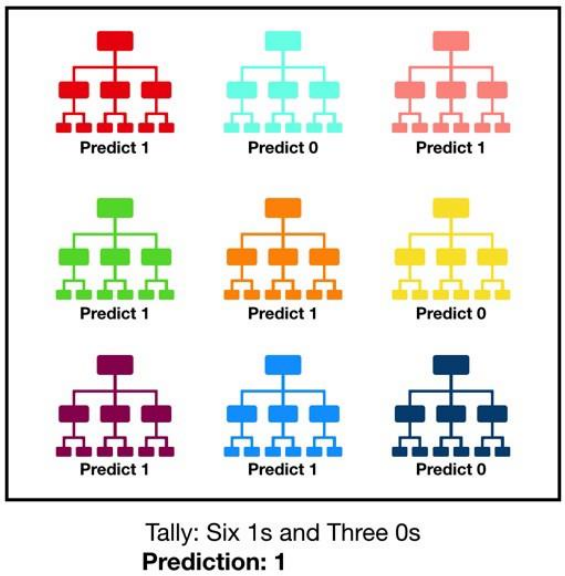

Figure 3 – Random Forest Classifier example (image obtained from [12]).

2.1.3. AdaBoost

AdaBoost, short for Adaptative Boosting, is a ML algorithm that works in an equivalent way to random forests, since it is also a tree-based technique. Decision trees in random forests are unrelated and do not need to have any predefined depth. In AdaBoost, however, decision trees usually have only one node with two leaves (*stump*). Stumps alone do not make accurate predictions, since they can only use one variable to make a classification decision, but they are represented with different values for their relevance, with some getting more say in the classification than others, therefore reducing wrong predictions. Furthermore, each stump considers the previous stump's errors into account, so, unlike random forests, decision trees are correlated and have a direct or indirect impact with each other [13].

2.1.4. Gradient Boost

Gradient boost is a machine learning algorithm based on the same technique of AdaBoost, albeit it contrasts with it in several ways. Gradient boosting starts by making a single leaf, instead of a stump or a tree, which contains an empirical initial guess for the value to predict. For example, to predict house prices, the initial guess could be the average of all the house prices present in the dataset. Then, gradient boost builds a tree based on the error of the previous tree, thus sharing similarities to AdaBoost, even though the tree can be larger than a stump. Also, similarly to AdaBoost, gradient boost scales the trees, although it scales all trees by the same amount [14].

The full algorithm for Friedman's Gradient Boost algorithm can be seen below in Algorithm 1.

Algorithm 1 – Friedman's Gradient Boost algorithm

Inputs:

- Input data $(x, y)_{i=1}^{N}$
- Number of iterations M
- Choice of the loss-function $\Psi(y, f)$
- Choice of the base-learner model $h(x, \theta)$

Algorithm:

1. Initialize $\widehat{f_0}$ with a constant
2. **For** $t = 1$ to M **do**
    a. Compute the negative gradient $g_t(x)$
    b. Fit a new base-learner function $h(x, \theta_t)$
    c. Find the best gradient descent step-size:

$$\rho_t = \arg\min_{\rho} \sum_{i=1}^{N} \Psi[y_i, \widehat{f_{t-1}}(x_i) + \rho h(x_i, \theta_t)]$$

    d. Update the function estimate

$$\widehat{f_t} \leftarrow \widehat{f_{t-1}} + \rho_t h(x, \theta_t)$$

3. **End for**

A. XGBoost

Extreme Gradient Boost (XGBoost) is a machine learning system that adapts Gradient Boost. The system is available as an open-source package [15] and gives state-of-the-art results on a wide range of problems, with its main factor being its scalability, running more than ten times faster than other popular solutions. XGBoost uses more advanced regularization than Gradient Boost, which improves model generalization capabilities. XGBoost delivers high performance, with a fast training time and parallelization possible across clusters [16].

2.1.5. Support Vector Machine

SVM is a powerful technique for general (nonlinear) classification, regression and outlier detection, originally developed for binary classification by [17], although the same method can be used with regression, with SVM's subclass Support Vector Regression (SVR). The method consists in separating data points into two classes, where each data point will be assigned one. The main goal is that the data are separated with the maximum *margin* between the classes' closest points, like we see in Figure 4, where the points lying in the boundaries are called *Support Vectors*, and the line separating those points is the *Separating Hyperplane*. If the margin is not maximized, there is a risk that future data will be incorrectly classified, even if the hyperplane rightly separates the present data. Furthermore, to deal with the problem of overlapping classes, where some points fall within the "wrong" side of the discriminant margin, the method weighs the points down and therefore reduces their influence (*soft margin*). Lastly, the problem of nonlinearity, where a linear separator cannot be found, is fixed by projecting the data points into an higher dimensional space where the data points become effectively linearly separable [18].

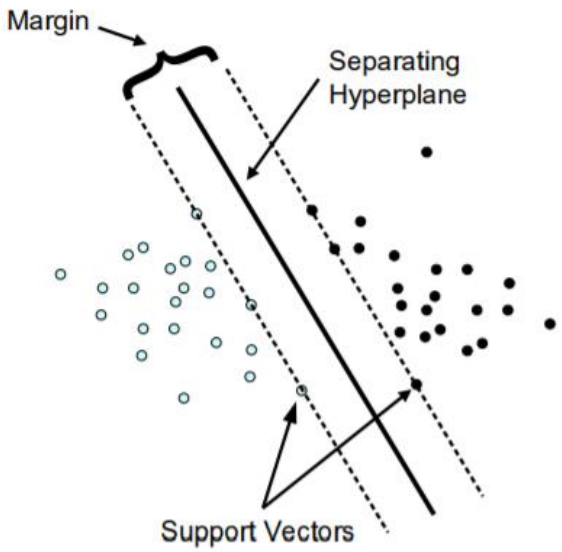

Figure 4 – SVM classification (linear separable case) (image obtained from [18]).

2.1.6. Performance Metrics

To find how well the algorithms perform, the methods need to be compared and evaluated. To do this, it is necessary to calculate their performance metrics. Some of the most common metrics are documented in this section, where Y are the observed values and P are the predicted.

A. Mean Error (ME)

Mean differences between the observed values and the predicted.

$$ME = \frac{1}{n}\sum_{t=1}^{n} Y_t - P_t$$

(1)

B. Mean Absolute Error (MAE)

Mean absolute difference between the observed values and the predicted.

$$MAE = \frac{1}{n}\sum_{t=1}^{n} |Y_t - P_t|$$

(2)

C. Mean Square Error (MSE)

Mean of the square of the differences between the observed values and the predicted.

$$MSE = \frac{1}{n}\sum_{t=1}^{n} (Y_t - P_t)^2$$

(3)

D. Root Mean Square Error (RMSE)

Square root of the Mean Square Error.

$$RMSE = \sqrt{MSE}$$

(4)

E. Mean Absolute Percentage Error (MAPE)

Mean absolute percentage of the difference between the observed values and the predicted.

$$MAPE = \frac{1}{n}\sum_{t=1}^{n} \left|\frac{Y_t - P_t}{Y_t}\right| * 100$$

(5)

F. Weighted Mean Absolute Percentage Error (wMAPE)

Sum of the absolute values of the difference between the observed values and the predicted, divided by the sum of the absolute observed values.

$$wMAPE = \frac{\sum_{t=1}^{n} |Y_t - P_t|}{\sum_{t=1}^{n} |Y_t|} * 100$$

(6)

G. Pearson Correlation Coefficient (r)

Covariance of the two variables divided by the product of their standard deviations.

$$r = \frac{1}{n-1}\sum_{i=1}^{n}\left(\frac{x_i - \bar{x}}{s_x}\right)\left(\frac{y_i - \bar{y}}{s_y}\right) \quad (7)$$

H. Coefficient of Determination (R^2)

Relation between the differences of the observed and predicted values with the differences between the observations and the average.

$$R^2 = 1 - \frac{\sum_{i=0}^{n_{samples}-1}(x - \hat{x})^2}{\sum_{i=0}^{n_{samples}-1}(x - \bar{x})^2} \quad (8)$$

2.2. Related Work

In this section, previous work on energy consumption with the use of ML techniques is explained, with the respective advantages and disadvantages of each related article.

The authors in [19] describe the Machine Learning (ML) techniques they used to improve self-consumption in Renewable Energy Communities. The authors propose data analytics modules which aim at helping the community members to schedule the usage of their resources (generation and consumption) in order to minimize their electricity bill. The model uses two state-of-the-art Neural Network models, the feedforward Multilayer Perceptron (MLP) and Long Short-Term Memory Network (LSTM), with its bidirectional variant BLSTM, and another two tree-based techniques, Random Forest (RF), and Gradient Boosting Decision Tree (GBDT). Finally, the authors use an original method that is simply the average of the output of the previous algorithms. The process can be visualized in Figure 5, where the data are fed to the four distinct algorithms (RF, GBDT, MLP, and BLSTM), each producing different outputs, computed into a final score.

While the users revealed to not change their consuming habits actively, they were still able to predict, with some accuracy, the wind power of the day-ahead, as we can see in Figure 6. The authors also concluded that their original algorithm, the average of the results of the other algorithms, produces the best results compared to the other algorithms individually. The Root Mean Squared Error (RMSE) of the ensemble method is the lowest, scoring 2327 kW, whereas RF, GBDT, MLP and Bi-LSTM score, respectively, 2347 kW, 2387 kW, 2338 kW and 2389 kW.

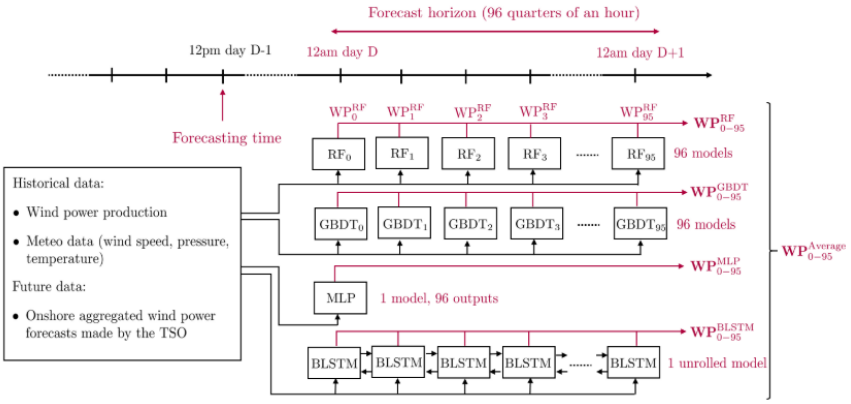

Figure 5 – Wind power forecasting model (image obtained from [19]).

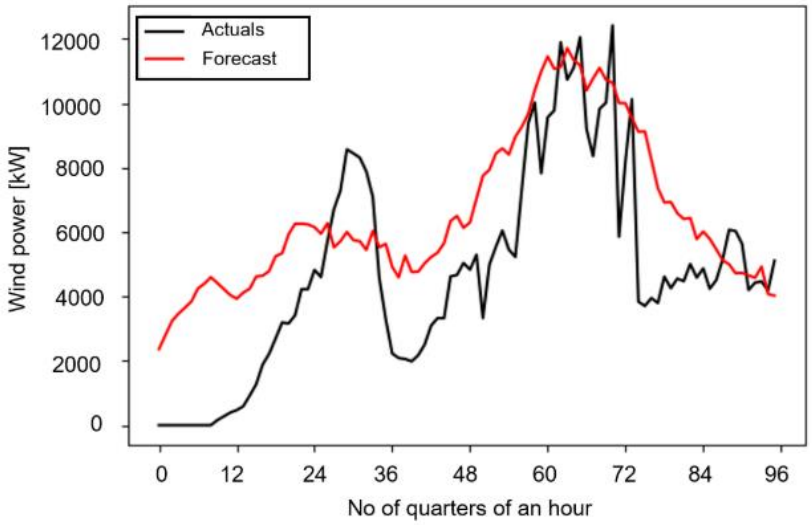

Figure 6 – Time series of wind power forecast and actuals for a day of the test set (image obtained from [19]).

The scope of the paper [20] is to present an approach for forecasting an energy cooperative's solar plant short term production by using its infrastructure and monitoring system. To do this, the authors propose and train four ML algorithms in an operational solar plant producing high accuracy short-term forecasts up to 6 hours. The algorithms are LSTM, Support Vector Regression (SVR), Multiple Linear Regression (MLR) and XGBoost and they were trained using previous performance and weather data. The results can be used for scheduling supply of the energy communities and set the base for more complex applications that require accurate short-term predictions, such as predictive maintenance. The overall methodology is represented in Figure 7, where the available data (temperature, humidity, etc.) are trained with the algorithms resulting in a production forecast.

The authors evaluated their algorithms by splitting the data into a train set (80% of the dataset) and test set (20%). All algorithms performed well, with XGBoost having the best results of the four of them. The results can be used for several actions regarding the PVs such as scheduling supply of the energy communities and set the base of more complex applications that require accurate short-term predictions, such as predictive maintenance or energy trading.

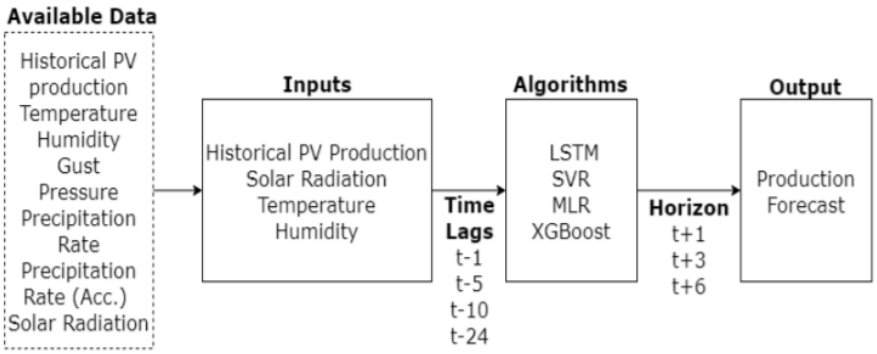

Figure 7 – Methodology schema (image obtained from [20]).

The authors in [21] compared the efficiency of three different techniques used to predict the daily power consumption for a local industrial region. First, the paper explains a probabilistic-based technique that relies on the Multiple Model Particle Filter (MMPF) to forecast the daily power consumption for a full upcoming year. The Particle Filter is a technique for estimating the state of a dynamic system. It could be described as a recursive Bayes filter that takes the current belief and updates it, depending on the motion information and the successive observations. Then, it compares it with two ANNs with one and two hidden layers, with design (5-5-1) and (5-5-5-1), respectively. Finally, a new adaptable ANN (with design (5-15-5-1), represented in Figure 8) is used with the Hebbian learning rule [22], an oriented training technique that relies on the historical features included inside the given dataset with the goal of adjusting the weights of the links between the input layer and the first hidden layer.

The performance metrics of the test results, as seen in Table 1, suggest that the third ANN, i.e., the ANN with design (5-15-5-1), performs better in terms of RMSE and MAPE, with 4.921 and 0.094, respectively. Therefore, the authors conclude that this method with this architecture provides the best results compared to the other studied methods.

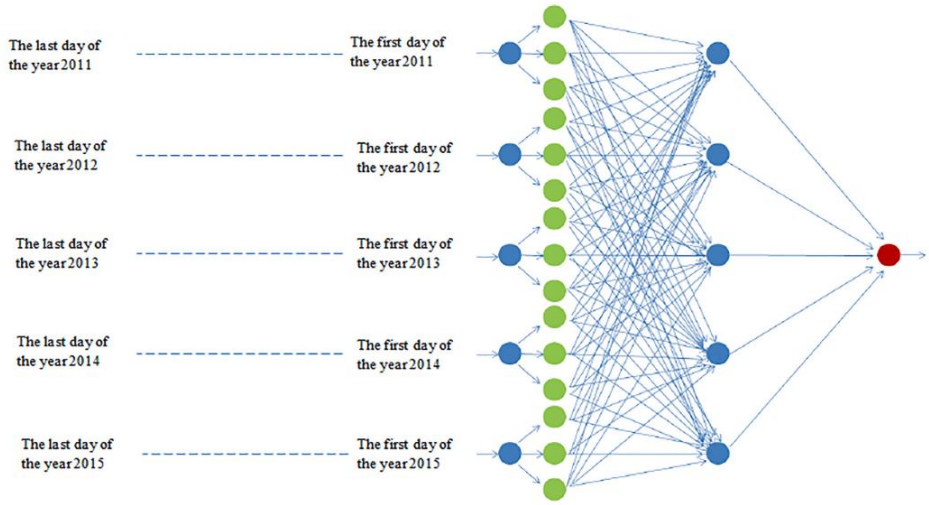

Figure 8 – The general structure of the suggested design (5-15-5-1) ANN (image obtained from [21]).

Table 1

	Linear	(5-5-1) ANN	(5-5-5-1) ANN	Probabilistic MMPF	(5-15-5-1) Improved ANN
RMSE (Kw)	18.19	5.714	5.613	5.631	4.921
MAPE	0.383	0.120	0.116	0.118	0.094
R	0.011	0.837	0.862	0.861	0.911

The authors of the paper [23] describe an energy management system that organizes the power flow in hybrid energy sources. In this work, a hybrid energy system (HES) composed of wind, gasoline and diesel generator is used as a case study to electrify a specific remote area. The work is divided into two stages. In the first stage, a historical demand side dataset is used to model and calculate the five criteria with the TOPSIS method [23]. The TOPSIS method is a multi-criteria decision-making method. These criteria are efficiency of the energy, CO2 emission, gasoline and diesel fuel prices, labour, and consumption of fuels. However, the TOPSIS model has the downside that there is no equation to calculate the weight values of the criteria. To calculate those values, other methods like Analytic Hierarchy Process (AHP) and Fuzzy Analytic Hierarchy Process (FAHP) were used, and later compared. In the second stage, machine learning algorithms were applied. Namely, random forest (RF) and light gradient boosted machine (LightGBM) algorithms are used to predict the combination of the energy sources as a way of validating the proposed work. This process is represented in Figure 9.

Evaluating the algorithms shows the superiority of the RF algorithm with an accuracy of 81.81% over LightGBM with accuracy of 68.6%. It also showed that the differences between AHP and FAHP are residual, and therefore, either of them can be used to show similar results.

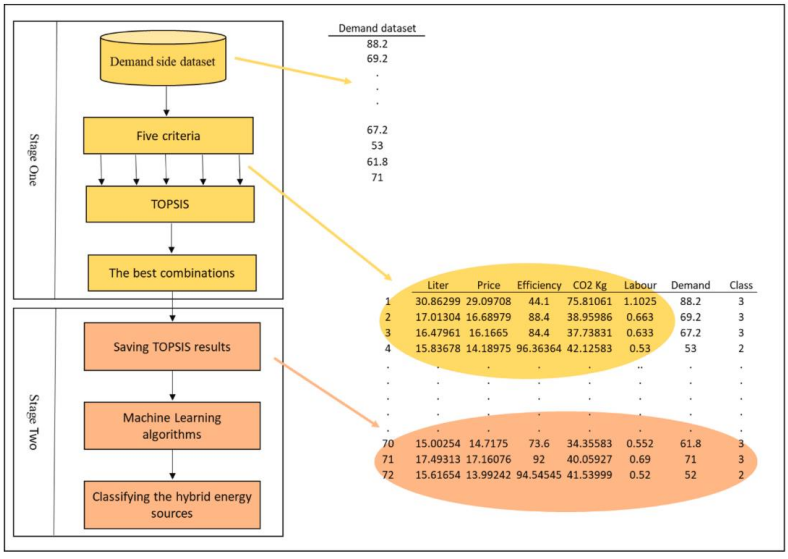

Figure 9 - The overall process of the model (image obtained from [23]).

In [24] it is propose a contextual learning approach for energy forecasting, which supports the decisions of Building Energy Management Systems (BEMS). The proposed forecasting approach includes a contextual dimension that identifies different observed contexts and clusters them

according to their similarity. The identification of such contexts is used by the learning process of artificial intelligence forecasting methods to select and adapt the most relevant data that is used in the training phase in each context. Forecasts for energy consumption, generation, temperature, brightness, and occupancy are used by the BEMS to provide recommendations to the consumers and to support automated control of building devices. Real consumption, generation and contextual data gathered from several sensors in a building are used to validate the results. The different AI techniques used for this paper are SVM, HyFIS [24], WM [25], and GFS.FR.MOGUL [26].

The main goal of this paper is to forecast energy consumption and generation of the building as well as occupancy, ideal temperature, and brightness of the rooms. The system uses, at first, clustering methods to recognize patterns in the consumption behaviour in the building. Then, based on these patterns, it selects the best data to train the forecasting models to predict future values. The tests show good enough results for the system to be trusted to provide recommendations related to consumption and with smaller error metrics than other ML methods (Table 2). The system uses a cloud-based strategy to create and use the ML models. This approach makes it possible to use this system in any other building without concerns about availability of required resources for ML.

Table 2 – Results comparison with previous studies.

Model		**MAPE**
Deep Learning	15 min. ahead	36.20%
	60 min. ahead	64.70%
CNN-LSTM w/ no clustering		44.76%
CNN-LSTM w/ clustering		40.38%
LSTM w/ clustering		44.68%
SVM	15 min. ahead	**10.70%**
	60 min. ahead	**12.44%**
HyFIS	15 min. ahead	**13.00%**
	60 min. ahead	**10.52%**
WM	15 min. ahead	**13.10%**
	60 min. ahead	**10.52%**
GFS.FR.MOGUL	15 min. ahead	**15.93%**
	60 min. ahead	**15.73%**

The paper [27] presents a study using the genetic fuzzy system for fuzzy rule learning based on the MOGUL methodology (GFS.FR.MOGUL) in order to have a better profile of the electricity consumption of the following hours. The GFS.FR.MOGUL is a forecasting method that implements a genetic algorithm determining the structure of the fuzzy IF-THEN rules and the membership function parameters. The proposed approach uses the electricity consumption of the past hours to forecast the consumption value for the following hours. Results from the study are compared to those of previous approaches, namely two fuzzy based systems and several different approaches based on artificial neural networks. The studied method uses the electricity consumption from the 10 previous hours to the moment that is to be forecasted to be trained and forecast the end results.

The comparison of the achieved results with those achieved by the previous approaches shows that this approach can calculate a more reliable value for the electricity consumption in the

following hours, as it is able to achieve lower forecasting errors (see Figure 10), and less standard deviation of the forecasting error results (see Table 3).

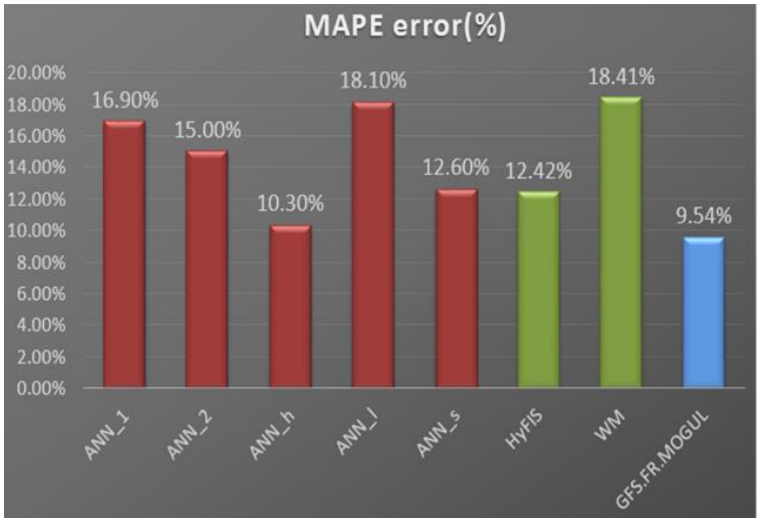

Figure 10 – Average forecasting errors of the HyFIS, WM, GFS.FR.MOGUL and several ANN approaches (image obtained from [27]).

Table 3 – Standard deviation between HyFIS, WM and GFS.FR.MOGUL

Forecast Method	**Forecast Error Standard Deviation (%)**
HyFIS	13.90
WM	17.32
GFS.FR.MOGUL	5.95

The aim of the present study [28] was to investigate the potential of Fuzzy Interference System (FIS) to estimate the energy consumption of various types of residential buildings in the northern part of Cyprus. The climate zone in which the house was constructed, floor area of house, the year of construction, the type of house and the number of occupants per house was considered as input parameters and energy consumption per floor area as the output.

The energy consumption values estimated by the proposed FIS model were in close agreement with their actual counterparts reported previously, i.e., R^2 value was as 0.9884 and 0.8851 for training and testing phase, respectively, and with a RMSE of 6.6115 and 20.3347. The training and test results are compared in Figure 11, demonstrating that the FIS model can be a powerful tool for predicting energy consumption of residential buildings.

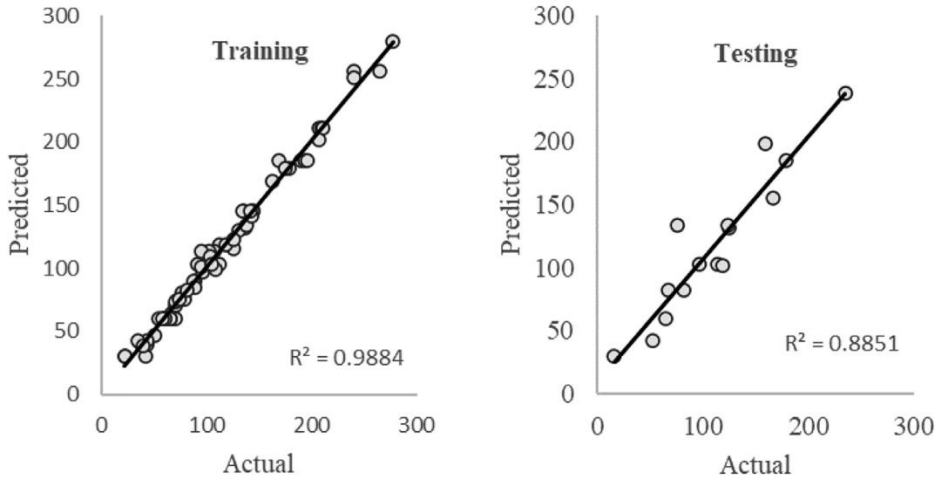

Figure 11 – Comparison of predicted and actual counterparts for training and testing data sets (image obtained from [28]).

In this study [29], the adaptive neuro-fuzzy inference system (ANFIS) is designed and adapted to estimate the energy consumption of buildings in relation to three parameter groups: building structure, insulation value and insulation thickness. There are three membership functions on each input, one ANFIS network for building heating, and one ANFIS network for building cooling. For this study, bell-shaped membership functions were chosen with maximum of 1 and minimum of 0.

The performance statistics for heating (Table 4) and cooling prediction models (Table 5), which are compared to those of the ANN and genetic programming (GP), suggest that the prediction results agree with the actual values of building energy consumption. The ANFIS predictions were superior to ANN and GP results, performing better in RMSE, r and R^2.

Table 4 – Performance statistics of heating prediction models

	RMSE	**R^2**	**r**
ANFIS	98	0.9959	0.9979
ANN	300	0.9410	0.9422
GP	279	0.9477	0.9485

Table 5 – Performance statistics of cooling prediction models

	RMSE	**R^2**	**r**
ANFIS	85	0.9567	0.9780
ANN	150	0.9463	0.9622
GP	140	0.9342	0.9585

This study [30] explores beyond the various studies which have developed diverse models for predicting building energy consumption focused on the current building stock but have not considered energy efficiency at the design stage. The authors aim to explore the application of hyper parameter tuning and feature selection methods in developing a design stage ML energy predictive

model. To do this, nine machine learning classification-based algorithms were compared for energy performance assessment at the design stage of residential buildings. Those algorithms are SVM, Gradient Boosting (GB), RF, Decision Tree (DT), K Nearest Neighbour (KNN), Extra Trees (ET), Ada boost, Gaussian Process (GP), and MLP. Additionally, feature selection and hyper parameter tunning were implemented. The overall methodology, which contains five steps: **Data collection**, **Data pre-processing**, **Feature Selection**, **Model development (training)**, and **Model evaluation (testing)**, is represented in Figure 12.

The results show that it is possible to develop a high performing ML model for building energy use prediction at the design stage. In Table 6, it is possible to see the performance results, where Gradient Boosting (GB) outperformed the other models with an accuracy of 0.67 for predicting building energy performance. Although GB has not received much attention in the field of energy performance prediction, the performance level of the ML algorithm proffers GB as an effective predictive model in the field of energy prediction.

Table 6 – Performance results after hyper parameter tunning.

Model	Training time	Accuracy	F1
Random Forest	9m3s	0.66	0.64
Gradient Boosting	5m15s	**0.67**	**0.65**
Extra Trees	14s	0.64	0.63
Decision Tree	1m9s	0.63	0.60
K Nearest Neighbours	**6s**	0.64	0.59
Support Vector Machines	14m31s	0.66	0.62
Gaussian Process	16m45s	0.65	0.59
Multi-Layer Perceptron	6m39s	0.63	0.58
Ada Boost	12m4s	0.65	0.64

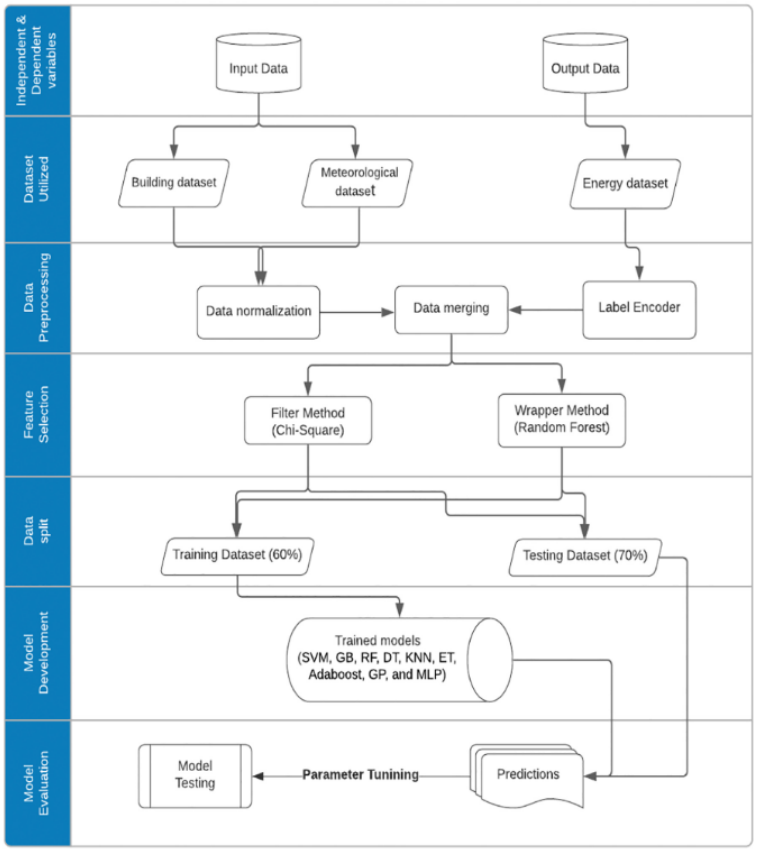

Figure 12 – Flowchart diagram of the research methodology (image obtained from [30]).

This investigation [31] presents a review of ML techniques for forecasting energy consumption time series using actual data, collected from a smart grid installed in an experimental building and used to evaluate the efficacy and effectiveness of statistical and ML techniques. The authors studied three categories of ML models: single, where only one technique is used to predict the output; ensemble, where multiple prediction models' outputs are integrated into one; and a hybrid model, which combines two or more ML techniques and are more robust than the others as they frequently exhibit the advantages of the incorporated techniques and provide improved forecasting accuracy. For the single models, several methods were investigated, such as ANNs, C&R Trees (Classification and Regression Tree), SVR, and Linear Regression. For ensemble models, Voting and Bagging ensemble models were used. Voting is an ensemble method where each model's output is combined in some way, such as taking the mean or the mode of the predictions, allowing each model to *vote* on what the outcome should be. Bagging involves having each model in the ensemble vote with equal weight [32]. Finally, for hybrid models, SARIMA-MetaFA-LSSVR and SARIMA- PSO-LSSVR were the methods chosen for further study.

After focusing on several methods, both single, ensemble, and hybrid, the models with the best performance measures were compared, showing a clear advantage for hybrid models, as it can be seen in Table 7, where the best models in each scenario are compared. The hybrid model is more accurate, since both the accuracy of prediction and the suitability for use of these models are considered to support users in planning energy management.

Table 7 – Performance measures for the best single, ensemble, and hybrid models

Scenario	Best Model	Performance Measures					
		R	RMSE	MAE	MAPE (%)	SI	Improvement
Single	**ANN**	0.556	**0.092**	0.057	36.834	0.70	
Ensemble	**Bagging (ANN)**	0.607	0.094	0.049	42.31	0.64	9%
Hybrid	**SARIMA-MetaFA-LSSVR**	**0.796**	0.164	**0.028**	**15.657**	**0.25**	64%

This paper [33] addresses the problem of energy sharing in Zero Energy Communities (ZEC), which is a collection of Zero Energy Buildings (ZEB). ZEBs are buildings designed to have their net energy usage over a one-year period equal to zero, i.e., its energy use is not larger than its overall renewables generation. The communities are represented as a multi-agent environment, where each building is an agent that learns optimal behaviour independently and is entirely responsible for making energy transactions on behalf of that building. The authors propose a Deep Reinforcement Learning (DRL) based algorithm, called DQN to approximate Q-values. A Q-value is the expected value of taking an action in a particular state while considering the expected long-term reward of taking that action in that state. To evaluate their approach, two sets of weather conditions, Winter, and Summer, were used. Furthermore, three different scenarios were evaluated representing different community configurations and different scales (i.e., different number of households). The three tested scenarios are composed of three, four, and ten houses, respectively, with varying generation capacity and initial battery charge, where in the second scenario one of the houses has zero generation capacity. An example of the behaviour of the agents is documented in Figure 13, where it can be seen the energy borrowed from the supply grid in the first scenario, during Winter.

Results indicate that, with time, buildings learn to collaborate and learn a policy comparable to the optimal policy, in which turn improves the ZEC's energy from their neighbours rather than from the supply grid. Specifically, an improvement of 40 kWh with three houses during winter and over 60kWh with four houses during summer over three days, when compared to a no-energy-sharing strategy.

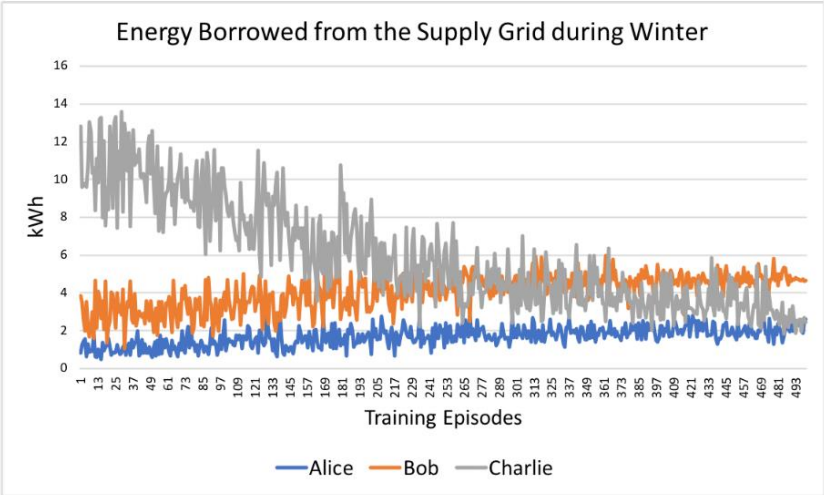

Figure 13 – Energy borrowed from the Supply Grid during Winter in Scenario 1 (image obtained from [33]).

This study [34] aims to compare prediction capabilities of five different intelligent system techniques by forecasting electricity consumption of an administration building. These five techniques are Multiple Regression (MR), Genetic Programming (GP), ANN, Deep Neural Network (DNN), and SVM. The predicted electricity consumption of all models is compared with actual consumption of another year. The dataset used to make these predictions contains data from the year 2007-2011 and is composed of the following parameters: the daily electricity usage (Wh/m^2); the daily mean surrounding temperature (K); the daily mean humidity ($\%$); and the weekday index ($0/1$). The dataset is further split onto a training set, which contains the data from year 2007-2010, and a testing set, with the year 2011.

In Figure 14, it can be seen a comparison between the spread of predicted energy consumption data by different methods and the real data in the form of box plots. None of the methods correctly predict the minimum or maximum value, although the prediction of some methods was closer to actual than others. Looking at the figure it can be said that results of ANN and GP for median, second and third quartile are close to the actual values and these two methods seem to perform better than others in overall prediction. Furthermore, results demonstrate that ANN performs better than all other four techniques with a MAPE of 6% whereas MR, GP, SVM and DNN have MAPE of 8.5%, 8.7%, 9% and 11%, respectively.

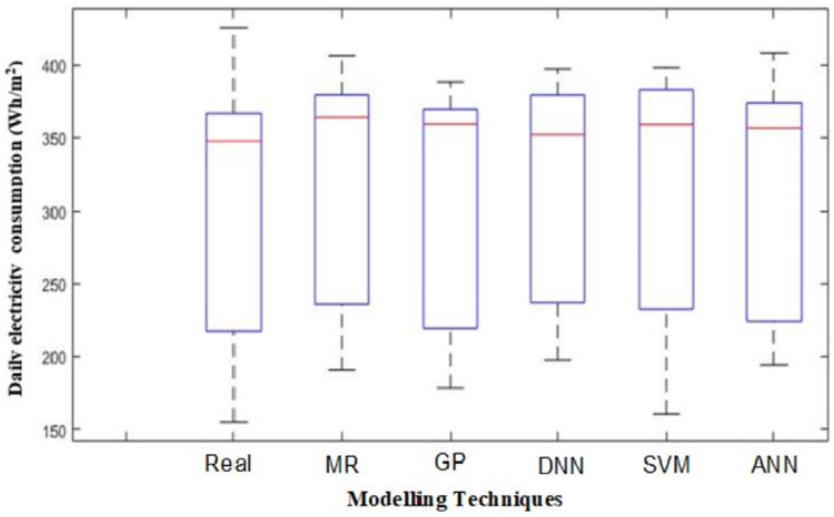

Figure 14 – Comparison of outputs of different modelling techniques (image obtained from [34]).

In [35], it is studied the prediction of future energy usage using the XGBoost algorithm. First, the authors used this algorithm to perform a technique called "feature extraction". Feature extraction or feature selection is the study of algorithms for reducing dimensionality of data to improve machine learning performance [36]. The process of this phase consisted in extracting the load of the past week, with a granularity of 30 minutes, therefore having 336 variables (48x7), and using XGBoost for feature importance calculation. The authors removed the features bellow a threshold, which was set by repeating the experiment multiple times. In the end, it was considered the 35-40 features with highest importance value.

The following phase consisted in splitting the data into a train dataset with 75% of the data and a test dataset with 25%. The model was trained with the train dataset and evaluated with the test dataset. These processes are documented in Figure 15. The XGBoost load forecasting model resulted in a MAPE of 10.08%, 97.21% accuracy and 88.90% MAE. We can see that the results are good, however, as we can see in Figure 16, the high load instances are not being followed by the XGBoost algorithm.

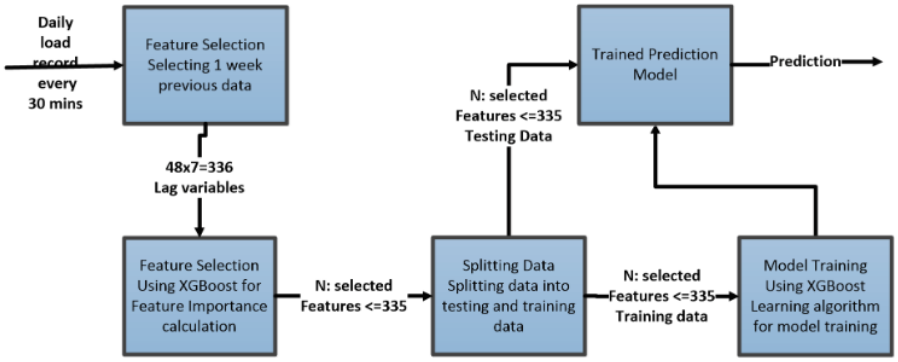

Figure 15 – System model (image obtained from [35]).

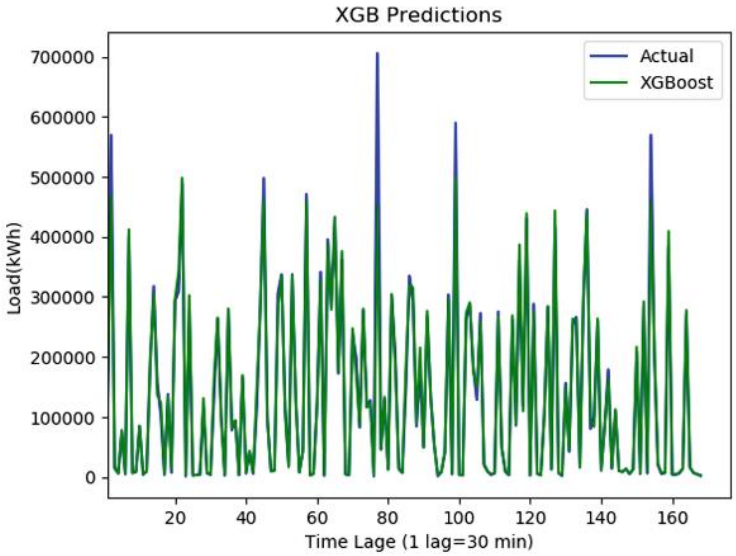

Figure 16 – XGB Predictions (image obtained from [35]).

2.2.1. Discussion

Table 8 summarizes the results of the different techniques for each analyzed paper. In this sense, the method that produced the best outcome is identified.

Table 8 – Summary of the techniques used in related work.

Source	Application	Used Techniques	Best Performance
Z. de Grève *et al.* [19]	Consumption Forecast in Communities	RF, GBDT, MLP, BLSTM, ensemble (average of all outputs)	ensemble
N. Dimitropoulos *et al.* [20]	Production Forecast in communities	LSTM, SVR, MLR, XGBoost	XGBoost
A. Baba [21]	Consumption Forecast of a local industrial region	MMPF, ANN (5-5-1, 5-5-5-1, 5-15-5-1)	ANN (5-15-5-1)
H. Musbah, G. Ali, H. H. Aly, and T. A. Little [23]	Energy management	RF, LightGBM	RF
A. Jozi, T. Pinto, and Z. Vale [24]	Consumption and generation forecast with contextual data	SVM, HyFIS, WM, GFS.FR.MOGUL, compared to Deep	SVM, HyFIS, WM, GFS.FR.MOGUL

		Learning, CNN-LSTM and LSTM	
A. Jozi, T. Pinto, I. Praca, F. Silva, B. Teixeira, and Z. Vale [27]	Consumption forecast	GFS.FR.MOGUL, ANNs, HyFIS, WM	GFS.FR.MOGUL
F. Al-Shanableh and A. Evcil [28]	Consumption forecast	FIS	-
S. Naji *et al.* [29]	Consumption forecast of buildings	ANFIS, ANN, GP	ANFIS
R. Olu-Ajayi, H. Alaka, I. Sulaimon, F. Sunmola, and S. Ajayi [30]	Building energy consumption forecast	SVM, GB, RF, DT, KNN, ET, AdaBoost, GP, MLP	GB
J. S. Chou and D. S. Tran [31]	Consumption forecasting	Different types of single, ensemble and hybrid models	Hybrid models
A. Prasad and I. Dusparic [33]	Energy communities' management	DRL	-
K. P. Amber, R. Ahmad, M. W. Aslam, A. Kousar, M. Usman, and M. S. Khan [34]	Electricity consumption forecast of an administrative building	MR, GP, ANN, DNN, SVM	ANN
N. Javaid, M. Nauman Javid Ghuman, Z. Ali Khan, R. Abid Abbasi, and S. Ur Rehman[35]	Electricity consumption forecast	XGBoost	-

In this chapter, the topic of ML was approached, explaining what ML is, as well as its most common algorithms. Then, existing literature was reviewed. Several papers were summarized, pointing their goals, the algorithms used, and the results retrieved by them. The papers also approached the problem using several different methods, each with its specific advantage. Single methods, such as Random Forest or Gradient Boost, were often present in the different articles, but also ensemble methods and hybrid models were used. It does not seem, however, to exist a

unique preferred method, with results varying from paper to paper. Nonetheless, GB, ANN, and XGBoost could be selected as the most promising, providing satisfactory results when evaluated with other algorithms. Most articles focus on the energy consumption forecast in residential buildings, either individual or in communities. In contrast, this project will focus on energy communities without discrimination of their type, they can be individual residential households, stores, business buildings, etc. It is expected that consumption patterns vary depending on the type of building and the different use people give to them, such as the energy consumed in weekends and weekdays can be different in stores closed on weekends and individual households, where people will likely consume more energy outside business hours.

The next line of work will be data collection and analysis. The data should be extensive, with diverse types of buildings, and with other possible metrics, such as weather information. Finally, the system will be developed, extensively experimenting various algorithms and methods. The process is documented in the thesis, which will provide the necessary conclusions of the project.

3. Architecture

In this chapter, an architecture for the system is proposed and will be implemented and described in the next chapters. The architecture is documented in Figure 17. The system contains, at the center, a decision algorithm will make decisions on the charging and discharging of the batteries. To do this, the algorithm will be fed with a forecast of the production of solar photovoltaic panels (PV). This forecast is done with the use of the prediction of the solar irradiance, publicly available, and the area and efficiency of the solar panels. This results in an analytical model to predict the energy production of a REC. The algorithm will also consider the market electricity prices, as well as the state-of-charge of the batteries. Finally, the energy consumption forecast, using AI//ML models, is used, with the goal of improving the consumption of green energy, or reducing the overall price.

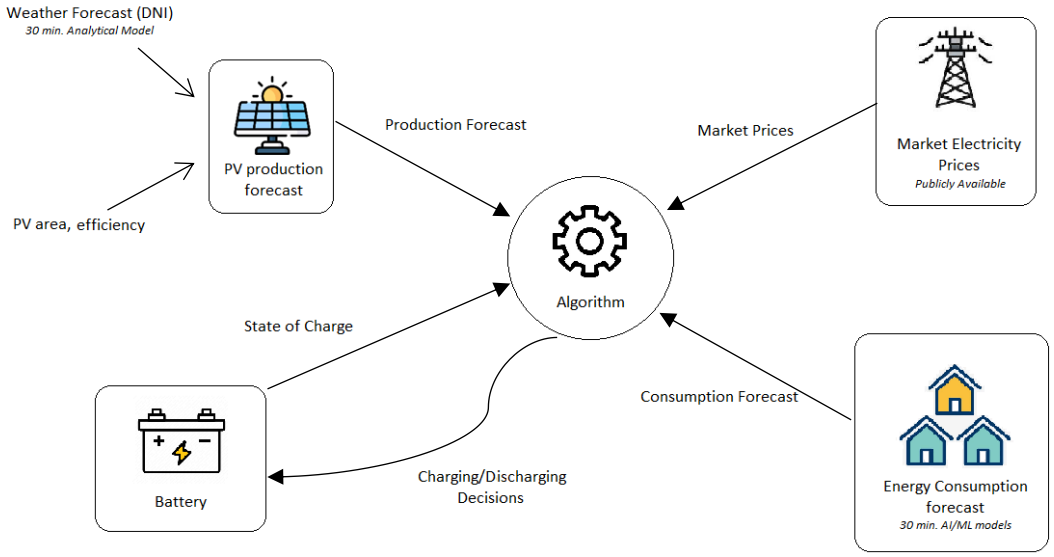

Figure 17 – Project Architecture.

The **algorithm**, at the center of the architecture, makes the decisions about the energy management of the community. Particularly, it manages the REC, providing the details of what should the community do, regarding the energy produced, stored, and the energy that must be fetched from the grid. It executes every half-hour and incorporates the outputs of the other components to make these decisions.

The **PV production forecast** component uses an analytical model to make a forecast of the energy produced by the solar PV's. The forecast can be a short-term forecast or the daily forecast. Regardless, it needs to know the Direct Normal Irradiance (DNI) or solar irradiance to make a high-quality prediction.

The **electricity market prices** are used to evaluate the algorithm. With the goal of reducing the value of this component, the calculation of the bill of the REC provides insight on whether the algorithm is viable for the users or if it is too expensive.

The **consumption forecast** is provided by a ML algorithm, which supplies the decision algorithm with a forecast of the energy consumption of the community. The decision algorithm will account for this prediction and make the management decisions accordingly.

The **battery** is the storage component. For a user to be a prosumer, it needs capacity of production and storage of energy. So, this component will provide the algorithm with its state-of-charge, i.e., the current charge of the batteries. The algorithm will also connect with this component by informing it of the energy the batteries need to charge or discharge.

Later, on this document, a detailed explanation of the implementation and validation of each of the individual components is provided.

4. Implementation

This chapter provides a detailed explanation of the implementation of the five different components of the algorithm. Specifically, it details the steps necessary to implement the components, as well as the reasoning behind those steps.

4.1. Energy Consumption Forecast

This section documents the approach followed to apply the Machine Learning algorithms to predict the energy consumption of a REC.

4.1.1. About the data

To train the Machine Learning algorithms, it is necessary to obtain data from real life energy consumption sources, as well as weather information. The data was obtained thanks to the *Bath: Hacked* project, which contains several datasets detailing aspects of life in Bath & North East Somerset (BANES), UK, including a dataset on the electricity energy usage in Council buildings [37]. Furthermore, weather information such as air temperature and relative humidity was acquired using the Solcast API [38].

A. BANES dataset

The BANES dataset contains energy consumption data in kWh of seventy-two council buildings, such as schools and libraries, with the help of smart meters from 1 October 2006 to 8 February 2020, and a granularity of 30 minutes. The data used, however, was only after 2007, since the Solcast API did not accommodated data from 2006. There were further alterations to the data, which contained a small number of negative values. These readings were eliminated, considering that it would not make much sense to consume negative values of energy, and, instead, they were replaced by the interpolation of the previous and following readings. Finally, duplicate values were deleted.

The dataset has its columns represented in Table 9. The columns *id*, *totalunits*, *units*, *mpan*, and *msid* were not necessary and were, therefore, dropped. The *date* column contains the information regarding the day of the readings, the *location* column the name of the building, and the following columns the hour and minute of the energy consumption values with a granularity of 30 minutes.

Table 9 – Columns of the dataset.

date	**location**	**00:30**	**01:00**	**…**	**23:00**	**23:30**	**24:00**

Since it is not very flexible to work with the hours extended in the columns horizontally, the values were joined into a single column *energy*. The *date* column will no longer represent just the day, but also represent the hour of the readings, being renamed *time* in the process. A sample of the final dataset is documented in Table 10.

Table 10 – Sample of the final dataset.

time	**location**	**energy**
2008-06-14 00:30:00+00:00	## OLD Paulton Library Electricity Supply 1	0.08
2008-06-14 01:00:00+00:00	## OLD Paulton Library Electricity Supply 1	0.07

2008-06-14 01:30:00+00:00	## OLD Paulton Library Electricity Supply 1	0.08
2008-06-14 02:00:00+00:00	## OLD Paulton Library Electricity Supply 1	0.07
2008-06-14 02:30:00+00:00	## OLD Paulton Library Electricity Supply 1	0.08

Below, in Figure 18, the number of readings each building had present in the dataset is shown. Figure 19 documents the mean energy each building consumed at every half hour interval. It can be seen that the data is highly unbalanced, with some locations averaging over 30 kWh and some close to 0 kWh. This could, potentially, bring some "reality" to the project since the model that will be trained needs to be used in communities with quite different consumption patterns.

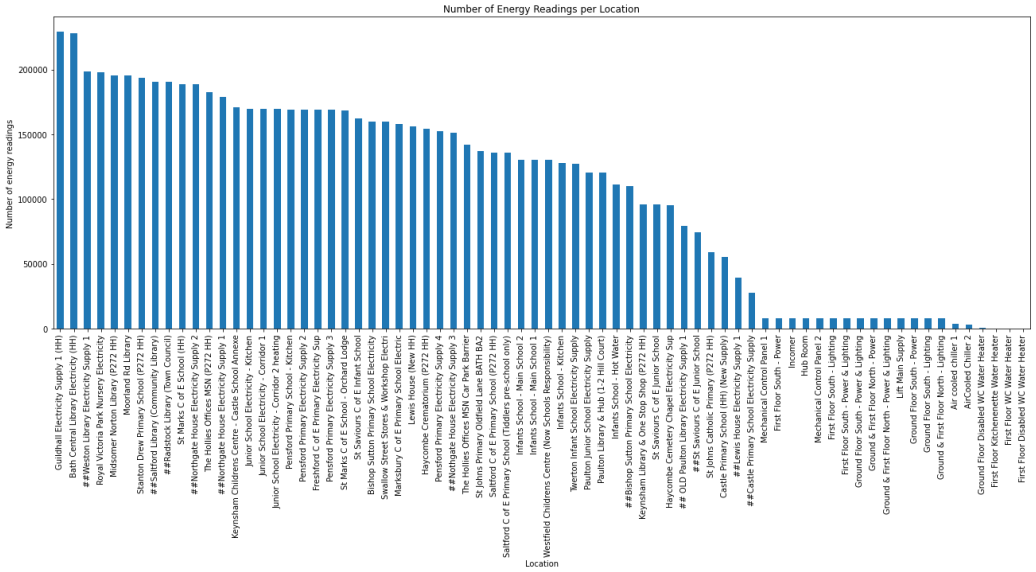

Figure 18 – Number of readings per council building in the dataset.

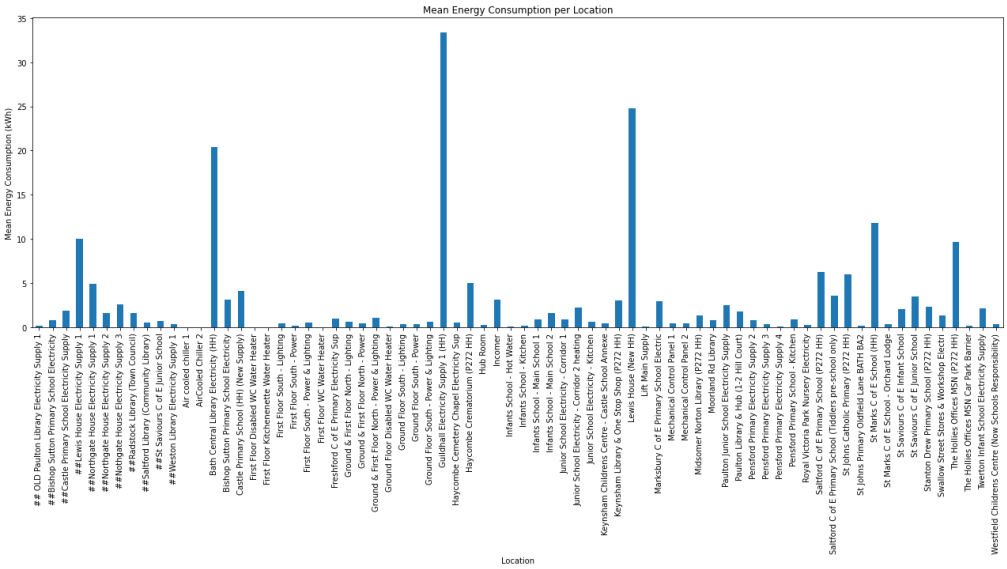

Figure 19 – Mean energy consumption at every half hour per building.

B. Solcast API

The Solcast API is more extensive than the BANES dataset, with data from several spots around the world. Since the energy readings we have available are from Bath, UK, a request was made for a time series dataset for the coordinates 51.378102, -2.359683, the latitude and longitude, respectively, of the city of Bath. The retrieved dataset accommodated data from 1 January 2007 to 8 February 2020, where it was requested a granularity of 30 minutes to match the BANES dataset. The specified variables contained in the dataset were the Air Temperature, Relative Humidity, and DNI, the solar irradiation. Note that the DNI will not be used in this phase, but it was requested, nonetheless, because it will be important to estimate the energy production of solar panels in the next chapter.

The air temperature and relative humidity are documented in Figure 20 and Figure 21, respectively. There were no alterations to the data.

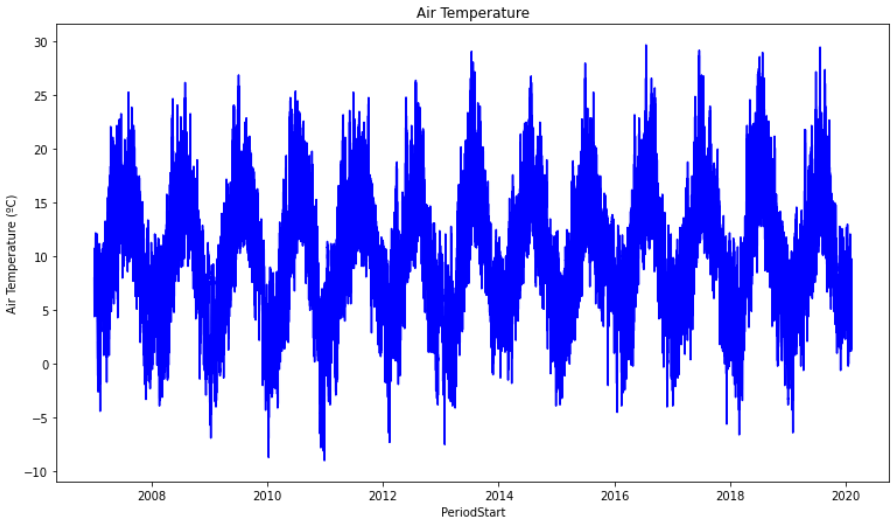

Figure 20 – Air Temperature in Bath, UK.

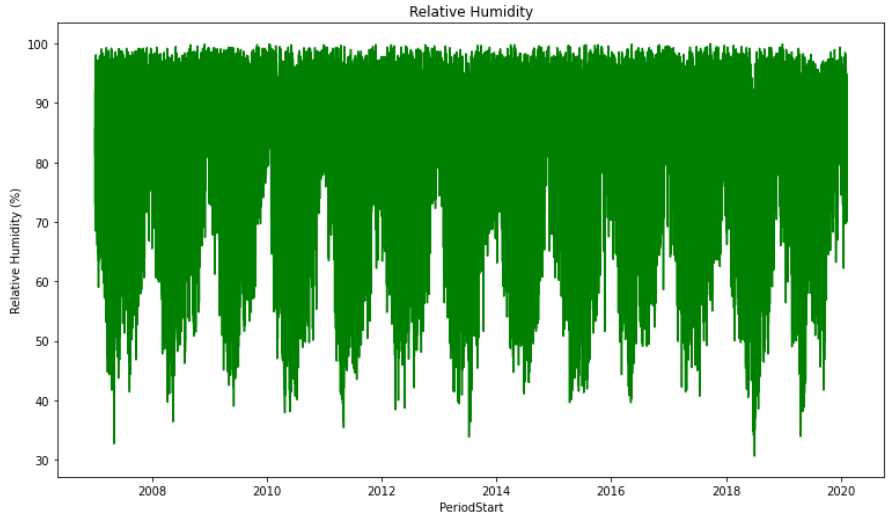

Figure 21 – Relative Humidity in Bath, UK.

C. Merge datasets

Finally, the Solcast dataset was merged with the BANES dataset and the relative humidity column was divided by one hundred to be in intervals of 0-1, instead of 0-100. A sample of the result is represented below.

Table 11 – Sample of the merged dataset.

time	location	energy	AirTemp	RelativeHumidity	Dni
2007-01-01 00:30:00+00:00	Bath Central Library Electricity (HH)	18.0	8.0	0.814	0.0
2007-01-01 00:30:00+00:00	Guildhall Electricity Supply 1 (HH)	33.0	8.0	0.814	0.0
2007-01-02 00:30:00+00:00	Bath Central Library Electricity (HH)	17.8	5.1	0.800	0.0
2007-01-02 00:30:00+00:00	Guildhall Electricity Supply 1 (HH)	29.8	5.1	0.800	0.0
2007-01-03 00:30:00+00:00	Bath Central Library Electricity (HH)	17.9	6.4	0.856	0.0

4.1.2. Machine Learning

This section explains the process and results of the training and testing of the ML algorithms to forecast the electricity consumption of buildings using the datasets explained above.

In the state-of-the-art chapter, we looked at several algorithms that could be used to train and evaluate our model to predict the consumption of electricity. Although there was no consensus between the papers for a single algorithm, XGBoost, Gradient Boost, and ANN were often present, providing satisfactory results.

A. Algorithms Comparison

To pick a single algorithm to continue the study of Energy Consumption models, the already mentioned methods, XGBoost, Gradient Boost, and ANN, will be trained and tested under similar circumstances. The proposed models, which will be trained with these different algorithms, will predict the short-term load energy consumption at every location, i.e., the *energy* column is the output parameter that will be predicted by the models. The input parameters, or features, the algorithms will use are the *air temperature*, *relative humidity*, and each of the energy consumption readings of the same building of the last 24 hours. At the end, there will be a column with the output values, *energy*, and fifty other columns with the two weather features already acknowledged, and the forty-eight columns with the energy reading of the building at the previous 30 minutes, previous hour, previous 90 minutes, etc., until the previous 24 hours of the energy reading to predict. Finally, the dataset is split into train and test datasets, with 80% for the train, and 20% for the test sets.

- **ANN** – The ANN model has four layers. The input layer has a size of fifty, which is equal to the number of features. There are also two hidden layers, each with a size of ten and activated with the *reLU* function. The model is compiled with the *Adam* optimizer and uses MSE for the loss metric. Finally, it is trained with twenty epochs, and a batch size of one thousand.
- **GBDT** – The GBDT algorithm is used in this context with one hundred estimators and 0.1 as its learning rate, as it is described in [30].

- **XGBoost** – To train the XGBoost model, the histogram as the tree method was used while running on GPU. Running the algorithm on GPU increases the performance on the training time, while making negligible changes on the other metrics. The XGBoost also has six as its maximum depth, 0.3 as its learning rate and one hundred estimators. The default values are used in the other hyperparameters.

The results, which are analysed in greater detail further on this document, show that XGBoost performed better in all error metrics – MSE, RMSE, MAE, wMAPE, and R^2, compared to the other algorithms, and, therefore, will be the focus of the next section.

B. Feature Classification

Following the results of the previous subsection, which tests different ML algorithms, XGBoost was picked for further study on diverse ways to forecast the energy consumption of a REC. In this section, using a technique called Feature Classification, a smaller set of features will be selected according to their importance on the training of an XGBoost model, adapting the steps in [35]. To do this, instead of using the previous 24 hours like in the previous section, the entire previous week is used, resulting in 336 features, plus the *air temperature* and *relative humidity*, adding up to 338 input parameters. The high number of features leads to a heavy amount of memory usage, and, to tackle this problem, only a year in the data was used for training the model.

- **Perform feature classification**

The XGBoost algorithm will have as hyperparameters one thousand estimators, a maximum depth of five, and a learning rate of 0.05, and will be trained with the entire data from the year 2019, taking 259.9533 seconds to perform. The features have the name $energy\_lag\_i$, where i is equal to the i^{th} previous half hour readings.

The input parameter that has the largest impact in the prediction is the direct previous energy reading, with over 80% of importance. Further details on the results are explained in the next chapter.

- **Train models using the features' classification**

With the results of the features' classification, the forty most prominent features will be used, as well as the features with an importance classification above 5%, i.e., the six most important input parameters. The training phase will be performed similarly as before, with a separation of the data into training and test datasets with a ratio of 80% and 20%, respectively. Furthermore, a different approach is also followed where different *n* buildings are left out of the training phase and used to evaluate the model. This method is followed because, in a real-life scenario, the model that will be used on communities is not trained with their data, but the data of other communities. To find out the minimum acceptable number of buildings that can be left out of training, different models are trained leaving 5, 10, and 15 houses, chosen at random, using only the features with an importance classification higher than 0.5%. The experiment is repeated five times and the results show that, when only five houses are used, the error metrics vary too much. With ten or fifteen houses, however, the performance metrics of all experiments record similar values. Therefore, for this approach, a minimum number of ten buildings must be used as the test dataset.

4.2. Energy Production Forecast

To predict the production of energy in a community with PV panels, we used an analytical model. The model is based on the following formula [39]:

$$E(t) = A * r * g(t) * \rho \quad (9)$$

Where:

- $E(t)$ is the energy produced in kWh at time t.
- A is the area of the solar panels in m^2.
- r is the yield of the solar panel given by the ratio: electrical power (in kWp) of a solar panel divided by the area of a PV panel.
- $g(t)$ is the solar irradiance in kWh/m^2 at time t.
- ρ is the overall performance, with values ranging between 0 and 1.

With these metrics, we can predict with relative accuracy the energy produced in a community if we can correctly predict these factors, particularly the solar irradiance. We can obtain the solar irradiance with the already mentioned Solcast API, whose values are present in the DNI (Direct Normal Irradiance) column, the other values are assigned following three scenarios of production: high production; low production – which is a tenth of the high production; and average production – average of the high production and low production. The high production scenario has a total area of **28.8 m^2** (eighteen panels with 1.6m^2 each) and a yield of the solar panel of **0.16**. In standard conditions, i.e., with a performance of one and a solar irradiance of 1kWh/m^2, the energy produced at time *t* is equal to around **4.6080 kWh**, which means that in the low production scenario, it produces around **0.4608 kWh** and in the average scenario **2.5344 kWh**. Moreover, in both scenarios, the PV performance is equal to **0.95**, reducing these figures. It is considered, for simplicity, that the weather forecast, as well as this model's output, have no associated error.

4.3. Market Prices

To determine the price of electricity that the consumers must pay, we must factor a substantial number of different variables, which can make the simulation overly complex, and out of scope for this project. This section provides a simplistic explanation of the electricity bill and in what way the simulation will calculate the price of the REC's energy consumption usage.

As this project is based in continental Portugal, it is important for the simulation of the prices to have a resemblance with the real prices practiced in Portugal, even if the data that are currently available are set in a different location. Portugal's electricity prices are regulated by ERSE – Entidade Reguladora dos Serviços Energéticos [40], and are charged with three different factors [41]:

- **Access to the grids fare** – regulated by the ERSE.
- **Energy Market Prices** – from either the regulated market, or the liberal market.
- **Taxes** – regulated by the state.

Furthermore, it is possible to use different fares based on the time and day the energy was consumed. There can be three different schedule's tariffs that can be chosen by the clients:

- **Simple** – a flat rate fare for the entire day.

- **Bihourly** – two different prices for separate times of day.
- **Trihourly** – three different prices.

The trihourly tariff will not be considered, since most operators only offer this option to clients with an overtly large energy consumption. Simple fares have only one fare, regardless of time and day. Bihourly fares are divided into two categories: peak and out of peak time. In out of peak time, the fares are lower than the simple tariff, whereas the peak fares are higher. Furthermore, the clients have two options, weekly and daily cycle. In the daily cycle option, only the time of day is considered, while in a weekly cycle option, both the time of day, the day of the week, and day of the year (summertime, or wintertime) are considered. The full schedule is documented in Figure 22.

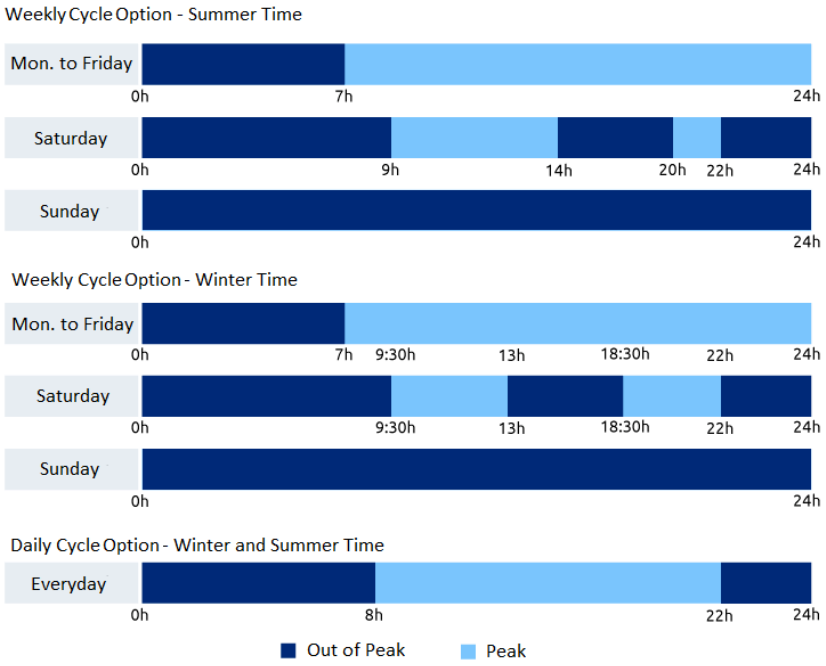

Figure 22 – Bihourly Cycles peak and out of peak time.

For this project, to simplify the process, the price will be calculated using only the tariffs from the regulated market, without taxes, or other expenses. Specifically, the tariffs from the *Normal Low Voltage* option, after 1 April 2022, which can be observed in Table 12.

Table 12 – Prices for Normal Low Voltage from the Regulated Market after 1 April 2022 [42].

		€/kWh
Simple > 2.3 kVA		0.1583
Bihourly	Out of Peak	0.1044
	Peak	0.1917

Finally, with these values, the price of energy consumption can be calculated by the simple formula:

$$P(t) = e_c(t) * tariff(t)$$

(10)

Where:

- $P(t)$ is the cost in € for the consumed energy in the time interval t.
- $e_c(t)$ is the energy consumed in kWh for the time interval t.
- $tariff(t)$ is one of the options already mentioned in Table 12 in €/kWh. It is dependent on the time interval t, unless it is the simple tariff that is being calculated, in that case, $tariff$ is a constant.

4.4. Battery

To consider a user of a REC a prosumer, it is necessary that user has the capacity to produce energy and store it. For that reason, a REC needs one or more batteries, so it has capacity of storage. The full capacity of storage of a community can depend not only on the number of batteries, but their capacity as well.

Analysing real-life examples of solar batteries, we defined three scenarios, small, average, and large storage capacity. An Hoppecke 4 OPzS bloc, for example, can have a capacity of 1.62 kWh, whereas a Tesla Powerwall can have 13.5 kWh. So, with these values, we define that each building in the community will average a capacity of storage of 1.6 kWh for a small capacity scenario, 6.4 kWh for an average capacity, and 12.8 kWh for large capacity. Specifically, each house will have a number N of batteries, so that the full real capacity of all N batteries reaches the defined capacity in each of the scenarios. Moreover, the simulation will not account for the degradation of batteries, or natural draining, but it will consider inverter losses, when the system charges and when discharges the batteries, of 3%.

4.5. Algorithm

The final component of the system is the algorithm. The algorithm is at the center of the process, and, at each time step, it will make the main decisions regarding what to do with the energy produced and stored, as well as choosing the source of energy to consume. In this section, we describe the three different algorithms which will, every 30 minutes, make these decisions, with the goal of reducing the CER's price demands.

4.5.1. Baseline

The first algorithm to be implemented and evaluated is the baseline algorithm. In the baseline scenario, the simulation is made without capacity of storage and with no use of ML. The algorithm runs at every half-hour and orders the consumption of the energy produced. If there is still a need for more energy, then that energy is fetched from the grid. The algorithm is described with a flowchart in Figure 23.

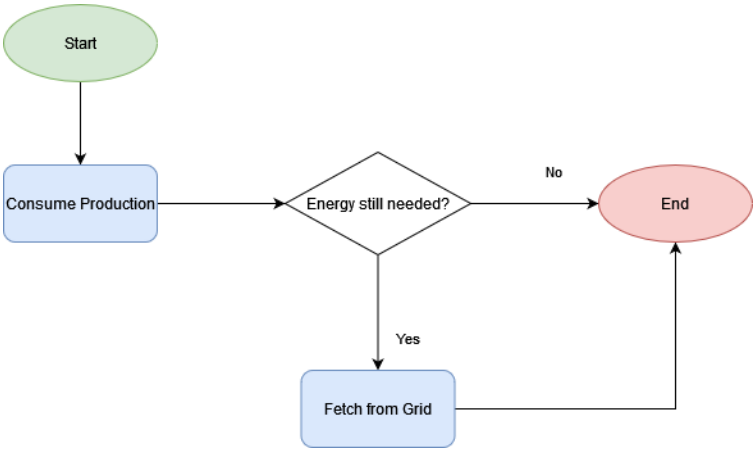

Figure 23 – Baseline algorithm flowchart.

4.5.2. Batteries

A new algorithm is proposed and documented in this section that enriches the baseline algorithm by adding the possibility of storage to the batteries. With this algorithm, the system will consume the produced energy. Then, if there is an excess, it will store it in the battery until it reaches the battery's capacity, selling the remainder energy to grid by a fixed price (0.03 €/kWh). If there is no excess production, and energy is still needed, the system will discharge the battery and consume from it, fetching the rest, if necessary, from the grid. In the same way as the baseline algorithm, the system will also run this algorithm every half-hour and it is described with a flowchart in Figure 24.

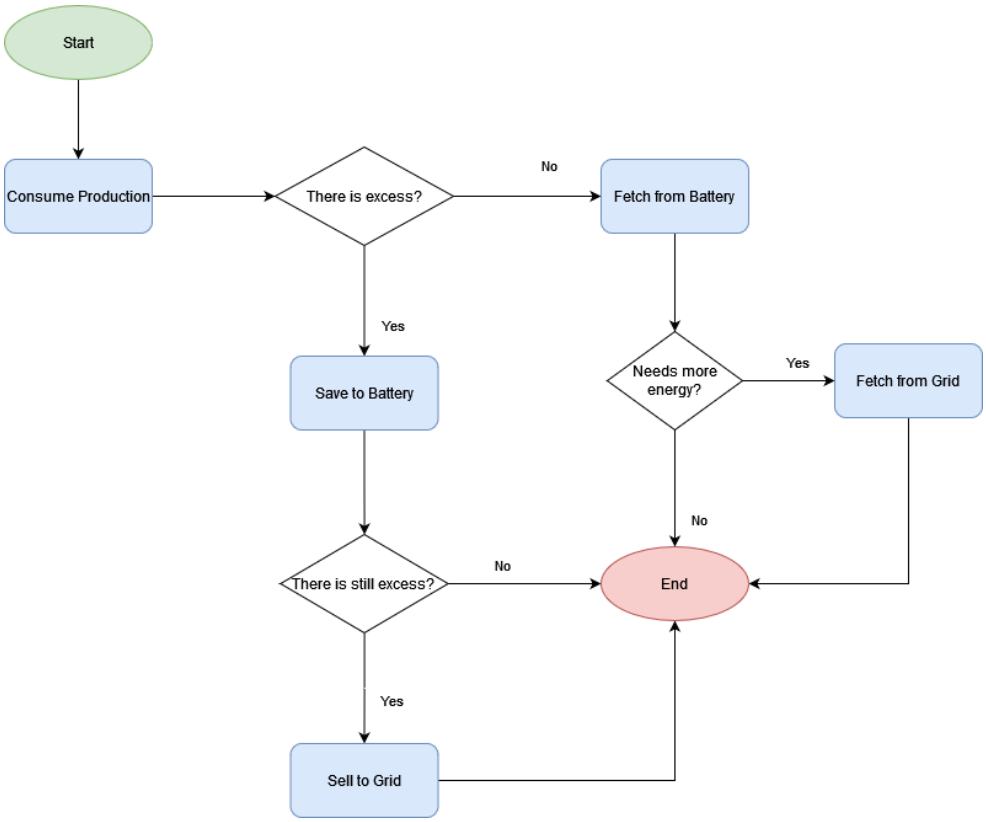

Figure 24 – Battery algorithm flowchart.

4.5.3. ML based

The next algorithm uses ML, specifically the energy consumption forecast, to make decisions regarding the charging and discharging of the batteries. The algorithm adapts [43] by fetching energy from the grid at midnight, when the price tariffs are lower (if the bihourly tariff is chosen).

The algorithm, at midnight, uses the energy production forecast model's output, as well as the output of the energy consumption forecast model, to predict the full day values at midnight. Based on these values, the algorithm will perform one of three actions, according to each of these cases:

- **Case 1**: when there is excess production, i.e., the predicted production is larger than the predicted energy consumption of the REC between 9:00 and 18:00, and the excess is less than the battery capacity available. In this situation, the algorithm will fetch energy from the grid to charge the battery with its capacity minus the predicted excess.
- **Case 2**: when the predicted excess PV energy is larger than the battery capacity available. In this case, the algorithm will fetch from the grid the energy equivalent to the forecast of the morning consumption and save the energy to the battery. More specifically, it will fetch the predicted energy consumption usage between 6:00 and 9:00 of the next day.
- **Case 3**: there is no predicted excess PV energy during the day and therefore the battery is pre-charged to its full battery capacity.

Afterwards, until 10:00, every time the algorithm runs, it orders the consumption of the produced energy. If there is excess production, then it saves it to the battery, until the full capacity is reached. If there is still a remainder of energy, it is sold to the grid (at 0.03 €/kWh). If there is no excess production, the algorithm orders the consumption of energy from the grid.

After 10:00, the algorithm also commands the system to consume the energy produced. However, if there is still need for energy, it first discharges the battery with the energy needed and only then will consume it from the grid. If there is excess production, it proceeds as before 10:00, saving the energy to the battery and selling the remainder, if present, to the grid.

The flowchart of the algorithm is documented in Figure 25, where δ_1 is the ratio of the predicted energy consumption between 9:00 to 18:00 and the full day, and δ_2 is the energy usage forecast of the morning between 6:00 and 9:00 divided by the energy consumption forecast of the day.

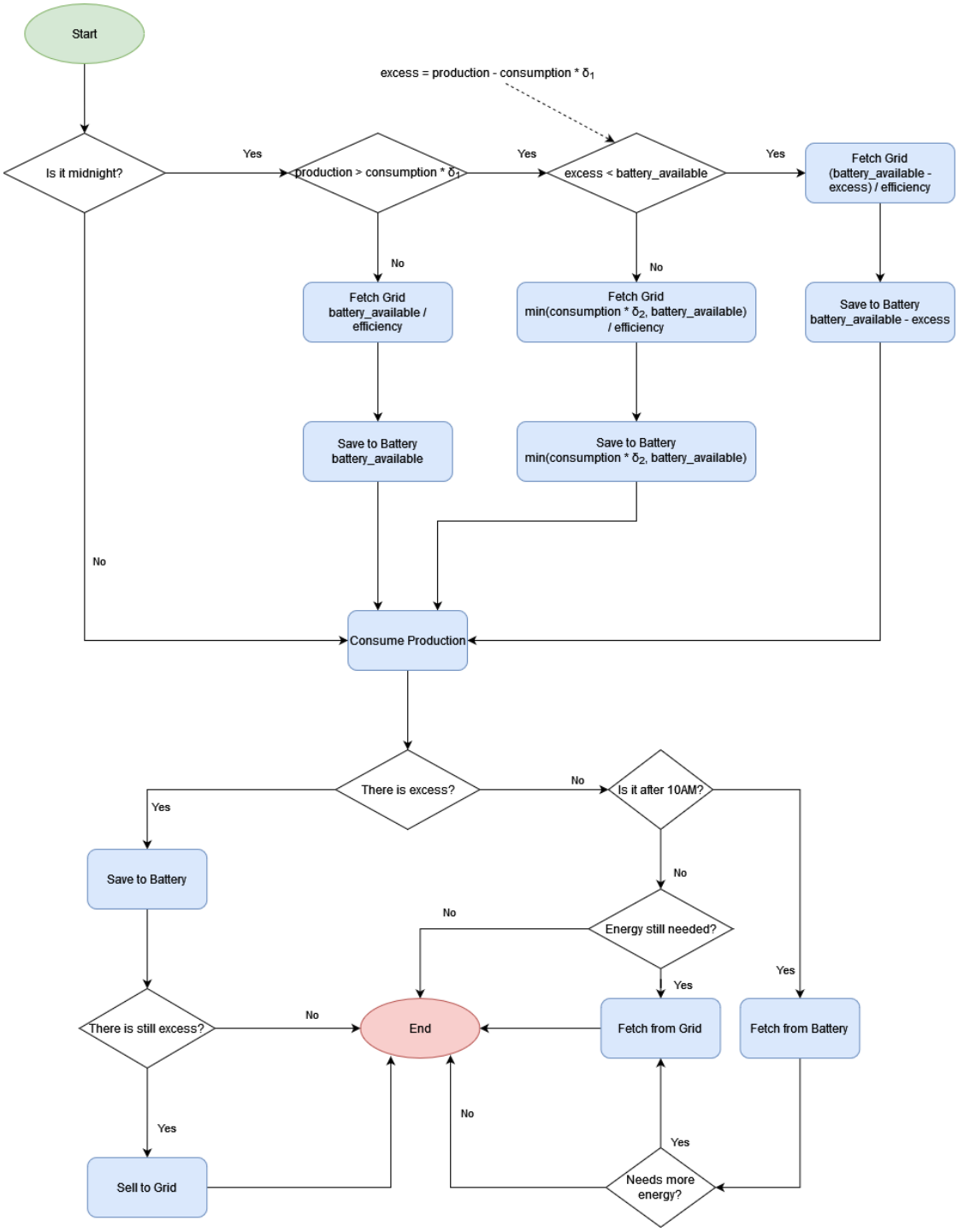

Figure 25 – Flowchart of the ML based algorithm.

5. Validation

This chapter provides, in greater detail, the results of the components of the system. It also draws conclusions on the viability of the project, deciding on how satisfactory those results are.

5.1. Energy Consumption Forecast

The consumption forecast of energy is made with the use of an ML model. The analysis of the state-of-the art concluded that the use of XGBoost, ANN, or GB were the most suitable for this problem. The results indicated that, of these three algorithms, XGBoost would perform better, and further studies on how to improve the algorithm were conducted. Specifically, the feature classification method was used. This section contains the analysis of the results of both experiments – algorithms comparison, and feature classification.

5.1.1. Algorithms Comparison

After the training and testing phases were completed, the performance of the algorithms is compared using six different metrics – the time it took to perform the training phase, the time it took to predict the test dataset, MSE, RMSE, MAE, wMAPE, and R^2. Below, in Table 13, the results of these error metrics on the test dataset are documented, as well as the time it took to train the algorithms and to make the predictions. We can see that all the algorithms perform well, particularly XGBoost, which performs better than all the other algorithms in every error metric. Furthermore, although the error can be large in some circumstances, the distribution of the error is mostly around zero, as it is documented in Figure 26. XGBoost is, therefore, the focus of the next subsection, where the results of other ways to maximize the capabilities of the model are explored.

Table 13 – Results of the predictions on the test dataset, with the best results of each metric at bold.

	ANN	**GBDT**	**XGBoost**
Training Time (s)	268.7992	11676.7068	**54.2282**
Prediction Time (s)	45.1752	8.1269	**1.9277**
MSE (kWh)	0.8107	0.9097	**0.6629**
RMSE (kWh)	0.9004	0.9538	**0.8142**
MAE (kWh)	0.3601	0.3683	**0.3290**
wMAPE (%)	8.9294	9.1313	**8.1575**
R^2	0.9886	0.9872	**0.9907**

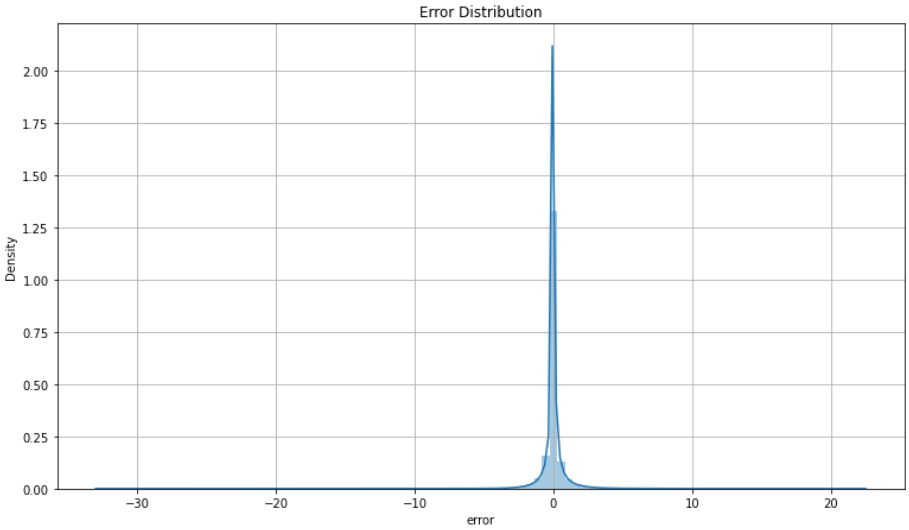

Figure 26 – Error distribution of XGBoost model trained with the previous 24 hours and weather metrics.

5.1.2. Feature Classification

The feature classification consisted of two phases, a preliminary phase that trains a model containing the information of the previous week, on which the feature classification technique is used, and, following its results, a training phase where two models are trained and evaluated. The first model uses only the features with an importance classification of over 0.5%, and the other uses the top forty features according to their classification.

- **Perform feature classification**

The results of the preliminary phase are documented in this subsection. The importance classification of the forty most prominent features is registered in Figure 27. We can see that the direct previous energy reading impacts the algorithm the most, with a score of over 80%, surpassing every other feature by a substantial margin.

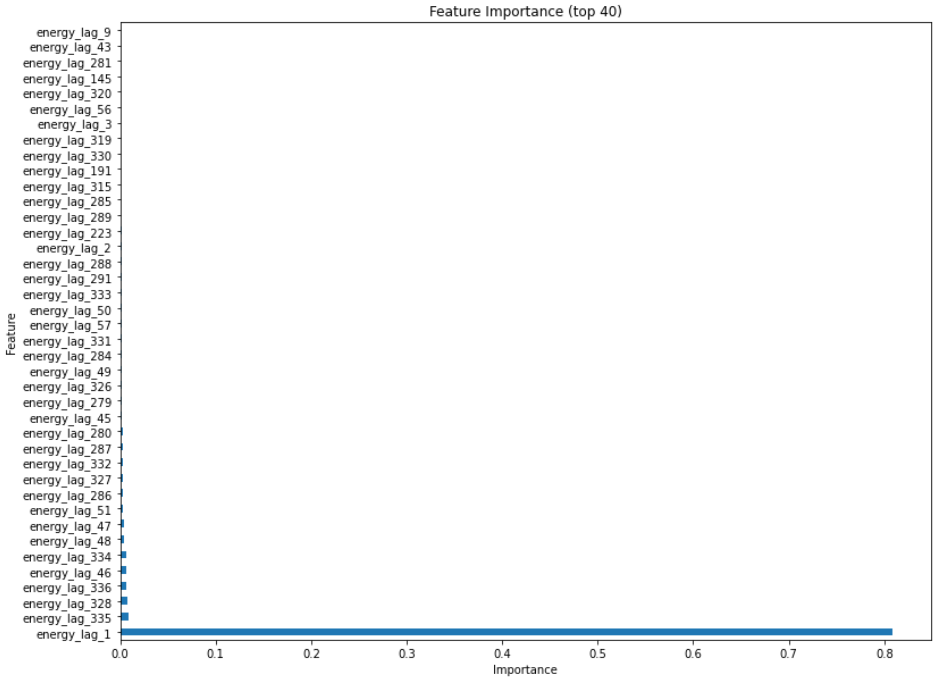

Figure 27 – The 40 features with the highest importance classification

In Figure 28, the same graph as above is documented, but without the *energy lag 1 feature*. We can see that the most important parameters are the ones around the same hours the week earlier, as well as the day before. With these results, the next phase details the results of the evaluation of two models, one with the forty most predominant features, represented in the *y* axis in the figure above, and another with the features with an importance classification of over 0.5%.

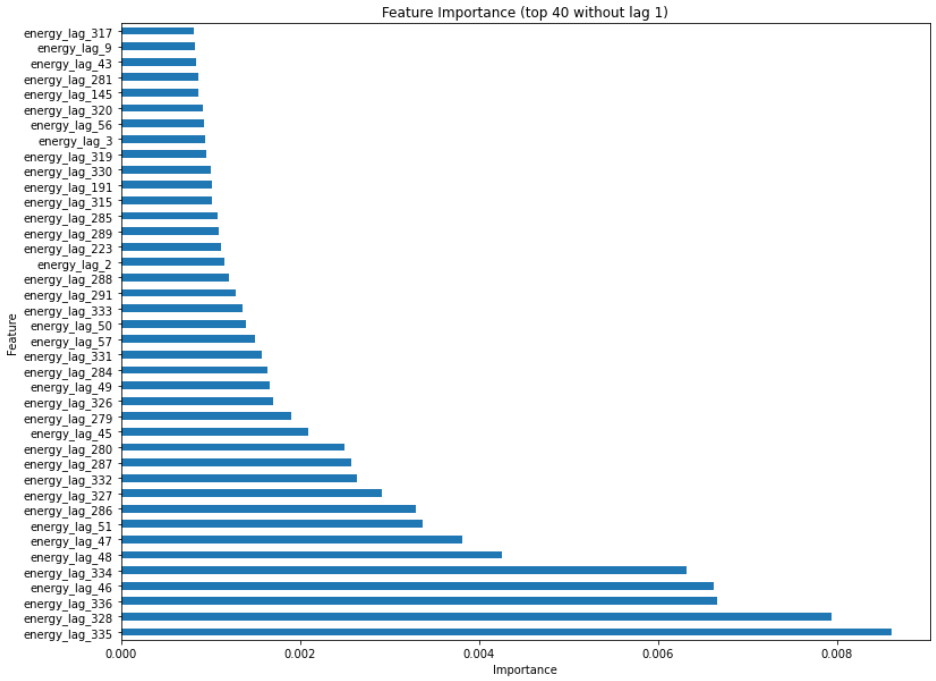

Figure 28 – The 40 features, besides *energy lag 1*, with the highest importance classification

- **Train models using the features' classification**

In this phase, the two models mentioned before, which are trained by splitting the data into a train a test dataset with 80% and 20% of the entire data, respectively, are evaluated using the test dataset. The results of the performance metrics are presented in Table 14.

Table 14 – Performance metrics on 20% of the data.

	Feature importance > 0.5%	**Top forty features**
Training Time (s)	**11.4464**	42.7416
Prediction Time (s)	**0.7795**	1.2840
MSE (kWh)	0.7200	**0.5372**
RMSE (kWh)	0.8485	**0.7329**
MAE (kWh)	0.3320	**0.3006**
wMAPE (%)	8.1936	**7.4197**
R^2	0.9899	**0.9925**

Another approach, where different *n* buildings are left out of the training phase and used to evaluate the model, is also analysed in this phase, since in a real-life situation the model would be trained with data from different communities. Below, we can see the results of the study, where, for each *n* houses, the experiment was repeated five times, choosing new random houses each time. It is also documented the mean ($\bar{x}$) and standard deviation (σ) of each performance metric. With a large standard deviation in most metrics when five houses were picked, it can be deduced that the model cannot be evaluated with such small number of buildings. The wMAPE metric alone records values ranging from less than 12% to more than 21%, which is unacceptable, when the deviation should be around zero. On the other hand, when ten or fifteen houses were used, the performance metrics of all five tests recorded similar values. So, for this approach, a minimum number of ten houses must be used as the test dataset.

Table 15 – Performance metrics with five houses as the test data.

	1	**2**	**3**	**4**	**5**	$\bar{x}$	σ
training time (s)	14.2463	13.3309	13.2278	12.8021	12.8308	*13.2876*	*0.5850*
prediction time (s)	0.4349	0.4177	0.5004	0.9100	0.6033	*0.5733*	*0.2018*
MSE (kWh)	0.2096	0.5362	3.1264	0.3500	0.4609	*0.9366*	*1.2303*
RMSE (kWh)	0.4578	0.7323	1.7682	0.5916	0.6789	*0.8458*	*0.5260*
MAE (kWh)	0.2166	0.3063	0.8334	0.2576	0.2479	*0.3724*	*0.2597*
wMAPE (%)	13.1885	13.6776	5.7013	20.3205	11.0990	*12.7974*	*5.2622*
R^2	0.9555	0.9464	0.9895	0.9004	0.9720	*0.9528*	*0.0336*

Table 16 – Performance metrics with ten houses as the test data.

	1	**2**	**3**	**4**	**5**	$\bar{x}$	σ

training time (s)	12.1195	11.3366	12.5359	10.9098	11.8923	*11.7588*	*0.6424*
prediction time (s)	0.7870	0.9550	1.0450	1.1390	1.0160	*0.9884*	*0.1307*
MSE (kWh)	0.8164	0.6926	0.6846	0.8845	0.6866	*0.7529*	*0.0923*
RMSE (kWh)	0.9035	0.8322	0.8274	0.9405	0.8286	*0.8664*	*0.0524*
MAE (kWh)	0.3480	0.3303	0.3286	0.3417	0.3282	*0.3354*	*0.0090*
wMAPE (%)	8.5893	8.1511	8.1108	8.4348	8.1005	*8.2773*	*0.2220*
R^2	0.9886	0.9903	0.9904	0.9877	0.9904	*0.9895*	*0.0013*

Table 17 – Performance metrics with fifteen houses as the test data.

	1	**2**	**3**	**4**	**5**	$\bar{x}$	σ
training time (s)	11.5723	11.0528	10.2737	11.1070	10.2865	*10.8585*	*0.5653*
prediction time (s)	1.0780	1.0716	1.3300	0.8800	0.9640	*1.0647*	*0.1695*
MSE (kWh)	0.7457	0.7167	0.6920	0.7007	0.6883	*0.7087*	*0.0234*
RMSE (kWh)	0.8635	0.8466	0.8318	0.8370	0.8296	*0.8417*	*0.0138*
MAE (kWh)	0.3346	0.3300	0.3303	0.3324	0.3312	*0.3317*	*0.0019*
wMAPE (%)	8.2573	8.1452	8.1515	8.2033	8.1751	*8.1865*	*0.0457*
R^2	0.9896	0.9900	0.9903	0.9902	0.9904	*0.9901*	*0.0003*

Finally, the approach was repeated with the forty features with highest importance. As the results above showed, using ten buildings to assess the model is sufficient. In Table 18 the results of the performance metrics are presented and compared with the metrics using the entire dataset, when ten houses were used to evaluate the model using the features with a classification larger than 0.5%.

Table 18 – Performance metrics on ten buildings for the top forty features and features with importance classification higher than 0.5%.

	Feature importance > 0.5%	**Top forty features**
training time (s)	**11.0428**	45.7238
prediction time (s)	**1.2434**	1.3590
MSE (kWh)	**0.2370**	1.7960
RMSE (kWh)	**0.4868**	1.3401
MAE (kWh)	**0.2098**	0.4715

wMAPE (%)	9.9973	**8.8847**
R^2	0.9798	**0.9871**

5.1.3. Results and discussion

Regarding the energy consumption forecast component of the system, it is possible to draw several conclusions. The first corresponds to the fact that the XGBoost algorithm performs better in this situation than other state-of-the-art algorithms like ANN and GB. With a wMAPE of just 8.1575% when evaluated with the previous 24 hours and weather metrics, another assessment that can be made is that the algorithm loses little, or not at all, when the number of features is reduced to the most prominent features. Specifically, when the forty parameters with highest classification (a reduction of ten features) is used, the model reduces the wMAPE to 7.4197%, and when only the features with an importance classification higher than 0.5% (a reduction of forty-four features) are evaluated, the model retrieves an error of 8.1936%, a difference of less than 0.1%. In the scenario where ten houses are left out of the training phase, and are, instead, used to evaluate the model, the performance metrics of the model with only six features are better than the model with forty features, except the wMAPE, which has an increase of 1.1126%, and the R^2.

With these results, although using forty input parameters improves the wMAPE error metric, the model that generates the output of the energy consumption forecast component is the model with the features that have an importance score of above 0.5%. The reasoning behind that is the higher flexibility that a small number of features offers, and the fact that other metrics, like the training time, have a better score than with a higher number of features.

Although we are predicting the short-term consumption of individual buildings, the goal of this project is the management of a community. Therefore, the individual houses' forecast is not so important, but the community's consumption as a whole. So, after the algorithm predicts the separate building's consumption, the actual forecast of the REC's energy consumption is the sum of those predictions. Furthermore, the algorithm will also need to predict, at midnight, the consumption of the entire day. This means that the algorithm will forecast at midnight the short-term consumption using real values, and then predicts the next step using the real values when possible, or the values that were predicted, thus propagating the error. The Table 19 describes the performance metrics of both situations, when the algorithm predicts the short-term forecast of the community, and when predicts the entire day at midnight, with the values of each house in both cases being grouped and summed. The all-day prediction and short-term forecasting are compared to the real values in Figure 29 and Figure 30, respectively. All-day forecasting lacks in comparison to short-term forecasting, but still manages to maintain a similar pattern with the real values.

Table 19 – Performance metrics of the energy consumption forecast of all the community

	Short-term load forecasting	**All-day load forecasting**
MAE (kWh)	1.1819	2.6361
MSE (kWh)	3.1929	17.0603
RMSE (kWh)	1.7869	4.1304
MAPE (%)	6.1067	13.3331
R^2	0.9700	0.8396

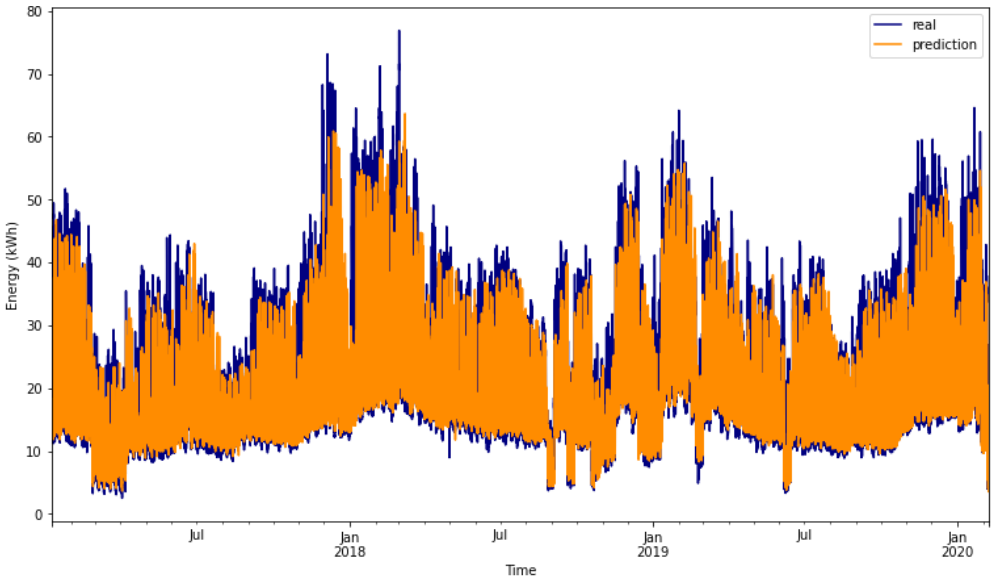

Figure 29 – All-day load forecasting compared to real values.

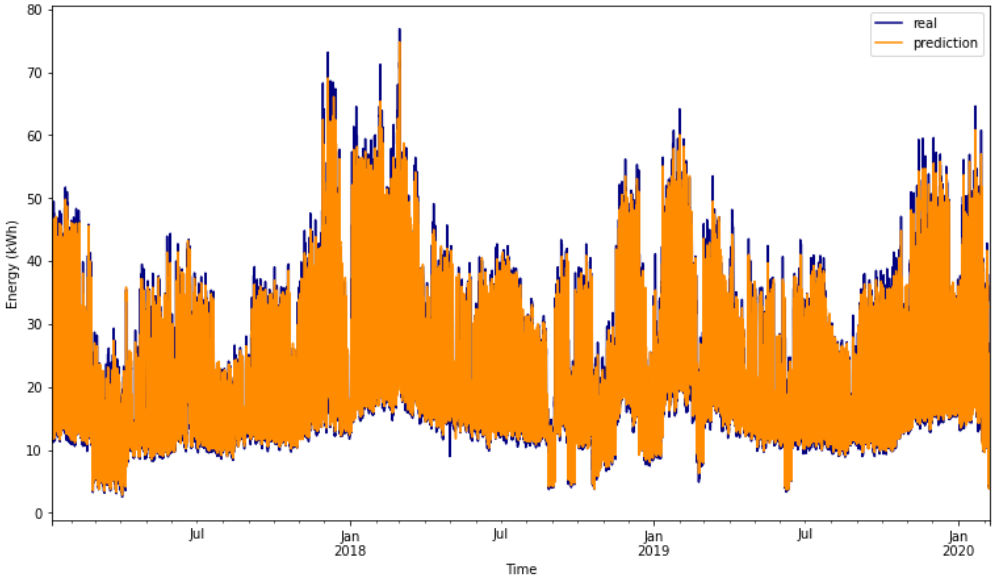

Figure 30 – Short-term load forecasting compared to real values.

5.2. Energy Production Forecast

In this section, the values of the energy production forecast model are presented and described. In Figure 31, the output of the model over time is presented. Table 20 provides the description of the outputs of the model, including the mean ($\bar{x}$), standard deviation (σ), and sum ($\sum$), as well as the minimum and maximum values, and the percentiles. Finally, the output is multiplied by the number of buildings present in the community.

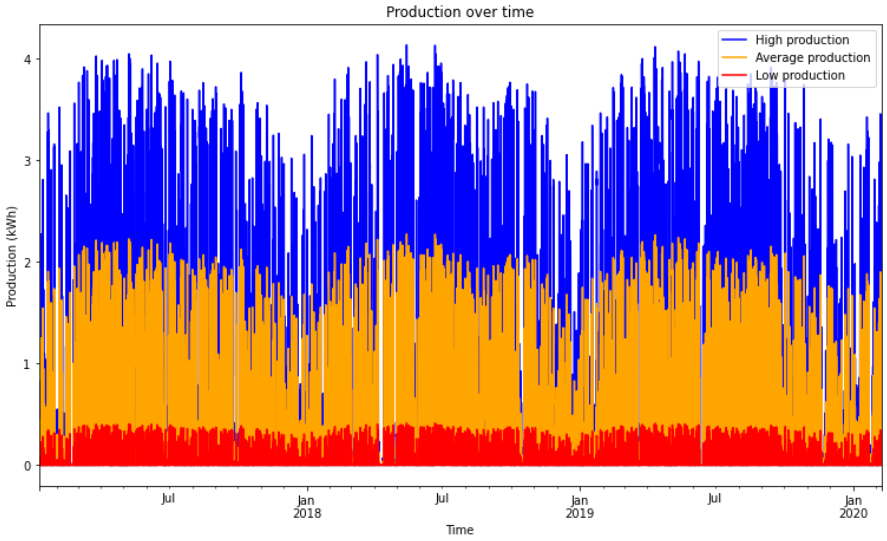

Figure 31 – Production of energy per building over time.

Table 20 – Description of the model's outputs, according to scenarios of high, low, and average production.

	High Production (kWh)	Low Production (kWh)	Average Production (kWh)
$\bar{x}$	0.4320	0.0432	0.2376
σ	0.9680	0.0968	0.5324
Min	0.0000	0.0000	0.0000
25%	0.0000	0.0000	0.0000
50%	0.0000	0.0000	0.0000
75%	0.0744	0.0074	0.0409
Max	4.1368	0.4136	2.2752
$\sum$	23352.8281	2335.2828	12844.0555

5.3. Market Prices

This section presents the results of the average price paid by each household over time, since the beginning of the dataset. Table 21 represents the description of all the values, including the mean ($\bar{x}$), standard deviation (σ), and sum ($\sum$), as well as the minimum and maximum values, and the percentiles of the average values paid by a user. Figure 32 shows the cumulative sum of the mean electricity bill paid by the user over time, if all the energy is fetched from the distribution network. During the simulation, the values of the energy are grouped, and may account for the possibility of storage and production, which can make these values vary from the documented here.

Table 21 – Description of the model's outputs, according to scenarios of bihourly tariff (daily), Bi-hourly tariff (weekly), and simple tariff.

	Bihourly tariff (daily) (€)	Bihourly tariff (weekly) (€)	Simple tariff (€)
$\bar{x}$	1.2443	1.2344	1.2003
σ	0.9680	0.0968	0.5324
Min	1.8411	1.8388	1.5997
25%	0.3369	0.3044	0.3981
50%	0.5800	0.6037	0.6106

75%	1.2634	1.2706	1.0751
Max	11.5308	11.5308	9.5217
∑	285857.4185	283567.7304	275740.5707

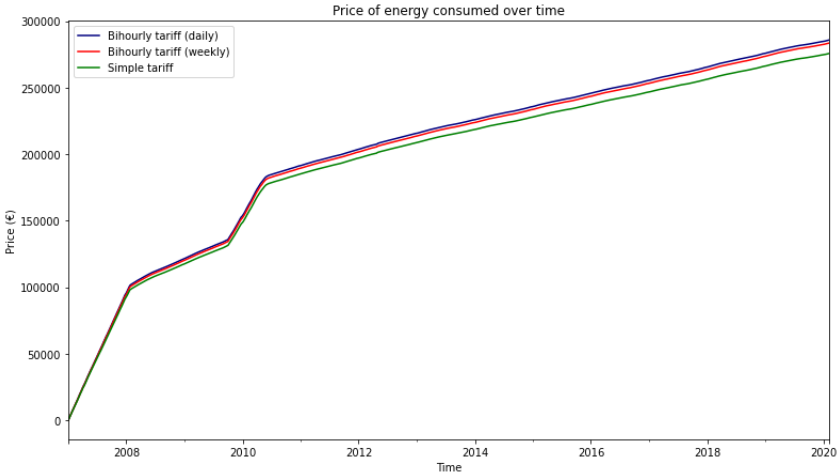

Figure 32 – Cumulative sum of the average price paid by the consumer, according to the three tariff scenarios.

5.4. Algorithm

In the previous chapter, we proposed three algorithms. In this section, those algorithms are evaluated regarding the electricity bill and consumption from green sources, concluding on the viability of each method. The algorithms are tested using a REC of ten buildings, which were left out of the training phase for the energy consumption forecast component, and evaluated using three scenarios of storage capacity, and three scenarios of production.

5.4.1. Baseline

The algorithm was evaluated using ten houses in the dataset, between 8 January 2017 and 7 February 2020. According to the results, documented in Table 22, the REC consumes more than 160 000 kWh in the three-year span when the production is low, than when there is large production, which accounts for more than a 25 500 € difference when the simple tariff (the lowest in all scenarios) is used. The next algorithms will compare to the baseline in terms of prices and grid energy consumption.

Table 22 – Price and consumption from the grid, in high, low, and average scenarios, using the baseline algorithm for a REC of ten buildings.

Production		**Baseline**
High Production	*Bihourly tariff (weekly option) (€)*	146 364.0864
	Bihourly tariff (daily option) (€)	146 434.8987
	Simple tariff (€)	144 322.7950
	Consumed From Grid (kWh)	911 704.3270
Low Production	*Bihourly tariff (weekly option) (€)*	176 246.0450
	Bihourly tariff (daily option) (€)	175 254.3673
	Simple tariff (€)	169 883.2251
	Consumed From Grid (kWh)	1 073 172.6156

Average Production	*Bihourly tariff (weekly option) (€)*	158 095.2314
	Bihourly tariff (daily option) (€)	158 055.0320
	Simple tariff (€)	154 340.5272
	Consumed From Grid (kWh)	974 987.5379

5.4.2. Batteries

In the previous algorithm, the results were presented for three scenarios regarding the REC's capacity of production. Table 23, Table 24, and Table 25, document, respectively, the low, average, and high scenarios for capacity of storage. Each one also presents the scenarios for high, low, and average production, when the batteries algorithm is executed, showing a comparison with the baseline algorithm, regarding the price of the three possible tariffs, and the consumption from the grid.

Table 23 – Comparison between batteries algorithm and baseline for low storage scenario (16 kWh).

Production		**Baseline**	**Batteries**
High	*Bihourly (weekly) (€)*	146 364.0864	**140 829.1992**
	Bihourly (daily) (€)	146 434.8987	**141 125.5911**
	Simple (€)	144 322.7950	**139 058.6055**
	Grid (kWh)	911 704.3270	**903 360.8796**
Low	*Bihourly (weekly) (€)*	176 246.0450	**176 245.0032**
	Bihourly (daily) (€)	175 254.3673	**175 253.4215**
	Simple (€)	169 883.2251	**169 882.2200**
	Grid (kWh)	1 073 172.6156	**1 073 171.5146**
Average	*Bihourly (weekly) (€)*	158 095.2314	**156 903.2119**
	Bihourly (daily) (€)	158 055.0320	**156 989.9957**
	Simple (€)	154 340.5272	**153 228.0538**
	Grid (kWh)	974 987.5379	**972 550.0141**

Table 24 – Comparison between batteries algorithm and baseline for average storage scenario (64 kWh).

Production		**Baseline**	**Batteries**
High	*Bihourly (weekly) (€)*	146 364.0864	**138 906.5757**
	Bihourly (daily) (€)	146 434.8987	**139 523.4337**
	Simple (€)	144 322.7950	**137 474.3989**
	Grid (kWh)	911 704.3270	**890 829.1492**
Low	*Bihourly (weekly) (€)*	176 246.0450	**176 245.0083**
	Bihourly (daily) (€)	175 254.3673	**175 253.4243**
	Simple (€)	169 883.2251	**169 882.2242**
	Grid (kWh)	1 073 172.6156	**1 073 171.5413**
Average	*Bihourly (weekly) (€)*	158 095.2314	**156 547.5470**
	Bihourly (daily) (€)	158 055.0320	**156 757.9280**
	Simple (€)	154 340.5272	**152 930.9751**
	Grid (kWh)	974 987.5379	**970 189.6801**

Table 25 – Comparison between batteries algorithm and baseline for high storage scenario (128 kWh).

Production		Baseline	Batteries
High	*Bihourly (weekly) (€)*	146 364.0864	**137 912.5288**
	Bihourly (daily) (€)	146 434.8987	**138 594.9123**
	Simple (€)	144 322.7950	**136 286.6575**
	Grid (kWh)	911 704.3270	**881 433.6229**
Low	*Bihourly (weekly) (€)*	176 246.0450	**176 245.0083**
	Bihourly (daily) (€)	175 254.3673	**175 253.4243**
	Simple (€)	169 883.2251	**169 882.2242**
	Grid (kWh)	1 073 172.6156	**1 073 171.5413**
Average	*Bihourly (weekly) (€)*	158 095.2314	**156 454.1503**
	Bihourly (daily) (€)	158 055.0320	**156 685.2525**
	Simple (€)	154 340.5272	**152 805.2904**
	Grid (kWh)	974 987.5379	**969 195.4626**

With these results, we can conclude that the use of batteries improves the baseline algorithm in every scenario. However, for a low production situation, the gains from using batteries are negligible. So, we can assume that the batteries are not worth the cost if the produced energy is not in enough quantity. With an average production, the REC has gains of between 1065.04€, and 1641.08€. Assuming each user in the community has the same capacity of storage, each user gains around 106.50€ and 164.10€. If the profits remain constant, it is possible that a user can return a profit, assuming a battery has ten years of lifetime. However, the investment on batteries could be of too much risk to be advisable. With a high production, the gains can be of 5264.19€ or 8451.56€. In this scenario, it is highly likely a user can return a profit.

5.4.3. ML based

Following the results of the previous algorithms, the same approach is followed for the ML based algorithm. This algorithm, however, does not have the goal of reducing the consumption of energy from the grid, nor the simple tariff price. Instead, the aim is to reduce the bihourly tariffs prices, so that they have the lowest price in that specific scenario. The following tables detail the results of the evaluation of the algorithm.

Table 26 – Comparison between baseline, batteries algorithm and ML based for low storage scenario (16 kWh).

Production		Baseline	Batteries	ML based
High	*Bihourly (weekly) (€)*	146 364.0864	140 829.1992	**140 100.7655**
	Bihourly (daily) (€)	146 434.8987	141 125.5911	**140 538.2377**
	Simple (€)	144 322.7950	**139 058.6055**	139 773.1007
	Grid (kWh)	911 704.3270	**903 360.8796**	908 738.0779
Low	*Bihourly (weekly) (€)*	176 246.0450	176 245.0032	**174 833.5646**
	Bihourly (daily) (€)	175 254.3673	175 253.4215	**174 060.1188**
	Simple (€)	169 883.2251	**169 882.2200**	170 055.3370
	Grid (kWh)	1 073 172.6156	**1 073 171.5146**	1 074 265.1157

Average	*Bihourly (weekly) (€)*	158 095.2314	156 903.2119	**155 688.4566**
	Bihourly (daily) (€)	158 055.0320	156 989.9957	**155 942.4465**
	Simple (€)	154 340.5272	**153 228.0538**	153 557.7401
	Grid (kWh)	974 987.5379	**972 550.0141**	**974 871.9488**

Table 27 – Comparison between baseline, batteries algorithm and ML based for average storage scenario (64 kWh).

Production		**Baseline**	**Batteries**	**ML based**
High	*Bihourly (weekly) (€)*	146 364.0864	138 906.5757	**135 421.2991**
	Bihourly (daily) (€)	146 434.8987	139 523.4337	**136 634.3897**
	Simple (€)	144 322.7950	**137 474.3989**	139 820.2150
	Grid (kWh)	911 704.3270	**890 829.1492**	908 292.8994
Low	*Bihourly (weekly) (€)*	176 246.0450	176 245.0083	**170 602.3039**
	Bihourly (daily) (€)	175 254.3673	175 253.4243	**170 492.4297**
	Simple (€)	169 883.2251	**169 882.2242**	170 576.6233
	Grid (kWh)	1 073 172.6156	**1 073 171.5413**	1 077 558.1433
Average	*Bihourly (weekly) (€)*	158 095.2314	156 547.5470	**151 409.0531**
	Bihourly (daily) (€)	158 055.0320	156 757.9280	**152 395.9916**
	Simple (€)	154 340.5272	**152 930.9751**	153 970.8334
	Grid (kWh)	974 987.5379	**970 189.6801**	977 321.1353

Table 28 – Comparison between baseline, batteries algorithm and ML based for high storage scenario (128 kWh).

Production		**Baseline**	**Batteries**	**ML based**
High	*Bihourly (weekly) (€)*	146 364.0864	137 912.5288	**130 170.7481**
	Bihourly (daily) (€)	146 434.8987	138 594.9123	**132 201.3186**
	Simple (€)	144 322.7950	**136 286.6575**	140 009.8316
	Grid (kWh)	911 704.3270	**881 433.6229**	908 788.3554
Low	*Bihourly (weekly) (€)*	176 246.0450	176 245.0083	**164 985.0157**
	Bihourly (daily) (€)	175 254.3673	175 253.4243	**165 992.7034**
	Simple (€)	169 883.2251	**169 882.2242**	171 270.0897
	Grid (kWh)	1 073 172.6156	**1 073 171.5413**	1 081 938.8536
Average	*Bihourly (weekly) (€)*	158 095.2314	156 454.1503	**145 943.0327**
	Bihourly (daily) (€)	158 055.0320	156 685.2525	**147 922.0113**
	Simple (€)	154 340.5272	**152 805.2904**	154 515.1524
	Grid (kWh)	974 987.5379	**969 195.4626**	980 555.5266

The results show that, in every situation, the algorithm can successfully reduce the prices of the bihourly tariffs. However, the algorithm cannot always improve the results of the bihourly tariffs, as they are not always lower than the simple tariffs' prices of either the baseline, or batteries algorithm. In the low storage capacity scenario, the ML based algorithm cannot improve the simple tariff prices of both production scenarios. With average storage capacity, only with low production the algorithm showed poor results. With a high production, the algorithm improved the batteries algorithm by 2053.10€, and 8901.50€ to the baseline algorithm. With average production, the

algorithm resulted in an improvement of 1521,92€ to the batteries' algorithm and 2931.47€ the baseline. These results could turn a potentially risky investment into a profitable deal. In the high storage capacity scenario, the algorithm heavily reduced the electricity bill, with an improvement of 14152.04€, 4898.20€, and 8397.49€ for high, low, and average productions.

6. Conclusion

This investigation aims at the creation of a management system for a REC, to reduce the energy usage from "non-green" sources, or the reduction of the electricity bill, paid by a REC, by using ML techniques to forecast the energy consumption of a community. At first, a study of the State-of-the-Art, which makes an analysis of ML algorithms, and a review of related literature, was made. The investigation pointed to three different algorithms that could be used on the energy consumption forecast (ANN, XGBoost, and GB), as well as the overall process necessary to conduct the research. Then, an architecture is proposed. The architecture contains four key components (PV production forecast, Market Electricity Prices, Energy Consumption forecast, and Battery) plus the algorithm. Based on the information from these components the algorithm can decide when to charge or discharge the batteries. The components' characteristics and behaviour are explored in greater detail in the implementation and validation chapters, where their implementation and the results are investigated. The first study, regarding the energy consumption forecast module, concludes on the viability of the XGBoost algorithm, providing immediately better results than the other algorithms. Further studies, using XGBoost, also deduce on the most important input parameters, which do not include weather features, but include the immediate previous energy reading, as well as readings from the previous day and week. The results were found to be acceptable since the error metrics did not retrieve high values. Therefore, we can conclude on the viability of ML to predict the energy consumption of a REC. Moreover, an experiment where buildings are left out of the training phase and are used to evaluate the algorithm showed that a minimum number of ten buildings must be used as the test dataset, since the deviation of the error metrics across experiments is low.

After defining three scenarios for storage, and three scenarios for production, three algorithms were proposed and evaluated according to the sum of the electricity bill of the community, as well as the consumption of energy from the distribution network. The results show that adding batteries to the baseline algorithm, and selling the excess to distribution network, reduce the consumption from the grid, as well the energy prices in every situation. However, the reduction of prices may not be enough in the scenarios where the production is low to outweigh the expenses of purchasing batteries. The ML based algorithm, which charges the batteries at midnight, can successfully reduce the prices of energy if the REC chooses to use bihourly tariffs, instead of a flat rate, in the situations where the capacity of storage is large, or, with an average capacity, if the production capacity is at least average. In the case where the capacity of storage is low, although it improves the bihourly tariffs, the improvement is not enough to reduce the costs to lower than the price of energy of simple tariff, when the batteries algorithm is used. This happens similarly to the case where the production is low, and the storage capacity is average. The reason behind the lack of improvement in this last case is because the hours of daylight, i.e., the hours of more production, usually overlap with the peak hours, when the energy is more expensive. Therefore, when there is not enough production to cover this energy, the cost of energy using bihourly tariffs is more expensive than the use of simple tariffs, and even with the improvements of the ML based algorithm (which were larger than in other scenarios of production), it is still not enough to improve the simple tariff case. When storage capacity is low, the batteries are pre-charged at midnight, but do not have capacity to provide energy during the day to the community and are mostly discharged after 10:00, leaving little to no energy for the rest of peak hours. This reduces the usefulness of the algorithms, which results in a higher price with bihourly tariffs using the ML based algorithm when compared

to the price with a single tariff using the batteries algorithm. Moreover, the consumption from the grid is larger in every situation, with the ML algorithm, since there is more energy charged to the batteries, which suffer from losses when charged and discharged, and those losses are covered by the distribution network.

6.1. Future Work

For future work, some improvements and changes should be made to this project. Firstly, the energy consumption forecast is done using XGBoost to predict the short-term energy consumption. However, the algorithm needs to predict the energy at midnight for the next day, so an algorithm with multiple outputs should present more reliable results. Moreover, the energy production forecast is made using a model that uses the real weather parameters and assumes no error from it. In real life, however, only a forecast will be available, and not the actual values, so the simulation should assume some errors. Lastly, the system manages the REC as a single entity, but, in the future, it should also manage the members of the community, in regards of trading of energy between themselves.

7. References

[1] T. Mitchell, *Machine Learning*. McGraw-Hill Science/Engineering/Math, 1997.

[2] S. Russel and P. Norvig, *Artificial Intelligence: A Modern Approach*, 4th Ed. Prentice Hall, 2020.

[3] P. Cunningham, M. Cord, and S. J. Delany, "Supervised Learning," *Cogn. Technol.*, pp. 21–49, 2008, doi: 10.1007/978-3-540-75171-7_2.

[4] G. Hinton and T. Sejnowski, *Unsupervised Learning: Foundations of Neural Computation*. MIT Press, 1999.

[5] B. Mehlig, "Machine learning with neural networks," *Mach. Learn. with Neural Networks*, Jan. 2019, doi: 10.1017/9781108860604.

[6] L. Noriega, "Multilayer Perceptron Tutorial," *undefined*, 2005.

[7] "1.17. Neural network models (supervised) — scikit-learn 1.0.2 documentation." https://scikit-learn.org/stable/modules/neural_networks_supervised.html (accessed Dec. 28, 2021).

[8] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, doi: 10.1162/NECO.1997.9.8.1735.

[9] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling," Dec. 2014, Accessed: Dec. 28, 2021. [Online]. Available: https://arxiv.org/abs/1412.3555v1.

[10] N. Walia, H. Singh, and A. Sharma, "ANFIS: Adaptive Neuro-Fuzzy Inference System- A Survey," *Int. J. Comput. Appl.*, vol. 123, no. 13, pp. 32–38, Aug. 2015, doi: 10.5120/IJCA2015905635.

[11] T. K. Ho, "Random decision forests," *Proc. Int. Conf. Doc. Anal. Recognition, ICDAR*, vol. 1, pp. 278–282, 1995, doi: 10.1109/ICDAR.1995.598994.

[12] "Understanding Random Forest. How the Algorithm Works and Why it Is… | by Tony Yiu | Towards Data Science." https://towardsdatascience.com/understanding-random-forest-58381e0602d2 (accessed Dec. 29, 2021).

[13] R. E. Schapire, "Explaining AdaBoost," in *Empirical Inference: Festschrift in Honor of Vladimir N. Vapnik*, Springer, Berlin, Heidelberg, 2013, pp. 37–52.

[14] A. Natekin and A. Knoll, "Gradient boosting machines, a tutorial," *Front. Neurorobot.*, vol. 7, no. DEC, p. 21, 2013, doi: 10.3389/FNBOT.2013.00021/BIBTEX.

[15] "dmlc/xgboost: Scalable, Portable and Distributed Gradient Boosting (GBDT, GBRT or GBM) Library, for Python, R, Java, Scala, C++ and more. Runs on single machine, Hadoop, Spark, Dask, Flink and DataFlow." https://github.com/dmlc/xgboost (accessed Jun. 17, 2022).

[16] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," doi: 10.1145/2939672.2939785.

[17] C. Cortes, V. Vapnik, and L. Saitta, "Support-vector networks," *Mach. Learn. 1995 203*, vol. 20, no. 3, pp. 273–297, Sep. 1995, doi: 10.1007/BF00994018.

[18] D. Meyer, “Support vector machines,” *mran.microsoft.com*, 2017, Accessed: Dec. 30, 2021. [Online]. Available: https://mran.microsoft.com/snapshot/2018-06-22/web/packages/e1071/vignettes/svmdoc.pdf.

[19] Z. de Grève *et al.*, “Machine learning techniques for improving self-consumption in renewable energy communities,” *Energies*, vol. 13, no. 18, Sep. 2020, doi: 10.3390/EN13184892.

[20] N. Dimitropoulos *et al.*, “Forecasting of short-term PV production in energy communities through Machine Learning and Deep Learning algorithms,” Jul. 2021, doi: 10.1109/IISA52424.2021.9555544.

[21] A. Baba, “Advanced AI-based techniques to predict daily energy consumption: A case study,” *Expert Syst. Appl.*, vol. 184, Dec. 2021, doi: 10.1016/j.eswa.2021.115508.

[22] J. Dong-Gyu and L. Soo-Young, “Merging back-propagation and Hebbian learning rules for robust classifications,” *Neural Networks*, vol. 9, no. 7, pp. 1213–1222, Oct. 1996, doi: 10.1016/0893-6080(96)00042-1.

[23] H. Musbah, G. Ali, H. H. Aly, and T. A. Little, “Energy management using multi-criteria decision making and machine learning classification algorithms for intelligent system,” *Electr. Power Syst. Res.*, vol. 203, p. 107645, Feb. 2022, doi: 10.1016/j.epsr.2021.107645.

[24] A. Jozi, T. Pinto, and Z. Vale, “Contextual learning for energy forecasting in buildings,” *Int. J. Electr. Power Energy Syst.*, vol. 136, p. 107707, Mar. 2022, doi: 10.1016/j.ijepes.2021.107707.

[25] L. X. Wang and J. M. Mendel, “Generating Fuzzy Rules by Learning from Examples,” *IEEE Trans. Syst. Man Cybern.*, vol. 22, no. 6, pp. 1414–1427, 1992, doi: 10.1109/21.199466.

[26] O. Cordon, M. J. Del Jesus, F. Herrera, and M. Lozano, “MOGUL: A Methodology to Obtain Genetic Fuzzy Rule-Based Systems under the Iterative Rule Learning Approach*,” doi: 10.1002/(SICI)1098-111X(199911)14:11.

[27] A. Jozi, T. Pinto, I. Praca, F. Silva, B. Teixeira, and Z. Vale, “Energy consumption forecasting using genetic fuzzy rule-based systems based on MOGUL learning methodology,” *2017 IEEE Manchester PowerTech, Powertech 2017*, Jul. 2017, doi: 10.1109/PTC.2017.7981219.

[28] F. Al-Shanableh and A. Evcil, “Prediction of energy consumption of residential buildings in northern Cyprus using fuzzy interference system,” *Energy Build.*, vol. 256, p. 111555, Feb. 2022, doi: 10.1016/J.ENBUILD.2021.111555.

[29] S. Naji *et al.*, “Application of adaptive neuro-fuzzy methodology for estimating building energy consumption,” *Renew. Sustain. Energy Rev.*, vol. 53, pp. 1520–1528, Jan. 2016, doi: 10.1016/J.RSER.2015.09.062.

[30] R. Olu-Ajayi, H. Alaka, I. Sulaimon, F. Sunmola, and S. Ajayi, “Machine learning for energy performance prediction at the design stage of buildings,” *Energy Sustain. Dev.*, vol. 66, pp. 12–25, Feb. 2022, doi: 10.1016/j.esd.2021.11.002.

[31] J. S. Chou and D. S. Tran, “Forecasting energy consumption time series using machine learning techniques based on usage patterns of residential householders,” *Energy*, vol. 165, pp. 709–726, Dec. 2018, doi: 10.1016/J.ENERGY.2018.09.144.

[32] L. Kabari and U. Onwuka, “Comparison of Bagging and Voting Ensemble Machine Learning

Algorithm as a Classifier," *Int. J. Comput. Sci. Softw. Eng.*, vol. 9, pp. 19–23, 2019.

[33] A. Prasad and I. Dusparic, "Multi-agent Deep Reinforcement Learning for Zero Energy Communities," *Proc. 2019 IEEE PES Innov. Smart Grid Technol. Eur. ISGT-Europe 2019*, Sep. 2019, doi: 10.1109/ISGTEUROPE.2019.8905628.

[34] K. P. Amber, R. Ahmad, M. W. Aslam, A. Kousar, M. Usman, and M. S. Khan, "Intelligent techniques for forecasting electricity consumption of buildings," *Energy*, vol. 157, pp. 886–893, Aug. 2018, doi: 10.1016/J.ENERGY.2018.05.155.

[35] N. Javaid, M. Nauman Javid Ghuman, Z. Ali Khan, R. Abid Abbasi, and S. Ur Rehman, "Short Term Load Forecasting Using XGBoost," doi: 10.1007/978-3-030-15035-8_108.

[36] J. Brank *et al.*, "Feature Selection," *Encycl. Mach. Learn.*, pp. 402–406, 2011, doi: 10.1007/978-0-387-30164-8_306.

[37] "BANES Energy Data Electricity - Datasets - Bath: Hacked Datastore." https://data.bathhacked.org/datasets/banes-energy-data-electricity (accessed Apr. 20, 2022).

[38] "Solcast API Toolkit." https://toolkit.solcast.com.au/live-forecast (accessed Apr. 20, 2022).

[39] T. Filik, Ü. B. Filik, and Ö. N. Gerek, "Solar radiation – to – power generation models for one-axis tracking PV system with on-site measurements from Eskisehir, Turkey," *E3S Web Conf.*, vol. 22, p. 00046, Nov. 2017, doi: 10.1051/E3SCONF/20172200046.

[40] "ERSE - Início." https://www.erse.pt/inicio/ (accessed Jun. 16, 2022).

[41] "TARIFAS E PREÇOS DE ELETRICIDADE EM 2022." [Online]. Available: http://www.erse.pt.

[42] "SU ELETRICIDADE." https://sueletricidade.pt/pt-pt/page/601/tarifas-baixa-tensao-normal (accessed Jun. 19, 2022).

[43] S. Pholboon, M. Sumner, E. Christopher, and S. A. Norman, "Real-time battery management algorithm for peak demand shaving in small energy communities," *2015 IEEE PES Innov. Smart Grid Technol. Lat. Am. ISGT LATAM 2015*, pp. 19–24, Jan. 2016, doi: 10.1109/ISGT-LA.2015.7381123.