Regression Analysis

Dimitrios Kemos

September 2018

School of Mathematics,
Cardiff University

A dissertation submitted in partial fulfilment of the
requirements for MSc (in Data Science and Analytics) by taught programme.

CANDIDATE'S ID NUMBER	1753774
CANDIDATE'S SURNAME	**Please circle as appropriate** [✓ **Mr**] / **Miss / Ms/ Mrs / Rev / Dr / Other ....**
CANDIDATE'S FULL FORENAMES	DIMITRIOS KEMOS

DECLARATION

This work has not previously been accepted in substance for any degree and is not concurrently submitted in candidature for any degree.

Signed ....[illegible] DIMITRIOS KEMOS (candidate) Date ...27/9/2018......

STATEMENT 1

This dissertation is being submitted in partial fulfilment of the requirements for the degree of ....MSc.................(insert MA, MSc,MBA, etc, as appropriate)

Signed ....[illegible] DIMITRIOS KEMOS... (candidate) Date ..27/9/2018.........

STATEMENT 2

This dissertation is the result of my own independent work/investigation, except where otherwise stated. Other sources are acknowledged by footnotes giving explicit references. A Bibliography is appended.

Signed ....[illegible] DIMITRIOS KEMOS.... (candidate) Date ..27/9/2018.....

STATEMENT 3 –

I hereby give consent for my dissertation, if accepted, to be available for photocopying and for public viewing, and for the title and summary to be made available to outside organisations.

Signed ....[illegible] DIMITRIOS KEMOS......... (candidate) Date ...27/9/2018......

STATEMENT 4 - BAR ON ACCESS APPROVED

I hereby give consent for my dissertation, if accepted, to be available for photocopying and for public viewing **after expiry of a bar on access approved by the Graduate Development Committee.**

Signed ..[illegible]..... DIMITRIOS KEMOS......... (candidate) Date ...27/9/2018.......

TABLE OF CONTENTS

Executive Summary

A raise in energy consumption can be observed worldwide a phenomenon mainly caused by the continuous population growth. So, an accurate energy forecasting technique is essential for the accurate investment planning of energy production. The main limitation faced while trying to implement such a technique is the lack of precise and sufficient energy consumption measurements. A major factor that helped to overcome this particular limitation is the development and evolution of smart meters. High resolution energy consumption readings produced by smart meters can provide valuable insight on general consumption patterns and lifestyles of each customer. Given these advances, the need to efficiently manage massive smart meter data in order to improve the efficiency and sustainability of the power industry and particularly energy consumption has become an important issue worldwide.

The dataset used for the purpose of this study originated from the Low Carbon project and contains 5,567 London households that took part in the UK power networks during the period of November 2011 and February 2014. The energy consumption readings contained in the dataset where measured in half hour intervals and ACRON geodemographically information is also included. Additionally, a data set containing weather reading for the same period was included in order to observe whether the model's performance can be upgraded using such additional information.

This study implements machine learning models in order to effectively forecast the energy consumption on households in London, the models will be tested using various forecasting techniques and evaluated on their accuracy and computational complexity. Furthermore, weather information will be considered in order to test whether this additional information can improve the implemented models. The models were implemented on three different forecasting scenarios. On the first one, which serves as the baseline model, a one day ahead prediction of energy consumption was forecasted using the past 28 days as a training set. On the second stage models produced forecasts using a rolling window approach and on the third and last stage, again a rolling forecast technique was used but weather information was added as predictors.

The evaluation of the models was implemented using various accuracy measures as well as the compilation time. To further evaluate the effectiveness of the models, statistical hypothesis testing procedures will be also used to evaluate the model's performance and solidify the validity of the results. Another issue that this study tackles is data transformation techniques and their ability to improve forecasting accuracy, three data transformation techniques where implemented and the results were monitored in order to end up eventually with the most effective. Additionally, some data

exploration techniques where applied in order to discover hidden patterns and further understand the data used for this study.

The results obtained from this study recommend the use of K-Nearest Neighbours algorithm for modelling the energy consumption on London households. The aforementioned algorithm not only presented the best results on accuracy measures and computational complexity, it was also the one that could effectively use the weather information to further improve its score.

Acknowledgments

I would like to thank and acknowledge the following important people how have supported me not only during the period of this project, but throughout my Master's degree.

Firstly, I would like to thank my Cardiff University supervisor Mr. David Marshall for his continuous support and guidance throughout the whole period of the project.

I would like to also thank Centrica and my industrial supervisor Ms. Laura Shemilt for the support and the industrial insight she provided.

And finally, I would like to thank my family and all my close friends. Your continuous encouragement and belief in me helped me focus on what has been a hugely rewarding and enriching experience.

List of Figures

List of Tables

List of Acronyms and Abbreviations

Mean Absolut Error	MAE
Not a Number	NaN
Symmetric Mean Absolute Percentage Error	sMAPE
Cross Validation	CV
Kernel Ridge Regression	KRR
Long Short-Term Memory	LSTM
Multilayer Perceptron	MLP
Neural Network	NN
Quantile Regression Forests	QRF
Support Vector Regression	SVR
The K-Nearest Neighbor Regression	KNN
Computational Complexity	CC
Mean Absolute Scaled Error	MASE
Autoregressive	AR
Moving Average	MA
Autoregressive Moving Average	ARMA
Autoregressive Integrated Moving Average	ARIMA
Seasonal Autoregressive Integrated Moving Average	SARIMA
Vector Autoregressive	VAR
Seasonal Autoregressive	SAR
Seasonal Moving Average	SMA
Mean Squared Error	MSE
Diebold-Mariano	DM
Threshold Autoregressive	TAR
Autoregressive Conditional Heteroscedastic	ARCH

Abstract

The rapid advance in measuring sensor technologies and particularly smart meters has enabled the massive growth of energy data sets both in quality and quantity. By combining these data with the constant evolution of machine learning algorithms, energy forecasting is a field that can benefit from those progressions. The data used for the purpose of this project consist of smart meter energy consumption readings obtained from 5,567 London households. This study implements some popular learning algorithms that can effectively handle non-linear relationships and evaluated their forecasting performance by measuring their sMAPE, MAE and Computational Complexity scores. Also, different transformation techniques where implemented to test their effectiveness on forecasting learning algorithms. The K-Nearest Neighbours Regression algorithm was found the most effective among the compiled algorithms, and the one able to make the most effective use of weather information data added in the later stages of the analysis, with 6.56% sMAPE 0.13 MAE and 350ms CC. To conclude, this study proposes the implementation of KNR when modelling energy consumption with smart meters in London.

1 Introduction

Worldwide energy consumption is rising fast and there are many causes for this phenomenon. The constant increase in human population, the continuous pressures for better living standards and the need to sustain positive economic rates are some of the factors that cause this raise. Given this fact, a sound forecasting technique is essential for the accurate investment planning of energy production. The main limitation that researchers faced while trying to create such models was the lack of precise and sufficient energy consumption measurements. But not long after the lunch of smart meters this limitation ceased to exist.

Smart meters have been deployed all over the world during the past decade. The number of smart meters installed in the UK, US and China has reached 2.9 million (Department for Business Energy and Industrial Strategy, 2017), 70 million (US Energy Information Administrator , 2017), and 96 million, respectively by the end of 2016. This dynamic introduction to smart meters along with the improvements in Information Technology and communication industries eliminated the restriction of precise consumption measurement for each customer, making adaptive billing mechanisms to work more efficiently and financially motivate consumers to change their consumption to off-peak (Rashed Mohassel, 2014). However, a smart meter's function is not limited to billing. High-resolution, energy consumption readings provide insight on general consumption patterns and lifestyles of customers (Wang *et al.*, 2018). In the interim, numerous nations around the world are continuously advancing towards the deregulation of power industry, especially on the delivery side. Increasingly more participators, including retailers, consumers, and aggregators, are involved in making the retail market more prosperous, active and competitive (Yang *et al.*, 2017). So, the need to efficiently manage massive smart meter data in order to improve the efficiency and sustainability of the power industry and particularly energy consumption has become an important issue worldwide.

Although the technology and the design of the smart meters depends on the company manufacturing them a mutual process exists for data collection, communication, and data analysis. The general process of smart metering can be described in Figure 1, although there are some variations according to deployment across different countries (Alahakoon and Yu, 2016). The main purpose of a smart meter is to gather data and transfer them via a local area network (LAN) to a prespecified data collection point. The observations transferred are energy consumption data measured mainly in kilowatt hours (kWh), the reading and the transmission of the data is set to regular intervals. Depending on the technology and the settings of the smart meter the collection of the data can be set from 1 hour to 15 minutes.

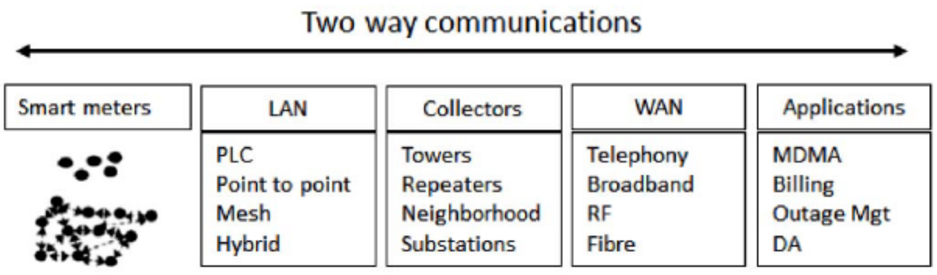

Figure 1: Smart Metering Process (Ilic, Karnouskos and Goncalves da Silva, 2012)

Sensor-based approaches to energy forecasting depend on readings from sensors or smart meters often depend on contextual information, for example, meteorological data or work routines to deduce future energy behaviour. Usually, the energy consumption is used as input in a machine learning algorithm along with historical data such as temperature, day of week and time of day, and that model learns from them and therefore can predict future energy consumption. Evidence exists, that the accuracy of these sensor-based model approaches is similar or many time superior to traditional approaches based on models using the properties of the building (Fumo, 2014).

The massive amount of data provided by smart meters, can be beneficial for governments because they can provide long term energy consumption forecasting which by itself is the basis for energy investment planning. Underestimating consumption may cause high operation costs whereas overestimation would lead to superfluous idle capacity which means wasted financial recourses (Kaytez *et al.*, 2015). Therefore, creating a framework that can produce accurate forecasting results is of high importance in order to avoid such costly mistakes.

1.1 Time Series

Time series forecasting can be successfully used in a variety of practical domains, one of them being energy consumption forecasting. A time series can be defined as a sequence of observations that measure a specific phenomenon over a recorded period of time (see Figure 1). These observations are commonly available at discrete, equal time intervals (Amadou Boukary and Rivest, 2016). A purpose of time series is that it can be applied in forecasting models which make use of the observations from a time series available at time t in order to predict its value at time $t + l$; where l represents the forecasting horizon or lead time (C. M. Douglas, 2016).

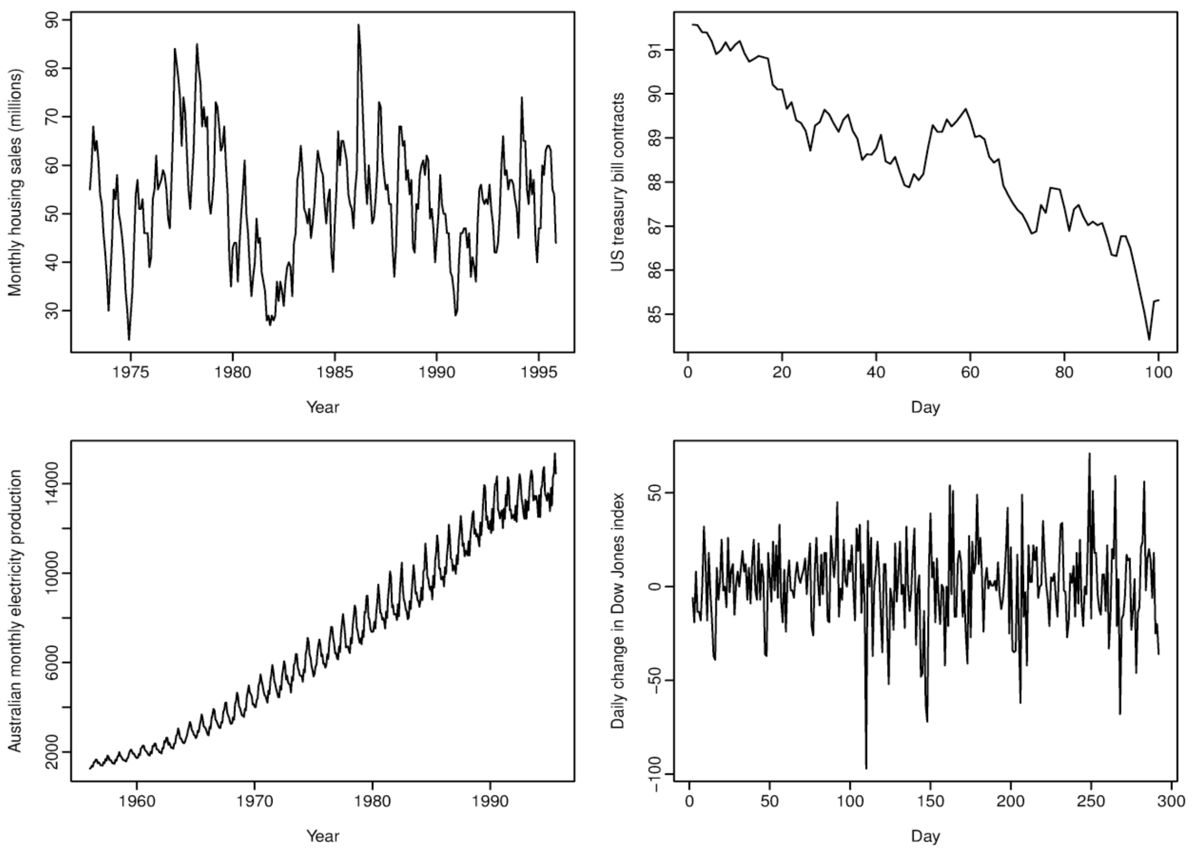

Figure 2: Demonstration of different time series plots. Source: (Hyndman and Athanasopoulos, 2014)

Forecasting can provide valuable insight for decision making, economic and business planning, inventory management, as well as the control and optimization of industrial processes. As an example, applications and studies which concern total social financing (TSF) in economy and finance are forecasting gross domestic product (GDP), forecasting of sales and prediction of stick markets (Amadou Boukary and Rivest, 2016). In science time series forecasting has been frequently applied for weather and earthquake forecasting, energy consumption forecasting. In the field of management one can find forecasting studied for room bookings and emergency calls and in the medical field, forecasting for various diseases and so on (Amadou Boukary and Rivest, 2016). The procedure a forecasting model follows in order to make a prediction is straight forward, the forecasted value $\widehat{y_t}$ for a time interval t is being computed using antecedent values $(y_{t-1}, y_{t-2}, y_{t-3}, \dots)$. A plethora of forecasting algorithms using machine learning have been proposed, with the most popular, in relative literature, being the standard statistical forecasting models followed by machine learning artificial neural network models.

Many authors studied the electric consumption in order to get a better understanding which are the demand drivers and which are the fundamental pillars in the development of a forecasting model.

De Martino Jannuzzi and Schipper (1991) conducted an analysis of the consumption of electrical energy for a residential sector in Brazil and they concluded that the demand was faster than the income. Harris (1993) studied the existence of a relationship between electricity consumption and some potentially relevant variables such as weather, price and consumer income. Using a 30 year energy consumption dataset from south east USA concluded that there is a high seasonality in electricity demand. Ranjan and Jain (1992) conducted a study regarding the consumption pattern of electrical energy in India and more specifically in Delhi, for the period 1984-1993 as a function of population and weather and they developed multiple linear regression models of energy consumption for different seasons.

1.2 Thesis Statement

The research objectives of this study are as follows:

1) Implementation of Machine Learning models to forecast energy consumption readings produced by smart meters. Test the algorithms behaviour with different forecasting techniques and their forecasting accuracy when supplementary weather data are inserted in the forecasting process.
2) Statistically compare the implemented algorithms to gain solid insight about the algorithm's performance.
3) The creation of a framework that specifies all the stages needed to select the best machine learning algorithm for a given problem.

2 Background and Literature Review

In this section the research literature review will be presented. The main areas that this study is focused on are two. The first one is forecasting and the second one is machine learning. So, the following sections are divided accordingly.

2.1 Forecasting

Forecasting involves using past information in order to predict the future. The core process that forecasting methods or algorithms use is making use of past observations in order to predict possible future trends or outcomes of the same data or time series.

In the field of statistics, the main challenge of forecasting is to provide insight in a variety of decision-making procedures. Generally, the field of forecasting can be divided into two categories, the qualitative approaches and the quantitative approaches. The qualitative forecasting methods is an estimation technique that considers judgement from experts, rather than numerical analysis. This type of forecasting is simply based on decisions made from employees with high experience on a specific field, commercial knowledge and any other relevant sources of information. A popular method used for qualitative forecasting is the Delphi technique (Ranjan and Jain, 1992). An appropriate situation that requires qualitative techniques to be used is when there are no data available for a problem. On the other hand, quantitative techniques make use of data in order to make a prediction. Quantitative forecasting can be divided into 2 sub-categories which are, univariate and multivariate. In univariate forecasting techniques the process of forecasting is based on past data of a single time series whereas in multivariate forecasting the forecast of a measurement can only be accurate if more than one predictor variables are included in the model. Furthermore, multivariate forecasting models are mainly used in business and environmental studies where data are related with several variables (Chatfield, 2016).

2.2 Statistical Forecasting Models

This section substantiates some core univariate and multivariate models. In field of forecasting linear statistical models prevail, the most popular of them being the Autoregressive (AR), the Moving Average (MA), the Autoregressive Moving Average (ARMA), the Autoregressive Integrated Moving Average (ARIMA) and the Seasonal Autoregressive Integrated Moving Average (SARIMA).

2.2.1 Autoregressive Model (AR)

An AR is a model where certain output variable y_t is forecasted using its own lagged values $y_{t-1}, \dots, y_{t-p}$. The term *autoregression* in the model's name indicates that this is a regression of the variable itself (Hyndman and Athanasopoulos, 2014). Thus, an AR model can be defined by the equation:

$$y_t = c + \varphi_1 y_{t-1} + \varphi_2 y_{t-2} + \dots + \varphi_p y_{t-p} + \varepsilon_t \ , \tag{1}$$

where ε_t represents the white noise. This model can be referred as AR*(p)*, an autoregressive model of order p. The p is a parameter that determines how may past observations will be utilized by the model in order to forecast the current value. If more than one predictor variables are required by in order to produce accurate predictions then the multivariate autoregressive model, also known as vector autoregressive (VAR), can be considered. The VAR model can be defined as:

$$y_t = \sum_{k=1}^{p} A_k y_{t-k} + \varepsilon_t \ . \tag{2}$$

In Equation (2) $y_n = [y_n(1), y_n(2), \dots, y_n(d)]$ is the nth sample of a d-dimensional time series and every *A(i)* is a d-by-d matrix of coefficients (weights) and $e_n = [e_n(1), e_n(2)], \dots, e_n(d)]$ is the additive Gaussian noise with zero mean and variance R (Karl Friston, 2006).

2.2.2 Moving Average (MA)

In contrast with AR models that use past values in order to forecast a variable, a MA model makes use of previous forecast errors to formulate a model similar to regression (Hyndman and Athanasopoulos, 2014). The MA model can be defined as:

$$y_t = c + e_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \dots + \theta_q \varepsilon_{t-q} \ , \tag{3}$$

where ε_t is white noise. This model can be referred as MA*(p).* Similar to the AR model, in case we need to forecast using multiple predictors the univariate model can be transformed to a multivariate one as following:

$$y_t = \sum_{k=0}^{q} B_k x_{t-k} \ . \tag{4}$$

In Equation (4) x_t represents an exogenous N-dimension time series and B_k a number of M-by-N matrices containing parameters.

2.2.3 Autoregressive Moving Average (ARMA)

An ARMA model is a combination of the two models described before, AR and MA, and it is one of the most popular forecasting techniques. In was introduced by Whitle, (1951) but only until it was descried by G. Box and G. Jenkins, (1976) it became popular. An ARMA models describes an order of *(p,q)* as ARMA*(p,q)* and can be defined as following:

$$y_t = a_1 y_{t-1} + \dots + a_p y_{t-q} + \varepsilon_t + \beta_1 \varepsilon_{t-1} + \dots + \beta_q \varepsilon_{t-q} \ , \qquad (5)$$

y_t represents the original time series and the ε_t is the random values of white noise which are assumed to follow a Gaussian distribution. In case there is a need for a multivariate model to be used the equivalent to ARMA is called vector-auto regressive average (VARMA), which can be described by the following equation:

$$y_t = \sum_{k=1}^{p} A_k y_{t-k} + \sum_{k=0}^{q} B_k x_{t-k} \ . \qquad (6)$$

In Equation (6) y_t is the forecasted output, y_{t-k} and x_{t-k} are the past inputs and the past outputs exogenous variables accordingly. Also, A_k represents an M-by-M matrix and B_k an M-by-N matrix (Amadou Boukary and Rivest, 2016).

2.2.4 Autoregressive Integrated Moving Average (ARIMA)

All the forecasting techniques described until now are effective only when used on a stationary time series (Zhang *et al.*, 2013). A time series can be defined as stationary if the observations it contains do not depend on the time that this series was measured and no trend can be observed (Kwiatkowski *et al.*, 1992). Non-stationarity is present in the majority of the time series, so in order to fit all these stationary models described before it is necessary to remove this non-stationary source of variation (Chatfield, 2016). The solution to this limitation was given by G. Box and G. Jenkins, (1976) which is the ARIMA model. The technique that ARIMA models use in order to face the problem of non-stationarity is a process called differencing (Murphy, 1989; Chatfield, 2016). The process of differencing is a simple subtraction of the current observation from the previous one. ARIMA *(p,d,q)* models and can be described as:

$$\acute{y}_t = \nabla^k y_t = a_1 \acute{y}_{t-1} + \dots + a_p \acute{y}_{t-p} + \varepsilon_t + \beta_1 \varepsilon_{\tau-1} + \dots + \beta_q \varepsilon_{\tau-q} \ . \qquad (7)$$

The parameters p,d and q contained in the general form describe the autoregressive part, the degree of differencing used, and the order of moving average past accordingly (Amadou Boukary and Rivest, 2016). ARIMA's ability to successfully handle non-stationarity is the main factor that made ARIMA one of the most widely used approaches in the field of time series forecasting (Zhang *et al*., 2013).

2.2.5 Seasonal ARIMA (SARIMA)

A SARIMA is and extension model of ARIMA and it is formed by including seasonal terms to the original ARIMA model (Chatfield, 2016). It can be written as: ARIMA (p,d,q) X $(P,D,Q)m$ where the first part of the equation is the non-seasonal part and the second the seasonal one. Also, P is the number of seasonal autoregressive (SAR) terms, D the number of seasonal differences and Q the number of seasonal moving average (SMA) terms (Amadou Boukary and Rivest, 2016).

2.3 Machine Learning

Machine learning is responsible for teaching computers to learn from experience, just like humans do. Machine learning models gained popularity during the last decade as serious competitors to classic statistical models in the field of forecasting, with the former ones having an advantage due to the fact that they include models that do not require prior assumptions about the underlying structure of the data (Zhang, Patuwo and Hu, 2001; Zhang, 2003). The research around the field started in the eighties with the development of the neural network model. Afterwards, research extended towards other models, such as support vector machines, decision trees, and others, that are collectively called machine learning models (Friedman, Hastie and Robert, 2007; Ethem, 2014). A portion of these models have roots from early statistics literature. The advances observed in the field of machine learning during the past few years, both in the theoretical understanding and documentation of the models and the plethora of new model's development is noticeable. That growth made machine learning an interesting crossroad for one to explore. Machine learning is divided into three main categories supervised, unsupervised and reinforcement learning. In the following section a brief discussion will be conducted about each one of these categories and their core methods.

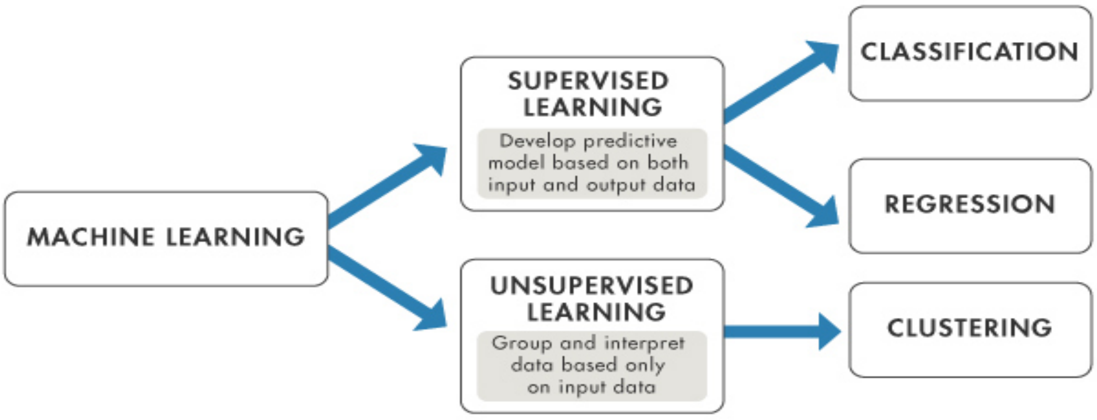
MACHINE LEARNING
SUPERVISED LEARNING
Develop predictive model based on both input and output data
UNSUPERVISED LEARNING
Group and interpret data based only on input data
CLASSIFICATION
REGRESSION
CLUSTERING

Figure 3: Different Techniques in the field of Machine Learning. Source: (MathWorks, 2012)

2.3.1 Supervised Learning

Supervised learning is considered the most popular of the machine learning techniques. Its general objective is to learn a mapping from inputs x to outputs y , given a labelled set of input-output pairs $D = \{(x_i, y_i)\}_{i=1}^{N}$. Here D represents the training set and N the training examples (P. Murphy, 2012). Supervised models can be further divided into two distinct type of algorithms described below.

Classification is the process of predicting the class of some given data points. Classification algorithms can handle data which are divided into classes and their goal is to create a mapping function *(f)* to correctly classify inputs x to outputs y. The output y depends on the number of classes contained in the data and can be described as $y \in \{1, \dots, C\}$, with C being the number of the classes (P. Murphy, 2012). If the number of classes is $C = 2$ it's a binary classification problem, otherwise if $C > 2$ it's a multiclass classification problem. Classification is likely the most extensively used method of machine learning, and it has been applied in order to remedy interesting and often challenging real-world problems such as email spam filtering, and face recognition.

Regression techniques are similar to classification differencing only in the output variable, which in regression should be continuous. which in regression should be continuous. So, a regression problem requires the prediction of a quantity and the input feed to the model should be real valued or discrete variables. Regression models usually use a single real-valued input $x_i \in \mathbb{R}$ along with a single real valued response $y_i \in \mathbb{R}$ (P. Murphy, 2012). When a problem requires multiple input variables then multiple regression are available to deal with such problems. Regression techniques are often used in real-world problems like predicting temperature at any location inside a building using weather data, time, door sensors, etc (P. Murphy, 2012). A detailed discussion about supervised models is being conducted later in this study.

2.3.2 Unsupervised Learning

Unsupervised learning is a type of machine learning algorithm used is to discover patterns on the dataset using just the output data without any inputs, it is often called knowledge discovery. The most common technique of unsupervised learning is cluster analysis, which is mainly used for explanatory analysis to find hidden patterns or invisible groupings in the data.

A popular clustering algorithm which will be implemented for the needs of this study is k-means clustering proposed by MacQueen, (1967). In k-means every cluster is represented by the mean by the mean of the observations contained in the cluster. In detail, an object n is partitioned into k clusters so that the observations contained in the same cluster present high similarity whereas the observations not contained in the same cluster present low similarity (Yadav and Sharma, 2013). The aforementioned similarity is measured using the Euclidean Distance defined as:

$$d(i,j)\sqrt{\sum_{i=0}^{n}(x_i - y_i)^2}\ , \qquad (7)$$

where x_i and y_i are two n-dimensional data objects.

Another popular unsupervised learning technique, particularly useful for organizing and summarizing data, utilized for the purposes of this study is Principal Components Analysis (PCA) (Jolliffe, 1986; Costa and Ceser, 2009). The mathematical formulation of PCA follows. We can define matrix X as the data used, Q each column contained in the data set and i each row which represents a respective feature $\overrightarrow{X_\imath}$. Furthermore, each observation measured for the j th object can be represented as $\overrightarrow{X_\jmath}$. An important perspective that need to be taken into consideration while implementing PCA is that it represents a linear transformation expressed by the following equation (Gewers *et al.*, 2018):

$$Y = W \times X\ , \qquad (8)$$

where W represents the transformation matrix and X the original measurements. So, in order to implement PCA, given that the original data is available, the transformation matrix W is needed to apply the Equation 8. The descent of the W matrix can be descried as following. The i th element of the mean vector $\vec{\mu}_X = (\mu_{\chi 1}, \ldots, \mu_{\chi N})^2$, with dimension $N \times 1$ can be defied as:

$$\mu X_i = \frac{1}{Q}\sum_{j=1}^{Q} X_{ij}\ , \qquad (9)$$

where X_{ij} represents the observation of the i th element taken from the j th object. Furthermore, the elements are the brought to the coordinates origin by subtracting the respective means as following (Gewers *et al.*, 2018):

$$\hat{X}_i = X_i - \mu X_i \ . \tag{10}$$

After the creation of the transformation matrix the non-negative eigenvalues λ_i have to be calculated, sorted in decreasing order, and respective eigenvectors $\vec{v_\imath}$ (Gewers *et al.*, 2018). The sorted eigenvectors can be described as:

$$W = \begin{bmatrix} \overrightarrow{v_1} \\ \cdot \\ \cdot \\ \cdot \\ \overrightarrow{v_N} \end{bmatrix} . \tag{11}$$

Finally, in order to calculate the PCA projection of an observation I can be obtained using the following Equation:

$$\vec{Y_\imath} = W \times \vec{X_\imath} \ , \tag{12}$$

where $\vec{Y_\imath}$ and $\vec{X_\imath}$ are the feature vector and the projected space (Gewers *et al.*, 2018).

2.4 Applications to Time Series

Although the majority of the available research in time series and forecasting are implemented using a variety of models in order to achieve more reliable forecasts, a few studies where found containing the AR model as a standalone technique. Fadhilah *et al.*, (2009) carried out a study and concluded that the pure autoregressive model with an order 2, or AR(2) is the appropriate model for forecasting the Malaysian peak daily load using 3 days ahead prediction. Hill and Infield, (1995) compared the forecasts of electricity load of Shetland Islands produced by operators versus the ones obtained by an AR model. They concluded that the AR forecasts provide increased overall system performance. Many researchers have exploited ARIMA's ability to successfully handle non-stationary data. In detail, (Zhou *et al.*, 2006) implemented an ARIMA model to forecast the electricity price and then included an error correction technique in order to improve performance. This study concluded that the proposed ARIMA approach is highly effective for electricity price forecasting. Following similar context Contreras *et al.*, (2003) proposed a ARIMA methodology to predict next day electricity prices on Spain and California markets.

All the statistical time series forecasting models described all strictly linear models. Although their simplicity and stable performance for specific linear tasks made them replicable, their disadvantages became conspicuous due to their incapability to capture non-linear relationships(Zhang, 2003). This limitation became obvious in early 1980s when the could not adapt to many real applications due to their linear nature (Bontempi, Taieb and Borgne, 2003). To overcome the linearity assumption of the traditional time series, new models where introduced. Some of these models are the Bilinear Model, the Threshold Autoregressive (TAR) model and the Autoregressive Conditional Heteroscedastic (ARCH) model (Zhang, 2003). Kim, (2011) implemented an ARCH model which outperformed an ARIMA in the task of forecasting internet traffic. Although this proposed techniques indicated an improvement towards non-linear problems, their ability to generalize is limited since they are developed to identify specific non-linear relationships (Zhang, 2003). Therefore, the progress of Machine Learning algorithms and Neural Networks (NN) can be considered a capable alternative to the traditional time series forecasting models since they include a plethora of algorithms capable of capturing non-linear relationships.

Machine Learning can be applied on forecasting time series producing better results compared to statistical forecasting models (Zhang, 2003). This performance difference can be observed due the ability of NN to explain non-linear patterns in the data-set and their capability to deal with noisy time series (Zhang and Patuwo, 1998). Nonetheless, NN present their own limitations. Feed Forward Neural Networks are not able to retain information from past observations. Long Short-Term Memory networks where created to overcome the issue of the vanishing gradient but their limitation is their computational complexity which makes the training process slow. More about this phenomenon will be explained later in this paper.

3 Model Selection Framework

On this section the proposed framework that a researcher has to follow in order to obtain the optimal model for the given dataset will be descried.

At the start of the framework one can observe the raw data, also mentioned in the field of machine learning as primary data, which is simply a data set obtained automatically by a machine, a computer program or manually. Simply, raw data is data that have not yet being processed for further use. A discussion about the raw data used for this study is conducted on Section 4. Moving to the pre-processing stage, this procedure includes the formatting of the raw data to suit the needs of the conducted analysis, and the researcher itself. This is an iterative process meaning that it has be done multiple times until the data set is ready to be used on machine learning algorithms. Detailed information about the pre-processing techniques used in this study can be found on Section 4.1.

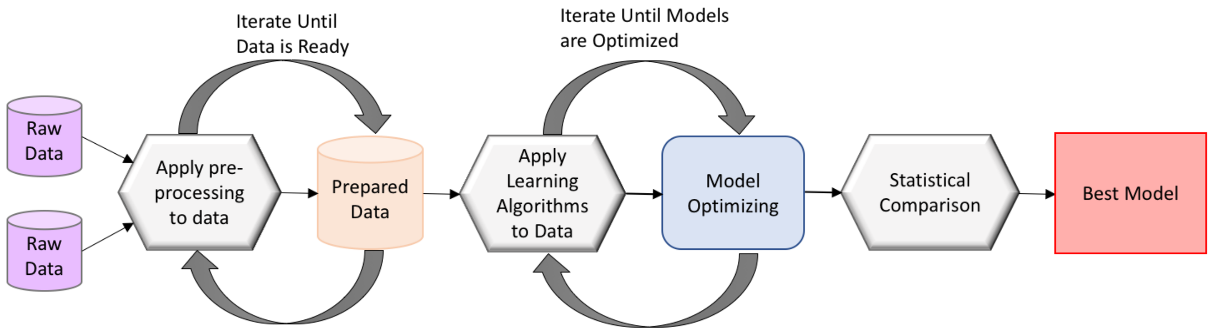
Iterate Until Data is Ready
Iterate Until Models are Optimized
Raw Data
Raw Data
Apply pre-processing to data
Prepared Data
Apply Learning Algorithms to Data
Model Optimizing
Statistical Comparison
Best Model

Figure 4: Model Selection Framework

The next stage of the proposed framework refers to algorithm application and model optimizing. During this stage the researcher has to select the algorithms that best describes the problem to be solved, on Section 5 the model selection of this study is being explained and the models are analysed from a statistical perspective. Furthermore, when the model selection has been made the algorithms have to be optimized in order to get the optimal results. This model optimization usually conducted using hyperparameter tuning algorithms that test the model's performance using different settings and returns the results to the researcher, the model optimization conducted for this study can be found on Section 6 along with a detailed explanation of each models hyperparameter tuning. Model optimization is also an iterative process and has to be implemented until the optimal results are obtained.

Moving to the last step of the framework, a statistical comparison has to be conducted. At this stage it is crucial that the appropriate statistical This will enable the researcher to gain a better understanding of the actual performance of the implemented models. Using statistical comparison will also point out whether a difference in the models' performance is significant or not. In this study Section 7.4 implements a statistical hypothesis test, suitable for forecasting models and compares the predictions of the learning algorithms to conclude if there is a significant difference between the implemented models.

4 Smart Meter Dataset Description

This section starts with a general discussion and review of different open energy consumption data sets, and continues with the review and the data exploration of the data set that was used for this study.

There are many limitations, such as privacy and security, which discourage power provides to release smart meter energy consumption data for public viewing and use. This is a challenging issue for any researcher interested to conduct any kind of study concerning smart meters analytics and its applications. Despite these limitations there are several datasets mainly containing household measurements that can be found publicly available (Wang *et al.*, 2018). Some of the most popular datasets containing smart meter observations are descried below:

- **Customer Behaviour Trials**: This smart metering project was started by the Commission for Energy Regulation (CER) which is the regulator for electricity and natural gas sectors in Ireland. The main scope of the project was to conduct customer behaviour trials in order to shape energy consumption patterns across a variety of demographics, lifestyles and home sizes (Irish Social Science Data Archive, 2012).
- **Low Carbon London**: The low carbon project involved over five thousand households in London (Schofield *et al.*, 2015). This project's scope was to study the influence of low carbon technologies on London's electricity distribution network (Wang *et al.*, 2018).
- **Pecan Street**: Promoted by the Pecan Street experiment in Austin, TX this dataset contains energy consumption observations measured on minute intervals. 500 homes took part on this experiment (Pecan Street, 2012).
- **Building Data Genome Project**: The difference between this dataset and the previous ones is that it contains 507 whole building electrical meters mainly from university campus buildings (Miller, 2017).
- **GEF Comp 2012**: In this global energy forecasting competition hourly load and temperature data were provided. The weather information were extracted by 11 weather stations, and the top ranked methods used in this competition was analysed by Hong, Pinson and Fan, (2014).
- **GEF Comp 2015**: In this competition the data set used contained hourly energy observations and 25 weather stations provided the weather information. An in depth study around the data set and the forecasting methods implemented during this competition can be found on the paper of (Hong *et al.*, 2016).

The dataset used to model and forecast the energy consumption contains smart meter energy consumption data. Those reading where extracted from a sample of 5,567 London households that took part in the UK power networks during the period of November 2011 and February 2014 also, the readings contained in the dataset where measured in half hour intervals and the households have been also assigned an CACI Acron Group (UK Power Networks, 2014). Acron is a geodemographic detachment of the UK's population (CACI, 2014). The customers whose households where included in the trial were recruited as a balanced sample representative of the Greater London population. After loading the database in IPython (Fernando Pérez, 2007) the data exploration can begin. Initially the dataset was formatted appropriately in order to have full time series, also each series should start and end from the same timestamp. After the cleaning processes described, the final dataset consists of 4961 series with data range from 30/10/2013 – 27/2/2014. The data set contains the following columns:

- **LCLid**: Unique ID for every household contained in the dataset.
- **stdorToU**: In this column the possible variables are two, standard or dynamic time of use. On the dynamic time of use customers were issued High (67.20p/kWh), Low(3.99p/kWh), Normal(11.76p/kwh). On the standard time of use the customers where charged a flat rate of (14.22p/kWh). To summarize, this variable contains information about every household if they are signed up for a standard or a dynamic time of use billing scheme (CACI, 2014).
- **KWH/hh**: Smart meter measurements of energy consumption.
- **Acron**: Acron Category.
- **Acron Grouped**: Each Acron category has further subcategories called Acron Grouped.

A separate data set containing weather information was also used during the last parts of the analysis. The weather dataset contained the following columns:

- **Visibility**
- **Wind Bearing**: Wind direction.
- **Temperature**
- **Dew Point**: The temperature at which the air needs to cool to reach saturation.
- **Pressure**: A measurement describing the pressure inside the atmosphere of the earth.
- **Apparent Temperature**: A measurement used to describe temperature as humans perceive it, the values of apparent temperature are produced by combining temperature, humidity and wind speed.
- **Wind Speed**: An atmospheric quantity caused by temperature change.
- **Humidity**: Describes how much vapour water is contained in the air.

4.1 Data Preprocessing

Data Preprocessing is a procedure that formats the dataset according to the needs of the analysis the researcher aims to achieve. The dataset analysed in this study contained a column with merged date and time information, using the library Pandas (McKinney, 2010) the time of every observation along with the weekday or weekend information was extracted and placed separately in a new column.

Another core processes of data pre-processing is the dataset health check. When dealing with real-world data like the one used for this study the phenomenon of missing values is common. Using Python and the library Pandas the dataset was checked for any missing value. 390 null values where found on regular intervals over groupings of houses, the cause of this may be the reset of the smart meter after some period of time or after a certain number of observations, the individual responsible for the manufacturing of the specific model of smart meters could be contacted for further insight about the causation of this phenomenon. The null values where removed in order to avoid a biased outcome.

5 Model Induction

In this section the models used in this study will be introduced and analysed from a statistical perspective in order to formulate an understanding of how each one of them produces its results. The goal of accurate consumption forecasting requires attention to a few points of paramount importance. The first one is to distinguish any additional parameters and variables that may affect energy consumption. For this study, the hour of the day, the day of the week and the weather information will be used. The second one is the selection of a modelling methodology that can cope with the difficulties occurring in the consumption modelling task. A common issue in this area, as with many other modelling studies, is that the input and the output variables may present a nonlinear relationship and the nature of nonlinearity is not known well.

There are many tests proposed that hypothesize if a sample of a dataset looks as it was drawn from a normal distribution or not, for the purpose of this study two such statistical tests where used. The first one is the Shapiro-Wilk Test, which is considered to be a reliable test of normality despite the fact that suggestions exist that do not recommend this specific test for large datasets (Shapiro and Wilk, 1965). That is the reason another normality test is considered in order to create a solid statement about the normality of the data. The second test used is the Anderson-Darling Test (Anderson, 1952), which is a customized and a more powerful version of the non-parametric goodness of fit statistical test Kolmogorov-Smirnov Test. The tests were implemented using Python's library SciPy (Oliphant, 2007), and each of them calculates a test-specific statistic and a p-value. The test statistic can be used to interpret the results of the test but it requires proficiency in the mathematical mechanics and the statistics of the test itself, which doesn't fall in the area of interest of this study so the $p-value$ will be used for a quick and accurate interpretation of the test statistic. A threshold of 5% (0.05) was used and our null hypothesis is that the data are normally distributed. A $p-value > threshold$ concludes that our data are normally distributed so we failed to reject the null hypothesis, a $p-value \leq threshold$ concludes that our data are not normally distributed and the null hypothesis. Both of the tests resulted in a $p-value < 0.001$ which means that means that the results are highly significant and we can reject the null hypothesis since $p-value \leq threshold,$ concluding that our data are non-linear. In order to address this issue of non-linearity presented in this study, all the models included can handle data that present nonlinear relationships.

For each model selected in this study there are numerous variations proposed by researchers and it would a hopeless project to think about every single existing combination. Therefore, the approach selected is to consider the basic method of each model, this decision was made also having in mind that most researchers will more likely opt to use the basic form of a model especially if they are newcomers to the field. For example, there are models used, such as KNN and MLP that can be further parametrized using different weighting techniques such as non-Euclidian distance-based or flexible metric KNN but in this study those variants are not being considered.

The main reason that these particular models descried below where chosen for this study is the fact that they are some of the most commonly used models and they can also handle effectively nonlinear relationships. Furthermore, all the models selected for this study are regression learning algorithms, this choice was made because the output variable, in our case energy consumption, is a continues variable. Also, another factor that lead to the selection of these particular models is the fact that they can effectively work with multiple input variables, this was one of the main requirements for this study since multiple predictor variables, such as hour and weather observations, will be used to enhance the forecasting of energy consumption. Hence, the models that satisfy the requirements of this study are the following:

- Long Short-Term Memory (LSTM)
- Support Vector Regression (SVR)
- K-Nearest Neighbours Regression (KNR)
- Quantile Random Forests (QRF)
- Kernel Ridge Regression (KRR)
- Multilayer Perceptron (MLP)

Below there is a separate section describing every model considered.

5.1 Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) networks are a variation of Recurrent Neural Networks (RNN), developed by Hochreiter and Schmidhuber (1997) as a neural network architecture used in order to temporal sequences of data (Monner and Reggia, 2012). Prior this development that RNNs faced, the standard model was capable, in theory, to make use of information containing long sequences but in practice they appeared to be limited in capturing just a few time steps. This limitation was studied in depth by Hochreiter (1991) and Bengio, Simard and Frasconi (1994) and they concluded that this limitation cannot be surpassed due to the decay of the backpropagated error when the size of the time lag between relevant information increases. This is called the “vanishing gradient” problem. The creation and architecture of LSTMs by (Hochreiter and Schmidhuber, 1997) helped RNNs overcome this fundamental issue of the “vanishing gradient” by learning tasks involving long term dependences (Gers and Gers, 2001). LSTM have the peculiarity, that they consist of long short-term memory blocks which are constituted of memory cell units able to remember the value of a state for an arbitrary long time, but also three different gate units that are able to keep utilize and destroy a state when appropriate (Amadou Boukary and Rivest, 2016). Despite the fact that LSTMs have the ability to learn long term dependences, their training was found to be extremely slow, often taking millions of iterations to be completed (Gers, Schraudolph and Schmidhuber, 2003).

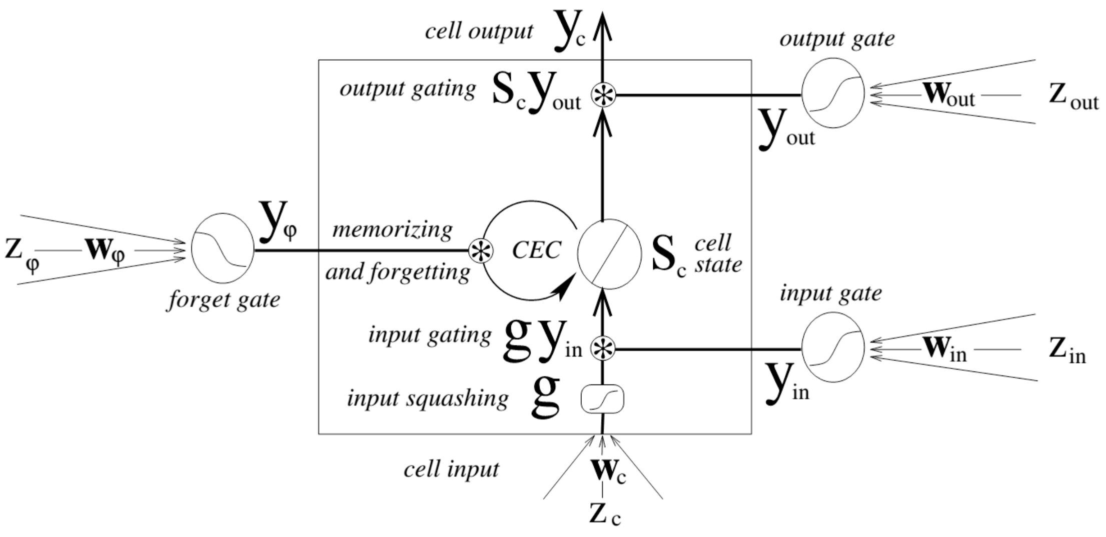

Figure 5: Example of LSTM memory block architecture. Source: (Gers, Schraudolph and Schmidhuber, 2003)

A simple yet powerful regularization technique used mainly in neural networks and deep learning models and will also be used in this study is dropout. Dropout was initially proposed by Srivastava *et al.*, (2014) and its main function is randomly select neurons and ignore them during the training phase. So, the dropped-out neurons do not contribute temporally to the activation of the downstream neurons, to the forward pass and any weight updates are not applied to the neuron on the backward press. On the learning phase of the neural network, neuron weights are being assigned within the network. These weights are assigned for specific features providing some specialization and neighbouring neurons often rely on this specialization which can lead to an over-specialized to the training set model. The usage of dropout can resolve this issue because the dropped-out neurons have to be replaced by other neurons ad handle the representation required in order to make predictions for the missing neurons. The general effect of dropout is that helps the network to create a model with better generalization and less chances to exhibit overfitting.

5.2 Support Vector Regression (SVR)

Support Vector Regression (SVR) is the regression process performed by a Support Vector Machine which tries to identify the hyperplane that maximizes the margin between two classes and minimize the total error under tolerance. In order for an efficient SVM to be constructed, a penalty of the ensuring complexity is also introduced to the error function, balancing forecasting accuracy and computational performance. Considering an illustration of a linear model, the prediction is given by:

$$f(x) = w^T x + b \tag{13}$$

With w representing the weight vector, b the bias and x the input vector. Let x_m and y_m denote respectively the m^{th} training input vector ad target output, $m = 1, \dots, M$. The error function is given by:

$$J = \frac{1}{2}||w||^2 + C \sum_{m=1}^{M} |y_m - f(x_m)|_e \tag{14}$$

The first term in the defined error function is the one responsible to penalize the model's complexity. The second term is the e-insensitive loss function, defined as $|y_m - f(x_m)|_e = \max\{0, |y_m - f(x_m)| - e\}$. It does not penalize errors below e, this provides some room for the parameters to move in order to achieve reduced complexity. Also, it can be shown that the solution that minimizes the error function is given by:

$$f(x) = \sum_{m=1}^{M} (a_m^* - a_m) x_m^T x + b \tag{15}$$

where a_m^* and a_m are Lagrange multipliers. The training vectors that is giving non-zero Lagrange multipliers are called support vectors, and this is the core concept in the Support Vector Regression theory. The non-support vectors do not contribute directly to the solution and the number of support vectors is some measure of model complexity (Chalimourda, Schölkopf and Smola, 2004; Cherkassky and Ma, 2004). The model can be extended to the nonlinear case through the concept of kernel K, and can be defied as:

$$f(x) = \sum_{m=1}^{M} (a_m^* - a_m) K(x_m^T x) + b \tag{16}$$

5.3 K-Nearest Neighbour Regression (KNR)

The K-Nearest Neighbor Regression (KNR) method falls into the category of nonparametric methods basing its forecast on a similarity measure, the Euclidean distance between the points used for training and testing. Thus, given N inputs, the method picks the closest K training data points and sets the prediction as the average of the target output values for these points (Friedman, Hastie and Robert, 2007). Quantitatively speaking, let $J(x)$ be the set of K nearest neighbours of point x. Then the prediction can be calculated using:

$$\hat{y} = \frac{1}{K} \sum_{m \in J(x)} y_m \ , \tag{17}$$

where y_m is the target output for the training data point x_m. The hyperparameter K *for* this method has to be selected with care. A large K will lead to a smoother fit, and therefore a lower variance, of course at the expense of a higher bias, and vice versa for a small K.

5.4 Quantile Regression Forests (QRF)

Quantile Regression Forests (QRF) is an extension of random forests developed by Nicolai Meinshausen (2006) that provides a non-parametric extension of the normal random forests developed by (Breiman, 2001). The normal random forests model approximates the conditional mean $E(Y|X = x)$ by a weighted mean over the observations of the response variable Y. It can be observed that the weighted observations do not provide a good approximation to the conditional mean but to the full conditional distribution. The conditional distribution function of Y, given $X = x$, can be described as:

$$F(y|X = x) = P(Y \leq y|X = x) = E(1_{\{Y \leq y\}} \ |X = x). \tag{18}$$

The last expression in the above equation is suited to draw analogies with the random forest approximation of the conditional mean $E(Y|X = x)$. Just as $E(Y|X = x)$ is approximated by a weighted mean over the observations of Y, a new approximation is defined as $E(1_{\{Y \leq y\}} \ |X = x)$ by the weighted mean over the observations of $1_{\{Y \leq y\}}$,

$$\hat{F}(y|X = \ x) \ = \sum_{i=1}^{n} w_i \, (x) 1_{\{Yi \leq y\}} \ , \tag{19}$$

using the same weights as the ones used in the normal random forests model. This approximation is at the heart of the quantile random forests regression algorithm. The key difference between quantile random forests regression and random forests is that, for each node in each tree, random forests keep only the mean of the observations that fall into this node and neglects all other information. In contrast, quantile random forests regression keeps the value of all the observations in this node, it just the mean, and assesses the conditional distribution based on this information (Meinshausen, 2006).

5.5 Kernel Ridge Regression (KRR)

The Kernel Ridge Regression (KRR) (Zhang, Duchi and Wainwright, 2013) is a combination of ridge regression, a method to address ill-posedness and overfitting in the standard regression setting via L2 regularization, and kernel techniques.

The Kernel method (Thomas Hofmann and Smola, 2008) is being implemented in many real problems when the underlying model cannot be described by a linear function, so the model of linear ridge regression suffers from poor prediction accuracy. A common approach to solve this issue is to map samples to a high dimensional space using non-linear mapping, and then learn the model in the new high dimensional space (You *et al.*, 2018).

Combining the Kernel method with Ridge Regression yields Kernel Ridge Regression presented in Equations 20 and 21. The $\|\cdot\|_H$ in Equation 20 is a Reproducing Hilbert Space approach (RKHS) (Zhang, Duchi and Wainwright, 2013). Using the n-by-n kernel matrix K the problem can be transformed to a linear system and defined in Equation 22. K is the kernel matrix and it's constructed by $K_{i,j} = \Phi(x_i, x_j)$, y is the input n-by-1 regressand vector corresponding to X, and a is the n-by-1 unknown vector.

$$argmin \frac{1}{n} \sum_{i=1}^{n} \| f_i - y_i \|_2^2 + \lambda \| f \|_H^2 \qquad (20)$$

$$f_i = \sum_{j=1}^{n} a_j \Phi(x_j, x_i) \qquad (21)$$

During the training phase, the algorithm is approximately solving the linear system demonstrated in Equation 22 in order to get a. During the prediction phase, the algorithm uses a calculated before, in order to predict the dependent variable of any unknown sample $\hat{x}$ solving the Equation 23 (You *et al.*, 2018).

$$(K + \lambda n I)a = y \qquad (22)$$

$$\tilde{y} = \sum_{i=1}^{n} a_i \Phi(x_i, \hat{x}) \qquad (23)$$

5.6 Multilayer Perceptron (MLP)

The Multilayer Perceptron (MLP) is constructed through a system of simple of interconnected neurons, and can be descried as a model representing a nonlinear mapping between an input vector and an output vector. The use of such a model has been applied to a variety of tasks such as pattern classification and pattern approximation (Bishop C., 1995) but in our case, the multilayer perception will be used for a regression task. The MLP algorithm can be descried as

$$\hat{y} = v_0 + \sum_{j=1}^{NH} v_j g(w_j^T x') \tag{24}$$

where x' is the input vector x, augmented with 1, w_j is the weight vector for j^{th} hidden node, $v_0, v_1, \dots, v_{NH}$ are the weights for the output node, and $\hat{y}$ is the network output. The function g represents the hidden node output, and it is given through the terms of a squashing function, the one used in this study is the hyperbolic tangent and can be defined as $g(u) = \ e^{2x} - 1/e^{2x} + 1$. The MLP is considered as a high parametrized model, and the complexity of the model highly depends on the number of the hidden nodes. The universal estimation property was responsible for the leap forward and loaned assurance to the importance of neural network models (Funahashi, 1989; Gybenko, 1989; Hornik, Stinchcombe and White, 1989; Leshno *et al.*, 1993). After validating some mild assumptions on the hidden node functions g, an approximation as close as arbitrary can be made for any given continuous function using a network with finite number of hidden nodes. Such a result can be reassuring but it is of great importance to avoid overparameterization, especially when neural network models are used in forecasting where typically the amount of data is limited and with high noise (Ahmed *et al.*, 2017). The model selection and particularly selecting how many hidden nodes should be created has attracted a lot of research interest in the neural network literature (Callen *et al.*, 1996; Medeiros, Teräsvirta and Gianluigi, 2006).

6 Optimizing the Models

For each of the selected models there is a wide range of parameters that a practitioner can tweak in order to optimize the model to suit the need of the analysis, some of these parameters are defined as hyperparameters and should be handled with caution because they are the ones responsible for the model's complexity. These, are input variables that have to be specified by the user and feed to the machine learning algorithm. The dominant approach used in machine learning literature in order to define the best hypermeter is the K-fold cross validation approach. The cross-validation (CV) approach involves randomly dividing the set of observations into *k* folds, or groups of approximately equal size. Then, the first fold is treated as a validation set, and the method is fit on the remaining *k-1* folds. It is important to note that any data preparation prior to fitting the model should take place on the CV-assigned training set within the loop rather than on the whole dataset, this technique should also be used during the hyperparameter tuning.

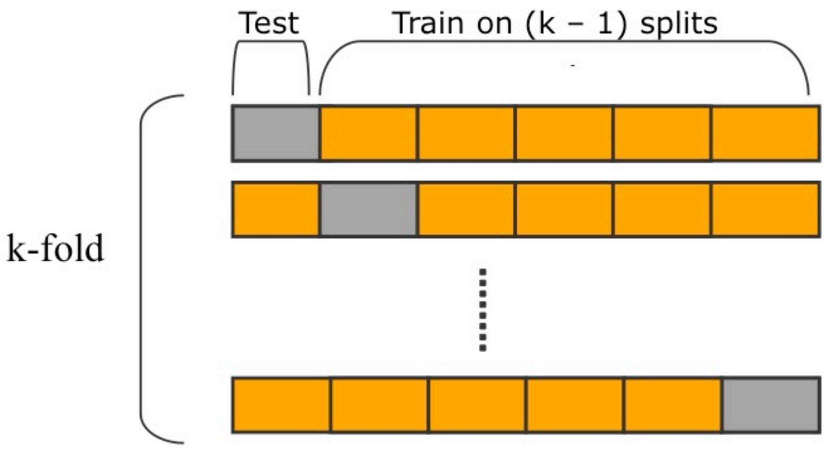

Figure 6 K-Fold CV Procedure. Source: (Kong, 2017)

The *k* value has to be provided by the user and should be chosen carefully. A poorly chosen *k* value may result in a mis-representative estimation of the effectiveness of the model, such as a score containing high variance or high bias. A lot of tactics are generally used in order to define the best value of *k,* the most popular of them are described below:

- Representative: In this technique the value of *k* is chosen such that each of the training / test set of data samples is large enough to be statistically representative of the full dataset.

- $k = 10$: Use the value of 10 as k, this is a technique that has been confirmed through experimental studies that results in a model estimate with low bias and reasonable variance.
- $k = n$: In this technique k is defined as n, where n is the total size of the data set. This technique is also known as leave-one-out cross validation.

Empirical comparisons have tested K-fold cross validation with other procedures such as the hold out, the leave one out and bootstrap methods, and concluded that K-fold cross validation is superior (Kohavi, 1995). Furthermore, a trade-off between bias and variance is highly associated with the value of *k*. Typically, when performing k-fold CV using k=10 it has been empirically verified that the yielded test error rate estimates suffer neither from excessively high bias nor from very high variance (James *et al.*, 2013). So for this study the k-fold cross validation with k=10 will be used along with the class GridSearchCV provided by the Scikit-Learn library (Pedregosa *et al.*, 2011). When creating this class, a dictionary containing the hyperparameters and the values to be optimized must be provided. Then the GridSearchCV will create and run the specified model using every combination of the parameters provided in the dictionary.

The number of neighbors K in the KNN Regression model was tested using GridSearchCV ,10-fold validation and the optimal outcome was found using $K = 3$.

For the QRF there are four hyperparameters that need to be defined. The number of trees *(n_estimators)* is a parameter that defines the number to be created in the forest, in general the more trees you use the better the results so this parameter is generally assigned a large value. There is no risk of overfitting by defining a large value in this hyperparameter because each tree is trained independently from each other. However, more trees will result in more computational cost after a certain number (Probst, Wright and Boulesteix, 2018). Next, the number of features *(max_features)* to consider at each split which will be left used with the default option so the square root of the total number of features will be used. The maximum depth *(max_depth)* which is essentially the number of nodes to be created and the minimum number of samples required at each leaf node *(min_samples_leaf)* are left to be optimized by GridSearchCV. The optimal outcome was found using *n_estimators = 10, max_depth=60, min_samples_leaf=4.* The hyperparameters that need to be tuned for GBR and QRF are similar so the same techniques were used for the tuning of both methods. The GBR model will therefore be used with *n_estimators = 30, max_depth=10, min_samples_leaf=4.*

The hyperparameters that are responsible for the complexity of the SVR model are *e (epsilon),* C and σ_K *(gamma)*. Chalimourda, Schölkopf and Smola, (2004) and Cherkassky and Ma, (2004) have completed an extensive analysis on the effects of these parameters when used for out of sample prediction performance and model complexity is presented. Chalimourda, Schölkopf and Smola (2004) concluded that the value of C should be initialized as the maximum of the target output value and that the performance prediction should not be very sensitive to the value of C. Also, in the same study Chalimourda, Schölkopf and Smola (2004) contented that the prediction performance is sensitive to σ_K and this parameter should be carefully selected. The above technique concerning the value of C will be used in this study also, by defining $C=4,421$. The rest of the hyperparameters where defined using GridSearchCV ,10-fold validation and the optimal value was found using $epsilon = 0.01$, $gamma = 30$. The kernel used for this model is the Radial Basis Function.

For the KRR the hyperparameters $alpha, gamma$ and $kernel$ need to be defined. As mentioned before KRR performs L2 regularization, $alpha$ is the term responsible for the weight in this regularization. If the value of $alpha$ is zero then the model created is just an Ordinary Least Squares Regression model. For small values of $alpha,$ the magnitude of the coefficients would be higher, moreover in the official documentation of KRR in Scikit-Learn (Pedregosa *et al*., 2011) it is stated that small positive values of alpha improve the conditioning of the problem and reduce the variance of the estimates, having all that I mind the values of $alpha$ and $gamma$ were assigned using $GridSearchCV$. The $kernel$ chosen is the Radial Basis Function because it can capture non-linear relationships, which is highly important for this study, by uplifting the sample values into a higher dimensional feature space. The final hyperparameters used for the KRR model are $alpha=20,\ gamma=0.01,$ $kernel=RBF.$

The hyperparameters that need to be optimized for the MLP model are described in the following section. A single layer Neural Network (NN) is constructed at first and the best number of input nodes was optimized using 10-Fold CV. After, using the optimized input node value, the number of hidden nodes is optimized with the same technique. It possible to create an MLP with zero hidden nodes which will outcome to a linear model. Balkin and Ord, (2000) have concluded that may times switching to a linear model can improve performance, unfortunately a linear model doesn't work well with our data so this possibility won't be further investigated. After using GridSearchCV for the rest of the hyperparameters the values to be used for the MLP model are the following $hidden\_layer\_sizes= (10,50),\ activation=tanh,\ solver=lbfgs,\ alpha=0.05,\ max\_iterations=500.$

For the LSTM method the hyperparameters considered are the $epoch$ which is the number of times that the training data set is shown to the network during training and its generally defined as "one pass over the entire dataset", the $\mathrm{batch\_size}$ which defines the number of samples that are going to be propagated through the network. A batch generally approximates the distribution of the input data better that a single input. If batch_size has a large value the approximation will be also better. However, a large *batch_size* will have an impact on the processing time and the result would be still only one update (Chollet, 2015). After considering all the above insights and running a GridSearchCV function the parameters to be used are: *batch_size=500, epoch=250, optimizer=Adam, loss=mae.*

Keras provides a powerful regularization technique that can be implemented in both neural networks and deep learning modes. This technique is called drop_out and can be applied on the different layers by the user if needed. The original paper that proposed dropout by Srivastava *et al*. (2014) provided some useful heuristics that will be considered while parametrizing dropout for this study:

- Generally, a low dropout value should be used, between 20%-50% of neurons was found to be optimal. An extremely low value would have minimal effect on the overall model performance but on the other hand an extreme value would result in an under-learning network.
- Dropouts effectiveness is better utilized when used on larger networks because of the fact that the model will be given more opportunities to learn independent representations.
- The use of dropout on both visible and hidden layers has indicated good results.
- Implementing a constraint on the size of the network also provides improved results. A popular technique is the max-norm regularization.

For the model used in this study *drop_out* will be used both on the input neurons, also called the visible layer, and the hidden layers with a *drop_out* value of *0.2.*

Keras has available a set of call-back functions that can be applied during the training procedure. From those functions the ones included in our model is the EarlyStopping function which monitors the quality of a specified quantity and stops the training phase if no significant improvement has been monitored. The EarlyStopping was implemented using 10 epoch patience (number of epoch that the model should present and improvement before the training is stopped) and the quantity to be monitored is the model's loss. Also, *ReduceLROnPlateu* is included in our model which is responsible for reducing learning rate when a metric has stopped improving, 10 epoch patience with model loss is also used as previously (Chollet, 2015). The learning rate will be reduced by $factor = 0.5$ if there is no significant improvement to the model's loss, and the new learning rate will be re-defined as following: $new\_LearningRate = LearningRate \times factor$. Furthermore, the TerminateOnNaN call-back function is included in our model. The scope pf this function is to terminate the training is a not a number *(NaN)* (Chollet, 2015). The general form of the LSTM model in an input layer followed by 3 hidden layers and the output.

7 Experimental Setup

The forecasting process for this study is divided into three stages. The first one is to construct a baseline forecasting model which will provide a point of reference for the more complex forecasts used in the later stages. A baseline model should be a simple method that requires limited training intelligence, also it should be implemented and require minor computational power in order to produce a prediction and lastly is should be model that can be easily repeated. The model created to serve as a benchmark for this study is a one day ahead forecasting model, meaning that it will use the last 28 days for training in order to forecast the next day's *energy consumption.* This technique is applied to all of the models included in our study.

The second stage of the process involves creating a model similar to the previous one with the only difference that now a back-test technique will be used. There are various validation techniques used in machine learning such as train-test split, k-fold CV and more. Although these methods are effective in pure machine learning problems its effectiveness is limited in time series problems. The reason for that is the fact that they ignore the temporal components that are essential for any time series model. In order to overcome this issue some back testing techniques have been developed in order to enable researchers to validate machine learning model's in time series data. The technique used in this stage of our study is walk forward validation and as its name's suggests it involves moving along the time series data set one step at a time. Also, because an expanding window is used during the training phase this technique may be also referred as rolling forecast. Before starting the validation of the models using this technique there are some parameters to be defined, first the width of the sliding window which is essentially the minimum number of observations that the model will use during the training phase. Next a choice between a sliding or an expanding window must be made. The first one will use the whole data set for training whereas the latter on only the later number of the observations defined by the windows width. Both of these techniques have their applicable cases but the sliding window will be used for our experiment mainly due to the fact that its more effective when applied to high frequency data like ours.

The third step is similar to the previous with the only difference being that more variables will be introduced in order to see if the model's performance improves. Until now the variables used to make predictions was the *hour* and *weekday,* during this last step *weather* information (visibility, wind bearing, temperature, dew point, pressure, apparent temperature, wind speed, humidity) are included in the forecasting process, again using rolling forecast. Before moving to the results, the following sections describe what accuracy measures and data transformations has been used.

7.1 Computational Complexity

Computational Complexity (CC) will be used in order to measure the time it takes for a model to train, fit and predict. Hence, the computation of CC can be defined as the computational time a model needs to produce a prediction. The computation time is highly dependent on the characteristics of the system used to run the algorithm, all the computations used for this study was measured using a system with the following characteristics: $Intel^{®}$ Core i7 @ 2.6GHz and 16 GB ram on a macOS environment. The time was measured using the built-in command of Jupyter $\%\%time$.

7.2 Accuracy Measures

In order to assess the ability of the forecasted variables, there are different forecast accuracy measures that can be used depending on the task and the model. An error measure can be simply defined as the difference between a forecasted value and the actual one. If y_m can be described as the m^{th} original observation and $\hat{y}_m$ as the forecasted value of y_m, then the forecast error can be defined as $e_t = y_t - \hat{y}_t$, which should be on the same scale as the data, because making comparisons between series that are measured on different scales is not effective.

Percentage error measurements will be used in this study because of their advantage of being scale independent and that's also the reason they are popular when comparing performance between different datasets (Hyndman and Athanasopoulos, 2014) and commonly used in electricity prediction studies (Zhao and Magoulès, 2012; Grolinger *et al.*, 2016; Grolinger, Capretz and Seewald, 2016). Since our values lie in the positive half plane, Shcherbakov *et al.*, (2013) recommends that a symmetric error measure should be used. The Symmetric Mean Absolute percentage error (sMAPE) will be used as an error measurement for this study, which can be defined as:

$$sMAPE = \frac{1}{M}\sum_{m=1}^{M} \frac{|\widehat{y_m} - y_m|}{(|\widehat{y_m}| + |y_m|)/2}\ , \qquad (25)$$

where $\widehat{y_m}$ is the predicted value and y_m is the initial value. The sMAPE is a relative measurement of error so it is possible to combine the errors of different time series into one number. Also, sMAPE self has an error range of 0-200%.

Moreover, an absolute forecasting error will also be used. These errors include estimates based on the calculation of the value e_i (Shcherbakov *et al.*, 2013), which can be defined as:

$$e_i = (y_t - f_t^{(m)})\ , \qquad (26)$$

In the Equation 27, y_t is the measured observation at time t and the $f_t^{(m)}$ is the predicted value at time t obtained using the model. The Mean Absolute Error (MAE) included in this study is given by:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|e_i| \ , \qquad (27)$$

where n is the forecasting horizon (Shcherbakov *et al.*, 2013). MAE measures the average magnitude of the forecasting errors, without contemplating their direction.

7.3 Data Transformation

When using traditional time series forecasting methods achieving stationarity in the mean and variance is a considered essential, there are several studies in machine learning literature claiming that modern machine learning algorithms can be applied to the original data without any loss in performance (Gorr, 1994). On the other hand there are studies supporting the opposite side, claiming that without data pre-processing machine learning models may present unstable performance and yield suboptimal results (Zhang and Qi, 2005).

In general, transformation techniques can be enforced in three distinct categories: seasonal adjustments, log or power transformation, and removing the trend (Makridakis, Spiliotis and Assimakopoulos, 2018). Sharda and Patil, (1992) concluded that MLP lacks in capturing seasonality appropriately, on the other hand Nelson, (1994) supports the exact opposite view. Studies concerning the trend in time series suggest that the bouncing of MLP's activation function can become more stable if a detrending technique is applied in the time series (Cottrell *et al.*, 1995). Yet, there is a general inadequacy in empirical results able to provide a general conclusive way of data pre-processing, as well as a technique to eradicate the trend in the data (Nelson and Plosser, 1982). For the purpose of this study the following pre-processing methods will be tested:

- **Original Data:** No pre-processing method applied.
- **Log–Transformation**: Simply take the *log* of the time series. Log transforms are popular in the field of time series because it is an effective way to remove the exponential variance.
- **Box-Cox Transformation** (Box *et al.*, 1964): The Box-Cox transformation was utilized using the *scipy.stats* library (Oliphant, 2007) which will evaluate a suit of transforms automatically and select the best *lambda* parameter. The *lambda* parameter is responsible for the type of transformation the *boxcox()* function will perform.

In order to choose which data transformation works best with our problem, we apply the techniques described above to the baseline one day ahead model of the QRF technique and the data transformation that produces the best results is the applied to the remaining machine learning models.

Table 1. The effect of different transformation methods in the QRF baseline model

Method	sMAPE (%)	MAE	CC
Original Data	35.17	0.19	516ms
Log-Transformation	27.97	0.12	526ms
Box-Cox Transformation	170.20	0.21	525ms

The results of the data transformation experiment are noted in *Table 1*. Before moving to the interpretation of the results it is worth noting that since this test was made on our baseline model of QRF the CC is just the time it took to produce the forecasting results. Looking on the results of *Table 1* it can be concluded that there are important differences in sMAPE and MAE across the different transformation techniques and that the Log-Transformation is the most effective almost in all the measures. It produced the best results in both sMAPE and MAE error measurements, however on the CC it is ranked last, while using our original data without any pre-processing we can achieve the lowest processing time. Also, it can be observed that using a Box-Cox Transformation the result of the sMAPE value is extreme. This explain a limitation of the sMAPE itself, when we transform the values of *energy measurement* using the Box-Cox transformation the range changes from *[0.18 – 4.21]* to *[-2.44 - 1.094]* thus negatives exist in *energy measurement,* these negative values can result in a division by zero based on the *Equation 25* of the sMAPE. This is the main cause that sMAPE outputs such an extreme value when Box-Cox transformation is applied (Shcherbakov *et al*., 2013). As mentioned earlier we can choose what type of transformation the *boxcox()* will apply to the data but this choice was left to be optimized by the library itself, after some manual changes to the *lambda* parameter, no noticeable improvement was observed. Since log transformation gives the most promising results it will be used for every other model in this study.

8 Results

In this chapter the various results obtained from this study will be presented and explained. Starting with the feature extraction and the k-means clustering experiment, continuing with a section containing the results of the classification attempt on the ACRON category and the energy consumption, and finally with the forecasting modelling results and the statistical evaluation.

8.1 Feature Extraction and Clustering

The purpose of this section is to cluster the series contained in the data set using some extracted features, like mean, variance and lumpiness, to get a better understanding of the data and the clusters that can be created. The extracted features were generated using two techniques. The first one is non-overlapping windows, using this technique the variance of the means and the variance of the variances was generated for all the non-overlapping windows with 24 width. Furthermore, the next technique used is overlapping windows also known as sliding window. Using this technique, the $max\_variance\_shift$ which is the largest variance shift between two consecutive sliding windows and the max_mean_shift which is the maximum mean shift between two consecutive windows has been calculated. Moreover, another feature extracted is the $ratio\_beyond\_r\_sigma$ which can be descried as the ratio of values that are more than r sigma away from the mean and the $crossing\_points$ which is the number of times a time series passes the median. The feature extraction was implemented using the programming language Python (Rossum, 1995) and the libraries Pandas (McKinney, 2010), NumPy (Oliphant, 2010). The features described have been used to create a new dataset and the correlation matrix described on Table 2, to explore any possible relationship between the extracted features.

	mean	crossing_poits	lumpiness	max_var_shift	ratio_beyond
mean	1.000000	-0.205468	0.243240	0.592375	0.359188
crossing_points	-0.205468	1.000000	0.033286	-0.071982	-0.129883
lumpiness	0.243240	0.033286	1.000000	0.588151	-0.014175
max_var_shift	0.592375	-0.071982	0.588151	1.000000	-0.042850
ratio_beyond_r_sigma	0.359188	-0.129883	-0.014175	-0.042850	1.000000

Table 2. Correlation Matrix for the time series extracted values

We can observe from Table 2 that there is a wide variety of correlation coefficients in the extracted features, from negative to ~0.60. Moving further, Principal Component Analysis (PCA) will be used in order to curtail the dimensionality of the matrix containing the extracted features. Although the dimensionality may be reduced the variation will be retained up to the maximum extend, PCA's goal is to achieve a low-dimensional representation of the data that explains a good fraction of variance. Before implementing the PCA technique the data must be scaled, using the StandardScaler() function from the Scikit-Learn library each observation will be scaled. Another vital step for an effective PCA is the choice of component number, this number will be determined with the help of Figure 1 which is the cumulative explained variance as a function of the number of components.

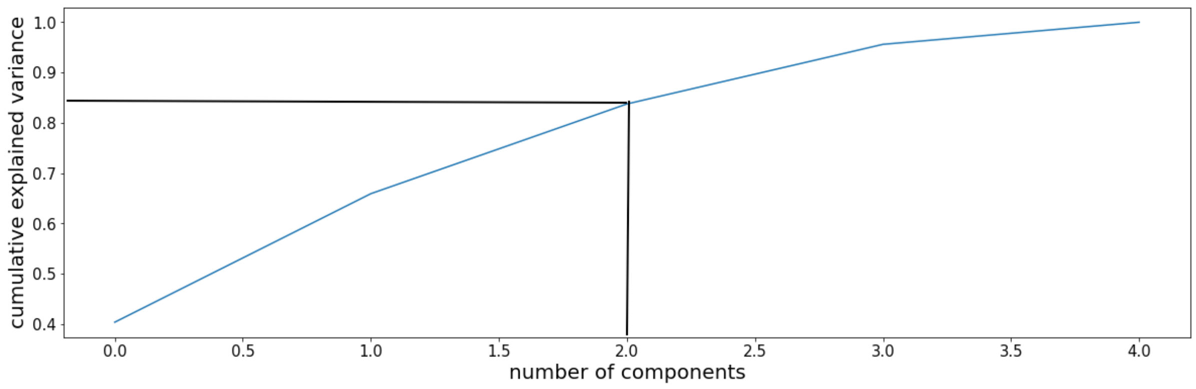

Figure 7. The cumulative explained variance as a function of the number of components

Figure 7 presents a curve which quantifies how much of the total variance is contained within the 4 components available. For this study the first 2 eigenvectors will be used as they are able to capture more than 80% the total variance, this means that our data will be projected from 5 to 2 dimensions. After running the PCA, the 2 components where created representing 40% and 25% of the total variance accordingly.

Continuing with the analysis, the K-means clustering algorithm will be implemented using the principal components created in the previous step. This choice was made based on the research of Chris Ding and Xiaofeng He and He, (2004) which states that both algorithms try to minimize the mean-squared reconstruction error. The PCA by trying to represent all n data vectors as linear combination of some eigenvectors, and similarly the K-means is to also represent all n data vectors but using cluster centroids. The k-means is an unsupervised learning algorithm having as a goal to create a number of groups with some given data, the number of this groups is represented by the variable K. The algorithm iterates over the data and tries to assign each data point to one of the K clusters based on the features provided. This procedure is accomplished by following two assumptions of what an optimal cluster should look like, which are also the fundamentals of the k-means model (VanderPlas, 2017):

- The cluster centre should be the arithmetic mean of all the members of the cluster.
- Every point should be closer to its own cluster centre than any other neighbour cluster centre.

In order to choose the optimal k for the k-mean algorithm, Figure 8 was generated.

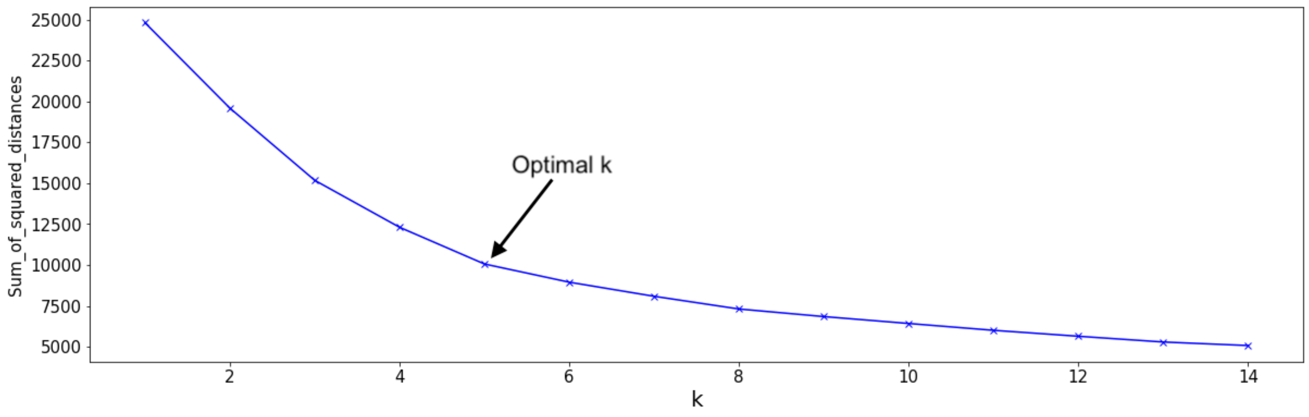

Figure 8. Elbow method for optimal k=5

A metric commonly used to compare the performance of the algorithm with different number of clusters is the *sum_of_squared_diference,* which is used to measure the distances of observations to the nearest cluster centre. It can be observed from Figure 6 that as the value of k increases, which is the number of clusters created by the algorithm, the *sum_of_squared_diference* teds to reach zero. If the k is set to the maximum then each observation of the dataset will create its own cluster which means that the value of sum*_of_squared_diference* will be zero. In Figure 6 the optimal number of k for this problem is demonstrated, the technique used to make this choice is the 'elbow' technique (Ng, 2012). The 'elbow' can be identified by following the curve doesn't follow such a large decrease as it used to, hence an elbow is created. Looking at Figure 6 the 'elbow' is identified at $k = 5$ so this will be the variable used for the k-means clustering technique and the output of the algorithm is demonstrated in Figure 9.

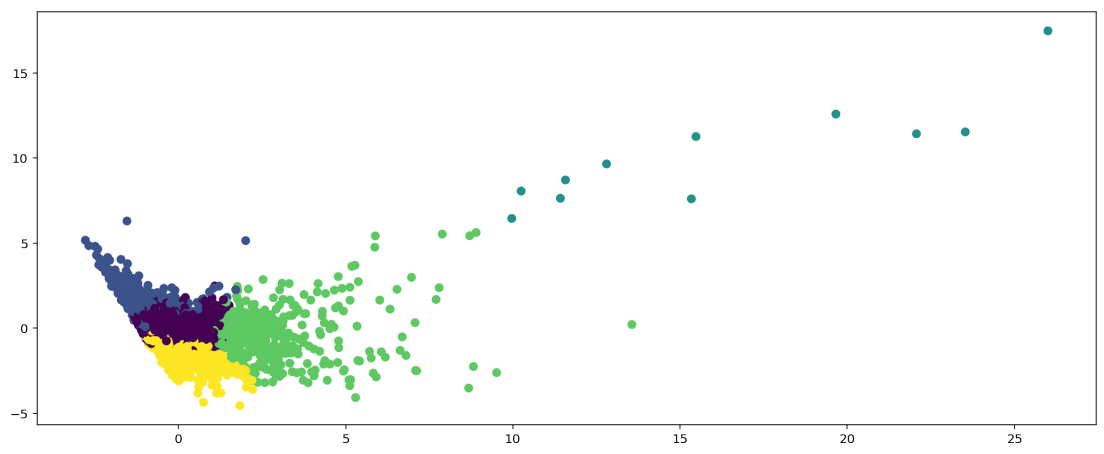

Figure 9. Output of the k-means algorithm used to cluster the extracted features

8.2 ACRON Classification

As mentioned before the dataset contains an ACRON label for each household, this seems an ideal situation to apply some classification algorithms and explore the Acron labelling in depth. So, a classification problem is formulated that will try to predict the *acron_description* using as predictor variables the $kwh, day, hour$, it should be noted that the id column ,which is a unique id for every house, is excluded from the analysis because it produces an 100% accurate classifier and a biased result. The K-Nearest Neighbour (KNN) was chosen to handle the classification process through the Scikit-Learn package (Pedregosa *et al.*, 2011) , and after using GridSearchCV function the hyperparameters used are *n_neighbours=39* and *metric=euclidian* with *test_size=0.2*. There are many studies that suggest the use of *StandardScaler* function and scale the data before applying the KNN algorithm but after testing on this problem the results are better without it. Furthermore, after applying the KNN algorithm the accuracy received is *accuracy_score = 0.34,* which is considered a bad score. After some further exploration on the dataset a possible cause of this low accuracy was found and demonstrated on Figure 10, created using Seaborn (Waskom *et al.*, 2017).

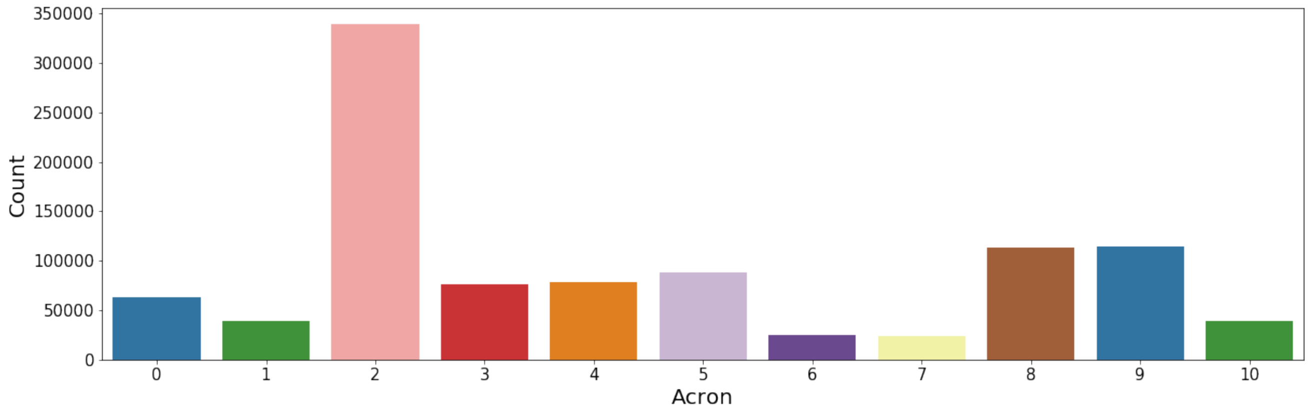

Figure 10. ACRON group population in the data

It can be concluded from Figure 10 that the households registered as *acron=2* can be found in the majority of the households contained in the dataset, so this may be a reason that the bad classification accuracy demonstrated before. To validate if that is the root of the problem under sampling was used for the *acron=2*. To deal with the imbalanced classes of the acron observations an under-sampling procedure will be implemented through the *resample()* function of Scikit-Learn (Pedregosa *et al*., 2011). Under-sampling methods use a subset of the majority class which in our case is *acron=2* to train the classifier (Ganganwar, 2012). Since many majority observations are removed from the class, the training set becomes more balanced and the training process becomes faster (Ganganwar, 2012). The under-sampling procedure used in this study involves removing observations randomly from the *acron=2* class. The observations from the acron column were divided in two separate data frames, the first data frame contained the observations with *acron=2* and the other one the rest. Next the data frame containing the *acron=2* observations is resampled without replacement by defining *replace=False* and by defining the number of observations to match a minority class *n_samples=88206.* After the resampling is completed a merge between the data frames is being made and the results are demonstrated in Figure 11, created using Seaborn (Waskom *et al*., 2017).

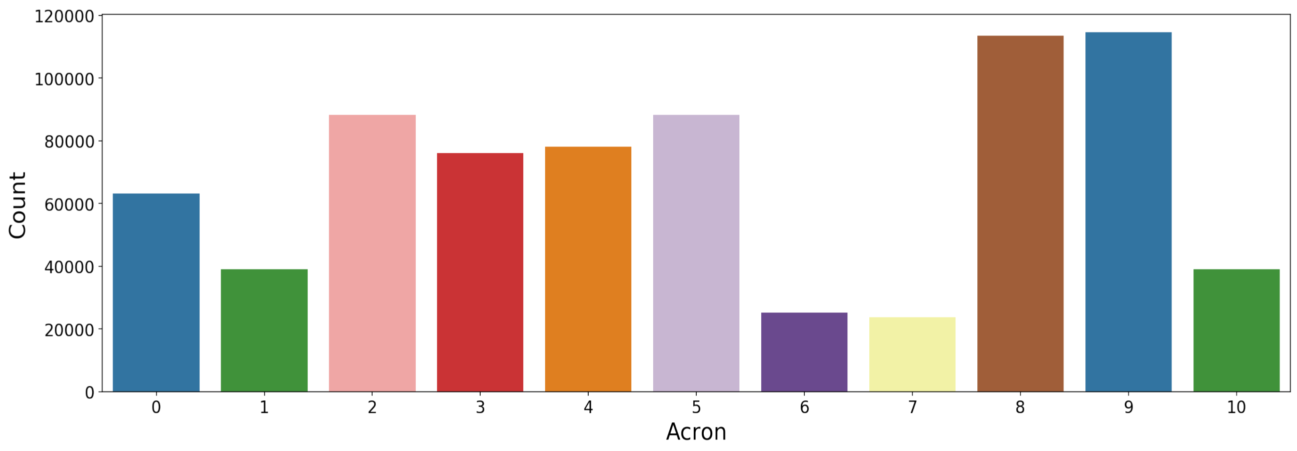

Figure 11. Results after down sampling on *acron=2*

After the resampling process is completed the KNN algorithm was applied again but the accuracy is lower than before, $accuracy\_score = 0.23$. A possible explanation for this phenomenon might be that some useful information was removed using the under-sampling so the classifier can't perform any better. Another reason might be the random under sampling caused by *replace=False* created an unrepresentative population and a biased sample.

Looking this issue from another view, one could argue that an oversampling could be used to increase the size of the minority acron classes instead of under-sampling the *acron=2.* Over-sampling is a non-heuristic method that randomly replicates observations in order to balance the class distribution. Nevertheless, since this method replicates existing examples in the minority class , overfitting is more likely to occur (Ganganwar, 2012).

Furthermore, to continue with the exploration of the acron categories an ensemble method will be used. Ensemble methods tend to capture more complicated and robust decision boundaries (Wang *et al*., 2017), to test if ensemble models can provide with a better classification accuracy the *RandomForrestClassifier()* is deployed from the Scikit-Learn library (Pedregosa *et al*., 2011). The model was applied without any hyperparameter tuning and no scaling and the resulting accuracy was $accuracy\_score = 0.26$ on the original data set and $accuracy\_score = 0.20$ on the down sampled dataset. A minor increase in accuracy can be noticed but the results are still not acceptable.

8.3 Energy Classification

Continuing with the data exploration a classification problem is formulated using the energy consumption observations. The KWH/hh column containing energy reading from smart meters is a continuous variable, in order to divide it into categories to format our classification problem the *pandas.qcut()* function was used (McKinney, 2010). This is a quantile-based discretization function, which will be used to divide the KWH/hh observations into three equal size buckets with *low, medium* and *high* labels accordingly. After the labels were created for every observation the *labelencoder()* function from Scikit-Learn (Pedregosa *et al*., 2011) was deployed in order to encode the labels into numerical values. The next step in the classification analysis in to feed the data into the classification algorithm. The KNN model will be used with *n_neighbours=39* and *test_size=0.2* which indicates the percentage of the data that will be stored for testing the algorithm and will not be used for training. After compiling the algorithm, the accuracy received is $accuracy\_score = 0.52$, concluding that out classification algorithm cannot be considered accurate.

8.4 Modelling Results

Starting with the results received from the first stage of the analysis a one day ahead baseline model with log transformation as the data pre-processing technique was created for every model included in this study and the results can be found on Table 3. The main purpose of this stage is to create a baseline model using a simple forecasting technique. The results obtained will be used as a point of reference for the more advanced forecasting techniques in order to observe any increase or decrease in the performance.

Model	sMAPE (%)	MAE	CC
SVR	28.81	**0.02**	484ms
KNR	**10.16**	0.06	**476ms**
QRF	29.37	0.07	507ms
KRR	12.14	0.12	491ms
MLP	29.69	0.09	1.06s
GBR	29.15	0.05	492ms

Table 3. Performance summary for the baseline model with one day ahead forecasts

The results of the baseline method, demonstrated in Table 3, provide some initial insight about the performance of the models. Inspecting the error measurements, it can be observed that two model seem to work significantly better that the others. KNR is the best performing model with 10.16% sMAPE and KRR the second with 12.14%. Looking at the MAE, the differences are small and the model achieving the lowest value is SVR with a MAE of 0.02 and the worst is KRR with a MAE score of

0.12. The runtime results are interesting, all of the models exhibited similar performance except NN which has the largest runtime with 1.06s.

Model	sMAPE (%)	MAE	CC
LSTM	21.13	0.15	4min17s
SVR	28.93	**0.05**	485ms
KNR	**13.5**	0.08	**324ms**
QRF	29.95	0.07	1.07s
KRR	13.65	0.08	1.73s
MLP	28.98	0.09	3m3s
GBR	29.06	**0.05**	486ms

Table 4. Performance summary for models with rolling forecast

Moving to the second step, where a rolling forecast was introduced to all of the models the performance results received are demonstrated on Table 4. The purpose of this step of the analysis is to see whether more advanced forecasting techniques, like the rolling forecast implemented, can produce better modeling results. Minor improvements can be observed compared to the one day ahead forecast. Most of the model perform similar with the previous method, the KNR method, which was the best performing model on the first step, posed ~3% increase in sMAPE. The only method that produced a lower sMAPE is the MLP and KRR with ~1% lower error rate from the previous step. The processing time presents a slight increase compared to the baseline model. The model presenting the worst CC is the LSTM with 4min and 17s, this is something that was expected given the number of *epochs* defined during the model's optimization.

Moving to the third and last step of the process, weather information was added to the rolling forecast model implemented in step two and the results are demonstrated on Table 5. It can be observed that moving from step two to step three resulted in an improvement on almost every model. A noticeable decrease in performance was inferred in the KRR model, which presented an ~80% increase in sMAPE and MLP an ~5%. A noticeable improvement can be observed on KNR, it was the best performing model on the previous stage and during this last experiment its sMAPE dropped another ~7%. On the MAE there is a general increase during this step, all of the models except from LSTM which only presented a minor decrease. The CC is also higher as expected, due to the fact that the models had to work with a bigger data set. Furthermore, the KNRs performance in CC is the best observed.

Model	sMAPE (%)	MAE	CC
LSTM	19.21	0.14	1min29s
SVR	25.18	0.19	1.97s
KNR	**6.56**	0.14	**350ms**
QRF	28.56	0.1	8.6s
KRR	97.44	0.34	6.7s
MLP	34.15	0.19	3.19
GBR	28.17	**0.09**	5.95s

Table 5. Performance summary for multivariate models with rolling forecast

A general conclusion made while observing the modeling results is that there are no significant improvements in the model's performance between the baseline models and the rolling forecast models that can justify the double CC required to execute the later ones. On the other hand, the third step which included weather information appears to have the best performing models overall.

8.5 Statistical Analysis

Along with model improvement in the field of Machine Learning there has to be a parallel development on the empirical evaluation and the comparison of such models. This would be of gigantic incentive towards the practitioner, as it would limit his conceivable choices, and give him knowledge into the strong and weak points of the available models(Ahmed *et al*., 2017). Therefore, the field of Machine Learning is now filled with studies trying to improve the basic methods, but only marginally succeeding in doing so (Hand, 2006).

Learning algorithms can be evaluated on a broader set of performance metrics. So, to accurately evaluate the performance of an algorithm, different performance metrics should be used because different learning algorithms are designed to optimize different criteria. It is not uncommon for an algorithm to have optimal performance on one metric and be suboptimal to another. However, these are not by themselves sufficient to fully evaluate the difference in performance between different learning algorithms on one or more test data sets (Japkowicz and Shah, 2011). More precisely, even though two models may perform different on a particular set of data, it needs to be confirmed that this difference is statistically significant and not merely coincidental. Statistical significance testing enabled researchers to move from performance measures techniques to more precise assessments of significance of the results obtained (Japkowicz and Shah, 2011). Nonetheless, the use of available statistical tools for such testing in the field of machine learning and data mining has been limited due to the fact that researchers have concentrated on using the paired *t* test, many times inappropriately, to confirm the difference between the performance of two models. (Japkowicz and Shah, 2011).

In order to test if a forecast produced by one model is better that a forecast produced by another, the Diebold-Mariano test (Diebold and Mariano, 1995) will be used. The main idea behind the DM approach is to make assumptions based entirely on the forecast errors (Diebold, 2015). The test compares two forecasting results, so two of the models included in this study must be selected. KRR and KNR are selected because these are considered the best two models while looking this study as a whole and the forecasts produced by them on the benchmark model will be tested, KNR scored 10.16% sMAPE / 0.06 MAE and KRR scored 13.65% sMAPE / 0.08 MAE. So, the Diebold-Mariano will test if the difference in the forecasting errors measured is significant or simply because of the specific data values contained in the sample. The mathematical formulation of the test will be described in the following section along with the outcome. The residuals of the forecasts produced by the two models included are:

$$e_i = y_i - f_i \ , \tag{28}$$

$$r_i = y_i - g_i \ . \tag{29}$$

And the measurement d_i can be defined as:

$$d_i = e_i^2 - r_i^2 \ \text{ or} \tag{30}$$

$$d_i = |e_i| - |r_i|. \tag{31}$$

On the Equations 29 and 30 demonstrated above, it can be observed that the first one refers to the MSE error statistic and the second one to the MAE test statistic. It can now be defined:

$$\bar{d} = \frac{1}{n}\sum_{i=1}^{n}(d_i). \tag{32}$$

and the population mean of the loss differential is,

$$\mu = E[d_i]. \tag{33}$$

For $n > k \geq 1$, the γ_k can be defined as the autocovariance at $lag\ k$:

$$\gamma_k = \frac{1}{n}\sum_{i=k+1}^{n}(d_i - \bar{d})(d_{i-k} - \bar{d}) \tag{34}$$

For $h \geq 1$,the DM test statistic can be defined as following:

$$Diebold - Marino = \frac{\bar{d}}{\sqrt{\frac{\left[\gamma_0 + 2\sum_{k=1}^{h-1}\gamma_k\right]}{n}}}. \quad (35)$$

The null hypothesis to be tested can be defined as:

$$H_0 \colon E(d_t) = 0 \ \forall t, \quad (36)$$

and the alternative hypothesis:

$$H_1 \colon E(d_t) \neq 0. \quad (37)$$

The null hypothesis of the DM test states that the two forecasted observations have the same accuracy. Whereas the alternative hypothesis that the two forecasted observations don’t have the same accuracy. Under the assumption of the null hypothesis the DM test statistic is asymptotically $N(0,1)$ distributed. The null hypothesis of the DM which means that there is no difference in accuracy, will be rejected if the test statistic is outside the range of $-z_{a/2}$ to $z_{a/2}$, which will happen if the following equation is true:

$$|DM| > z_{a/2}. \quad (38)$$

In Equation 3 $z_{a/2}$ represents the positive z-score of the standard normal table corresponding to half of the desired a level of the test.

Moving to the implementation of the DM test, first the appropriate h value has to be selected. This can be acquired by $h = n^{1/3} + 1 = 10$, where n is the total number of forecasts, so the p value in h=10 will be inspected. A transformation on the acquired z value to $p - value$ is being done resulting in $p - value > 0.5$ so it can be concluded that there is no significance difference between the forecasting results of the KNR and KRR is not significant.

9 Discussion

Looking at this study as a whole it is sensible to start this discussion section with the hyperparameter tuning, which took place after the model selection. Hyperparameter tuning plays a crucial role on a model's performance because it defines what model parameters should be used in order to achieve optimal performance. Furthermore, this procedure was found to be the most computational demanding process during this study. Models had to be compiled several times and the demand for better hardware was obvious in most of the model's hyperparameter determination and especially in LSTM and MLP. An improvement to this situation could be implemented by using cloud platforms for such computational demanding processes like hyperparameter tuning.

Moving to the data-pre-processing, it can be concluded that transformation techniques can significantly help machine learning algorithms to produce more accurate forecasts. In detail, the QRF where the transformation test was conducted on produced a 7.2% less sMAPE and 0.7 less MAE. Furthermore, the Box-Cox transformation was found to be completely inappropriate transformation method for this dataset. So, this study suggests that choosing an appropriate data transformation technique can improve forecasts and supports the conclusions of Zhang and Qi, (2005) that machine learning algorithms can return suboptimal results without the appropriate data transformation.

In the results obtained, the difference in each algorithms performance is reflected on both error measurements and computational complexity. sMAPE and MAPE follow different approaches towards error calculation they tend to follow a different trend, meaning that models scoring good on sMAPE may score lower on MAE. This is the main reason that these error measurements cannot be compared with each other. Generally, while using rolling forecast both the error rates increased compared to the one day ahead forecast. This issue was also observed in the multivariate rolling forecast models for most of the algorithms except KNR and LSTM which presented better results for every step. Looking at the CC it can be concluded that it follows an increasing trend for each step as expected, because more complex models require more time to be processed. Also, by inspecting the CC of the models it can be concluded that NN models need significantly more time to produce a prediction. The criterion to choose the best performing algorithm is selecting the one that produces balanced results during all the phases of the analysis and across all the performance measurements.

So, this study supports that the algorithm capable of producing accurate energy consumption forecasts based on smart meter readings in London is the KNR. This algorithm outperforms any other on the sMAPE and present almost optimal results in MAE. Also, it is an algorithm than doesn't require high computational power since its CC is the best in every step. This feature is crucial in real life working environments where time is limited and more complex models can take up to four minutes to be tested, trained and produce forecasts.

A strong competitor of KNR is KRR. Both algorithms performed equally good in every step of this analysis except from the multivariate rolling forecast step. During this last step KRR produced forecasts with 97.44% sMAPE and 0.34 MAE which are considered the highest values thought the whole study. This performance was caused from the weather variables inserted on the model during the last step. This malfunction of the KRR during multivariate forecasting could be a subject for further research.

Moving to the last section of this study, a discussion about statistical testing follows. A limitation of the DM statistical test is that it cannot be applied on nested models and that's the reason it was applied on the model that didn't contain a rolling forecast. That was an expected result because the performance measures for both models where close. This test was conducted in order to further validate if the accuracy measures chosen in this study are representative of the actual forecasting power of the model, and it can be concluded that the metric used are accurate.

By observing the results obtained in the multivariate model, in which weather data are included a crucial conclusion can be made. All of the models performed slightly better as far as error measurements are concerned but the CC was higher. So, the trade-off between a small improvement in accuracy and the CC is up to the researcher's point of view. Also, this slight increase can become significant if the correct model is chosen. If the KNR algorithm was not included in the selected algorithms one could conclude that weather information doesn't help the forecasting procedure, whereas in this study that the KNR algorithm was able to utilize that extra information and further improve its sMAPE and MAE scores. So, the selection of an algorithm capable of using every information contained in the given data should be every researcher's goal. This outcome follows the same direction as the No Free Lunch theorem (Wolpert, 1996) which states that sophisticated learning algorithms may perform poorly on some specific problems, and a bad models performance on a given dataset doesn't mean it's a bad algorithm in general. That is the reason the conclusion of this study states that the KNR algorithm presented the best results while modelling London's energy consumption, using a different dataset maybe from other countries with different characteristics may found another algorithm performing better than KNR.

A problem faced while reviewing the Machine Learning literature to gain insight about the algorithms used in his study is that a great part of the published papers provide forecasting techniques and report forecasting accuracies without including a comparison with simple forecasting methods or at least with a naïve model used as a benchmark. By doing so the expectations of specific models are superior compared to their actual forecasting power, without any empirical argument proving the aforementioned statement. The same situation was observed in the traditional forecasting techniques during the 1980s (Makridakis, Spiliotis and Assimakopoulos, 2018). At that time traditional forecasting techniques where considered admirable mostly due to their mathematical elegance. Nowadays, there are numerus studies proposing new Machine Learning algorithms or effective ways to optimize novel ones, academic journals should require a benchmark or model comparisons. Also, the data sets contained is such empirical studies should available to researchers that want to replicate the proposed techniques. During my research this was not found implemented on any research article, something that makes the replication of proposed techniques impossible and allows conclusions about learning algorithms and comparisons that may not hold.

10 Conclusion

In recent years the technological advances in smart meters has enabled the ability to collect a vast amount of energy consumption data. By making equitable use of them energy consumption understanding and prediction can be vastly improved.

The proposed framework of this study provides the fundamental steps that This study explores recent Machine Learning approaches with respect to energy consumption forecasting. The main focus was on evaluating if and how the outcomes of the implemented algorithms can benefit energy forecasting. The presented case study compares the forecasting ability of LSTM, SVR, KNR, KRR, MLP and GBR, concluding that KNR outperforms the aforementioned learning algorithms in terms of error percentage and training time. The presented KNR learning algorithm was implemented on household smart meter readings measuring energy consumption in the area of London but it could also be applied for different energy consumption forecasting scenarios. Also, KNR was able to make use of weather data, in a multivariate experimental setup, significantly better that most of the other algorithms studied.

10.1 Future Work

To further evolve the presented study, an anomaly detection technique could be implemented. Anomalous energy consumption is often presented in such datasets for many reasons, one of them being that the customer may be on vacation hence the energy consumption may be lower than expected. As a result, a biased estimation could be made or the models could underperform due to these anomalies. It is believed that by detecting removing such contextual anomalies the models implanted could be further improved. Such techniques were not implemented due to the lack of time, since this study had to be completed during a three-month period.

References

1. Ahmed, N. K. *et al.* (2017) 'An Empirical Comparison of Machine Learning Models for Time Series Forecasting', 4938(April). doi: 10.1080/07474938.2010.481556.
2. Alahakoon, D. and Yu, X. (2016) 'Smart Electricity Meter Data Intelligence for Future Energy Systems: A Survey', *IEEE Transactions on Industrial Informatics*, 12(1), pp. 425–436. doi: 10.1109/TII.2015.2414355.
3. Amadou Boukary, N. and Rivest, F. (2016) *A Comparison of TIme Series Forecasting Learning Algorithms on the Task of Predicting Event Timing*. Available at: https://espace.rmc.ca/handle/11264/883.
4. Anderson, T. W. (1952) 'Asymptotic theory of certain goodness-of-fit criteria based on stochastic processes', *Ann. Math. Stat.*, 23(June).
5. Balkin, S. D. and Ord, J. K. (2000) 'Automatic neural network modeling for univariate time series', *International Journal of Forecasting*, 16(4), pp. 509–515. doi: 10.1016/S0169-2070(00)00072-8.
6. Bengio, Y., Simard, P. and Frasconi, P. (1994) 'Learning Long-Term Dependencies with Gradient Descent is Difficult', *IEEE Transactions on Neural Networks*, 5(2), pp. 157–166. doi: 10.1109/72.279181.
7. Bishop C. (1995) 'Multilayer feedforward networks with a nonpolynomial activation function can approximate any function'.
8. Bontempi, G., Taieb, S. Ben and Borgne, yann-ael le borgne (2003) 'Machine Learning Strategies for Time Series Forecasting Gianluca', *Business Intelligence*, (January), pp. 59–73. doi: 10.1016/B978-155860916-7/50006-3.
9. Box, G. E. P. *et al.* (1964) 'An Analysis of Transformations An Analysis of Transformations', *Analysis*, 26(2), pp. 211–252. doi: 10.2307/2287791.
10. Breiman, L. (2001) 'Random forests', *Machine Learning*, 45(1), pp. 5–32. doi: 10.1023/A:1010933404324.
11. C. M. Douglas, L. J. C. and K. M. (2016) *Time Series Analysis and Forecasting*. doi: 10.1007/978-3-319-28725-6.
12. CACI (2014) *The Acron User Guide*. Available at: https://acorn.caci.co.uk/.
13. Callen, J. L. *et al.* (1996) 'Neural network forecasting of quarterly accounting earnings', *International Journal of Forecasting*, 12(4), pp. 475–482. doi: 10.1016/S0169-2070(96)00706-6.
14. Chalimourda, A., Schölkopf, B. and Smola, A. J. (2004) 'Experimentally optimal ν in support

vector regression for different noise models and parameter settings', *Neural Networks*, 17(1), pp. 127–141. doi: 10.1016/S0893-6080(03)00209-0.

15. Chatfield, C. (2016) 'The Analysis of Time Series: An Introduction'.
16. Cherkassky, V. and Ma, Y. (2004) 'Practical selection of SVM parameters and noise estimation for SVM regression', *Neural Networks*, 17(1), pp. 113–126. doi: 10.1016/S0893-6080(03)00169-2.
17. Chollet, F. (2015) 'Keras'. Available at: https://keras.io.
18. Chris Ding and Xiaofeng He and He, C. D. and X. (2004) 'K-means Clustering via Principal Component Analysis', *Proc. of Int'l Conf. Machine Learning (ICML 2004)*, pp. 225–232.
19. Contreras, J. *et al.* (2003) 'ARIMA Models to Predict Next-Day Electricity Prices', *IEEE Transactions on Power Systems*, 18(3), pp. 1014–1020.
20. Costa, L. de F. and Ceser, R. M. (2009) 'Shape Classification and Analysis: Theory and Practice'.
21. Cottrell, M. *et al.* (1995) 'Neural modeling for time series: a statistical stepwise method for weight eliminationfeedforward networks are universal approximators', *IEEE Transactions on Neural Networks*, 6, pp. 1355–1364.
22. Diebold, F. X. (2015) 'Comparing Predictive Accuracy, Twenty Years Later: A Personal Perspective on the Use and Abuse of Diebold–Mariano Tests', *Journal of Business and Economic Statistics*, 33(1). doi: 10.1080/07350015.2014.983236.
23. Diebold, F. X. and Mariano, R. S. (1995) 'Comparing predictive accuracy', *Journal of Business and Economic Statistics*, 13(3), pp. 253–263. doi: 10.1080/07350015.1995.10524599.
24. Ethem, A. (2014) *Introduction to Machine Learning*, *MIT press*. doi: 10.1007/978-1-62703-748-8_7.
25. Fadhilah, A. R. *et al.* (2009) 'Malaysian day-type load forecasting', *ICEE 2009 - Proceeding 2009 3rd International Conference on Energy and Environment: Advancement Towards Global Sustainability*, pp. 408–411. doi: 10.1109/ICEENVIRON.2009.5398613.
26. Fernando Pérez, B. E. G. (2007) '{IP}ython: A System for Interactive Scientific Computing, Computing in Science and Engineering', *Comput. Sci. Eng.*, 9(3), pp. 21–29.
27. Friedman, J., Hastie, T. and Robert, T. (2007) *The Elements of Statistical Learning*. 2nd edn. Springer-Verlag New York.
28. Fumo, N. (2014) 'A Review on the Basics of Building Energy Estimation', *Renewable and Sustainable Energy Reviews*, 31, pp. 53–60. doi: 10.1016/j.rser.2013.11.040.
29. Funahashi, K. I. (1989) 'On the approximate realization of continuous mappings by neural

networks', *Neural Networks*, 2(3), pp. 183–192. doi: 10.1016/0893-6080(89)90003-8.

30. G. Box and G. Jenkins (1976) *Time Series Analysis: Forecasting and Control*. Englewood Cliffs, N.J Prentice-Hall.
31. Ganganwar, V. (2012) 'An overview of classification algorithms for imbalanced datasets', *International Journal of Emerging Technology and Advanced Engineering*, 2(4), pp. 42–47. Available at: http://www.ijetae.com/files/Volume2Issue4/IJETAE_0412_07.pdf.
32. Gers, F. A., Schraudolph, N. N. and Schmidhuber, J. (2003) 'Learning precise timing with LSTM recurrent networks', *Journal of Machine Learning Research*, 3(1), pp. 115–143. doi: 10.1162/153244303768966139.
33. Gers, F. and Gers, F. (2001) 'Long Short-Term Memory in Recurrent Neural Networks'.
34. Gewers, F. L. *et al*. (2018) 'Principal Component Analysis: A Natural Approach to Data Exploration'. Available at: https://arxiv.org/pdf/1804.02502.pdf.
35. Gorr, W. L. (1994) 'Editorial: Research prospective on neural network forecasting', *International Journal of Forecasting*, 10(1), pp. 1–4. doi: 10.1016/0169-2070(94)90044-2.
36. Grolinger, K. *et al*. (2016) 'Energy forecasting for event venues: Big data and prediction accuracy', *Energy and Buildings*, 112, pp. 222–233. doi: 10.1016/j.enbuild.2015.12.010.
37. Grolinger, K., Capretz, M. A. M. and Seewald, L. (2016) 'Energy Consumption Prediction With Big Data: Balancing Prediction Accuracy and Computational resources', *Proceedings of the IEEE BigData Congress*, (May), pp. 2006–2007.
38. Gybenko, G. (1989) 'Approximation by superposition of sigmoidal functions', *Mathematics of Control, Signals and Systems*, 2(4), pp. 303–314. doi: 10.1016/0196-8858(92)90016-P.
39. Hand, D. J. (2006) 'Classifier Technology and the Illusion of Progress', *Statistical Science*. The Institute of Mathematical Statistics, 21(1), pp. 1–14. doi: 10.1214/088342306000000060.
40. Harris, J. L. (1993) 'Dynamic structural analysis and forecasting of residential electricity consumption', *International Journal of Forecasting*, 9, pp. 437–455.
41. Hill, D. C. and Infield, D. G. (1995) 'Modelled operation of the Shetland Islands Power System comparing computational and human operators' load forecasts', *Transmission and Distribution (former title)*, 142(6), pp. 3–7. doi: 10.1049/ip-gtd:19952248.
42. Hochreiter, J. (1991) 'Untersuchungen zu dynamischen neuronalen Netzen', *Master's thesis, Institut fur Informatik, Technische Universitat, Munchen*, pp. 1–71. Available at: http://people.idsia.ch/~juergen/SeppHochreiter1991ThesisAdvisorSchmidhuber.pdf.
43. Hochreiter, S. and Schmidhuber, J. J. J. J. (1997) 'Long Short-Term Memory', *Neural Computation*, 9(8), pp. 1735–1780. doi: 10.1162/neco.1997.9.8.1735.

44. Hong, T. *et al*. (2016) ‘Probabilistic energy forecasting: Global Energy Forecasting Competition 2014 and beyond’, *International Journal of Forecasting*, 32(3), pp. 896–913. doi: 10.1016/j.ijforecast.2016.02.001.
45. Hong, T., Pinson, P. and Fan, S. (2014) ‘Global energy forecasting competition 2012’, *International Journal of Forecasting*, 30(2), pp. 357–363. doi: 10.1016/j.ijforecast.2013.07.001.
46. Hornik, K., Stinchcombe, M. and White, H. (1989) ‘Multilayer feedforward networks are universal approximator’, *Neural Networks*, 2, pp. 359–366. doi: 0893-6080/89.
47. Hyndman, R. J. and Athanasopoulos, G. (2014) *Forecasting: Principles and Practice*, *OTexts*. doi: 10.1017/CBO9781107415324.004.
48. Ilic, D., Karnouskos, S. and Goncalves da Silva, P. (2012) ‘Sensing in Power Distribution Networks via Large Numbers of Smart Meters’, *2012 3rd IEEE PES International Conference and Exhibition on Innovative Smart Grid Technologies (ISGT Europe),* pp. 1–6. doi: 10.1109/ISGTEurope.2012.6465863.
49. Irish Social Science Data Archive (2012) ‘Commission for energy regulation (cer) smart metering project’. Available at: http://www.ucd.ie/issda/data/%0Dcommissionforenergyregulationcer/.
50. James, G. *et al*. (2013) ‘An Introduction to Statistical Learning’.
51. Jolliffe, I. T. (1986) *Principal Component Analysis*, *Chemometrics Intellig. Lab. Syst.* doi: 10.1007/978-1-4757-1904-8.
52. Karl Friston, J. A. K. T. N. W. P. (2006) ‘Statistical Parametric Mapping: the Analysis of Functional Brain Images’.
53. Kaytez, F. *et al*. (2015) ‘Forecasting Electricity Consumption: A Comparison of Regression Analysis, Neural Networks and Least Squares Support Vector Machines’, *International Journal of Electrical Power and Energy Systems*, 67, pp. 431–438. doi: 10.1016/j.ijepes.2014.12.036.
54. Kim, S. (2011) ‘Forecasting internet traffic by using seasonal GARCH models’, *Journal of Communications and Networks*, 13(6), pp. 621–624. doi: 10.1109/JCN.2011.6157478.
55. Kohavi, R. (1995) *A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection*, *Appears in the International Joint Conference on Articial Intelligence (IJCAI)*. doi: 10.1067/mod.2000.109031.
56. Kong, Q. (2017) *Machine learning 9 - More on Artificial Neural Network*. Available at: http://qingkaikong.blogspot.com/2017/02/machine-learning-9-more-on-artificial.html#.

57. Kwiatkowski, D. *et al.* (1992) ‘Testing the null hypothesis of stationarity against the alternative of a unit root. How sure are we that economic time series have a unit root?’, *Journal of Econometrics*, 54(1–3), pp. 159–178. doi: 10.1016/0304-4076(92)90104-Y.
58. Leshno, M. *et al.* (1993) ‘Multilayer feedforward networks with a nonpolynomial activation function can approximate any function’, *Neural Networks*, 6(6), pp. 861–867. doi: 10.1016/S0893-6080(05)80131-5.
59. MacQueen, J. B. (1967) ‘Some Methods for Classification and Analysis of Multivariate Observations’, *Proc. Berkeley Symposium on Mathematical Statistics and Probability*, 1, pp. 281–297.
60. Makridakis, S., Spiliotis, E. and Assimakopoulos, V. (2018) ‘Statistical and Machine Learning forecasting methods: Concerns and ways forward’, *PLoS ONE*, 13(3). doi: 10.1371/journal.pone.0194889.
61. De Martino Jannuzzi, G. and Schipper, L. (1991) ‘The structure of electricity demand in the Brazilian household sector’, *Energy Policy*, 19(9), pp. 879–891. doi: 10.1016/0301-4215(91)90013-E.
62. MathWorks (2012) *Machine Learning in MATLAB (R2018a)*. Available at: https://uk.mathworks.com/help/stats/machine-learning-in-matlab.html#.
63. McKinney, W. (2010) ‘Data Structures for Statistical Computing in Python’, *Proceedings of the 9th Python in Science Conference*, 1697900(Scipy), pp. 51–56. Available at: http://conference.scipy.org/proceedings/scipy2010/mckinney.html.
64. Medeiros, M. C., Teräsvirta, T. and Gianluigi, R. (2006) ‘Building Neural Network Models for Time Series: A Statistical Approach’, *Journal of Forecasting*, 25, pp. 49–75. doi: 10.1002/for.974.
65. Meinshausen, N. (2006) ‘Quantile Regression Forests’, *Journal of Machine Learning Research*, 7(D24), pp. 983–999. doi: 10.1111/j.1541-0420.2010.01521.x.
66. Miller, C. (2017) ‘The Building Data Genome Project : An open , public data set from non-residential buildings electrical meters’, *Energy Procedia*, 122(February), pp. 439–444. doi: 10.13140/RG.2.2.17155.09765.
67. Monner, D. and Reggia, J. A. (2012) ‘A generalized LSTM-like training algorithm for second-order recurrent neural networks’, *Neural Networks*, 25, pp. 70–83. doi: 10.1016/j.neunet.2011.07.003.
68. Murphy, K. P. (1989) ‘Analysis and evaluation of five short-term load forecasting techniques’, *IEEE Transactions on Power Systems*, 4(4), pp. 1484–1491.

69. Nelson, C. R. and Plosser, C. I. (1982) 'Trends and random walks in macroeconmic time series: Some evidence and implications', *Journal of Monetary Economics*, 10(2), pp. 139–162. doi: 10.1016/0304-3932(82)90012-5.

70. Nelson, M. (1994) 'Can neural networks applied to time series forecasting learn seasonal patterns: an empirical investigation', *Proceedings of the Twenty-Seventh Hawaii International Conference on System Sciences HICSS-94*, pp. 649–655. doi: 10.1109/HICSS.1994.323316.

71. Ng, A. (2012) 'Clustering with the k-means algorithm', *Machine Learning*.

72. No Author (2017a) *How many smart meters are installed in the United States, and who has them?*, *US Energy Information Administrator*. Available at: https://www.eia.gov/tools/faqs/faq.php?id=108&t=3 (Accessed: 29 July 2018).

73. No Author (2017b) *Smart Meters, Quarterly Report to end December*. Great Britain.

74. Oliphant, T. E. (2007) 'SciPy: Open source scientific tools for Python', *Computing in Science and Engineering*, pp. 10–20. doi: 10.1109/MCSE.2007.58.

75. Oliphant, T. E. (2010) *Guide to NumPy*, *Methods*. doi: 10.1016/j.jmoldx.2015.02.001.

76. P. Murphy, K. (2012) *Machine Learning: A Probabilistic Perspective*, *The MIT Press*. doi: 10.1007/SpringerReference_35834.

77. Pecan Street (2012) *Real energy. real customers. in real time.*

78. Pedregosa, F. *et al.* (2011) 'Scikit-learn: Machine learning in Python', *Jmlr.Org*12, (Oct), pp. 2825–2830. doi: 10.1007/s13398-014-0173-7.2.

79. Probst, P., Wright, M. and Boulesteix, A.-L. (2018) 'Hyperparameters and Tuning Strategies for Random Forest'. Available at: http://arxiv.org/abs/1804.03515.

80. Ranjan, M. and Jain, V. K. (1992) 'Modelling of electrical energy consumption in Delhi', *Fuel and Energy Abstracts*, 24, pp. 351–361.

81. Rashed Mohassel, R. *et al.* (2014) 'A survey on Advanced Metering Infrastructure', *International Journal of Electrical Power and Energy Systems*. Elsevier, pp. 473–484. doi: 10.1016/j.ijepes.2014.06.025.

82. Rossum, G. V. (1995) 'Python tutorial, Technical Report CS-R9526'. doi: 10.1111/j.1094-348X.2008.00203_7.x.

83. Schofield, J. R. *et al.* (2015) 'Low Carbon London project : Data from the dynamic time-of-use electricity pricing trial , 2013', (January), pp. 1–5. doi: 10.5255/UKDA-SN-7857-1.

84. Shapiro, S. S. and Wilk, M. B. (1965) 'An Analysis of Variance Test for Normality (Complete Samples)', *Biometrika*, 52(3/4), p. 591. doi: 10.2307/2333709.

85. Sharda, R. and Patil, R. (1992) 'Connectionist approach to time series prediction: an empirical

test', *Journal of Intelligent Manufacturing*, pp. 317–323. doi: 10.1007/BF01577272.

86. Shcherbakov, M. V. *et al.* (2013) *A survey of forecast error measures*, *World Applied Sciences Journal*. doi: 10.5829/idosi.wasj.2013.24.itmies.80032.
87. Srivastava, N. *et al.* (2014) 'Dropout: A Simple Way to Prevent Neural Networks from Overfitting', *Journal of Machine Learning Research*, 15, pp. 1929–1958. doi: 10.1214/12-AOS1000.
88. Thomas Hofmann, B. S. and Smola, A. J. (2008) 'Kernel Methods in Machine Learning', *The Annals of Statistics*, 36(3). doi: 10.2307/25464664.
89. UK Power Networks (2014) *SmartMeter Energy Consumption Data in London Households*, *London Datastore*. Available at: https://data.london.gov.uk/dataset/smartmeter-energy-use-data-in-london-households.
90. VanderPlas, J. (2017) *Python Data Science Handbook*. doi: 10.1017/CBO9781107415324.004.
91. Wang, Q. *et al.* (2017) 'A novel ensemble method for imbalanced data learning: Bagging of extrapolation-SMOTE SVM', *Computational Intelligence and Neuroscience*, 2017. doi: 10.1155/2017/1827016.
92. Wang, Y. *et al.* (2018) 'Review of Smart Meter Data Analytics: Applications, Methodologies, and Challenges', *IEEE Transactions on Smart Grid*. doi: 10.1109/TSG.2018.2818167.
93. Waskom, M. *et al.* (2017) *mwaskom/seaborn: v0.8.1 (September 2017)*. doi: 10.5281/zenodo.883859.
94. Whitle, P. (1951) 'Hypothesis testing in time series analysis', 114(4), pp. 29–30. Available at: http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Hypothesis+Testing+in+Time+Series+Analysis#0.
95. Wolpert, D. H. (1996) 'The Lack of a Priori Distinctions between Learning Algorithms', *Neural Computation*, 8(7), pp. 1341–1390. doi: 10.1162/neco.1996.8.7.1341.
96. Yadav, J. and Sharma, M. (2013) 'A Review of K-mean Algorithm', *International Journal of Engineering Trends and Technology*, 4(7), pp. 2972–2976.
97. Yang, J. *et al.* (2017) 'Decision-Making for Electricity Retailers: A Brief Survey', *IEEE Transactions on Smart Grid*, pp. 1–1. doi: 10.1109/TSG.2017.2651499.
98. You, Y. *et al.* (2018) 'Accurate, Fast and Scalable Kernel Ridge Regression on Parallel and Distributed Systems'. doi: 10.1145/3205289.3205290.
99. Zhang, G. P., Patuwo, B. E. and Hu, M. Y. (2001) 'A simulation study of artificial neural networks for nonlinear time-series forecasting', 28.
100. Zhang, G. P. and Qi, M. (2005) 'Neural network forecasting for seasonal and trend time series',

European Journal of Operational Research, 160(2), pp. 501–514. doi: 10.1016/j.ejor.2003.08.037.

101.Zhang, G. and Patuwo, B. E. (1998) ‘Forecasting with artificial neural networks : The state of the art’, *International Journal of Forecasting*, 14, pp. 35–62. doi: https://doi.org/10.1016/S0169-2070(97)00044-7.

102.Zhang, P. G. (2003) ‘Time series forecasting using a hybrid ARIMA and neural network model’, *Neurocomputing*, 50, pp. 159–175. doi: 10.1016/S0925-2312(01)00702-0.

103.Zhang, X. *et al.* (2013) ‘Comparative Study of Four Time Series Methods in Forecasting Typhoid Fever Incidence in China’, *PLoS ONE*, 8(5). doi: 10.1371/journal.pone.0063116.

104.Zhang, Y., Duchi, J. and Wainwright, M. (2013) ‘Divide and conquer kernel ridge regression’, *Conference on Learning Theory*, 30, pp. 1–26. Available at: http://jmlr.org/proceedings/papers/v30/Zhang13.html.

105.Zhao, H. X. and Magoulès, F. (2012) ‘A review on the prediction of building energy consumption’, *Renewable and Sustainable Energy Reviews*, 16(6), pp. 3586–3592. doi: 10.1016/j.rser.2012.02.049.

106.Zhou, M. *et al.* (2006) ‘Electricity price forecasting with confidence-interval estimation through an extended ARIMA approach’, *IEE Proceedings - Generation, Transmission and Distribution*, 153(2), p. 187. doi: 10.1049/ip-gtd:20045131.