Solar Generation Forecast from the Perspective of a DSO

Miguel Fonseca Monteiro

Thesis to obtain the Master of Science Degree in Electrical and Computer Engineering

Supervisor: Prof. Rui Manuel Gameiro de Castro

Examination Committee

Chairperson: Prof. Maria Eduarda de Sampaio Pinto de Almeida Pedro
Supervisor: Prof. Rui Manuel Gameiro de Castro
Member of Committee: Prof. João Francisco Alves Martins

November 2017
Acknowledgements

The realization of this work became possible with some contributions. Here, I will express my words of gratitude to those who had an important role to the accomplishment of my master thesis.

First, I would have never reached this point without all the support of my parents and my sister. Their contribution was essential for the conclusion of the master degree and for so many other aspects of my life. I am who I am thanks to them. My most honest words of gratitude are directed for them.

To Professor Rui Castro, for having accepted me to work with him and for the all the guidance, teachings and availability. Thank you.

EDP-Inovação is deeply acknowledged for providing the data that supported the validation of the algorithms developed in this work.

I would like to thank to Carolina, who has always supported me in the good and in bad moments. The motivation she gave me and the joy she transmitted me clearly gave me extra strengths to perform this work.

To my home mates, Luis, Miguel and Samuel, thanks for the friendship, for the support, for the relaxing moments shared.

To my college friends, who have always helped me during the course, for the good times spent and for the companionship.

To Cristina, thank you for the availability to help me to revise the English writing of this work.

To José, for the help and those relieving talks.

To David, a recent friend with who I was able to trade some ideas.
Abstract

The integration of renewable energy into the power system is getting more attention during the last years. The need to find an ecologic alternative to fossil fuels is increasing, and if one wants to preserve the planet Earth one should act as fast as possible. There are several ways of assisting the integration of renewables. The use of artificial intelligence techniques allows the accomplishment of forecasts, what aids the electrical companies to avoid unnecessary penalty payments in the market, for instance. In this work, it will be compared the performance of four AI techniques: artificial neural networks (ANN), adaptive neuro-fuzzy inference systems (ANFIS), random forests (RF) and k-nearest neighbors (kNN) in the task of predicting solar PV power output. The procedure will consist in the implementation of these four methods for three days with different meteorological conditions (a sunny day, a partly cloudy day and a cloudy day), in the RStudio software. After making a one-day-ahead solar PV power forecast, the goal is to conclude about the impact caused by cloudiness and to determine the algorithm with the best performance, including the computation of the mean average percentage error (MAPE) and measuring the simulation time. In the end, kNN clearly outperformed the other techniques for the three tests that were made. A global comparison between all the simulations is made.

Keywords

Solar PV power forecast, artificial intelligence, ANN, ANFIS, random forest, kNN
Resumo

A integração das energias renováveis no sistema elétrico está a ganhar atenção durante os últimos anos. A necessidade de encontrar uma alternativa ecológica aos combustíveis fosseis está a aumentar, e se queremos preservar o planeta Terra devemos agir o mais celer e possível. Há várias formas de auxiliar a integração das renováveis. A utilização de técnicas de inteligência artificial (AI) permitem a realização de previsões, o que ajuda as companhias elétricas a evitar pagamentos de multas desnecessárias, por exemplo. Neste trabalho vai ser comparado o desempenho de quatro técnicas de IA: redes neuronais artificiais (ANN), redes neuro-fuzzy adaptativas (ANFIS), florestas aleatórias e o método dos k-vizinhos mais próximos (kNN) na tarefa de prever a potência solar de saída de um painel fotovoltaico. Este procedimento consistirá na implementação desses quatro métodos em três dias com diferentes condições meteorológicas (dia de sol, dia parcialmente nublado e dia nublado), no software RStudio. Depois de ser feita uma previsão da potência solar fotovoltaica para um dia de avanço, o objetivo é concluir sobre o impacto causado pela nebulosidade e determinar qual o algoritmo com melhor desempenho, incluindo o cálculo do erro médio e medindo o tempo da simulação. No final, o kNN foi superior nos três testes realizados. Uma comparação geral entre todas as simulações vai ser feita.

Palavras-chave

Previsão de potência solar fotovoltaica, inteligência artificial, ANN, ANFIS, floresta aleatória, kNN
# Table of Contents

Acknowledgements ......................................................................................................................... iii  
Abstract ................................................................................................................................................ v  
Resumo ................................................................................................................................................ vii  
List of Figures ........................................................................................................................................ xi  
List of Tables .......................................................................................................................................... xiii  
List of Acronyms ................................................................................................................................... xvi  
1 Introduction ........................................................................................................................................ 1  
  1.1 Overview of the Topic .................................................................................................................... 1  
  1.2 Motivation ...................................................................................................................................... 3  
  1.3 Objectives and Structure of the Thesis ........................................................................................... 3  
2 State of the Art .................................................................................................................................... 5  
  2.1 Introduction ..................................................................................................................................... 5  
  2.2 Solar Irradiance ............................................................................................................................... 7  
  2.3 Numerical Weather Prediction (NWP) ............................................................................................ 8  
    2.3.1 The Evolution of NWP ............................................................................................................. 8  
    2.3.2 Modern NWP Models ............................................................................................................. 9  
  2.4 Forecasting Models ....................................................................................................................... 11  
  2.5 Artificial Intelligence Models ....................................................................................................... 14  
3 Algorithm Description ...................................................................................................................... 19  
  3.1 Artificial Neural Network (ANN) .................................................................................................. 19  
    3.1.1 The Concept of Artificial Neuron ........................................................................................... 19  
    3.1.2 The Evolution of Neural Networks ......................................................................................... 22  
  3.2 Adaptive Neuro-Fuzzy Inference System (ANFIS) ...................................................................... 28  
    3.2.1 Introduction to the Neuro-fuzzy System ............................................................................... 28  
    3.2.2 Description of the Fuzzy System ........................................................................................... 28  
    3.2.3 Adaptive Network ................................................................................................................ 29  
    3.2.4 ANFIS Architecture .............................................................................................................. 30  
  3.3 Random Forest (RF) ..................................................................................................................... 33  
    3.3.1 Ensemble Learning .................................................................................................................. 33  
    3.3.2 The Evolution of Bagging ..................................................................................................... 33  
    3.3.3 Mathematical Framework of Random Forest ........................................................................ 34  
  3.4 K-Nearest Neighbors (kNN) .......................................................................................................... 35  
    3.4.1 First Approach to kNN ........................................................................................................... 35  
    3.4.2 Analog Method ..................................................................................................................... 36  
  3.5 Persistence Method ....................................................................................................................... 37  
4 Simulation Conditions ..................................................................................................................... 39  
  4.1 Dataset and Data Selection ......................................................................................................... 40  
  4.2 Implementation of the Techniques ............................................................................................... 42  
    4.2.1 Neural Network Implementation ............................................................................................ 43  
    4.2.2 ANFIS Implementation ......................................................................................................... 44  
    4.2.3 Random Forest Implementation ............................................................................................ 45  
    4.2.4 kNN Implementation ............................................................................................................ 45  
  4.3 Performance Assessment .............................................................................................................. 46
List of Figures

Figure 2.1. Sun-earth angles ........................................................................................................ 7
Figure 2.2. Spatial resolution and time horizon for different forecasting techniques .......... 15
Figure 3.1. Simplified diagram of a biological neuron .............................................................. 20
Figure 3.2. Artificial neuron ....................................................................................................... 20
Figure 3.3. Step function (unipolar and bipolar, respectively) .................................................. 21
Figure 3.4. Linear function (unipolar and bipolar, respectively) ............................................... 21
Figure 3.5. Sigmoid function (unipolar and bipolar, respectively) .......................................... 21
Figure 3.6. Representation of a simple perceptron .................................................................... 23
Figure 3.7. Multi-layer ANN ..................................................................................................... 24
Figure 3.8. Multi-layer perceptron with BPL ........................................................................... 25
Figure 3.9. Fuzzy Inference System ......................................................................................... 29
Figure 3.10. Type 3 fuzzy reasoning ......................................................................................... 30
Figure 3.11. ANFIS structure for Type 3 fuzzy reasoning .......................................................... 31
Figure 3.12. kNN classification ................................................................................................. 36
Figure 4.1. Time evolution of the irradiance for the 4th of January ........................................... 41
Figure 4.2. Time evolution of the irradiance for the 5th of January .......................................... 41
Figure 4.3. Time evolution of the irradiance for the 3rd of August ........................................... 42
Figure 5.1. Results obtained for NN for the 4th of January ...................................................... 51
Figure 5.2. Results obtained for ANFIS for the 4th of January ................................................ 52
Figure 5.3. Results obtained for RF for the 4th of January ....................................................... 53
Figure 5.4. Results obtained for kNN for the 4th of January .................................................... 54
Figure 5.5. Results obtained for NN for the 5th of January ...................................................... 57
Figure 5.6. Results obtained for ANFIS for the 5th of January ................................................ 58
Figure 5.7. Results obtained for RF for the 5th of January ....................................................... 59
Figure 5.8. Results obtained for kNN for the 5th of January .................................................... 60
Figure 5.9. Results obtained for NN for the 3rd of August ....................................................... 63
Figure 5.10. Results obtained for ANFIS for the 3rd of August ............................................... 64
Figure 5.11. Results obtained for RF for the 3rd of August ...................................................... 65
Figure 5.12. Results obtained for kNN for the 3rd of August .................................................... 66
Figure 5.13. Time evolution of the PV power output for the time scale change test ................ 69

Figure A.1. Specifications of the manufacturer of the PV panel (1) ............................................ 82
Figure A.2. Specifications of the manufacturer of the PV panel (2) .......................................... 83
Figure B.1. Power evolution for the 4th of January for 10, 20, 40 and 50 cycles (NN) ............. 86
Figure B.2. Power evolution for the 4th of January for 10, 20, 30 and 40 epochs (ANFIS) ... 87
Figure B.3. Power evolution for the 4th of January for 2000, 3000, 4000 and 5000 (RF) ....... 88
Figure B.4. Power evolution for the 4th of January for 5, 10, 15 and 20 nearest neighbors (kNN) .......................................................................................................................... 89
Figure B.5. Power evolution for the 5th of January for 10, 20, 40 and 50 cycles (NN) ............ 90
Figure B.6. Power evolution for the 5th of January for 10, 20, 30 and 40 epochs (ANFIS) ... 91
Figure B.7. Power evolution for the 5th of January for 2000, 3000, 4000 and 5000 (RF) ........... 92
Figure B.8. Power evolution for the 5th of January for 5, 10, 15 and 20 nearest neighbors (kNN)................................................................................................................ 93
Figure B.9. Power evolution for the 3rd of August for 10, 20, 40 and 50 cycles (NN)................. 94
Figure B.10. Power evolution for the 3rd of August for 10, 20, 30 and 40 epochs (ANFIS) ....... 95
Figure B.11. Power evolution for the 3rd of August for 2000, 3000, 4000 and 5000 (RF)......... 96
Figure B.12. Power evolution for the 3rd of August for 5, 10, 15 and 20 nearest neighbors (kNN)........................................................................................................................................ 97
List of Tables

Table 3.1. Passes of the hybrid algorithm ................................................................. 32
Table 5.1. Final results for the 4th of January .................................................................. 55
Table 5.2. Comparison of the panel efficiency for the first test ........................................... 56
Table 5.3. Overall results for the 5th of January ............................................................... 61
Table 5.4. Comparison of the panel efficiency for the second test ...................................... 62
Table 5.5. Overall results for the 3rd of August ................................................................. 67
Table 5.6. Comparison of the panel efficiency ................................................................... 68
Table 5.7. MAPE comparison for the time scale change test .............................................. 70
Table 5.8. Simulation time for different number of cycles (s); ANN ................................... 71
Table 5.9. MAPE for different number of cycles (%); ANN .................................................. 71
Table 5.10. Simulation time for different number of epochs (s); ANFIS .............................. 72
Table 5.11. MAPE for different number of epochs (%); ANFIS ......................................... 72
Table 5.12. Simulation time for different number of trees (s); RF ..................................... 73
Table 5.13. MAPE for different number of trees (%); RF .................................................. 73
Table 5.14. Simulation time for different values of k (s); KNN .......................................... 74
Table 5.15. MAPE for different values of k (%); KNN ......................................................... 74
# List of Acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADALINE</td>
<td>Adaptive Linear Neuron</td>
</tr>
<tr>
<td>AI</td>
<td>Artificial Intelligence</td>
</tr>
<tr>
<td>ANFIS</td>
<td>Adaptive Neuro-Fuzzy Inference System</td>
</tr>
<tr>
<td>ANN</td>
<td>Artificial Neural Network</td>
</tr>
<tr>
<td>AR</td>
<td>Auto Regression</td>
</tr>
<tr>
<td>ARIMA</td>
<td>Auto-Regressive Integrated Moving Averages</td>
</tr>
<tr>
<td>BPL</td>
<td>Back Propagation Learning</td>
</tr>
<tr>
<td>BVE</td>
<td>Barotropic Vorticity Equation</td>
</tr>
<tr>
<td>CARET</td>
<td>Classification and Regression Training</td>
</tr>
<tr>
<td>DNI</td>
<td>Direct Normal Irradiance</td>
</tr>
<tr>
<td>ECMWF</td>
<td>European Centre for Medium-Range Weather Forecast</td>
</tr>
<tr>
<td>GFS</td>
<td>Global Forecast System</td>
</tr>
<tr>
<td>HISIMI</td>
<td>Historical Similar Mining</td>
</tr>
<tr>
<td>HRRR</td>
<td>High Resolution Rapid Refresh</td>
</tr>
<tr>
<td>HS</td>
<td>Hybrid Systems</td>
</tr>
<tr>
<td>IEA</td>
<td>International Energy Agency</td>
</tr>
<tr>
<td>ISO</td>
<td>Independent System Operator</td>
</tr>
<tr>
<td>kNN</td>
<td>k-Nearest Neighbors</td>
</tr>
<tr>
<td>LSE</td>
<td>Least Square Estimates</td>
</tr>
<tr>
<td>MA</td>
<td>Moving Averages</td>
</tr>
<tr>
<td>MADALINE</td>
<td>Many Adaptive Linear Neurons</td>
</tr>
<tr>
<td>MAE</td>
<td>Mean Absolute Error</td>
</tr>
<tr>
<td>MAPE</td>
<td>Mean Average Percentage Error</td>
</tr>
<tr>
<td>MF</td>
<td>Membership function</td>
</tr>
<tr>
<td>MLP</td>
<td>Multi-Layer Perceptron</td>
</tr>
<tr>
<td>Acronym</td>
<td>Full Form</td>
</tr>
<tr>
<td>---------</td>
<td>-----------</td>
</tr>
<tr>
<td>NCAR</td>
<td>National Center of Atmospheric Research</td>
</tr>
<tr>
<td>NCEP</td>
<td>National Centers for Environmental Prediction</td>
</tr>
<tr>
<td>NOAA</td>
<td>National Oceanic and Atmospheric Administration</td>
</tr>
<tr>
<td>NWP</td>
<td>Numerical Weather Prediction</td>
</tr>
<tr>
<td>PV</td>
<td>Photovoltaics</td>
</tr>
<tr>
<td>REN21</td>
<td>Renewable Energy Policy Network for the 21st Century</td>
</tr>
<tr>
<td>RF</td>
<td>Random Forests</td>
</tr>
<tr>
<td>RMSE</td>
<td>Root Mean Square Error</td>
</tr>
<tr>
<td>RTM</td>
<td>Radioactive Transfer Model</td>
</tr>
<tr>
<td>RUC</td>
<td>Rapid Update Cycle</td>
</tr>
<tr>
<td>SVM</td>
<td>Support Vector Machines</td>
</tr>
<tr>
<td>TLU</td>
<td>Threshold Logic Unit</td>
</tr>
<tr>
<td>WRF</td>
<td>Weather Research and Forecast</td>
</tr>
<tr>
<td>WT</td>
<td>Wavelet Transform</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

Since the origin of mankind that there is one indispensable and considered practically infinite source of energy: the sun! The sun has always been assuring the survival of the human being. Solar energy was the cornerstone for the primitive societies, providing, directly or not, ways to subsist and to also to evolve. These societies have been developing themselves and have always attempted to build new technologies that enable the profitable use of the solar energy.

1.1 Overview of the Topic

The Industrial Revolution, during the XVIII and XIX centuries, brought significant changes in industrial processes and the recognition of the coal as the most important fossil fuel. Sources of energy such as coal, oil and natural gas existed in high quantity and were cheap, which held the increasing energetic demand, in what concerns to heat, electric and fuel needs. Lately, during the 80s, an oil crisis was another turning point. The arise of problems such as the awareness of the limited nature of fossil fuels
and the environmental impact caused by its combustion, led to an extraordinary increase of the prices. More and more emerged the necessity to take advantage of solar energy, not only in economic terms but also of the well-being of our planet. Currently, 75% of all the consumed energy in the world derives from those three fossil fuels [1]. The self-described social engineer Jacques Fresco, once said “Earth is abundant with plentiful resources. Our practice of rationing resources through monetary control is no longer relevant and is counter-productive to our survival.” (João Teixeira, Impacto do Erro da Previsão Eólica nas Necessidades a Longo Prazo da Reserva Operacional, 2015), which is very pertinent relatively to the underlying issue.

Therefore, it comes the need to replace that kind of primary energy for a more sustainable one. The solar radiation has a tremendous potential and it is considered the major renewable energy source, being the oldest and the most modern energetic source at the same time. Renewable energy sources are considered sustainable sources of energy once its existence is not affected by its continued use, because of the low emission of polluting gases, thanks to the symbiosis with the environment and due to the fact of being practically endless. Having this in mind, probably the main challenge of today is to develop new technologies that can be as efficient as possible in order to take advantage of that kind of energy.

Let us make a short overview about the actual global situation of renewable energy. According to REN21, during the last year, numerous renewable energy technologies had noticed a decrease in its prices and also there was a higher focus given to energy storage [2]. According to the IEA, in European Countries, in 2015, there was an equal investment in research and development for energy efficiency, renewables and nuclear power.

There has been an annual growth of the world primary energy demand, in which developing countries contributed the most. On the other side, the rise of CO₂ is lower than the years before, mainly due to the reduction of the global use of coal and also derived from the continuous improvement in the utilization of renewables. Renewable energy represents approximately 20% of the global energy consumption, in the last two years. The major cause of this recent increase in renewable’s capacity comes from solar photovoltaics (PV), while hydropower represents the highest value of renewable power capacity and generation. Just to have an idea of how huge was the rise of solar PV, in 2016, there was an increase in power capacity what can be equivalent to 31.000 solar panels installed every hour. Cheaper PV modules, lower capital expenditures and better capacity factors are turning solar PV energy into one of the most viable renewable energy to invest in, what is bringing new competitors into the industry.

In the next sections, there will be a description of the motivation that led to the choice of this thesis’ topic. In the end, the objectives and the structure will be approached.
1.2 Motivation

As referred before, solar energy is considered the most actual and important renewable energy source. It is this kind of source in which is being made the biggest investment and that shows more potential to deal with actual energetic and also environmental problems. There are two main purposes for the conversion of the energy related to solar radiation: to produce electricity and to obtain thermal energy, in the heat form. The focus of this work will be the process of obtaining electricity, what is possible through the employment of photovoltaic panels. Unfortunately, the global efficiency of the conversion process is only around 15%, despite the existence of more efficient solar cells. Besides the physical and chemical structure of solar PV cells, there is a particular attention in what concerns to the attempt to improve its profitability through the forecasting of meteorological features. The ability to forecast certain meteorological characteristics helps the companies to plan their actions. Reliable forecasts avoid the operators of the power grid to take risks. If the electric power is not correctly predicted, it may become necessary to compensate that power through backup units, which are particularly expensive. From an economic point of view, the operators also need to previously point out how much energy are they willing to sell at a certain time, otherwise they will have to pay a penalty. It is then necessary to perform a forecast and the most accurate to do that is through the use of Artificial Intelligence (AI) tools as forecasting techniques. Not only in this area and not only for forecasting, these tools are also used in several other fields of application. AI is more and more an actual matter and it seemed a very interesting topic to work in. The use of these type of tools is the best way to reach reliable predictions, so that can be possible to benefit from them. There is a constant attempt to upgrade older and to create new mechanisms that enable to obtain good results.

All of this seemed very stimulating and served as motivation for the accomplishment of this work. Artificial intelligence tools, the integration of renewables, the improvement of the energetic sector and all the advantages brought by these topics led to the choice of this theme.

1.3 Objectives and Structure of the Thesis

There is no doubt about the importance of a reliable forecast. In this work, it will be made a comparison of the results given by four different methods, methods appropriately chosen, which strengths have already been proved in a large number of scientific reports. The AI methods that are going to be used are Artificial Neural Networks (ANN), Adaptive Neuro-Fuzzy Inference Systems (ANFIS), Random Forest (RF) and k-Nearest Neighbor (kNN). A precedent description for each algorithm is going to be made. These algorithms will be implemented in RStudio software to realize a solar PV power forecast. Therefore, in terms of the solar forecasting issue, this thesis report will be a contribution to that. Three days with different conditions (sunny day, partly cloudy day and cloudy day) will be used as the base of the predictions. The intention is to compare the different performances of those methods and to discuss the influence of cloudiness for the problem in question.
This thesis report will be structured in six chapters, which will be described above.

First, one will start by doing an overview about the actual state of renewable energy, namely solar photovoltaic energy, and discussing the companies’ need to have actual reliable forecasts, avoiding eventual penalty payments. Then, two main topics will be approached. NWP models that compute meteorological features through ground based monitoring network and forecasting models, from simple regressive methods to artificial intelligence ones.

In the third chapter, a description of each methodology will be made so that can be understandable the inner mathematical processes. The algorithms will be approached not only in a mathematical view but also exploring an historical point of view.

Fourth chapter serves to introduce the way of implementing each algorithm. The simulation conditions will be exploited, going from the dataset used to the evaluation and selection of the parameters for each technique. This contextualization is fundamental for the interconnection between the theoretical bases of the algorithms and the software that allowed their implementation.

After defining the procedures and implementing all the methods, the results become available. The fifth chapter will present the outcomes of the simulations that were run in the software. The comparison between the real values and the predicted ones will be made, followed by the error assessment and consequent justifications.

Lastly, the conclusion of the whole work will be exhibited.
Chapter 2
State of the Art

2.1 Introduction

Nowadays, we are facing a changing world, where there is the need of replacing fossil fuels by clean ways of producing electricity. Embracing alternative energy sources is more and more the right path to follow. To accomplish that, a great and required improvement in the electrical network is the integration of renewable sources, since there is an increasingly demand for energy. This change will be more positive according to the degree of efficiency of the referred integration. Changing into a more sustainable energy supply not only requires the ongoing introduction of renewable energy but will also bring consequences for the organization, the structure and the management of all levels of electricity
supply systems. Fossil fuels can be forced to generate a certain quantity of power, at any time, in opposite to renewable sources that cannot be adjusted to the actual power demand. The availability of solar and wind energy is determined by the prevailing weather conditions and so it is highly variable. There is a larger investment in these power sources and it is expected that, in 2050, wind energy reaches 12% of the global electricity consumption [3] and PV energy provides 11% [4].

This work is related to solar energy and so it is suitable to approach the topic. Solar energy comes from the sun and it is a source of energy that does not have significant carbon dioxide emissions. For modern societies, solar energy is, with no doubt, the most abundant power resource. The supply of electrical power that derives from photovoltaic systems (PV system) is increasing because there is a reasonable number of large-scale grid-connected PV power plants that have been built recently, and more will come. Its worldwide implementation is influenced by the location, whether there are favourable weather conditions, intra-hour variability and dawn and dusk ramping rates.

Despite of being a fact that is more relevant wind companies, there is a need to deal with fluctuating input from the PV system. Here arises the need of reliable forecast information, so that can be possible an efficient utilization of those fluctuations. This variability turns the power output of a PV plant into a nondeterministic and stochastic process. Some variables, such as the variations of weather conditions, prevent an accurate determination of the total power produced in a specific time interval. What one can say about it, is that a PV power plant might be compared to an uncontrollable and non-dispatchable generator in the utility network. It is, then, extremely important the power output forecast so that it could become possible a correct planning and operation in the power distribution system and also the management of the distribution automation in the power demand side. Another thing to point out is that unexpected variations of the power output directly affect capital and operational costs of the power system, also leading to a decrease in capacity factors. The possibility to make a prediction of the load is assumed as the basis in what refers to the schedule of a power plant and it is important to ensure a reliable grid operation, by helping to plan transactions in the electricity market. An essential goal is to adapt conventional power plants so they can compensate the fluctuations of the renewable power production. The fact is that renewable power systems cannot assure a steady supply of energy.

Utility companies, transmission and distribution system operators, energy service providers, energy traders and also independent power producers reveal a tremendous need for convenient forecasts so they may be able to schedule, dispatch and regulate power. For instance, in order to use a hybrid power system (storage batteries, solar cells, wind generators, and others) an electric company imperatively need a forecast technique. By other words, the profitability of the power system will be affected by the quality of the forecast. At the Independent System Operator (ISO) level, solar forecast allows the optimization in the decision-making through a correction of the unit commitment and out-of-region trader. This means that if power producers cannot fulfil their schedule, they are penalized. More accurate forecasts can reduce penalty risks and are needed by companies to optimize their profits.

The challenge concerned at this point is to develop algorithms able to minimize the prediction errors so that can occur an efficient integration into the grid. The aim is to create inexpensive and easy-to-use techniques. Of course that this is quite struggling because it can be very hard to accomplish.
2.2 Solar Irradiance

Forecasting solar quantities, such as irradiance, is of extreme importance for the development of equipment appropriated for its generation and for a better control of them.

The first and the most important step for almost all power systems is the irradiance forecast. There are lots of studies about solar irradiance forecasting and hence it reveals the need to discuss a little about this subject.

The irradiance beam that comes from of the sun is not the same one that reaches the ground. The existence of atmosphere is the reason for that. Reflections, absorptions and re-emissions cause an attenuation on the intensity of the extra-terrestrial beam. It is known the value of the irradiance incident on the earth’s atmosphere and that value is about 1360 Wm\(^{-2}\), called extra-terrestrial irradiance, \(I_0\) [5]. Due to this effect, it is necessary to separate the irradiance into two components, given by

\[
GHI = DHI + DNI \cdot \cos \theta
\]  

(2.1)

where GHI stands for Global Horizontal Irradiance, DHI for Direct Horizontal Irradiance and DNI is the Direct Normal Irradiance. \(\theta\) is called the solar zenith angle and can be easily understood in figure 2.1

![Figure 2.1. Sun-earth angles (extracted from https://www.esrl.noaa.gov/gmd/grad/solcalc/azelzen.gif)](https://www.esrl.noaa.gov/gmd/grad/solcalc/azelzen.gif)

The DNI can be expressed as a function of two important coefficients, the airmass (AM) and the Linke turbidity (\(T_L\)), both of them well known in the solar engineering field. The first one is about the optical path length that the radiation travels from the atmospheric boundary until it reaches the ground. When there is a zenith angle of 0 degrees, the sun is directly overhead and so airmass has a unit value. According to the paper [6], many models assume that the airmass is a function of the zenith angle and so it is denoted as AM(\(\theta\)). On the other side, to produce a certain irradiance at ground level one could reproduce it by using a certain number of clean and dry atmospheres. This is called Linke turbidity coefficient. Having this, Linke proposed in 1922 that the direct normal irradiance could be expressed as:

\[
DNI = I_o \cdot \exp(-\delta_{\text{cld}} \cdot T_L \cdot AM)
\]  

(2.2)
where $\delta_{cda}$ represents the optical thickness of a clean and dry atmosphere. However, this model has suffered several improvements during the years. Having this reference done, let us focus on the techniques that allow to forecast.

### 2.3 Numerical Weather Prediction (NWP)

One can say that NWP methods derived from the lack of ground resources able to efficiently measure solar radiation. Traditional time series methods for prediction were limited by that incapability and so NWPs seemed to be a valuable alternative for national or even global scale ground based monitoring network.

#### 2.3.1 The Evolution of NWP

The first approach to what is referred as a NWP was conducted in the beginning of the 20th century by a Norwegian scientist called Vilhelm Bjerknes in the paper “The Problem of Weather Forecasting, Considered from the Viewpoints of Mechanics and Physics” [7]. The author studied a fact of extreme importance for a successful forecast, which is still valid nowadays and that is considered as the heart of NWP methodology. According to the author, it is imperative to analyse how the atmosphere changes from one initial time to another state, by knowing the physical laws under that transition. The methodology follows three steps:

- Choose the physical domain;
- Spatial discretization at a desired resolution;
- Application of physical laws of motion and thermodynamics in order to solve the problem on the discrete special grid.

The request to reach the solutions relatively to the physical and thermodynamic equations was of great importance for the evolution of NWP forecasts. Although these equations provided the basis for forecasting, the major struggle was to numerically obtain the solution for a non-linear partial differential equations system.

A few years later, Felix M. Exner, an Austrian meteorologist, used a very simple model based on the assumption of geostrophically balanced flow to perform some reasonably good forecasts. After, between 1916 and 1922, Lewis F. Richardson tried to numerically solve weather forecast equations [8]. However, this method failed due to the imbalance of the initial data chosen. Introducing finite differences into weather forecast equations was the choice of Courant, Friedrichs and Lewy (CFL), in 1928. These mathematicians created the CFL condition in which, given a spatial discretization, it is restricted the appropriated time steps [9]. Firstly, they agreed that it was necessary to have stability in what concerns to the integration of the physical model. Therefore, to accomplish that requirement, they suggested that the order of magnitude of the maximum allowable time step must be equal to the relevant time scales.
in the physical model. This study allowed to conclude that an increase in the spatial resolution prompted a decrease in the time scale of the physical process and in the maximum allowable time step.

The meteorologist J. G. Charney has considered the factor of modern dynamical meteorology and has helped in the simplification of weather equations through the development of filtered models and the quasi-geostrophic assumption. Charney’s work led to a better understanding of atmospheric motions and made possible successful numerical weather forecasts with computer assistance, around 1950 [10]. It is now convenient to introduce the equations that allowed these predictions, the Barotropic Vorticity Equation (BVE). For this model, it is assumed that the atmosphere is considered almost barotropic, what means that speed and direction of the geotropic wind is independent of the height. Let us then make a short mathematical reference to that.

\[
\frac{\partial \zeta_b}{\partial t} + \nabla_h \cdot \left( (\zeta_b + f) \mathbf{v}_\Psi \right) = 0
\]

(2.3)

\[
\zeta_b \equiv \nabla^2_h \Psi_b
\]

(2.4)

\[
\mathbf{v}_\Psi \equiv k \times \nabla_h \Psi_b
\]

(2.5)

where \( \zeta_b \) is the barotropic vorticity, \( \nabla_h \) is the two-dimensional horizontal gradient, \( \mathbf{v}_\Psi \) is the purely rotational two-dimensional wind velocity, \( \nabla^2_h \) is the two-dimensional horizontal Laplacian, \( \Psi_b \) is the barotropic streamfunction, \( k \) is the unit vector normal to geopotential surfaces and \( f \) is the Coriolis parameter, which is given by

\[
f \equiv 2\Omega \sin(\phi)
\]

(2.6)

where \( \Omega \) is the angular frequency of Earth’s rotation and \( \phi \) is referred to the latitude.

It is believed that the first computer assisted NWP model was performed in Aberdeen, Maryland, in 1950, precisely with the use of the BVE and with the Electronic Numerical Integrator and Computer (ENIAC). This achievement was an important step for the improvement of the NWP. Since then more and more investigators have focused their attention in the development of NWP models, by including new ideas and a diverse range of mathematical contents. Some examples are:

- A better understanding of baroclinic instability;
- The enhancement of some primitive equations;
- The establishment of general circulation modelling;
- The progresses related to spectral methods and transform methods;
- The new researches about semi-implicit time differencing;
- The atmospheric initialization.

### 2.3.2 Modern NWP Models

After these initial approaches of NWP models, the increase in computational power made possible to
accomplish more sophisticated methods. The reason for that was the improvement in the resolutions, which were becoming higher, and the possibility to cover larger domains.

Nowadays, NWP models are separated into two key parts according to the dimension of the domain that is being used, global and regional, which will be described and addressed later. NPWs cannot make an exact prediction of the position and the extent of the clouds. Once NWP spatial resolution are approximately 1-20 km, it becomes almost impossible to resolve micro-scale physics regarding to the cloud formation process. This cloud unpredictability is the main reason for the lack of accuracy of a NWP based solar forecast. On the other hand, NWP models allow another diagnosis. The fact of being able to measure atmospheric saturation, or near saturation, can be associated with cloud formation and also with the possibility to assess precipitation. Having this, the capability of NWPs to reasonably predict cloud formation is intrinsically related to solar forecast, yielding many benefits for this topic. Another important feature about NWP is the long time horizon, which goes from 15 to 240 hours. According to some studies [11,12], for time horizons higher than four hours, NWPs have a better accuracy than satellite based models, which turns NWPs into a more appealing option for medium-term and long-term forecasts.

The next paragraphs will describe the two distinct categories of NWP models.

Let us start with the global models. As the name indicates, a global NWP model is such a model that can be used for more than a single region or location. In this context, two models will be described.

The Global Forecast System (GFS), which is one the most famous global NWP models, is a model run by the National Oceanic and Atmospheric Administration (NOAA). GFS is able to forecast until 384 hours, 16 days, on a 28 km × 28 km grid for the global domain and it has a loop time step that varies between 6 to 180 hours [13]. Besides the horizontal discretization, there are also 64 vertical layers of the atmosphere modelled by this system. The GFS uses a Radiative Transfer Model (RTM), whose inputs can be predicted values of a 3D aerosol concentration field, predicted values of a 2D horizontal concentration field of H₂O, O₂ and O₃ and also a 2D concentration field of CO₂ that remains constant. Another relevant point of GFS model is the sophisticated scheme that computes wavelength specific attenuation of diffuse irradiance in both ascending and descending paths [14]. GFS reveals to be sensitive to temperature errors.

The European Centre for Medium-Range Weather Forecast (ECMWF) is a NWP model that has already proven its high quality as a basis for wind and solar power forecasts. ECMWF can predict until 15 days ahead, forecasting solar irradiance and other parameters. Implemented in January 2010, the actual version, T1279, has a spatial horizontal resolution of 16 km × 16 km. There are 91 hybrid vertical layers, what is equivalent to approximately 80 km, which resolve the atmosphere up to 0.01 hPA. During the first three days to be forecasted, the model has a temporal resolution of three hours.

On the other hand, regional NWP models are only applied in a sub-part of the whole global domain. By using smaller domains, it is somehow obvious that both the time horizon and the time step will be smaller too. The resolution in the regional case is reduced in order to allow a better forecast. Depending on each region, whether one is working in America or Europe, for instance, there are diverse models that try to
be as accurate as possible. Let us approach some models.

The Weather Research and Forecast (WRF) model was created in 2004 through a partnership between NOAA and the National Center of Atmospheric Research (NCAR). Since its development, this model has been used by a wide range of communities. WRF aims to perform a mesoscale atmospheric prediction, which assures a close relation between research and operational forecasting communities. Ongoing efforts such as workshops and on-line documentation are the choice to maintain WRF software. WRF is preferred for spatial resolutions until 10 km and it is due its flexibility that is elected for many forecasting applications.

Until May, 2012, the Rapid Update Cycle model (RUC) was the operational NWP model of NOAA/NCEP (National Centers for Environmental Prediction) in the United States. With a horizontal resolution of 13 km × 13 km, it was able to make predictions from 1 hour to 18 hours, using 50 atmospheric layers. The RUC only used the water vapour for its wavelength model, neglecting other atmospheric gases. Derived from the incapability for accurate forecasts, RUC was replaced by RAPid refresh (RAP) model. Thanks to a new rapid update configuration of the WRF model, RAP model became the improved model for NCEP, possessing the same spatial and atmospheric resolutions.

The High Resolution Rapid Refresh (HRRR) model is an atmospheric model run by NOAA that is hourly updated with a resolution of 3 km × 3 km. In June 2009, the HRRR model switched its coverage from 2/3 of the continental region of the United States to the entire region. The initial conditions of this model are determined using the same resolution of 13 km of the RUC/RAP model. Derived from the fact of being updated every hour, the HRRR allows predictions for a shorter time scales with increased resolutions.

### 2.4 Forecasting Models

There has always been an interest in the analysis of time series. In the main context of this thesis, it appears to be pertinent, and also interesting, to make an approach inside the regression methods field, which can be thought as the initial step for the actual forecasting processes.

During the decade of 1920, the statistician George Udny Yule started to show some results about the underlying analysis. Yule lived during rough times and that made more difficult to find applications, but, in 1922, he reached the first of his two approaches for stationary time series analysis, the Moving Averages (MA) which is mentioned in [16] and [5]. Few years later, the linear Auto-Regression (AR) has arisen, which was referred in [18], where Yule wrote about the periodicities of disturbed series, with reference to Wolfer's sunspot numbers. There were some attempts to perform this kind of analysis but, in the 1960s, Box and Jenkins revolutionized time series analysis. They were motivated by the principle of parsimony and so they developed two models with a world-wide recognition. For stationary time series, the Auto-Regressive Moving Averages (ARMA) and the Auto-Regressive Integrated Moving Averages (ARIMA) attempted to solve non-stationary processes. All of these methods are considered a
base for a large variety of more complex algorithms. ARMA and ARIMA find not only application for solar forecasts but also in a diversity of other fields, such as economic and business planning and for production matters as it was referred in [19]. Let us make a quick summary about each approach.

An AR models is a model which process of reaching a value is through a finite and linear combination of the previous ones and a single shock \( \omega_t \). So, as the name indicates, there is a regressive process on the previous values. The AR process of order \( m \) can be written as

\[
\bar{z}_t + \phi_1 \bar{z}_{t-1} + \phi_2 \bar{z}_{t-2} + \ldots + \phi_m \bar{z}_{t-m} = \omega_t
\]

(2.9)

where \( \{ \bar{z} \} \) are stochastic portions of the time series and \( \{ \phi \} \) are the \( m + 2 \) unknown parameters of the process, which are computed from the data. One can also define this method as a transfer function of order \( m, \Phi_m(q) \), such as

\[
\Phi_m(q) = \sum_{k=0}^{m} \phi_k q^{-k}
\]

(2.10)

Notice that it implicit from the above equation that

\[
\bar{z}_t = \Phi^{-1}_m(q) \omega_t
\]

(2.11)

and one can look at AR(m) technique as linear filter with transfer function \( \Phi^{-1}_m(q) \) with \( \omega_t \) being the white noise as the input.

In opposite to AR models that perform a weighted sum of previous values, the MA model \( \{ \bar{z} \} \) computes a finite sum of \( n \) previous shocks \( \omega_t, \omega_{t-1}, \omega_{t-2}, \ldots, \omega_{t-n} \). So, the MA process of order \( n \), MA(n), can be expressed as

\[
\bar{z}_t = \omega_t + \theta_1 \omega_{t-1} + \theta_2 \omega_{t-2} + \ldots + \theta_n \omega_{t-n}
\]

(2.12)

Similarly to what was done for AR, the transfer function comes as

\[
\Theta_n(q) = \sum_{k=0}^{n} \theta_k q^{-k}
\]

(2.13)

And so the output can be written as

\[
\bar{z}_t = \Theta_n(q) \omega_t
\]

(2.14)

The MA method can be seen as a way to obtain the output \( \bar{z} \) by passing through a linear filter with a transfer function \( \Theta_n(q) \) and the input being the white noise \( \omega_t \). There are some undetermined parameters that should be determined from the data too. Contrasting with the previous model, MA models are unconditionally unstable.

The need for more useful models led to the creation of ARMA models, which came from the good relationship between MA weights and AR weights. With reference to the equations (2.11) and (2.14), it
is possible to rearrange them and obtain the following result.

$$\Theta(q)\Phi(q)\tilde{z}_t = \Theta(q)\omega_t = \tilde{z}_t$$

(2.15)

One can conclude that

$$\Phi^{-1}(q) = \Theta(q)$$

(2.16)

which means that each set of weights can be obtained from the knowledge of the other. ARMA models look for a parsimonious parametrization, and so the need to use both AR and MA terms for its accomplishment. The final model of the mixed Auto-Regressive Moving Averages of order (m,n) can be understood as a linear filter with a transfer function that corresponds to the ratio between $\Theta_n(q)$ and $\Phi_m(q)$, where the input is the white noise, $\omega_t$.

$$\tilde{z}_t = \frac{\Theta_n(q)}{\Phi_m(q)}\omega_t = \frac{1 + \theta_1q^{-1} + \ldots + \theta_nq^{-n}}{1 + \phi_1q^{-1} + \ldots + \phi_mq^{-m}}\omega_t$$

(2.17)

Normally, the order of the polynomials are less or equal to two as it was stated in [7,8].

When talking about non-stationarity, ARIMA arises as one of many solutions for the problem. Besides the possibility of having a stationary character in some sections, non-stationary processes do not fluctuate about a static mean. One action that can lead to the modulation of a non-stationary process is to particularize an appropriate difference as stationary.

When the roots of $\Phi_m(q) = 0$ lie outside the unit circle, an ARMA(m,n) is considered stationary. However, for roots inside the unit circle, one is towards a non-stationary behaviour. Let us assume that $d$ of the roots of $\Phi_m(q) = 0$ lie on the unit circle and the others are outside. For the referred situation the model can be described as

$$\Phi_m(q)\tilde{z}_t = \Theta_n(q)(1 - q^{-1})^d\tilde{z}_t$$

(2.18)

where $\Phi_n(q)$ is an AR(m) operator that is stationary and invertible and $q$ is a delay operator. Knowing that $S^d\tilde{z}_t = S^d\tilde{z}_t$, it is possible to write, for $d \geq 1$ :

$$\Phi_m(q)S^d\tilde{z}_t = \Theta_n(q)\omega_t$$

(2.19)

In order to reach a more explicit model, let us consider that $y_t = S^d\tilde{z}_t$ and so equation (2.19) gets the form below

$$\Phi_m(q)y_t = \Theta_n(q)\omega_t$$

(2.20)

So, the $d$th difference of a time series can be regarded as a stationary ARMA(m,n) process. Introducing a new variable $d$, number of differences, it is produced an ARIMA(m,d,n) model. The quantity $d$ is relatively small, not being bigger than two [19]. Again, making the analogy, the representation of the
model may be compared to a linear process with input $\omega_t$, the white noise, a linear filter with the transfer function $\Phi_m^{-1}S^d\Theta_n$ and the output $\tilde{z}_t$. The whole ARIMA$(m,d,n)$ process is also described in the book [18].

2.5 Artificial Intelligence Models

Non-linearity has always caused troubles to researchers of a diverse range of fields. The better you deal with non-linear issues the better the results will be. So, what is the best way to do it? Well, the question can be easily answered with two words: Artificial Intelligence. AI techniques started to be developed in the 1950s and have been gaining more attention in the latest decades, in a wide range of areas and applications. Nowadays, there are a huge number of AI methods, each one with its own strengths and weaknesses. The choice of an AI technique thrives according the need of the user, the type of problem, the extent of the data, and so on. In this section, it will be made a reference to some other techniques, beside the ones that will be described next chapter. However, a special attention will be given to hybrid methods that seem to be more interesting, useful and more actual.

Let us start by referring a model that forecasts the electric energy production in a photovoltaic plant, in short-term, called Historical Similar Mining (HISIMI) [21]. This technique explores historical data from previously forecasted values of weather variables and electric power to then search for the degree of similarity with current situations. Data mining techniques are applied and, in the end, a genetic algorithm is used for optimization. HISIMI supplies new information about PV plants. Besides making a spot forecasting, this model is also able to predict the associated uncertainty, which is useful for the analysis and the evaluation of the risk in the electricity markets.

Another proposed method was the one in [22], which aims to make a one-day-ahead PV power output forecast. Through the implementation of the principle of Support Vector Machines (SVM) and a weather classification approach, this method can reach promising results. The main idea of the classification is to split the data according to the weather conditions, i.e., to divide that data into four groups: clear sky day, cloudy day, foggy day and rainy day. The dataset contains weather forecasting data and actual historical power output data of a PV station in China, where the method was applied. Then, four SVM methods are set up according to the algorithm principle and the characteristics of each dataset.

Derived from the need to expand the bounds of forecasting and to improve the quality of the predictions, the idea of using not one but two or more methods have grown. In order to produce more accurate predictions, it started to been used a mix of different methods, which are then called hybrid systems (HS). This is a truly efficient way to increase the quality of solar features forecasts, because one can grab the best of each algorithm and combine it with another one to build a stronger technique. By adding the strengths of two or more methods, it is possible to reach better results. Creating HS also lead to an increase in the range of the spatial resolution and the time horizon. In figure 2.2, it is shown the referred
ability and other areas of interest where HS can span.

![Image](image.png)

Figure 2.2. Spatial resolution and time horizon for different forecasting techniques (extracted from [55])

In the same figure, the arrows show where HS can spread, to areas in which a single methodology cannot reach, and it is also shown the improvement in NWP by a future progress of WRF models. From the analysis of the image, it is clear that stochastic and artificial intelligence methods have a large time horizon coverage. This is due the fact they rely on observational time series and so the coverage is almost only dependent on how the data acquisition is processed, according to the sampling frequency used. For instance, having a sampling frequency of 2 Hz lead to an equal bound on temporal resolution and, consequently, would be suitable to choose the order of seconds as the forecast horizon. The time horizon of a prediction is intrinsically related to the capacity for acquiring data and the sampling frequency of that acquisition. Another significant point is to have the same order of magnitude between the time horizon and the temporal resolution of the data. However, sometimes, wanting to increase too much the temporal extrapolation can cause a reduction in the forecasting quality.

In the next paragraphs, it will be made a brief description of some hybrid systems that were chosen accordingly to the main purpose of this work. For other words, those hybrid methods were picked because of their context with respect to the algorithms applied in this project and because of their relation with the adjacent solar prediction topic.

Recently, it has been introduced a new concept that helps to deal with uncertainty. The Wavelet Transform (WT) is a way to isolate the various fluctuations and other non-stationarities that characterize solar PV power data series. A combination between WT and a few types of neural networks was conducted in [23] for a one-hour-ahead power output forecasting. It was possible to demonstrate the good response against ill-behaved PV power time series data through the filtering property of WT. The study concluded that the proposed hybrid model of WT with a radial basis function neural network (RBFNN) obtained lower error values.
The blending of meteorological data from different sources is the main ensemble technique in what concerns to wind and solar power forecasting. An example of that is reported in [24], where it was proposed a new method that applies the concept of SVM with random forest. The idea is to generate forecasts from SVM and then an ensemble learning method is applied to allow a combination of the forecasts by the random forest. The random forest incorporates solar power forecasts from some models, and the related meteorological data, in order to increase the precision of short-term forecasts. A stable performance is reached through the usage of associative rules in the ensemble method which come from an intelligent weighting approach. It is also interesting to notice that the combination of different forecasts, present and past forecasts, leads to more accurate forecasts during the year.

There has been an increasingly attention in what concerns to hybrid models using ARIMA and ANFIS, most of all because they have been successfully implemented and provided good results. At this point, it is not necessary to describe none of the referred methods because ARIMA was already described this chapter and ANFIS will be deeply discussed next chapter. Let us make a short list of applications that regards to this problem. With respect to an ANFIS hybrid method, Azadeh et al. proposed a monthly electricity demand forecasting model, for Iran, which actually performed better than time series models and neural networks, for example [25]. Another example, and despite its complexity, was the good performance produced by Li et al. with a genetic-ANFIS hybrid model used to predict daily energy demand of a hotel [26]. An improvement can be made in neural networks by adding the ARIMA concept. One of those implementations was carried by Babu and Reddy, who proposed a hybrid ARIMA-ANN model that differs from other models due to the fact of using a moving-average filter [27]. The variables used for this work were a sunspot data, electricity data and stock market data. The referred paper made a comparison between the proposed hybrid model against individual ARIMA and ANN models and also against other existing hybrid ARIMA-ANN models.

To end this short list, here comes a project led by Portuguese researchers related to wind power forecasting [28]. This work derived from the creation of new wind farms and the rise of the adjacent capacity, arising then the need for well-performed forecasts. They proposed an evolutionary-adaptive algorithm for short-term wind power prediction. The success of this hybrid concept came from the combination of mutual information, WT, evolutionary particle swarm optimization and ANFIS. Good results were obtained and it was considered an advance over previous methods. Li and Hu developed a more similar method for time series forecasts, applying an ARIMA-fuzzy system model [29]. This model initially produces fuzzy rules for input and output data by imposing a Sugeno fuzzy model. Then, the ARIMA model integration occurred in the answer part of the fuzzy rules. The whole process led to good results.

In order to deal with the lack of data and, therefore, the incapacity of reliable energy demand forecasts in developing economies, Barak and Sadegh proposed three different solutions in their paper [30]. These authors predicted the annual energy consumption in Iran through three patterns. In the first one, they implemented the ARIMA model for four input features and then six ANFIS structures are held to forecast the nonlinear residuals. For the second pattern, it is performed a prediction by six different ANFIS structures in which the inputs are the forecasting result of ARIMA and other input features. In
this case, the accuracy is improved due to the use of ARIMA’s output as one of the input of ANFIS. At last, the third pattern is similar to the second one, but for this case, it is applied with Adaptive Boosting, known as AdaBoost, data diversification model as an attempt of dealing with missing or insufficient data. The conclusion was that one can obtain better results by implementing two methods with good forecasting strength, which is the case of ARIMA, strong for linear data, and ANFIS, with recognized strength against nonlinearity. In the end, the best results came from the third pattern, in which the use of AdaBoost increases the performance. The whole idea of this work was to forecast in lack of data situations and it was successfully accomplished.
Chapter 3

Algorithm Description

This chapter has the objective of making a theoretical overview of the implemented methodologies, describing the inner mathematical equations and the properties of each algorithm. It seems interesting to do a short historical approach previously, leading to an easier comprehension and also facilitating the interaction with the techniques.

3.1 Artificial Neural Network (ANN)

3.1.1 The Concept of Artificial Neuron

Artificial Neural Network is a learning algorithm that tries to make a computational approach to the
human brain cells. In the human body, there are more than 100 billion nerve cells, known as neurons, which function is to transmit information in order to realize some body tasks. For instance, if the human body touches something, there is a signal that is sent to the brain and then the brain will react, according to the type of stimuli. The response of the brain is also carried by the neurons. Let us make a brief description of biological neurons so one can compare those with the artificial ones. In figure 3.1, it is showed a representation of a natural neuron.

![Figure 3.1. Simplified diagram of a biological neuron (extracted from https://online.science.psu.edu/bisc004_activewd001/node/1907)](image1)

The communication between one cell, other cells and the environment is allowed by the dendrites, which receive and send diverse signals. Dendrites are responsible to accept input signals and, on the other hand, the axon transmits output signals. That output information, generated by each neuron, is transmitted through that canal for other neurons. There must be an impulse so that the neuron can send that information. This is because the cell is able to maintain an electrical potential gradient by an active regulation of some substances across the membrane. When this potential goes higher than a certain threshold, occurs the action potential and so the synapse happens at the end of the axon. So, one say this is the “firing” process of a neuron. In figure 3.2 it is represented the artificial neuron in order to make the analogy with the biological one.

![Figure 3.2. Artificial neuron (extracted from https://inspirehep.net/record/1300728/plots7)](image2)
Similarly to the natural neuron, this computational approach receives the inputs, \( \{x_1, \ldots, x_m\} \), and attributes different weights, \( \{w_1, \ldots, w_m\} \), to each one according to the importance of each input to make the neuron to fire. These input weights can be described as the storage of knowledge or as the memory of the network, which are adjusted during the process of creating the network. The value \( b \) is called bias and is concerned to adjust the threshold. As one can notice in figure 3.2, there is this function, which is a transfer function. The aim of this function is to model the firing rate of the action potential, which can assume different forms. Let us just make a reference about the transfer functions that have been used and the respective forms.

- The step function, unipolar (between 0 and 1) and bipolar (between -1 and 1).

![Figure 3.3. Step function (unipolar and bipolar, respectively) (extracted from [56])](image)

- The linear function, unipolar and bipolar like the step function, which allows the neuron to perform a linear regression.

![Figure 3.4. Linear function (unipolar and bipolar, respectively) (extracted from [56])](image)

- The sigmoid function. The unipolar (logistic) function is very useful and is normally adopted due its monotonicity, continuity and differentiability. The bipolar one (hyperbolic tangent) is quite similar but permits the symmetry of the outputs instead.

![Figure 3.5. Sigmoid function (unipolar and bipolar, respectively) (extracted from [56])](image)

Commonly, the sigmoid transfer function is used because it has better mathematical properties for the problem [31].
The output, before going into the transfer function, adopts the form showed in equation (3.1).

\[ y = b + \sum_{i=1}^{m} w_i x_i \]  

(3.1)

whose variables have the same meaning as in figure 3.2.

Assuming that a sigmoid logistic function was used, the final output of an artificial neuron, \( z \), will be expressed as

\[ z = \frac{1}{1 + e^{-(b + \sum_i w_i x_i)}} \]  

(3.2)

After making the relation between a biological and artificial neuron, let us discuss a little about the first approaches of the neural network.

### 3.1.2 The Evolution of Neural Networks

A neural network has not always been as it is known today. It has suffered some transformations since the first time someone tried to accomplish something like that.

The first formulation of an artificial neuron was conducted by McCulloch and Pitts in the year 1943 [32]. It was named Threshold Logic Unit (TLU) and it made an initial approach to the human brain cell. The implications related to the problem were dealt using recursive functions and the authors integrated that process inside of the activity of nervous networks. The unipolar step or Heaviside step function, like the one in figure 3.3, was the chosen transfer function to perform the TLU. In their work, the authors made five physical assumptions associated to their neuron:

1. The activity of a neuron is an “all-or-none” process.
2. A certain fixed number of synapses must be excited within the period of latent addiction in order to excite a neuron at any time, and this number is independent of previous activity and position of the neuron.
3. The only significant delay within the neural system is synaptic delay.
4. The activity of any inhibitory synapse absolutely prevents the excitation of the neuron at that time.
5. The structure of the network does not change with time.

It is obvious that these assumptions cannot be entirely in agreement with current neural networks, once they were formulated in the beginning of its conception. For instance, the last assumption is not applied nowadays because one know that the structure of an artificial neural network is constantly changing.

Few years later, Frank Rosenblatt developed the simple perceptron, which is truly one of the first neural networks [33]. Rosenblatt's intention was to build a machine, not properly a program. He intended to create a system that could trigger appropriated responses according to the class of the inputs. Therefore, as one can see in figure 3.6, there are two layers. The first one, the input layer, used the
unipolar linear transfer function, and the values of the response layer are reached by adjusting weights on that connection. Going against the assumption of McCulloch and Pitts, at this point, the network reacts when there is an undesired response. One can say it has a supervising learning process that corrects the weight values in the previous case. In ref. [33] it is explained how the weights are updated and how it leads to the convergence of the method.

The next step was the development of the Adaptive Linear Neuron (ADALINE) by Widrow and Hoff [34]. As the name indicates, it is a logic neuron, not a network. ADALINE goes further than the perceptron and employed a bipolar step function, which makes the output to be +1 or -1. This is mainly because the inputs are normally bipolar. The authors’ principal innovation was the Least Mean Square (LMS) training rule. This rule provides an adaptive learning ability for the weights of each connection of each input. The LMS training rule truly converges when the number of training sets tends to infinite.

![Figure 3.6. Representation of a simple perceptron (extracted from [56])](image)

After almost two decades, researchers returned to the neural computing idea. This loss of interest was due to the need for non-linear separability in computing mapping problems. Solving mapping problems that require linear separability did not seem difficult for the simple perceptron. However, for more demanding problems it was not enough. Therefore, after this small break in the neural research and derived from the incapability of solving some problems, the multi-layer network was created. Widrow and Winter developed MADALINE that was a set of Many ADALINEs. The novelty in the architecture was the introduction of additional layers, as it can be seen in figure 3.7. These layers are called hidden layers because, as the name indicates, it is not possible to see partial outputs, i.e., there is no way of training those inner nodes. The interest in neural computing was then renewed.
There were previous attempts to implement a multi-layer neural network but only in the 1980s it was significant. Derived from the initial work of some researches, such as Sprecher, Funahashi, Hornik and Cybenko, it was showed that for a continuous non-linear function on a finite closed interval it was possible to make an approximation by using a multi-layer ANN with a certain number of hidden layers.

Besides the concept of MADALINE, Widrow and Winter also introduced a new training algorithm in which the weights were changed at each training set in order to minimize the error according to the principle of minimal disturbance. This principle’s idea is to minimize the error avoiding to produce disturbances in the network. Notice that, when the response is not the wanted, at least one of the neurons of the first layer is giving a wrong vote. Therefore, the overall idea is to make minimum changes to the weights of the network. This algorithm is known as the MADALINE adaptive Rule (MR) and has been refined for three times. The description of the algorithm is not of interest for the studying case.

Another multi-layer network is the Multi-Layer Perceptron (MLP). Having the ability to execute arbitrary non-linear mappings, this ANN architecture is one of the most implemented. MLP applies the Back Propagation Learning (BPL) method and it is often known as Multi-Layer Feed Forward (MLFF). The idea comes from the forward flow of information related to the adjustment of the weights, which is done backwards, that is why those denominations. Despite the similarity, BPL has an advantage against LMS and perceptron learning rules, which is to allow the adjustment of the weights of hidden layers. This problem is overcome through partial derivatives of errors that are propagated backwards. A scheme of MLP with this algorithm is presented in figure 3.8. Besides other previous efforts, David Rumelhart and his team from the University of San Diego are credited for having developed BPL algorithm, in 1985 [35].
The BPL algorithm is one of the most used for error minimization. The majority of the actual neural network uses BPL algorithm and so let us make an approach of how the error is computed and how the weights are updated. In ref. [35], it is made a description of the generalized process for an arbitrary number of hidden layers. One will denote the weight that connects the $i^{th}$ input neuron with the $j^{th}$ hidden layer neuron by $w_{ij}$ and $w_{jk}$ for the weight between the $j^{th}$ neuron of the hidden layer and the $k^{th}$ output neuron. The input for the hidden and output layers’ neurons are, respectively, defined as

$$H_j = \sum_{i} w_{ij} x_i$$  \hspace{1cm} (3.3)$$

and

$$O_k = \sum_{j} w_{jk} y_j$$  \hspace{1cm} (3.4)$$

where $x_i$ and $y_j$ are the input and output values, as one can see in figure 3.8.

Then, each of those values pass through its own differentiable transfer function, for example a sigmoid function, the output for each layer is written as

$$z_j = f(H_j)$$  \hspace{1cm} (3.5)$$

and

$$z_k = f(O_k)$$  \hspace{1cm} (3.6)$$

So, the idea for a training process is to reduce the total system error and this is achieved through an adjustment of the weights. The global system error is calculated using the following formula.

$$E_{sys} = \frac{1}{N} \sum_{n=1}^{N} E_n$$  \hspace{1cm} (3.7)$$

Figure 3.8. Multi-layer perceptron with BPL (extracted from [55])
\( E_n \) is the mean square error and it is computed by

\[
E_n = \frac{1}{2} \sum_{n=1}^{N} (d_k - z_k)^2
\]  

(3.8)

where \( d_k \) is the target value and \( z_k \) is the final output, as can be seen in figure 3.8.

The error update follows a negative gradient. Thus, for the \( k+1^{th} \) iteration, one have

\[
\Delta W^{(k+1)} = -\alpha \frac{\partial E_n}{\partial W^{(k)}},
\]  

(3.9)

where \( \alpha \) is the learning rate coefficient.

At this point, one must repeatedly apply the chain rule to explain how the update rule works for the system parameters. Knowing the output for the \( k^{th} \) unit let us express

\[
\frac{\partial E}{\partial z_k} = -(d_k - z_k)
\]  

(3.10)

and

\[
\frac{\partial z_k}{\partial I_k} = f'(I_k)
\]  

(3.11)

where \( I_k \) represents \( \sum_j w_{jk} z_j \), in order to reach equation (3.12):

\[
\frac{\partial E}{\partial I_k} = \frac{\partial E}{\partial z_k} \frac{\partial z_k}{\partial I_k} = -(d_k - z_k) f'(I_k)
\]  

(3.12)

For an easier understanding of the update rule, it is defined a generalized delta term, \( \delta_k \), which represents

\[
\delta_k = (d_k - z_k) f'(I_k)
\]  

(3.13)

and so comes the rule for updating the output units as

\[
\Delta w_{jk} = -\alpha \frac{\partial E}{\partial w_{jk}} = \alpha \delta_k y_j
\]  

(3.14)

In analogy with this reasoning, it is equally applied for the hidden layer weights. The purpose is, again, to find an expression for the update rule

\[
\Delta w_{ij} = -\alpha \frac{\partial E}{\partial w_{ij}} = -\alpha \frac{\partial E}{\partial H_j} \frac{\partial H_j}{\partial w_{ij}}
\]  

(3.15)

where all the variables were already defined. In this case, it is also needed to apply the chain rule in the
right side of the previous equation. So, for the first term comes as

$$\frac{\partial E}{\partial H_j} = \frac{\partial E}{\partial z_j} \frac{\partial z_j}{\partial H_j} = \frac{\partial E}{\partial z_j} f'(H_j)$$

(3.16)

and, then, for the second one as

$$\frac{\partial H_j}{\partial w_{ij}} = \sum_i \frac{\partial}{\partial w_{ij}} (w_{ij} x_i) = x_i$$

(3.17)

Having this, one can express the output for node $z_k$ as

$$z_k = f(I_k) = f\left(\sum_j w_{jk} z_j\right)$$

(3.18)

And so, in equation (3.16), one are able to perform directly the differentiation of $\partial E / \partial z_j$. Substituting this expression in the error formula, equation (3.8), the result is

$$\frac{\partial E}{\partial z_j} = \frac{1}{2} \sum_k \frac{\partial}{\partial z_j} (d_k - f(\sum_j w_{jk} z_j))^2 = -\sum_k (d_k - z_k) f'(I_k) w_{jk}$$

(3.19)

As done before, let us introduce another generalized delta operator such as

$$\delta_j = f'(H_j) \sum_k \delta_k w_{jk}$$

(3.20)

Finally, for the hidden layer, the update rule has the form

$$\Delta w_{ij} = \alpha \delta_j x_i$$

(3.21)

The updating of the weights is similar for both the hidden layer and the output layer as one can see in equation (3.22) and (3.23), respectively.

$$w_{ij}^{(n+1)} = w_{ij}^{(n)} + \Delta w_{ij}$$

(3.22)

$$w_{jk}^{(n+1)} = w_{jk}^{(n)} + \Delta w_{jk}$$

(3.23)

Summarizing this process, for a certain input $x_i$ , there is a forward flow of information that goes until the output layer and then the output $z_k$ is computed. The update of the weights happens following the idea expressed in equation (3.23). Since one do not have desired values to calculate the error as in equation (3.8), it is made a distribution of the output errors as an attempt to adjust those inner weights. It is due to the existence of the value $\delta_j$ that the adjustment occurs, according to the equation (3.21).

There is no universal rule or method to find out the best number of hidden layers and hidden neurons. For each case there will be an optimum, or at least close to the best, number for those variables, which depend on the dataset or the number of inputs, for example. The latest neural networks are similar to the one in figure 3.8 with certain changes, such as the number of hidden neurons and layers or the
algorithm used to minimize the error.

Neural Networks find applications in a wide diversity of areas. In what concerns to the field of solar forecasting, there was an increase in the interest after the 1990s. The range of that research includes many areas of energy science, not only for prediction purposes but also for modelling of solar steam-generators, solar water heating systems, heating ventilation and air conditioning systems and refrigeration [36]. There are several ways of implementing a neural network algorithm. This varies according to the type of neural network that is supposed to be used and there also exists the possibility of creating a hybrid method which combines an ANN with other others methodologies.

3.2 Adaptive Neuro-Fuzzy Inference System (ANFIS)

3.2.1 Introduction to the Neuro-fuzzy System

Adaptive Neural Fuzzy Inference System, or simply ANFIS, is an AI algorithm that matches adaptive neural network with fuzzy set theory. At this point, one already know the ability of learning and adapting of a neural network. On the other side, the fuzzy system helps to approximate to the human knowledge by efficiently using linguistic information. One can say that ANFIS consists in a neuro-fuzzy system, which goal is to model the qualitative aspects of human knowledge by a learning capacity that can lead to take correct decisions. This process does not try to employ quantitative analyses. By contrast, a neuro-fuzzy system introduces a new idea, which is the existence of fuzzy if-then rules. Takagi and Sugeno were the first ones to explore this kind of fuzzy modelling [37]. Before going deeper in the definition of the architecture of ANFIS let us just refer two basic aspects that allow a better comprehension.

- There is no typical way of converting human knowledge into a fuzzy one.
- It is of extreme importance for the performance to have methods to generate the membership functions (MFs) that will be later described.

3.2.2 Description of the Fuzzy System

Let us start by defining what fuzzy if-then rules are. These fuzzy conditional statements came from the need of adapting to an uncertain and imprecision environment. Fuzzy if-then rules have a concise form and they can be implemented as a way of getting close to the human ability of making decisions in such harsh environments. They are expressions with the form: IF $A$ THEN $B$. $A$ and $B$ are labels of fuzzy sets and each one has its own specific MF. It will be given a simple example to show how it works.

*If pressure is high, then volume is small.*
So it is said that *pressure* and *volume* are linguistic variables and *high* and *small* are linguistic labels, characterized by appropriated membership functions. Takagi and Sugeno developed two methods. The first one is that the right side of the expression, the consequent, has fuzzy sets like the left one, the premise. For the second method, the fuzzy property only appears in the premise part.

![Fuzzy Inference System](extracted from [37])

In figure 3.9, one can notice that there are five functional blocks. Let us make a short description of each block. In 1) occurs the transformation of crisp inputs into degrees of match with linguistic values. Blocks 2) and 3) are associated and often referred as *knowledge base*. Block 2) contains the fuzzy if-then rules and block 3) defines the MFs of the fuzzy sets. The fourth block is responsible to perform inference operations. The last block turns the fuzzy results into crisp values, again.

The fuzzy inference system can be described by referring the four associated stages [38]. Let us try a brief description of each one. The first stage is commonly called fuzzification, where the input variables and the MFs are defined and compared in order obtain the membership values and to turn the crisp dataset into fuzzy type. After this, the database defines the rules and the membership values are combined to get firing strength, or weight, for each rule. Then the inference happens. Here, the database information is assessed and generates the qualified consequent for each rule, according to the weight. The last stage is the defuzzification, where the qualified consequents are aggregated in order to produce a crisp output.

There are considered three types of fuzzy inference systems, according to the fuzzy reasoning and according to the rules that are implemented. Since R software adopted the Type 3 by Takagi and Sugeno [39], only this one will be approached. For this type, it is made a linear combination of the input variables plus a constant term for the output of each rule and, then, it is computed the weighted average to reach the final output.

### 3.2.3 Adaptive Network

By adaptive network, one is referring to a different type of network that contains adaptive nodes. The output of these nodes depends on some parameters and on learning rules to adjust those parameters in order to obtain better results. As a multi-layer feed forward network, each node has an appropriated
function that may not be the same for the other nodes. The links between nodes in an adaptive network do not have weights associated, they only point out which direction the flow must follow. So, one can attribute different functions to the nodes. There are fixed nodes, with no parameters represented by circles, and adaptive nodes, represented by squares. The overall parameter set needs to be changed according to the training data and following a learning method, according to a descendent gradient. For this training procedure, it is possible to identify two hybrid learning methods. Firstly, the batch learning or off-line learning. Then the pattern learning or on-line learning. Let us focus on the off-line learning, because it is the one that was implemented in R software.

A Type 3 fuzzy inference system with a hybrid off-line learning method was used for the forecasting process. The parameters are identified by a combination of the gradient method and the Least Square Estimates (LSE), because using only the gradient causes slowness and can be stuck in local minima. The description of this learning method is in [38] and due its complexity will be out of the context.

3.2.4 ANFIS Architecture

After separately introduced the components that constitute an ANFIS, it is now time to describe ANFIS itself. Let us use figure 3.10 as an example. This scheme is composed by two inputs and a single output and it is assumed that there are two fuzzy if-then rules of Takagi and Sugeno’s type [38]. The rules are formulated such as:

- Rule 1 – if \( x \) is \( A_1 \) and \( y \) is \( B_1 \), then \( f_1 = p_1 x + q_1 y + r_1 \);
- Rule 2 – if \( x \) is \( A_2 \) and \( y \) is \( B_2 \), then \( f_2 = p_2 x + q_2 y + r_2 \).

where the values \( p, q \) and \( r \) are a set of parameters for both rules.

Figure 3.10. Type 3 fuzzy reasoning (extracted from [38])

In figure 3.10, it is illustrated the inner fuzzy reasoning for a Takagi and Sugeno’s type 3 system.
Figure 3.11. ANFIS structure for Type 3 fuzzy reasoning (extracted from [38])

Figure 3.11 intends to show the structure of the underlying ANFIS type. Notice that, for each layer, the array of nodes belong to the same function family, as can be seen in the figure. So, each layer has its properties and it appears to be reasonable to address what happens in each one.

- In the first layer, the nodes are square what means that they are adaptive. Then comes the following node function.

\[ O_i^1 = \mu_{A_i}(x) \]  

\[ (3.24) \]

\( O_i^1 \) is the MF and it expresses how the input \( x \) satisfies the linguistic label \( A_i \) associated. Normally, \( \mu_{A_i}(x) \) is bell-shaped, between 0 and 1, and therefore the equation:

\[ \mu_{A_i}(x) = \frac{1}{1 + \left[ \frac{x - c_i}{a_i} \right]^2} \]  

\[ (3.25) \]

where the values \( a_i, b_i \) and \( c_i \) are parameters that allow the shaping of the MF. These values are called premise parameters. It is obvious that the equation (3.25) is valid for the square nodes of other layers. It is possible to use MFs besides the bell-shaped one.

- For layer 2, the nodes are circles and are labelled \( \Pi \). This means that the adjacent output comes from the multiplication of the input signals, as shown in equation (3.26).

\[ w_i = \mu_{A_i}(x) \times \mu_{B_i}(y), \text{ for } i = 1, 2 \]  

\[ (3.26) \]

The value \( w_i \) of each node is the firing strength of a rule.

- The circle nodes of the third layer are labelled \( \mathcal{N} \) and have the goal to compute the ratio of the firing strength of the \( i^{th} \) rule as it is indicated in equation (3.27).
\[
\bar{w}_i = \frac{w_i}{w_1 + w_2}, \text{ for } i = 1, 2
\]  

(3.27)

For this reason, these outputs are denominated *normalized firing strengths*.

- In the fourth, all square nodes have the node function shown in equation (3.28).

\[
O^4_i = \bar{w}_i f_i = \bar{w}_i (p_i x + q_i y + r_i)
\]  

(3.28)

Parameters \(p_i, q_i\) and \(r_i\) are referred as *consequent parameters*.

- Last layer calculates the final output by adding all the incoming signals. Consequently, the overall output is computed by

\[
O^5_i = \sum_i \bar{w}_i f_i = \sum_i w_i f_i
\]  

(3.29)

Figure 3.11 allows a better understanding of the process and helps in the interpretation of the previous equations. One can notice that the output \(f\), in figure 3.10, can also be written as a linear combination of the consequent parameters, assuming that the premise parameters are known. Therefore, the output \(f\) comes as

\[
f = \frac{w_1}{w_1 + w_2} f_1 + \frac{w_2}{w_1 + w_2} f_2
\]

\[
= \bar{w}_1 f_1 + \bar{w}_2 f_2
\]

\[
= (\bar{w}_1 x) p_1 + (\bar{w}_1 y) q_1 + (\bar{w}_1) r_1 + (\bar{w}_2 x) p_2 + (\bar{w}_2 y) q_2 + (\bar{w}_2) r_2
\]  

(3.30)

The bidirectional hybrid learning algorithm is directly applied. In the forward pass, the process reaches the fourth layer and then the LSE determines the consequent parameters. By contrast, there is a backward propagation of error rates for the update of the premise parameters, in a descending gradient way. Let us have a look in table 3.1, which enables a clearer comprehension of these steps.

**Table 3.1. Passes of the hybrid algorithm**

<table>
<thead>
<tr>
<th></th>
<th>Forward Pass</th>
<th>Backward Pass</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Premise Parameters</strong></td>
<td>Fixed</td>
<td>Gradient Descent</td>
</tr>
<tr>
<td><strong>Consequent Parameters</strong></td>
<td>LSE</td>
<td>Fixed</td>
</tr>
<tr>
<td><strong>Signals</strong></td>
<td>Node Output</td>
<td>Error rates</td>
</tr>
</tbody>
</table>
The choice of Jang for the learning rule was a hybrid method based on gradient descent and the LSE, which derived from a balance between the computation complexity and the performance of the results [38].

3.3 Random Forest (RF)

3.3.1 Ensemble Learning

This topic started to receive more and more attention in the last years. Ensemble learning is related to methods that are able to produce classifiers in order to make a global aggregation of their results.

In terms of classification trees, there are two well-known methods: boosting and bagging of classification trees.

Boosting of classification trees was introduced by Shapire et al. [40]. The concept of boosting is the use of successive trees to reweight points that are thought to be wrongly predicted by previous predictors. The predictions come from a weighted voting for all the classification trees.

On the other hand, Breiman [41] developed bagging of classification trees. So, bagging predictors are used in this method in such way that it is possible to create an aggregated predictor, instead of having multiple ones. Those multiple versions are formed by bootstrapping samples of the learning set and use them as new training sets. Having said this, each tree is built independently of previous ones. In the end, it is done a plurality vote for prediction.

3.3.2 The Evolution of Bagging

A Random Forest is seen as an evolution of the bagging method. As told before, the idea of growing an ensemble of classification trees, so that they vote for the most popular class, has been updated lately.

After the bagging idea of Breiman [41], there is a random selection, where the split in each node is randomly selected between the best splits, according to Dietterich [42]. Then, Breiman explored the idea of randomizing the outputs in the original training set to create new ones [43]. Ho developed “the random subspace”, where there is a random selection of a subset of features that will be employed in the growth of each tree [44]. The paper of Amit and Geman [45] made an important contribution for the work of Breiman in 2001 [46]. They had defined a large amount of geometric features and sought for the best split in each node by performing a random selection of these features. In [46], Breiman’s idea was to add another property to random forests. So, besides bagging, the author included a modification to enable the change in the construction of the tree. Differently from a standard tree, where each node is splitted using the best split among all variables, in a random forest tree, it is used the best split among a subset of predictors that were randomly chosen at that node. This strategy is considered to be better than other methods like support vector machines (SVM), neural networks and others, as it was
demonstrated in that work. Random forests are also known for being strong against overfitting.

### 3.3.3 Mathematical Framework of Random Forest

The procedure inside a random forest is going to be focus of attention. Let us say that, for the $k^{th}$ tree, it is generated a random vector, $\theta_k$, that has the same distribution of the other random vectors being, however, independent of them. The arrangement between that vector and the training set leads to a classifier, $h(x, \theta_k)$, where $x$ expresses an input vector. So, each tree will vote for the most suitable class for each input. According to the different ways of building a tree, the vector $\theta$ will have a certain nature and dimension.

So, in regression terms, it is made an unweighted average among all the trees, such as

$$\bar{h}(x) = \frac{1}{K} \sum_{k=1}^{K} h(x, \theta_k) \quad (3.31)$$

Let us assume $X$ as the random vector associated to the inputs and $Y$ as the numerical outcome. Applying the Law of Large Numbers, when $k \to \infty$ it is guaranteed that

$$E_{X,Y}(Y - \bar{h}(X))^2 \to E_{X,Y}(Y - E_{\theta}h(X, \theta))^2 \quad (3.32)$$

where the quantity on the right side is defined as the prediction error of the random forest, designated $PE_f^*$. Robustness against overfitting is assured by the convergence of equation (3.32). For a single tree, the average prediction error is defined as

$$PE_t^* = E_{\theta}E_{X,Y}(Y - E_{\theta}h(X, \theta))^2 \quad (3.33)$$

So, assuming that each tree is unbiased for all $\theta$, it is true that

$$PE_f^* \leq \bar{\rho}PE_t^* \quad (3.34)$$

The parameter $\bar{\rho}$ is the weighted correlation between residuals $Y - h(X, \theta)$ and $Y - h(X, \theta')$ for independent values of $\theta$. There are two important aspects to perform an accurate regression with random forest. It is needed a low correlation between residuals and a low prediction error for the individual trees, which is regulated by the factor $\bar{\rho}$. In order to realize these goals, it is necessary to perform the actions exposed thereafter. The tree is grown until maximum depth so can be possible to maintain a low individual prediction error. On the other hand, a low residual correlation is achieved within two steps:

1. use a bootstrap sample from the training dataset to build each tree;
2. each node of each tree must select a certain number of covariates, lower or equal to the total number of covariates, and pick the best split based on those.
3.4 K-Nearest Neighbors (kNN)

3.4.1 First Approach to kNN

The k-Nearest Neighbors (kNN) algorithm is considered one of the simplest machine learning techniques. The first approach to this subject was driven by Fix and Hodges in a technical report that had not been published. The underlying approach was afterwards exploited and published by Silverman and Jones [47]. This report regarded the density estimation for classification theory and contained a lot of information that even nowadays is of huge importance.

The kNN method is a pattern recognition algorithm for classifying patterns or features. It is a nonparametric technique used to estimate a density function from a sample pattern. kNN differs from the statistical methods in such a way that, the model itself, is constructed by the training data, i.e., not requiring a testing subset after. Unlike some of the methods described before, kNN do not have unknown coefficients during its process. This method, in what concerns to classification theory, is well defined and there are no doubts about its mathematical scheme and its properties. However, outside this perspective, it is possible to heuristically define kNN by two steps:

1. Initially, it is created a set of objects from the data and then it is attributed a property or a value to each one;
2. To classify a new object, a decision rule is applied. This rule’s function is to use $k$ of the nearest neighbors of that unclassified new object so that the new one can be assigned according to its class or value.

It is now explained why this algorithm is designated k-nearest neighbors. The value of $k$ is positive and integer and normally assumes low values.

Despite being developed following the classification theory, kNN has also been applied for regression matter in time series. Yakowitz thought that kNN deserved to be implemented in time series applications due to its appealing nature and competitive properties [48]. Therefore, this method started being used as a forecasting technique too. Recently, it has been used for a wide diversity of forecasting problems. Figure 3.12 will help to understand the classification process of the kNN algorithm.

On the left side, it is represented one unclassified object, the triangle, and he three nearest neighbors by grey circles. So, a weighted average among the nearest neighbors is performed by the neighbors’ classes. The class to be attributed to the new object will derive from the inherent weights and the new object’s classification can adopt any value inside the shaded area.
3.4.2 Analog Method

kNN is known as analog method when related to meteorology and weather forecasting. Some examples of papers that discussed this question are in references [49-51]. The purpose of this method is to look for information of past weather patterns that has the best proximity to the current conditions. The past weather conditions that have been chosen are called analogs and its utility is to forecast weather and its underlying components. In what concerns to time series, the analog method searches over the time series history in order to find previous timestamps as close as possible to the actual situation. Those timestamps are considered the nearest neighbors. Then, the prediction is calculated.

Let us make the summary of the implementation of this algorithm. For a certain time $t$, it is assumed that $n$ features or patterns are assembled in a vector $p$ in the form \{ $p_1, \ldots, p_n$ \}. A matrix $A$ is created for the historical data, where each row has the information of the features of the past. So, for each $p_i$, the $k$ nearest neighbors can be calculated within two steps:

1. In equation (3.34), it is computed the Euclidean distance between $p_i$ and all the values of the historical dataset.

$$
    d_j = \sum_i \sqrt{(p_i - A_{ij})^2}
$$

(3.34)

2. The $k$ values that produced a smallest distance are extracted with their linked timestamps \{ $\tau_1, \ldots, \tau_k$ \}.

After established a set of nearest neighbors, the forecasting is computed by

$$
    z_{t + \Delta t} = \frac{\sum_{i=1}^{k} \alpha_i z_{\tau_i + \Delta t}}{\sum_{i=1}^{k} \alpha_i}
$$

(3.35)
The weights are represented by $\alpha_i$ and are often a function of the Euclidean distance. Therefore, the final result, $z$, comes as a linear combination of the time series values with the weight values.

### 3.5 Persistence Method

The persistence method is a linear model and it is the simplest way to make a prediction. It is commonly associated as a benchmark in what concerns to the performance of other techniques. The idea of comparing a forecasting tool with this method is to find out whether the tool outperforms the persistence method or other trivial model. This baseline model is assumed as the most common reference model in the solar and wind forecasting community, for short term forecasts. According to a specific time horizon, the persistence's prediction will have the same value as in the last time period. In other word, the predicted value of some variable at the time $t$, $X_t$, is the value at the time $t-1$.

$$\hat{X}_{t+1} = X_t$$  \hspace{1cm} (3.36)

Note that the accuracy of the persistence model decreases with the forecast duration, since there can be a notorious change in the cloudiness or other meteorological factors. That is why it is recommended to use it for predictions for time horizons less than one hour.

For the problem that is being solved, the persistence method will set the “forecasted” value for each minute, the power of the previous minute.
In this chapter, it will be made a description of the process of implementing the algorithms, including the data handling and therefore the way of dealing with each method.

All the work was driven in RStudio. RStudio is a recent open-source integrated development environment (IDE) for R, a statistical language used for data analysis. Using some R packages and some of their functions, it became possible to implement the four distinct algorithms that were described in the previous chapter. A vast research was made as an attempt to seek the best way to take advantage of each algorithm. Knowing that the main purpose of this thesis is to perform a one-day-ahead PV power forecast, it will be made three different forecasts, for different meteorological conditions. These differences will be crucial for the final evaluation of the main question.
4.1 Dataset and Data Selection

For the development of this work, it was used the same data for all the mentioned methods. The available dataset\(^1\) contained values of the module temperature, solar irradiance and PV power output, per minute, of the first five days of January and the first five of August. Even if it is known that the solar irradiance, and furthermore the solar power, is affected by a large number of external variables, such as cloud cover, air pollution, among others, none of these features will be included in the training process of these techniques.

All the available values came from a group of photovoltaic panels, where each panel is from JINKO Company and the module type is the JKM250P version. The specifications of the underlying panel can be found in ANNEX A.

The purpose of this work is to predict PV power output and it was decided to accomplish that for three days with different conditions. The main factor that distinguishes these days is the cloud cover. Therefore, it will be made a forecast for a clear day (practically with no clouds), a relatively cloudy day (with both clear and cloudy intervals) and a completely cloudy day. A forecast will be performed in order to reach a value of PV power as close to the real one as possible. The goal is to use the solar irradiance and the module temperature as inputs in order to obtain an output, i.e. a predicted value of the solar PV output power, for each minute. So, it is an initial approach that can be applied to real situations in the future, since, in this situation, it has been applied for past cases. The data contains measurements for five winter days and for five summer days, in which three different days will be chosen according to its own characteristics. The days with the desirable conditions are the 4\(^{th}\) of January, the 5\(^{th}\) of January and the 3\(^{rd}\) of August, a clear day, a partly cloudy day and a cloudy day, respectively. In agreement with each season, different time intervals were chosen ‘to be forecasted’, intervals that had been selected because they were thought to be more relevant for that experiment. In what concerns to January, the selected period to perform the forecast was from 9:00 to 16:00 for both days of January, while for August it was preferred to work between 10:00 and 17:00. These time periods were chosen because there are low values of the solar irradiance, and so the PV power output, in the early morning and in the late afternoon what would difficult an accurate forecast. This happens because a small difference between low values can result in bigger errors.

After mentioning the data that is going to be used, let us explain how the process was handled. Initially, a data frame was generated and then attached in the RStudio environment, which allowed an easier utilization of the data. Then, the data was splitted into a training dataset and a test dataset. In order to try to reach a prediction as efficient as possible, all the values outside those specific time intervals were integrated into the training process. The idea is to use the maximum number of values for the training set, attempting to improve the performance of the algorithms.

---

\(^{1}\) The dataset contains measurements of 2015 and it was kindly provided by EDP-Inovação.
The highest the variability of the solar irradiance, the bigger the related weather non-linearity, which will highly affect the time evolution of the PV power output too. For a completely sunny day, with total absence of clouds, the irradiance will be a smooth bell-shaped curve, which concavity points down, as illustrated in figure 4.1. Figure 4.2 contains the time evolution of the solar irradiance for a partly cloudy day, which can be split into two different sections: a cloudy and a sunny part. Finally, a cloudy day will have an irregular evolution, with several ups and downs, what can be observed in figure 4.3. Here, the existence of several peaks shows the huge variations caused by cloud cover during the day, which will result in relevant perturbations in the PV power. These time evolutions are required to demonstrate the desirable conditions of each day.

![Figure 4.1. Time evolution of the irradiance for the 4th of January](image1)

![Figure 4.2. Time evolution of the irradiance for the 5th of January](image2)
Let us sum up the process of data selection. The AI techniques should go through a training phase, and all the methods followed the same strategy. For this stage, the used data consisted in all the values of module temperature, solar irradiance and PV power output outside the interval detailed to each season, for the three different cases. The training will enable the techniques to make their own one-day-ahead PV power output forecast. After this stage, the algorithms will only receive the values of the module temperature and the solar irradiance for all the minutes, i.e. between 9:00 and 16:00 for January and from 10:00 to 17:00 for August. It is explicitly assumed that the learning process must work with the PV power output as the output. After predicting the PV power for the underlying cases, a comparison between the real power and the predicted one will be made, also including the time evolution of the error. There is one exception, kNN does not integrate two different phases, what will be explained later in this chapter. Another assessment that is going to be performed is the variation of some inner parameters of the algorithms in order to study how the simulation time and the mean average percentage error, MAPE, are affected by that variation.

### 4.2 Implementation of the Techniques

The next sections seek to demonstrate how the algorithms were applied, how the parameter selection was run and some other important actions. Each method has its unique specifications, properties and way to be implemented.

All the methods, for the days of study, were submitted to an optimization stage. This phase assumed different strategies, starting from the basic process of trial and error to the use of R packages. After developing the ability to deal with that point, it was necessary to adjust each method to each test. Different datasets will need different approaches to solve the problem.

Before start to describe how each algorithm was implemented, let us make reference to one package.
that was also used, called “caret”. The R package “caret”, which stands for Classification And REgression Training, has the role of helping in the process of creating predictive models. This step was conducted as an attempt of improving the global performance. The application of the \texttt{train} function to a particular training grid or vector, where some important parameters are tuned, leads to an optimum parameter. This conclusion is reached following a control method, which was the \texttt{Cross Validation}, excepting for random forest, where the chosen method was the \texttt{oob}, \textit{out-of-bag}. The \texttt{out-of-bag} represents the alternative to the \textit{cross-validation} method for a random forest. When describing the random forest algorithm, it was said that each tree of the forest is built from different bootstrap sample of the data and approximately one-third of the data does not enter in this process, being used later for the classification. The ratio between the mistaken classifications over all the cases represents the \texttt{oob} error estimate. The parameters given by this package are the ones with the lowest value of the RMSE.

After this, it is time to implement the R functions of each technique. Let us follow the same order of the algorithms as it was in the previous chapter.

\subsection{4.2.1 Neural Network Implementation}

Starting from neural network implementation, the \texttt{"neuralnet"} package was used so that could be possible the training of a network. A group of different methods of training are available: backpropagation, resilient backpropagation (with [52] or without weight backtracking [53]) and the modified globally convergent version [54]. The custom-choice of error and activation function led to flexible settings. Additionally, there is the computation of generalized weights. For this thesis' problem, the resilient backpropagation training was elected.

One important step to take into account was the need to perform a normalization, which was applied to all the variables. Therefore, the values of each variable were normalized between 0 and 1, by dividing all the variables for its maximum value before the process of data splitting. This act was of extreme importance in order to allow the convergence of the method also constrained the simulation time, which was drastically reduced. Afterwards, it is imperative to decide how many neurons and layers should exist to design the network. After some tests, it was concluded that better results were obtained for one and two hidden layers, not being necessary to use more than two layers. For each layer, one does not a huge number of neurons. Besides increasing the simulation time, several neurons will not necessarily lead to better results.

Having this in mind, after applying the \texttt{caret} training function, one concluded that was better to use one hidden layer for the first test and two hidden layers for the others. The number of neurons used for each cases was not the same of the one of the “caret”, because, after testing, one noticed that it was possible to obtain better results with a different number of neurons. Then, calling the \texttt{“neuralnet”} function of the \texttt{“neuralnet”} package, a neural network was created. The number of layers and neurons was, for the first test, a single hidden layer with 20 neurons, two inner layers with 25 and 10 neurons, respectively for each layer, and the last test was performed also with two hidden layers with 50 neurons for the first one and 10 neurons for the second. As an artificial intelligence technique, and due to the different learning capabilities, each network will have unique weights and so there will be no equal networks. In other
words, it is practically impossible to have the same predicted values for different simulations. Trying to deal with this question, it seemed reasonable to make more than one network and so it was run 30 times the training and the predicting processes. This number for the method to be run did not have a theoretical reason for its choice, it simply seemed reasonable for the purpose. In the end, the predicted power was assumed to be the average of all that 30 values of power. Through this, it was possible to achieve more exact results, according to the parameters that had been selected. It is pertinent to say that the total time of the simulation depended on how many cycles, i.e. the number of times that the forecasting process is run, were created, what was only made to try to obtain a more precise result. This was done since, for different simulations with the same parameters, there was sensibly different values of the error.

At the end, it was chosen to do a sensitivity analysis varying the number of cycles with the purpose study how this variation influences the simulation time and the MAPE.

4.2.2 ANFIS Implementation

The implementation of an Adaptive Neuro-Fuzzy Inference System needed the “anfis” package. This package implements the Type 3 Takagi and Sugeno’s fuzzy if-then rule network [38], with a flexible MF framework that allow the user to an easier manipulation of the number and type of the MFs for each input. The concept of MF has already been described last chapter. The package has also full rule combinations, what means that, depending on the number of inputs and MFs, always leads to the maximum number of fuzzy rules. The package also offers different learning strategies from where the trainHybridJangOffLine have been selected [38]. This hybrid learning applies a descent gradient for precedents and least square estimation for consequents.

For this case, the “caret” did not exhibit an important role for resolution of the problem. This fact is justified since the available tuning parameters in the “caret” were not relevant for the implementation of the underlying methodology in this work.

After splitting the data, it is required a formulation of a MF that must be in agreement with the input dataset. Different datasets should have different MFs and so it is necessary to adjust one to each other. Said this, each input must have its own MF, a function that must be adjusted to the data. The chosen MFs were of bell-shaped form for all the experiments, where two MFs were created for each of the input variable. So, knowing that the input dataset is different for each case, different MFs were created. Having an ANFIS class object already produced, it was submitted to the trainHybridJangOffLine training method, as it was mentioned. In this stage, a number of epochs for the training must be elected, which is dependent on the input dataset and on the MF. The number of epochs applied for each case was 6, 3 and 7 for each case, respectively. For this algorithm, it was not necessary to create any cycle, since the way of building the ANFIS’ structure always lead to the same form and therefore the predicted values did not vary for plural simulations.

In what concerns to the sensitivity analysis, one decided to change the number of epochs of training to make a general assessment about the effect caused by this variation in the MAPE and the simulation
4.2.3 Random Forest Implementation

Following the general idea, and after splitting the data into training and testing sets, let us start the construction of the forest. The R package "randomForest" was used to perform this action. Inside the package, there is a function called "randomForest" that allows the implementation of the Breiman’s random forest algorithm [46], which code was based on the original Fortran code created by Breiman and Cutler [55]. As it is known at this point, random forest can be used in both regression and classification problems. Here will be obviously a regression problem.

When working with the "caret" package, the type of control used was "oob" that stands for an out-of-bag control and it is commonly used and specific for random forests. From “caret” training came the conclusion that the value that led to the smallest OOB error for the mtry parameter was 2, what was the best choice for all the experiments. This parameter is related to the number of variables randomly sampled as candidates at each split to design the tree. Then, it must be selected the number of trees for each problem. For the first test, it was chosen to use 900 trees and for the other two tests, 1500 was the number trees to build. Each random forest is unique, what leads to a unique prediction, similarly to neural networks. The internal randomness of the algorithm leads to the singularity of each attempt. So, here also arises the need to perform more than one prediction, by generating more than one forest (with the same parameters), through a cycle that makes 30 runs, the same value used for neural network. This step has the purpose of reaching a more precise value of the predicted power, according to the parameters of the forest and its way of being built. The final power was calculated as the mean value of all the 30 different predicted values.

Afterwards, one decided to vary the number of trees to see how the MAPE and the simulation times react to the change. For this analysis, the number of cycles remains constant and equal to the value used for the main test, 30.

4.2.4 kNN Implementation

This technique, already known for its simplicity, reveals to be very easy to implement because, in opposite to the other methods, kNN has the training phase in simultaneous with the prediction one. The R package that made possible the implementation of this method was the “class”package. Applying the “knn” function, one is applying a k-nearest neighbor classification into a test set from the training set. The k-nearest, in Euclidean distance, training set vectors are sought for the respective value of the test set. Then, a majority vote induce the classification.

The only variable to be tuned is the number of the nearest neighbors, k. The value of k that led to the best results is one, for all the tests. After this process, it was necessary to make a conversion of the type of the output, to be possible to have numeric values of the final results to enable the comparison with the real values.
Besides the main purpose, the sensitivity analysis was performed by changing the number of nearest neighbors of the algorithm.

### 4.3 Performance Assessment

All those methods needed to be submitted to some kind of performance evaluation so it can be possible a more reliable comparison between all the obtained results.

There are several ways to conclude whether a forecast is well done or not. Different features have different degrees of variability, and, in what concerns to the prediction of solar variables, it is already known the inherent high variability. This issue can be assessed through some traditional statistical methods. Let us list some of the most known ways to evaluate the performance. The following equations are expressed in a universal formula, where $z$ represents the input data, $\hat{z}$ refers to the predicted values and the number of samples is referred by $N$.

- **Root Mean Square Error (RMSE)**

  $$RMSE = \sqrt{\frac{\sum_{t=1}^{N} (\hat{z}_t - z_t)^2}{N}} \tag{4.1}$$

- **The Mean Absolute Error (MAE):**

  $$MAE = \frac{1}{N} \sum_{t=1}^{N} |\hat{z}_t - z_t| \tag{4.2}$$

- **The mean absolute percentage error (MAPE):**

  $$MAPE = \frac{100%}{N} \sum_{t=1}^{N} \frac{|\hat{z}_t - z_t|}{z_t} \tag{4.3}$$

Besides these more traditional ways, there are other coefficients that held the resolution of this topic, such as

- **The coefficient of determination (R):**

  $$R^2 = 1 - \frac{Var(\hat{z} - z)}{Var(z)} \tag{4.4}$$

where it is compared the variance of the error with the input data variance;

- **The correlation coefficient ($\rho$):**

  $$\rho = \frac{(Cov(\hat{z} - z))^2}{Var(\hat{z})Var(z)} \tag{4.5}$$

which relates the covariance of the difference between the real and the predicted value with product of the variances of both the forecast and the input. However, none of these will be used.
Passing the years, there was a development in the way of measuring the accuracy of forecasting models. Using some of the above formulas and applying different kinds of normalizations, new performance assessment techniques have arisen.

For the current work it was preferred the use of the MAPE. The lower its value the better. Another strategy that was implemented to evaluate the performance of each method was the relation between the MAPE of the prediction and the MAPE of the persistence model.

\[
\text{MAPE}_{\text{model}} = \frac{\text{MAPE}_{\text{forecast}}}{\text{MAPE}_{\text{persistence}}}
\]  

(4.6)

It was previously said that the persistence model is a good way of comparing the performance of an artificial intelligence method, for short term predictions. Hence, the error of the persistence should be higher than the error of the prediction or, at least, it is one important objective among the forecasting community. Equation 4.6 is a decent manner of analysing that relation with a single value. The MAPE of a model will be less than one for the cases that perform better than the persistence. The lower the MAPE\text{model} the better the performance will be and the better will be the model compared with the persistence.
In this chapter, the results will be shown. The main goal of this work is to compare the performances of the used algorithms. Therefore, let us present the results and analyse them. The simulations came from RStudio and the values were passed to EXCEL in order to build a more suitable and presentable graphical evolution of the PV power output, for each day in study. This section will be splitted into three major parts, the first regarding to the clear day of January (4th of January), the second related to the partly cloudy day (5th of January) and the third for a summer day of August, which is a cloudy day, (3rd of August). At the end, the sensitivity analysis will also be presented.
5.1 Results for the Sunny Day

The first test consists in a one-day-ahead PV power forecast for a sunny, clear day. This was proven in the previous chapter in figure 4.1. The referred day is the fourth of January and the time interval between 9:00 and 16:00 was chosen for that day. Since there are two winter days to be analysed, the previous interval will be used for both cases, because in the winter the values of the irradiance are lower and outside that range are practically meaningless. Small values of power would only bring additional difficulties for the prediction process.

At this point, every forecasting step had already been completed, according to the parameter selection referred in the previous chapter. Therefore, the results of each model will be shown, graphically comparing the real power with the forecasted one. Besides this representation, it will also be presented and explained the evolution of the forecasting error. The forecasting error is computed by

\[
error_i = \left| \frac{p_i' - p_i}{p_i} \right| \times 100\% \quad (5.1)
\]

where \( p_i \) represents the real value of the power and \( p_i' \) is the forecasted power, for each minute. After these steps, it will also be computed the MAPE for all the cases of study. Another computation will be performed in order to compare the efficiency of the panels, applying the equation 5.2 for both powers. The implemented formula was

\[
\eta = \frac{P}{A \times G} \quad (5.2)
\]

Where \( P \) represents the power output, \( A \) is the area of the panel given by the manufacturer and \( G \) represents the value of the solar irradiance. This result will be, in fact, the average of all the values of the efficiency calculated for each minute of the time interval that is being analysed.

In the end, a global comparison of the simulation time and the MAPE of the models will be made.

5.1.1 Neural Network Results for the Sunny Day

For this first test, according to the parameters established last chapter, the neural network lead to the following result.
The MAPE obtained was 2.00%. In general, the prediction follows the real values. The biggest errors came from the smallest values of the power, where a slight variation leads to higher miscalculations.

### 5.1.2 ANFIS Results for the Sunny Day

After running the function capable to create the ANFIS, it were obtained the results illustrated below.
Then, a MAPE of 1.71% was measured. Here again, the smallest values, even with slight differences, lead to more significant errors.

5.1.3 Random Forest Results for the Sunny Day

Random forest’s parameters had been established last chapter and after the simulation, one can take a look in the results.
For random forest, the final MAPE was 2.16%. This situation appears to be more instable than in the other cases. There are several discrepancies between both power evolutions, visible in figure 5.3 (a), where the prediction drives out the real values of the power.

5.1.4 kNN Results for the Sunny Day

It was implemented this method in RStudio and therefore the results are shown in figure 5.4.
kNN revealed the best results with a MAPE of 0.68%, practically the same as the persistence’s MAPE (0.678%). The biggest errors also arise for the first minutes, the ones with the lowest value of the power. Besides this, the overall results seem to be very reasonable.

5.1.5 Discussion of the Results for the Sunny Day

In order to make an initial overview about the results of this first day, it seems appropriate to reveal them in a table. In table 5.1, it is possible to compare the values of the MAPE of each model, the ratio of the MAPE when compared to the persistence method and the time each simulation took.
Table 5.1. Overall results for the 4th of January

<table>
<thead>
<tr>
<th>Models</th>
<th>MAPE (%)</th>
<th>MAPE model</th>
<th>Simulation Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Persistence</td>
<td>0.678</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>Neural Network</td>
<td>1.993</td>
<td>2.945</td>
<td>981.24</td>
</tr>
<tr>
<td>ANFIS</td>
<td>1.707</td>
<td>2.546</td>
<td>60.64</td>
</tr>
<tr>
<td>Random Forest</td>
<td>2.158</td>
<td>3.181</td>
<td>444.96</td>
</tr>
<tr>
<td>kNN</td>
<td>0.677</td>
<td>0.998</td>
<td>0.03</td>
</tr>
</tbody>
</table>

It is important to start by referring that the persistence achieves a MAPE of 0.68%, what is a really low value. This is because of the fact that the 4th of January is a sunny day and so, during this day, there were not relevant oscillations of the power, and the variation per minute is very short. As it was already said, the persistence’s behaviour is better for minor intervals, since in these cases it may face lower variations.

The algorithm with the best performance is the kNN. It not only achieves the lowest MAPE, even lower than the persistence’s MAPE, but also it is the one that takes less time to run. After kNN, ANFIS was the algorithm that did better, needing more time than kNN. Random forest was the method with the worst MAPE. For a sunny day, with insignificant variations of the power during the day, the general behavior of all the methods was acceptable. Another point is the fact that the data appears per minute, what also points out to the difficult of reaching the same values of the persistence. So, for this first test, the persistence has a very low value of the MAPE and, besides kNN, the other algorithms did not had enough strength to match its results. It could have happened because of a lack of input variables or even the incapability to find the best parameters to reach a more accurate forecast.

There is a huge difference between the simulation time of neural network and random forest and the other two. This fact is explained by the establishment of a cycle to obtain an average value for the power, which required much more time. Besides the fact that the creation of a neural network and a random forest itself takes more time, introducing a cycle will lead to a more significant increase in the time taken to achieve the final predictions. Therefore, the final values introduced in the table are the full time required to reach the final predictions.

In table 5.2, it was computed the solar efficiency according to equation (5.2).
Table 5.2. Comparison of the panel efficiency for the 4\textsuperscript{th} of January

<table>
<thead>
<tr>
<th>Method</th>
<th>Efficiency (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Real power</td>
<td>13.78</td>
</tr>
<tr>
<td>Persistence</td>
<td>13.77</td>
</tr>
<tr>
<td>Neural Network</td>
<td>13.73</td>
</tr>
<tr>
<td>ANFIS</td>
<td>13.74</td>
</tr>
<tr>
<td>Random Forest</td>
<td>13.80</td>
</tr>
<tr>
<td>kNN</td>
<td>13.74</td>
</tr>
</tbody>
</table>

This study is just a way to compare how the predicted values could affect the efficiency, not being, however, a method that can permit to reach other important conclusions. From these values, it is only possible to say that, excepting for random forest, the overall values of the forecasted power are lightly inferior than the real power.

At the end of this chapter, different experiments will be made as an attempt to find a relation between a variation of the MAPE and the time needed for the simulation.

5.2 Results for the Partly Cloudy Day

The second test consisted in the PV power prediction for partly cloudy, winter day. The referred day was the 5\textsuperscript{th} of January and the time evolution of the irradiance is illustrated in figure 4.2. The time interval between 9:00 and 16:00 was chosen for that day, the same as the previous test. As it was already said, this interval was chosen due to the low values of the irradiance outside that range.

The underlying day has a cloudy period, for the initial hours, and then it behaves as a sunny day. The existence of peaks in figure 4.2 allows to conclude that the variations might be caused by cloud cover, although those variations can also come from other meteorological features. On the other hand, the second half of the day behaves as a sunny day since there are not any relevant peaks in the time evolution of the irradiance for that time period. Through what was read previously and having the results of the previous test, one can say that for the second half of this day the forecast will provide better values. This sentence comes as a previous conclusion for the purpose of this work, because it is known that a better concordance between inputs results in a more successful prediction.
So, in agreement with the others chapters, let us now exhibit the results, following the same order of the methods as before.

### 5.2.1 Neural Network Results for the Partly Cloudy Day

The design of the created neural network and the details of the entire process for the 5th of January has already been discussed last chapter. Let us present the results that were obtained from the implementation of a neural network in RStudio.

As can be seen in the preceding figure, there is a better approach to the real values in the second half of the underlying interval, confirming what was previously said. It is seasonable to show that fact by analysing the time evolution of the error of the prediction for each minute. This comes as a way of showing that higher variations of the power, like what happens for the first hours, lead to more erroneous forecasts. For this experiment, unfortunately, the MAPE was not better than the persistence (4.523%). The value obtained was slightly higher, with a value of 5.24%.
5.2.2 ANFIS Results for the Partly Cloudy Day

According to last chapter’s procedure for the implementation of ANFIS, the time evolution of the forecasted and the real power is illustrated in figure 5.6.

![ANFIS Results for the Partly Cloudy Day](chart)

This test reached a MAPE of 5.06%. Through the visualization of the graphics, it is possible to say that there is not a bad similarity between real and predicted values. The chosen membership functions were reasonably selected and worked well with the dataset used. However, a better performance could be eventually obtained if more proper membership functions were found, with a proper number of epochs of training. Other conclusions will be made at the end of the subsection.
5.2.3 Random Forest Results for the Partly Cloudy Day

Having the shape of the forest already defined, the result of the forecast is compared to the real power.

![Graph showing time evolution of power and error](image)

(a) Time evolution of the power

(b) Time evolution of the error

Figure 5.7. Results obtained for RF for the 5th of January

Random forest algorithm introduces the idea of randomness into the forecasting process. The time evolution illustrated is considered to be unique because no other forest could lead to the exact same values. The results are quite similar to the others cases and it was computed a MAPE value of 5.31%.
5.2.4 kNN Results for the Partly Cloudy Day

The forecast performed with this technique led to the results illustrated in figure 5.8.

As the figure indicates, the first hours, where there are more significant changes in the real value of the power, have bigger errors. So, after 11:30, approximately, the prediction followed almost exactly the real power. The kNN algorithm with the value of $k$ of one and using the described data performs quite well for the underlying problem, with a MAPE of 3.69%.
5.2.5 Discussion of the Results of the Partly Cloudy Day

In table 5.3, are presented the values of the MAPE, in percentage, for each method and the $\text{MAPE}_{\text{model}}$ to specify the relation of each case with the persistence method.

**Table 5.3. Overall results for the 5th of January**

<table>
<thead>
<tr>
<th>Models</th>
<th>MAPE (%)</th>
<th>MAPE model</th>
<th>Simulation Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Persistence</td>
<td>4.523</td>
<td>----</td>
<td>--------</td>
</tr>
<tr>
<td>Neural Network</td>
<td>5.238</td>
<td>1.16</td>
<td>564.41</td>
</tr>
<tr>
<td>ANFIS</td>
<td>5.057</td>
<td>1.12</td>
<td>23.25</td>
</tr>
<tr>
<td>Random Forest</td>
<td>5.312</td>
<td>1.17</td>
<td>453.25</td>
</tr>
<tr>
<td>kNN</td>
<td>3.688</td>
<td>0.815</td>
<td>0.03</td>
</tr>
</tbody>
</table>

The introduction of cloud cover for this test led to an increase in the error, once there is a bigger uncertainty, whose drawbacks are known. This increase is mainly originated from the first half of day, where the variations must be caused by external meteorological features. Before showing the results, it is already known that experiments with non-sunny days would produce worst forecasts.

The persistence was affected by the referred variability and the MAPE value was 4.523%. The MAPE of the persistence reveals to be low because of the time scale that is being used and, therefore, the variations among neighbor values are very short. Working with per minute data increases the difficulty of a forecast. So, once again, kNN’s performance was the best of all, not only concerning to the MAPE but also in terms of the simulation time. The performance of the neural network, ANFIS and random forest was similar between these three, where the MAPE is around 5% for all cases. These values themselves are not considered bad at all, however the computational power of these algorithms could lead to better results. The reasons for this powerlessness can be the lack of variables for the training process or the possible bad choice of some parameters. Nonetheless, the difficulty imposed by the problem itself also represents a big challenge and so the results are not considered bad. The results obtained by the ANFIS were better than the network neural that, in turn, were also better than the random forest, even though with small differences. It seems pertinent to refer that, in analogy with the previous test, the neural network and the random forest have higher simulation times due to the decision of performing cycles.

It is clear that for the first hours of the day, the oscillations of the error are bigger than in the rest of the day, what can be explained by the variation of the irradiance, and so the power. These variations led to uncertain predictions, caused by the disparity between the dataset of the training and the test values. In
other words, similar values of the inputs (temperature and irradiance) produce different power. In order to explain that, let us exploit the situation that happened around 11:30. The huge value of the error derives from a large variation of the irradiance and of the power, caused by external meteorological reasons. In all the cases, this consists in the most relevant peak of the error. This value strongly inflates the average error (MAPE).

The prediction from each method leads to an own value of efficiency. After applying equation 5.2 to each case, six different results emerge. These results come from the four relevant techniques of this work in addition to the values of the efficiency regarding to the real measured power and the persistence method, as it was done for the previous test.

Table 5.4. Comparison of the panel efficiency for the 5th of January

<table>
<thead>
<tr>
<th></th>
<th>Efficiency (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Real power</td>
<td>13.06</td>
</tr>
<tr>
<td>Persistence</td>
<td>12.99</td>
</tr>
<tr>
<td>Neural Network</td>
<td>13.01</td>
</tr>
<tr>
<td>ANFIS</td>
<td>12.97</td>
</tr>
<tr>
<td>Random Forest</td>
<td>12.94</td>
</tr>
<tr>
<td>kNN</td>
<td>12.89</td>
</tr>
</tbody>
</table>

One can notice a minor discrepancy between, let us say, the “measured” efficiency and the ones calculated from forecasted values. All the values of efficiency are lower than the real one, and so it is possible to say that the average value of the power of the forecast is inferior compared to the one measured in the panel.

5.3 Results for the Cloudy Day

The last test intended to perform a PV power forecast for a cloudy day, namely the 3rd of August. In spite of being a summer day, this day has high cloud cover, what was already shown in figure 4.3. As it was previously said and justified, for this day the time interval will be between 10:00 and 17:00.

For this test, one can expect higher difficulties, since it is know that one of the biggest challenges concerned to the solar forecast topic is to overcome problems associated to cloud cover. Then, the last
experiment will represent the hardest test of all and will provide worst forecasts. Having this small introduction made, let us expose the result.

5.3.1 Neural Network Results for the Cloudy Day

The last test required more a more complex network, which specifications have already been reported. So, the comparison between the real power and the predicted one is illustrated in figure 5.9.

![Graph of time evolution of power and error](image)

Figure 5.9. Results obtained for NN for the 3rd of August

There is a visible increment of the error, as it was expected. The higher variations of the irradiance this day’s, and hence the power, result in a higher average of the error. For this experiment, the MAPE value was 13.96%. It is clear that, when the biggest variations happen, the error assumes high values. There are three more salient peaks in the time evolution of the error, with significant values of the error, around
100%! The overall error is quite high, but may not be a matter of concerning due to the difficulties imposed by the problem. Many and noteworthy variations occurred for the underlying day, which caused some troubles in the forecasting process. However, since the beginning, it was already expected that one would face this struggle.

### 5.3.2 ANFIS Result for the Cloudy Day

The built fuzzy system for the cloudy day had different MFs and number of epochs of training. The results generated from RStudio are shown in figure 5.10.

![Graph showing time evolution of power and error](image)

(a) Time evolution of the power

(b) Time evolution of the error

Figure 5.10. Results obtained for ANFIS for the 3rd of August

In the end, the MAPE was 10.56%. It is possible to notice the existence of three irregular peaks in the time evolution of the error, similarly to what happened in the previous case. These miscalculations derived from the huge variation in the solar irradiance at each time. The discrepancy of values can be
justified by the high variability in those moments, leading to those tremendous errors. Therefore, what happens is that, for approximated inputs, i.e. similar values of the temperature and the irradiance, the power output assumes significantly different values. This is why so significant errors are generated. Here, it is also pertinent to refer that those peaks occurred at the same moment, as the ones in the preceding test.

5.3.3 Random Forest Results for the Cloudy Day

The forest had been built with the parameters previously established. The comparison between the real and the forecasted power, in addition with the time evolution of the error, is illustrated below.

![Graph showing time evolution of power and error for Random Forest results.](image)

Figure 5.11. Results obtained for RF for the 3rd of August

Once again, the prediction errors with the highest values took place at the same time of the day. The justification follows the previous ones, which is because the huge variations of the irradiance lead to
these ridiculously elevated values of the error. The forecast had a MAPE of 13.63%. Errors above 100% clearly show the striving of solving problems with such high variations during the day.

5.3.4 kNN Results for the Cloudy Day

One more time, applying the kNN method with $k$ equal to one, the results are visible below.

![Graphs showing time evolution of power and error](image)

Figure 5.12. Results obtained for kNN for the 3rd of August

The overall error, the MAPE, was 4.18% what reveals a great agreement between the forecasted and the real power. kNN did not present large errors as what happen for the other cases. The way of implementing this algorithm, with only one single step, presented the whole dataset and that fact allows the method to reach similar values to the reality.
5.3.5 Discussion of the Results for the Cloudy Day

In order to finish this work’s main analysis, let us make the comparison of the results for the cloudy day. Here again, it is appropriate to begin by showing the MAPE values and the simulation time for each algorithm.

<table>
<thead>
<tr>
<th>Models</th>
<th>MAPE (%)</th>
<th>MAPE model</th>
<th>Simulation Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Persistence</td>
<td>11.196</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>Neural Network</td>
<td>13.956</td>
<td>1.157</td>
<td>12650.18</td>
</tr>
<tr>
<td>ANFIS</td>
<td>10.558</td>
<td>0.943</td>
<td>96.19</td>
</tr>
<tr>
<td>Random Forest</td>
<td>13.631</td>
<td>1.217</td>
<td>581.12</td>
</tr>
<tr>
<td>kNN</td>
<td>4.182</td>
<td>0.373</td>
<td>0.03</td>
</tr>
</tbody>
</table>

Table 5.5 confirms what was previously said: the errors are considerably higher. The high value of the persistence’s MAPE is another evidence of the variability of the weather in the concerned day. The value of 11.2% indicates the diverse deviations in the time evolution of the solar irradiance and therefore the deviations of the PV power output. This value also justifies what was said before. The used time scale shows the huge variations of the weather itself. Using other words, the value of the MAPE represents an average variation of 11.2% from a minute to the following one, which also validates the referred variability of the underlying day.

In this last experiment, it is clearer the better performance of the kNN in relation to the others methods. This can be explained by the fact of kNN to perform the training and the prediction stages together. By including all the values of the dataset and by using a single nearest neighbor, the algorithm found values closer to the reality. The others cannot reach so accurate values, what can have multiple explanations. There are three peaks in the time evolution of the error, which caused a significant increase in the average error, otherwise the MAPE would have had lower values. If there were not those peaks, the errors would not have been so big for the global conditions of the problem. Besides this point related to the results, there are other external reasons. The inexistence of more input variables may affect the performance of these methods, where the option of inserting more meteorological features would lead to more accurate and consistent forecasts. Using variables that are not directly correlated to the cloud cover topic hampers the resolution of this problem.

More complex problems demand more exigent ways to be solved. Assuming this test as the most difficult test of this thesis, it was expected that the simulations required more time to be solved, besides carrying
worst results. kNN revealed to be practically instantaneous while ANFIS needed one and a half minute, approximately. Neural network and random forest required more time, once they were run 30 times, strongly increasing the simulation time. However, running a neural network needed a huge amount of time, significantly higher when compared with a random forest.

Lastly, the computation of the efficiency of the panel, for the real values and for the predicted ones, is going to be exposed in table 5.6.

<table>
<thead>
<tr>
<th>Efficiency (%)</th>
<th>Real power</th>
<th>13.13</th>
</tr>
</thead>
<tbody>
<tr>
<td>Persistence</td>
<td>12.95</td>
<td></td>
</tr>
<tr>
<td>Neural Network</td>
<td>12.15</td>
<td></td>
</tr>
<tr>
<td>ANFIS</td>
<td>12.6</td>
<td></td>
</tr>
<tr>
<td>Random Forest</td>
<td>12.15</td>
<td></td>
</tr>
<tr>
<td>kNN</td>
<td>12.95</td>
<td></td>
</tr>
</tbody>
</table>

In this situation, the real efficiency reveals to be slightly higher than the one of the AI methods. The conclusion that can be extracted from here is that, in general, the predicted power is relatively smaller than the real one, according to the equation 5.2. This event revealed to be common to all experiments, where the efficiency of the studied methods never reach the value obtained by the measurement related to the panel. Neural network and random forest show the lowest values of the efficiency, a little less than the others, which indicates that their forecasts are inferior to the other ones. None of the methods was able to reach the value of the efficiency given by the manufacturer, which is 15.27%.

5.4 Time Scale Change

In order to justify the important role of the used time scale, a different test was performed. A time scale of minutes holds the variability of the weather features and so it seemed pertinent to convert the data into hourly averages, just to make a comparison. Consequently, the data was transformed so that can be applied with a time scale of hours. The test was performed for the sunny day, where there was a
significant proximity among the values. After this transformation, the new dataset is composed for hourly average values of solar irradiance, module temperature and PV power output between 7:00 and 17:00. There are 11 “measurements” for each day. Applying the same criteria, the dataset will be splitted for training and for testing and the partition will be made, as it was for main test. The forecast will be performed for the same amount of hours, i.e. from 9:00 until 16:00, but this case will only have 9 values to be predicted. For this particular test, one decided only to implement two algorithms and compare them with the persistence. The neural network, for its well-known reputation, and the kNN, due to its performance on the carried tests, were now picked.

Results of the Time Scale Change

To emphasize the main purpose of this test, it is enough to show the time evolution of both methods and the persistence and a table where can be seen the comparison of the final MAPE.

![Time evolution of the PV power output for the time scale change](image)

Figure 5.13. Time evolution of the PV power output for the time scale change
After the graphical representation, let us expose a table where one shows the values of the MAPE obtained.

<table>
<thead>
<tr>
<th>Method</th>
<th>MAPE (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Persistence</td>
<td>49.81</td>
</tr>
<tr>
<td>Neural Network</td>
<td>1.33</td>
</tr>
<tr>
<td>kNN</td>
<td>1.41</td>
</tr>
</tbody>
</table>

The persistence is clearly worst against the other methods. The role played by the used time scale is of extreme importance. Let us remember that one is working with the sunny day and, however, the MAPE of the persistence was 49.8%! The neural network for this situation was even better than kNN. This test helped to reach some important conclusions. A small time scale improves the “strength” of the persistence method and makes more difficult to achieve a good forecast. For large time scales, the persistence becomes practically useless since its values have no major utility, because of the fact of being so different to the reality. In other words, for instance, it makes no sense to assume that the prediction, with the persistence method, for 10:00 is the real value for 9:00. There is huge difference, i.e. the size of the time scale really compromises the effectiveness of the persistence. Another thing that must be pointed out is the fact of the MAPE of the neural network being lower than the kNN’s. The existence of a reduced number of inputs and the distance between them leads to worst results for the kNN, while the MAPE of the neural network reveals to be similar to the one of the main test.

5.5 Sensitivity Analysis

This subpart’s purpose is to perform a global comparison of two key results: the simulation time and the MAPE, when faced alterations of one single parameter in the training phase. The idea of this evaluation is to see how this variation changes those assessment values, keeping all the other parameters constant. The sensitivity analysis will be implemented for all the three days previously used: the sunny day, which one will call 1st test, the partly cloudy day, the 2nd test, and the cloudy day, the 3rd test.

The procedure will be simple: for each algorithm, a single variable will suffer some modifications in order to compute the MAPE and measure the simulation time. For each case, five tests were performed, using the previously applied values, the ones referred in chapter 4, of each parameter (marked with a * in the
tables with the results) and four other values that seemed reasonable to achieve the main goal. So, the main goal is to find a possible relation between the variation of the parameters and the changes of the simulation time and the MAPE.

Starting with neural networks, it was preferred to change the number of cycles, i.e. times to run the forecasting process, as a way to see the impact caused in the results. It will be tested five different cases: for 10, 20, 30, 40 and 50 cycles, for the three days that have been analysed before. In chapter 5, it was used a number of cycles of 30, which is marked with a *.

Table 5.8. Simulation time for different number of cycles (s); ANN

<table>
<thead>
<tr>
<th></th>
<th>1st Test</th>
<th>2nd Test</th>
<th>3rd Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>229.56</td>
<td>231.76</td>
<td>4520.78</td>
</tr>
<tr>
<td>20</td>
<td>420.78</td>
<td>312.83</td>
<td>10649.81</td>
</tr>
<tr>
<td>30*</td>
<td>981.24</td>
<td>564.41</td>
<td>12650.18</td>
</tr>
<tr>
<td>40</td>
<td>1020.31</td>
<td>581.55</td>
<td>16677.22</td>
</tr>
<tr>
<td>50</td>
<td>1122.24</td>
<td>788.84</td>
<td>20398.75</td>
</tr>
</tbody>
</table>

Table 5.9. MAPE for different number of cycles (%); ANN

<table>
<thead>
<tr>
<th></th>
<th>1st Test</th>
<th>2nd Test</th>
<th>3rd Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>2.095</td>
<td>5.457</td>
<td>13.632</td>
</tr>
<tr>
<td>20</td>
<td>2.033</td>
<td>5.274</td>
<td>14.132</td>
</tr>
<tr>
<td>30*</td>
<td>1.993</td>
<td>5.238</td>
<td>13.956</td>
</tr>
<tr>
<td>40</td>
<td>2.026</td>
<td>5.247</td>
<td>13.972</td>
</tr>
<tr>
<td>50</td>
<td>2.053</td>
<td>5.285</td>
<td>13.893</td>
</tr>
</tbody>
</table>

Firstly, it was expected the increase in the simulation time with the increment of the number of cycles. The 3rd test, for the cloudy day, was extremely time-consuming compared to the other days and even more compared with the next tests, as one will notice. However, the MAPE has small variations because the number of cycles do not directly affect the performance. As it was said in the previous chapters,
creating cycles will not introduce high variations but it will only lead to reach a more accurate result.

The sensitivity analysis for ANFIS will be executed by changing the number of epochs in the training of the algorithm. It is explicit that the increase in the number of epochs will lead to a higher time required to perform the simulation. The number of epochs chosen to perform this analysis will be the number used in the main simulations, epochs (because of the fact that this number varies for each case: 6 for the 1st test, 3 for the 2nd and 7 for the last one), 10, 20, 30 and 40.

**Table 5.10. Simulation time for different number of epochs (s); ANFIS**

<table>
<thead>
<tr>
<th>epochs*</th>
<th>1st Test</th>
<th>2nd Test</th>
<th>3rd Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>60.64 (6)</td>
<td>23.25 (3)</td>
<td>96.19 (7)</td>
<td></td>
</tr>
<tr>
<td>112.88</td>
<td>103.53</td>
<td>178.39</td>
<td></td>
</tr>
<tr>
<td>221.49</td>
<td>222.46</td>
<td>337.53</td>
<td></td>
</tr>
<tr>
<td>333.31</td>
<td>360.77</td>
<td>513.56</td>
<td></td>
</tr>
<tr>
<td>477.72</td>
<td>452.94</td>
<td>714.86</td>
<td></td>
</tr>
</tbody>
</table>

**Table 5.11. MAPE for different number of epochs (%); ANFIS**

<table>
<thead>
<tr>
<th>epochs*</th>
<th>1st Test</th>
<th>2nd Test</th>
<th>3rd Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.707 (6)</td>
<td>5.057 (3)</td>
<td>10.558 (7)</td>
<td></td>
</tr>
<tr>
<td>1.720</td>
<td>5.085</td>
<td>13.109</td>
<td></td>
</tr>
<tr>
<td>1.713</td>
<td>5.200</td>
<td>14.040</td>
<td></td>
</tr>
<tr>
<td>1.834</td>
<td>5.460</td>
<td>14.295</td>
<td></td>
</tr>
<tr>
<td>6.349</td>
<td>6.349</td>
<td>14.327</td>
<td></td>
</tr>
</tbody>
</table>

Higher number of epochs do not lead to better results. Although the simulation time raised, there was a slight increase in the MAPE.

The random forest is performed by select a specific number of trees to be built during its implementation. It might be interesting to analyse how the number of trees can influence both the simulation time and
the MAPE. For this reason, the parameter chosen was the number of trees. Again, it was decided to make the test for five different quantities: the number of trees that were previously used and reported in chapter 4 for each day defined by $trees^*$ (which consists in 900 trees for the 1st test and 1500 trees for the others), and for 2000, 3000, 4000 and 5000 trees.

For this experiment, the variation of the number of trees do not cause a visible change in the MAPE. One could make an analogy with the neural network, where the increase in the parameter only rises the simulation time, not producing modifications in the MAPE.

As it is known, the only flexible parameter of kNN is the $k$, which represents the number of nearest neighbors for the resolution of each problem. Therefore, the values of $k$ that were applied in this analysis were 1, 5, 10, 15 and 20 nearest neighbors.

<table>
<thead>
<tr>
<th></th>
<th>$1^{st}$ Test</th>
<th>$2^{nd}$ Test</th>
<th>$3^{rd}$ Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>$trees^*$</td>
<td>444.96 (900)</td>
<td>453.25 (1500)</td>
<td>581.12 (1500)</td>
</tr>
<tr>
<td>2000</td>
<td>539.78</td>
<td>558.74</td>
<td>757.72</td>
</tr>
<tr>
<td>3000</td>
<td>829.21</td>
<td>789.63</td>
<td>1096.16</td>
</tr>
<tr>
<td>4000</td>
<td>1183.37</td>
<td>1110.32</td>
<td>1489.17</td>
</tr>
<tr>
<td>5000</td>
<td>1417.32</td>
<td>1330.87</td>
<td>1822.38</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>$1^{st}$ Test</th>
<th>$2^{nd}$ Test</th>
<th>$3^{rd}$ Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>$trees^*$</td>
<td>2.158 (900)</td>
<td>5.312 (1500)</td>
<td>13.631 (1500)</td>
</tr>
<tr>
<td>2000</td>
<td>2.163</td>
<td>5.313</td>
<td>13.641</td>
</tr>
<tr>
<td>3000</td>
<td>2.165</td>
<td>5.314</td>
<td>13.638</td>
</tr>
<tr>
<td>4000</td>
<td>2.163</td>
<td>5.314</td>
<td>13.635</td>
</tr>
<tr>
<td>5000</td>
<td>2.162</td>
<td>5.315</td>
<td>13.631</td>
</tr>
</tbody>
</table>
This experiment allows us to conclude that the kNN is an incredibly fast method. The increase in the value $k$ do not practically reveal changes in the simulation times but, however, leads to a deterioration of the MAPE.

**Comment of the Sensitivity Analysis**

A sensitivity analysis intends to evaluate how a certain method reacts when a parameter changes, i.e. with different training conditions. In this analysis, one chose one parameter of each of the four used techniques and attributed to them different values to see how it affected the simulation time and the MAPE.

For a neural network, changing the number of cycles was not profitable, only leading to significant
increases in the simulation time, with a MAPE approximately constant.

The variation of the number of training epochs in ANFIS produced a small increase in the MAPE with the raise of the simulation time. Choosing more epochs does not appear to be worth for the problem and so it seems to be more advantageous to work with few training epochs.

For the random forest, it is noticed something similar to the neural networks. The increase in the number of trees practically only leads to the raise of the simulation time, since one can observe a practically null variation of the MAPE among the different tests.

In the kNN, the simulation time remained constant for all the experiments. Using more nearest neighbors only led to an increase of the MAPE, not being worthy to use more than one nearest neighbor.

In the end, one was able to see the variations caused by the change of some parameters and to assess whether it was worthwhile or not to change the conditions against the required simulation time.
The importance of solar forecasting was discussed in this thesis report. The idea of making an approach from a distribution system operator (DSO) point of view initially indicated that there should be a close relation between forecasting and the operation of the power grid. Renewables are gaining strength in the electrical world, what represents a healthy advantage for the planet and its users. The interconnection between the integration of renewables into the power system and the forecasting field is led in a large part through the usage of artificial intelligence (AI) techniques. For instance, AI techniques applied to renewable power forecast can help companies to avoid unnecessary penalty payments and help them to schedule their energy transitions in the market.

The main focus of this thesis was the forecast of solar PV power output. It was chosen to apply four AI
techniques: artificial neural network, adaptive neuro-fuzzy inference system, random forest and k-nearest neighbors, in order to perform a comparison between them. The four methods have been employed in three distinct days, with different meteorological conditions. These differences served to exploit the influence, in terms of forecasting quality, supported by them. This is one of the most important issues, since a prediction is obviously influenced by external meteorological conditions. That is why one decided to implement the algorithms for a sunny day, a partly cloudy day and a cloudy day. All the algorithms were implemented in the RStudio environment. After processing the data into a suitable form to be handled, one needed to select different time intervals according to the characteristics of each day. The used dataset contained measurements, per minute, of the module temperature, the solar irradiance and the solar PV power output for the first five days of January and the first five days of August. As an attempt to adjust the time interval to the day to be tested, one decided to choose the interval between 9:00 to 16:00 for the days of January (the first and the second tests) and from 10:00 to 17:00 for the day of August, to perform a one-day-ahead solar PV power output forecast. This decision was made because of the low values of solar irradiance, and therefore solar power, outside these ranges, what could lead to relevant values of the error. In this process, it was necessary to split the data into training dataset and testing dataset, where the testing dataset consisted in the values of the previously referred intervals and the training dataset was constituted for all the remaining data, according to each test. Then, the techniques passed through an optimization stage, where one tried to find the most suitable parameters for each case. Afterwards, the implementation process was finished and it was predicted the solar PV power output for all the minutes of those intervals.

For the sunny day, it was computed a MAPE of the persistence of 0.678%, which was only higher than the kNN’s MAPE (0.677%). The other methods could not reach values lower than 1%, which, anyway, do not represent a bad value. For the partly cloudy day, with a persistence’s MAPE of 4.523%, the kNN obtained the best values and reached a MAPE of 3.688%. Neural networks, ANFIS and random forest were not able to accomplish that result and led to a MAPE around 5%. Lastly, the MAPE of the persistence was 11.196% for the cloudy day. This value showed the high variability of the cloudy day, which revealed many oscillations of the power during the day. For this case, there were two algorithms that outperformed the persistence. kNN obtained a MAPE of 4.182% and ANFIS achieved a MAPE of 10.558%. Both of the other methods could not do better than the persistence, with MAPE values close to 14%.

The simulation time of each experiment depended on the inner parameters and its ways of implementing. The kNN method, which is known for its simplicity, was the fastest technique, where each simulation ran is less than one second. ANFIS obtained quite good results and did not require more than one and a half minute. Neural networks and the random forest required more time since one decided to create cycles to reach a more reliable result. Besides being the algorithm that requested more time, approximately 16 minutes for the first test, 10 minutes for the second and more than two hours for the last test, neural network did not achieved the best results. On the other side, random forest did not need more than 10 minutes for none of the tests.

The AI methods that had been used really revealed to be worth for forecasting. Depending on each
case, i.e. depending on the type of day, the urge for the predictions or the available resources, one can now decide which way to follow and which method to implement. For instance, a large dataset and an urgent request would get us to choose kNN, since it is a fast method that can provide reliable predictions. Otherwise, one would recommend to use ANFIS because it will achieve reasonable forecasts and it is not expensive in the matter of time. However, the type and the number of the membership functions must be chosen in agreement with the dataset. Although our choice was to use ANFIS for a more robust artificial intelligence algorithm, it would also be possible to obtain good results with the use of the neural networks and the random forests, not despising the power of these last algorithms. One possible disadvantage of these last two methods is that the results obtained by them may vary, sometimes a lot, for different simulations under the same conditions and may also require a large amount of time.

There are some actions that can be done as an attempt to improve the global results. Starting from the methods themselves, the inner parameters could be modified so that better values can be found. There are some ways to achieve that, such as the use of some packages of R or the implementation of some control functions. Besides these facts, if one could introduce some meteorological features into the problem, maybe one could obtain better results, especially if these variables can be correlated to cloudiness. Cloud cover is a crucial meteorological characteristic among forecasting communities. Dealing with it represents one of the biggest actual challenges, since its unpredictability highly affects the quality of a forecast. The error of a prediction got higher when faced to the existence of cloudiness. Some atmospheric variables can allow to predict the state of the sky at any time, and that represents an advantage for the main topic. Variables such as air pressure, air temperature or air density may help to understand and determine future conditions of the sky. New variables would let the learning process of an AI technique to be more aware of how the final output would react to more specific weather changes. The inclusion of some of these atmospheric variables can be something interesting to try in the future and to evaluate the influence carried by this action. The quality of a forecast can be improved through other ways. It is possible to combine AI with a diverse range of methods, going from satellite approaches to the integration of other forecasting techniques, for instance the use of genetic algorithms, kernel filters and much more. What one is trying to say is that the possibility to create a hybrid system may lead to better results. The combination of more than one method will bring new capabilities into the forecasting process. In the future, trying to combine AI algorithms with different kinds of techniques also seems an interesting subject to work on.
Annex A

Specifications of the JKM250P PV module

This annex contains the specifications, from the manufacturer, for the type of module that was used. The dataset that allowed the realization of this work is prevenient from measurements performed in panels with this module type.
Figure A.1. Specifications of the manufacturer of the PV panel (1)
Figure A.2. Specifications of the manufacturer of the PV panel (2)
Annex B

Sensitivity Analysis Graphics

Annex B has the purpose of showing the graphical evolution of the real and the forecasted power for all the experiments performed in the sensitivity analysis. Therefore, the graphics that are going to be shown here will contain the power comparison for each test performed, with the exception of the main tests, whose evolutions were already showed in the 5th chapter.
Figure B.1. Power evolution for the 4th of January for 10, 20, 40 and 50 cycles (NN)
Figure B.2. Power evolution for the 4th of January for 10, 20, 30 and 40 epochs (ANFIS)
Figure B.3. Power evolution for the 4th of January for 2000, 3000, 4000 and 5000 (RF)
Figure B.4. Power evolution for the 4th of January for 5, 10, 15 and 20 nearest neighbors (kNN)
Figure B.5. Power evolution for the 5th of January for 10, 20, 40 and 50 cycles (NN)
Figure B.6. Power evolution for the 5th of January for 10, 20, 30 and 40 epochs (ANFIS)
Figure B.7. Power evolution for the 5th of January for 2000, 3000, 4000 and 5000 (RF)
Figure B.8. Power evolution for the 5th of January for 5, 10, 15 and 20 nearest neighbors (kNN)
Figure B.9. Power evolution for the 3rd of August for 10, 20, 40 and 50 cycles (NN)
Figure B.10. Power evolution for the 3rd of August for 10, 20, 30 and 40 epochs (ANFIS)
Figure B.11. Power evolution for the 3rd of August for 2000, 3000, 4000 and 5000 (RF)
Figure B.12. Power evolution for the 3rd of August for 5, 10, 15 and 20 nearest neighbors (kNN)
References


