Machine Learning Applied to Wind and Waves Modelling

Aprendizaje Automático Aplicado al Modelado de Viento y Olas

Author:
Montserrat Sacie Alcázar

Bachelor's Degree Final Project in Software Engineering
Trabajo de Fin de Grado en el Grado de Ingeniería del Software

UNIVERSIDAD COMPLUTENSE
MADRID

Facultad de Informática

Director:
Dr. Matilde Santos Peñas
Co-director:
Dr. Rafael López Martínez

Madrid, Academic Year 2019/2020

Document created with LaTeX
prepared to be double-sided printed

A mi familia. Este TFG y yo
llevamos un poquito de vosotros

Agradecimientos

Quiero dar las gracias a mi tutora Matilde Santos Peñas por confiar en mí, introducirme en el mundo de la energía eólica marina y guiarme en la elaboración del TFG y a mi co-tutor Rafael López Martínez por ofrecerme su conocimiento sobre los datos y estar siempre disponible para resolver mis dudas. Sin vosotros, este TFG no hubiera sido posible.

No puedo cerrrar este TFG sin antes emocionarme al dar las gracias a mis padres por darme la vida e inculcarme su fortaleza, motivación, constancia y trabajo duro necesarios para conseguir cualquier objetivo que me proponga en la vida. A mi hermano por ser mi referente y enseñarme lo que sabe por experiencia cuando yo aún no sé que camino escoger. Gracias por tu silla también. A mi hermana por su apoyo durante todo el proceso y la tranquilidad que me transmite.

Doy las gracias a todos mis seres queridos y amigos por confiar siempre en mí y hacerme el camino más llevadero. Y gracias a los profesores de instituto de los que no dejo de acordarme, en especial al profesor de física Marcos Naz que me infundó la pasión por la investigación científica y las ganas infinitas de aprender.

Por último, quiero dar las gracias a mis profesores del grado de Ingeniería del Software y compañeros de clase por su apoyo y generosidad que hacen que la experiencia universitaria sea realmente bonita y enriquecedora.

Abstract

In the fight against climate change, Offshore wind energy is at the forefront, in the development phase. The problem with turbines anchored to the seabed lies in the enormous cost of installation and maintenance, leading to the theoretical approach of floating offshore wind turbines. However, floating turbines are exposed to new wave loads and stronger wind loads. To enable their implementation while maximizing the electricity production and ensuring the protection of the structure, more accurate predictive models than the physical and statistical ones found in the literature are needed for the metocean (meteorological and oceanographic) variables involved.

This project aims to model the wind speed in the time domain, the significant waves height in the frequency domain and the misalignment between wind and waves direction in the time domain, applying Machine Learning techniques.

Offshore data collection as well as an exploratory data analysis and data cleaning phases have been carried out. Subsequently, the following algorithms were applied to train the models: Linear Regression, Support Vector Machines for Regression, Gaussian Process Regression and Neural Networks. Nonlinear Autoregressive with exogenous input neural networks (NARX) have proved to be the best algorithm both for wind speed and misalignment forecasting and the most accurate predictive model for significant waves height prediction has been the Gaussian Process Regression (GPR).

In this project we demonstrated the ability of Machine Learning algorithms to model wind variables of a stochastic nature and waves. We emphasize the importance of evaluating the models through techniques such as Learning Curves to make better decisions to optimize them. This work not only makes predictive models available for later use, but it is also a pioneer in misalignment modelling, leaving a door open for future research.

Key words

- Data cleaning
- Exploratory Data Analysis
- Floating Offshore Wind Turbines
- Machine Learning
- Misalignment
- Optimization
- Regression
- Waves models
- Wind energy
- Wind models

Resumen

En la lucha contra el cambio climático, la energía eólica marina se sitúa en cabeza encontrándose en fase de desarrollo. El problema de las turbinas ancladas al lecho marino reside en el enorme coste de instalación y mantenimiento, llevando al planteamiento teórico de turbinas eólicas marinas flotantes. Estas, sin embargo, están expuestas a nuevas cargas de olas y cargas de viento más fuertes. Para hacer posible su implantación maximizando la producción eléctrica a la vez que asegurando la protección de la estructura, se necesita disponer de modelos predictivos más precisos que los físicos y estadísticos de la literatura para las variables metoceánicas (meteorológicas y oceánicas) implicadas.

El objetivo de este proyecto es modelar la velocidad del viento en el dominio del tiempo, la altura significativa de la ola en el dominio de la frecuencia y la desalineación entre la dirección del viento y de las olas en el dominio temporal, aplicando técnicas de Aprendizaje Automático.

Se ha llevado a cabo una fase de recopilación de datos medidos en alta mar, así como el análisis exploratorio y limpieza de los mismos. Posteriormente, para el entrenamiento de los modelos se aplicaron los algoritmos: Regresión Lineal, Máquinas de Vectores Soporte para Regresión, Proceso de Regresión Gausiano y Redes Neuronales. Las redes neuronales autorregresivas no lineales con entrada externa (NARX) han resultado ser el mejor algoritmo tanto para la predicción de la velocidad del viento como para la desalineación y para la altura significativa de la ola el modelo predictivo más preciso ha sido el proceso regresivo gausiano (GPR).

En este proyecto demostramos la capacidad de los algoritmos de Aprendizaje Automático para modelar las variables del viento de naturaleza estocástica y del oleaje. Destacamos la importancia de la evaluación de los modelos mediante técnicas como las Curvas de Aprendizaje para tomar mejores decisiones en la optimización de los mismos. Este trabajo no pone solo a disposición modelos predictivos para su posterior uso, además es pionero en el modelado de la desalineación dejando una puerta abierta a futuras investigaciones.

Palabras clave

- Análisis Exploratorio de Datos
- Aprendizaje Automático
- Desalineación
- Energía eólica
- Limpieza de Datos
- Modelos de olas
- Modelos de viento
- Optimización
- Regresión
- Turbinas eólicas marinas flotantes

Contents

List of Acronyms

ADALINE Adaptative Linear Element referred to an ANN model variant

ANN Artificial Neural Networks. Also denoted by 'NN' (Neural Network)

AR Autoregressive model

ARIMA Autoregressive integrated moving average model

ARMA Autoregressive-moving average model

EDA Exploratory data analysis

FFBP Feed Forward Back-Propagation referred to an ANN model variant

FOWT Floating Offshore Wind Turbine

GP Gaussian Process

GPR Gaussian Process Regression

Iqr Interquartile range

LDOF Local Distance-based Outlier Factor

MA Movil average model

Metocean Meteorology and Oceanography

ML Machine Learning

NAR Nonlinear Autoregressive neural network

NARX Nonlinear Autoregressive with External (Exogenous) Input neural network.

NRA Seawater Monitoring in Off-shore Sea Area

NWP Numerical Weather Predictions

PCA Principal component analysis algorithm

RBF Radial Basis Function referred to an ANN model variant

SVM Support Vector Machine

SVR Support Vector Machine for Regression or Support Vector Regression

List of Figures

List of Tables

Chapter 1

Introduction

Nowadays, one of the greatest global challenges the society is facing is Climate Change. As The United Nations Framework Convention on Climate Change (UNFCCC) recognized [1], the governments worldwide must bet on the development, application and diffusion of new technologies which will promote the exploitation and use of renewable energies, reducing the emissions of greenhouse gases and mitigating the global warming [2], [3].

In contrast to Onshore (On-land) Wind Energy, an already mature technology [4], Offshore Wind Energy has appeared in the last two decades as a promising solution for some of the disadvantages of the on-land wind turbines. This allows us to take advantage of stronger and more constant winds that are produced in the open sea due to the absence of geographical accidents that stop them as on land. The US Department of Energy (DOE) claims that Offshore wind has the potential to generate more than 2.000 GW of capacity per year which is nearly double the current electricity generated in USA [5].

However, the big problem is the cost of installing the turbine offshore and the cost of its maintenance. Numerous research lines have been opened looking for optimal location and orientation for the installation of the turbines anchored to the seabed.

In the interest of reducing installation costs and increasing energy production capacity, emerge the idea of installing offshore wind farms in deeper waters resulting in the design of **the new Floating Offshore Wind Turbines (FOWTs)**. FOWTs expands the area for wind energy development, taking advantage of stronger and steadier winds than near the coast and reducing the visual impact and noise pollution caused by OWT near the coast and on-land turbines [6]. Although their installation is simpler, they pose new challenges to be faced: the buoyancy of the turbine and the structure control.

Optimal blade, tower and floating platform control systems are needed to overcome wind and wave loads and minimize fatigue to which turbines are exposed [6].

To collaborate with current projects involved in this recent technology in increasing development, the present project is proposed:
"Machine Learning Applied to Wind and Waves Modelling".

1.1 Motivation

Machine Learning is a discipline of Artificial Intelligence whose potential lies in the processing of Big Data and the ability it gives to computers to learn rules and patterns inferred from raw gathered data. The need to optimize decisions, e.g. structure design decisions in scientific projects, considering those rules and behaviors observed in the data can not be covered by humans and conventional models. Therefore, Machine Learning techniques are revolutionizing the way we work with the available data and it is starting to be applied more and more in practically all fields of research.

The problem that appears with wind energy is the need to forecast wind power generation and therefore the wind speed on which it depends. In particular, floating offshore wind turbines (FOWTs) are exposed to even stronger winds and ocean conditions too. The stability of FOWTs is affected significantly more than if they were anchored to the seabed by the impact of new loads from waves and currents they are subjected to. This requires further studies on meteorological and oceanic (metocean) conditions to reduce the maintenance costs and apply the results in the optimal control of power generation combined with the damping of new oscillations generated in the floating platform, highlighting four main types of platforms as seen in 1.1 in the quest to create reliable FOWTs.

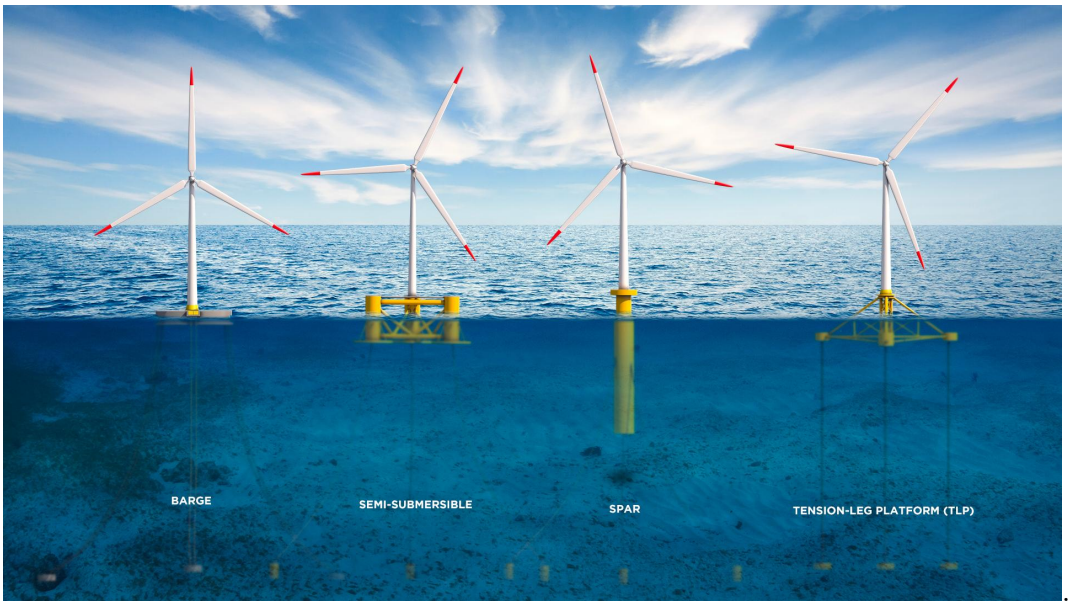


Figure 1.1. *Typical floating wind turbines platforms types*

Models with good performance for wind and waves forecasting are essential, but very few acceptable results are available in the literature for floating wind energy requirements. Besides, authors and researchers involved in wind and waves modelling have considered constant wind direction values (based on statistic probabilities from the concrete location for a potential wind farm construction) and null misalignment between wind and waves. Intuitively it is logical to think that misalignment is null since the wind is the main source that propitiates the generation of waves. Notwithstanding, some recent researches have observed that wind and waves direction are significantly misaligned most of the time

and they increase the overall loading on the support structure so this as yet unknown phenomenon has a greater impact on FOWTs than any previous author might have imagined.

In this work, we have carried out an analysis of wind and waves data and we apply different Machine Learning techniques to model wind and waves features implicated in the FOWT operation. This project is also a pioneer in the study of misalignment and its modelling. The results may be useful in projects involved in the optimization of turbines control strategies while producing more energy, achieving the offshore wind energy to being a reliable energy source and bringing governments closer to integrating floating offshore wind farms into electricity grids.

1.2 Goals

A literature research work and the development of wind, waves and misalignment between wind and waves models with Machine Learning as well is carried out. This study is oriented to its application on floating offshore wind turbines, thus this will be taken into account, e.g. for the station selection from which the data is collected.

In particular, the objectives of the project are set out below:

- Study of the literature on the treatment and analysis of wind and waves data.
- Study of wind, wave and misalignment, if developed, previous modelling approaches as well as techniques used.
- Selection, comprehensive analysis and preprocessing of Metocean historical data.
- Modeling of wind and waves in the time and frequency domain by the application of Machine Learning techniques
- Models comparison.
- Presentation of the results obtained in each phase of the project.

1.3 Work plan

A work plan was designed at the beginning of the project as an indicative work guide. Figure 1.2 shows every task with an assigned start and end date. Some tasks are concurrent.

Task name	Start date	Completion date	Duration (days)
Bachelor's Final Project Degree	**15/09/2019**	**07/09/20**	**358**
State of Art - Research	15/09/2019	15/10/2019	30
Documentation (memory)	15/10/2019	30/10/2019	15
Part I: Data preparation	30/10/2019	30/05/2020	213
Data bases searching	30/10/2019	15/11/2019	16
Wind and waves features study	15/11/2019	30/12/2019	45
Do Mathwork courses for Data processing	15/11/2019	30/12/2019	45
Data model design	18/11/2019	30/12/2019	42
Documentation (memory)	30/12/2019	10/1/2020	11
Data analysis and cleaning	10/1/2020	30/5/2020	141
Documentation (memory)	10/1/2020	30/5/2020	141
Feature selection study	30/5/2020	5/6/2020	6
Documentation (memory)	5/6/2020	10/6/2020	5
Part II: Modelling	10/6/2020	15/8/2020	66
Do Mathwork courses - Initial learning	10/6/2020	15/6/2020	5
Wind speed Forecasting	10/6/2020	10/7/2020	30
Models training and optimization	10/6/2020	10/7/2020	30
Sig.Waves Height prediction	10/7/2020	10/8/2020	31
Models training and optimization	10/7/2020	10/8/2020	31
Misalignment modeling	10/8/2020	5/9/2020	26
Models training and optimization	10/8/2020	5/9/2020	26
Results analysis and conclusions	10/7/2020	7/9/2020	59
Documentation (memory)	10/7/2020	6/9/2020	58
Uploading code and used materials to repository	3/9/2020	6/9/2020	3
Draft Submission	7/9/2020		
Final Submission	09/09/2020		

Figure 1.2. *Project Work plan*

1.4 Repository

The code generated during the project development is shared in a public Github repository which can be accessed at the following link: https://github.com/MontseSacie/Machine_Learning_Applied_to_Wind_and_Waves_Modelling.

1.5 Structure of the project

To describe the work carried out in each phase of the project, we have divided the memory into the following chapters:

- Chapter 1 picks up the introduction where we exposed the motivation behind this research, the initial objectives and the work plan to get them.
- Chapter 2 is the State of Art. It summarized the research context around Offshore wind energy and previous results about wind and waves forecasting found in the literature.
- Chapter 3 (Materials and methods) is a theoretical chapter where we explain algorithms and methods used in data preparation and modeling phases.
- Chapter 4 describes the Data collection process, justifying the selected location of buoy station from where we gather the data as well as explaining the data set variables meaning. This chapter together with chapter 5 and 6 form the data preparation phase.
- Chapter 5 includes the Exploratory Data Analysis techniques and data cleaning methods applied to the data set. Exploratory Data analysis pretends to get an understanding of data and data cleaning is applied to delete missing values and outliers.
- Chapter 6 explains how we have structured data sets and which variables include each one for model training.
- Chapter 7, 8 and 9 present the modeling results for wind speed, significant waves height and wind-waves misalignment forecasting respectively. In each chapter different Machine Learning techniques applied are described and we finish each chapter with a section where we compared different algorithms' performance and decide which is the best.
- Chapter 10 presents the conclusions of the project and future work.

After the bibliography, the following parts of this document translated into Spanish are placed as an appendix:

A. Introduction

B. Conclusions and Future work

Chapter 2

State of art

The problem faced in this project is analyzing and modelling wind and wave loads by Machine Learning techniques and study how they affect the FOWTs motion.

We divide it into three parts:

- The study of wind as a perturbation in the time domain. It is intended to forecast Wind Speed in the time domain.
- The study of waves as a perturbation in the frequency domain. We will train models to predict Wave Significant Height.
- The study of misalignment between wind and waves. Models to forecast the misalignment in the time domain will be created.

2.1 Research context

The first offshore wind park was installed in Vindeby(Denmark) in 1991 and it was composed of 11 turbines installed at the distance from shore of 2 meters, with a power of 450 kW each one resulting in a total capacity of 4.95 MW [7]. Since then, offshore wind farm deployment has grown exponentially, in an initial phase in North and Baltic seas and progressively in new markets outside Europe [8].

According to annual statistics released by WindEurope, the European wind energy association, Europe connected 502 new offshore wind turbines to the grid across 10 projects which increased the energy capacity in 3627 MW. Figure 2.1 shows a Europe new's record off Offshore Wind installations in 2019 translated into 3,6 GW of new wind capacity. Inside Europe, The UK has incorporated almost half of new capacity ($47,2\%$) followed by Germany ($30,5\%$), Denmark($10,4\%$) and Belgium ($10,3\%$). We can mention Portugal installed this past 2019 8,4MW of offshore wind energy while Spain installed two turbines corresponding to 5 MW of new net capacity [9].

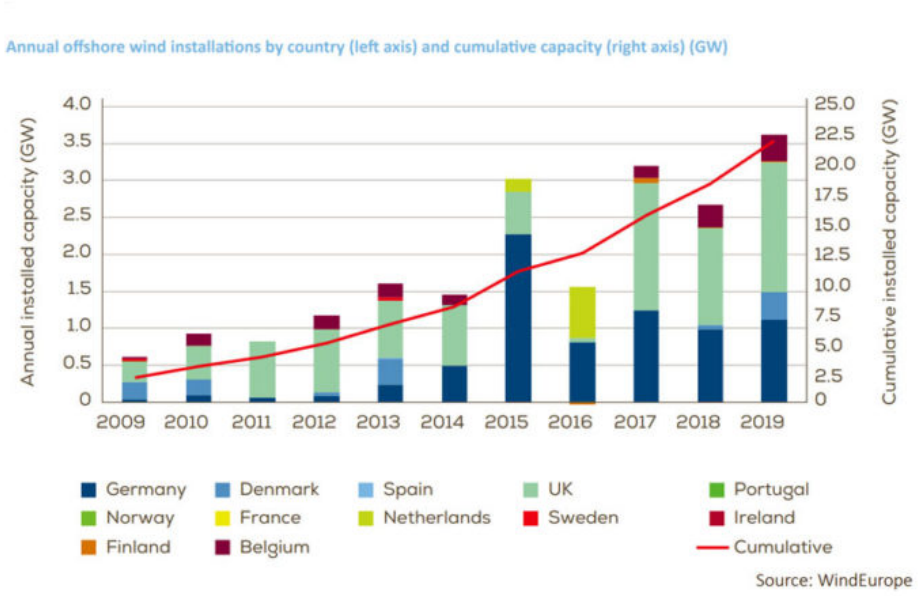


Figure 2.1. *New wind farms installed in Europe from 2009 to 2019. Picture downloaded from WindEnergyReview*

The International Renewable Energy Agency (IRENA) estimates that Asia would dominate global offshore wind power installations by 2050, followed by Europe and North America as seen in figure 2.2 [10].

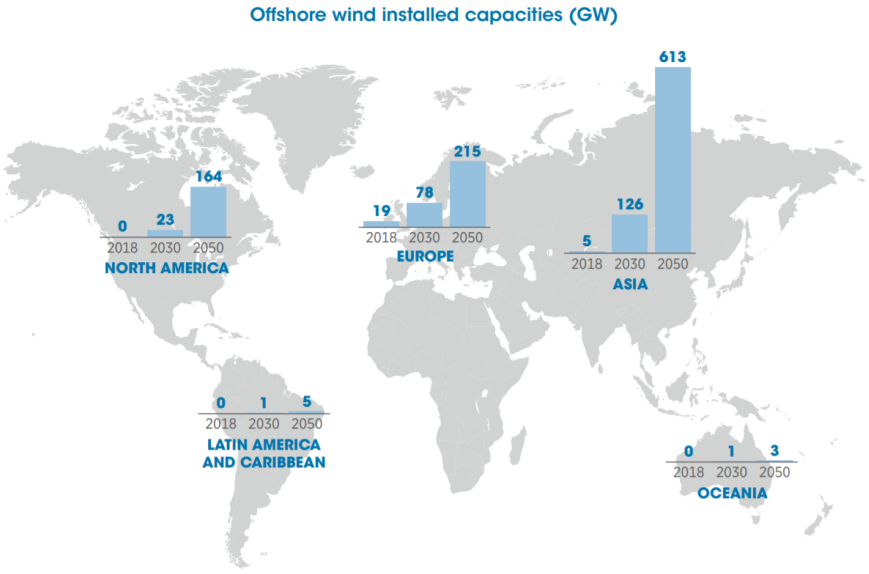


Figure 2.2. *Global offshore wind installed capacity evolution estimated until 2050 by IRENA statistics. Image obtained from Irena-offshore-wind-statistics.pdf*

In the course of the growth of offshore wind energy, the first 80 KW floating wind turbine was deployed off the coast of Apulia (Italy) in 2007 by Blue H Technologies of the Nether-

lands. It was a prototype with a tension-leg platform and was operative during a year gathering wind and waves sea conditions data. At present, the ambition of governments to integrate more wind energy generation in the national power grid does not stop growing and floating offshore wind energy is a promising technology in which almost every country is investing. Highlights the world's first floating wind farm, the 30-MW Hywind Scotland Pilot Park, under development which has five turbines. There are still no integrated and operational floating wind farms. This recent technology is still immature and not yet consolidated in society, so many investigations are still open.

2.1.1 Lines of study

Simulations and studies involved in technology advance for floating offshore wind energy may be framed in four areas:

- Platform designing
- Control Strategies planning
- Aero and hydrodynamic modelling
- External wind and waves loads modelling.

All line of works are connected and collaborate to get reliable turbine prototypes since results and decisions in one area may affect the decisions from another as seen in figure 2.3.

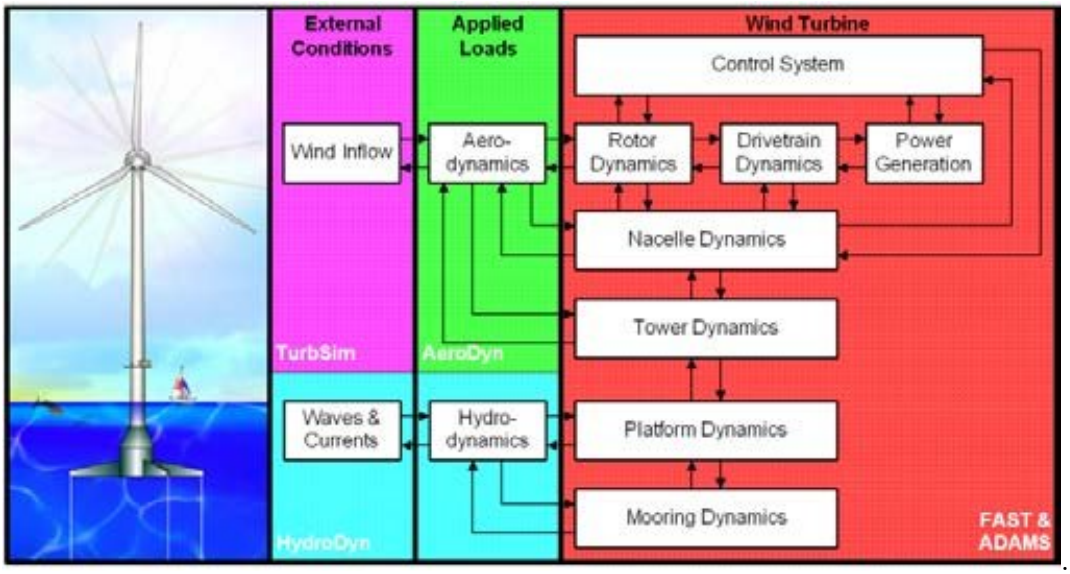


Figure 2.3. *Components and dynamics present in the model of a FOWT. Downloaded from researchgate.net*

The transition from the installation of turbines in shallow waters of 20 meters deep to 20-40 km off the coast raises the need to design a floating structure that will limit the

turbines foundation costs [11]. During platform designing, it must be considered the buoyancy requirement so a first-order static stability analysis will partially determine the architecture selected for the platform. More decisive factors studied are the mooring line system selected to prevent the platform from drifting, the installation difficulty analysis, the maintainability as well as the corrosion resistance. The corrosion is caused by continuous-wave loads crashing against the platform. Moreover, the turbine weight, tower top motion and control system might generate vibrations on the platform so a thorough study of platform movements in the 6 degrees of freedom is carried out. Thus the expected movement ranges and damping mechanisms to dissipate the possible vibrations are established. Figure 1.1 from previous chapter showed the four main structures designs: the barge platform consisted on a floating structure which achieve stability through the use of distributed buoyancy, the *semi sumergible* one, the *spar* structure formed by an underwater monopole and the *tension leg platform* (TLP). The first three are loosely moored to the seabed and their installation is easier while the TLP is more firmly connected to the seabed and it increases the stability of structure [12].

The second area listed includes blade pitch, generator torque and tower damping control strategies. The first two are called active control systems because they consist of taking decisions in real-time. Blades control tries to regulate the angle of the blades according to wind speed and wind direction so for higher wind speeds blades are turned to decrease the sustainability of the blades and vice versa. The objective is maximizing electricity production as time as protecting blades to be twisted. The rotor and generator are also regulated to prevent high wind speeds damaging it and overloading the electric transformation system. In the case of tower damping control, we are talking about passive control. The vibrations generated in the tower by loads in fore-aft and side-side directions are dissipated by a mass damping system calibrated before being collocated in the nacelle of the turbine.

To optimize control strategies and reduce the fatigue produced in the structure, the aerodynamic and hydrodynamic to which the different parts of the turbine are exposed must be analyzed and they must include metocean conditions [13]. So wind and waves loads must be modelled. This project is therefore part of the fourth presented research line.

2.2 Wind forecasting

2.2.1 Motivation

Given the stochastic and intermittent nature of wind, numerous authors and researchers have employed their efforts to develop the best wind predictive models.

Wind power prediction models, dependent of wind speed forecasting, have been carefully studied [14] as these predictions are given to the power system regulators which must make detailed scheduled plans for energy supply. Wind power generation P is determined fundamentally by wind speed $v(m/s)$ as seen in equation (2.1) where ρ is density of

air(kg/m^3) and A is the swept area of the wind turbine.[15].

$$P = \frac{1}{2}\rho A v^3 \tag{2.1}$$

Thus, we will review in this section **models** developed to forecast **wind speed**.

2.2.2 Time-scales of predictions

A point of interest in wind forecasting is the moment in time covered by predictions made with a model. Thus, forecasting models can be separated into four categories[16]:

(1) **Very short-term forecasting**. Models give wind predictions from 30 s to 30 minutes ahead the moment of forecasting. Applications inside this category are related to electricity market clearing and regulatory actions.

(2) **Short-term forecasting** Predictions are made with data from 30 minutes to 6 hours ahead. These models are oriented to help the Economic Load Dispatch planning and taking load increment/decrement decisions.

(3) **Medium-term forecasting**. Models are accurate while forecasting wind conditions of a time moment from 6 hours to 1 day ahead. Some applications of medium-term wind forecasting are taking generator decisions and planning operational security strategies for the electricity market a day ahead.

(3) **Long-term forecasting**. Predictions from 1 day to 1 week or more ahead are obtained by models classified in this category. These models are essential for the strategy of Maintenance and reserve requirements decisions.

In this project we are going to train models for short-term forecasting (1 hour ahead of the forecasting moment).

2.2.3 Models classification

According to type of techniques employed, Metocean forecasting models can be separated into *(1) Naive,(2) physical*, *(3) statistical* and *(4) intelligent* models.

2.2.3.1 Naive models

Persistence Method also called the "Naive method" [16] is the reference method in industrial applications. It consist of the assumption that the wind speed at time $t + \triangle t$ is the same as it was at time t. Despite its simplicity, it is really effective for very short-term and short-term predictions and continues to be used in some cases when its

performance is good enough and other more complex physical and statistics methods do not achieve a significant improvement [17]. Thus this model is considered in literature the benchmark model to compare the performance of later models with so after evaluating the increasing/decreasing of RMSE or MSE to the persistence one, just in the decreasing case the new model is said to be good enough and can be a substitute [18].

2.2.3.2 Physical models

Physical models are based on mathematics and physical considerations like terrain, obstacle, pressure or temperature to estimate wind speed. They involve simulating variations in wind flow throughout the farm to estimate its speed and direction at each generator[19].

In this category class, **NWP** (Numerical weather predictions) and mesoscale models stand out [20]. They are so antique since weather forecasting have been applied in many fields such as weather information for humanity by meteorologists or first wind applications [21]. NWP models solve complex mathematical functions dependent on weather data like temperature, pressure, surface roughness and obstacles. Generally, the main drawbacks of these models are computational complexity and susceptibility to unstable weather. Moreover their performance is good for long-term predictions . Table 2.1 lists some physical NWP models.

Model	Developer	Comments
GFS	NCEP	- Predicts weather 16 days ahead - Formed by 4 models: atmosphere, ocean, land/soil and ice models -It is usually re-optimized and works on data from NOAA.
ECMWF model	ECMWF	- Beside the numerical weather forecasting, it monitors planetary systems that influence weather. - Time horizon of predictions: medium range (15 days) and large range (30 days)
HARMONIE-AROME	AEMET	- It has a 48-hour time frame -The available surface variables are: temperature, pressure, wind, precipitation, cloudiness, electrical discharges and maximum streak.

Table 2.1. *Physical Models examples*

2.2.3.3 Statistical models

Statistical models do not need a physical model of the farm. Instead, their background is formed by statistical distributions and algorithms. They are designed by curve fitting

and their model is adjusted by comparing the actual data and the immediate past and predicted one, which make them effective for short-term forecasting [22].

The stochastic behavior of the wind leads us to define it as a set of random variables v_t representing the speed for every moment in time t. Thus statistical models try to find patterns between these variables from historical data. Statistical models are subdivided into time-series models and neural networks (NN) for time-series distribution prediction. Auto-Regressive Movil Average **ARMA** models are the main time-series based statistical models accurate for very short-term and short-term predictions. They are formed by one **AR** (Autoregressive) model and one **MA** (Movil Average) model [23] .

An AR model explain the current value of a variable in moment t as a combination of some previous values for this variable plus an error term:

$$y_t = a_0 + a_1 \cdot y_{t-1} + ... + a_p \cdot y_{t-p} + e_t$$

where a_i are coefficient models and e_t is the "white error". The order p of the equation marks the number of predecessor values needed for prediction. Authors of [24] proposed an AR model considering the non-normal wind speed distribution and proposing use separating monthly data from several years of wind time series.

In MA models, the value of the variable to be estimated depends on an independent term μ and a weighted sequence of q errors (q is the equation order) corresponding to the predecessors as is formulated in equation (2.2). θ_i are models parameters and e_{t-i} is the error in the $t - i$ moment.

$$y_t = \mu - \theta_1 e_{t-1} - \theta_2 e_{t-2} - ... - \theta_q e_{t-q} + e_t \tag{2.2}$$

Therefore, ARMA model arises from joining the AR and MA model formulated below (Note that the coefficient μ doesn't appear because it has been joined to a_0):

$$\text{ARMA}(p, q) = y_t = a_0 + a_1 \cdot y_{t-1} + ... + a_p \cdot y_{t-p} + e_t - \theta_1 e_{t-1} - \theta_2 e_{t-2} - ... - \theta_q e_{t-q}$$

In [25], authors used an ARMA model to average hourly wind-time series forecasting obtaining that error compared to the persistence model was 1% better for 1 hour ahead predictions and between 12% and 20% for $10-12$ h ahead forecasts. While ARMA models assume that the data underlying process is stationary (median and variance are constant over time) and autocovariance depends on the time lag, it has been observed that many time series are not stationary so they tend to show time-changing means and/or variances. In this case, the variant of the ARMA model, **ARIMA** (Auto-Regressive Integrated Moving Average) model is used. It is a non-stationary model also called an "integrated process" [26]. To deal with nonstationary, ARIMA makes a simple transformation. Given the time serie characterised by a non constant mean plus an error:

$$x_t = \mu_t + \varepsilon_t$$

If the mean is decomposed with constant parameters through time plus a white error as follows

$$x_t = \beta_0 + \beta_1 t + a_t$$

and we subtract to it $x_{t-1} = \beta_0 + \beta_1(t-1) + a_{t-1} = \beta_0 - \beta_1 + \beta_1 t + a_{t-1}$ we obtain a stationary model:

$$x_t - x_{t-1} = \beta_1 + a_t - a_{t-1}$$

Another ARMA variant is the **ARX or ARMAX** model which is an ARMA model with a exogenous input variable. Equation (2.3) shows the ARX model estructure, explained in detail in bibliography [27].

$$y(t) + a_1 y(t-1) + ... + a_{na} y(t-n_a) = b_1 u(t-n_k) + ... + b_{n_b} u(t-n_b-n_k+1) + e(t) \quad (2.3)$$

2.2.3.4 Intelligent Models

Artificial Intelligence-based models are the most recent and last models authors have focused on. They include the use of methods of artificial intelligence such as Fuzzy Logic or Machine Learning methods like artificial Neural Networks (ANN), Support Vector Machines (SVM), Random Forests and Gaussian Process Regression (GPR). They use historical data to train and tune the model parameters instead of having a predefined model as statistics one. Their objective is minimizing the error calculated by the difference between actual and predicted wind speed of training data [19]. One of the decisive reasons for applying Machine Learning techniques is that prior models assumptions, such as normality, linearity and homogeneity took in statistical methods are not necessary [28] making easier to non experts on data domain apply ML algorithms.

A GMDH-based abductive neural network model for mean hourly wind speed time-series forecasting in May month is proposed in [18] as an ANN approach that outperforms the benchmark persistence model accuracy [29]. The Group Method of Data Handling (GMDH) is an application based on an iterated polynomial regression that is able to produce a high-degree polynomial model with significant model predictors. One of the benefits of abducted networks versus other network variants is the faster model development, the convergence without the risk of falling into a local minimum and almost no intervention of user by automating input data selection. [18]. In [30], a comparison between Adaptative Linear Element (ADALINE), Radial Basis Function(RBF) and Feed Forward Back-propagation (FFBP) neural networks(NN) in 1-hour ahead wind speed forecasting is done. A back-propagation (BP) NN is obtained by its authors as the most optimal model experimented with an RMSE of 1,254.

In [31], Traybeb Brahimi and his team propose an ANN model to predict daily wind speed with meteorological measurements ATMP, WDIR, GHI (Global Horizontal Irradiance), Relative Humidity and PRES as input features selected between the 13 attributes available in their data set. The same authors [31] also applied Random Forest, Random Tree, and SVM techniques to compared every model trained with a cross-validation scheme. The optimum network they obtained with one hidden layer and 30 neurons gives them an RMSE of 0.8078 with the optimal cv-scheme: 70% data examples for training and 30% for validation. Comparing the five algorithms, they found out the random forest presents higher performance than the rest of the algorithms [31]. In the same line, Senthil Kurman compares some variants of ANN namely Back Propagation Network (BPN), Radial Basis Function(RBF) and Nonlinear Autoregressive neural network with exogeneous inputs (NARX) ([32]).In his paper, Senthil remarks on the importance of the feature selection

process and his results confirm it: RBF model without previous feature selection technique application obtained an RMSE of 1,2963 and nevertheless the RMSE boils down to 0,7040 by applying the Mutual Information (MI) technique before RBF training. The NARX model has utilized for the time series oriented data composed by reflected shortwave radiation, wind direction, ambient air temperature, relative humidity, shortwave radiation and wind speed in $t-1$, $t-2$ and $t-3$ time delays features. For BPN and NARX models although their errors are greater, it is significantly lesser when applying MI feature selection.

Breaking with the above, authors in [22] address the Long-term wind speed forecasting by chained patterns-based models that form a forecasting system.

Concerning other ML algorithms, some researchers have explored Support Vector Machines as a modern promising ML technique. Particularly relevant are the results and tests during modelling carried out in [33]. Authors of this paper have applied a faster computationally variant of SVM, Least-Squared Support Vector Machine (LS-SVM) technique for short-term wind speed forecasting. It's striking how they have trained and optimally configured independent models for each season of the year which can be a good practice to get more accurate models. Thus they obtained RMSEs of 1,650 (Spring), 1,209 (Summer), 1,584 (Fall) and 1,216 (winter).

2.3 Waves forecasting

2.3.1 Motivation

Metocean conditions affect the stability of the FOWT to a greater degree than if they were anchored to the seabed. However, the effect of waves is not usually considered, though they are the main source of disturbances for floating structures. In the best cases, the wave model is simplified just defining sea states (SNN) based on wind speed.

Waves forecasting has been traditionally applied in Weather systems forecasting [34], shipping and naval engineering. Some results from the literature are the start point in offshore wind energy research. Moreover, waves energy is considered one of the most promising renewable energy resources. Thus waves energy converters (WECs) have been studied during the last three decades and numerous researchers groups study the sea state for waves power production and waves forecasting applied to these systems [35]. In fact, *WindWave* is a current national research project in development which studies the combination of a floating wind turbine and a WEC to generate energy [36]. In this project which arises from the collaboration between the Complutense University of Madrid and País Vasco University, the tutor and co-tutor of this final degree project are part of the team and the author is collaborating with the project since the start of this work.

On the other hand, the wave direction is typically represented by a misalignment relative to the wind, until recently considered null in most designs of FOWTs. [37]. Most of the research works carried out so far do not consider the wind and waves correlation neither

the misalignment between them but some studies have realized that this misalignment is significant at deep sea and it causes large loads on the tower in the side-side direction, which has very little structural damping compared to the fore-aft direction.

The above reasons lead us to point out the importance of Waves loads and misalignment modelling to make FOWT a reliable wind energy source. Waves models from other authors as well as some exceptions of studies in recent literature that address the issue of misalignment have been found and their results will be reviewed and commented in this section.

2.3.2 Significant waves height models

Significant wave height (WVHT feature in data set) denoted in literature as H_s is the average of the highest one-third (33%) of waves (measured from trough to crest) that occur in a given period known as "wave-peak spectral period" or "dominant wave period", T_s (DPD feature in our data set) [38]. These features together with the average wave period (APD feature) denoted in the bibliography by T_m, are widely used to describe ocean state in coastal, marine engineering as well as in recent studies in offshore energy investigation.

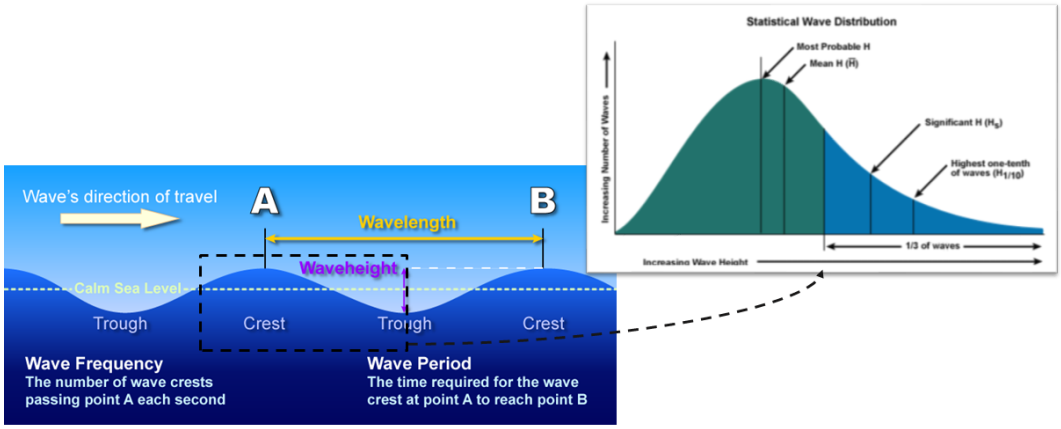


Figure 2.4. *Ocean measurements description (left sub-image) and Significant wave height (H_s) representation in one wave diagram (right sub-image). Images obtained from www.marine_science.com and paper [39] respectively*

Waves models are classified into the same categories as wind models according to the forecast time horizon (very short-term, short-term, medium-term an long-term predictions) and the modelling technique used (physical, statistical, intelligent or hybrid models).

In paper [34], authors present a linear regression and dressing combined model to post-

process the significant wave height predictions made by the numerical weather prediction model used by the European Centre for Medium-Range Weather Forecasting (ECMWF). The wave model predicts the directional wave spectra at each grid point and some input features that it received are historical DPD, APD and WVHT. They conclude that the ensemble model of ECMWF has better skills than persistence model and the main improvement achieved by the post-processing model is to better forecast the uncertainty of wave state. Significant waves height have been addressed also by Fuzzy logic approaches [40] and it was demonstrated that outperformed the forecasting made with statistical ARMAX models. Getting closer to the techniques used in this project, authors from [34] model the WVHT feature with Support Vectors Machine for Regression (SVR) and compare results with ANN, Multi-layer Perceptron (MLP) and Radial Basis Function (RBF) models. Data set used in [34] collect data from September and December of two years gathered from the NDBC center. Data from one year was selected as a training set and the other year data for the testing set. For validating, a 10-fold cross-validation configuration was applied. They conclude that optimal parameters for their objective of SVM are $C = 100$ and $\varepsilon = 0,001$. Parameters selected for RBF kernel function are $\gamma = 0,01$ and $p = 1.0$ for the polynomial kernel function. On the other hand, the best topology found by a trial and error is a three-layer feed-forward network with Sigmoid Transfer function and $7 \times 15 \times 1$ neurons in the input $\times$ hidden $\times$ output layers [34].Errors obtained for testing set are: $RMSE = 0,21$ for SVM with RBF kernel, $RMSE = 0,26$ for SVM with Polynomial kernel, $0,23$ for MLP artificial neural network and $0,25$ for RBF ann. SVM is shown in this study that generalizes better than ANN obtaining better $RMSE$ and they are considered by authors as a more reliable technique for offshore energy application.

2.3.3 Wind-wave misalingment studies

Misalignment between waves and wind (as represented in 2.5) is defined as the time difference between the direction of the wind (in degrees, origin at 0 (N) and clockwise) and the mean direction of the waves [41]. Under ideal conditions (straight shoreline, constant wind and deep waters), the difference is zero by definition. But in high seas, this value is rarely null due to different physical and orographic conditions.

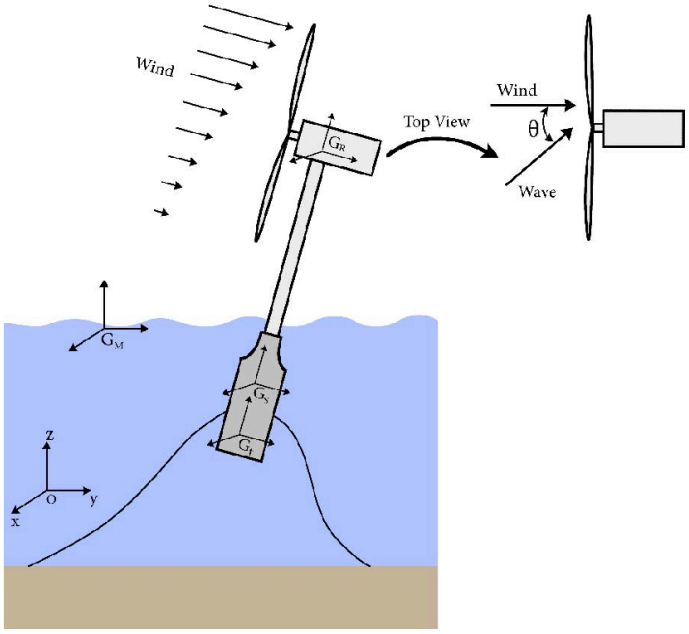


Figure 2.5. *Misalignment angle between wind and waves to which the turbine is attached [42]*

One of the exceptional studies found in the literature that take into account the misalignment is the work by [43]. Despite it is focused on structural control, it highlights the importance of considering wind-wave misalignment when analyzing loads of offshore wind turbines. In [44], the response of turbine structure from an Offshore wind farm to the directional spreading of waves in the inline (thrust) direction and crosswise direction (the perpendicular direction to inline one) is studied. Although structure response to forces isn't clear, they conclude with a clear wind-wave misalignment influence on the turbine observed. In the same line, [45] investigates the responses of a spar, tension leg platform (TLP) and two semi-submersible floating wind turbines in selected wind and waves misalignment conditions. Although some misalignment conditions result in increased motions both parallel and perpendicular to wave direction, aligned wind and waves cause the largest short-term tower base fatigue damage for the studied platforms and conditions. The same author [45] compares the dynamic response of a representative TLP-WT in both aligned and misaligned conditions.

Recently, authors in [46] have carried out statistical analysis for different properties of wind and wave loading such as wind-wave misalignment, significant wave height and wind velocity, in order to predict long-term metocean conditions, on monopile WTs.

2.4 Frequency domain vs. time domain in modelling

In the field of real data modelling, we can address a modelling problem in the time domain or in the frequency domain.

When we have a continuous set of observations ordered in the timeline and separated

equally in spaced intervals, we are talking about time series [47]. The time domain analysis is a widely used analysis approach of nature phenomena whose variation is related to the moment in time we observe them. Physic, statistical and recently intelligent models try to forecast the state of the phenomena in a future moment (t) by learning from phenomena state in past moments in time (t-1, t-2,...). This is called time series forecasting. In this project, we will forecast wind speed and wind-waves misalignment in the time horizon. In the scientific community when we talk about study the waves direction respect to the wind direction (misalignment) we say that we are analysing the waves in the time domain.

On the other hand, the **Spectral or frequency domain analysis** tries to predict or describe some phenomena by learning the frequency with which it is repeated in relation to the state of other variables or phenomena. In this work, we will predict the significant waves height in the frequency domain considering other variables as model input such as wind speed or waves direction.

Chapter 3

Materials and methods

3.1 Used databases

The data used in this project corresponds to standard meteorological and descriptive wave data obtained from the **National Oceanic and Atmospheric Administration (NOAA)** in the USA.

Figure 3.1. *Header from National Data Buoy Center website. Image captured from www.ndbc.noaa.gov.*

Data measurements are taken offshore by floating buoys distributed through the U.S and international waters maintained by **the National Data Buoy Center (NDBC)**. The NDBC is part of the NOAA's National Weather Service and makes available their databases with historical and real-time data on their web page for researching community and any interested user to be downloaded.

3.2 Machine Learning

"Machine Learning is the field of study that gives computers the ability to learn without being explicitly programmed"

Arthur Samuel, 1959

3.2.1 Introduction

Machine Learning or ML is an area of Artificial Intelligence focused on the study of algorithms that allows mathematical models executed on computers to learn from experience. The term Machine Learning appears in 1959 and is due to Arthur Samuel who first applied ML in the development of a game that learned to play checkers in 1952. To this day, this discipline of computer science continues to grow into a powerful collection of algorithms widely applied in data science and modelling projects which allows for the automation of learning.

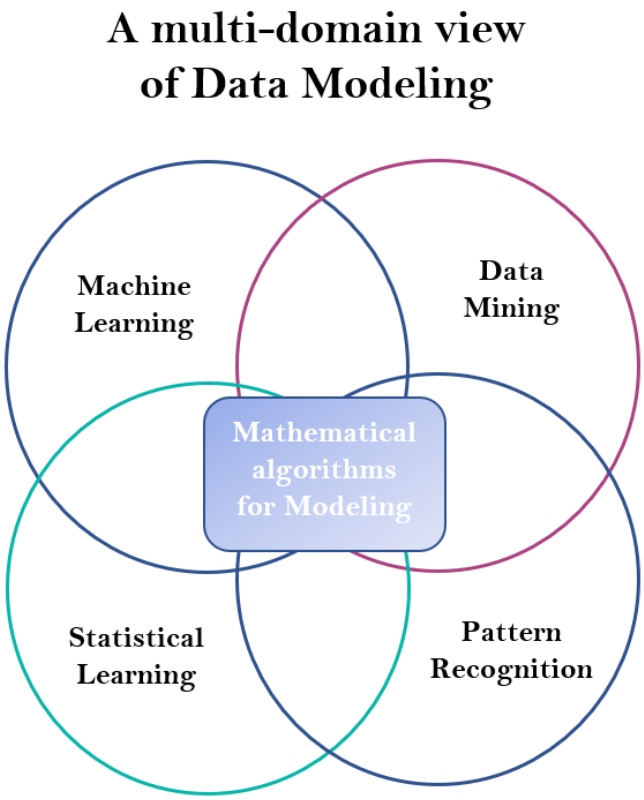


Figure 3.2. *Disciplines involved in a data modelling process*

The idea behind the Machine Learning methods is that the models ”learn” from input data examples formed by considered domain characteristics or features values and they update by themselves the value of parameters to adapt their response to the observed reality.

We can already see the enormous potential of ML, which gives intelligent systems the ability to change the program that runs on them just by analyzing and inferring information from the big data received without being explicitly programmed. This learning process is assimilated into the way human beings learn from experience and data they received from outside. However, when dealing with complex problems from different fields such as medicine, engineering, social sciences or any statistical analysis project, which handles a large amount of data, human computational capacity is not sufficient. Therefore, ML let us create applications that can overcome complex challenges like the piloting

of an autonomous helicopter, handwriting recognition, computer vision, recommending systems developing, etc.

3.2.2 Workflow of a standard ML application

When Machine Learning is applied in a project, a series of standard steps are met regardless of the ML algorithm selected (see figure 3.3).

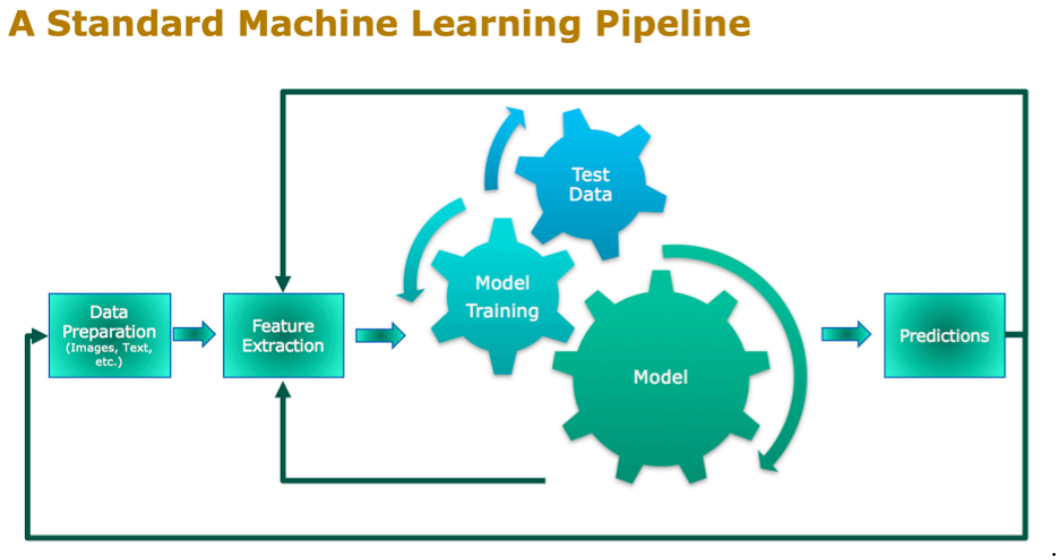


Figure 3.3. *Functional diagram of Machine Learning project workflow. Image retrieved from www.datanami.com*

The starting point that motivates to apply Machine Learning is the approach of a problem or complex task which involves a huge amount of data that needs to be sold. For instance, the problem that exists when we talk about eolic energy is that we need to know the amount of electricity power turbines will be able to produce, in order to allocate demand among power plants to minimize generation costs and ensure that the population's energy demand can be met. The electrical energy produced by a turbine depends on factors such as the wind speed. Consequently, there is a need to predict wind speed in the near future and this task will be addressed by applying ML techniques.

The first step of the project is Data preparation which includes the search and collection of the data and the pre-processing of data gathered. We can find data sets available in databases that we can use as input data for the algorithm. In other cases, data must be generated or built by us for the concrete project we are addressing, which may require some time. The data collected must be examined and pre-processed to adapt it to the format of the model input.

Features extraction is the process before the training phase when we reduce the dimension of raw data collected to more manageable groups for processing. We start from a large data pool with several variables and rows, so it can be necessary to split it into some data

sets and combine some variables to give rise to a single feature of the model's input data set[48]. These two phases can consume the 50-70 % of the ML project time because it is not a trivial activity and requires expert knowledge of the domain and data handling until obtaining useful and clean data sets for model training.

In modelling phase, the ML algorithm is executed and trained. The algorithm receives constantly input data examples and is updating model parameters. After training, the model is tested with new examples that were not included in the training data set. The performance of the model on these examples is measured and until errors calculated aren't considered good enough, it goes back to previous phases. That's why in picture 3.3 all phases are connected with the previous one forming a cycle. While applying techniques to analyze the results and observe if our model suffers from bias or variance, we will decide whether to collect more data or reduce data set size as well as change features to retrain the model and test it again.

After optimizing the data set, tuning model parameters and training the model as many times as needed, we obtain the model to apply to real-time data or to predict future features, covering our initial problem. Models are usually optimized and retrained over time as the reality we know at any given time changes.

3.2.3 ML types

ML problems are classified into 3 types according to the form of the input data set or 'training' data set that the algorithm receives:

- **Supervised Learning**. Applications in which training data comprises examples of input feature values and their corresponding expected output (one or more values) or 'target' vector are known as supervised learning problems. The supervised learning algorithms use the known outputs of training examples to optimize their performance so after the training phase they can predict output for a new input example not present in training data set[49].
  In supervised learning, we distinguish between classification and regression problems. In classification problems, the output can take a number of discrete values, i.e. each input example belongs to one of the few available output classes. In contrast, if the expected output consist of one or more continuous variables, then the problem is called a regression problem.
- **Unsupervised Learning**. In some pattern recognition problems, the training data consist of input examples with features but there is not a target value or expected result. These problems are called unsupervised learning problems and the main objective can be discovering groups of similar examples within the data (clustering), determine the distribution of data within the input space (density estimation) or project data from a high-dimensional space down to low dimension in order to compress data or visualize it [49].
- **Reinforcement learning**. Reinforcement learning algorithms are typically present in games where there is a reward and there are different tactics or actions that can

be chosen in each situation. The objective is to maximize the reward and the algorithm discovers optimal actions and outputs reachable by a process of trial and error instead of receiving them as training examples as in supervised learning.

The aim of the regression analysis commented above is to construct mathematical models that describe or explain relationships that may exist between variables [50]. This project covers the supervised problem of Regression, as we want to construct models to forecast wind and waves features (continuous variables) by inferring relationships between the predicted feature and predictors. However, we will deal with some unsupervised learning problems during the data preparation phase so unsupervised methods will be also explained here below.

3.2.4 Model representation

First of all, we establish the used notation to define the elements of a regression model across the project:

- **Input variables or predictors**. A vector $x^{(i)} = [x_1^{(i)}, x_2^{(i)}, ..., x_n^{(i)}], x^{(i)} \in \mathbb{R}^n$ corresponds to features vector from the i^{th} sample of the data set that algorithm receives as input. $n \in \mathbb{R}$ is the number of variables or columns in the data set. The space of input values is also denoted as X.
- **Output variable or target**. y is the dependent variable of x, i.e. the model output value $y^{(i)}$ is given by the i^{th} input values vector, $x^{(i)}$. The targets are the known outputs of the model for training and testing sets in Supervised Learning problems. The target space is also represented as Y.
- **Model Parameters**. $\theta = [\theta_0, \theta_1, ..., \theta_n]$ is the parameters vector that model optimize when is in training phase. These parameters are considered just on algorithms interpreted in terms of weight-space view, like Linear Regression. Instead, other algorithms make their inferences in the space of functions, the function-space view [51].
- **Model definition or hypothesis function**. $h(\theta)$ is the model or mathematical function that predicts an output $h(\theta)^i$ given an input vector x^i. The form of this function depends on the selected algorithm.
- **Predictive output**. The predictive value for an input example $x^{(i)}$ obtained by the trained model will be denote as $\hat{y}_i$ or $h_\theta(x_{(i)})$.
- **Data set**. $D = \{(x^{(i)}, y^{(i)}) \mid i \in [1, m]\}$ represents a data set consisting of m labeled examples (rows) in total and n features(columns).We add a first column full of 1s that corresponds to θ_0.Thus the data set is a table of size $m \times (n + 1)$.

When we index the number of examples in a data set, we will use superscript i and when we are referring to components of a vector-like x or θ we will use j sub-index. On some

occasions, i could appear as sub-index for convenience and better reading but it continues indicating the number of the example. We will also denote the space of input variables with X and will use Y to denote the space of output values.

On this point, to describe the supervised learning problem slightly more formally, our goal is:

Given a training set, to learn function $h : X \rightarrow Y$ so that $h(x)$ is a good "predictor" for the corresponding value of y.

We can measure the accuracy of our model or hypothesis function by using a **cost function** $J(\theta_0, ..., \theta_n)$ [52] dependent on the vector of parameters learnt θ. Thus, learning algorithms try to minimize the cost function by updating the parameters iteratively applying an algorithm typically Gradient descent.

3.2.5 Data pre-processing

In Machine Learning, we are usually working with multidimensional data sets whose features are of different nature. That means features values range can be so different one to another as well as features values can be not distributed around the same mean point. Not scaled and not normalized data could lead to a bad combination of features in algorithms running. For instance, a feature x_1 whose values are within a range [1,1000] when combined with values from a variable x_2 withing a range of $[0, 1]$ would have much more influence in the modelling than the last. [53].

In this line we address the **z-score method** for data normalization and feature scaling if needed to be applied in the PCA algorithm explained in this chapter too.

3.2.5.1 Z-score

$z-score$ technique is applied to a data set when it needs to be normalized and scaled. This method transforms every feature value as time as maintains its distribution. Given the training set $x^{(1)}, x^{(2)}, ..., x^{(m)}$ we must normalize and scale the data. Firstly, we compute the mean of each jth feature (data set column) μ_j as follows

$$\mu_j = \frac{1}{m} \sum_{i=1}^{m} x_j^{(i)}$$

Normalizing data consists on replacing each $x_j^{(i)}$ by $x_j^{(i)} - \mu_j$ where i indicate the concrete data example and j the number of feature or component of the example.

Then, feature scaling consist on dividing the new $x_j^{(i)}$ by the standard deviation of j feature s_j.Thus the $z - score$ value for each $x_j^{(i)}$ is

$$z_j^{(i)} = \frac{x_j^{(i)} - \mu_j}{s_j}$$

3.2.6 Unsupervised Learning

3.2.6.1 LDOF algorithm

Local Distance-Based Outlier Factor (LDOF) is calculated as part of an outliers detection algorithm. LDOF measures the relative location of a point to its neighbors to determine how much deviates from the neighborhood [54]. Depending on the data set, a threshold is chosen with which to compare the LDOF factor. If the threshold is passed by a point, this will be considered to be an outlier.

LDOF computes distance for observations to its k-nearest neighbors and compares the distance with the average distances between the nearest neighbours.

The k-nn distance of a point x_p , given the set of k-nearest neighbours N_p can be calculated as follows:

$$\overline{d_{xp}} := \frac{1}{k} \sum_{x_i \in N_p} dist(x_i, x_p)$$

On the other hand, we need to know the KNN inner distance of x_p obtained by the following expression:

$$\overline{D_{xp}} := \frac{1}{k(k-1)} \sum_{x_i, x_{i'} \in N_p, i \neq i'} dist(x_i, x_{i'})$$

Finally, the local distance-based outlier factor of x_p is defined as [54]

$$LDOF_k(x_p) := \frac{\overline{d_{xp}}}{\overline{D_{xp}}}$$

3.2.6.2 PCA algorithm

PCA formulation

The motivation for applying Principal Components Analysis (PCA) algorithm on our data sets is to overcome the unsupervised problem of dimension reduction. This can be useful to

- Speed up the learning algorithms
- Reduce our complex data sets to a 2-dimension or 3-dimension set in order to visualize it.

Therefore, given the training set $\{x^{(1)}, x^{(2)}, ..., x^{(m)}\}$ where $x \in \mathbb{R}^n$, PCA algorithm reduce the data from n-dimensions features to k-dimensions. This is achieved by looking for a k-dimensional space where to project data points so the average squared projection error is minimum. This space will be represented by k directional vectors $\{u_1, u_2, ..., u_k\}$ called

```
[U, S, V] = svd(Sigma)
```

Figure 3.4. *Single value decomposition function call in Matlab*

eigenvectors.k is in fact the number of principal components we want to obtain for each training example.

Before applying PCA, we normalize the data and if it is necessary we scale it so we replaced each $x_j^{(i)}$ by its z-score value:

$$x_j^{(i)} = \frac{x_j^{(i)} - \mu_j}{s_j}$$

Then we can compute the PCA, starting by calculating the covariance matrix $\Sigma \in \mathbb{R}^{n \times n}$ as showed in equation (3.1).

$$\Sigma = \frac{1}{m} \sum_{j=1}^{m} (x^{(i)})(x^{(i)})^T \tag{3.1}$$

Then, eigenvectors are calculated by the single value decomposition function of Σ (see Matlab command in 3.4). where $U = \begin{bmatrix} | & | & ... & | \\ u_1 & u_2 & ... & u_n \\ | & | & ... & | \end{bmatrix} \in \mathbb{R}^{n \times n}$. k eigenvectors will be the k first columns of matrix U.

Now, we calculate the projection of the points $x^i \in \mathbb{R}^n$ on the k-dimensional space resulting in new data sets whose examples will be represented by $z^{(i)} \in \mathbb{R}^{(k)}$. Equation (3.2) shows how obtained principal components based points from the k eigenvectors.

$$\begin{aligned} z^{(i)} &= U_{reduce} x^{(i)} \\ &= \begin{bmatrix} | & | & ... & | \\ u_1 & u_2 & ... & u_k \\ | & | & ... & | \end{bmatrix}^T x^{(i)}, \quad \forall i \in [1, m] \end{aligned} \tag{3.2}$$

Reconstruction of original data

Given the original training set $\{(x^{(i)}, y^{(i)}) \mid \forall i \in [1, m]\}$, when we apply PCA algorithm we obtain a new training set $\{(z^{(i)}, y^{(i)} \mid \forall i \in [1, m]\}$ where predictors are composed by k principal components instead of n features as in the original set. Now, we can obtain an approximation of the original $x^{(i)}$ by transforming the corresponding $z^{(i)}$.
Considering $Ureduce$ as a matrix formed by k first columns of U obtained in and remembering the equation to obtain $z^{(i)}$ (3.2), $x_{approx}^{(i)}$ for all $i \in [1, m]$ can be obtained as

$$x_{approx}^{(i)} = U_{reduce} \cdot z^{(i)}$$

Optimal selection of k parameter

Before applying PCA, we must choose the number of principal components k we want to obtain for the new data set. The aim is to select the smallest k that will allow retaining

the greatest percentage of data variation, usually 95% or 99%. We decide to consider 99%.

The experimental process can be try PCA by computing U_{reduce} and $\{z^{(1)}, z^{(2)}, ..., z^{(k)}\}$ with $k = 1$, $k = 2$ and so on until 99% of data variation is retained. This is the same as check if the ratio calculated as the division of the average squared projection error by the total variation of data is less than 1%, as seen in equation (3.3).

$$\frac{\frac{1}{m}\sum_{i=1}^{m} x^{(i)} - x_{approx}^{(i)}{}^{2}}{\frac{1}{m}\sum_{i=1}^{m} x^{(i)2}} \leq 0.01? \tag{3.3}$$

As computing this equation is computational inefficient, there is another way to calculate the ratio explained below.

Given a k value and the diagonal and square matrix $S = \begin{bmatrix} S_{11} & 0 & 0 & ... & 0 \\ 0 & S_{22} & 0 & ... & 0 \\ 0 & 0 & S_{33} & ... & 0 \\ 0 & 0 & 0 & ... & S_{nn} \end{bmatrix}$ as an output of the *svd* calling (3.4), we can compute the ratio as follows

$$1 - \frac{\sum_{i=1}^{k} S_{ii}}{\sum_{i=1}^{n} S_{ii}} \tag{3.4}$$

where S_{ii} are the diagonal components of matrix S. Now we will check if the expression (3.4) is less or equal than 0,01 or 0.05 in the case of 99% or 95% of variance is maintained respectively.

3.2.7 Supervised learning Algorithms

In this subsection we will intend to explain the mathematical theory behind the supervised algorithms applied in the training phase:

1. Linear Regression
2. Gaussian Process Regression
3. Support Vector Machine for Regression.
4. Artificial Neural Networks

3.2.7.1 Linear Regression

Model representation

Linear Regression is the simplest model of regression in terms of implementation and interpretability.The hypothesis function establishes a linear relationship between input

variables and output as follows:

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + ... + \theta_n x_n \tag{3.5}$$

where $\theta_0, \theta_1, ..., \theta_n$ are model parameters and $x_1, ..., x_n$ is the representation for the components of an input example. θ_0 represents the bias or offset, multiplied by an 'additional' input feature added to data set whose value is always one, i.e. $x_0 = 0\ \forall i \in [0, n]$ [51]. The hypothesis function can be vectored for further implementation as follows:

$$h_\theta(x) = \begin{bmatrix}\theta_0 & \theta_1 & ... & \theta_n\end{bmatrix} \begin{bmatrix}x_0 \\ x_1 \\ ... \\ x_n\end{bmatrix} = \theta^T x$$

When x has an unique component and therefore $\theta = \begin{bmatrix}\theta_0 & \theta_1\end{bmatrix}$, we are talking about **univariate linear regression**. If input examples have more than one feature instead, we are in a **multivariate linear regression** problem.

Optimization algorithm

The cost function used to measure the accuracy of the hypothesis function is $\frac{1}{2}$ multiplied by the "Squared error function" or "Mean squared error" (MSE) which is the mean of differences between the predicted value and actual output value for every examples in data set, as seen in equation (3.6).

$$J(\theta_0, \theta_1, ..., \theta_n) = \frac{1}{2m} \sum_{i=1}^{m} (\hat{y}_i - y_i)^2 = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x_i) - y_i)^2 \tag{3.6}$$

In the execution of the algorithm, it is intended to update θ parameters vector for every input training examples $(x^{(i)}, y^{(i)})$ by minimizing the cost function. This is achieved by applying the batch gradient descent method. It consist on getting the new value of each θ_j by subtracting from the current θ_j the partial derivative j^th component of the cost function $J(\theta_0, ..., \theta_n)$ as seen in algorithm equation (3.7).The updating process is repeated iteratively until the cost function J converges to a global minimum, i.e., when the decreases of $J(\theta_0, ..., \theta_1)$ is less than a constant ε established by us.

repeat until convergence:{

$$\theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta) = \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)} \ \ \forall j \in [0, n] \tag{3.7}$$

}
given a learning rate α and m training examples.

Intuitively, we must consider that if we choose a too small α, the gradient descent computation could be so slow but if α is too large, the gradient descent could overshoot the minimum, making impossible to converge or even generating the diverging of the algorithm.

3.2.7.2 Gaussian Process Regression

Gaussian Process Regression (GPR) is a non-parametric, Bayesian approach to regression [51].

This algorithm instead of learning concrete values for parameters of a defined model infers the probability distribution over all admissible functions that fit the data (function-space view).

Bayesian linear model

To understand the Bayesian approach working on Gaussian Process, we first address the case of linear regression:

$$y = \theta^T \times x. \tag{3.8}$$

remembering θ is the parameters vector and x is the input features vector.

Known the training data set, $\mathcal{D} = \{X, y\}$ of n observations containing predictors of every example in X and targets in y, the Bayesian treatment of linear regression calculate a prior distribution $p(\theta)$ on the parameter θ and recalculate the distribution (posterior distribution) based on observed data set $\mathcal{D}$, computed by Bayes' Rule:

$$posterior = \frac{\text{likelihood} \times \text{prior}}{\text{marginal likelihood}} \tag{3.9a}$$

$$p(\theta \mid y, X) = \frac{p(y \mid X, \theta) \cdot p(\theta)}{p(y \mid X)} \tag{3.9b}$$

The marginal likelihood of the denominator in explaining equation (3.9a) or $p(y \mid X)$ in equation (3.9b) is the normalizing constant given by:

$$p(y \mid X) = \int p(y \mid X, \theta) \cdot p(\theta)\, d\theta$$

The likelihood $p(y, \theta)$ corresponds to the multiplication of the likelihood for each case y_i. Assuming that $h(\theta)$ and y just differ in the noise factor θ_0 and θ_0 follows a Gaussian Distribution, $\theta_0 \sim \mathcal{N}(0, \sigma_n^2)$, the likelihood is calculated as follows:

$$\begin{aligned} p(y, \theta) &= \prod_{i=1}^{n} p(y_i \mid x_i, \theta) \\ &= \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}\sigma_n} \exp\left(-\frac{(y_i - \theta^T x_i)^2}{2\sigma_n^2}\right) \\ &= \frac{1}{(2\pi\sigma_n^2)^{\frac{n}{2}}} \exp\left(-\frac{1}{2\sigma_n^2}|y - \theta^T X|^2\right) \\ &= \mathcal{N}(\theta^T X, \sigma_n^2 I) \end{aligned} \tag{3.10}$$

On the other hand, the prior term included in the equation (3.9), $p(\theta)$ must be assigned before any training according to the previous knowledge we estimate on data. It will be

considered that θ follows a Gaussian distribution with a zero mean and a variance Σ_p which represent the co-variance matrix of weights [51].

$$\theta \sim \mathcal{N}(0, \Sigma_p)$$

Developing the above expression, we establish that $p(\theta, y) \sim \mathcal{N}(\overline{\theta} = \frac{1}{\sigma_n^2} A^{-1} X y, A^{-1})$ where $\overline{\theta}$ is the mean of θ components and A^{-1} is the co-variance matrix from the posterior distribution, given A by the expression:

$$A = \sigma_n^{-2} X X^T + \Sigma_p^{-1}$$

At this point, we can calculate the predictive distribution f_* of a set of unseen points or test set x_* as follows

$$\begin{aligned} p(f_* \mid x_*, X, y) &= \int p(f_* \mid x_*, \theta) p(\theta, y)\, d\theta \\ &= \mathcal{N}(\frac{1}{\sigma_n^2} x_*^T A^{-1} X y, x_*^T A^{-1} x_*) \end{aligned}$$

Gaussian Process Regression

The difference between GPR and Bayesian treatment made to linear regression is that GPR isn't parametric while LR is, so instead of calculating the parameters linear function probability distribution, GPR calculates the probability distribution of all functions that could fit data and finally calculate the posterior distribution of test set points.

The Gaussian process (GP) can be defined as a multi-variate collection which follows a Gaussian distribution as well as any subset of variables from this collection have a joint Gaussian distribution. That means that if a data set $(x_1, x_2) \sim \mathcal{GP}(\mu, \Sigma)$ then we confirm that $x_1 \sim \mathcal{N}(\mu_1, \Sigma_{11})$ were Σ_{11} is a sub-matrix of Σ or what it is the same the latent variable $f(x_1)$ is Gaussian.

Gaussian process comes specified by a mean function $m(x)$ and the Kernel co-variance function $k(x, x')$ of a real process $f(x) \sim \mathcal{GP}(m(x), k(x, x'))$. Our previous knowledge about the space of functions is reflected in the selection of the mean and the co-variance matrix of the prior $f(x)$ as well as the possibility of incorporating noise with the same distribution:

$$y \sim \mathcal{GP}(m(x), k(x, x') + \delta_{ij} \sigma_n^2)$$

GPR hyper parameters tuning

In the model selection, we can specify the form of the mean function and the co-variance kernel function which will be tuned during the optimization phase. Among the kernel functions are constant, linear square exponential, Matern kernel or a combination of some of them [55]. For instance, it is widely used the combination of the constant kernel with the Radial Basis Function (RBF) kernel:

$$k(x, x') = \sigma_f^2 \exp\left(-\frac{1}{2l^2} x - x'^2\right)$$

where σ_f^2 and l are the hyperparameters must be tuned.

3.2.7.3 Support Vector Machine for Regression

Support vector Machine also called "large margin classifier" is a Machine Learning classification technique that tries to find the line or hyper plane that best separate observations of the training set into different output classes, by maximizing the distance of observations to decision boundaries called "support vectors" (see figure 3.5).

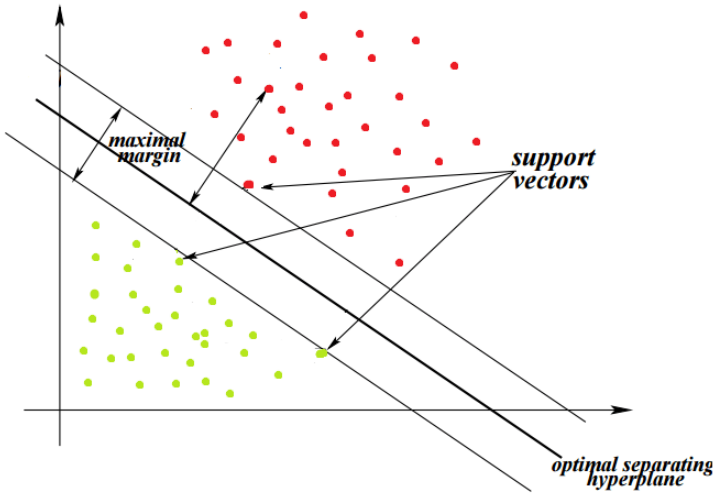


Figure 3.5. *Support vector machines for classification representation. Image downloaded from www.coursera.com*

Depending on the data, linear or nonlinear kernel functions can be chosen to obtain optimal support vectors. As this algorithm proved to be very promising in classification problems, concepts of SVM have been generalized to regression problems, resulting in Support Vector Machines for Regression or **Support Vector Regression** algorithm.

We can see the regression problem as a classification problem with infinite output classes (continuous output variable). Thus SVR introduces a ϵ-sensitive region around the function y we want to predict, called the ϵ-tube. SVR optimization problem consists of find the tube that best approximates the continuous-valued function [56].

Model representation

We are trying to find a function (linear or not)

$$f(x) = \theta^T x + b$$

considering the goal of

$$\min \frac{1}{2}\|\theta\|^2$$

and constraints [57]:

$$\|y_i - \theta_i x_i\| \leq \epsilon$$

3.2.7.4 Artificial Neural Networks

Artificial Neural Networks (ANNs) or just Neural Networks (NNs) are computing systems inspired by the biological neural networks that constitute animal brains [58].

Neural networks locate its origin in neuro-rewiring experiments which proved that concrete parts of the brain tissue able to interpret a concrete type of input signals (sound, touch,etc.) could learn to interpret other type of signals by just changing the input connections (see figure 3.6). Thus leads us to conceive of a neural network as a brain algorithm that learns by itself to interpret input signals no matter what type they are.

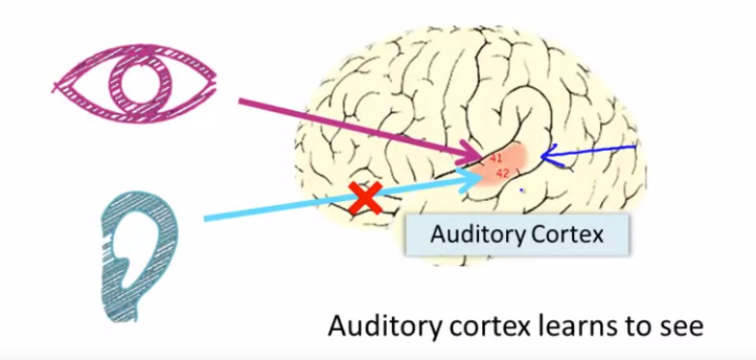


Figure 3.6. *Example of a neuro-rewiring experiment about the Auditory cortex of brain learning. Image obtained from www.coursera/Machine-learning*

Model representation

Artificial NN is a computational solution to modern-day machine learning applications. Simulating brain neurons that receive input information by dendrites, process it and give an output by axons connected to other neurons; an intelligent neuron model can be seen in figure 3.7. An artificial neuron receive input examples with n features by neuron inputs wires and return an output $h_\theta(x) = f(x)$ taking parameters of the models or "weights" assigned to each input wire and an activation function f assigned to neuron also known as Transfer function. x_0 corresponds to the bias unit.

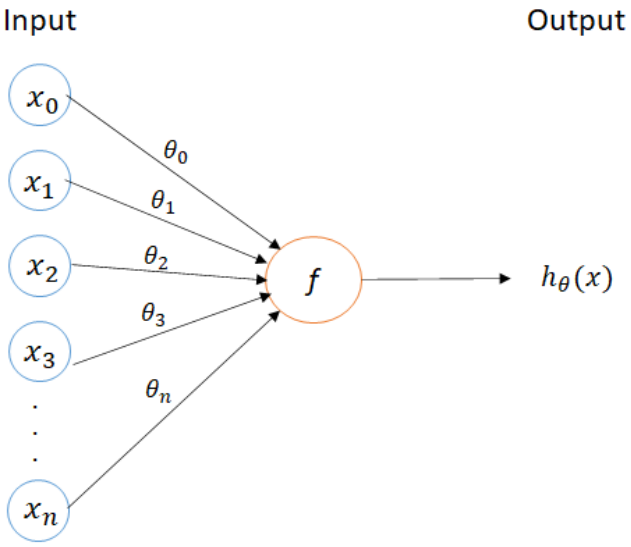


Figure 3.7. *Artificial neuron model*

In the neuron model, $h_\theta(x) = f(\theta_0 + \theta_1 \cdot x_1 + \theta_2 \cdot x_2 + ... + \theta_n \cdot x_n)$ where f is the Transfer function assigned to the neuron. Some f functions from which we can choose are:

- **Linear transfer function** (also called **Purelin**) : $f(h_\theta(x)) = h_\theta(x)$. This function is the linearly one usually used in the output layer of the neural network.
- **Log-sigmoid transfer function** (also called **Logsig**): $f(h_\theta(x)) = \frac{1}{1+\exp -h_\theta(x)}$. This function returns values between 0 and 1 and it useful in classification problems where 1 means belonging to a class and 0 the opposite.
- **Hyperbolic tangent sigmoid transfer function** (also called **Tansig**): $f(h_\theta(x)) = \frac{2}{1+\exp -2 \cdot h_\theta(x)} - 1$. This function is commonly configured in neurons from hidden layers and it returns values between -1 and 1.

A neural network is thus a group of neurons structured by layers where the outputs of the neurons from one layer are the inputs of the neurons in the next layer (see diagram 3.9).

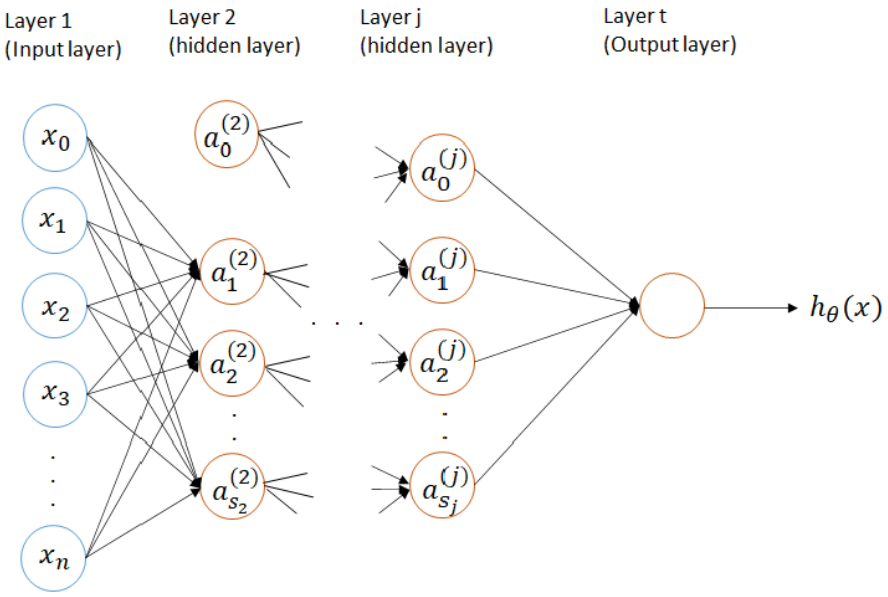


Figure 3.8. *Artificial neural network model*

Notation used in neural network (see figure 3.9) is:

- L = the total number of network layers.
- s_j = number of neurons of layer j. The input layer contains as many inputs as features that form the training examples.
- $a_i^{(j)}$ = activation of neuron i in layer j. $a_0(j)$ is the bias unit (neuron) of this layer.
- $\theta^{(j)}$ = matrix of weights of connecting wires from layer j to layer $j+1$. If layer j contains s_j neurons and layer $j+1$ contains s_{j+1} neurons (without including the bias unit in both), then the size of matrix θ_j will be $s_{j+1} \times (s_j + 1)$. Thus $\theta_{ir}^{(j)}$ corresponds to the weight of the wire that connect neuron x_i from layer j with neuron x_r of layer $j+1$.

Mathematically, we calculate $a_i^{(j)}$ by the following expression :

$$a_i^{(j)} = f(\theta_{i,0}^{(j-1)} x_0 + \theta_{i,1}^{(j-1)} x_1 + ... + \theta_{i,s_{j+1}}^{(j-1)} x_{s_{j+1}}) = f(z_i^{(j)})$$

we represent the hypothesis $h_\theta(x)$ as follows:

$$h_\theta(x) = a^{j+1} = f(z^{j+1})$$

where in this case j = last hidden layer and j + 1 = output layer.

Optimization algorithm

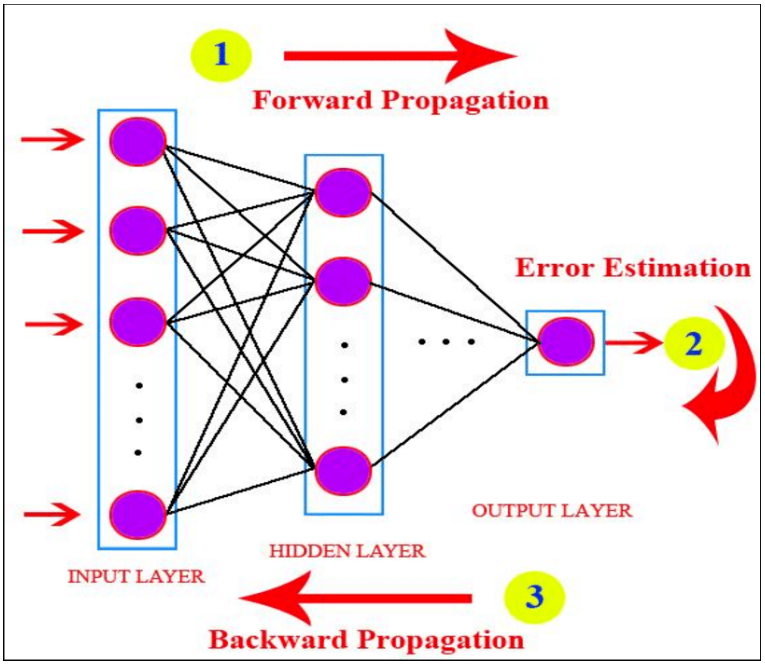


Figure 3.9. *Forward and Backward propagation learning algorithm scheme. Image downloaded from www.researchgate.net*

Backpropagation algorithm is a neural-network terminology used to refer to minimizing the cost function $J(\theta)$ just like we do with gradient descent in other ML techniques such as linear regression [59].

Given the cost function $J(\theta)$,for instance Mean Squared error function, our goal is minimize the cost J giving a optimal selection of parameters θ:

$$\min_{\theta} J(\theta)$$

Considering the notation $\Delta_{ij}^{(l)}$ to denote the partial derivative of $J(\theta)$ for layer l and $\theta_{i,j}$:

$$\frac{\partial}{\partial \theta_{i,j}^{l}} J(\theta)$$

The backpropagation algorithm can been stated in the following steps:

- Given the training set $\{\{x^{(1)}, y^{(1)}\}, \{x^{(2)}, y^{(2)}\}, ..., \{x^{(m)}, y^{(m)}\}\}$, we set firstly $\Delta_{i,j}^{l} := 0$ for all i, j, l.
- For i = l to m:
  - Set $a^{(1)} = x^{(i)}$
  - Perform forward propagation to compute $a^{(l)}$ for $l = 2, 3, ..., L$.
  - Using $y^{(i)}$, compute $\delta^{(l)} = a^{(l)} - y^{(l)}$
  - Compute $\delta^{(l-1)}, \delta^{(l-2)}, ..., \delta^{(2)}$

- We set $\Delta_{i,j}^{(l)} := \Delta_{i,j}^{(l)} + a_j^{(l)} \delta_i^{(l+1)}$
- Hence we update our new Δ matrix we obtain the partial derivative of $J(\theta)$ (to update θ parameters as in equation (3.7) in Linear Regression section):

$$\frac{\partial}{\partial \theta_{i,j}^{l}} J(\theta) = D_{i,j}^{(l)}$$

where

$$D_{i,j}^{(l)} := \frac{1}{m}(\Delta_{i,j}^{(l)} + \lambda \theta i, j^{(l)})$$

if $j \neq 0$ or

$$D_{i,j}^{(l)} := \frac{1}{m}\Delta_{i,j}^{(l)}$$

if $j = 0$

Some variants of the backpropagation optimization algorithm that have been used in this project are:

- **Stochastic Gradient Descent**. SGD consists of updating parameters θ in the negative direction of the gradient g (for a minimization objective) by taking a subset or "mini-batch" of data size (m) [60].
- **Lavenberg-Marquardt**. The Lavenberg-Marquardt algorithm also knows as the damped least-squares method works with a loss function (the sum of squared errors) and its main advantage is the speed up compared to SGD [61].

Finally, we highlight there is no rule to select initial parameters of neural networks such as the number of hidden layers and neurons. The unique way to do that is by trial and error to get an intuition about what will work better for our concrete problem solving applying NN, which can be time-consuming [62].

NAR and NARX neural networks

Nonlinear autoregressive neural network (NAR) a Nonlinear autoregressive with exogenous neural networks (NARX) are models that can be trained to forecast any feature in the time domain.

Both NAR and NARX relates the current value of the time series to the past value of the time series. Moreover, the NARX network relates the current value of time series with current or past external series (other features time series that influence the time series we want to forecast)[63]. Thus, NAR model which predict $y(t)$ series can be represented by the following expression:

$$y(t) = f(y(t-1), y(t-2), ..., y(t-d)$$

The number of input parameters is established by choosing a delay d which means that d past values of y are considered together as predictors to forecasting y(t).

On the other hand, NARX model can be represented with the function:

$$y(t) = f(x(t-1), ..., x(t-d), y(t-1), ..., y(t-d))$$

3.2.8 Evaluation

3.2.8.1 Accuracy Metrics

Three metrics have been computed to evaluate the performance of forecasting models:

- **Mean Square Error** (MSE). It measures the average of the straight line distance of the output estimated by the model and the real output for every samples in the data set. MSE will be used as the cost function to minimize in the training phase. It is obtained with the following expression:

$$MSE = \frac{1}{m}\sum_{i=1}^{m}(h(\theta)^i - y^i)^2$$

- **Root Mean Square Error ($RMSE$).** The $RMSE$ error is the square root of MSE. It allows us to compare trained models by comparing $RMSE$ for testing sets. Thus $RMSE$ is calculated as follows:

$$RMSE = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(h(\theta)^i - y^i)^2}$$

  The smaller the $RMSE$, the closer the predicted and observed values are.

- **Mean absolute error** (MAE). The Mean absolute error performance function is as its name indicates an average of absolute errors that we try to minimize in ML problems. It is used as an additional performance measure for its simplicity. It is calculated as:

$$MAE = \frac{1}{m}\sum_{i=1}^{m}(h(\theta)^i - y^i)$$

- **R-Squared (R^2)**. The R-Squared or coefficient of determination measure what extent the variance of the dependent output variable is explained by the variance of the independent input variable in a regression model. R-squared values range from 0 to 1. An R-squared of 1 means that variation in a dependent variable is completely explained by independent variables. For instance, this metric is useful to distinguish a case where the model is very accurate because most output values in the training set are the same. In this case, a model consisting of a constant will be even more accurate on training examples than a complex model, but it does not represent the whole population and it will possibly have a very high $RMSE$ on new data examples but a small R. The calculation of this metric for a trained model is provided by the Regression Learner tool in Matlab.

- **Success Rate (%)**. The success rate is the percentage of accurate predictions made by the model. Given the maximum value taken by the feature we want to predict as max, the success rate of a model for this variable prediction is calculated as follows

$$\% = (1 - \frac{RMSE}{max}) \cdot 100$$

3.2.8.2 Train/Validation/Test sets

The main technique used to validate our models is the partition of the original data set into three subsets (Holdout validation scheme):

- **Training set**. The data set used in the training phase of the model
- **Validation set**. This data set is used to apply trained models on new non-observed examples and assesses the performance of the model. It also allows selecting the model which best fits to the validation set, i.e. for outputs observed in validation set we choose the model which makes the least $RMSE$ in predictions.
- **Testing set** This set of non-observed data examples by the model is used to evaluate the performance of the selected model and decide whether it needs to be optimized and which techniques should be applied next or decide if choose another model.

The percentages we have applied to divide the large data sets used for training is 60%, 20% and 20% (see figure 3.10).

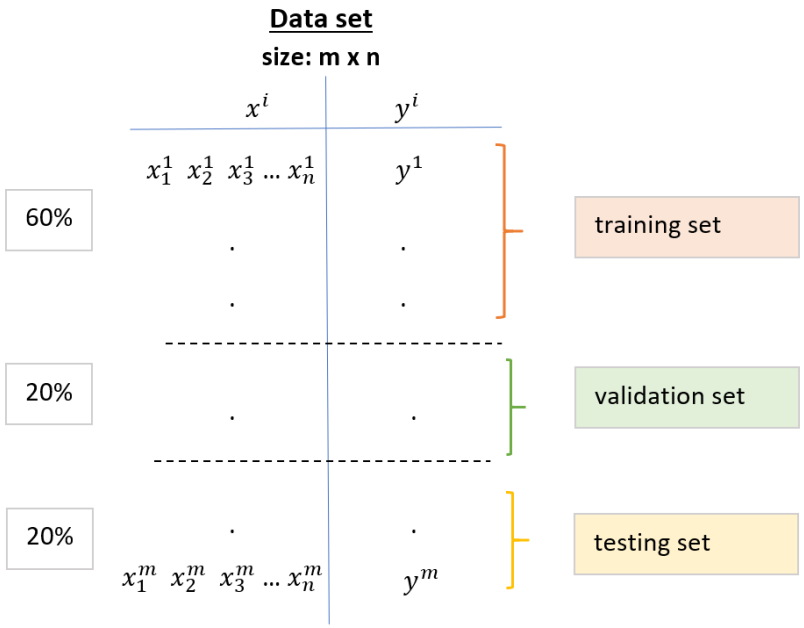


Figure 3.10. *Diagram of division of original data set into training, validation and testing sets using percentages 60%, 20% and 20% of data rows respectively to form each one.*

3.2.8.3 *K*-fold cross validation method

The k-fold cross validation or CV is one of the most commonly applied technique to evaluate machine learning model accuracy on unseen data. This method takes advantage of the complete data set by training the model several times with different training set and tests its performance on a different subset of data not used for training in each iteration.

The general procedure of the method [64] is as follows:

1. Shuffle the data set randomly.
2. Choose the parameter k which is the number of equal-sized folds or subset of data examples in which we want to divide the original data set. (see figure 3.11) .
3. For each fold:
   a. Train the model which out-of-fold data examples.
   b. Assesses model performance using in-fold data by calculating the accuracy measure or error.
4. Calculate the average test error overall folds errors. If we have calculated for each fold i , an error measure E_i, the performance of the model corresponds to

$$E = \frac{1}{k}\sum_{i=1}^{k} E_i$$

Figure 3.11. *Diagram of k-fold cross-validation algorithm represented for k = 10. Image obtained from http://karlrosaen.com/ml/learning-log/2016-06-20/*

CV has been applied to training in data set of a unique year in order to compare the accuracy of different models on these data. After selecting the best one who fits the

data, we retrained the model with the entire training set applying the Holdout validation scheme and tested its accuracy on the testing set.

3.2.8.4 Diagnosing Bias or Variance

After the training phase, if we obtain a model with an unacceptable misclassification ratio, it is so important to figure out if it suffering from high Bias or high Variance.

For this purpose, we study the tendency of growth or decrease of the training error and the validation error, depending on the hyper-parameters selected for our algorithm. We can be in two situations(3.12):

- If $J_{train}(\theta)$ is high and $J_{cv}(\theta)$ is high too, that means our **model suffers from high bias (underfitting problem)**.
- If $J_{train}(\theta)$ is low but $J_{cv}(\theta)$ is far greater, it indicates our **model is suffering from high variance (overfitting problem)**. In overfitting problems the error for training examples is pretty good but the model can not generalize well for new data examples.

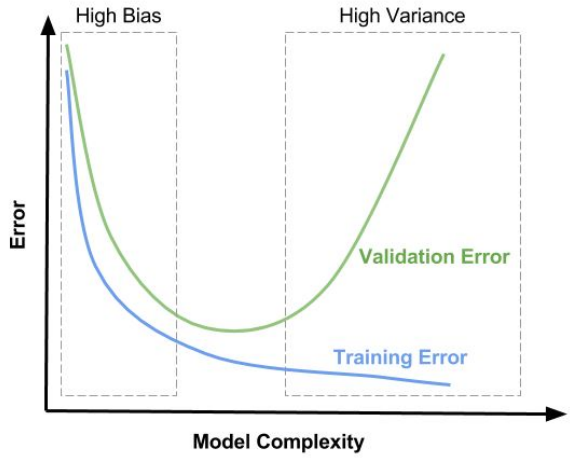


Figure 3.12. *Representation of Bias and Variance areas dependent of the model complexity. Image downloaded from www.model-tunin-with-validaton.*

To prevent overfitting, one of the techniques used is Regularization.

3.2.8.5 Learning Curves

Learning curves are useful plots to evaluate the learning performance of our model over experience [65] how well is our algorithm working and check if we need to improve its

performance. In these plots we represent the errors J_{train} and J_{cv} (or J_{test}) as a function of the number of training examples m used for algorithm training.

Trying to train the algorithm with higher m values each time and plotting J_{train} and J_{cv} for each m give us a sight of the evolution of errors and how far or close are both of them.

If our hypothesis has high bias, the cross-validation error J_{cv} and the test error J_{test} tends to decrease for higher training set sizes m because more data leads to a better generalization for new examples. However the training error J_{train} will increase until being so close to J_{cv} value (see figure 3.13). Intuitively, if our model is so simple to fit the data (underfit), more data implies more points to fit by model function so training error will increase.

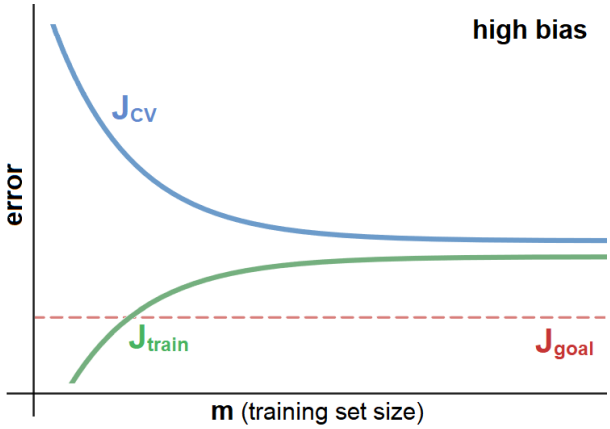


Figure 3.13. *General learning curves obtained for an underfit model. Image downloaded from www.modelDesin.es*

As seen on the graphic, when a model suffers from high bias getting more training data examples will not help much.

In the case of a model with high variance, the model is almost perfectly fit to training data set so the training error is low. If we get more training data, the J_{train} may increase a bit but it will continue being a good error. By contrast, the J_{cv} for little m is pretty higher than J_{train} and for higher m values it will tend to decrease getting close to J_{train} as seen in figure 3.14.

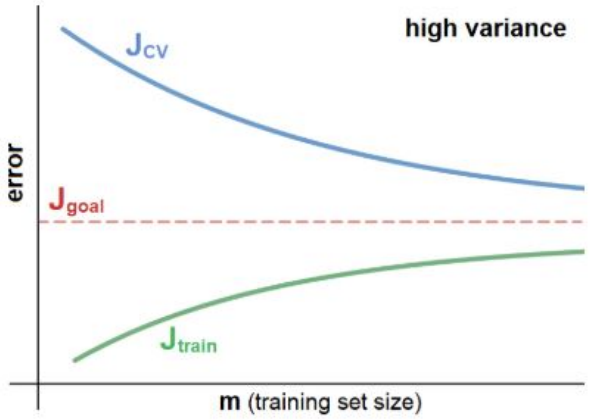


Figure 3.14. *General learning curves obtained for an overfit model. Image downloaded from www.modelDesin.es*

In the last case, getting more training examples is likely to help to achieve the J_{goal} or what is considered by us a good and acceptable error.

3.2.9 Optimization methods

3.2.9.1 Regularization

Regularization is an optimization technique that consist of adding to the cost function a regularization term (see equation (3.11)) that alter the values of the learned parameters θ.

$$\frac{\lambda}{2m}\sum_{j=2}^{n}\theta_j^2 \tag{3.11}$$

where λ is the regularization parameter. Choosing a value for λ is the critical point when applying regularization. Given the cost equation now by the following expression:

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h(\theta)^i - y^i)^2 + \frac{\lambda}{2m}\sum_{j=2}^{n}\theta_j^2$$

If λ is too little the penalization onto high will be insignificant so the parameters θ can be greater and model may suffer from high variance (overfit). In the opposite site, if we take a large λ, θ parameters will be smallest in order to let the model to minimize the cost function so model may suffer from high bias (underfit). Therefore, a good practice is trying different values of λ and select one intermediate λ which allows minimize both the $J_{train}(\theta)$ and $J_{cv}(\theta)$, as seen in figure 3.15.

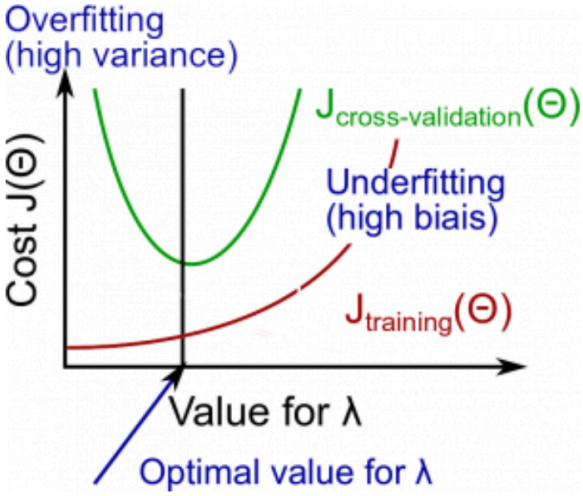


Figure 3.15. *Model errors dependent on the regularization parameter value chosen. Image downloaded from www.regularization.es*

3.2.9.2 Other decisions for optimization

Generally, low order polynomials as hypothesis function or the simplest models have high bias (or low variance) while high-order polynomials and complex models have high variance [66]. Table 3.1 shoes certain decisions that may help to optimize our model after evaluating if it suffers from high bias or high variance as well as not to waste time on useless tests.

To fix high bias	To fix high variance
Try getting additional features	Get more training examples
Try adding polynomial features	Try select smaller set of features
Try decreasing regularization parameter	Try increasing regularization parameter
*For ANN: selecting more complex architecture (more neurons and hidden layers)	*Try selecting a simplest neural network architectura

Table 3.1. *Optimization decisions to fix underfitted (first column) or overfitted (second column) models*

3.3 Matlab

Figure 3.16. *Matlab's logo.*
Image downloaded from www.matlab_logo.es

In this project, MATLAB has been chosen as the language and development environment for data preprocessing implementation techniques and Machine Learning application. MATLAB is specially recommended for visualizing data as it has a wide variety of plots and its computationally efficient for making math vectored operations with huge amounts of data. It offers us useful toolboxes too in addition to all necessary courses and documentation by Mathworks platform.

Part I

Data Preparation

Chapter 4

Data Collection

To apply Machine Learning algorithms in model training, it is crucial to gather sufficient and effective data first.

After a previous data sources search, NOAA database has been considered the most advantageous for this project as it meets the following requirements to ensure the quality and accuracy of our models:

- Data sets must be in the **public domain** or accessible for scientific research purposes. Most of the data sets that collect useful wind and wave characteristics founded belong to monetized private investigations and therefore we couldn't have access to them. By contrast, NOAA run by American Government is put at the service of the scientific community and all users have free access.
- **Metocean data** must be **measured offshore** because our models goal is forecasting wind and wave behaviour to which FWTs are exposed. FWTs are installed about 20-40 km off the coast as well as NDBC buoys installed offshore (see figure 4.1).
- The data set must contain **historical data from previous years** (at least 5 years) to ensure a statistical significance and a effective training. This will let us combining it in various time-series data and trying train models which distinguish data behaviour patterns from point anomalies.
- We are looking for **Wind and Waves features obtained in the same geographic location**. Building a model to forecast misalignment between wind and waves will generate more realistic and representative results if all input features have been measured at the same station.
- It is a beneficial factor to have **real-time data** for the evaluation of the trained models in order to predict recent results. Of the more than 300 National Data Buoys Stations, we have selected one that has available real-time data.

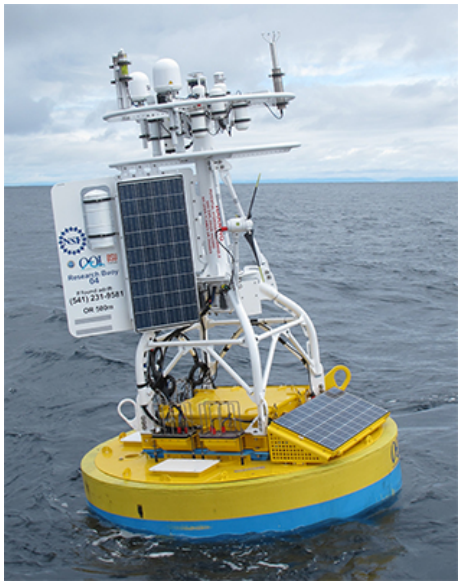


Figure 4.1. *NDBC floating buoy from station 46098, Waldport (Oregon)*

4.1 Santa Maria station, California

The selected NDBC station with moored buoy offshore for data gathering has been Santa Maria station, located in Pacific Ocean, in the Northwest of California (*see figure 4.2*). We have downloaded the meteorological files from last 10 years as well as Real-Time files from January to April 2020. Santa Maria data has been considered by experts in the

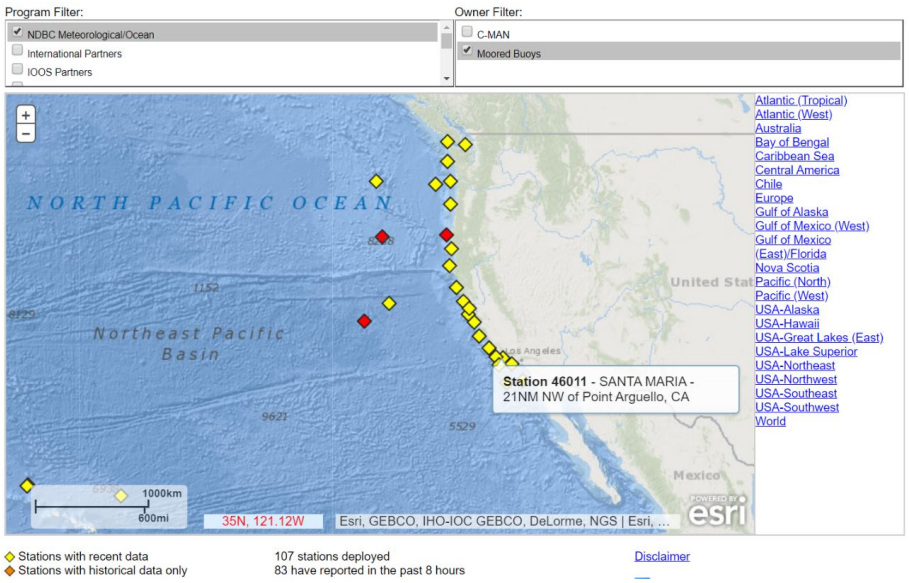


Figure 4.2. *Santa Maria Station marked on map, saw in NOAA website station finder*

creation of a metocean data set for offshore project [67] too, which give us a guarantee

that we have chosen a station of interest for offshore applications. One of the decisive reasons for choosing this station has been the moderate state and stability of the weather and sea state, without the presence of huge disturbances or abrupt weather phenomena that could position this location as unsuitable for the installation of a offshore wind farm and could affect the wind and waves models development in the project. However, the main reason for select this station has been the availability of historical and real-time measurements of the wind and waves essential variables [67]:

- Wind speed
- Significant wave height
- wave peak-spectral period
- wind direction
- and wave direction

The station measurement characteristics are summarized in *table 4.1.*

Station Id	46011
Buoy model	3-meter foam buoy w/ seal cage
Geo-location coordinates	34.956 N 121.019 W (34°57'22" N 121°1'7" W)
Site elevation	Sea level
Air temp height	3.7 m above site elevation
Anemometer height	4.1 m above site elevation
Barometer elevation	2.7 m above mean sea level
Sea temp depth	1.5 below water line
Water depth	464.8 m
Watch circle radius	811 yards
historical data	From 1980 to February,2020
Real - time data	Last 45 days

Table 4.1. *Characteristics of Santa Maria Station and measurements conditions*

4.2 Features description

Standard Meteorological Data files contains one row for hourly atmospheric, wind and waves variables averaged values actually measured generally for 8 minutes period. His-

torical data files are classified by year while real-data files contain last 45 days measures. The same meteorological features and units are available in both type of files. The most significant differences between them are the missing values' representation; numbers of 9's in historical files and 'MM' characters in real-time and the reports of measures in Real-Time files, given each 10 min instead of each hour.

```
#YY  MM DD hh mm WDIR WSPD GST  WVHT   DPD   APD MWD   PRES  ATMP  WTMP  DEWP  VIS  TIDE
#yr  mo dy hr mn degT m/s  m/s     m   sec   sec degT   hPa  degC  degC  degC   mi    ft
2018 01 01 00 00 190  2.4  2.7 99.00 99.00 99.00 999 1019.9  19.6  21.3  14.3 99.0 99.00
```

Figure 4.3. *Header of a data file which shows features and units, downloaded from webpage*

The following table describes variables considered that could be useful for this project:

Variable Name	**Characteristic Name**	**Units**	**Description**
YY	Year	yr	Year when the row measurements were taken
MM	Month	mo	Month when the row measurements were taken
DD	Day	dy	Day when the row measurements were taken
hh	Hour	hr	Hour when the row measurements were taken
mm	minute	mn	First Minute of the 10 minutes while measurements are being taken.This value is always 0 in NDBC historical files and 50 in real-time file we've accessed.
WDIR	Wind direction	degT	Wind direction measured in degrees clockwise from North
WSPD	Wind speed	m/s	Wind speed measured in the same 8 minutes period than Wind direction.
GST	Gust speed	m/s	Maximum peak 5 or 8 second gust speed during 8th minutes interval.

WVHT	Significant wave height	m	Significant wave height in meters calculated as the average of the three highest waves observed during a 20 minutes period.
DPD	Dominant wave period	s	The period of seconds when the maximum wave energy takes place.
APD	Average wave period	s	The average wave period calculated considering all waves during the 20-minute period.
MWD	Waves direction in DPD	degT	The direction from which waves are coming during the Dominant wave period.
PRES	Sea level pressure	hpa	Sea Level pressure is directly measured offshore by sensors installed in buoys.
ATMP	Air temperature	degC	The air temperature in Celsius degrees is taken from 3 metres to 12 metres above the sea level, depending on the height of the buoy model.
WTMP	Sea surface temperature	degC	This variable is measured by buoys at the depth referenced to the hull's waterline while fixed platforms sea surface temperature measures are commonly referenced to Mean Lower Low Water (MLLW)
DEWPT	Dewpoint temperature	degC	This is the temperature at which the air is saturated with water vapour. It is taken at the same height as ATMP.
TIDE	Water Level	ft	the water level above or below Mean Lower Low Water (MLW) in feet.

Table 4.2. *Metocean variables accessed together with the actual characteristic they represent, the units of measurements and a short description [68]*

Chapter 5

Exploratory Data Analysis and data cleaning

"EDA is more an art or even a bag of tricks, than a science"
Irving John Good, 1983

In this initial stage of the project, it is intended to examine and gain insight into the raw data collected. The statistician John W.Turkey first defined this data analysis philosophy in 1977, coined **Exploratory Data Analysis** or **EDA**.

EDA encompasses the visualization of the row data, the calculation of metrics that summarize the distribution of data such as the median or standard deviation, and data re-expression techniques including scaling, cleaning missing values or normalizing variables values [69].

Scaling and normalizing data as well as cleaning missing values are steps of what is called **Data Cleaning**. Although Data Cleaning and EDA can be considered as differentiated processes in the data science field [70], it is a very good practice applying techniques from both indistinctly during **the Data Preparation phase** for analysis and further use. In consequence, it makes a lot of sense to us starting with EDA by making a visual inspection of data loaded and after knowing the data model we have available, we will handling missing data and anomalies; explained in the Data Cleaning section, to continue then applying other interesting EDA techniques on the clean data set.

The interest in applying EDA is the flexibility to explore the data without having a hypothesis or prior knowledge of the domain. After analyzing data distribution and correlations between variables, we can be motivated to make choices like increase the dimensionality of the data set, select a subset of features or include new variables to achieve a more accurate model training. In any case, changes made to the data set must be no intrusive either change the data real distribution [71]. They just modify the way to represent reality enabling us to better understand it.

5.1 Data loading

Data files are downloaded from the Santa Maria Station web page [72] and saved into the current directory, to be then read and stored in a Matlab Table type.

In the function implemented for data loading (see function 5.1), we distinguish between historical or real time files in a *if-else* instruction because the URL constructed within the function to download a historical data file is different from a real time file one. Furthermore, the units of variables are assigned and the first variable name is modified because by default is read with the comment symbol concatenated at the beginning.

```
function [data] = loading(fileName, historical, year)
%LOADING data from a txt file (downloaded from webpage) into a table

if (historical && ~isempty(fileName))
    url = strcat('https://www.ndbc.noaa.gov/view_text_file.php?filename=46011h',...
                 year,'.txt.gz&dir=data/historical/stdmet/');
    fileNamePath = websave(fileName, url);
    data = readtable(fileNamePath, 'ReadVariableNames',true);
    data.Properties.VariableNames{1} = 'YY';
    data.Properties.VariableUnits = {'yr',  'mo', 'dy', 'hr', 'mn', 'degT',...
                                     'm/s',  'm/s', 'm', 'sec', 'sec', 'degT'...
                                     'hPa',  'degC',  'degC', 'degC', 'mi','ft'};
elseif ~isempty(fileName)
    url = 'https://www.ndbc.noaa.gov/data/realtime2/46011.txt';
    fileNamePath = websave(fileName, url);
    data = readtable(fileNamePath, 'ReadVariableNames',true, 'TreatAsMissing', 'MM');
    data.Properties.VariableNames{1} = 'YY';
    data.Properties.VariableUnits = {'yr',  'mo', 'dy', 'hr', 'mn', 'degT',...
                                'm/s',  'm/s', 'm', 'sec', 'sec', 'degT'...
                                'hPa',  'degC',  'degC', 'degC', 'mi','hPa',...
                                'ft'};
end
end
```

Figure 5.1. *Implementation of data loading function*

5.2 Visual inspection of data

Loading function called in main file *ExploratoryDataAnalysis.mlx* makes the data table read with scrolling to be displayed (see figure 5.2 and 5.3).

```
historical = true;
dataRead2019 = loading('SantaMaria2019.txt',historical,'2019')
```

dataRead2019 = 16214×18 table

	YY	MM	DD	hh	mm	WDIR	WSPD	GST	WVHT	DPD	APD
1	2018	12	31	23	50	321	5.6000	7.0000	2.6700	10.8100	7.8000
2	2019	1	1	0	50	332	7.1000	8.4000	2.6400	11.4300	7.6100
3	2019	1	1	1	50	329	7.2000	9.1000	2.7400	10.0000	7.6600
4	2019	1	1	2	50	3	6.2000	8.4000	2.9400	10.0000	7.7500
5	2019	1	1	3	50	36	7.4000	8.4000	2.9100	10.8100	7.9700
6	2019	1	1	4	50	52	7.3000	8.7000	2.8900	10.8100	7.7200
7	2019	1	1	5	50	35	8.0000	9.7000	2.7800	10.8100	7.5300
8	2019	1	1	6	50	36	6.7000	8.5000	2.5000	11.4300	7.2800
9	2019	1	1	7	50	58	6.6000	8.2000	2.4200	10.0000	7.3500

Figure 5.2. *Call to 2019 data loading and data table displayed*

dataRead2019 = 16214×18 table

	APD	MWD	PRES	ATMP	WTMP	DEWP	VIS	TIDE
1	7.8000	319	1013	12.6000	13.6000	999	99	99
2	7.6100	314	1.0132e+03	12.7000	13.6000	999	99	99
3	7.6600	318	1.0135e+03	12.7000	13.6000	999	99	99
4	7.7500	316	1.0141e+03	13.0000	13.5000	999	99	99
5	7.9700	323	1.0146e+03	12.2000	13.5000	999	99	99
6	7.7200	319	1.0155e+03	12.1000	13.5000	999	99	99
7	7.5300	313	1.0161e+03	11.8000	13.5000	999	99	99
8	7.2800	321	1.0162e+03	11.6000	13.5000	999	99	99
9	7.3500	329	1.0166e+03	11.5000	13.5000	999	99	99

Figure 5.3. *Continuation of data table displayed*

Taking a first look at data and features (columns) has helped us to find out the following aspects:

- **Data types** - We check that data types for every column read are numbers (int or float type) and we establish the ranges of the values of variables.
- **Columns with too little data examples available** - When we look around, we discover some variables do not have any information.Particularly, last three columns are completely null (full of 9 series in historical files). These columns correspond to variables DEWP (dewpoint temperature), VIS (station visibility limited for buoys from 0 to 1.6 nautical miles) and TIDE (The water level in feet above or below Mean Lower Low Water). These variables seem not to be closely related with wind and waves forecast as they are not deemed necessary for offshore wind energy applications in the creation of metocean data set for FOWT simulations [67]. With this in mind, we will handle them in Cleaning data section.
- **Dates and Times** - First five columns represents time information (year, month, day, hour and minute respectively) of row measurements.They have been coded as one variable,Date, for time series visualization.On the other hand, we observe that minutes columns (mm) is always equals to 50 in historical files. The same is not true for real-time files which contains rows for each 10 minutes measurements. It

seems that rows which mm column different from 50 contains NaN values in some important columns like WVHT (Significant wave height) or DPD (Dominant wave period) as we can see in figure 5.4. This factor should be considered in Feature Engineering section because mm variable in the training set does not provide useful information and the accuracy of the model may be affected.

dataReadRealTime = 6457×19 table

	mm	WDIR	WSPD	GST	WVHT	DPD	APD	MWD	PRES	ATMP	WTMI
1	0	220	6	7	NaN	NaN	NaN	NaN	1.0169e+03	12.1000	
2	50	220	6	7	1.4000	17	7.4000	240	1.0169e+03	12.1000	12.
3	40	220	6	7	NaN	NaN	NaN	NaN	1.0171e+03	12.1000	12.
4	30	210	4	6	NaN	NaN	NaN	NaN	1.0177e+03	12.0000	12.
5	20	210	5	6	NaN	NaN	NaN	NaN	1.0178e+03	12.0000	12.
6	10	240	4	5	NaN	NaN	NaN	NaN	1.0176e+03	12.0000	12.
7	0	250	5	6	NaN	NaN	NaN	NaN	1.0175e+03	11.9000	12.
8	50	260	4	6	1.5000	16	8.7000	209	1.0179e+03	11.9000	12.
9	40	270	3	4	1.5000	NaN	8.7000	209	1.0178e+03	11.7000	12.

Figure 5.4. *Real-Time data table displayed after loading process*

The loaded data model we will work on is represented in the following diagram:

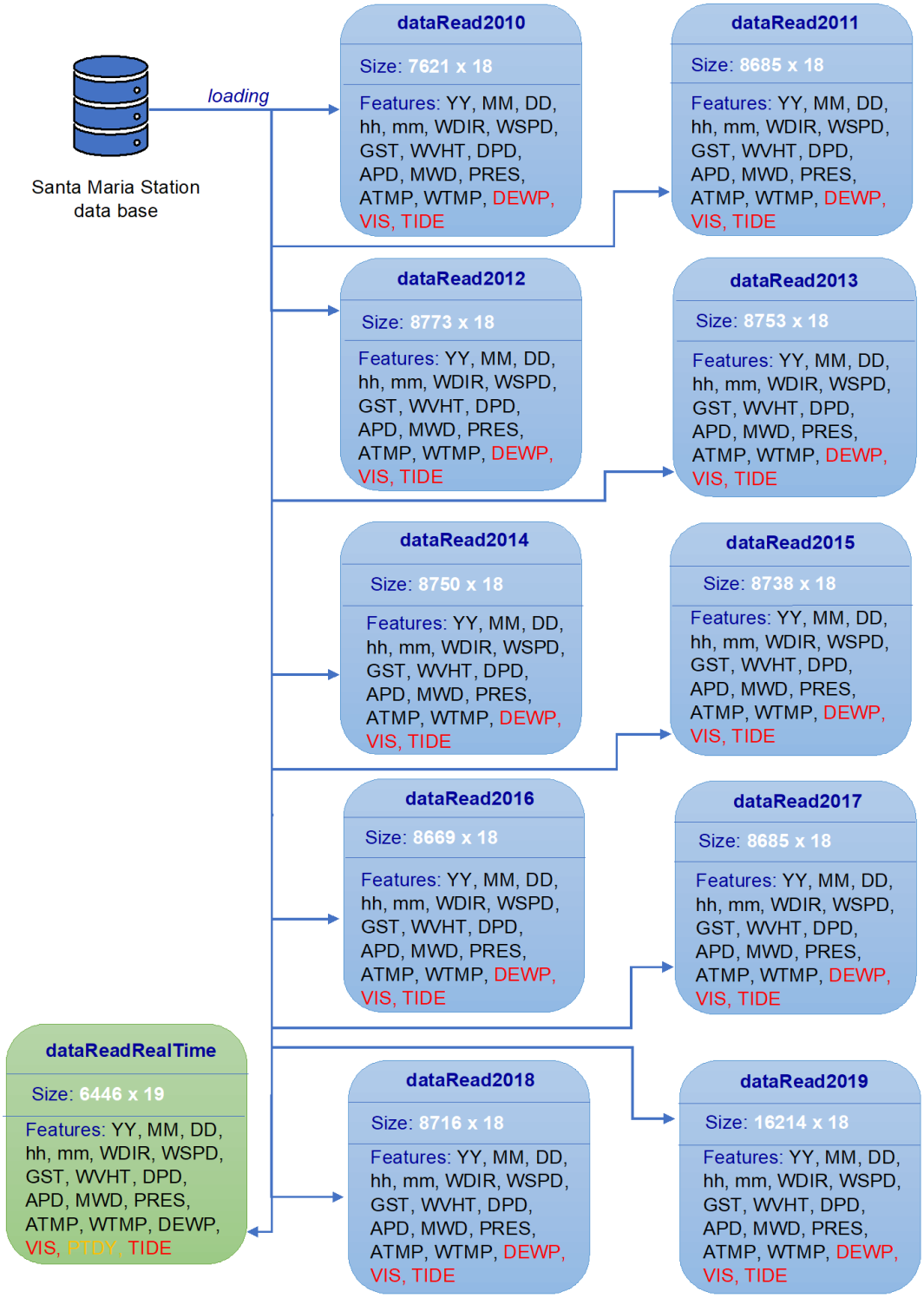
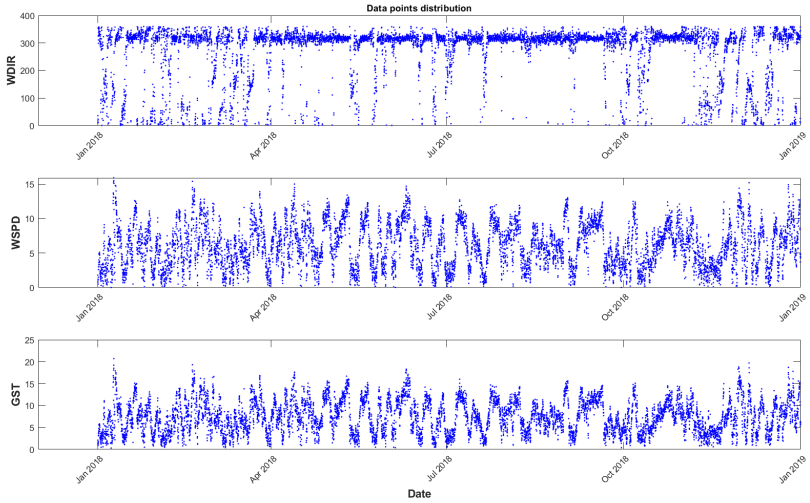
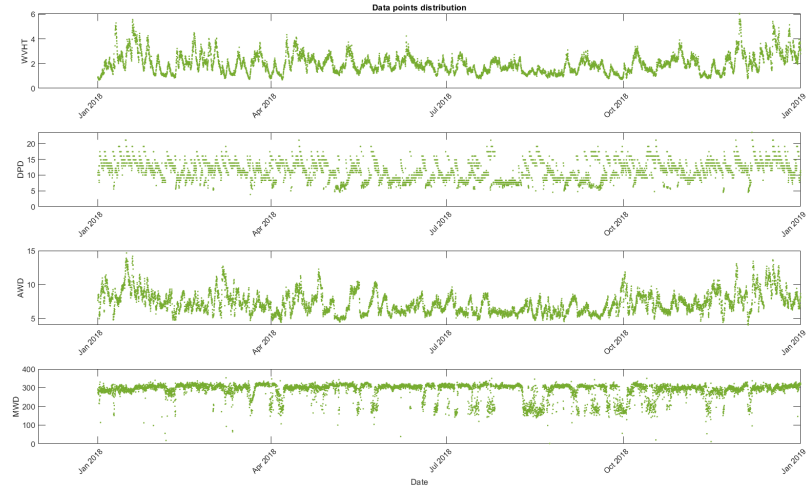


Figure 5.5. *Data model diagram before cleaning process. Red features represent null variables and orange one are available just in Real-Time file.*

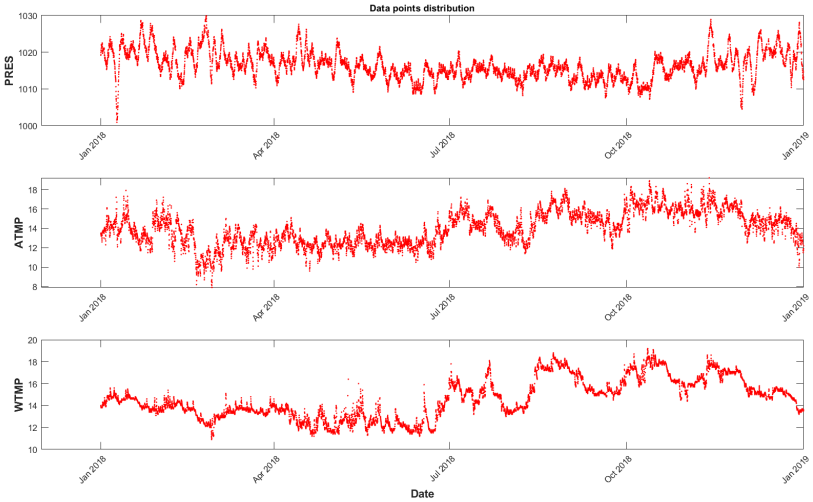
Null variables will be directly discarded. In figure 5.6, Data distribution of waves, wind, temperature and pressure features have been graphed for year 2018.



(a) Wind features



(b) Waves features



(c) Temperature and Pressure features

Figure 5.6. *Wind (a), Waves (b), Temperature and Pressure (c) features time series for 2018*

5.3 Data cleaning

Incorrect records on a data set has an unimaginable impact on the quality of the conclusions and results given by the algorithm that uses it. Indeed, Data cleaning is a time-consuming process that takes care of inspecting for missing values and inconsistencies and errors in data, cleaning them by fixing or removing these registers discovered and verifying correctness of data after cleaning.

We will apply it to the data set of each year in two phases addressed in the following subsections:

- Missing data handling
- Outliers detection

Finally, we will represent features time series to compare it with those represented before the clean-up to ensure that no error has been made and the distribution of data is maintained.

5.3.1 Missing data handling

The missing values in meteocean data set can be caused by a specific fault in the bouy sensors or by the lack of measuring in the selected station buoy.

Features preparation

First step has been to select useful variables, discarding null variables observed in the previous section. DEWP, VIS and TIDE from every table load and PITDY from real time table have been excluded (see figure 5.7).

```
%Extracting Features available in both files and not completely null
dataUsed2018 = dataRead2018(:, {'YY', 'MM', 'DD', 'hh', 'mm', 'WDIR', 'WSPD','GST', 'WVHT','DPD', 'APD', 'MWD', 'PRES', 'ATMP','WTMP'});
dataUsed2019 = dataRead2019(:, {'YY', 'MM', 'DD', 'hh', 'mm', 'WDIR', 'WSPD','GST', 'WVHT','DPD', 'APD', 'MWD', 'PRES', 'ATMP','WTMP'});
dataUsed2018 = dataRead2018(:, {'YY', 'MM', 'DD', 'hh', 'mm', 'WDIR', 'WSPD','GST', 'WVHT','DPD', 'APD', 'MWD', 'PRES', 'ATMP','WTMP'});
dataUsed2017 = dataRead2017(:, {'YY', 'MM', 'DD', 'hh', 'mm', 'WDIR', 'WSPD','GST', 'WVHT','DPD', 'APD', 'MWD', 'PRES', 'ATMP','WTMP'});
dataUsed2016 = dataRead2016(:, {'YY', 'MM', 'DD', 'hh', 'mm', 'WDIR', 'WSPD','GST', 'WVHT','DPD', 'APD', 'MWD', 'PRES', 'ATMP','WTMP'});
dataUsed2015 = dataRead2015(:, {'YY', 'MM', 'DD', 'hh', 'mm', 'WDIR', 'WSPD','GST', 'WVHT','DPD', 'APD', 'MWD', 'PRES', 'ATMP','WTMP'});
dataUsed2014 = dataRead2014(:, {'YY', 'MM', 'DD', 'hh', 'mm', 'WDIR', 'WSPD','GST', 'WVHT','DPD', 'APD', 'MWD', 'PRES', 'ATMP','WTMP'});
dataUsedRealTime = dataReadRealTime(:, {'YY', 'MM', 'DD', 'hh', 'mm', 'WDIR', 'WSPD','GST', 'WVHT','DPD', 'APD', 'MWD', 'PRES', 'ATMP','WTMP'});
```

Figure 5.7. *Features not completely null selected from data tables read*

There are three types of techniques for handling missing data:

- **Imputation**. Imputation consists of fill in or impute missing values in a data set. It is common to replace missing values with a mean value of the affected variable for all data set examples or with a close value in time series.

- **Interpolation**. the interpolation method aims to mathematically estimate the missing value at the point, considering the rest of the points in data set.
- **Deletion**. This technique is based on eliminating the rows of the data set that contain missing values

Among the three types of techniques, we have applied deletion. This decision is taken because of the large amount of data available on the website (annual data sets since 1980) and the lack of expert knowledge of the domain necessary to interpolate or estimate the missing registers. Therefore, rows containing missing values in at least one of the selected columns have been deleted. According to the percentage amount of clean data that passed the filter in relation to the total data in a year read table, loading more data from previous years has been considered.

To visualize the amount of missing data for each year by the date and by feature, we have plotted a graphic with date in the x-axis and two categories, missing value (Rejected) or valid one (Useful), in the y-axis. This will allow us to carry out the classification of the data rows according to the indexes matrix. The index 1 indicate there is a missing value in that position(see figure 5.8)[1].

```
historical = true;
[data2019, idxNaN2019]  = cleaning(dataUsed2019,historical);
plotCleaningResult(dataUsed2019, idxNaN2019,'2019')
```

Figure 5.8. *Call to cleaning function and plot cleaning results for 2019 read data*

We must ensure that we have sufficient examples of data so that all the dates are represented in them and our models are accurate to forecast variables at any time of the year.

We have loaded year data files from 2010 to 2019 and Real time data files which have been published on the web progressively over the past few months of 2020.

Data from 2019

We observe data rows from 2019 contains missing values in all features (see figure 5.9). At first glance, the variables that contain the most missing values are Wind direction, Wind speed and Peak of gust speed. On the other hand, in the month of October, there are no rows of measurements available in the data set so we will focus on getting data for October from other years.

[1]Code implemented of *cleaning* and *plotCleaningResult* functions can be consulted in the repository.

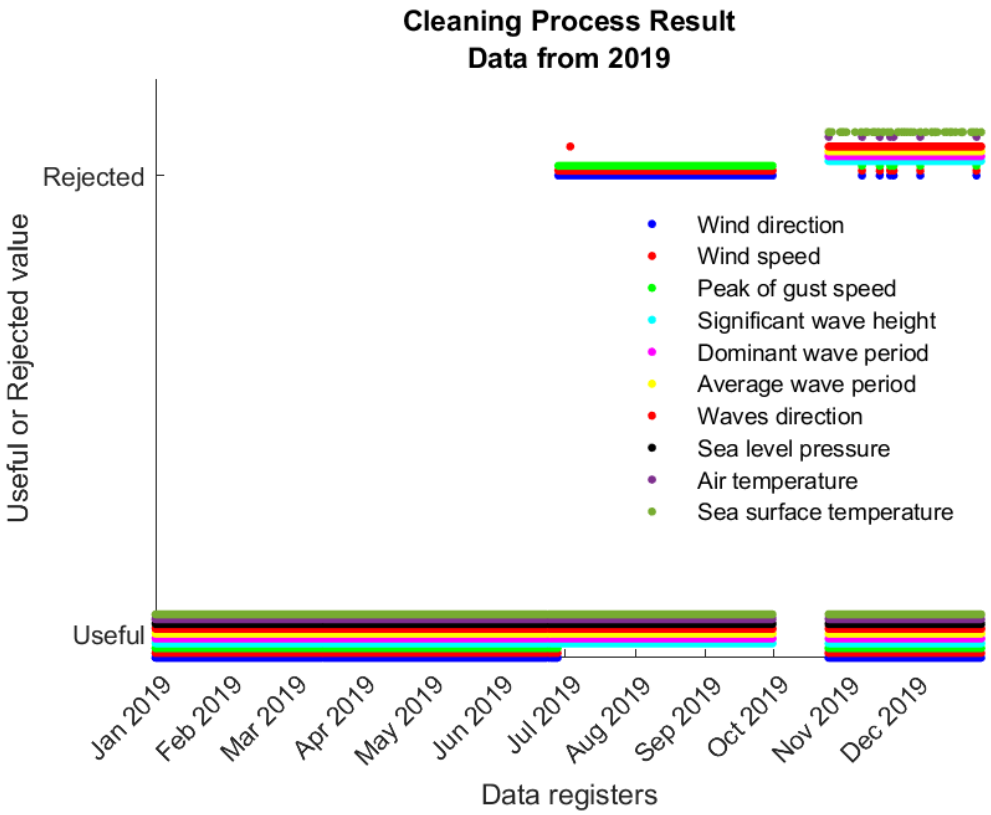


Figure 5.9. *Missing values plot for 2019*

The results of 2019 data cleaning are summarized in the table 5.1 which shows the number of initial rows in data set, the number of rows after cleaning process, the number of deleted rows (the difference between previous ones) and the percentage of rows removed from the total number of rows available before cleaning.

Data Cleaning results - 2019	
Rows before cleaning	16214
Rows after cleaning	5803
Rejected rows	10411
% of rejected rows	**64.2099**

Table 5.1. *Data cleaning summary results for 2019 data set*

Data from 2018

Data from 2018 is surprisingly complete and it does not have a sigle missing value as it can be seen from the plot in the figure 5.12 and check in table 5.2.

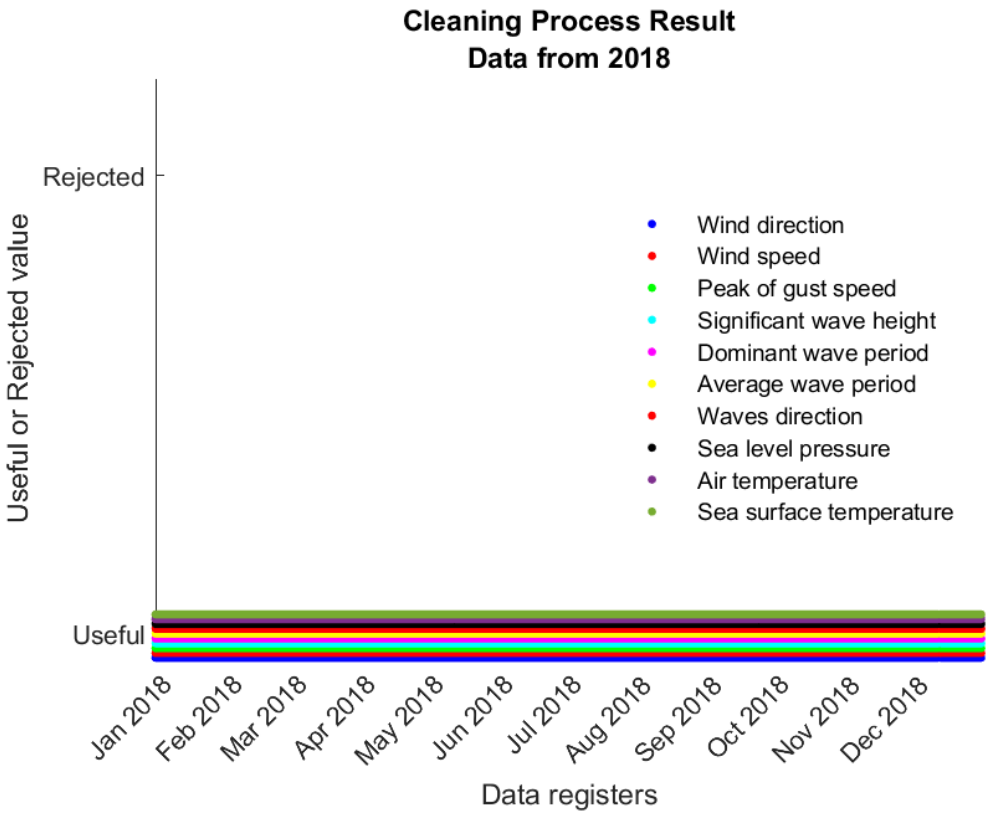


Figure 5.10. *Missing values plot for 2018*

Data Cleaning results - 2018	
Rows before cleaning	8716
Rows after cleaning	8716
Rejected rows	0
% of rejected rows	**0**

Table 5.2. *Data cleaning summary results for 2018 data set*

Data from 2017

Data from 2017 is has just 1 row with missing values for wind speed, significant wave height, average wave period and dominant wave period features, as shown below.

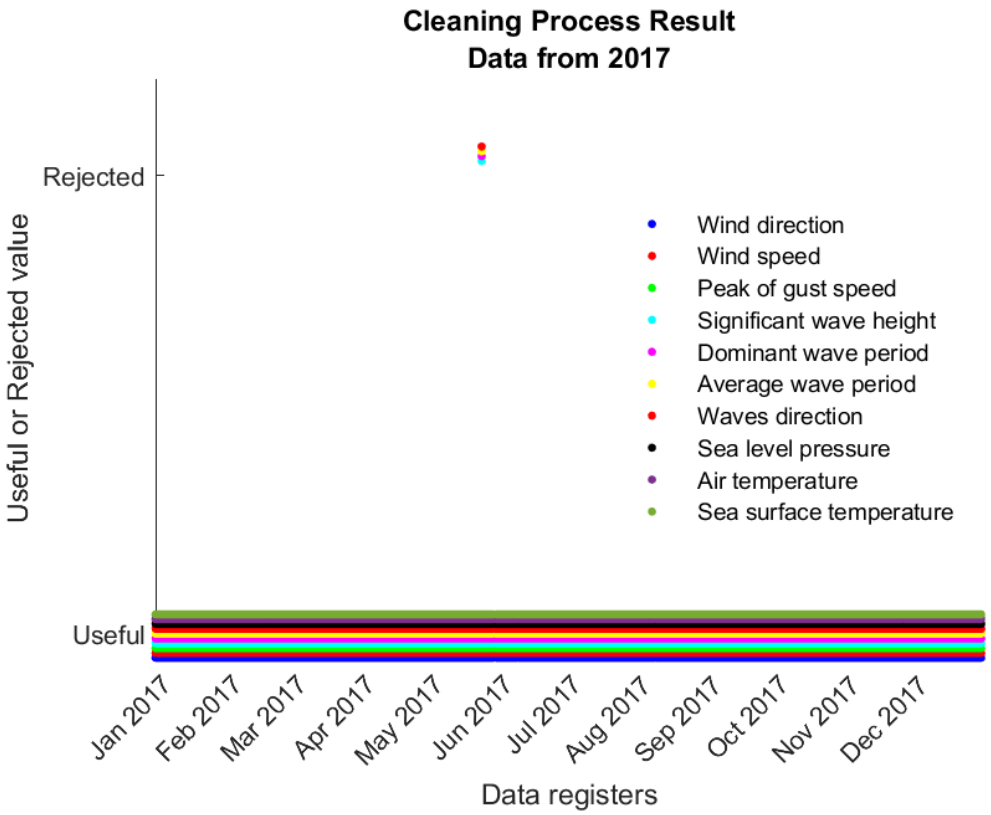


Figure 5.11. *Missing values plot for 2017*

Data Cleaning results - 2017	
Rows before cleaning	8685
Rows after cleaning	8684
Rejected rows	1
% of rejected rows	**0.0115**

Table 5.3. *Data cleaning summary results for 2017 data set*

Data from 2016

Data from 2016 does not have any missing value as it happened with 2018 data set.

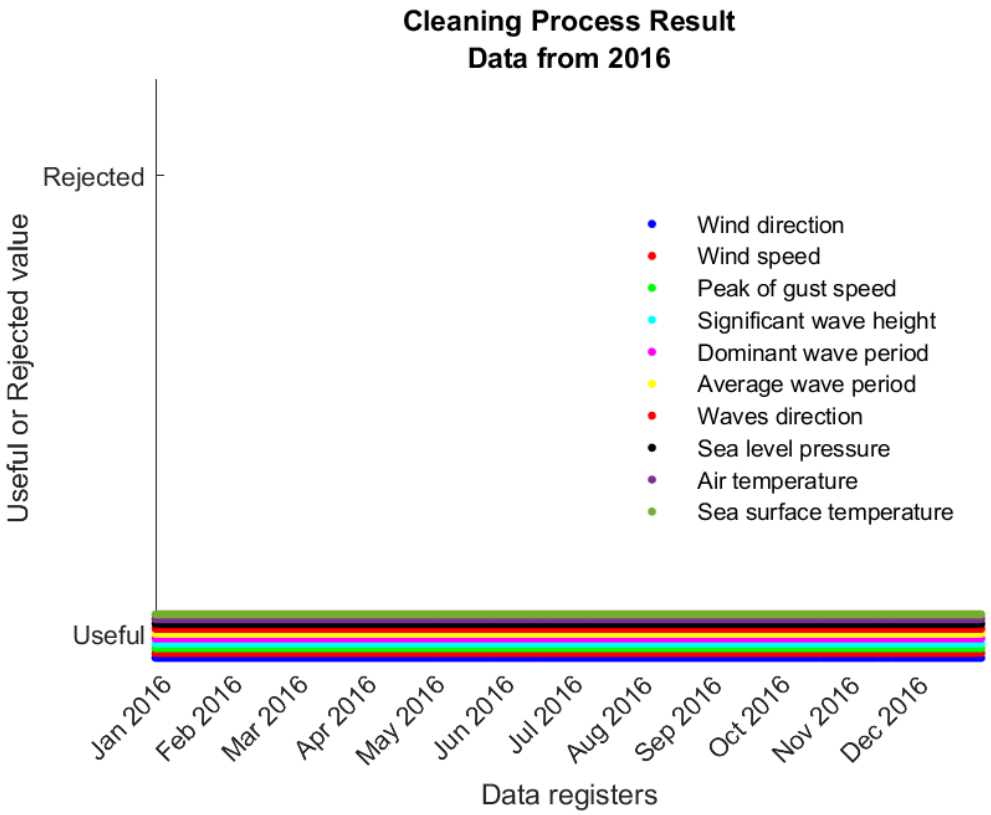


Figure 5.12. *Missing values plot for 2016*

Data Cleaning results - 2016	
Rows before cleaning	8669
Rows after cleaning	8669
Rejected rows	0
% of rejected rows	**0**

Table 5.4. *Data cleaning summary results for 2016 data set*

Data from 2015

In the 2015 data set, we observe wind speed, wind direction and peak of gust speed columns are full of missing values until June approximately. However, we distinguish columns like dominant wave period or average wave period that contains both missing values and available measurements along year time line (see figure 5.13).

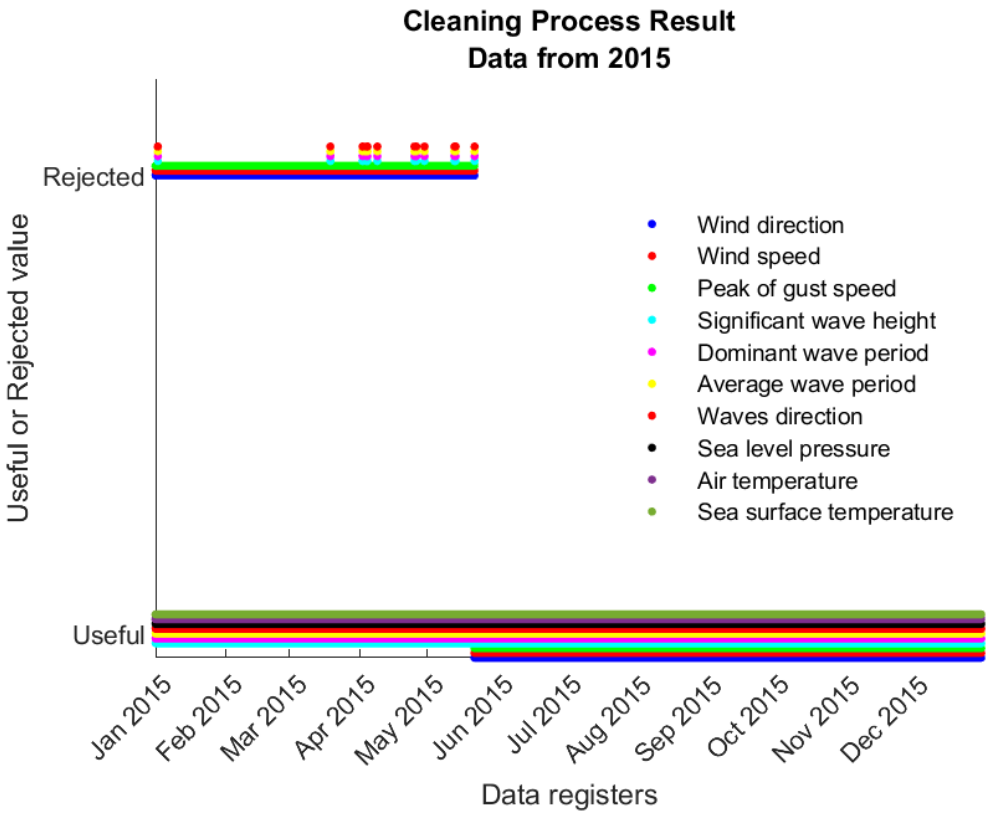


Figure 5.13. *Missing values plot for 2015*

Data Cleaning results - 2015	
Rows before cleaning	8738
Rows after cleaning	5358
Rejected rows	3380
% of rejected rows	**38.6816**

Table 5.5. *Data cleaning summary results for 2015 data set*

Data from 2014

We explore the plot for the last historical data table load from 2014. It seems in plot 5.14 that many rows contains at less one missing value. We will check it numerically , as shown in table 5.6.

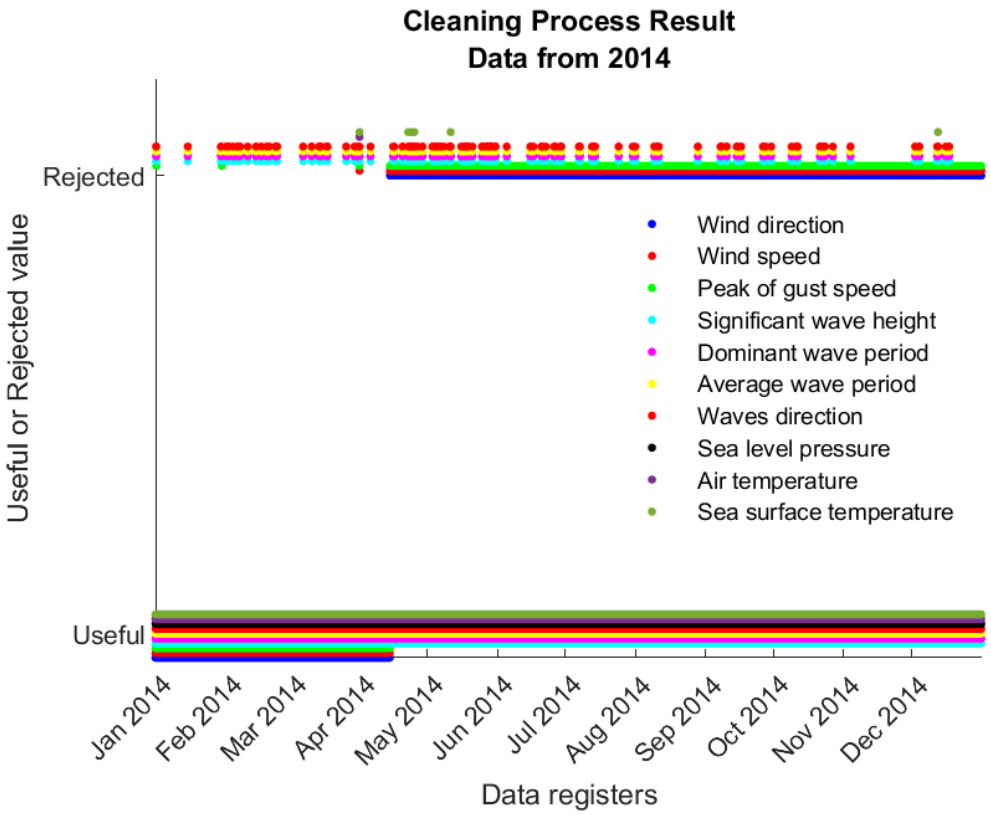


Figure 5.14. *Missing values plot for 2014*

Data Cleaning results - 2014	
Rows before cleaning	8750
Rows after cleaning	2444
Rejected rows	6306
% of rejected rows	**72.0686**

Table 5.6. *Data cleaning summary results for 2014 data set*

Data from last 45 days (Real-Time)

As it is obvious, the number of rows in Real-Time data table load is fewer in number than those in the data tables from previous years.Before calling plotting the cleaning results, we have order the rows in table from the oldest data (row one) to the most recent date(last row) in order to be able to plot it chronologically as we have done with data tables above. This data will be used for testing our models after training phase. Let's explore in detail the plot showed in figure 5.15.

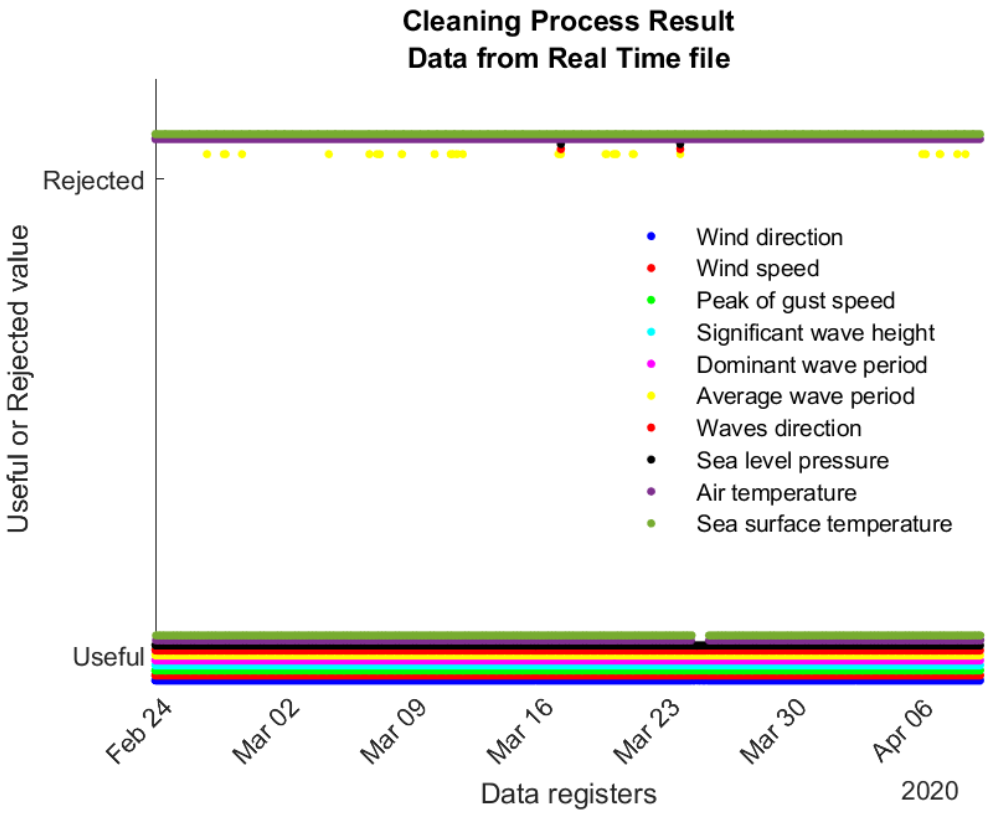


Figure 5.15. *Missing values plot for last 45 days data or Real-time data*

Data is available from mid-February to the end of March and there are missing values are located in sea surface temperature, air temperature, sea level pressure, waves direction and average wave period columns. Looking at the table 5.7, we are shocked that more than 83% of rows have been deleted for having missing values, but if we look at the table after cleaning (see figure 5.16) we confirm our assumption that the rows with the value 50 in minutes or mm column are the only without any missing value.

Data Cleaning results - Last 45 days	
Rows before cleaning	6464
Rows after cleaning	1039
Rejected rows	5425
% of rejected rows	**83.9264**

Table 5.7. *Data cleaning summary results for Real-time data set*

dataRealTime = 1039x16 table

	YY	MM	DD	hh	mm	WDIR	WSPD	GST	WVHT	DPD	APD	MWD
1	2020	2	24	0	50	320	11	14	2.8000	15	6.6000	314
2	2020	2	24	1	50	330	11	14	2.9000	14	6.6000	321
3	2020	2	24	2	50	320	12	15	3.0000	15	6.6000	315
4	2020	2	24	3	50	330	11	14	3.0000	15	6.8000	317
5	2020	2	24	4	50	330	10	13	3.2000	15	7.1000	316
6	2020	2	24	5	50	330	10	12	3.2000	15	7.1000	314
7	2020	2	24	6	50	330	11	14	3.0000	14	7.1000	326
8	2020	2	24	7	50	330	10	13	3.2000	14	7.3000	320
9	2020	2	24	8	50	330	10	13	3.0000	14	7.4000	317
10	2020	2	24	9	50	340	10	12	3.0000	14	7.4000	318
11	2020	2	24	10	50	330	7	10	3.1000	13	8.0000	316
12	2020	2	24	11	50	330	9	11	3.0000	14	7.6000	321
13	2020	2	24	12	50	340	9	11	2.7000	14	7.0000	307
14	2020	2	24	13	50	340	7	9	2.8000	12	7.5000	309
15	2020	2	24	14	50	340	7	9	2.8000	14	7.5000	299
16	2020	2	24	15	50	350	9	10	2.7000	15	7.6000	304
17	2020	2	24	16	50	340	8	11	2.7000	12	7.4000	303
18	2020	2	24	17	50	320	8	11	2.5000	14	7.1000	302
19	2020	2	24	18	50	330	9	11	2.4000	14	7.1000	306
20	2020	2	24	19	50	320	9	12	2.5000	12	7.3000	297
21	2020	2	24	20	50	310	10	12	2.5000	14	7.0000	304
22	2020	2	24	21	50	320	10	12	2.4000	15	6.7000	299
23	2020	2	24	22	50	320	11	13	2.4000	14	6.8000	305
24	2020	2	24	23	50	320	9	11	2.6000	14	7.2000	305
25	2020	2	25	0	50	320	10	13	2.5000	14	7.1000	309

Figure 5.16. *Real-time data table fragment after cleaning process*

In essence, we have found that loaded data from 2018 and 2017 is pretty complete while data tables from 2019 ans 2016 contained more than 60 % of rows with one or more missing values. In order to have sufficient data examples for optimization model phase and compensate deleted rows previously, we decide to load and clean data from years 2013, 2012, 2011 and 2010 too (Graphics and percentages obtained for these years are available in repository).

5.3.2 Outliers detection

Detecting outliers or anomalies in data is one of the core problems of data-mining to be addressed when preprocessing data sets in order to develop models by machine learning techniques. Some ML algorithms, like logistic or linear regression, are sensitive to features distributions of inputs examples and patterns contained in the data. The problem comes when outliers distort the actual distribution of data resulting in less accurate models and ultimately poorest results [73].

Outliers can be caused by human errors, errors in data recording or collection, environmental conditions, unusual phenomena (hurricanes, typhoons,etc.) or a faulty and non-calibrated sensor. Detecting these anomalies is a challenge and risky task because the normal and unusual behaviour definitions vary greatly depending on the correction technique used.

We have decided to apply **the majority voting method** consisting of execute more than one anomalies detection algorithm and consider anomalies just those points classified as outliers by at least two of the run algorithms.

Particularly, we will use **Box-plot** and **Local distance-based outlier factor (LDOF)** techniques independently so outliers will be points classified as anomalies by both algorithms. In the implementation of the LDOF (the main function is showed in figure 5.17) 150 has been chosen for k considering this value enough to obtain n accurate results according to Mahmoodi and Ghassemi in their study [74].

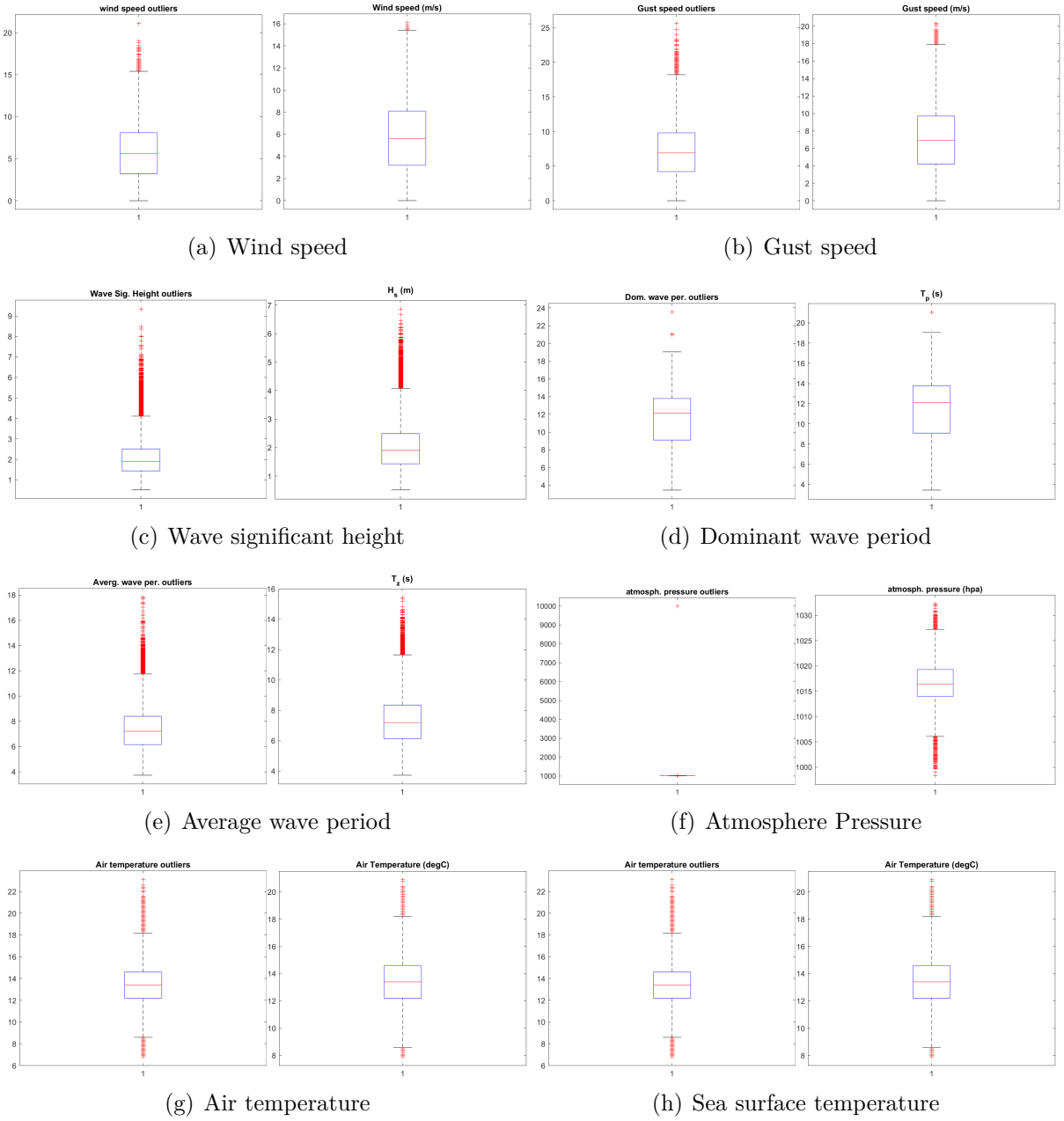
It is important to clarify that in the case of wind and wave direction variables we cannot apply the above techniques because they are circular variables [75]. That means they are not in a linear scale so an angle of x is the same angle as $x + k \times 360$ for real positive x and natural k values. Moreover, every value taken by wind and wave direction after doing mod operation with reference to 360^{o} will be inside the range $[0, 360)$ so that, it not makes sense to apply outlier or "out of range" points identification. We are going to use CircStat toolbox to explore the directional variables [76] in next subsections.

```
function [LDFO, D150,Idx150, knnInnerDistance]=  ldfo_outliers(data)
%This method calculate the coefficient LDFO for every point in data set
%Author: Montserrat Sacie Alcázar
%%-------------------------------------------------------------------------
%Create the model by training with the data set: data
Mdl =  ExhaustiveSearcher(data);

%Find 150 nearest neighbour for each point (k = 150)
k = 150;
Idx150 = knnsearch(Mdl, data, 'K',k);
D150 = nearestNeighboursDistance(data, Idx150,k); %para cada punto

%Calculate the k-inner distance for each point
knnInnerDistance = zeros(size(Idx150,1),1);
for p = 1:size(Idx150,1)
    %Distance for all posible pair-points between k neighbours for
    %register p
    kpoints = data(Idx150(p,:),:);
    idxM = combnk(1:size(kpoints,1),2);
    D = euclideanDistance(kpoints(idxM(:,1),:),kpoints(idxM(:,2),:));
    knnInnerDistance(p,1) =  1/(k*(k-1)) * sum(D);
end

%Calculate the LDFO coefficient
LDFO = D150./ knnInnerDistance;
save('resultado.mat')
end
```

Figure 5.17. *Local distance-based outlier factor function implementation in Matlab*

Boxplots for features of interests are represented in figure 5.18 before deleting outliers (left column) and after deletion (right column). As we observe, the majority voting method implies that not all outliers identified with the boxplot method are eliminated. Intuitively,

this technique is in line with the objective of eliminating the minimum number of rows as possible from the data, since by deleting a row with an anomalous value we also delete the useful values of the rest of the variables.



(a) Wind speed (b) Gust speed

(c) Wave significant height (d) Dominant wave period

(e) Average wave period (f) Atmosphere Pressure

(g) Air temperature (h) Sea surface temperature

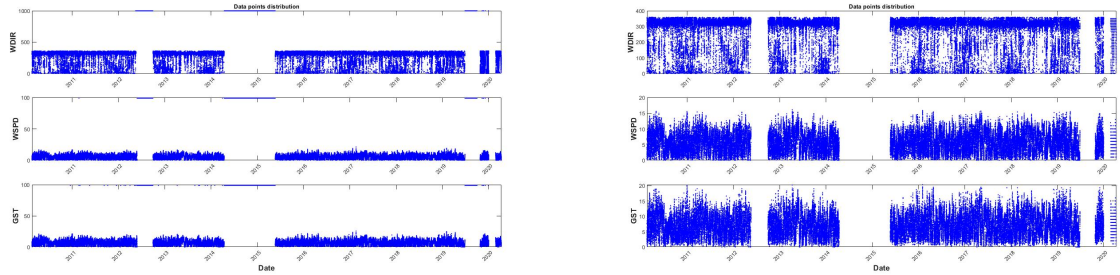
Figure 5.18. *Boxplot before (left) and after (right) outlier deletion for wind speed (a), Gust speed (b), H_s (c), T_p (d), T_z (e), Atmosphere pressure(f), Air temp. (g) and Sea surface temp.(h) features*

The results of outliers deletion are presented in the table 5.8. Among the 71031 rows of the total data set (from 2010 to Real-time), 1749 outliers have been identified with at least one anomalous value and have been deleted.

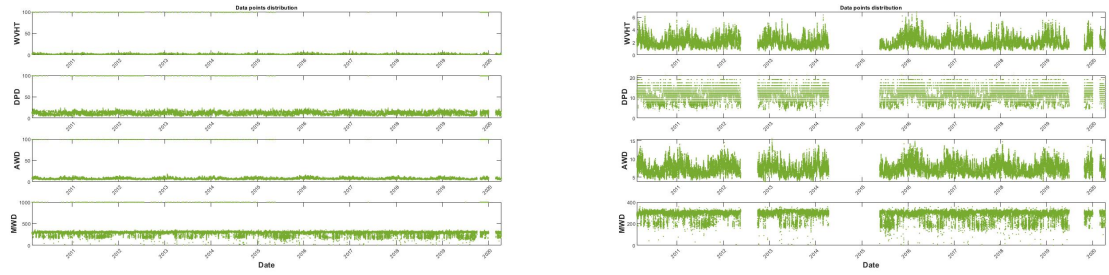
Used technique	Number of identificated outliers
Boxplot	5330
LDOF	4820
Final number of deleted outliers	1749

Table 5.8. *Outliers detection and deletion results*

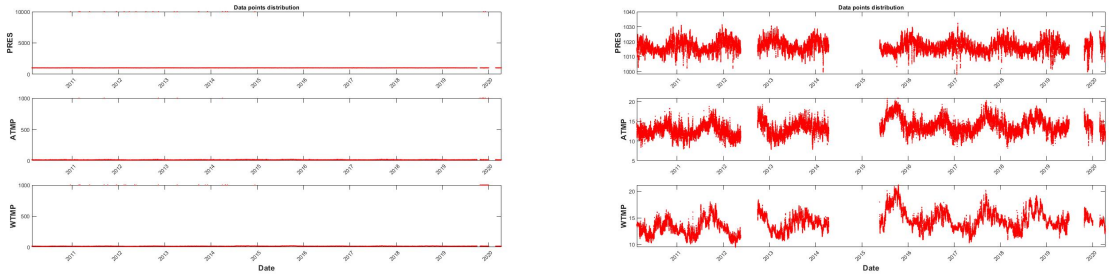
After data cleaning, we have plotted each feature time-series using clean data and row data collected without cleaning in order to ensure that the distributions of them are the same and we have not made any mistakes during the cleaning process that could have modified it. As seen in figure 5.19, deleting outliers is especially significant as we see how the maximum extreme values disappear with these points deletion, creating the visual effect of expansion of the graph (upper limit is reduced). We can zoom in on figures in Matlab to check the distribution with more guarantee.



(a) Wind features



(b) Wave features



(c) Air temperature and pressure features

Figure 5.19. *Features time series before (left) and after (right) Data cleaning process*

5.4 Univariate analysis

The first analysis made to data will be the separate study of each main metocean characteristic taken from the buoy location without caring about relationships or causes. The main features to study are those we want to model in the project:

- Wind speed (m/s)
- Significant waves height (m)
- Wind direction (º)
- Waves direction (º)

Descriptive statistics including central tendency measures (mean, median and mode), spread measures (range of values, standard deviation, variance and quartiles) and shape descriptive measures (central moment, maximum and minimum values) are calculated for each feature.

Finally, we will look for the best **density distribution** that fits to each feature for the examples in the data set. We will compare the results with Stewart's ones obtained in their analysis and creation of a metocean database and data prototypes of 27 NOAA stations data [67].

We will use for this purpose

- Statistics functions
- histograms with fit line to data distribution (histfit)

Data from the year 2018 have been used in this study, although results can be extrapolated to another year's data. Data from this year has been selected because it is one of the annual data sets loaded from the page with less missing values and anomalous points detected.

Wind speed (WSPD)

Wind speed variable is one of the characteristic we want to model in this project. Statistical measures for annual wind speed are summarized in table 5.9

Wind speed (m/s)		
Central measures	**mean**	6.0450
	median	6
	mode	6.4
Spread measures	**range**	15.4
	std	3.1328
	variance	9.8145
	quartiles	[0, 3.4, 6, 8.5, 15.4]
Shape measures	**central mom.**	5.1228
	max.	15.4
	min.	0

Table 5.9. *Descriptive statistics calculated for annual Wind speed (units: m/s)*

Median and mean wind speed values are almost equal while mode is a bit higher. Winds in Santa Maria station appears to be stable which makes this location suitable for FOWT installation. Histogram of WSPD observations distribution is plotted in figure 5.20. We can observe wind speed data is a bit left skewed.

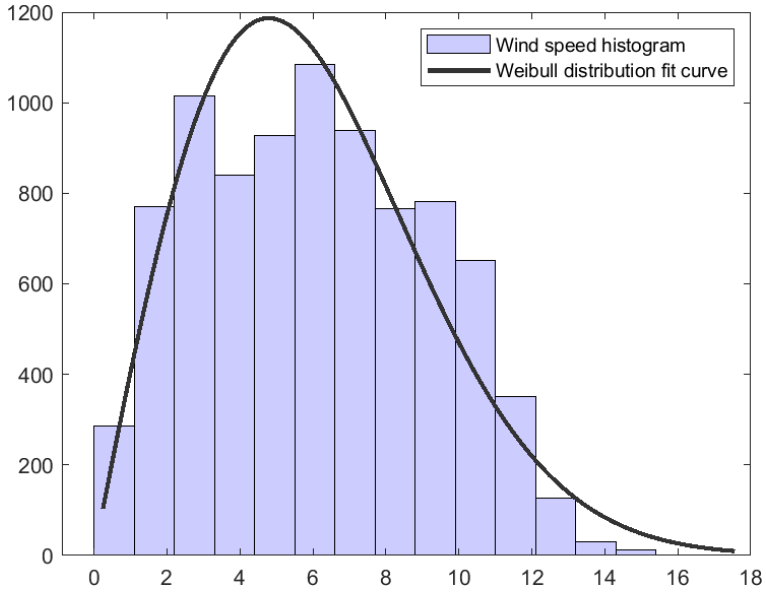


Figure 5.20. *Wind speed histfit plot*

As Stewart claims in [67], Weibull distribution best fit to Wind speed data.

Significant Waves height (WVHT)

Significant Waves Height is another important feature to consider. Descriptive statistics of WVHT are listed in table 5.10.

Significant waves height (m)		
Central measures	**mean**	1.9779
	median	1.87
	mode	1.67
Spread measures	**range**	4.82
	std	0.7523
	variance	0.5660
	quartiles	[0.67, 1.42, 1.87, 2.39, 5.49]
Shape measures	**central mom.**	0.4237
	max.	5.49
	min.	0.67

Table 5.10. *Descriptive statistics calculated for annual Sig. waves height (units: m)*

Histogram plot for WVHT (figure 5.21) reveals that Gamma distribution best fit to data, coinciding again with Stewart [67].

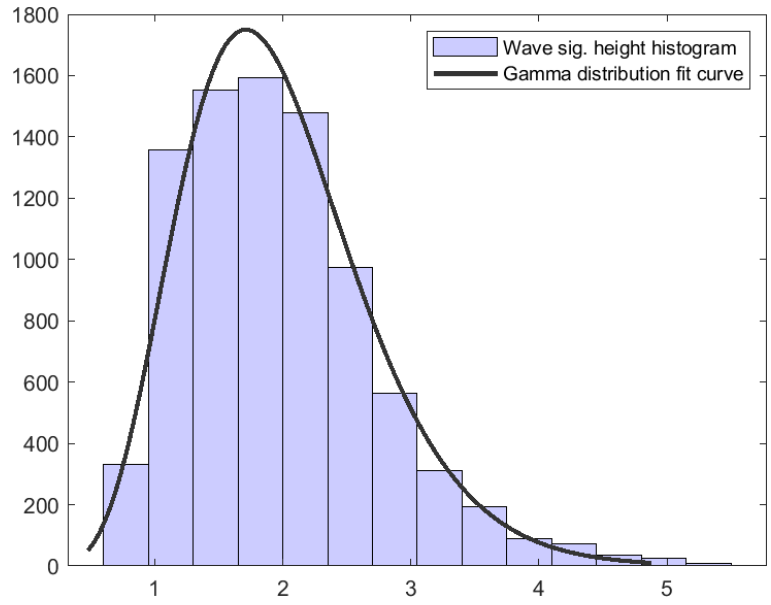


Figure 5.21. *Significant waves height histfit plot*

Wind direction (WDIR) and Waves direction (MWD)

Wind and waves directions are circular variables so the way they have been explored was different. For this purpose, we used the toolbox CircStat of Matlab [76].

Firstly, we have plotted the wind rose of Santa Maria station (see figure 5.22) to visualize the relation between wind speed and wind direction.

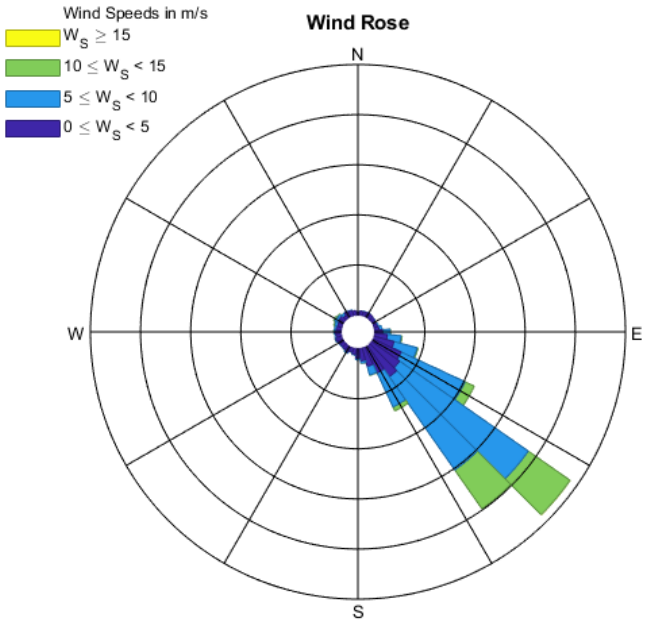


Figure 5.22. *Wind rose plot*

Descriptive statistic for wind direction are summarized in table 5.11

Wind direction (º)		
Central measures	**mean**	-39.74
	median	-41.99
	R	0.6936
Spread measures	**std**	44.85
	variance	17.56
Shape measures	**central mom.**	1

Table 5.11. *Descriptive statistics calculated for annual Wind direction (units: º)*

On the other hand, waves direction statistics measures are listed in table 5.12

Waves direction (º)		
Central measures	**mean**	-67.54
	median	-58
	R	0.7846
Spread measures	**std**	37.61
	variance	12.34
Shape measures	**central mom.**	1

Table 5.12. *Descriptive statistics calculated for annual waves direction (units: º)*

If we compare central measures of wind direction with those of waves direction, we can confirm that the misalignment between both directions is remarkable, since the WDIR mean is approximately 30º away from MWD mean. Circular variables are best fit by the Von Mises distribution or the circular normal distribution according to literature [67].

The same analysis has been made for the rest of the features available in the data set which can be found in *ExploratoryDataAnalysis.mlx* file in the repository.

5.5 Correlation analysis

Exploring and measuring the correlation between a predictive variable and the rest of the available features is fundamental to making a good selection of predictors. This means that including insignificant variables loosely related to the target variable in the training set could lead us to obtain a poorest performance model as well as an increment of the computational cost. By contrast, predictors must not be highly correlated with each other

To measure the linear correlation we have calculated **Pearson** and **Spearman correlation coefficients** for pair-wise features, paying particular attention to the relationship of all variables with respect to:

- wind speed
- wind direction
- waves significant height
- and waves direction

After calculating correlation coefficients, we have found that Pearson and Spearman's coefficients are similar for huge amounts of data and we draw the same conclusions from both. Thus just Pearson coefficients are presented here (Spearman's coefficients are available in the repository).

5.5.1 Annual correlation

Annual correlation matrixes for 2016, 2017 and 2018 years have been calculated to prove the hypothesis that the annual correlation remains between variables regardless of the concrete year. Table 5.13 shows the Pearson correlation matrix for 2018.

Firstly, we can say wind speed is linearly correlated with Wind direction (0.3782), sig. waves height (0.4425) and Avg. waves period (-0.4823). The correlation with Gust speed is not revealing since GST and WSPD can be considered the same variable keeping a linear correlation of 0.9951.

Secondly, for significant waves height we observe a little correlation temperature and pressure features which could be considered for select these variables as predictors for WVHT prediction models.

Wind direction is not highly correlated with any feature. It is correlated with wind speed (0.3782), Avg. waves period (-0.2850), Dominant waves period (-0.1214) and pressure (-0.1360).Finally, waves direction feature keeps a linear correlation with wind speed (0.1473), significant waves height (0.3576), Dominant waves period (-0.4391) and Waves surface temperature (-0.2695).

	MM	DD	hh	WDIR	WSPD	GST	WVHT	DPD	APD	MWD	PRES	ATMP	WTMP
MM	1	0.0079	-0.0040	0.0198	0.0046	0.0070	-0.0434	-0.0255	-0.0725	-0.1357	-0.2633	0.5597	0.6003
DD	0.0079	1	0.0024	0.0771	0.0626	0.0693	0.1032	0.1041	0.0319	0.0116	0.1301	-0.0538	-0.0053
hh	-0.0040	0.0024	1	-0.0718	-0.0529	-0.0539	-0.0131	0.0108	0.0444	-0.0170	0.0642	-0.0387	0.0127
WDIR	0.0198	0.0771	-0.0718	1	0.3782	0.3762	0.0949	-0.1214	-0.2850	0.0075	-0.1360	0.0032	-0.0272
WSPD	0.0046	0.0626	-0.0529	0.3782	1	0.9951	0.4425	-0.2281	-0.4823	0.1473	-0.0972	-0.1025	-0.1894
GST	0.0070	0.0693	-0.0539	0.3762	0.9951	1	0.4658	-0.2264	-0.4662	0.1520	-0.1042	-0.1142	-0.1919
WVHT	-0.0434	0.1032	-0.0131	0.0949	0.4425	0.4658	1	0.0584	0.3284	0.3576	0.1459	-0.2296	-0.2464
DPD	-0.0255	0.1041	0.0108	-0.1214	-0.2281	-0.2264	0.0584	1	0.5034	-0.4391	0.3204	0.0884	0.1558
APD	-0.0725	0.0319	0.0444	-0.2850	-0.4823	-0.4662	0.3284	0.5034	1	0.0340	0.3454	-0.0114	0.0495
MWD	-0.1357	0.0116	-0.0170	0.0075	0.1473	0.1520	0.3576	-0.4391	0.0340	1	0.0616	-0.2105	-0.2695
PRES	-0.2633	0.1301	0.0642	-0.1360	-0.0972	-0.1042	0.1459	0.3204	0.3454	0.0616	1	-0.1925	-0.1327
ATMP	0.5597	-0.0538	-0.0387	0.0032	-0.1025	-0.1142	-0.2296	0.0884	-0.0114	-0.2105	-0.1925	1	0.8448
WTMP	0.6003	-0.0053	0.0127	-0.0272	-0.1894	-0.1919	-0.2464	0.1558	0.0495	-0.2695	-0.1327	0.8448	1

Table 5.13. *Anuual Pearson correlation coefficients matrix for 2018*

Chapter 6

Feature selection and extraction

In machine learning, **feature selection** also known as variable subset selection is the process of selecting a subset of relevant predictors from all existing features for the modelling.

The main reasons for applying feature selection are the reduction of the algorithm training time, enhanced generalization which prevents models from over-fitting improving their performance and the simplification of the data sets for easier understanding. The premise to use feature selection is that data contains some features that are either redundant or irrelevant [77]. A redundant variable between predictors is one variable that is highly correlated with one or more other predictors. On the other side, a predictor is considered irrelevant if it is poorly correlated or uncorrelated with the predictive feature.

Feature selection methods are classified in

(1) **Filter based**. They use a proxy measure like variance or correlation with the predictive variable, to evaluate the importance of predictors and score a feature subset [77]. After selecting important features, we can train any model using them.

(2) **Wrapper based**. Wrappers methods consist of testing all possible combinations of features for training the selected algorithm and keep the features subset which minimizes the error rate of the model. This is computationally expensive and unfeasible for the computer on which the project is being carried out. Moreover this is dependent on the selected training algorithm while Filter based is not.

(3) **Embedded**. Embedded methods are a group of techniques that perform feature selection as part of the model learning process [77].

We will use the filter method ***F*-test** which calculates a univariate feature ranking (based on co-variance measure) where all predictors are ordered by importance in the data set and they have an associated weight. This method has been chosen for its low computational running cost and effectiveness for regression problems with continuous variables as predictors [78].

Complementing the feature selection, **feature extraction** or data combination consists of introducing new features in data sets generated from the transformation and/or combination of some existing variables. Data combination techniques allow to include new information in data set extracted from some variables combined by mathematical transformations. At the same time, a new variable can replace some existing previous features, thus reducing the dimension of the data set without losing significant information or even increasing it, so we may obtain better model accuracy.

We will create manually some new variables as well as we will apply the Principal Component Analysis (PCA) method in the case that model training seems to be too slow.

6.1 Features extraction: MIS

For Misalignment forecasting, we will combine Wind direction (WDIR) and Waves direction (MWD) to obtain the new feature that represents the wave and wind misalignment: **MIS**.

To obtain the misalignment we first subtract the angle of the wind and waves direction. Then we calculate the rest by dividing the resultant angle by 180º. This can be made by the function *mod* in Matlab, as shown in the code 6.1.

```
for c = 1:length(data10_Real_clean.WDIR)
    data10_Real_clean.MIS(c) = data10_Real_clean.WDIR(c) - data10_Real_clean.MWD(c);
    if data10_Real_clean.MIS(c) > 180
        data10_Real_clean.MIS(c) = mod(data10_Real_clean.MIS(c),-180);
    elseif data10_Real_clean.MIS(c) < -180
        data10_Real_clean.MIS(c) = mod(data10_Real_clean.MIS(c),180);
    end
end
```

Figure 6.1. *Code implemented to obtain the new feature MIS (wind - waves misalignment) in º*

By calculating the mod, we always obtain the smaller angle between the wind and waves direction as seen in a example in the figure 6.2.

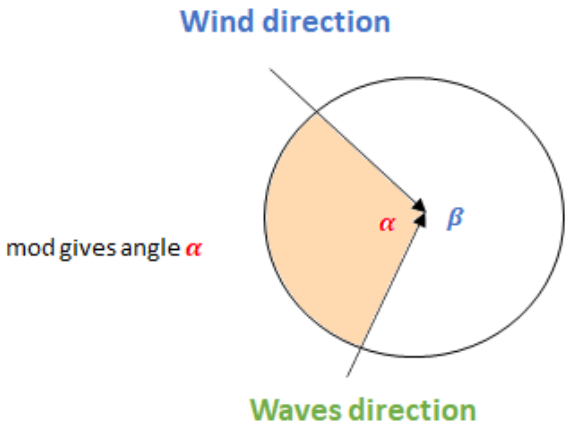


Figure 6.2. *Angle between wind and waves taken as MIS feature diagramm*

6.2 Data set

6.2.1 *data10_Real_clean*

Row data loaded at the start of the project from 2010 to 2020 (Real Time) have been saved after cleaning process in *data10_Real_clean* data set.

Then, we have selected the following subsets of data:

- *data2018*: data from 2018 used for training models.
- *data10_18*: data from 2010 to 2018 used for training models with a huge amount of data.
- *data2019*: data from 2019 applied to test additionally the models with the best performance.
- *dataRealTime*: data from 2020 applied also to test the best trained models

6.3 Features selection

We have applied *F-Test* into data for each predictive feature to find out which features have a higher importance score. We also have considered correlation coefficients obtained

in the previous chapter to select variables for training models. Every feature subset selection shares that do not contain minutes variable, since it has always the same value so *mm* feature has been deleted from *data10_Real_clean*.

6.3.1 F-test for wind speed forecasting

On the one hand, the Gust speed feature, which is a component of the wind speed, can be considered the same variable as speed so it is excluded from potential predictors to calculate their F-score. Different features subset will be taken for models training according to the F-score test results as well as common sense when thinking of the physic nature of wind speed and its relationship with the rest of the phenomena.

Figure 6.3 shows the F-test scores[1]. Wind direction and Significant Wave height (and therefore Average wave period and Dominant wave period) seem to be important features while the year has 0 importance because we are considering just data from 2018 to calculate the F-score.

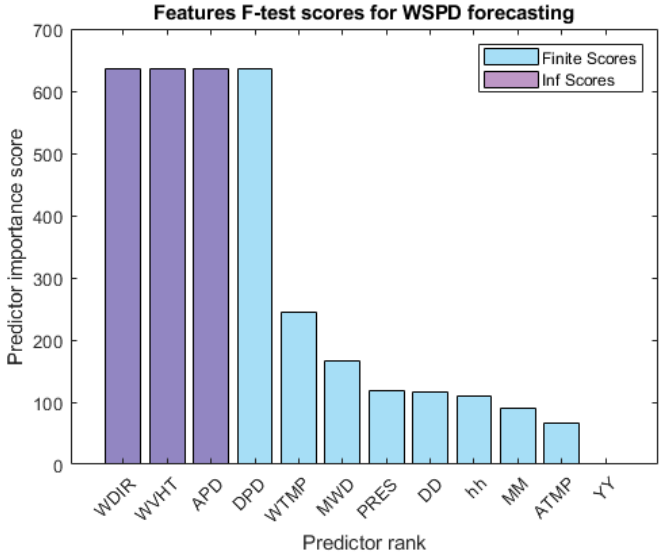


Figure 6.3. *F-test scores bars plot for features to predict wind speed. Infinite and finite values are distinguished by the bar color*

According to ρ-value interpretation, if $\rho \leq 0.05$ we have strong evidence about the null hypothesis. It is equivalent to say that a $score \leq -\log(0,05) \simeq 1,3$ indicates that the feature is insignificant. In this case, every feature except YY have a score higher than the threshold described.

Then we look at the Pearson correlation matrix for 2018 to check that no pair of indepen-

[1]The infinite values have been represented in bars plot as the maximum score between features. Infinite measures are given by F-test when ρ-value is less than $esp(0) = 4.4907 \cdot 10^{-324}$ so $log(\rho)$ would be almost infinite.

dent variables are correlated with a coefficient higher than 0,8. In this case, it would rule out one of them to avoid redundant information in data sets. We see that ATMP and WTM have a coefficient of 0,8448 which makes sense since both features are temperatures measures so we will include just one of them as a predictor.

On the other hand, Significant wave height (WVHT), Dominant wave period (DPD), Average Wave period(APD) and Waves direction will not be included as predictors for wind speed forecasting because physically these variables can be considered as dependent from wind state variables.

Figure 6.4 summarize initial data subset for wind speed forecasting, according to the above justification.

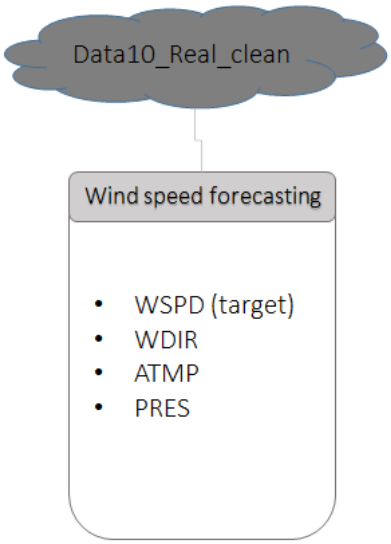


Figure 6.4. *Potential predictors for WSPD forecasting*

Date features are necessary to keep the data in order on the timeline for later combination of some values of different rows for time series forecasting as we will explain in the data combination section.

6.3.2 F-test for significant waves height prediction

As we have done for WSPD, for selecting features to Wave Significant height prediction, we apply first the same F-scored and we analyse correlation between WVHT and rest of features as well as each pair of potential predictors correlation.

Before F-test application, YY variable has been scrapped and also GST because it is considered the same variable as WSPD. Figure 6.7 shows that Wind speed and Waves direction are highly significant for WVHT prediction, as we have imagined because wind

speed is the main source that generates waves so probably their values tendency would be strongly related.

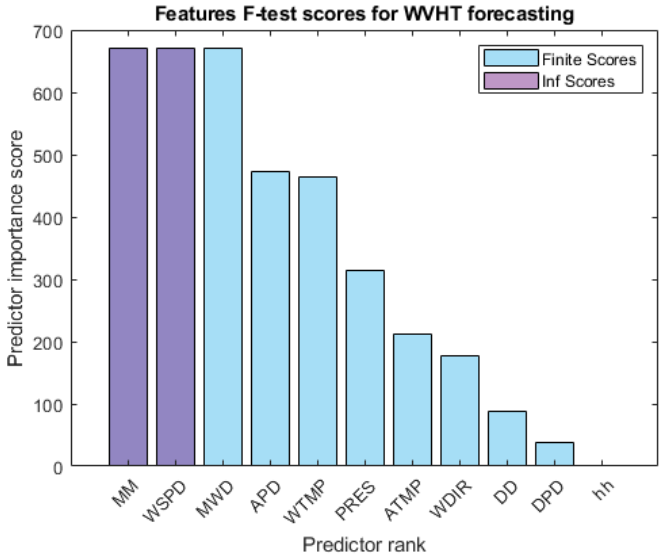


Figure 6.5. *F-test scores bars plot for features to predict wind speed. Infinite and finite values are distinguished by the bar color*

Looking at the correlation matrix, as stated above, ATMP and WTMP are highly correlated with a Pearson coefficient of 0.8448. This could be redundant so we select WTMP instead of ATMP in this case because of its higher F-score and physical intuition.

For WVHT characterization in the frequency domain, we will consider WSPD, WDIR and MWD as main potential predictors. Physically the wave height is determined by the current wind speed but it can also be affected by previous waves. This intuition leads us to consider directional variables too. WTMP and PRES variables are left in the data set for possible subset selection for training.

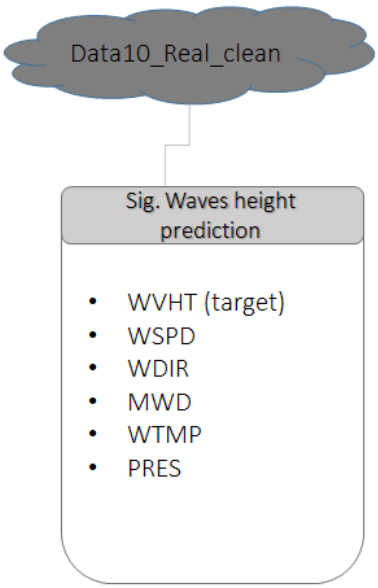


Figure 6.6. *Potential predictors for WVHT prediction*

6.3.3 F-test for misalignment forecasting

F-test scores calculated for the misalignment feature shows that Dominant waves period, Pressure, Significant waves height, Wind speed and Waves surface temperature features are the most significant.

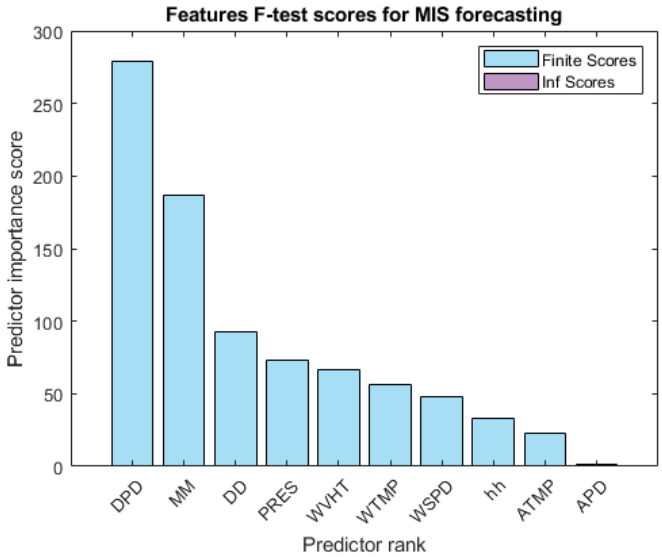


Figure 6.7. *F-test scores bars plot for features to forecast misalignment. Infinite and finite values are distinguished by the bar color*

For misalignment time series forecasting we select as potential predictors in addition to

MIS: WVHT, WSPD, PRES and WTMP as seen in figure 6.8. In spite of having a high F-test score, DPD is physically dependent of the waves height so we decide to exclude it as potential predictor.

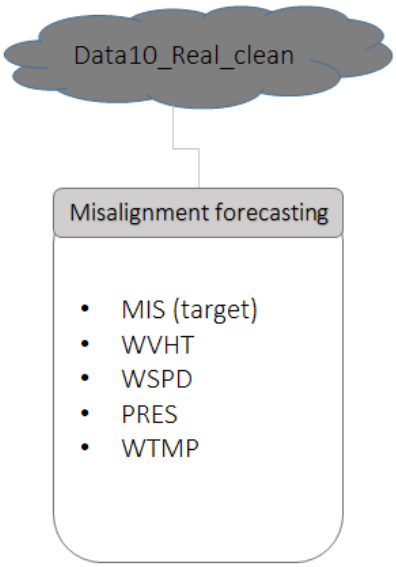


Figure 6.8. *Potential predictors for MIS forecasting*

6.4 Data combination

Time series forecasting models receive as input a set of predictors based on a temporal window of consecutive samples (rows) of the data [79]. Thus for applying ML techniques we must combine data rows to create new variables in each row corresponding to the predictive variable and other predictors dated some hours before.

For instance, if we want to forecast next hour wind speed (t) by considering as input wind speed in the previous 3 hour (t-1, t-2 and t-3), we would have the temporal window showed in figure 6.9.

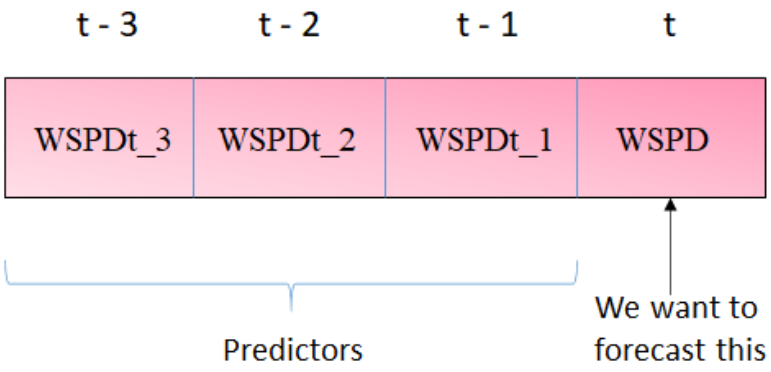


Figure 6.9. *Temporal window to predict next hour WSPD with WSPD in the previous 3 hours as predictors*

To get a data set with features from the temporal window 6.9 in each row we do the following data combination in the data set for each distinct pack of 4 contiguous rows:

Original Data set

	1 YY	2 MM	3 DD	4 hh	5 mm	6 WDIR	7 WSPD
1	2018	1	1	0	50	36	0.8000
2	2018	1	1	1	50	17	0.8000
3	2018	1	1	2	50	354	0.5000
4	2018	1	1	3	50	23	1.2000
5	2018	1	1	4	50	11	1.1000
6	2018	1	1	5	50	325	1.1000
7	2018	1	1	6	50	299	1.5000
8	2018	1	1	7	50	311	2.6000
9	2018	1	1	8	50	329	3
10	2018	1	1	9	50	338	2.6000
11	2018	1	1	10	50	358	3.3000
12	2018	1	1	11	50	350	3.6000

Data set with predictors in the timeline

7 WSPD	19 WSPDt_1	18 WSPDt_2	17 WSPDt_3
1.2000	0.5000	0.8000	0.8000
1.1000	1.2000	0.5000	0.8000
1.1000	1.1000	1.2000	0.5000
1.5000	1.1000	1.1000	1.2000
2.6000	1.5000	1.1000	1.1000
3	2.6000	1.5000	1.1000
2.6000	3	2.6000	1.5000
3.3000	2.6000	3	2.6000
3.6000	3.3000	2.6000	3
4	3.6000	3.3000	2.6000
	4	3.6000	3.3000
		4	3.6000
			4

Figure 6.10. *Data rows combination to relate WSPD in time t (predictive feature) with WSPD in t-1, t-2 and t-3 as predictors*

The data combination represented in the diagram 6.10 has been done to get data sets for wind speed and misalignment forecasting changing the time window longitude. All data sets are saved in the repository. Files to generate the data sets from original ones are also available in the repository with names like "1hourbefore$_W SPD.m$".

In the case of neural networks (NAR and NARX), the *Neural Net Time series* toolbox has been used where we establish a delay (how many hours before forecasting moment t to consider predictors in the time window) and it makes automatically the data combination.

Predictors (values in the moment t) and targets are given to the toolbox in separated arrays.

Part II

Modeling results

Chapter 7

Wind speed forecasting: algorithms application

In this part of the work, we proceed to train our models for Wind speed forecasting in the time domain. We applied the following Machine Learning algorithms:

- Linear Regression (LR)
- Support Vector Machines for Regression (SVR)
- Gaussian Process Regression (GPR)
- Nonlinear Autoregressive Neural Network (NAR)
- Nonlinear Autoregressive with External input Neural Network (NARX)

Models are trained **to forecast the next hour's wind speed** by trying different combinations of features of data set *data2018*. If needed, more data from other years is added to the training set during optimization phase.

For regression algorithms application, in the Regression Learner app, the holdout validation scheme is configured with a held-out data subset of 20%. That means that 20% of 8581 rows are used for validating the models with unseen observations.

In the case of artificial neural networks, the data set has been split in training (70%), validation (15%) and testing (15%) sets.

Finally, the best models have been additionally tested with data from 2019 and 2020 ("Real Time") as a demonstration of its performance. The performance is measured with the RMSE error (units are $\frac{m}{s}$).

7.1 LR

Linear regression has been the first algorithm we applied because of its simplicity and easy understanding. Model variants: linear, interactions linear, robust linear and stepwise linear have been performed in next hour wind speed forecasting giving all models except Robust variant equal results as seen in table 7.1. Thus we select, for instance, the Interactions linear regression model to train with wind speed registers from 1 hour to 6 hours before as predictors, obtaining the results summarized in 7.2.

wind speed input	Model type	RMSE	MAE	R-squared	Success rate
1 h-before	Linear	1.0546	0.7837	0.89	93.15%
	Interactions linear	1.0546	0.7837	0.89	93.15 %
	Robust linear	1.0558	0.7819	0.89	93.14 %
	Stepwise linear	1.0546	0.7837	0.89	93.15 %

Table 7.1. *Validation results of linear regression models trained with wind speed an hour before as predictor for next hour wind speed forecasting*

Model name	wind speed inputs	RMSE	MAE	R-squared	Success rate
wind_lr1	1 h-before	1.0546	0.7837	0.88	93.15 %
wind_lr2	2 h-before	1.0529	0.7837	0.89	93.16 %
wind_lr3	3 h-before	1.0539	0.7852	0.89	93.16 %
wind_lr4	4 h-before	1.0578	0.7872	0.89	93.13 %
wind_lr5	5 h-before	1.0624	0.7868	0.89	93.10 %
wind_lr6	6 h-before	1.0639	0.7886	0.89	93.09 %

Table 7.2. *Validation results of linear Regression models trained with wind speed from 1-hour to 6-hours before as predictors for next hour wind speed forecasting*

The *wind_lr2* model with two last hours wind speed as input shows the best cross validation error. Figures 7.1 shows its performance for the 20% hold-out validation data by plotting the true response (training targets) for the first 100 observations of validation set and the predicted output for these observations.

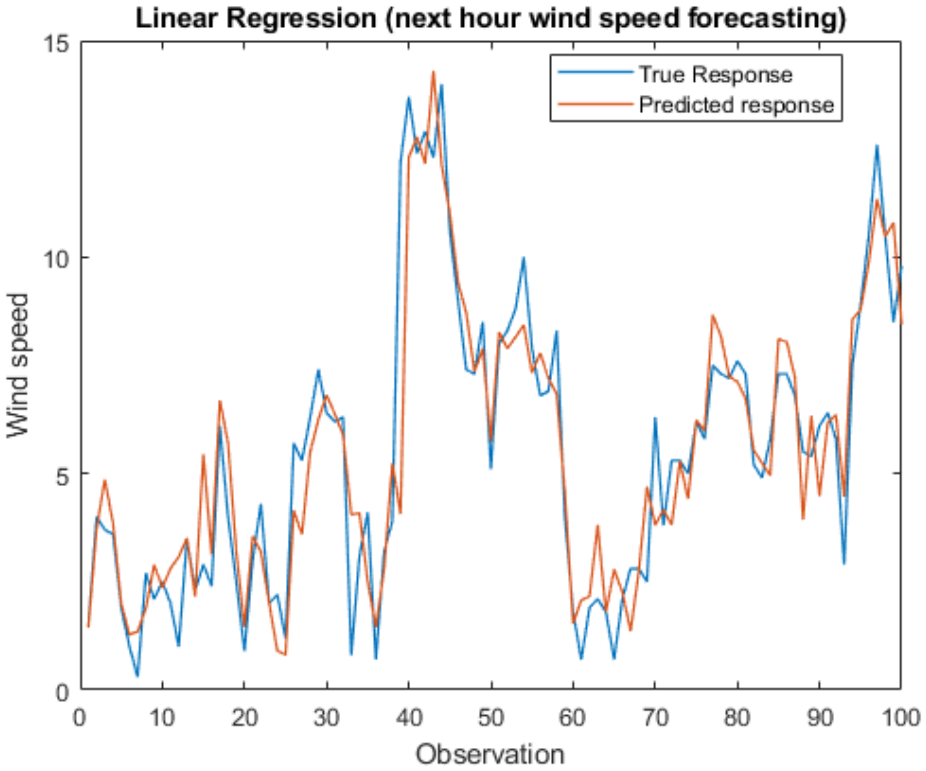


Figure 7.1. *Validation responses compared with predicted outputs by wind_lr2 model for 100 first validation observations subset of data from 2018*

We perform *wind_lr2* model to forecast wind speed with 2019 and Real time data rows, obtaining a RMSE of 1.1237 and 1.1723 respectively.

7.2 SVR

Support vector machines for regression model is firstly trained with data from 2018 and two last hours wind speed variables as predictors. Different configurations are tried obtaining the best results (listed in the table 7.3) with Gaussian kernel function, a box constraint of 9.78 and $\varepsilon = 0.066$.

Model name	inputs in time	predictors	RMSE	MAE	R-squared	Success rate
wind_svm1	2 h-before	WSPD	1.0359	0.7664	0.89	93.27 %

Table 7.3. *Validation results of Regressive SVM model for next hour wind speed forecasting with last two wind speed values as input*

To decide what to do to optimize the *wind_svm1* model, we first plot its learning curves

(figure 7.2) by training the model with different subsets of m training examples and calculating the training and validation RMSE for each m. The reference error represented as green discontinuous line as we explained is selected according to the range of errors found in the literature for models that predict the same variable as the one studied with the learning curves. This reference error is just a indicative little profitable value whose only purpose is be able to evaluate our model and decide whether to continue optimizing it or not.
As in literature the RMSE for wind speed forecasting with intelligent models vary between 1.650 and $0,7040$, we stablish the reference error to 0.9.

In figure 7.2 we appreciate that *wind_svm1* model suffers from high bias which means the model is underfitted. Thus try adding additional features (corresponding to other physic characteristics or more variables representing WSPD from hours in the past) as predictors could help to improve the RMSE.

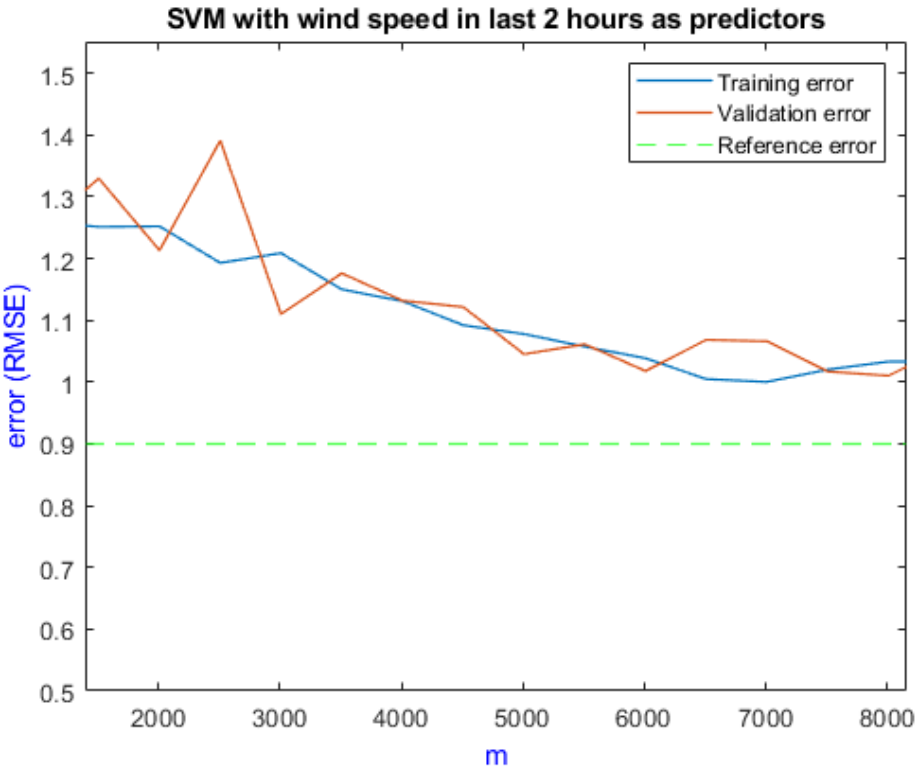


Figure 7.2. *Learning curves plot for wind_svm1 model*

Table 7.4 shows results given by the two best SVM optimized models trained:

- *wind_svm2* model with wind speed from 1 to 10 hours before the moment we want to forecast the speed as predictors, linear kernel function, a box constraint of 6.1141 and $\varepsilon = 0.1377$.
- *wind_svm3* model which receive as input wind speed from 1 to 14 hours before and configured with gaussian kernel function, a box constraint of 0.5056 and $\varepsilon = 0.0096$.

We reach the best RMSE equals to 0.9985 with *wind_svm3* model.

Model name	inputs in time	predictors	RMSE	MAE	R-squared	Success rate
wind_svm2	10 h-before	WSPD	1.0148	0.7654	0.90	93.41 %
wind_svm3	14 h-before	WSPD	0.9985	0.7574	0.89	93.52 %

Table 7.4. *Validation results of Regressive SVM models wind_svm2 and wind_svm3 for next hour wind speed forecasting*

The *wind_svm3* model gives a RMSE of 1.1462 for the year 2019 and 1.1722 for Real time wind speed forecasting. Response plot of *wind_svm3* for data from 2019 can be seen in figure 7.3

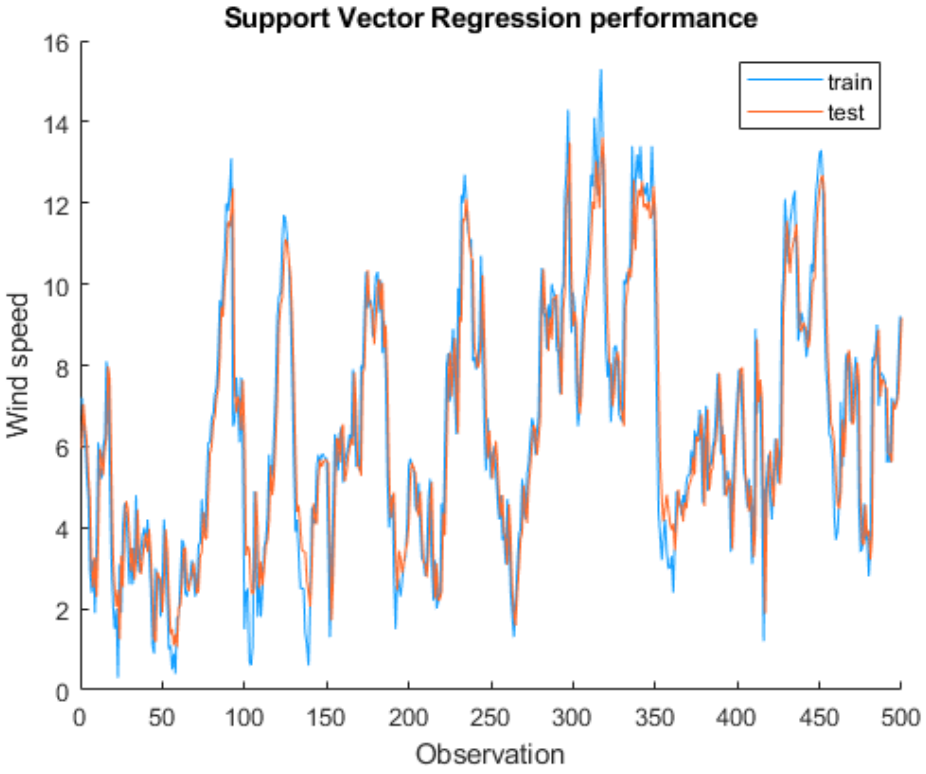


Figure 7.3. *Responses for data from 2019 compared with predicted outputs by wind_svm3 model for 500 first observations*

7.3 GPR

Table 7.5 includes the performance measures for the Gaussian Process Regression model configured with the isotropic matern 5/2 kernel function, a kernel scale of 0.0164 and

$\sigma = 0.0057$ as hyperparameters which gives us the best results between GPR configured models trained with data rows from 2018 and just one predictor: the current wind speed. Thus if t is the moment to forecast wind speed, the predictor will be wind speed one hour before (at $t - 1$ in time).

Model name	inputs in time	predictors	RMSE	MAE	R-squared	Success rate
wind_gpr1	1 h-before	WSPD	1.0424	0.7746	0.89	93.23 %

Table 7.5. *Validation results of wind_gpr1 model for next hour wind speed forecasting*

The wind speed predictions given by *wind_gpr1* model for 100 first observations of validation set are compared with true wind speed values in figure 7.4.

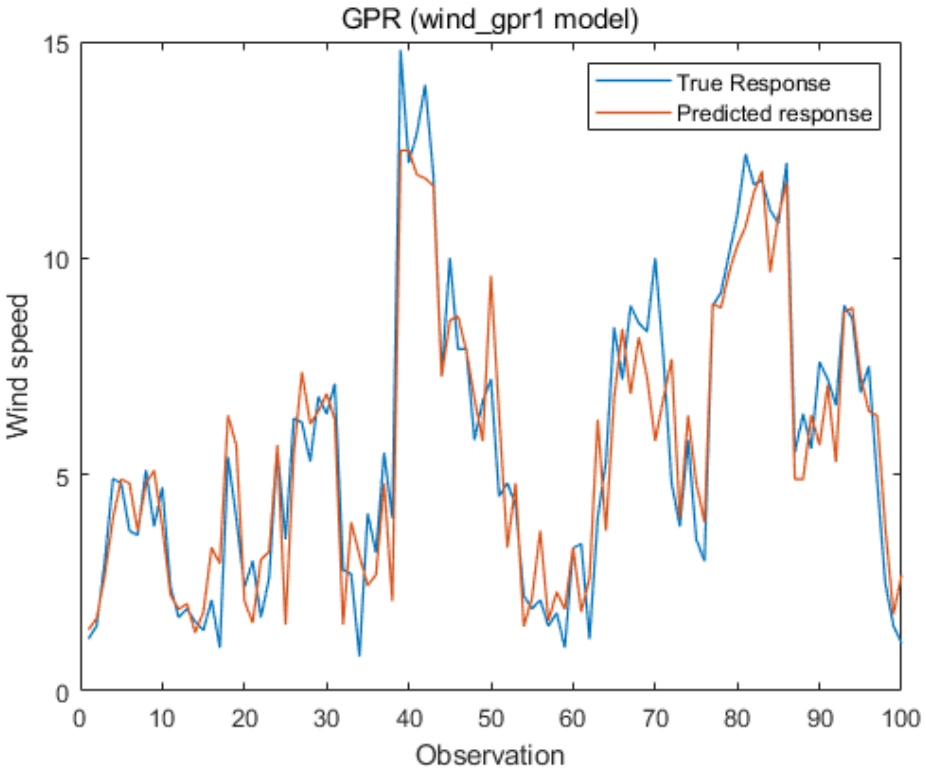


Figure 7.4. *Validation responses compared with predicted outputs by wind_gpr1 model for 100 first observations of 2018*

As seen in learning curves plot 7.5, predictions error of *wind_gpr1* is so close to the reference error selected by us which is a good indicative of the good performance of the model. Even so, we tried to train some more GPR models, highlighting the results for *wind_gpr2* and *wind_gpr3* models (see table 7.6). In this case, models are trained with Wind speed (WSPD), Air temperature (ATMP) and Pressure (PRES) in $t - 1$ (*wind_gpr2*) and $t - 1$, $t - 2$ (*wind_gpr3*) as predictors.

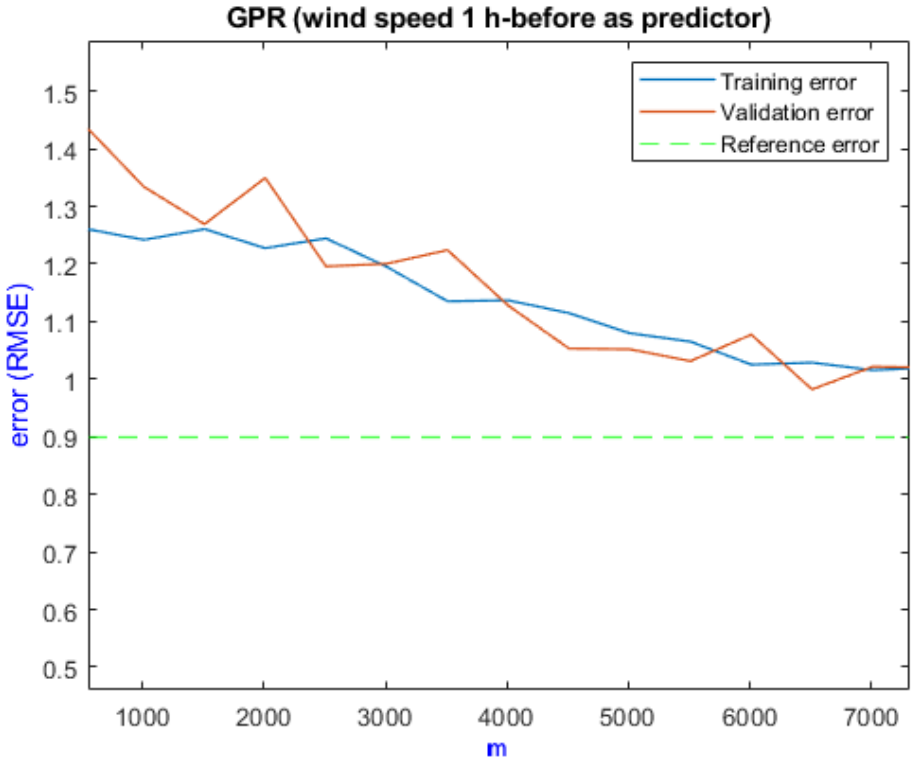


Figure 7.5. *Learning curves plot for wind_gpr1 model*

Model name	inputs in time	predictors	RMSE	MAE	R-squared
wind_gpr2	1 h-before	WSPD, ATMP,PRES	1.0258	0.7630	0.89
wind_gpr3	2 h-before	WSPD,ATMP,PRES	1.0774	0.7731	0.88

Table 7.6. *Validation results of wind_gpr2 and wind_gpr3 models for next hour wind speed forecasting*

Model name	Success rate
wind_gpr2	93.34 %
wind_gpr3	93.00 %

Table 7.7. *Success rate of wind_gpr2 and wind_gpr3 models*

Finally, RMSE for 2019 and Real Time data has been calculated as seen in table 7.8 for the three previous GPR models to observe which one generalize better for new data forecasting.

Model name	RMSE (2019)	RMSE (Real Time)	% 2019	% 2020
wind_gpr1	1.1274	1.1721	92.67 %	92.38 %
wind_gpr2	1.1344	1.1670	92.63 %	92.42 %
wind_gpr3	1.1102	1.1538	92.79 %	92.50 %

Table 7.8. *RMSE and success rate obtained for 2019 and 2020 (Real time) next hour wind speed forecasting with wind_gpr1, wind_gpr2 and wind_gpr3 models*

Although *wind_gpr3* model seems to generalize a bit better than other two, it also receives 6 input parameters as predictors which could be more computationally expensive. Thus, we propose *wind_gpr2* as a possible good choice to real wind speed forecasting.

7.4 NAR

As explained in chapter 3, Nonlinear Autoregressive Neural Networks predict series $y(t)$ given as input the same variable with a delay d established, i.e. the mathematical expression of the model follows the following form

$$y(t) = f(y(t-1), y(t-2), ..., y(t-d))$$

As there is not a concrete criteria to select the number of neurons or delay for the neural network, we start with a network of 15 neurons at the hidden layer and a delay of 6 (see figure 7.6). Generally, NAR are used with just one hidden layer so all NAR models trained here will have one hidden layer.

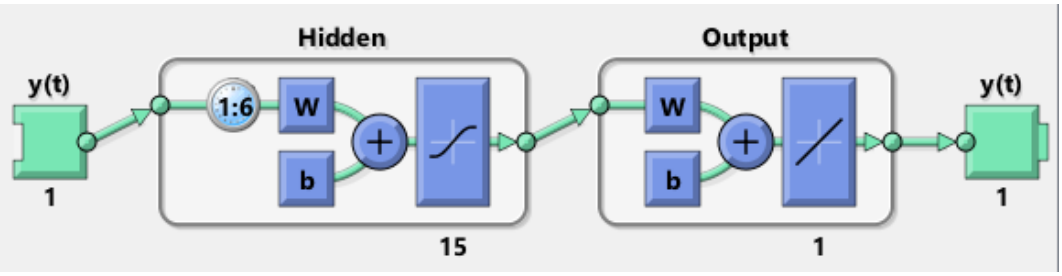


Figure 7.6. *wind_nar1 nonlinear autoregressive neural network structure diagram*

Table 7.9 shows the results obtained with the *wind_nar1* network.

Model name	Nº neurons	Nº delays	RMSE			R-squared			Success rate
			Train	Valid.	Test	Train	Valid.	Test	
wind_nar1	15	6	1.0235	1.0581	1.0871	0.94	0.93	0.94	92.94 %

Table 7.9. *Results for wind_nar1 model trained with data from 2018 for next hour wind speed forecasting*

As seen, testing and validation RMSE are a bit higher than training RMSE. We decide to plot the learning curves (see figure 7.7) to try to optimize the model .

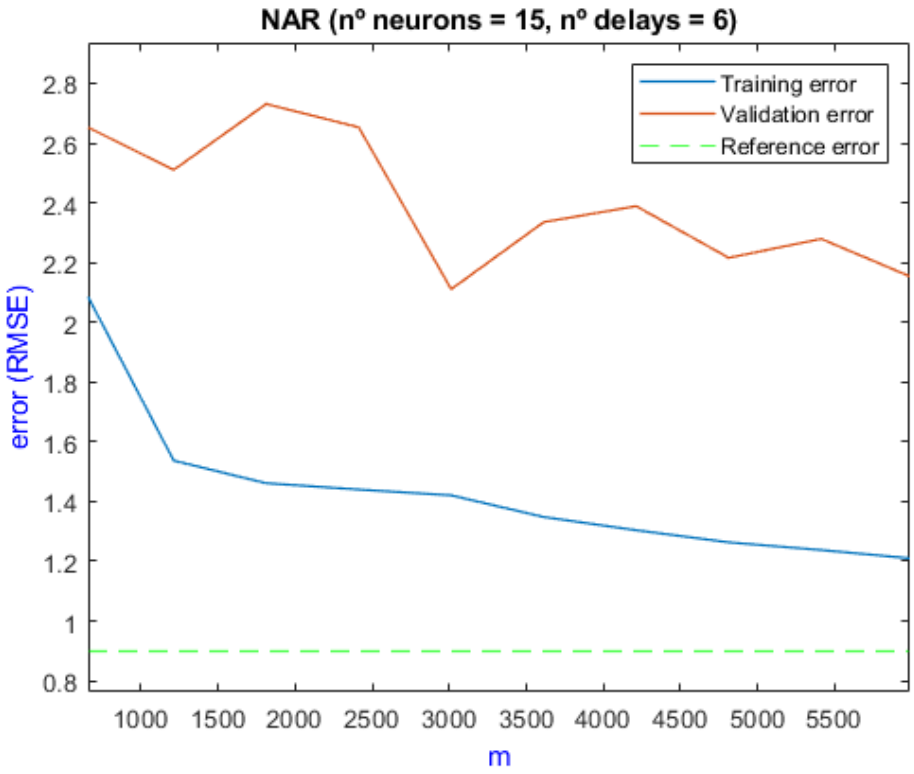


Figure 7.7. *Learning curves for wind_nar1 model*

Evaluating the distance between training and validation error curves which is higher than in other previous learning curves plots, we could say the *wind_nar1* model is suffering from high variance so it may help select a simpler structure for the neural network or use more training examples from other years. Some of the best NAR models obtained during the optimization phase are presented together with RMSE and R squared on table 7.10.

Model name	Input data	Nº neurons	Nº delays	RMSE			R-squared		
				Train	Valid.	Test	Train	Valid.	Test
wind_nar2	2016 to 2018	15	6	1.0303	1.0628	1.0389	0.95	0.94	0.94
wind_nar3	2018	15	1	1.0537	1.0821	**0.9762**	0.94	0.93	0.94

Table 7.10. *Results of wind_nar2 and wind_nar3 models training for next hour wind speed forecasting*

Model name	Success rate
wind_nar2	93.25 %
wind_nar3	**93.66 %**

Table 7.11. *Testing success rate for wind_nar2 and wind_nar3 models*

The Learning curves plot for the *wind_nar2* model (figure 7.8) shows an improvement in prediction with respect to *wind_nar1* learning curves showed in figure 7.7.

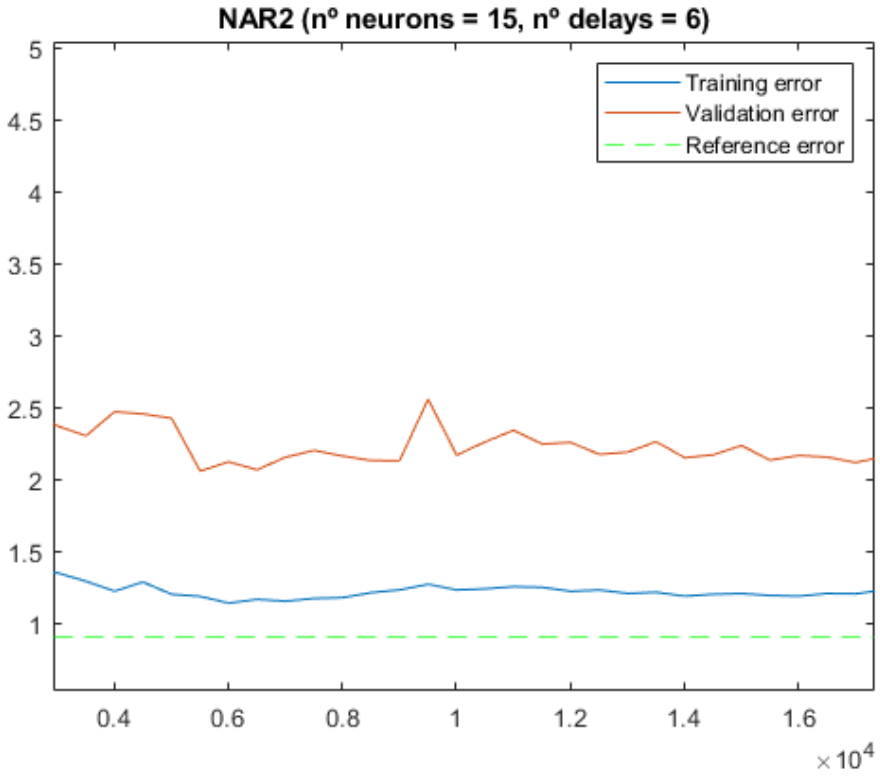


Figure 7.8. *Learning curves for wind_nar2 model*

Despite consider only the wind speed of the previous hour as predictor, the *wind_nar3* model gives us the best testing RMSE : 0.9762 (the RMSE of wind speed forecasting for a 20% unseen data). Thus the model is applied to forecast WSPD of 2019 giving a RMSE of 1.1496 and a RMSE for real time data of 1.1713.

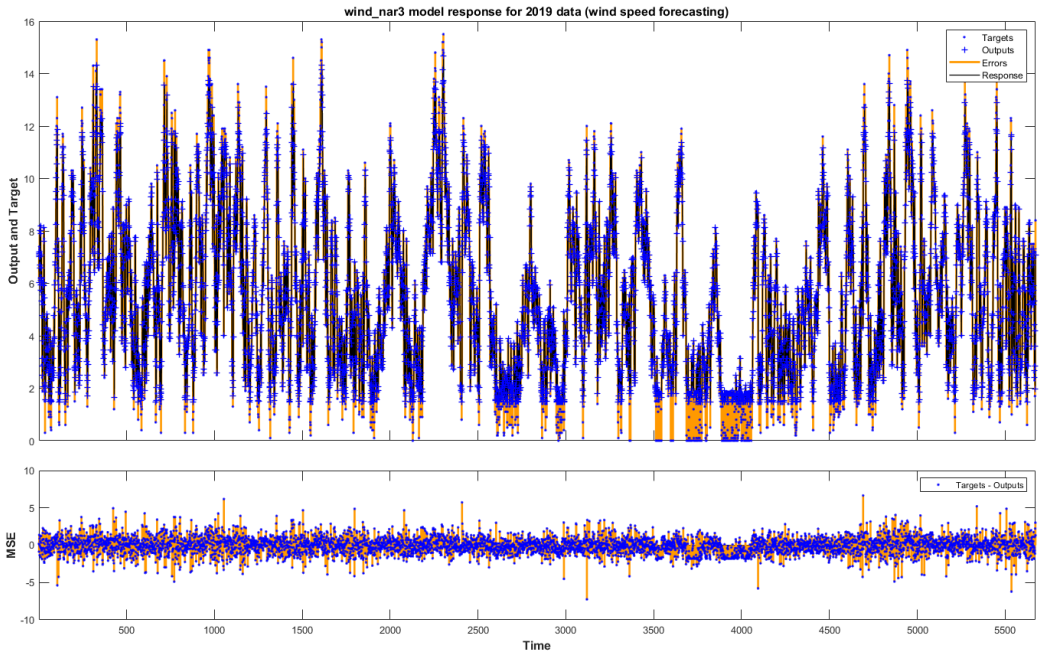


Figure 7.9. *Forecasting response of wind_nar3 model for data of 2019*

7.5 NARX

Nonlinear Autoregressive with External Input (NARX) neural networks differ from NAR networks in that they receive as input in addition to the variable to be predicted, past values of other variables. Thus the mathematical expression of the model would be for a given delay d and inputs variables x and y.

$$y(t) = f(x(t-1), ..., x(t-d), y(t-1), ..., y(t-d))$$

Guided by results obtained for NAR models, we started trying simple NARX architecture and training them with data from 2018 as we thought they would give us lower RMSE. Table 7.12 shows best three trained models, highlighting the *wind_narx2* whose architecture is represented in figure 7.10. All three models receive as input features: WSPD, ATMP, PRES and WDIR of previous d hours (remembering d is the stablished delay).

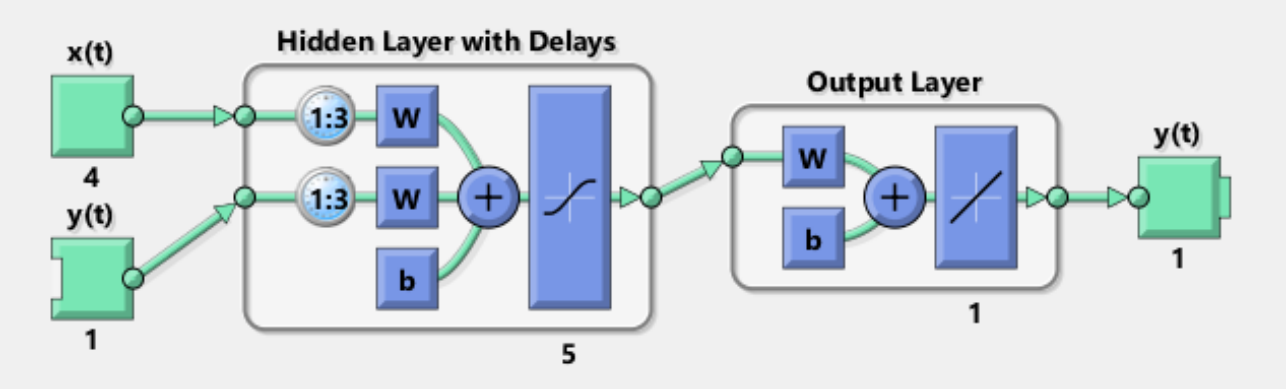


Figure 7.10. *wind_narx2 nonlinear autoregressive with external input neural network architecture diagram*

Model	Nº	Nº	RMSE			R-squared		
name	**neurons**	**delays**	**Train**	**Valid.**	**Test**	**Train**	**Valid**	**Test**
wind_narx1	10	1	1.0179	1.0889	1.0066	0.95	0.94	0.95
wind_narx2	5	3	0.9993	1.0162	0.9988	0.95	0.95	0.95
wind_narx3	9	6	0.9868	1.0630	0.9893	0.95	0.94	0.95

Table 7.12. *Results of wind_narx1, wind_narx2 and wind_narx3 models for next hour wind speed forecasting*

Model name	Success rate
wind_narx1	93.46 %
wind_narx2	93.51 %
wind_narx3	93.58 %

Table 7.13. *Testing success rate for wind_narx1, wind_narx2 and wind_narx3 models for next hour wind speed forecasting*

To compare the three models we use them to forecast the wind speed of 2019 and 2020.

Model name	RMSE (2019)	RMSE (Real Time)	% (2019)	% (2020)
wind_narx1	1.1517	1.1828	92.52 %	92.32 %
wind_narx2	1.1124	1.1340	92.78 %	92.63 %
wind_narx3	1.1309	1.1758	92.66 %	92.36 %

Table 7.14. *RMSE and success rate of three NARX best trained models for 2019 and 2020 (Real time) next hour wind speed forecasting*

Results summarized in table 7.14 shows that *wind_narx2* is a bit better than others by forecasting wind speed with new data as input as time as the structure and computationally cost of execute the network is lower. Thus *wind_narx2* could be choosen between the three models. Figure 7.11 shows *wind_narx2* forecasting response for data of 2019.

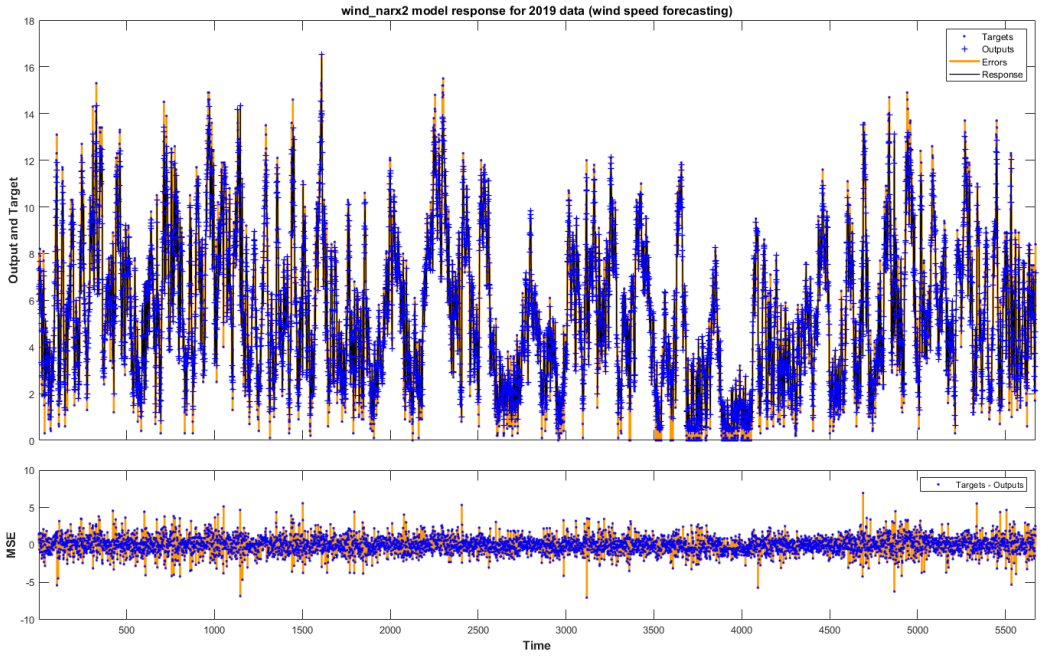


Figure 7.11. *Forecasting response of wind_narx2 model for data of 2019*

7.6 Models comparison

The best way to compare the peformance of models obtained by applicating the different algorithms for the next hour wind speed forecasting is to put together into a table the best model for each ML technique together with the validation RMSE and also the RMSE given for 2019 and 2020 data forecasting. Validation RMSE for neural networks correspond to

the test RMSE due to CV scheme applied to NN is training-validation-testing data set partition and testing RMSE is more significative than validation one.

Algorithm	Model Name	RMSE (validation)	RMSE (2019)	RMSE (2020)	Success rate (validation)
LR	wind_lr2	1.0529	1.1237	1.1723	93.16 %
SVR	wind_svm3	0.9985	1.1462	1.1722	93.52 %
GPR	wind_gpr2	1.0285	1.1344	1.1670	93.32 %
NAR	wind_nar3	0.9762	1.1496	1.1713	93.66 %
NARX	**wind_narx2**	**0.9988**	**1.1124**	**1.1340**	**93.51 %**

Table 7.15. *Best models for WSPD forecasting obtained by each ML algorithm together with validation RMSE, RMSE for 2019 and Real Time (2020) data and the validation success rate.*

Table 7.15 list the most optimized model for each ML technique applied. Even when *wind_nar3* gives the best validation sucess rate (93.66%), it generalize worst for 2019 and 2020 data forecasting as we observe in RMSE for 2019 and 2020 columns. Thus, the algorithm that performs the best with a validation RMSE of 0.9988 (second lowest) and the lowest RMSE for 2019 (1.1124) and 2020 (1.1340) is **Nonlinear Autoregressive with external input neural network (NARX)** algorithm and **best configuration** found is that applied to train the ***wind_narx2*** model:

- **Data set**: data from 2018
- **Predictors**: WSPD, ATMP, PRES, WDIR
- **Nº neurons in the hidden layer**: 5
- **Nº input delays**: 3 hours before forecasting moment (t)

wind_narx2 model has a **success rate of 93.51%** so we conclude it has a good performance. Figure 7.12 shows the predicted against true first 200 output values from 2020, for wind_narx2 model.

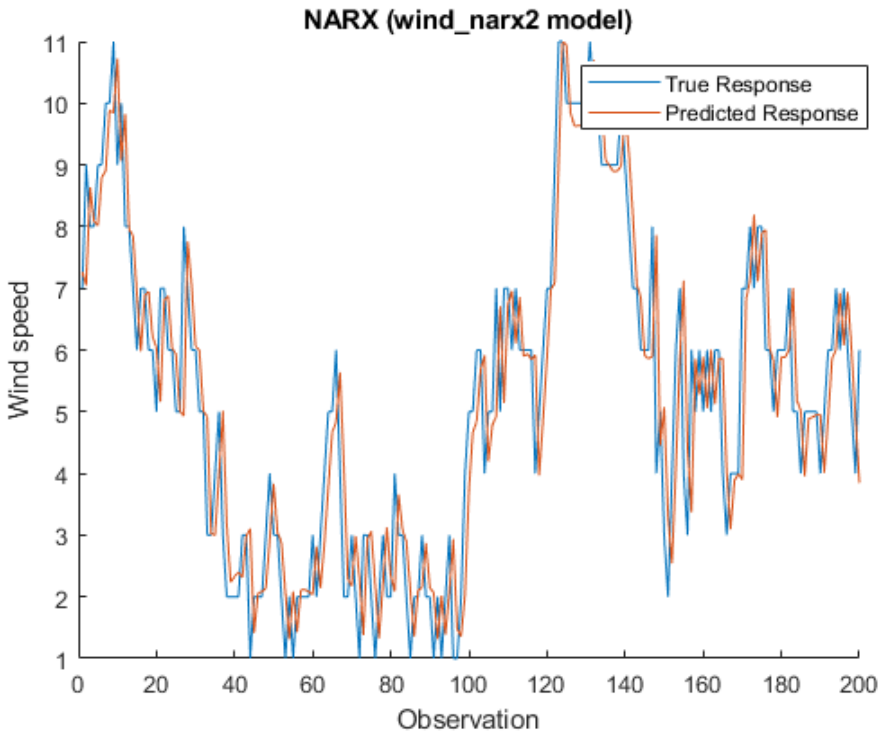


Figure 7.12. *Predicted against true responses for wind_narx2 model for 200 first observations of data from 2020*

Chapter 8

Significant waves height prediction: algorithms application

Waves modelling in the frequency domain consists of characterizing or predicting the waves state, based on known wind and weather measurements. Concretely, we will predict the significant height of waves given wind features as well as waves direction, temperature value or pressure as models input.

Machine learning algorithms applied are:

- Linear Regression (LR)
- Support Vector Machines for Regression (SVR)
- Gaussian process regression (GPR)
- Feed-forward backprop ANN

The cross-validation schemes used for several models have been: cross-validation with 5 sub-folders and holdout validation scheme configured with a held-out data subset of 20%.

On the other hand, for artificial neural networks training, the data set used has been split in training (70%), validation (15%) and testing (15%).

RMSE error has been used to measure models' performance whose units are meters (m).

8.1 LR

Linear Regression is used considering the great causality and high relation of the waves height attributed to the wind speed primarily. Moreover, significant waves height keeps

a relation with wind direction and waves direction, so these features may be interesting to include as model inputs.

Firstly, a LR model has been trained with WSPD feature as input, using the 5-fold cross validation scheme to partition the data from 2018 (training data set). Table 8.1 shows the results of training this first model: *waves_lr1*.

Model name	predictors	RMSE	MAE	R-squared	Success rate
waves_lr1	WSPD	0.6747	0.5149	0.20	87.71 %

Table 8.1. *Validation results of LR waves_lr1 model*

The validation RMSE obtained for *waves_lr1* model is a bit bad compared with most RMSE for significant waves height prediction found in literature (≈ 0.25) as well as R-squeared indicate that just 20% of data variation is explained by the predictor selected (WSPD).RMSE for 2019 data prediction given by *waves_lr1* is 0.8209 and 0.6223 for Real time data. True responses and predictions made by the same model on first 500 observation for data of 2019 can be seen in figure 8.1.

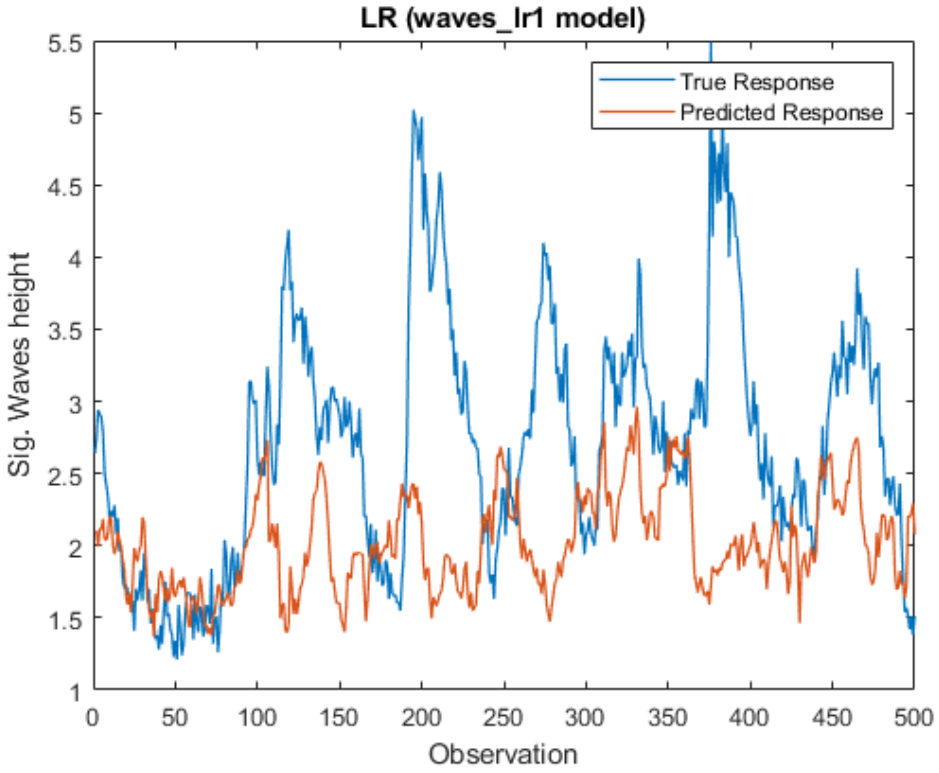


Figure 8.1. *Forecasting response of waves_lr1 model for data of 2019*

Optimization phase in this case is poorly significant because Linear regression model is too simple to explain or describe the significant waves height. Thus, including waves

direction and wind direction as predictors in addition to the wind speed, we only managed to improve the validation error to 0.6290 (see table 8.2).

Model name	predictors	RMSE	MAE	R-squared	Success rate
waves_lr2	WSPD, MWD, WDIR	0.6290	0.4749	0.29	88.54 %

Table 8.2. *Validation results of LR waves_lr2 model*

waves_lr2 model gives a RMSE of 0.7877 for 2019 data and 0.5692 for Real time data predictions.

8.2 SVR

Support Vector Machines with Gaussian kernel function are applied firstly to model WVHT with predictors: WSPD, MWD and WDIR. This model gives a validation RMSE of 0.5889 as seen in table 8.3 and a little R-squared value of 0.37. That can indicate us that we need more information (features) to describe more precisely the waves state in the selected location of Santa Maria station.

Model name	predictors	RMSE	MAE	R-squared	Success rate
waves_svm1	WSPD, MWD, WDIR	0.5889	0.4312	0.37	89.27 %

Table 8.3. *Validation results of SVM waves_svm1 model*

Predictions made by *waves_svm1* model are represented against true responses for 2018 data in figure 8.2, placing the waves direction on the x-axis and wave height value on the y-axis.

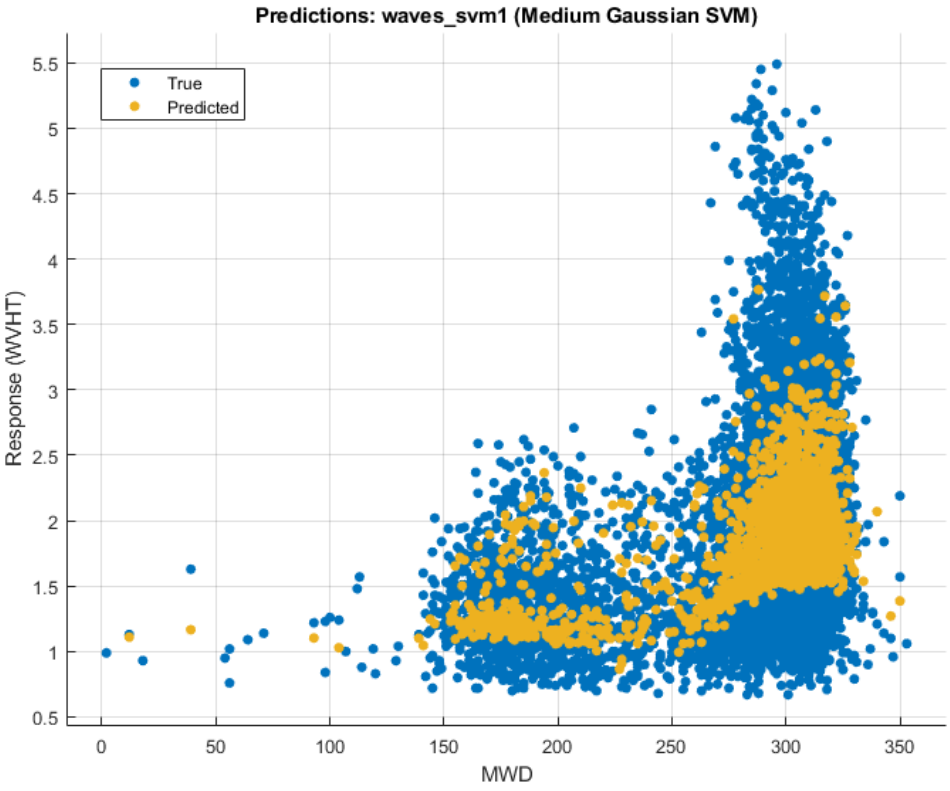


Figure 8.2. *Forecasting response of waves_svm1 model for data of 2018*

Learning curves for *waves_svm1* (figure 8.3) fluctuate making noisy movements and are so irregular while not decreasing. Learning curves theory says that training data set could be unrepresentative for the problem [65], so we select WTMP as additional predictor to train model. Moreover, if validation curve is lower than training curve in some points it may be because it is simpler to predict it than training set. To avoid the influence of one training-validation sets selection, we will use instead of 20% hold-out CV scheme (applied in *waves_svm1* training); the 5-fold CV for next models training we will carry out.

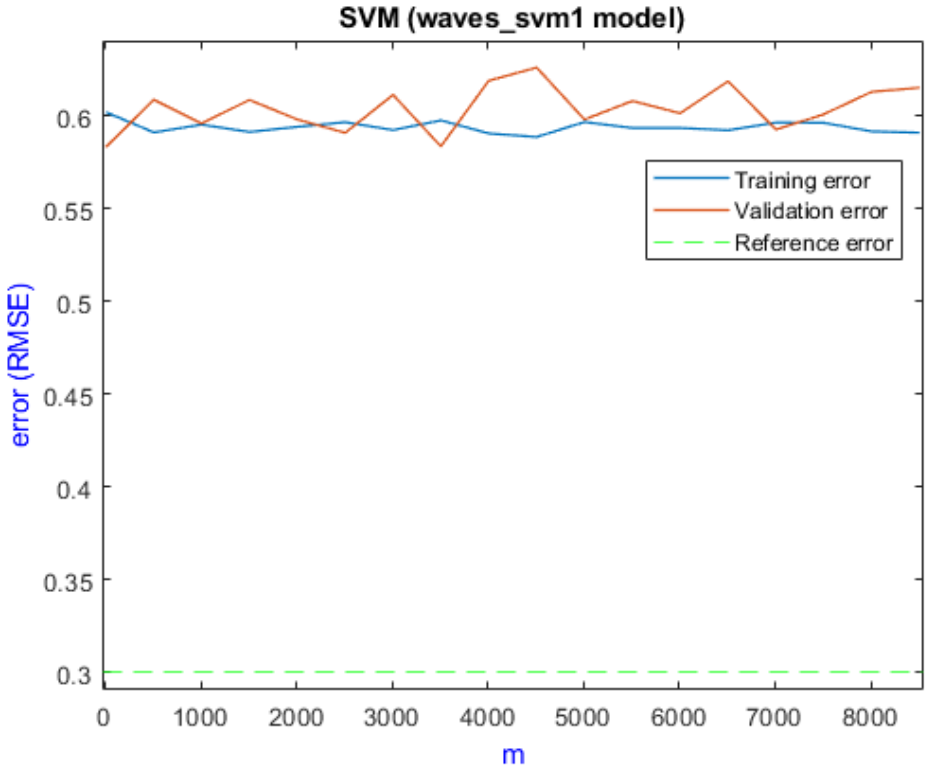


Figure 8.3. *Learning curves plot for waves_svm1 model trained with data from 2018 and using 5-fold Cross Validation scheme*

Two models have been obtained after optimization phase giving a better RMSE as can be seen in the table 8.5.

Model name	**predictors**	**RMSE**	**MAE**	**R-squared**	**Success rate**
waves_svm2	WSPD, MWD, WDIR, WTMP	0.5679	0.4082	0.43	89.66 %
waves_svm3	WSPD,MWD,WDIR, WTMP,PRES	0,5203	0,3709	0,52	90.52 %

Table 8.4. *Validation results of SVM waves_svm2 and waves_svm3 models*

Although *waves_svm3* gives a smaller validation error than *waves_svm2*; it generalize a little worst for WVHT forecasting of 2020 (see table 8.5).Considering also that is better to use a model with less input variables, we would choose *waves_svm2* as best SVM model between obtained models.

Model name	RMSE (2019)	RMSE (Real Time)	% (2019)	% (2020)
waves_svm2	0.8174	0.6149	85.11 %	88.80 %
waves_svm3	0.8019	0.7015	85.39 %	87.22 %

Table 8.5. *RMSE and sucess rate of waves_svm2 and waves_svm3 models for data of 2019 and 2020 (Real Time)*

Validation predicted response and true response for *waves_svm2* model have been plot (see figure 8.4). It shows a higher accurate predictions compared with the predictions made by *waves_svm1* model (figure 8.2).

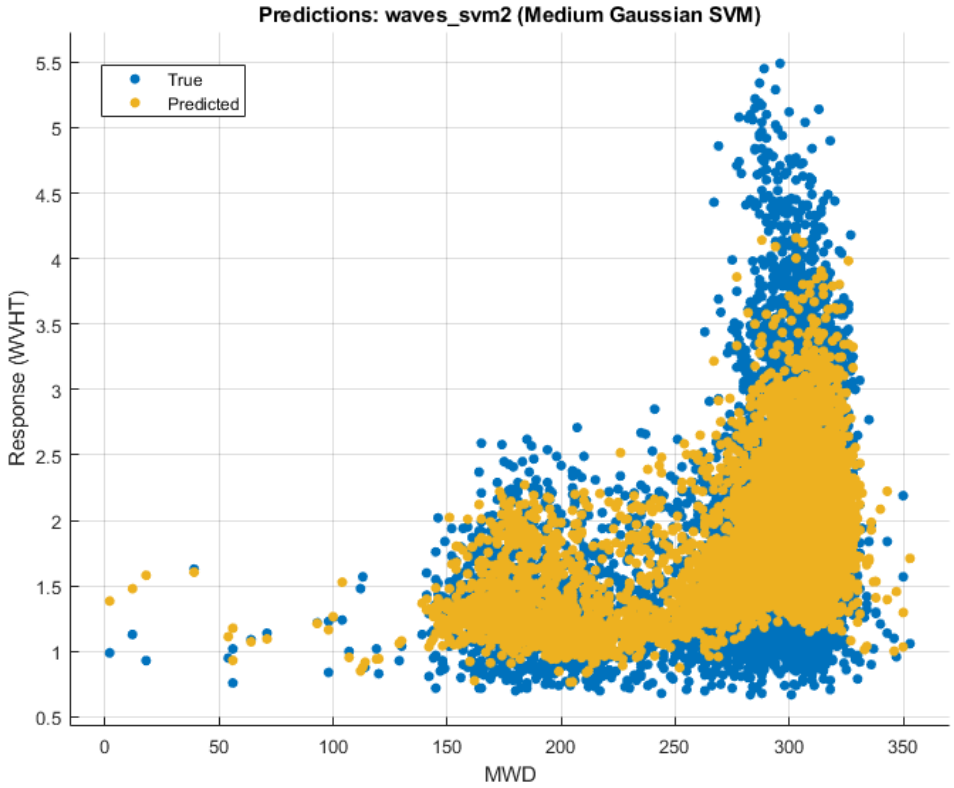


Figure 8.4. *Forecasting response of waves_svm2 model for data of 2018*

8.3 GPR

Gaussian Process Regression algorithm is applied firstly using WSPD as input model and in a second model using MWD and WDIR too as predictors. Results summarized in table 8.6 shows that the second model trained is better.

Model name	predictors	RMSE	MAE	R-squared	Success rate
waves_gpr1	WSPD	0.6655	0.5080	0.22	87.88 %
waves_gpr2	WSPD, WDIR, MWD	0.5820	0.4373	0.40	89.39 %

Table 8.6. *Validation results for GPR waves_gpr1 and waves_gpr2 models*

Although *waves_gpr2* has a good performance, if we want to optimize it, considering the reference RMSE of 0.3, we could say that the model suffers from high bias by observing its learning curves plot (8.5).

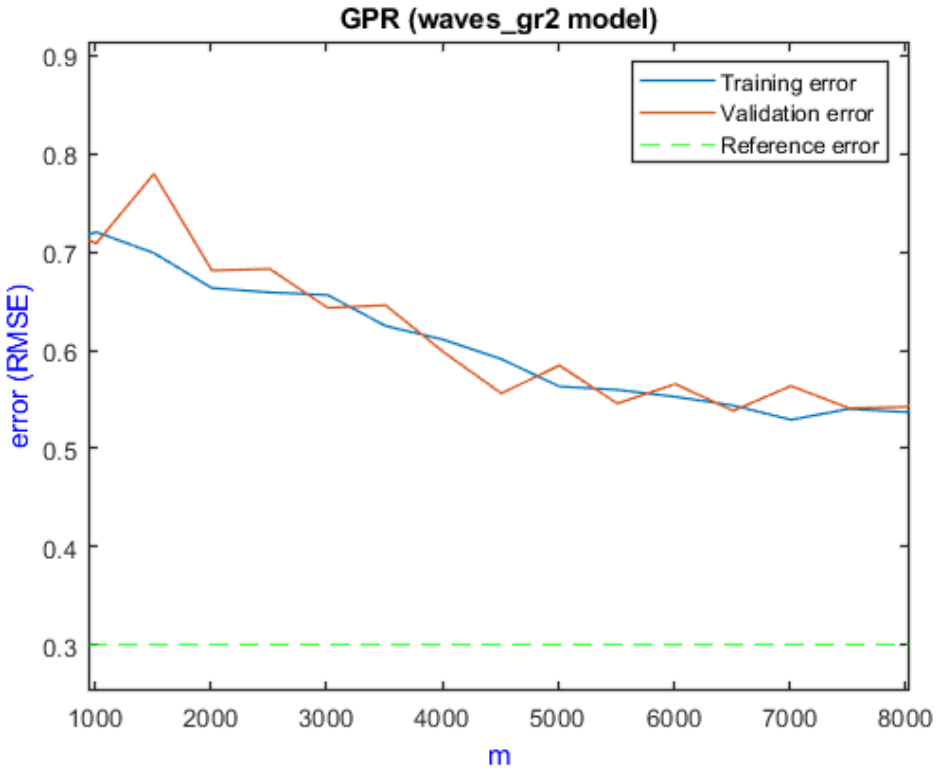


Figure 8.5. *Learning curves plot for waves_gpr2 model trained with data from 2018*

Adding PRES and WTMP as predictors in addition to WSPD, MWD and WDIR as well as using a data set with 2017 and 2018 data rows for training; we obtain the waves_gpr3 model which gives a validation RMSE of 0.5021, a RMSE for 2019 data of 0.7312 and 0.6940 for Real Time data (see table 8.7).

Model name	predictors	RMSE	MAE	R squared	RMSE (2019)	RMSE (2020)
waves_gpr3	WSPD, WDIR, MWD, PRES, WTMP	0.5021	0.3755	0.54	0.7312	0.6940

Table 8.7. *Validation and testing results for GPR waves_gpr3 model*

Model name	Success rate (validation)	Success rate (2019)	Success rate (2020)
waves_gpr3	90.85 %	86.68 %	87.36 %

Table 8.8. *Validation and testing success rates for GPR waves_gpr3 model*

8.4 Feed-forward ANN

Feed-forward Backprop Artificial Neural Networks (ANN) are pretty used in modeling letting us create different networks with different number of hidden layers and number of neurons in each layer to best overcome the regression problem.

Main parameters that we select while configuring our ANN are:

- Nº of hidden layers
- Nº of neurons for each hidden layer
- Training function (Lavenberg-Marquardt, gradient descent,...)
- Transfer function in neurons of each layer (Tansig, Purelin or Logsig). We will apply in all models trained the Tansig function in neurons from hidden layers and Purelin in the output layer.

We start with a simple network to evaluate its performance in Significant waves height prediction. Table 8.9 shows configuration for *waves_ann1* network and table 8.10 summarize results of training the neural network with data from 2018.

Model name	Predictors	Training Function	Nº hidden layers	Nº neurons/layer
waves_ann1	WDIR, WSPD, MWD	Trainlm	1	15

Table 8.9. *waves_ann1 model parameters selected for training*

Model name	RMSE			R-squared		
	Train	Valid.	Test	Train	Valid.	Test
waves_ann1	0.5130	0.5533	0.5365	0.73	0.68	0.70

Table 8.10. *training, validation and testing results for ann waves_ann1 model*

Model name	Success rate
waves_ann1	90.22 %

Table 8.11. *Testing success rate for waves_ann1 model*

It is remarkable how the R-squared is larger for one of the simplest neural networks trained than for other algorithms like SVR or GPR used in WVHT prediction. It is a good indication that the neural network captures the information present in data and learns from it pretty well.

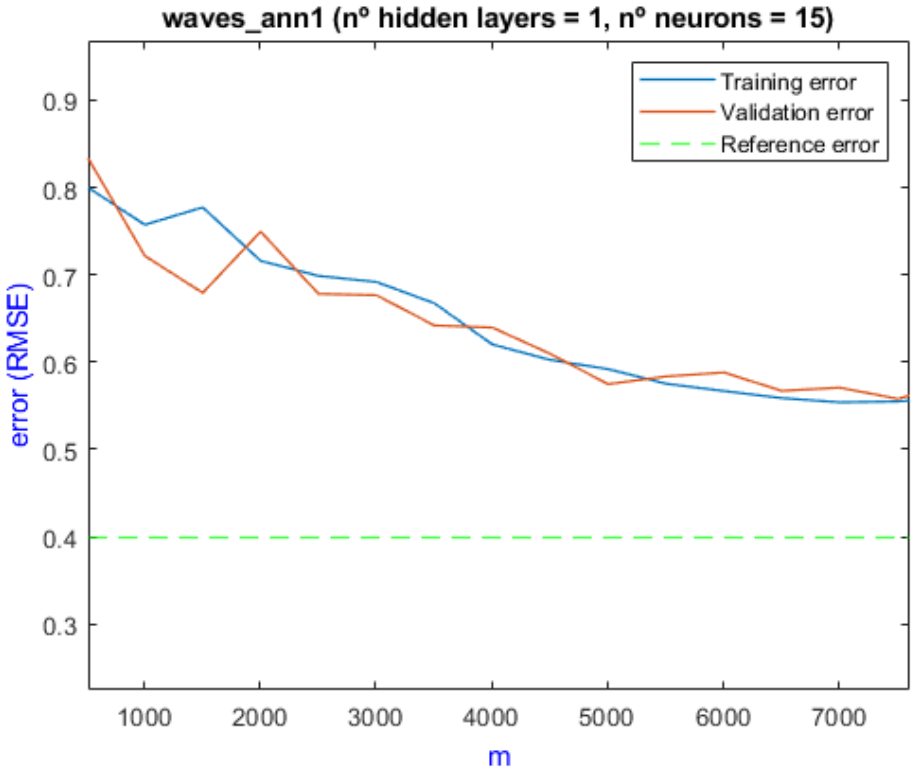


Figure 8.6. *Learning curves plot for waves_ann1 model trained with data from 2018*

Learning curves plotted for *waves_ann1* model (figure 8.6) are decreasing for larger number of training examples (m) as expected and comparing them with the reference error line, we could say this neural network suffers from high bias. To fix high bias, we try some more complex configurations of neural network, taking into account that a too complex network would be very computationally costly and not a feasible solution in FOWT applications context.

Table 8.12 shows best networks architecture found which give a good testing RMSE (see table 8.13) of the order of 0.4. Both *waves_ann2* and *waves_ann3* models are trained with data from 2018.

Model name	Predictors	Training Function	Nº hidden layers	Nº neurons/layer
waves_ann2	WDIR, WSPD, MWD, PRES, WTMP	Trainlm	4	25 x 20 x 15 x 7
waves_ann3	WSPD, WDIR, MWD, PRES, WTMP	Trainlm	4	20 x 25 x 7 x 2

Table 8.12. *waves_ann2 and waves_ann3 models parameters selected for training*

Model name	RMSE		
	Train	Valid.	Test
waves_ann2	0.4531	0.4948	0.4619
waves_ann3	0.4214	0.4947	0.4517

Table 8.13. *training, validation and testing RMSE for ann waves_ann2 and waves_ann3 models*

Model name	Success rate
waves_ann2	91.59 %
waves_ann3	91.77 %

Table 8.14. *testing success rate for ann waves_ann2 and waves_ann3 models*

The *waves_ann3* model (diagram 8.7) gives an RMSE of 0.8112 for 2019 and 0.7421 for 2020 data.

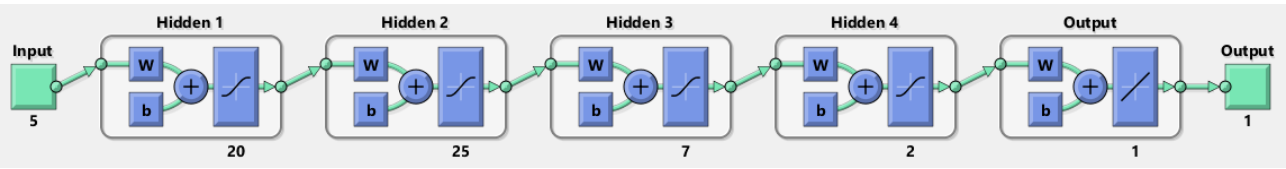


Figure 8.7. *waves_ann3 architecture diagramm*

8.5 Models comparison

As we did in the previous chapter, we compare best models trained with each ML technique applied to significant waves height prediction. Table 8.15 list these models in order to compare their performance.

Algorithm	Model Name	RMSE (validation)	RMSE (2019)	RMSE (2020)	Success rate (validation)
LR	waves_lr2	0.6290	0.7877	0.5692	88.54 %
SVR	waves_svm2	0.5679	0.8174	0.6149	89.66 %
GPR	**waves_gpr3**	**0.5021**	**0.7312**	**0.6940**	**90.85 %**
Feed-forward ANN	waves_ann3	0.4517	0.8112	0.7421	91.72 %

Table 8.15. *Best models for WVHT prediction obtained by each ML algorithm together with validation RMSE and RMSE for 2019 and Real Time (2020) data.*

The best algorithm to predict the Significant waves height turns out to be the **Gaussian process regression (GPR)** with the second lowest validation RMSE (0.5021) and best RMSE for 2019 (0.7312) and 2020 predictions (0.6940). *waves_gpr3* model training context which gives the best results of every GPR model trained consist of:

- **Data set**: data from 2018
- **Predictors**: WSPD, WDIR, MWD, PRES, WTMP

With a **success rate** of **90.85 %**, we claim the *waves_gpr3* model has an acceptable predictive performance although it could be improved. Predicted against true responses plot for first 200 observations from 2020 for *waves_gpr3* model can be seen in figure 8.8.

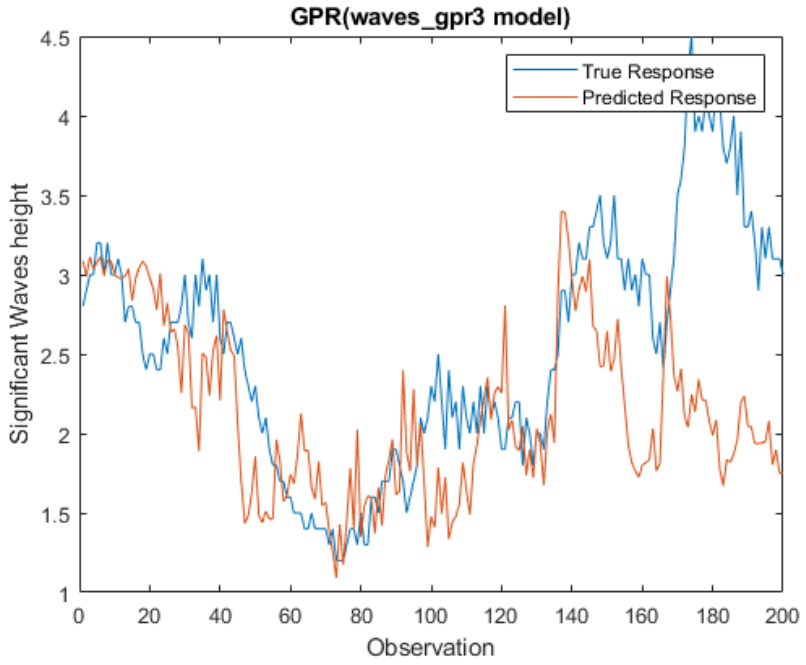


Figure 8.8. *predicted against true response for waves_gpr3 model for 200 first observations of data from 2020*

Chapter 9

Misalignment forecasting: algorithms application

Wind and waves misalignment forecasting is a novel and pioneering task carried out in this project.

Algorithms applied for Misalignment forecasting have been:

- Linear Regression (LR)
- Gaussian Process Regression (GPR)
- Support Vector Machines for Regression (SVR)
- Nonlinear Autoregressive Neural Network (NAR)
- Nonlinear Autoregressive with External Input (NARX)

Models trained **forecast the next hour misalignment** (short-term forecasting). For neural networks, the data set has been split in training (70%), validation (15%) and testing (15%) sets while for the rest of algorithms a hold validation scheme is configured with a held-out data subset of 20% (validation data set).

RMSE is measured in angle degrees and best models have been additionally tested with data from 2019 and 2020 ("Real Time").

9.1 LR

First linear regression model trained gives a good error in comparison with the initial reference error of 90º, as seen in table 9.1. However the R-squared value (0.33) indicates

that this model does not capture much information from data and it can be optimized. *mis_lr1* model has been trained with data from 2018.

Model name	Inputs in time	Predictors	RMSE	MAE	R-squared	Success rate
mis_lr1	1 h-before	MIS	53.9260	32.6721	0.33	70.04 %

Table 9.1. *Validation results of mis_lr1 LR trained model for next hour misalignment forecasting*

Figure 9.1 shows the predicted and true response for validation subset of data rows from 2018.

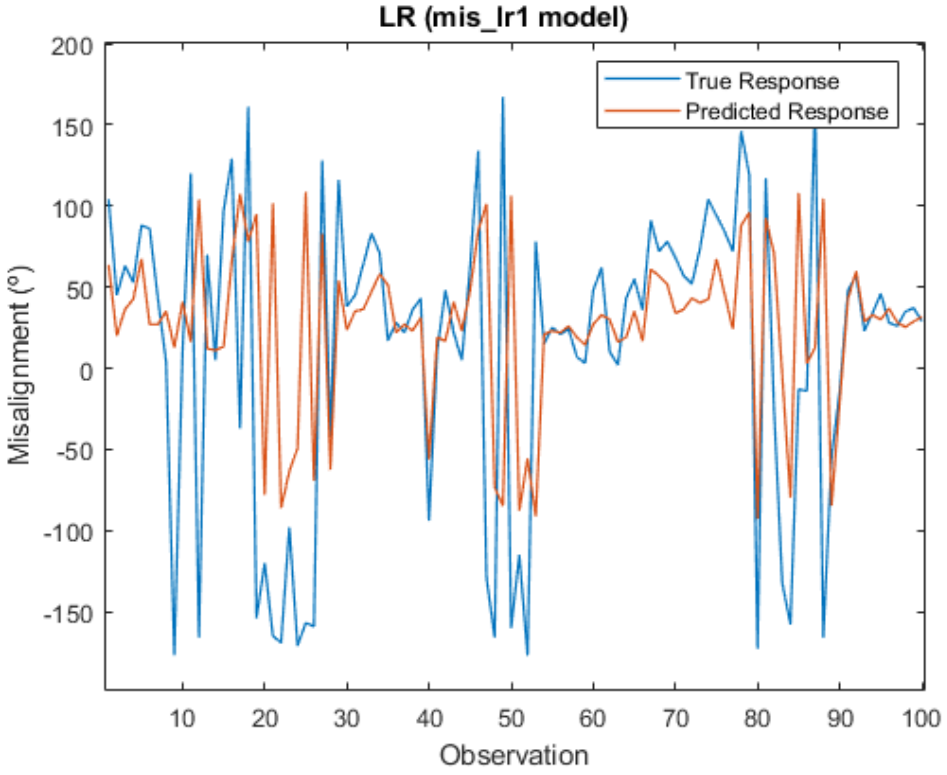


Figure 9.1. *Validation responses compared with predicted outputs by mis_lr1 model for 100 first observations of the validation subset of data from 2018*

We managed to improve the model a little bit by adding WTMP, WSPD and PRES 1-hour before values in addition to MIS feature as predictors. Results given by *mis_lr2* model are listed in the table 9.2.

Model name	Inputs in time	Predictors	RMSE	MAE	R-squared	Success rate
mis_lr2	1 h-before	MIS, WSPD, WTMP, PRES	52,1342	30,8361	0.35	71.04 %

Table 9.2. *Validation results of mis_lr2 LR trained model for next hour misalignment forecasting*

The little improvement of *mis_lr2* model compared to *mis_lr1* model can be appreciated in figure 9.2. (The 100 observations plotted in figure 9.1 and 9.2 are not exactly the same because validation set is a randomly selection of 20% of rows before training the model).

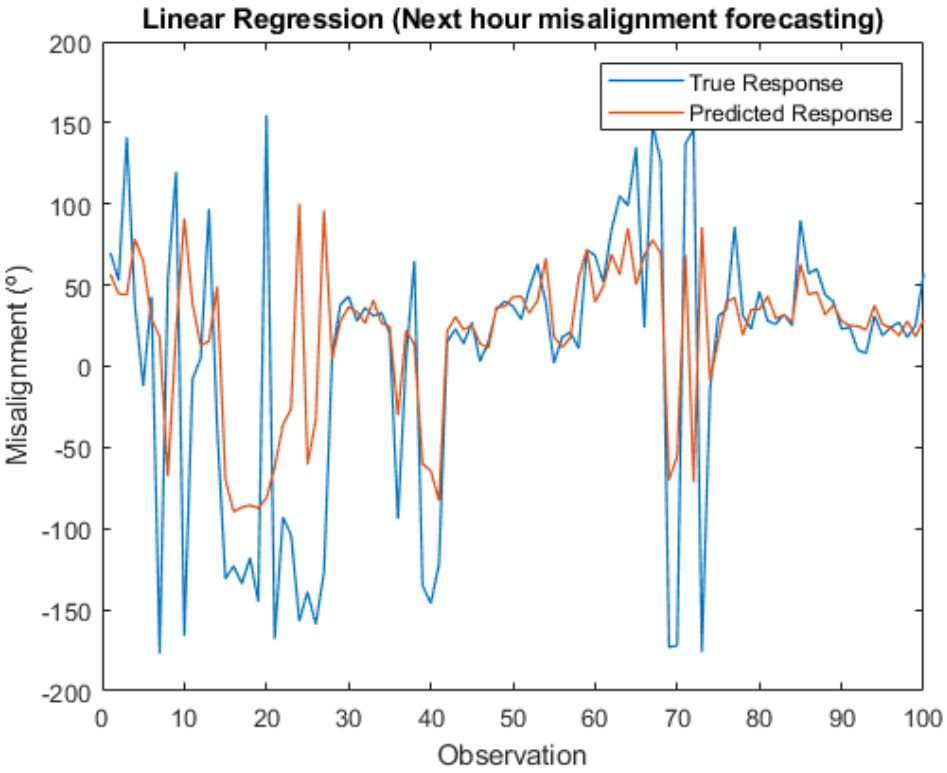


Figure 9.2. *Validation responses compared with predicted outputs by mis_lr2 model for 100 first observations of the validation subset of data from 2018*

Finally, we test the *mis_lr2* model with data from 2019 obtaining a RMSE of 62.4652 and a RMSE of 66.2944 for Real Time data.

9.2 SVR

Support Vector Regression with Gaussian kernel function has been applied selecting as predictor the current misalignment to predict it in the next hour. First SVR model trained with data from 2018 gives a RMSE of 54.8385 as shown in table 9.3.

Model name	Inputs in time	Predictors	RMSE	MAE	R-squared	Success rate
mis_svm1	1 h-before	MIS	54.8385	29.9080	0.31	69.53 %

Table 9.3. *Validation results for mis_svm1 model for next hour misalignment forecasting*

Learning curves for this model are as expected so good, even achieving a validation error in one point lower than 50 º (see figure 9.3).

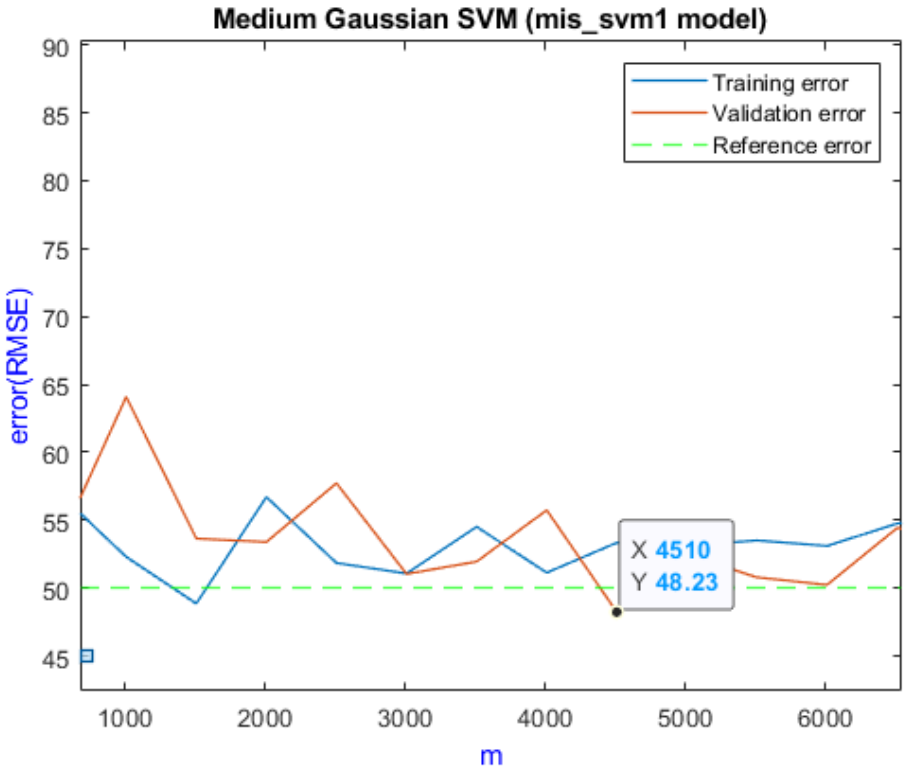


Figure 9.3. *Learning curves plot for mis_svm1 model*

To ensure the performance of this model, we test the *mis_svm1* with data from 2019 and 2020, obtaining RMSE values of 63.5201 and 65.4028 respectively. In order to improve these numbers, we try some models training obtaining similar RMSE. The *mis_svm2* model trained (see results in figure 9.4) gives best RMSE for 2019 of 61.7990 and 64.3001 for Real Time data .

Model name	Inputs in time	Predictors	RMSE	MAE	R-squared	Success rate
mis_svm2	1 h-before	MIS, WSPD, WTMP	51.9623	31.9612	0.32	71.13 %

Table 9.4. *Validation results for mis_svm2 model for next hour misalignment forecasting*

Figure 7.2 shows the true and predicted responses made by *mis_svm2* for 2018 data registers.

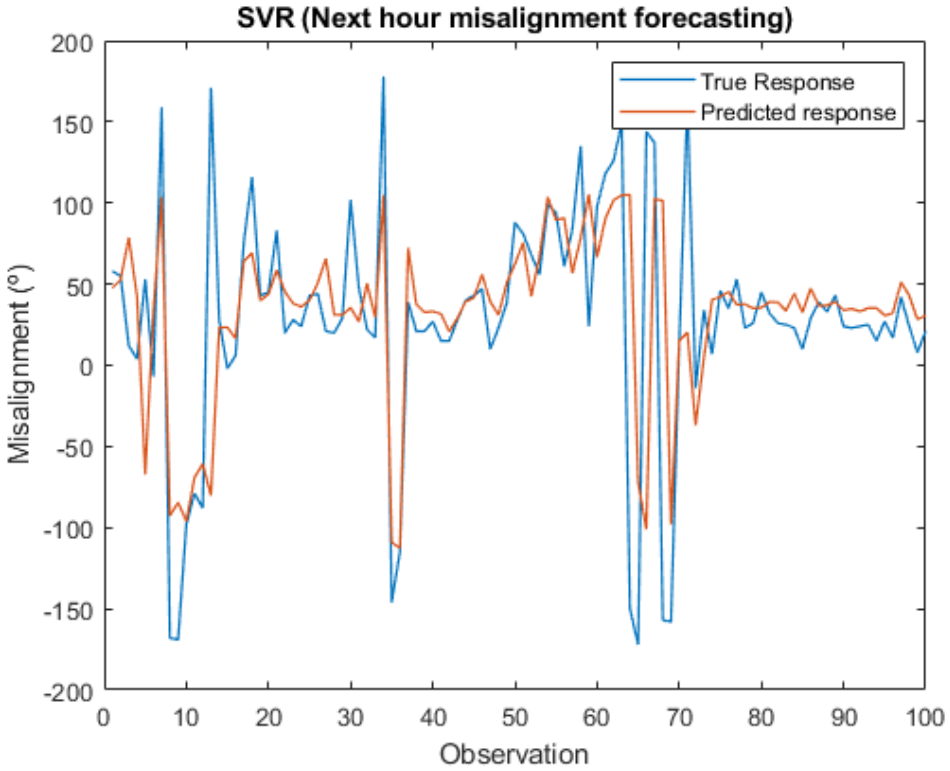


Figure 9.4. *Validation responses compared with predicted outputs by mis_svm2 model for 100 first validation subset observations of data from 2018*

9.3 GPR

Gaussian Process Regression first model has been trained with misalignment from this moment as predictor to next hour misalignment forecasting. We observe in table 9.5 that *mis_gpr1* model gives a good RMSE of 52.0531 although R-squared is still too little. We try to add WTMP and WSPD as predictors getting a RMSE of 51.5981.

Model name	Inputs in time	Predictors	RMSE	MAE	R-squared
mis_gpr1	1 h-before	MIS	52.0531	31.4042	0.38
mis_gpr2	1 h-before	MIS, WSPD, WTMP	51.5981	31.2231	0.39

Table 9.5. *Validation results for mis_gpr1 and mis_gpr2 trained models for next hour misalignment forecasting*

Model name	Success rate
mis_gpr1	71.08 %
mis_gpr2	71.33 %

Table 9.6. *Success rate for mis_gpr1 and mis_gpr2 models*

We plot learning curves for *mis_gpr2* stablishing a reference error of 50º (30º less than 90º which has been considered the criteria or top margin to determine if RMSE value is good). Figure 9.5 shows this model is good-fit and validation curve is located below the training curve which is one of the objectives in Machine Learning models. Thus in this case we finished the optimization phase.

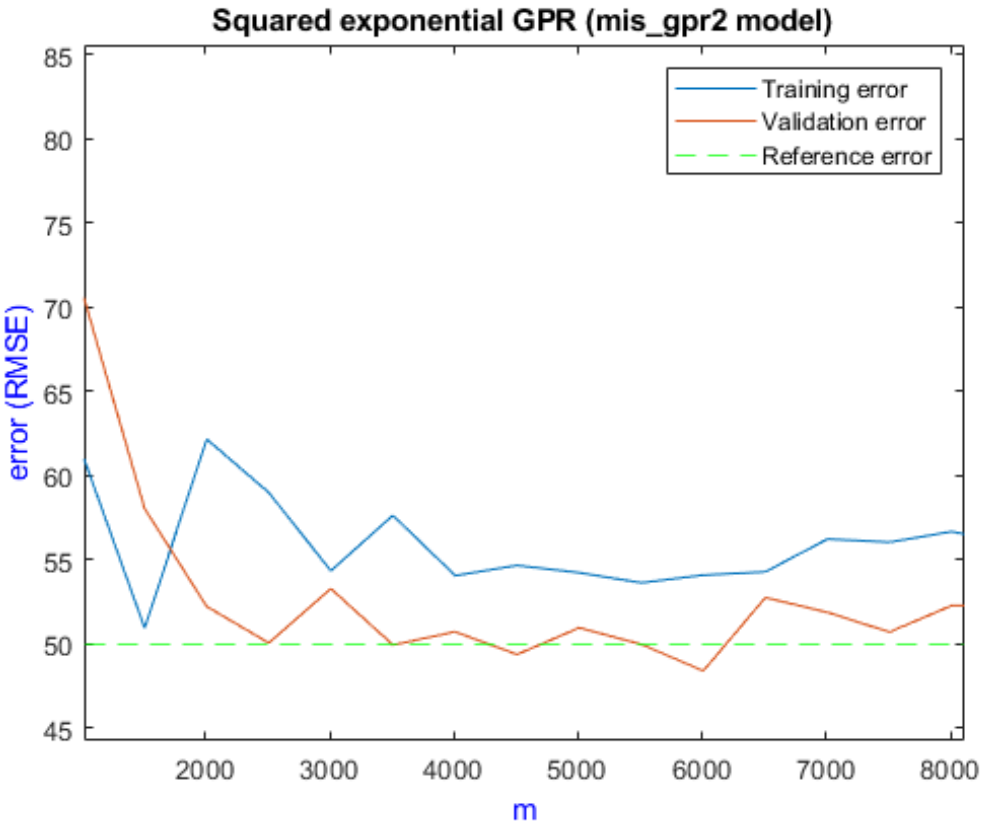


Figure 9.5. *Learning curves plot for mis_gpr2 model*

Plot of predicted and true responses for *mis_gpr2* (figure 9.6) reveal too the distribution of misalignment registers values from the data set used.

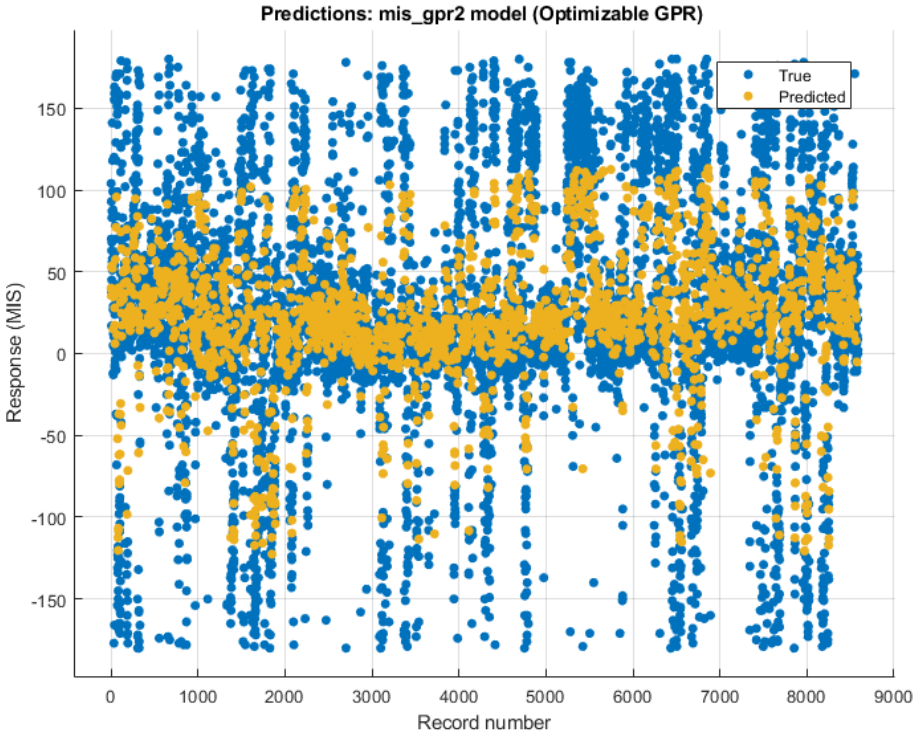


Figure 9.6. *True and predicted responses plot for mis_gpr2 model for data from 2018*

mis_gpr2 model is tested for data from 2019 and Real Time, giving RMSE values of 59.7840 and 62.7212 respectively.

9.4 NAR

Nonlinear Autorregressive Neural Networks have been found out to capture pretty well the information into the data set for misalignment forecasting. Every networks trained in this chapter (NAR and NARX explained in the next section) have one hidden layer.

First NAR model consist of a neural network with one hidden layer with 10 neurons that receive misalignment from one previous hour of forecasting as input (the delay is 1). This first network (*mis_nar1*) has been trained with data from 2018 giving the results shown in the table 9.7.

Model name	Nº neurons	Nº delays	RMSE			R-squared		
			Train	Valid.	Test	Train	Valid.	Test
mis_nar1	10	1	53.1438	54.0507	53.6406	0.60	0.59	0.62

Table 9.7. *Results of mis_nar1 model training for next hour misalignment forecasting*

Model name	Success rate
mis_nar1	70.20 %

Table 9.8. *Success rate for mis_nar1 model*

If we look at learning curves plot for *mis_nar1* model (figure 9.7), we appreciate that training error curve reaches the reference RMSE line and in general we can say that this model is good fit. However, the validation curve is a little bit above the training curve so we will try optimize the model. We try train more complex NAR establishing a higher delay and choosing a greater number of neurons for the hidden layer.

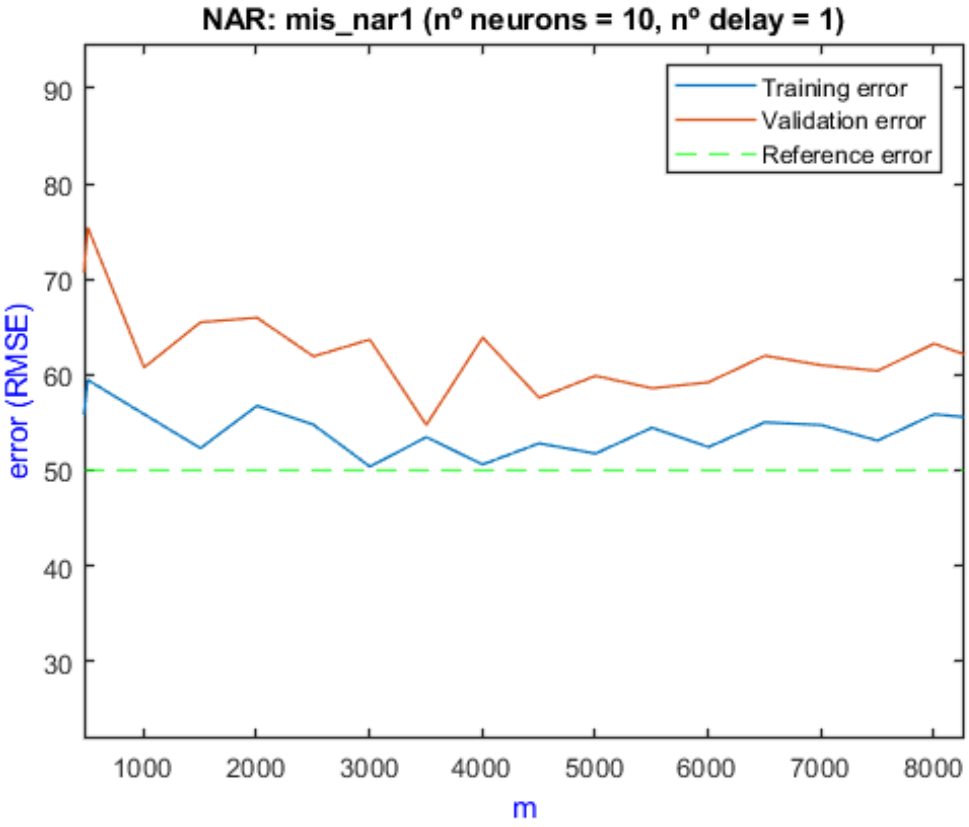


Figure 9.7. *Learning curves plot for mis_nar1 model*

The best NAR optimized model which gives a bit better results than *mis_nar1* is listed in

the table 9.9. It should be mentioned that *mis_nar1* has been configured with the training function "Lavenberg-Marquardt" (Trainlm) while *mis_nar2* model has been trained with "Bayesian Regularization Backpropagation" function (Trainbr). Trainbr apply a training-testing scheme data set partition (without validation subset). Thus validation RMSE is not available for *mis_nar2*.

Model	**Nº**	**Nº**	**RMSE**		**R-squared**	
name	**neurons**	**delays**	**Train**	**Test**	**Train**	**Test**
mis_nar2	23	4	50.3944	49.7858	0.66	0.63

Table 9.9. *Results of mis_nar2 model training for next hour misalignment forecasting*

Model name	**Success rate**
mis_nar2	72.00 %

Table 9.10. *Testing success rate for mis_nar2 model*

mis_nar2 gives a RMSE of 61.8016 for 2019 misalignment forecasting and a RMSE of 62.7157 for Real Time forecasting. Finally, we can see predicted against true responses and MSE error (RMSE squared) plot for *mis_nar2* in figure 9.8.

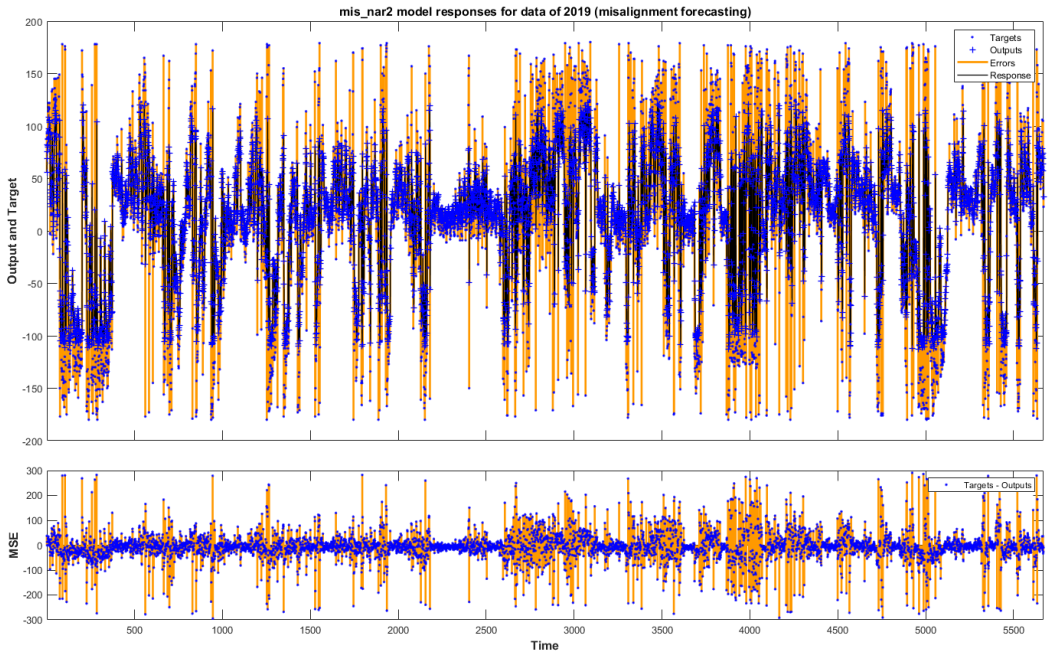


Figure 9.8. *Forecasting response of mis_nar2 model for data of 2019*

9.5 NARX

Nonlinear Autoregressive with Exogenous Input Networks (NARX) trained outperforms NAR models already exposed. First NARX with a remarkable good RMSE is *mis_narx1* (see figure 9.9) whose training measures results are summarized in table 9.11. We used as predictors MIS and WSPD of data set from 2018 and we trained the model with "Trainlm" function.

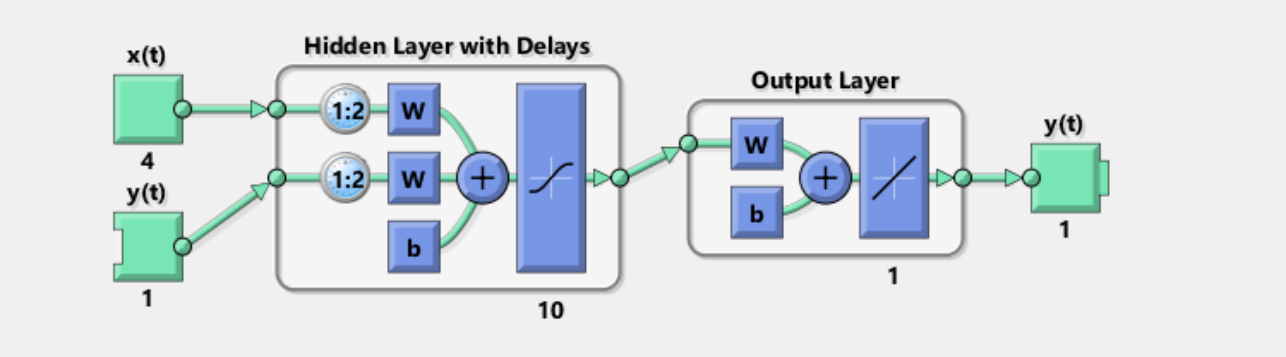


Figure 9.9. *mis_narx1 architecture diagramm*

Model name	Nº neurons	Nº delays	RMSE			R-squared		
			Train	Valid.	Test	Train	Valid.	Test
mis_narx1	10	2	51.6333	52.7523	**49.0421**	0.64	0.62	0.65

Table 9.11. *Training, validation and testing results for mis_narx1 model training for next hour misalignment forecasting*

Model name	Success rate
mis_narx1	**72.75 %**

Table 9.12. *Testing success rate for mis_narx1 model*

Learning curves plot for *mis_narx1* (figure 9.10) shows that validation and training RMSE are close to the reference error as time as gap between validation and training curves is significant which could mean *mis_narx1* model suffers from high variance. We will try to optimize it in order to get smaller validation RMSE despite obtaining a little higher training RMSE. Selecting more training examples from data or configuring a simpler neural network could help.

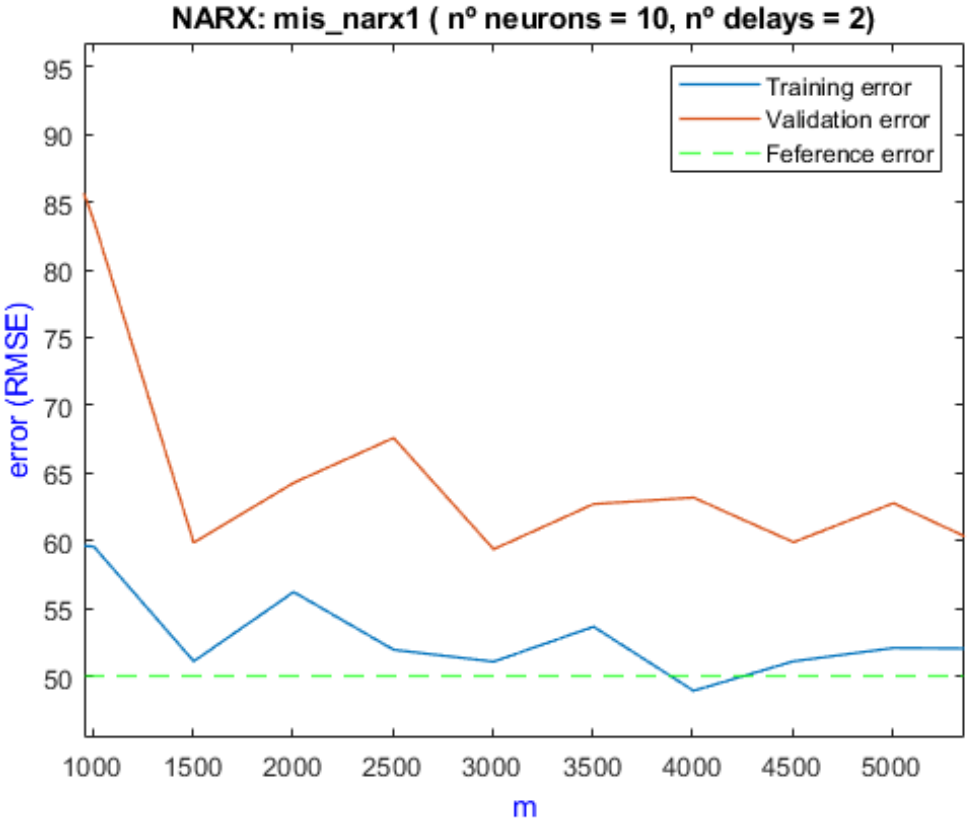


Figure 9.10. *Learning curves plot for mis_narx1 model*

After optimization, we obtain a NARX model, *mis_narx2*, which improves RMSE values for 2019 and Real time misalignment forecasting, compared with those for *mis_narx1* model (see comparison in table 9.15). *mis_narx2* model has been trained with data from 2010 to 2018 and taking same predictors as used for mis_narx1 training: MIS and WSPD . Results are summarized in table 9.13.

Model	**Nº**	**Nº**	**RMSE**			**R-squared**		
name	**neurons**	**delays**	**Train**	**Valid.**	**Test**	**Train**	**Valid.**	**Test**
mis_narx2	8	2	51.1227	50.5203	50.3699	0.67	0.68	0.68

Table 9.13. *Training, validation and testing results for mis_narx2 model training for next hour misalignment forecasting*

Model name	**Success rate**
mis_narx2	72.02 %

Table 9.14. *Testing success rate for mis_narx2 model*

Model name	RMSE (2019)	RMSE (Real Time)	% (2019)	% (2020)
mis_narx1	59.3815	63.0830	67.01 %	64.95 %
mis_narx2	58.1550	61.5061	67.69 %	65.83 %

Table 9.15. *RMSE and success rate for 2019 and 2020 MIS forecasting comparison between mis_narx1 and mis_narx2*

Response plot for 2020 data obtained with *mis_narx2* is showed in figure 9.11.

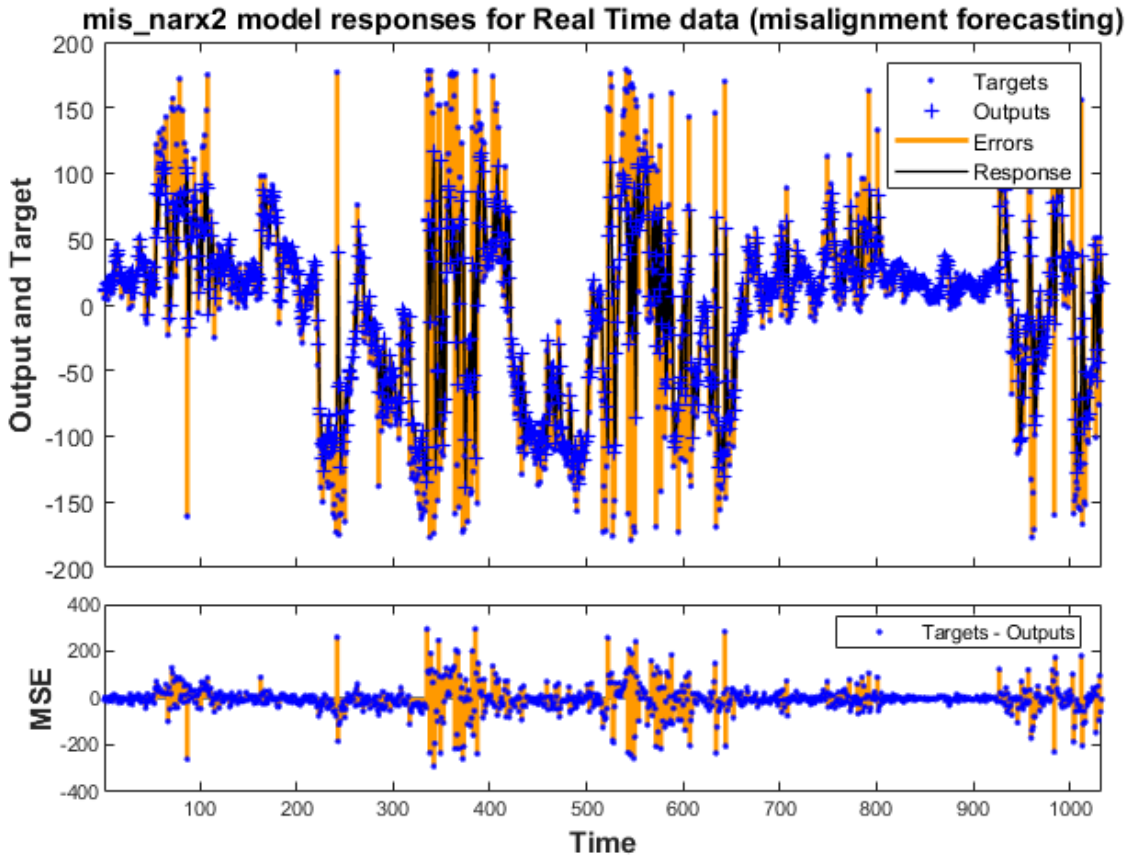


Figure 9.11. *Forecasting response of mis_narx2 model for Real time data (2020)*

9.6 Models comparison

Best models obtained for wind-waves misalignment forecasting by applying different ML algorithms are collected in table 9.16.

Algorithm	Model Name	RMSE (validation)	RMSE (2019)	RMSE (2020)	Success rate (validation)
LR	mis_lr2	52.1342	62.4652	66.2944	71.04 %
SVR	mis_svm2	51.9623	61.7990	64.3001	71.13 %
GPR	mis_gpr2	51.5981	59.7840	62.7212	71.33 %
NAR	mis_nar2	49.7858	61.8016	62.7157	72.34 %
NARX	**mis_narx2**	**50.3699**	**58.1550**	**61.5061**	**72.02 %**

Table 9.16. *Best models for misalignment forecasting obtained by each ML algorithm together with validation RMSE and RMSE for 2019 and Real Time (2020) data.*

Table 9.16 reveals that **Nonlinear Autoregressive with External Input (NARX) Neural Networks** outperformed the rest of algorithms for misalignment forecasting. Even when NAR model gives a bit lower validation RMSE than NARX model, the last one outperforms the first for 2019 and 2020 data. Thus, *mis_narx2* gives a validation error better than expected (50.3699) and best RMSE for 2019 (58.1550) and 2020 (61.5061) data compared with those obtained by all trained models with different algorithms. The **optimal configuration and training choices** made to train the *mis_narx2* model are:

- **Data set**: data from 2010 to 2018
- **Predictors**: MIS and WSPD
- **Nº neurons in the hidden layer**: 8
- **Nº input delays**: 2 hours before forecasting moment (t)

The **success rate** of *mis_narx2* model is **72.02%** and predicted against true responses of 200 first observations from 2020 plot for this model is represented in figure 9.12. The performance of this model is not good enough for real predictions but these are the first results of misalignment modelling and a good start point for other projects involved in the misalignment forecasting task.

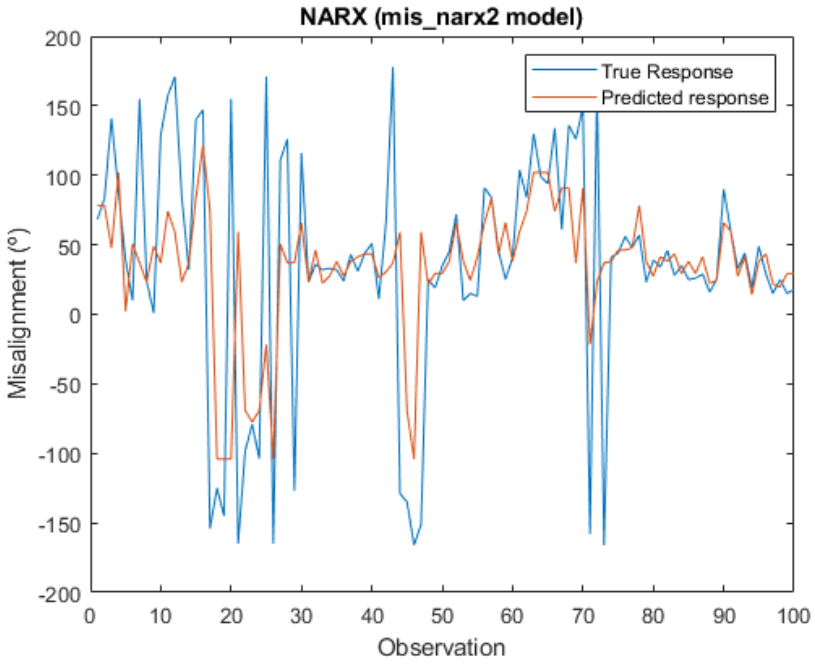


Figure 9.12. *predicted against true response for mis_narx2 model for 200 first observations of data from 2020*

Chapter 10

Conclusions and Future work

10.1 Conclusions

In this project we have achieved our main objective of modeling the stochastic wind and waves loads which impact floating offshore wind turbines by carrying out all phases in a Machine Learning process from scratch.

We can state that the data collection and data pre-processing phase have been crucial to getting more accurate models by machine learning application. Moreover, data cleaning and analysis led us to the more effective selection of predictors that greatly improves the performance of the models as well as ensuring the reliability of such models to make realistic predictions.

We explored some machine learning techniques to apply in next hour wind speed forecasting and compare their performance. We demonstrated that wind speed forecasting is possible applying only ML techniques instead of traditional physical and statistical models commonly used for this while obtaining acceptable prediction errors. For the specific case of data collected for this project from Santa Maria station (CA, USA) selected as a potential wind farm location, NARX neural network model outperformed the rest of the algorithms.

In the same line, ML techniques have been used to significant waves height prediction in the frequency domain. Trained models with wind components and weather variables as input have given similar RMSE to those found in literature and the GPR model proved to be the most accurate in the concrete predictive context of this work.

In the literature we have not found models for wind-waves misalignment forecasting. Therefore this project is a pioneer in this task. ML techniques have been applied for next hour misalignment forecasting. Between all models trained, the NARX algorithm has turned out to be the most efficient.

Finally, we conclude after experienced the need to devote time to the correct evaluation of models with techniques such as learning curves plot in order to make better optimization decisions, as there would not be enough time to test all possible models and parameter configurations. Given its complexity, with greater knowledge of optimization more accurate models can be obtained.

Beyond the models' results attained, an ML project has been developed following a proper methodology and focusing on learning during the process.

10.2 Future Work

As a continuation of this project, it would be interesting to start by training the models obtained here with data from other locations and comparing the results with those of this work. This line would allow study how much the particular site for wind farm location influences the models' performance.

On the other hand, future works focused on optimizing the current models or outperforming current RMSE values with new models can experiment:

- Collecting data features not considered here and applying advanced feature selection techniques to create new data sets.
- Applying ML techniques for wind and waves seasonal modeling. The phenomena of wind and waves can be linked to the season. Thus training models for each season separately may certainly improve the forecasting.
- Explore deep learning algorithms or other ML techniques not applied in this work.
- Carefully studying the computational cost of execution of the algorithms used both in Data pre-processing and Modelling stages; trying to optimize the execution time of them, in addition to the predictive error as was done in this project.
- The creation of hybrid models can lead to the integration of these models into more sophisticated predictive systems for wind power or wind and waves fatigue into the turbine.

To conclude, misalignment models obtained here are undoubtedly an open door to many other projects from different areas of the offshore wind energy research community, which together seek the integration of floating wind farms into our society.

Bibliography

[1] Esteban, M., & Leary, D. (2012). Current developments and future prospects of offshore wind and ocean energy. *Applied Energy*, *90*(1), 128–136.

[2] Mikati, M., Santos, M., & Armenta, C. (2012). Modelado y simulación de un sistema conjunto de energía solar y eólica para analizar su dependencia de la red eléctrica. *Revista Iberoamericana de Automática e Informática industrial*, *9*(3), 267–281.

[3] Ackermann, T. (2005). *Wind power in power systems*. John Wiley & Sons.

[4] Mikati, M., Santos, M., & Armenta, C. (2013). Electric grid dependence on the configuration of a small-scale wind and solar power hybrid system. *Renewable energy*, *57*, 587–593.

[5] Woods, B. (2019). Us has only one offshore wind energy farm, but a $70 billion market is on the way. Retrieved December 13, 2019, from https://www.cnbc.com/2019/12/13/us-has-only-one-offshore-wind-farm-but-thats-about-to-change.html

[6] Tomás-Rodríguez, M., & Santos, M. (2019). Modelado y control de turbinas eólicas marinas flotantes. *Revista Iberoamericana de Automática e Informática industrial*, *16*(4), 381–390.

[7] Henderson, A. R. (2002). Offshore wind in Europe: The current state of the art. *Refocus*, *3*(2), 14–17.

[8] Junginger, M., Louwen, A., Tuya, N. G., de Jager, D., van Zuijlen, E., & Taylor, M. (2020). Offshore wind energy. In *Technological learning in the transition to a low-carbon energy system* (pp. 103–117). Elsevier.

[9] Review, E. I. (2019). Europe's New Record for Offshore Wind Installations. Retrieved September 13, 2019, from https://energyindustryreview.com/renewables/europes-new-record-for-offshore-wind-installations/

[10] Dhabi, A. (2019). Future of wind: Deployment, investment, technology, grid integration and socio-economic aspects (A Global Energy Transformation paper).

[11] Butterfield, S., Musial, W., Jonkman, J., & Sclavounos, P. (2007). Engineering challenges for floating offshore wind turbines.

[12] Europe, W. (2017). Retrieved February 4, 2020, from https://windeurope.org/wp-content/uploads/files/about-wind/reports/Floating-offshore-statement.pdf

[13] Rubio, P. M., Quijano, J. F., López, P. Z., Lozano, J. J. F., Cerezo, A. G., & Casanova, J. O. (2019). Control inteligente para mejorar el rendimiento de una

plataforma semisumergible híbrida con aerogeneradores y convertidores de oleaje: Sistema de control borroso para la turbina. *Revista Iberoamericana de Automática e Informática industrial*, *16*(4), 480–491.

[14] Demolli, H., Dokuz, A. S., Ecemis, A., & Gokcek, M. (2019). Wind power forecasting based on daily wind speed data using machine learning algorithms. *Energy Conversion and Management*, *198*, 111823.

[15] Lei, M., Shiyan, L., Chuanwen, J., Hongling, L., & Yan, Z. (2009). A review on the forecasting of wind speed and generated power. *Renewable and Sustainable Energy Reviews*, *13*(4), 915–920.

[16] Soman, S. S. [S. S.], Zareipour, H., Malik, O., & Mandal, P. (2010). A review of wind power and wind speed forecasting methods with different time horizons. In *North American Power Symposium 2010* (pp. 1–8).

[17] Milligan, M., Schwartz, M., & Wan, Y.-h. (2003). *Statistical wind power forecasting models: Results for US wind farms*. National Renewable Energy Lab.(NREL), Golden, CO (United States).

[18] Abdel-Aal, R. E., Elhadidy, M. A., & Shaahid, S. (2009). Modeling and forecasting the mean hourly wind speed time series using gmdh-based abductive networks. *Renewable Energy*, *34*(7), 1686–1699.

[19] Torres, J., Aguilar, R., & Zuñiga-Meneses, K. (2018). Deep learning to predict the generation of a wind farm. *Journal of Renewable and Sustainable Energy*, *10*(1), 013305.

[20] Jones, M. S., Colle, B. A., & Tongue, J. S. (2007). Evaluation of a mesoscale short-range ensemble forecast system over the northeast united states. *Weather and Forecasting*, *22*(1), 36–55.

[21] Wang, X., Guo, P., & Huang, X. (2011). A review of wind power forecasting models. *Energy procedia*, *12*, 770–778.

[22] Azad, H. B., Mekhilef, S., & Ganapathy, V. G. (2014). Long-term wind speed forecasting and general pattern recognition using neural networks. *IEEE Transactions on Sustainable Energy*, *5*(2), 546–553.

[23] Armario, C. R. (2012). Viabilidad de un parque eólico con sistema de almacenamiento de energía mediante el uso de modelos de predicción. Retrieved July 28, 2020, from http://bibing.us.es/proyectos/abreproy/5116

[24] Brown, B. G., Katz, R. W., & Murphy, A. H. (1984). Time series models to simulate and forecast wind speed and wind power. *Journal of climate and applied meteorology*, *23*(8), 1184–1195.

[25] Torres, J. L., Garcia, A., De Blas, M., & De Francisco, A. (2005). Forecast of hourly average wind speed with arma models in navarre (spain). *Solar energy*, *79*(1), 65–77.

[26] Mills, T. C. (2019). Chapter 4 - ARIMA Models for Nonstationary Time Series. In T. C. Mills (Ed.), *Applied time series analysis* (pp. 57–69). Academic Press.

Retrieved June 12, 2020, from http://www.sciencedirect.com/science/article/pii/B9780128131176000041

[27] MathWorks. (n.d.[a]). Arx. Retrieved July 28, 2020, from https://es.mathworks.com/help/ident/ref/arx.html#mw_23d83be4-8bcf-4279-90c1-4fbd83133b6

[28] Vapnik, V. N. (1995). Constructing learning algorithms. In *The nature of statistical learning theory* (pp. 119–166). Springer.

[29] Soman, S. S. [Saurabh S], Zareipour, H., Malik, O., & Mandal, P. (2010). A review of wind power and wind speed forecasting methods with different time horizons. In *North american power symposium 2010* (pp. 1–8). IEEE.

[30] Li, G., & Shi, J. (2010). On comparing three artificial neural networks for wind speed forecasting. *Applied Energy*, *87*(7), 2313–2320.

[31] Brahimi, T. (2019). Using artificial intelligence to predict wind speed for energy application in saudi arabia. *Energies*, *12*(24), 4669.

[32] Senthil, K. P. (2019). Improved prediction of wind speed using machine learning. *EAI Endorsed Transactions on Energy Web*, *6*(23).

[33] Zhou, J., Shi, J., & Li, G. (2011). Fine tuning support vector machines for short-term wind speed forecasting. *Energy Conversion and Management*, *52*(4), 1990–1998.

[34] Roulston, M. S., Ellepola, J., von Hardenberg, J., & Smith, L. A. (2005). Forecasting wave height probabilities with numerical weather prediction models. *Ocean Engineering*, *32*(14-15), 1841–1863.

[35] Fernández, J. C., Salcedo-Sanz, S., Gutiérrez, P. A., Alexandre, E., & Hervás-Martínez, C. (2015). Significant wave height and energy flux range forecast with machine learning classifiers. *Engineering Applications of Artificial Intelligence*, *43*, 44–53.

[36] Floating offshore wind turbines. (2020). Retrieved from http://www.floatingwindturbineucm.com/

[37] Li, Z., & Adeli, H. (2018). Control methodologies for vibration control of smart civil and mechanical structures. *Expert Systems*, *35*(6), e12354.

[38] Service, N. W. (n.d.). Significant wave height. Retrieved June 3, 2020, from https://www.weather.gov/key/marine_sigwave

[39] Collins III, C. O. (2014). *Typhoon generated surface gravity waves measured by nomad-type buoys*. University of Miami.

[40] Özger, M., & Şen, Z. (2007). Prediction of wave parameters by using fuzzy logic approach. *Ocean Engineering*, *34*(3-4), 460–469.

[41] Van Vledder, G. P. (2013). On wind-wave misalignment, directional spreading and wave loads. In *International conference on offshore mechanics and arctic engineering* (Vol. 55393, V005T06A087). American Society of Mechanical Engineers.

[42] Chan, K., & Hong, Y. (2018). Simulation of spar type floating offshore wind turbine subjected to misaligned wind-wave loading using conservation of momentum method.

[43] Stewart, G. M., & Lackner, M. A. (2014). The impact of passive tuned mass dampers and wind–wave misalignment on offshore wind turbine loads. *Engineering Structures*, *73*, 54–61.

[44] Trumars, J. M., Jonsson, J. O., & Bergdahl, L. (2006). The effect of wind and wave misalignment on the response of a wind turbine at bockstigen. In *International conference on offshore mechanics and arctic engineering* (Vol. 47462, pp. 635–641).

[45] Bachynski, E. E., Kvittem, M. I., Luan, C., & Moan, T. (2014). Wind-wave misalignment effects on floating wind turbines: Motions and tower load effects. *Journal of Offshore Mechanics and Arctic Engineering*, *136*(4).

[46] Sun, C., & Jahangiri, V. (2019). Fatigue damage mitigation of offshore wind turbines under real wind and wave conditions. *Engineering Structures*, *178*, 472–483.

[47] W.G.S Konarasinghe, N. A., & Gunaratne, L. (2015). Comparison Of Time Domain And Frequency Domain Analysis In Forecasting Sri Lankan Share Market Returns. In *International Journal of Management and Applied Science (IJMAS)* (Vol. 1, pp. 16–18).

[48] DeepAI. (n.d.). Feature extraction. Retrieved April 27, 2020, from https://deepai.org/machine-learning-glossary-and-terms/feature-extraction

[49] Bishop, C. M. (2006). *Pattern recognition and machine learning*. springer.

[50] Seber, G. A., & Lee, A. J. (2012). *Linear regression analysis*. John Wiley & Sons.

[51] Rasmussen, C. E. (2003). Gaussian processes in machine learning. In *Summer school on machine learning* (pp. 63–71). Springer.

[52] Ng, A. (n.d.[a]). Model representation. Retrieved August 10, 2020, from https://www.coursera.org/learn/machine-learning/supplement/cRa2m/model-representation

[53] Microsoft. (n.d.). Normalización de datos. Retrieved April 12, 2020, from https://docs.microsoft.com/es-es/azure/machine-learning/algorithm-module-reference/normalize-data#:~:text=Normalization%5C%20is%5C%20a%5C%20technique%5C%20often,de%5C%20valores%5C%20ni%5C%20perder%5C%20informaci%5C%C3%5C%B3n.

[54] Zhang, K., Hutter, M., & Jin, H. (2009). A new local distance-based outlier detection approach for scattered real-world data. In *Pacific-asia conference on knowledge discovery and data mining* (pp. 813–822). Springer.

[55] Sit, H. (2019). Quick start to gaussian process regression. Retrieved June 19, 2019, from https://towardsdatascience.com/quick-start-to-gaussian-process-regression-36d838810319#:~:text=Gaussian%5C%20process%5C%20regression%5C%20(GPR)%5C%20is,uncertainty%5C%20measurements%5C%20on%5C%20the%5C%20predictions.

[56] Awad, M., & Khanna, R. (2015). Support vector regression. (pp. 67–80). doi:10.1007/978-1-4302-5990-9_4

[57] Sharp, T. (2020). An Introduction to Support Vector Regression (SVR). Retrieved August 11, 2020, from https://towardsdatascience.com/an-introduction-to-support-vector-regression-svr-a3ebc1672c2

[58] Wikipedia. (2020a). Artificial neural network. Retrieved September 1, 2020, from https://en.wikipedia.org/wiki/Artificial_neural_network#cite_note-1

[59] Ng, A. (n.d.[b]). Backpropagation algorithm. Retrieved September 11, 2020, from https://www.coursera.org/learn/machine-learning/supplement/pjdBA/backpropagation-algorithm

[60] Smolyakov, V. (2018). Neural Network Optimization Algorithms. Retrieved September 5, 2020, from https://towardsdatascience.com/neural-network-optimization-algorithms-1a44c282f61d

[61] Quesada, A. (n.d.). 5 algorithms to train a neural network. Retrieved September 5, 2020, from https://www.neuraldesigner.com/blog/5_algorithms_to_train_a_neural_network#Levenberg-Marquardt

[62] Mahjoobi, J., & Mosabbeb, E. A. (2009). Prediction of significant wave height using regressive support vector machines. *Ocean Engineering*, *36*(5), 339–347.

[63] Wikipedia. (2019). Nonlinear autoregressive exogenous model. Retrieved September 11, 2020, from https://en.wikipedia.org/wiki/Nonlinear_autoregressive_exogenous_model

[64] MathWorks. (n.d.[b]). Select data and validation for regression problem. Retrieved May 3, 2020, from https://www.mathworks.com/help/stats/select-data-and-validation-for-regression-problem.html

[65] Brownlee, J. (2019a). How to use Learning Curves to Diagnose Machine Learning Model Performance. Retrieved September 6, 2020, from https://machinelearningmastery.com/learning-curves-for-diagnosing-machine-learning-model-performance/

[66] Ng, A. (2020). Deciding what to do next revisited. Retrieved from https://www.coursera.org/learn/machine-learning/supplement/llc5g/deciding-what-to-do-next-revisited

[67] Stewart, G. M., Robertson, A., Jonkman, J., & Lackner, M. A. (2016). The creation of a comprehensive metocean data set for offshore wind turbine simulations. *Wind Energy*, *19*(6), 1151–1159.

[68] NDBC. (2018). Measurement descriptions and units. Retrieved October 14, 2019, from https://www.ndbc.noaa.gov/measdes.shtml

[69] Tukey, J. W. (1977). *Exploratory data analysis*. Reading, Mass.

[70] Ige, O. (2018). Pipeline for exploratory data analysis and data cleaning. Retrieved April 22, 2020, from https://medium.com/@oluwabukunmige/pipeline-for-exploratory-data-analysis-and-data-cleaning-6adce7ac0594

[71] Martinez, W. L., Martinez, A. R., & Solka, J. (2017). *Exploratory data analysis with matlab*. Chapman and Hall/CRC.

[72] NDBC. (n.d.). Station 46011 (LLNR 215) - SANTA MARIA - 21NM NW of Point Arguello, CA. Retrieved September 6, 2019, from https://www.ndbc.noaa.gov/station_history.php?station=46011

[73] Brownlee, J. (2019b). How to identify outliers in your data. Retrieved April 1, 2020, from https://machinelearningmastery.com/how-to-identify-outliers-in-your-data/

[74] Mahmoodi, K., & Ghassemi, H. (2018). Outlier detection in ocean wave measurements by using unsupervised data mining methods. *Polish Maritime Research*, *25*(1), 44–50.

[75] Abuzaid, A. H., Mohamed, I. B., & Hussin, A. G. (2012). Boxplot for circular variables. *Computational Statistics*, *27*(3), 381–392.

[76] Berens, P. (2009). Circstat: A matlab toolbox for circular statistics. *Journal of Statistical Software, Articles*, *31*(10), 1–21.

[77] Wikipedia. (2020b). Feature selection. Retrieved April 8, 2020, from https://en.wikipedia.org/wiki/Feature_selection#cite_note-Bermingham-prolog-

[78] Mathwork. (2020). Introduction to feature selection. Retrieved from https://es.mathworks.com/help/stats/feature-selection.html

[79] TensorFlow. (2020). Pronóstico de series de tiempo. Retrieved September 3, 2020, from https://www.tensorflow.org/tutorials/structured_data/time_series#feature_engineering

Appendix A

Introducción

Actualmente, uno de los mayores retos globales a los que se enfrenta la sociedad es combatir el cambio climático. Tal como declaró La Convención Marco de las Naciones Unidas sobre el Cambio Climático (CMNUCC) [1], los gobiernos de todo el mundo deben apostar por el desarrollo, aplicación y difusión de nuevas tecnologías que promuevan la operación y uso de energías renovables; reduciendo así la emisión de gases de efecto invernadero y mitigando por tanto el calentamiento global [2], [3].

A diferencia de la energía eólica terrestre, una tecnología ya madura [4]; la energía eólica marina surge hace dos décadas como una solución prometedora que combate los inconvenientes de las turbinas terrestres. Esta permite aprovechar los vientos más fuertes y constantes que se generan en mar abierto debido a la ausencia de accidentes geográficos que lo frenen como ocurre en tierra. El Departamento de Energía de los Estados Unidos (USDOE) afirma que los vientos en alta mar tienen la capacidad de generar más de 2.000 GW anualmente, casi el doble de la electricidad actual generada en los Estados Unidos [5].

Sin embargo, el gran problema asociado a la energía eólica marina es el coste de instalación de la turbina en alta mar, así como el coste de mantenimiento. Numerosas líneas de investigación buscan optimizar la localización y orientación hábiles para la instalación de turbinas ancladas al lecho marino.

Con el interés de reducir los costes de instalación y aumentar la capacidad productiva de energía, aflora la idea de instalar parques eólicos en aguas aún más profundas dando lugar al diseño de **las novedosas turbinas eólicas marinas flotantes**. Las turbinas eólicas flotantes expanden el área de implantación de energía eólica, aprovechando los vientos más fuertes y estables que aquellos dados en zonas cercanas a la costa, así como reduce la contaminación visual y sonora causada por las turbinas eólicas marinas cercanas a la costa y turbinas terrestres [6]. Aunque su instalación es más simple, estas lideran nuevos retos a los que hay que hacer frente: la flotabilidad de la turbina y el control estructural.

Se necesitan sistemas óptimos para el control de las palas, la torre y la plataforma flotante que permitan a la turbina soportar las cargas de viento y olas a la que está expuesta, así

como minimizar la fatiga que sufre [6].

En colaboración con los proyectos actualmente involucrados en esta reciente tecnología en creciente desarrollo, se propone este proyecto:
"Aprendizaje Automático Aplicado al Modelado de Viento y Olas".

A.1 Motivación

El Aprendizaje Automático es una disciplina de la Inteligencia Artificial cuyo potencial radica en la capacidad de procesamiento de grandes cantidades de datos y la habilitación de los computadores para aprender reglas y patrones inferidos de la gran masa de datos recabados. La necesidad de tomar decisiones óptimas, por ejemplo, decisiones sobre el diseño estructural dentro de un proyecto científico, considerando esas reglas y comportamientos observados en los datos es una tarea incapaz de ser abarcada totalmente por los humanos y los modelos tradicionalmente usados. Así, las técnicas de aprendizaje automático están revolucionando la manera en que trabajamos con los datos de los que disponemos, empezándose a aplicar cada vez más en prácticamente todos los campos de investigación.

El problema que se plantea al hablar de energía eólica es la necesidad de predecir la cantidad de energía eléctrica que se genera y por tanto la velocidad del viento de la que depende. En particular, las turbinas eólicas marinas flotantes están expuestas a vientos y condiciones oceanográficas mucho más duras. La estabilidad de estas turbinas se ve afectada significativamente más que si se encontraran ancladas al lecho marino, debido al impacto de esas cargas de olas y corrientes. Así se requieren estudios más precisos sobre la condición meteorológica y oceanográfica (metoceánica) para poder reducir los costes de mantenimiento y aplicar los resultados en un control óptimo, que maximice la producción de energía eléctrica procedente del viento a la vez que controla los sistemas de amortiguación de las nuevas oscilaciones generadas sobre la plataforma flotante; destacando cuatro tipos principales de plataformas como vemos en la figura A.1 con el objetivo de diseñar turbinas eólicas marinas flotantes fiables.

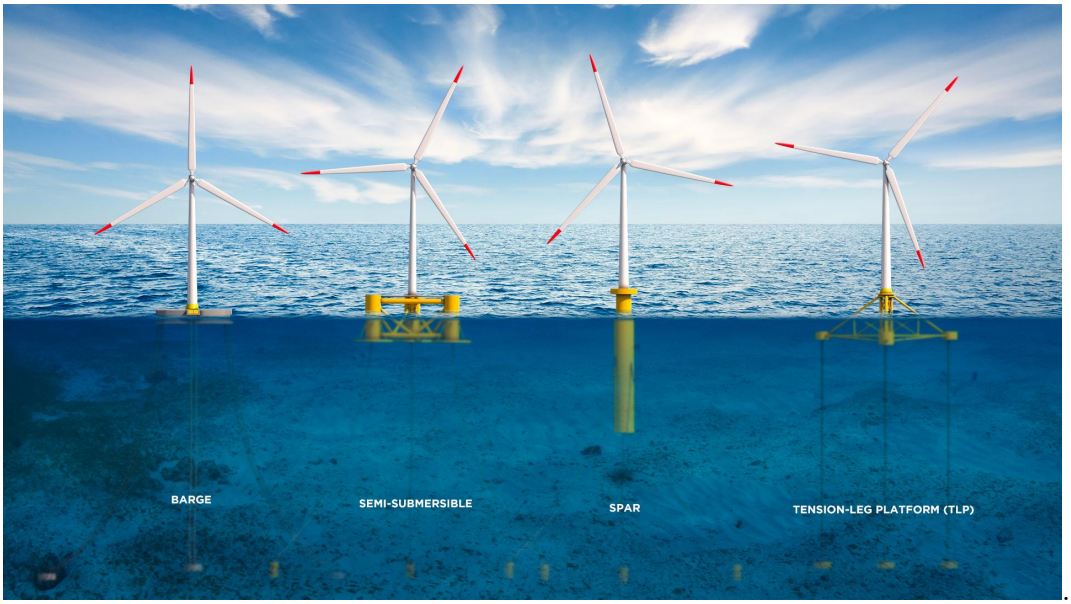


Figure A.1. *Tipos principales de plataforma para las turbinas eólicas marinas flotantes*

Disponer de modelos con buen rendimiento para la predicción de viento y olas es fundamental, sin embargo, hay pocos resultados aceptables disponibles en la literatura relacionada con las necesidades que plantea la energía marina flotante. Además, autores e investigadores involucrados en el modelado de viento y olas han venido considerando la dirección del viento como constante (basándose en las probabilidades estadísticas calculadas para una localización concreta considerada para la construcción de un parque eólico) así como la desalineación entre el viento y olas nula. Intuitivamente es razonable pensar que la desalineación es 0 ya que el viento es la principal fuente de generación de las olas. A pesar de ello, algunos investigadores han observado recientemente que la desalineación entre la dirección del viento y de las olas es significativa la mayor parte del tiempo y esta aumenta la carga total que ataca la estructura de soporte de la turbina, afectando a las turbinas eólicas marinas flotantes mucho más de lo que cualquier autor previamente pudo llegar a imaginar.

En este trabajo, se lleva a cabo un análisis de los datos del viento y de las olas y la aplicación de diferentes técnicas de Aprendizaje Automático para modelizar variables del viento y olas implicadas en el funcionamiento de las turbinas eólicas marinas flotantes. Este proyecto es además pionero en el estudio y modelado de la desalineación. Los resultados pueden llegar a ser de utilidad para proyectos involucrados en la optimización de las estrategias de control de las turbinas, así como en el aumento de la capacidad de producción de energía, logrando que la energía eólica marina sea una fuente de energía más segura y acercando a los gobiernos a la integración de parques eólicos marinos en la red eléctrica.

A.2 Objetivos

En este proyecto se llevará a cabo una investigación de la literatura y desarrollo de modelos de viento, olas y desalineación entre el viento y las olas mediante el uso de algoritmos de Aprendizaje Automático. Este estudio está orientado a ser aplicado en el ámbito de las turbinas eólicas marinas flotantes, por lo que esto se tendrá presente durante todo el proceso, por ejemplo, en la selección de la estación de donde se descargarán los datos.

En particular, los objetivos del proyecto se describen a continuación:

- Estudio de la literatura sobre el tratamiento y análisis de datos del viento y las olas.
- Estudio de resultados previos de modelado de viento, olas y en caso de existir, de desalineación, así como las técnicas usadas para ello.
- Selección, análisis y preprocesamiento de datos históricos metoceánicos.
- Modelado del viento en el dominio del tiempo y de las olas en el dominio del tiempo y la frecuencia mediante la aplicación de técnicas de Aprendizaje automático.
- Comparación de los modelos.
- Presentación de los resultados obtenidos en cada una de las fases del proyecto.

A.3 Plan de trabajo

Al inicio del proyecto se diseñó un plan de trabajo a modo de guía para su desarrollo. La imagen A.2 con dicho plan muestra cada tarea con una fecha de inicio y de finalización asociadas. Algunas de estas tareas como se puede observar son concurrentes.

Nombre de la tarea	Fecha de inicio	Fecha de finalización	Duración (días)
Trabajo final de grado	**15/09/2019**	**07/09/20**	**358**
Estado del arte - investigación	15/09/2019	15/10/2019	30
Documentación (memoria)	15/10/2019	30/10/2019	15
Parte I: Preparación de los datos	30/10/2019	30/05/2020	213
Búsqueda de bases de datos	30/10/2019	15/11/2019	16
Estudio de las variables de viento y olas	15/11/2019	30/12/2019	45
Hacer cursos en Mathworks sobre el procesamiento de datos	15/11/2019	30/12/2019	45
Diseño del modelo de datos	18/11/2019	30/12/2019	42
Documentación (memoria)	30/12/2019	10/1/2020	11
Análisis y limpieza de datos	10/1/2020	30/5/2020	141
Documentación (memoria)	10/1/2020	30/5/2020	141
Estudio de la selección de variables	30/5/2020	5/6/2020	6
Documentación (memoria)	5/6/2020	10/6/2020	5
Parte II: Modelado	10/6/2020	15/8/2020	66
Hacer cursos en Mathworks - Aprendizaje inicial	10/6/2020	15/6/2020	5
Predicción de la Velocidad del Viento	10/6/2020	10/7/2020	30
Entrenamiento y optimización de modelos	10/6/2020	10/7/2020	30
Predicción de la altura significativa de la ola	10/7/2020	10/8/2020	31
Entrenamiento y optimización de modelos	10/7/2020	10/8/2020	31
Modelado de la desalineación	10/8/2020	5/9/2020	26
Entrenamiento y optimización de los modelos	10/8/2020	5/9/2020	26
Análisis de los resultados y conclusiones	10/7/2020	7/9/2020	59
Documentación (memoria)	10/7/2020	6/9/2020	58
Subida del código y materiales usados al repositorio	3/9/2020	6/9/2020	3
Entrega del borrador	7/9/2020		
Entrega final	09/09/2020		

Figure A.2. *Plan de trabajo del proyecto*

A.4 Repositorio

El código generado durante el desarrollo del proyecto se aloja en un repositorio público de Github al que se puede acceder entrando en el siguiente enlace: https://github.com/MontseSacie/Machine_Learning_Applied_to_Wind_and_Waves_Modelling.

A.5 Estructura del proyecto

Para describir el trabajo desarrollado en cada fase del proyecto, dividimos la memoria en los siguientes capítulos:

- El capítulo 1 recoge la introducción donde se expone la motivación para la elaboración de este trabajo de investigación, los objetivos iniciales y el plan de trabajo para alcanzarlos.
- El capítulo 2 es el Estado del Arte. Este resume el contexto de investigación creado en torno a la energía eólica marina y los resultados previos sobre la predicción del viento y las olas encontrados en la literatura.
- El capítulo 3 (Materiales y métodos) es un capítulo teórico donde se explican los algoritmos y métodos usados en las fases de preparación de datos y modelado.
- El capítulo 4 describe el proceso de recopilación de datos, justificando la selección de la estación en la que se encuentra la boya que mide los datos y explicando el significado de las variables que conforman el conjunto de datos. Este capítulo junto con el 5 y el 6 conforman la parte de "Preparación o Preprocesamiento de Datos" del proyecto.
- El capítulo 5 incluye las técnicas de Análisis Exploratorio de Datos y métodos de limpieza de datos aplicados sobre nuestro conjunto de datos. El Análisis Exploratorio de Datos permite la comprensión de los mismos y la limpieza de datos se aplica para eliminar los valores faltantes o "vacíos" y anómalos.
- El capítulo 6 explica cómo se han estructurado los conjuntos de datos usados y qué variables incluye cada uno para el entrenamiento de los modelos.
- Los capítulos 7, 8 y 9 presentan los resultados del modelado de la velocidad del viento, la altura significativa de la ola y la desalineación entre el viento y las olas respectivamente. En cada capítulo se describen las diferentes técnicas de Aprendizaje Automático aplicadas y se concluye con una sección donde se compara el rendimiento de cada algoritmo para decidir cuál es el mejor en ese caso.
- El capítulo 10 presenta las conclusiones del proyecto y los trabajos futuros.

A continuación de la bibliografía, se incluyen como apéndice las siguientes partes traducidas al español de la memoria:

A. Introducción

B. Conclusiones y Trabajo Futuro

Appendix B

Conclusiones y Trabajo Futuro

B.1 Conclusiones

En este proyecto se ha alcanzado el principal objetivo de modelar las cargas del viento y olas de naturaleza estocástica que impactan significativamente sobre las turbinas eólicas marinas flotantes; desarrollando cada una de las fases propias de un proceso de Aprendizaje Automático desde cero.

Podemos afirmar que las fases de recopilación de datos y preprocesamiento de datos han sido fundamentales para conseguir modelos más precisos mediante la aplicación de aprendizaje automático. Además, la limpieza y análisis de datos nos lleva a seleccionar de forma más efectiva los predictores, mejorando enormemente el rendimiento de los modelos, así como asegurándonos de su efectividad para realizar predicciones realistas.

Se exploraron diferentes técnicas de aprendizaje automático para ser aplicadas en la predicción de la velocidad del viento una hora en el futuro y se comparó su rendimiento. Demostramos que la predicción de la velocidad del viento es posible aplicando exclusivamente técnicas de aprendizaje automático en lugar de usar los tradicionales modelos físicos y estadísticos a la vez que obtenemos errores predictivos aceptables. Para el caso concreto de datos recopilados en este proyecto de la estación de Santa Maria (CA, EEUU) seleccionada como localización potencial para un parque eólico, el modelo de red neuronal NARX supera al resto de algoritmos en cuanto a rendimiento.

En la misma línea, se han usado técnicas de aprendizaje automático para la predicción de la altura significativa de la ola en el dominio de la frecuencia. Los modelos entrenados con variables del viento y el tiempo como entrada han dado como resultado errores similares a los encontrados en la literatura y el modelo GPR resultó ser el más preciso en el contexto predictivo concreto de este trabajo.

En la literatura no hemos encontrado modelos para la predicción de la desalineación entre el viento y olas. Por ello, este proyecto es pionero en dicha tarea. Se han aplicado técnicas

de aprendizaje automático para la predicción de la desalineación para la próxima hora en el futuro. Entre todos los modelos entrenados, el algoritmo NARX se ha mostrado como el más eficiente de todos.

Finalmente, podemos remarcar tras la experiencia del desarrollo de este proyecto la necesidad de dedicar tiempo a la correcta evaluación de los modelos mediante técnicas como los gráficos de Curvas de Aprendizaje para tomar mejores decisiones de optimización; ya que sería imposible probar todos los posibles modelos y configuraciones de parámetros. Dada su complejidad, con un mayor conocimiento de optimización se pueden obtener modelos más precisos.

Mas allá de los resultados dados por los modelos, aquí se ha desarrollado un proyecto de Aprendizaje Automático siguiendo una metodología adecuada y centrándonos en el aprendizaje durante el proceso.

B.2 Trabajo Futuro

Como continuación de este proyecto, sería interesante empezar entrenando los modelos obtenidos aquí con datos de otras localizaciones y comparar los resultados con los de este trabajo. Esta línea permitiría estudiar en qué medida influye el sitio concreto de localización del parque eólico sobre el rendimiento de los modelos.

Por otro lado, los trabajos futuros centrados en la optimización de los modelos aquí presentados o en la mejora de los errores mediante nuevos modelos pueden probar a:

- Recopilar variables de datos no consideradas en este trabajo y aplicar técnicas avanzadas de selección de variables para crear nuevos conjuntos de datos.
- Aplicar técnicas de aprendizaje automático para el modelado del viento y las olas por estación. Los fenómenos del viento y el oleaje pueden estar ligados a la estación del año. Así entrenar modelos para cada estación de forma separada puede mejorar las predicciones realizadas.
- Explorar algoritmos de aprendizaje profundo y otros algoritmos de aprendizaje automático que no se han aplicado en este proyecto.
- Estudiar detenidamente el coste computacional de ejecución de los algoritmos utilizados tanto en la fase de preparación de datos como en la de modelado; intentando optimizar el tiempo de ejecución de los mismos, además del error predictivo como se hizo en este proyecto.
- La creación de modelos híbridos puede llevar a la integración de estos modelos en otros sistemas predictivos más sofisticados para la energía eléctrica o fatiga causada en las turbinas.

Para concluir, los modelos de desalineación obtenidos aquí son sin duda una puerta abierta a muchos otros proyectos de diferentes áreas de la comunidad investigativa de la energía

eólica marina, que junta busca la integración de los parques eólicos flotantes en nuestra sociedad.