INTERNATIONAL
HELLENIC
UNIVERSITY

Big Data Mining for Smart Cities

Christantonis Konstantinos

SID: 3308170005

SCHOOL OF SCIENCE & TECHNOLOGY

A thesis submitted for the degree of

Master of Science (MSc) in Data Science

DECEMBER 2018

THESSALONIKI – GREECE

Big Data Mining for Smart Cities

Christantonis Konstantinos

SID: 3308170005

Supervisor:	Prof. Christos Tjortjis
Supervising Committee Members:	Dr. Christos Berberidis
	Dr. Agamemnon Baltagiannis

SCHOOL OF SCIENCE & TECHNOLOGY

A thesis submitted for the degree of

Master of Science (MSc) in Data Science

DECEMBER 2018

THESSALONIKI – GREECE

Acknowledgment

I would like to express my complete gratitude to Professor C. Tjortjis, who throughout this work has guided me methodically and turned all my difficulties and concerns into an opportunity for critical thinking. This work would be far more difficult to complete, of course, without the support of my family and some fellow students who contributed mainly to the psychological part.

Abstract

This dissertation was written as a part of the MSc in Data Science at the International Hellenic University.

The accumulation and exploitation of data has brought about great variations in the way people perceive their usefulness. Actually, there is an increasing interest in systematic data acquisition and the parallel extraction of useful information through them in a number of different sectors from medicine to sales. Data analysis techniques can be applied to provide with predictions that could be utilised to the efficient scheduling and operation of electricity generation. The term smart city has been around for more than two decades. Nevertheless, it is increasingly enriched with new ideas and applications, which mainly aim to provide citizens with a better standard of living.

This dissertation focuses on the development and comparison of predictive algorithms under the smart city concept, utilising metered data on predefined time intervals. More specifically, electricity consumption as a total but also as main usages/spaces breakdown and weather data are used to develop, train and test the models. From a technical point of view, a significant comparison between different machine learning algorithms and methodologies is provided. The outcomes prove the necessity of weather data to predict residential electrical consumption. Beside the fact that the available data do not justify the term big data, the scalability of the model is examined in every step.

Christantonis Konstantinos

30 November 2018

Contents

List of tables

List of figures

1 Introduction

The exchange of data and information between systems and users has impelled society to assign the term *smart* in almost everything. From smart phones and watches to smart devices all around the dwelling, it is clear that there is a predisposition of people for intelligent systems, which could positively influence their everyday life. As part of this equation, a more general sense of smartness such as smart homes and smart cities has begun to develop.

So far, there is no universally accepted definition of a smart city. It means different things to different people. The British Standards Institute (BSI), for example, gives a broad definition to the term as "the effective integration of physical, digital and human systems in the built environment to deliver sustainable, prosperous and inclusive future for its citizens". On the other hand, IBM is more focused on a data-driven definition where a smart city is "one that makes optimal use of all the interconnected information available today to better understand and control its operations and optimize the use of limited resources". A generic and shared idea of a smart city, it could be summarized as a municipality that uses information and communication technologies to increase operational efficiency, share information with the public and improve both the quality of government services and citizen welfare.

A smart city is not desired only by citizens but also from governors who aim to convince society that such structures benefit the efficiency of their applied governance. Similarly, businesses that link their products and services with the intelligence factor show the utmost willingness to invest in such structures as they can be proven to bring about a higher quality and therefore a larger profit margin. In such projects, it is common for both governments and enterprises to be involved by offering either expertise or resources.

The required infrastructures are very complex, and it is necessary for the system to ensure a high quality exchange of data and information. When the data fail to be captured and stored in databases on a prerequisite way the whole process is jeopardized. In data mining, especially of real-world scenarios, it is almost impossible to capture the whole 'image' due to the data interoperability, which is, still a major problem. Moreo-

ver, data usually are generated through installed sensors, which quite often fail or deviate from the actual values.

The whole amount of data is transmitted into the so-called Internet of Things, which enables on the devices to exchange data and communicate with each other. Nowadays, the data that are being collected and analysed, are vastly increasing in volume. Big data is not more a theoretical concept but it is increasingly applied to more projects. This is also enhanced by further development and utilization of cloud computing technology, which offers several solutions, such as resources providing or software options through a network, obviously, the Internet.

When the access and administration of data is no longer a problem, then data mining or knowledge discovery techniques, turn these nonsense values into valuable information. In general, data mining is used to obtain new perspectives and capture hidden factors from unexploited information, which is available in the collected data; however, it is also a scientific field that can validate hypotheses and experience-based knowledge.

As a part of this dissertation, the problem of electricity consumption forecasting is examined. The first part is focused on the data analysis approach, applied for the selection of input variables that should be introduced to the predictive models. This preselection process is useful for the development of predictive models since it clarifies which sensors mostly affect the total consumption. The second part is focused on exploiting the gathered weather data to produce building energy consumption forecasting models.

Undoubtedly, electricity consumption is the result of an equation of various factors. More specifically, weather conditions are widely accepted to affect consumer behaviour. Additionally, time of the day is a factor that affects electrical power demand. Thus, a combined model of weather metrics such as temperature, humidity etc., is created with some extra lifestyle factors.

For this purpose, open data from the UMass Trace Repository were used. The *Home* dataset includes both weather and consumption data collected for seven different households for three consecutive years (2014-2016). However, since some homes were not compatible with each other, it was decided that they are excluded. Finally, only three homes were used to assess the repeatability of the models, which actually differ a lot both in size and in consuming behaviours. The houses under examination are located in

Eastern America, Massachusetts. Obviously, weather conditions are similar as the houses are relatively close to each other.

All the experiments on this thesis were executed in Python 3.6 and Weka. In addition, the implementations of all algorithms come from *Scikit-Learn*, which is a machine-learning package, on a predefined form.

2 Smart city

Smart city is a complex concept which does not follow a specific definition. In this chapter, there is a deeper analysis of the term to make it more understandable, through the concerns and opportunities around it.

2.1 Generic Concepts

In general, a city is considered smart if it uses ICT solutions to deal with real life urban challenges [1]. The reason that it is not easy to define it *fully* is the lack of a bound of 'smartness' to compare with. The development of such a city originates from the early urbanization phenomenon. Many estimations [2]-[3] reveal that in the near future more than half the population of the earth will live in urban areas. That phenomenon is even stronger in Europe where only one out of four people will reside outside the urban zones. Although the 'root' is the same, the perception of 'smartness' varies from city to city depending on the existing local infrastructure and culture. It is, widely accepted that smart cities are developed in order to improve citizens' quality of life and effectiveness of governance. In bibliography and on web, one can find many different definitions and projects, which actually have been able to attract the interest of both scientists and ordinary citizens.

The aim of this dissertation to explore and propose solutions on specific aspects of this concept, thus the prerequisite infrastructure is briefly examined. In general, all the data are gathered and form a network, out of sensors signals while these sensors are placed in numerous different objects around a city. They could be in vehicles or onto traffic lights; they could also be in domestic appliances or onto power line poles. The mainspring is stable and clear, save valuable time for citizens and fight against the environmental changes that increasingly lead to ecocide. The secondary or complementary incentive is the economic growth as it emerges.

These sensors connect all the objects that they are placed on with the well-known Internet of Things (IoT). IoT is the network of all these physical devices which enables the interconnection and exchange of data amongst them. Besides sensors, real time data

streams are required. This is achieved by implementing digital networks to manipulate and extract meaningful knowledge through them.

Most of these infrastructures exist, as they have been studied and implemented for over two decades. However, the most recent technology that is still on its early stages and has not reached its full potential yet is called cloud computing. Cloud computing is the delivery of computing services. In this case, cloud computing contributes and solves the most dominant problem of capacity. The huge amount of data that are extracted should lay 'somewhere', in order to be processed and analyzed. Cloud computing, however, is not just a storage service; it also offers networking and computational power on demand.

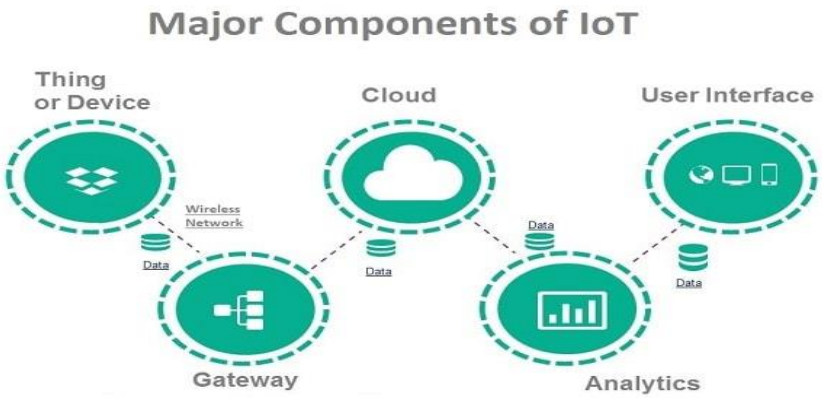


Figure 1: Major Components of IoT

Figure 1 [1] highlights the major components of IoT while it illustrates the continuous flow of data from the early stages of collecting them through single devices to returning meaningful conclusions to citizens. The last part of the chain is the applications that citizens use.

2.2 Applications

Smart cities become smarter due to the enrichment of digital technology, in which a smart city is equipped with different electronic equipment utilized by the various applications, such as street cameras for the surveillance system, sensors for the transportation system, and so on [4]. Smart cities utilize multiple technologies to improve the performance of health, transportation, energy, education, and water services leading to higher levels of comfort for their citizens [5]. All these applications can be divided into two major categories: Citizen-oriented and government-centric.

Actually, all applications are built around citizens and undoubtedly they are the driving force behind the development of such infrastructure. On top of that, however, the individual data gathered by government can assist in terms of e-government and society opinion. These applications that government builds for itself are not affecting citizens directly, but with a collateral way. For example, safety and crime reduction are two related terms, crime reduction leads to safety. Crime reduction is what government wants to acquire by building that kind of applications and safety is the benefit that applies on citizens.

A step further is the involvement of enterprises in this two-way connection between governments and citizens. When governments are not able to find the required resources and are lacking in infrastructure, colossal undertakings might replace them. This leads to huge concerns about, who actually is responsible for the integration and the privacy of the huge amount of collected data.

2.3 Concerns

Even though gathering of people in cities is growing rapidly, this is the major problem that a smart city must overcome. Globally, high urban density seems inevitable to lead to problems including traffic congestion, energy supply and consumption issues, escalation of greenhouse gases emissions [6], unplanned development, lack of basic services, dramatic increase in waste disposal needs and increases in crime and antisocial behaviour [7].

Concerns are numerous and the analysis of threats that a smart city might face form a completely individual subject to study. For this work, only three of the most common concerns are summarized below:

- What happens if the system collapses alone?
- How sensitive are those systems to malicious attacks?
- Has the government accommodated an essential legal ‘framework’ to protect personal data?

Of course, all these concerns are valid, and no one can guarantee a clear and straight answer. For sure, governments and technicians are working to dissolve every single doubt. For example, privacy is strongly governed in Europe by Article 8 of the European Convention on Human Rights, which acts as a benchmark against, which both EU and data protection (DP) rules and nation state laws can be judged [8]. Up to 2018 Eu-

rope is leading the run with the most successful smart cities projects than any other continent. This is mostly happening, as the European citizens are more aware of the dangers that may lurk out when personal data are being accessed by unauthorized third parties [9].

However, according to an extensive study of the Information Commissioner's Office of the United Kingdom [10] on twenty-five data protection regulators around the world, a relative huge number of devices do not properly inform users about the way that their personal data are being stored and manipulated. More specifically, only about four in ten devices informed users with about the way data were collected, used and stored. Not only that, but users were neither informed for actions like deleting personal data off the device or reach a contact for possible privacy concerns.

Another worrying study conducted by a famous American technology company resulted in finding huge security gaps based on IoT Home Security Systems. Below there are the most significant findings as there were listed in [11]:

- *Insufficient authorization:* Majority of systems related to cloud do not demand strong profile passwords (e.g. Six-alphanumeric length) resulting in a weak layer around sensitive data. Neither a prefix number of trials before profile locking existed.
- *Insecure interfaces:* Cloud based interfaces were vulnerable to account harvesting because of account enumeration, weak password policy and lack of account lockout.
- *Privacy concerns:* All the systems under consideration were collecting personal data such as name, address, phone number etc. Based on that are existing the major concerns about account harvesting. Thus, it is revealed that a major upcoming concern regards the privacy of collected video images inside the house.
- *Lack of transport encryption:* The problem mainly appears again in cloud-based connections, which are vulnerable to attacks due to the lack of transport encryption such as SSL/TLS.

In May 2018, the new General Data Protection Regulation came into force. The GDPR legislation is a huge step towards the smart city of the future as it regulates how city governments and administrations collect and use personal data [12]. GDPR drastically interferes on personally identifiable information (PII) and regulates the issue of deleting personal data. So far, governments and administrations were able to store per-

sonal data without time limitations, but now individual data may be deleted on request. The greatest oxymoron is that almost the whole of citizens are concerned about data privacy but only few of them are willing to read the terms of privacy before using an application or a device.

2.4 Infrastructure

As mentioned above not all governments have cultivated the required background for a smart city to be constructed on. The main difficulty for an infrastructure is not its construction but its protection and maintenance. Without a doubt, a smooth operation on the control systems demands a real-time supervision of the total system and continuous update of the effect a possible failure could cause. A flawless infrastructure should also take into consideration the emergency conditions and should be able to deal with them.

For a smart city though, it would be ideal even if its infrastructure were also 'smart'. In such cases, the crucial objective is minimum human interventions. In traditional systems, sensors act individually and are not communicating with each other. An application that evolves is the sensor actuator network (SAN). The SAN is a deployment of several devices equipped with sensors that perform a collaborative measurement process.

A recent proposed method by [13] is to create an interim layer between sensors (SAN) and control systems that will support an aggregation management interface. On such an architecture several nodes (sensors), usually assigned to a specific service, operates as a whole. In case of a partial failure of the system the SAN should cooperate to ensure the regular operation of the system, while providing meaningful information the responders, in order to act.

The flow of data is split in three parts. The first one is the activation of a set of sensors. These sensors can use (depending on infrastructure) any wireless communication module, such as RFID /NFC /3G /Wi-Fi and others. Finally, this value is transferred to a system (open source API). This transfer is wireless, so this network is also called wSAN. Sensors gather information about the physical world, e.g., the environment or physical systems, and transmit the collected data to controllers/actuators through single-hop or multi-hop communications [14].

More specifically as it is explained in [15] there are two ways to gather the sensory data to the control system. The first one is directly through a single-hop wireless link

and alternatively the most remote nodes collaboratively provide their data through other sensors. In the first case, a huge energy consumption and transmission power is observed, as the distance between the sensors and the control system (usually called sink) is increasing. On the other hand, the second approach reduces the wasted transmission power as the distances are shorter, but the intermediate sensors are charged with larger volume of data to transfer so this results in shorter life expectancy for them.

Infrastructure, although is based on similar technologies and architectures, is not stable and differs, depending on the service of interest. In [16] it is described the architecture of G.H.O.S.T, where images are fed to the databases by buses and moving citizens in order to have an extensive view of roads quality, parking spots availability and levels of impurity. In addition, in [17] there is a specialized analysis of electricity management focused on apartment complexes, in which more and more people decide to reside.

Those involved in the construction of infrastructure should also know computing infrastructure should be ready to face unusual and rare events /phenomena that demand greater capacity than the expected, for example natural disasters or big social events (Olympic Games, concerts etc.). The solution to this is the technology of Infrastructure as a Service (IaaS), where IaaS capacity can be utilized on demand.

As in all technologies (such as electricity, telecommunications etc.), the necessary part of the infrastructure is called critical infrastructure. A failure in a small component of a critical infrastructure can lead to chain reactions. In critical infrastructure, control systems are connected via Programmable Logic Controllers (PLC) to Supervisory Control and Data Acquisition systems (SCADA) [13]. It is crucial for the response system to suggest solutions that ensure that the critical parts of the infrastructure will main active.

The deployment of IoT infrastructure has attracted many research teams and a reasonable number of different technologies have been proposed. The key factor that could establish them as more dominant than the others is *interoperability* of information exchange. However, as [18] proposed, for achieving real interoperability the information modelling is essential when the IoT infrastructure is to be formed by a heterogeneous combination of systems, which is the most probable situation in future real scenarios. In general, these are the most established approaches of gathering data. Many researchers also, justifiably relate the quality of data with the infrastructure. For assessing the quali-

ty of data streaming environments as proposed in [19], there are eight different dimensions:

- Accuracy
- Completeness
- Timeliness
- Access Security
- Confidence
- Data volume
- Ease of access
- Interpretability

However, at this point, it is not analysed how to achieve each of the above as it constitutes a work that requires many years of experience and deep understanding on the service of interest.

2.5 Smartest cities

As mentioned earlier, many cities around the globe are on track to turn themselves into 'smarter'. Every year a lot of magazines and newspapers are conducting annual evaluations for the smartest cities on the map based on different criteria. There are cities which "traditionally" achieve near top positions and others that are new entries every year. One of the most reliable annual assessment of smart cities was published recently.

Berrone and Ricart [20] proposed a synthetic indicator CIMI (Cities in motion indexing) which is a function based on other partial indicators that are available to assess each city in total and by category. These partial indicators were governance, urban planning, technology, environment, international outreach, social cohesion, human capital, mobility and transportation, and economy. Top-3 consists of New York-United States, London-United Kingdom and Paris-France, which traditionally and admittedly are in top-5. Unfortunately, the only Greek city that made it to the list was Athens that placed in position 122 out of 165 cities. The detailed matrices show that Athens is relatively creditable in 'technology' and 'international outreach', while 'social cohesion' and 'governance' are of significant low level.

3 Big Data and Data Mining

Big Data as a technology is actual the solution to the difficulties that relational databases are facing when they need to handle and process a giant volume of data. However, the amount of data is considered as Big depending on the available resources.

3.1 Generic Concepts

Nowadays the increase of ICT has resulted in a permanent collection and processing of data around every single corner of the globe. This phenomenon leads to large amounts of data that are beyond of any expectations. Indeed, according to E. Schmidt (Google's former CEO) every two days, humanity creates as much information as it did from the dawn of man through 2003. All these huge amounts of data, which are forming the term 'Big Data', can be analysed in order to improve the decision-making around businesses and strategies. The Figure 2 [2], below, demonstrates amazing facts about the 'landscape' of Big Data.

Figure 2: Landscape of Big Data

At this point, such figures are everywhere around the web, however they lose value rapidly as the landscape is far from static. On the other hand, they form a good indication about the extravagant amount of data that must be exploited.

Initially the three main characteristics of Big Data technology were Variety, Velocity and Volume. Known as '**3 V's**', they were enough to exploit challenges and opportunities based on them. Data systematically were being collected for at least the last two decades. As a rule, those data were stored and processed in a structured way while now the bigger part of collected data is being gathered on an unstructured form. Almost 80 % of generated data come is unstructured (images, video, text etc.) Increasingly also, the unstructured data lead to the need for data streams to replace the classical 'batches' of data. Batch processing is where the processing happens of a block of data that have already been stored over a period [21]. Another point that is not clear for many people, unfamiliar with the technology of Big Data, is the bound in volume of data that finally characterized them as 'Big'. Such a line does not exist and only approximately one could say that usually Big Data are referred in volume of data higher than terabytes. However, as Big Data is turning into a trending term that everyone uses but only a few execute, at the end of the day only the Data Scientists and Analysts can decide if there is a need for adaptation to different approach that the analytics should be carried out.

Later, based on previous '3 V's', freshly established studies proposed that Big Data are consisted of more characteristics, more V's, such as Veracity and Value [22] or Variability and Visualization [23]. All these proposed characteristics are valid and justified but without equal importance. Velocity is of essential significance while Visualizations is supplementary.

The main challenge that the massive volume of data brings to the fore is the lack of capability to include all these data on a single processor or a single disc. The solution on that is called distributed computing and regards a sophisticated architecture. The idea behind distributed computing is simple, when the problem surpasses the capability of the available machine the problem is divided into smaller 'pieces' which are distributed on a group of individual computers that each is responsible to complete a unique required task. All these machines of course communicate with each other in a sort of network. Distributed computing is being applied for over a decade and constitutes a way of computing that has not any competitors. Of course, the distributed computing is not preferred over alternatives for every kind of projects and indeed is a 'painful' task to

achieve. However, computing in parallel saves a lot of valuable time and is often a one-way solution

In the next section the impact of big data is analysed, while it is also highlighted the fact that there is no term ‘smart city’ without the engagement of big data technologies.

3.2 Big Data Value Chain

In the 21st century, things are straight; the value of data has reassessed. Nowadays more and more companies are forming specific data policies that would allow them to gain competitive advantage. For smart cities, however, the volume of data is mostly a challenge to overcome than an opportunity to extract knowledge. That is the main reason there should be clear and well-defined processes in advance. The process that data should go through is complicated enough but does not deviate frequently. This process mostly known as big data value chain, now, identifies the following key high-level activities as analysed in [24]:

- **Data acquisition** is one of the most important processes, which includes all kinds of data preprocessing, from collection and gathering to data cleaning. The general purpose is to transform the raw collected data into the appropriate form in order to apply data analysis, the next chain link. Data acquisition requires a solid and trustworthy infrastructure, which has to minimize the latency of operation, while being able to handle a huge number of transactions.
- **Data analysis** is considered the core of the big data value chain. At this level, data are subject to further transformation such as aggregation and fusion in order to discover the valuable information that may lie ‘behind’ them. Data analysis can be conducted on several different ways. For example, semantic analysis and machine learning are two of the most trending techniques, among others.
- **Data curation** mostly refers to the need for data that meet some standard requirements during their life cycle. Some of the most usual requirements are accessibility, reusability and trustworthiness. More specifically, data curation addresses data quality issues in order to maximize the usability of them. Interestingly, data curation is not static as many might believe but in contrast is being constantly challenged, as there must be an adaptation to the occasional data sources that needs to communicate with.

- **Data storage** is an equally important part of the process, which needs to overcome some well-defined obstacles. Especially big data storage technologies must oblige with the volume and variety of data. An ideal big data storage system should have infinite capacity and should be able to keep up with any kind of data, structured or not. The efficiency on querying should overcome that of relational databases, which, by rule, are slower and more expensive. Briefly some of the state-of-the-art technologies currently are NewSql and NoSql databases.
- **Data usage**, the last component of the chain, includes all the possible ways that data are extracted from the platform of storage. The access and usage of data is achieved through specific tools, mostly through queries and scripting languages depending on the existing data stores, execution engines and APIs [24]. Some indicative areas of big data are decision support, predictive analytics, industry 4.0 and interactive exploration

3.3 Fields of Application and Potential

Data usually bring value, however when the magnitude is huge and the strategy is not clear, there is a danger of being unable to handle it and lose everything. In reality, that is not a deterrent for most projects as the technology has reached already a satisfying level, ensuring efficient tools to minimise the effort and risk.

The number of fields of application is increasing constantly. Obviously it is not intended to list all of them, however, some of the most significant with impressive success are telecommunications, [25], scientific research, society administration, [26] and electricity forecasting [27].

Related to Big Data, the three more important trends according to [28] are:

- Rapid growth of new types of unstructured data
- The rise of cloud computing infrastructure that makes the potential of big data increasingly accessible to more and more businesses and projects
- Rapid development of new capabilities for managing and making sense of data

Fortunately, nowadays there are several tools to implement big data management. Beside the fact that there is not a specific tool that outperforms the rest, some of them gained extreme fame due to their innovative way of operation Big Data in Smart Cities.

Smart city and big data are two interwoven terms. Smart cities, undoubtedly, generate data in great volume and velocity. Big data is the common connection link between smart city and IoT. As all these sensors, social networks, cell phones and web pages generate quintillions of bytes per day [29], the need for more sophisticated data analysis mechanisms is rising. Indeed, as the cost of data storage is decreasing, the main challenge to implement smart city projects remains the development of *time efficient* tools for analysis that would not become a deterrent to the whole of implementation. The volume of data that are being processed most likely justifies the validity of the results. The main goal of such analyses is to reach a level of knowledge that is not expected by intuition. For example, executing a series of actions to conclude that a rainy day metro stations are over-busy or that on New Year's Eve the energy consumption is on peak levels does not contain any smartness. Therefore, for a smart city, a lower level of abstraction is needed, and details are of high significance.

Following the concerns outlined in Chapter 2 regarding the privacy that each citizen wants to maintain, in [30] a few more real world malfunctions are raised. More precisely, there are confronted concerns regarding the failure of critical infrastructures such as e voting and healthcare. What could happen if the application suggested the user to visit a hospital that was wrongly thought to be on call? In a critical situation, who would be accused of the death of a patient who incorrectly entrusted the application? Following in [30] and according to Mina Hsiang, a big data specialist who has been called upon to help the well-known HealthCare.gov (health insurance exchange website) program presented in the United States of America which has encountered many difficulties, designers and governments should "toe a fine line between creepy and incredibly convenient". The main purpose of this comment was to highlight citizens' concerns and the possibility for programs to 'predict' the preferences of residents.

From all the above is clear that the need for every government to initially inform the citizens about the way their privacy is secured and then to give them anytime access on their personal data that are being stored, as well as the right to delete them is of major importance.

3.4 Data Mining

Data mining is a well-established field that started a short earlier than 90's and until the new millennium was considered amongst the most popular. The evolution of data min-

ing, nowadays, is related to the new kinds of data including multimedia, time-series, text, spatiotemporal data and data streams [31].

Scientific and academic communities mostly accept the definition of data mining as the exploration and analysis by automatic or semi-automatic means, of large quantities of data in order to discover meaningful patterns. In simple words, [32] defines data mining as the extraction of implicit, previously unknown, and potentially useful information from data. Data mining is also known as Knowledge Discovery in Databases (KDD), however in reality is just an essential part of it. The Knowledge Discovery process as illustrated in [31] consists of some well-illustrated steps:

- Data cleaning
- Data integration
- Data selection
- Data transformation
- Data mining
- Pattern Evaluation

The last step is called ‘knowledge presentation’ and mostly regards the visualisations, which give insights of higher level to whom is charged to take advantage through the KDD’s results. The first three steps are all under the ‘umbrella’ of data preprocessing and they are not always possible, as they are heavily depended on the structure and characteristics of the available data.

Data mining aims to extract the hidden patterns that are unknown at first place. Through the so-called data mining functionalities, the data mining tasks are being classified as descriptive and predictive.

- Descriptive mining tasks characterize the general properties of the data in the database [33]
- Predictive mining tasks perform inference on the current data in order to make predictions [33]

In simpler words, predictive tasks predict the value of unknown variables based on some already known attributes. Description, continuing, aims to associate different instances in order to form human-interpretable patterns and simplify the examination of data profiles.

3.4.1 Functionalities

Below, briefly, all these functionalities are listed:

- *Class/Concept Descriptions.* All the data entries set to be divided to concepts and classes. Classes might be a set of products while concepts can refer to customer profiles. These descriptions are achieved by data characterization and data discrimination.
- *Association Analysis.* It consists of the discovery of association rules. It focuses on transactions and seeks for the association of situation X while situation Y occurs. For that kind of analysis, two major metrics are used. *Support*, which is an indication of the frequency each item has into the dataset and *Confidence* which shows how often a result (rule) had been found true.
- *Classification.* It is a widely used concept, which is applied in order to discover a function that could distinguish the available data in classes. Most of the times the classes are two as they describe the existence of a situation or not (binary) however also multi-labelled classification exists. Classification is a supervised technique, which means that it is compulsory to have labelled available data in advance. Based on these class-known data, a classifier is built (trained) in order to assign the unknown upcoming data into a class.
- *Clustering.* It refers to the 'chopping' of the dataset into smaller subsets based on similarity. Those similarities are mostly expressed in terms of distance. The results hardly can be evaluated as each approach and metric may provide very different results. Unlike classification, clustering is an unsupervised method that analyses data without given the class label.

Outlier Analysis. Also known as, anomaly detection is the process to reach and identify all these individual data that deviate from other observations on dataset that follow a generic valid behaviour. There are two types of outliers, the univariate and the multivariate. Univariate outliers can be found when looking at a distribution of values in a single feature space, while multivariate outliers can be found in an n-dimensional space [34].

3.5 Challenges in Data Mining

As already discussed, Data Mining is a well-structured procedure with distinct levels that are attracting more and more scientists in order to extract useful conclusions out of data manipulation. That procedure however is far from an easy task to execute, as there

are numerous challenges that need to be overpassed. Each project faces different challenges and each implementation identical to another or not, cannot guarantee meaningful results. Below the major challenges of Data Mining, both from technical and theoretical view are listed:

- *Noisy Data.* To perform Data Mining the most essential requirement is what is called Clean Data. Indeed, this is the greatest and most time-consuming challenge. Although data are being collected and stored in semi and full-automated ways, it is inevitable to prevent noise. Noisy Data, thus, are all the Data that by human or sensor mistake were stored inaccurately.
- *Data silos and distribution.* The data to be processed may be stored in different locations and the gathering out of these individuals systems is usually a difficult task. A common challenge across multiple heterogeneous data sources is the absence of interoperability. Different data formats and data isolation cause these silos to raise numerous problems. Beside the lack of communication, the required infrastructure should avoid transferring data across systems, as it is a slow process.
- *Scalability and efficiency.* Training an algorithm on a common-sized dataset is turning into a standardised procedure while applying traditional algorithms into huge datasets of petabytes is turning into a non-functional process. Algorithms should be adjusted on the specific needs that each project requires and its individual characteristics. For example, as mentioned in [35], algorithms should reorganize computations in order to reuse intermediate results without storing them.
- *Complexity*. The data as mentioned above come in different formats. Some formats are much harder to manage but with greater potential. The more complex data are images, raw text, audio and video data. That kind of data only recently started being manipulated and indeed seem challenging and promising.
- *Privacy and Security.* The information that may be extracted through the Data Mining leads to several legal and ethical issues. The legal and policy foundation is based on specified protocols, which establish penalization for data security and privacy Government Act [36]. The acquired knowledge often overcomes the consensus of citizens and customers. Without their permission, the collected data cannot be processed in such ways.
- *Data visualisation*. Another technical challenge is visualisation of the results. Although it is of secondary importance, it affects the most crucial factor of the ‘chain’,

users. The high dimensionality is usually the biggest ‘curse’ in both executing and visualisation.

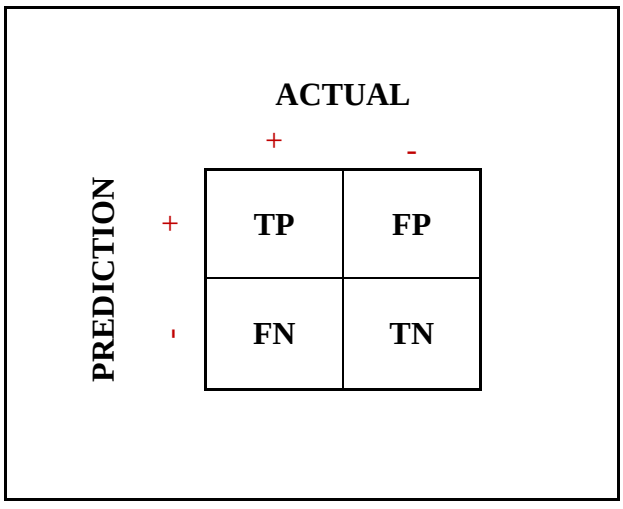
3.6 Evaluation and Metrics

Data mining may extract countless rules and patterns; however, there is no interest for every extracted rule. The rules of interest are only those that surpass a pre-defined threshold. Those patterns also follow a confidence level and it is up to the user to define it and extract only the stronger ones.

3.6.1 Classification

Classification as a major sub-field of data mining relies on several metrics that aim to highlight different aspects of the same problem. Most of these metrics have some components in common and actually, in an attempt to distinct them it is built what is widely called ‘confusion matrix’. Confusion matrix works even for more than two classes and several metrics rely on it.

Table 1: Draft Confusion Matrix



Above in Table 3, a draft matrix illustrates the components of the existing metrics. More specifically confusion matrix is a 2x2 matrix (in case of two classes) that shows the results of the classified instances in comparison with the ground truth. TP stands for True Positive, FP for False Positive, FN for False Negative and TN for True Negative. For example, if the classification task is to predict the students who will pass the exam,

FN shows the number of students that were false predicted to not pass but eventually pass. The most dominant metrics are being analysed below:

Accuracy is the number of the correct predictions over all the predictions that the model made, so is calculated by the following formula

$$\text{Accuracy} = \frac{TP+TN}{TP+FP+FN+TN}$$

In many real life problems, however, the size of classes is imbalanced and Accuracy is not a proper metric to use. A classic example around the bibliography is the prediction of cancer in a number of patients. When it is known that 10 out of 1000 of total patients are indeed stroke by cancer and the model predict that none of them suffers, that means that our model has Accuracy of 99% but did not manage to prevent a critical situation, thus is actually useless. Nevertheless, for balanced classes it is the most popular metric and an extremely good indication of the classifier performance.

Precision is the number of the correct positively classified instances over all the positively classified instances (correct or not) that the model made, so it is calculated by the following formula:

$$\text{Precision} = \frac{TP}{TP+FP}$$

In that case, the problem of imbalanced classes is overpassed as the precision remains low. In general, Precision is preferred when the aim is to evaluate with respect to false positives. So, Precision alerts for the number of relevant instances that our model managed, among the retrieved instances.

Recall (or Sensitivity) is the number of the correct positively classified instances of the model, over all the actual positive instances. The formula is:

$$\text{Recall} = \frac{TP}{TP+FN}$$

Similarly, to Precision, Recall is another measure of relevance that is preferred when the aim is to evaluate with respect to false negatives. It is also a common scenario to achieve high recall while precision is low, and this is another trap that always should be considered. In the previous example with cancer detection, if the model would classify all the patients with cancer then the recall would be one, which is ideal however, the

precision would only be 0.05. That trade-off varies heavily depending on the metric that is chosen to be maximized. In any case, the combination of those two is always a solution that could erase any doubts

F1-score is the metric that combines Precision and Recall. In fact, calculating the arithmetic mean of those two metrics is not a solid solution in many imbalanced classification tasks. Instead, F-test introduced to give a more reliable measure for the evaluation of the results and its formula is:

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall}$$

Mathematically this formula offers a confidence that when these two metrics differ a lot F1 tends to stay close to the lower value. That prevents the model automatically from considering a non-sense average, which could lead to misunderstandings.

3.6.2 Regression

Regression is the task of predicting a continuous quantity instead of a class and thus is generally considered as more informative type of prediction. Usually it requires higher amount of data to be accurate.

Variables

There are many names for a regression's dependent variable. It may be called an outcome variable, criterion variable, endogenous variable, or regressand. The independent variables can be called exogenous variables, predictor variables, or regressors [37]. In this work, none of them will be used and so they will be referred to as dependent and independent variables.

Regression Metrics

Unlike classification tasks, regression predicts continuous values instead of classes and thus it uses different evaluation metrics. In general, there is a plethora of criteria by which the formed models can be compared. On this work only the most famous will be used which are described below.

RMSE

Root mean squared error or RMSE for short, measures the error rate and illustrates the standard deviation of the residuals (prediction errors) from the regression line. It indicates so the concentration of data around the line of best fit [38]. The difference between actual and predicted values is what was referred above as residuals. RMSE is given by the formula:

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

Where $\hat{y}$ is the predicted value,

while y the observed (actual). RMSE is a negatively oriented score which obviously cannot be negative while is measured in the same units as the data. The RMSE is more sensitive than other measures to the occasional large error as the squaring process gives disproportionate weight to very large errors [39]. On this work, RMSE if preferred over mean absolute error (MAE) or mean absolute percentage error (MAPE). Indeed, there are still numerous controversies around the bibliography about the best metric to evaluate regression models as they were explained in [40] however RMSE is the most common metric and it was also chosen due to comparability and the high emphasis that puts on large errors (outliers). The basic assumption of RMSE is that errors are unbiased and follow a normal distribution. The square root, in addition, ensures a more robust result as the positive and negative errors are not cancelling each other. RMSE in general is considered the estimate of the noise in the system.

R^2 – Adjusted R^2

R-squared is a statistical measure that represents the proportion of the variance for a dependent variable that is explained by an independent variable. In other words, it is used to assess the goodness of fit of the regression model against a baseline model [41]. A baseline model is a model that takes no account of independent variables and their values, but instead predicts the dependent variable always through its mean value. R-squared is given by the formula:

$$R^2 = 1 - \frac{Explained\ Variation}{Total\ Variation}$$

Where explained variation (or sum of squared errors of regression model, SSE) is given by the formula:

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

and the total variation (or sum of squared errors of baseline model , SST) is given by the formula:

$$SST = \sum_{i=1}^{n}(y_i - \bar{y}_i)^2$$

Obviously $\hat{y}$ is the predicted value and $\bar{y}$ is the mean value a baseline model 'predicts'.

Above equation, makes clear that if the regression model is equal to the baseline the R-squared is equal to zero. In a more prolific and algorithmic way to compute the R-squared, it needs to fit the data points and find the line of best fit. In addition, since the predicted values are calculated, the actual values should be subtracted and then the result to be squared. Therefore, the explained variation SSE equals the sum of all these squared errors. The total variation SST is calculated through the sum of subtraction of the average actual value from the predicted values. If R-squared equals to 0.7 that means that 70% of the variation of dependent variable is explained through the independent variables that compose the model.

However, R-squared does not guarantees the required scale in terms of features. One of its major pitfalls is that always increases as more independent variables are added to model even if those variables are useless. This is easily explained through the equation ## above because as the fraction SSE/SST decreases the R-squared increases and in order for the fraction to decrease the SSE should decrease. Therefore, adding more explanatory variables decreases the SSE because adding more data points makes it easier for the model to fit them and reduce the sum of squared error [42]. R squared value increases if we increase the number of independent variables. Adjusted R-square increases only if a significant variable is added [43].

For that reason, in numerous projects adjusted R-squared has replaced the simple R-squared. This is because adjusted R-squared prevents this phenomenon from happening

with penalizing the adding of non-informative variables. The formula of adjusted R-squared is:

$$\bar{R}^2 = 1 - (1 - R^2) * \frac{(n-1)}{n-p-1}$$

Where n is the number of instances in the dataset (data points) and p is the number of independent variables (features)

Even though adjusted R-squared is a unitless statistic, there is no absolute standard for what is a "good" value. A regression model fitted to non-stationary time series data can have an adjusted R-squared of 99% and yet be inferior to a simple random walk model [44].

In particular, time series analysis is an ordered sequence of values of a variable at equally spaced time intervals. In the need for making predictions, there are two techniques to counter seasonality. First trying deseasonalizing based on moving average or usage of dummy variables to isolate those effects [45].

3.7 Hyper-parameters

All the different algorithms that are examined in 8.4 are subject to an extensive search for the most efficient hyper-parameters. Hyper-parameters are the parameters that the user should define before the training phase. In general, the mathematical models are based on parameters that are being self-taught by the algorithm through the data. The parameters that cannot be taught through data and should be specified from users are known as hyper-parameters. On this sub-chapter, there is a brief analysis of what each hyper-parameter is and the most basic characteristics of the meaning for each algorithm.

SVM

The kernel represents the function that is used in order to avoid an explosion in the dimensionality that could be happened due to construction of polynomial in an n-dimensional space. Through the kernel function, training data appear in the form of scalar products $x_i^T x_j$ [46]. These products will be replaced by scalar products in a feature space F, through the kernel function K $(x_i, y_i) = \Phi^T(x_i)\Phi(x_j)$. The kernel functions are

several, and on that tuning it was pointed out that the best was the default function of SVM in Scikit-learn, named radial basis function (RBF for short)

$$K(x, x') = \exp(-\frac{\|x - x'\|^2}{2\sigma^2})$$

where || symbol means the Euclidean distance, while σ is a free parameter.

Gamma (Greek letter - γ) is actual a parameter that contains the σ value of kernel under the formula:

$$\gamma = \frac{1}{2\sigma^2}$$

Generally, gamma is a kernel coefficient for different kernels (rbf, polynomial, sigmoid). If the value of gamma is increased, that results in a more specific fit around the data which might cause overfitting and put the generalisation of the model in peril.

For SVM the parameter C commonly known as the penalty of the error term represents the trade-off between smooth decision boundary and classifying the training points correctly [47].

Random Forest

The *n_estimators* parameter illustrates the number of the individual trees that the classifier constructed. Generally, a larger value leads towards better results however, it also increases the computation cost in a linear way. This was also pointed out in Figure 5 and Figure 6.

Min_samples_split and min_samples_leaf as parameters express the minimum number of samples required for an internal node to be split and to stand as a leaf node. As clarified in Scikit learn's documentation 'A split point at any depth will only be considered if it leaves at least *min_samples_leaf* training samples in each of the left and right branches'.

Max_features parameter requires a more complex tuning. Initially it represents the number of features when looking for best split however, options in the implementation of scikit learn package vary. The options under examination for this work are the following:

- *Sqrt* equals the square root of the number of features
- *Log2* equals the logarithm of the number of features

SGD

Stochastic gradient descent is not actually a classifier but mostly a technique. SGD as an optimization technique is preferred on large datasets, however it is implemented on this work to point out the fact that each algorithm's hyper-parameters should be carefully chosen. In SGD, only a small portion of the training data is used and sometimes even only one instance. Thus, it is preferred as it is much faster than traditional implementations; however, it is not as accurate as the standard gradient descent process.

Alpha is the number which indicates the magnitude of step that gradient moves. Indeed, it is a constant that keeps being multiplied with the regularization term in order to reach convergence.

N_iter again is just the number of iterations through the training data, also known as epochs in machine learning field.

Finally, *loss* function as already has be explained represents the object that the models needs to modify and that evaluates the given dataset.

Logistic Regression

Even though logistic regression was performed through the implementation of SGD techniques, it was chosen to be analysed a bit further.

Max_iter parameter apparently specifies the maximum number of iterations that solver can execute in order to converge. A major threat that could cause failure on convergence is the high correlation between inputs and the sparsity of data [48]. Indeed, that model includes many zero values due to dummying transformation of some features and undoubtedly, multicollinearity is present amongst temperature and apparent temperature.

For the penalty parameter, it is necessary to understand the concept of regularization, which is a technique to counter overfitting. Regularization in simple words adds a factor of penalty in the cost function as the complexity of model increases. That added factor to regularize the function is:

$$\lambda * \sum_{i=1}^{n} \beta_i^2$$

Eventually, L2 is the sum of the square of the weights, while L1 is just the sum of the weights. On [49] there is a nice and simple explanatory table, which summarizes the differences between their properties.

Table 2: Basic differences between L1-L2 regularizations

L2 regularization	L1 regularization
Computational efficient due to having analytical solutions	Computational inefficient on non-sparse cases
Non-sparse outputs	Sparse outputs
No feature selection	Built-in feature selection

The λ symbol on the equation above indicates the strength of regularization. If λ is very large, that may lead to under-fitting because of allocating too much weight. The key difference between these techniques is that L1 shrinks the less important feature's coefficient to zero thus, removing some feature altogether. Therefore, this works well for feature selection in case we have a huge number of features [50]. Parameter C expresses exactly the magnitude of this factor in an inverse way. In particular C=1/λ, so smaller values specify stronger regularization.

It is therefore clear that it is impossible to know in advance the combination of these parameters to ensure maximum predictability. Often the huge amount of data prevents even field experts from capturing a clear picture for applying a specific strategy. The trial and error scheme in the fields of predictive analytics and machine learning is the most reliable approach. The validity of the results often depends on the range of experiments.

4 Related Work

The most effective case when performing data mining is to form a clear goal and a specific approach. Detection of phenomena and analysis around citizens' habits are the traditional actions of interest. Recently, with the burst of *machine learning* technology in all sectors, data mining is constantly increasing on predictive analytics. Prediction is based on the existence of data that describe prior situations and mathematical models that focus on predicting one unit (variable) based on the rest units.

4.1 Smart*

*Smart** is an extremely valid project that can be accessed through the *UMassTrace* repository and offers real datasets regarding electrical consumption and weather data as they were recorded on more than 400 anonymous homes. Furthermore, datasets about Solar panels and Sun dances are also available.

Until now, many research teams have relied on data provided by the Smart* project with different goals and directions. In [51] weather and energy data were processed to predict the latitude and longitude of a smart meter that collect the data, as all these energy data are collected anonymously. In addition, in a more market-oriented approach of equal significance, [52] introduced an intelligent charging system called SmartCharge whose goal is to decrease the electrical bills by shifting consumptions to lower price periods. Similarly, in [53] SmartCap – a system for monitoring and controlling electric loads- tries through scheduling algorithms to flatten the electricity demand.

4.2 Applications

Besides the prior published work based on the particular *Smart** project it is extremely interesting to trace back into significant approaches around the general topic of electrical management systems, such as [54], and the prediction of consumption. More than ten years earlier, [55] compared different structures of ANN's on forecasting, concluding on a 2-hidden layer instruction of form 12-16-16-1 that achieved the highest per-

formance. ANN's are also examined and compared in [56] with SVM algorithm in an extensive review of building electrical energy consumption bibliography that results in superiority of least squares support vector machine algorithm (LsSVM). In addition, [57]-[58]-[59] propose to first build customer profiles or assign some already known and then try to predict consumption based on them.

In [60] the profiles are pre-defined based on the total consumption and thus a significant decrease on the daily expenses in electricity is achieved. Another analysis under the umbrella of flattening in consumption is the outlier analysis and the focus on anomaly detection as it considered a major factor that affects the consumers' pocket. Since the market's linearization, the kilowatt-hour price changes regularly and by understanding and preventing such phenomena it is expected to minimize unreasonable extra costs. Such a research was conducted by [61], where ARIMA methodology was applied.

It is clear that load forecasting is a well-studied subject, which has influenced several researchers to try different techniques and approaches. However, there is still a lot of room for improvement as well as several unexplored aspects in such a multidimensional problem.

The [62] presents three time-based approaches around the forecast as well as their characteristics; short-term, medium term and long term. The STLF could be used for reducing costs and secure operation of power systems. In MTLF, the interest focuses on normal operation, while LTLF is studied to ensure safer investments and long-term planning in general. A very interesting approach of load forecasting in distribution system is presented in [63] where Principal Components Analysis is applied on multi-linear regression on MTLF.

Regarding weather factor, [64] represents the responses of energy demand due to climate change in Massachusetts. The parameters that are being used describe the heating degree-day (HDD), the hours of daylight and the electricity price in a monthly scale for both residential and commercial sector. The study concludes that 'energy demand in Massachusetts is sensitive to temperature' while the average number of days exceeding 90°F will rise to double by 2030.

The results of prediction cannot always be that accurate even if weather indeed affects largely the electricity consuming behaviour. As explained in [63] electrical energy demand has a high non-linear behaviour, thus accurate predictions cannot be guaran-

teed. Another factor that affects negatively the prediction accuracy is the continuous pressure for better living standards [65] on a disproportionate rate. Indeed, according to [66] where the residential electrical consumption in Brazil was analysed, the increase in electricity demand was faster than the income. On a similar study for a very different climate, [67] resulted in similar results regarding temperature, however as it was highlighted 'relative humidity is not having significant impact on the energy consumption'.

The base period of forecasting, briefly, affects the complexity of the problem that needs to be modelled. The most studies focus on the STLF as it is a more difficult task due to the noisy effect of environmental factors. As stated in [68] the electric power consumption is growing rapidly and introduces a higher level of randomness due to the increasing effect of environmental and human behaviour. In addition, it is important to understand that like many time-series problems, load forecasting also reflects a seasonality and cyclic component, which leads many researchers to the vertical decomposition throughout the year. The load pattern is a non-stationary time-series problem and thus needs to be carefully fragmentized.

4.3 Strategy

On this work besides weather data, the constructed model includes also usage patterns. As long as the target is to predict the fluctuation of consumption individually for each *Home*, the consuming behaviour should also be considered. Authors in [62] state that 'Electric demand is often considered as a function of weather variables and human social activities'. More specific, typical families have cycles of consumption on daily and weekly basis. In general, families use the laundry or any other appliance X times per week. If the consumption stays low for a consecutive number of days, it is more likely that next days will show a rise in total consumption. Similar to that, the previous day consumption should also be a factor to predict the next day. It is expected for a house that consume a higher than average amount of electricity, the next day would result in a lower consumption.

To ensure such a model it is essential to guarantee that there is a constant tracking of the total consumption in each home. Unfortunately, the smart metering is a newly deployed technology that still must surpass numerous challenges and failures. Therefore, the first part of this thesis focuses on an alternative way to reach the total consumption if the central sensor fails. A very interesting survey [69] lists all the concerns and possi-

ble ways that a smart grid can fail. To a point, smart metering has not won its predecessor in reliability and success and so safeguards are an essential part. Through an analysis, it is also desired to capture the most dominant sensors around the houses and examine how reliable a prediction of the total consumption could be through a very small subset of essential sensors. Although all three houses have sensors installed in different locations and the results will not have a common benchmark, it is expected that a small subset of sensors could capture up to 80% of the original precision. It should also be made clear that Home's B sensors are divided into two separated groups, which share some common sensors and there is a comparison between them in terms of predicting accuracy of the total consumption. The sensors that will be tested in the so-called subset of sensors will be those that reflect the higher correlation.

5 Problem definition

Nowadays, besides constructing general-purpose smart projects, smart cities also focus on the development of smart homes. Smart homes transmit real time data generated by installed sensors to inform the residents or the organizations of interest about the perpetual behaviour of the home /apartment /building.

5.1 Electricity pricing

Electricity price is one of the most important elements of the electric power market [70]. The price of electricity is affected by many factors and initially differs based on the type of customer. The most common categories are:

- Residential
- Commercial
- Industrial
- Transportation

This work focuses, as already discussed, on the *Residential* area; however, all these categories share some common pricing strategies and the affecting factors do not differ dramatically. The maintenance and operation of power plants in combination with the distribution of electricity are such factors, although, they do not reflect a fixed cost for the provider. According to the Energy Information Administration in United States [71] some of the key factors that influence and interpret the fluctuations in electricity price are:

- *Fuels.* The price of fuels affects the cost to generate electricity and that is happening because a higher electricity demand can also increase the demand and price of fuels.
- *Power plants.* Power plants show lower deviations in cost however beside construction, the maintenance and operating costs are still not fixed.
- *Transmission and Distribution.* As in every system, distribution systems can face unexpected damages and require a frequent repair.
- *Weather Conditions.* Depending on the generation system of the provider, strong winds affect the wind turbines, while rain can be used for low-cost

hydropower generation. In summer, of course, demand is higher as there is a need for more mild temperatures through cooling appliances.

- *Regulations.* In general, regulations is not a common factor for providers. In certain countries, which are more developed, the need for stronger regulations that ensure the customers' satisfaction has already raised. In U.S, in some states the prices are fully regulated by State while others have a combination of regulated and not regulated prices.

Therefore, the price policy that each provider follows is not common. Market liberalization has turned this sector in a very competitive field, where efficient pricing even in hourly scale is becoming more and more necessary. Based on the conclusion of [72] 'hourly electricity prices do not follow a time series process but are in fact a panel of 24 cross-sectional hours that vary from day to day'.

5.2 Problem

Based on the uncertainty, as it was described above, scientists have already started looking for effective ways to forecast the electricity consumption and therefore the electricity price. Probably the most challenging part is to obtain reasonable data regarding the area of interest. This gathering of data leads to valuable results, which mainly affect the life of residents in positive manners but also allows providers to reschedule their generating and distributing plans. On these days, providers are turning to smart grids that focus on real-time pricing or critical peak rebate. In essence, this design aims to reward citizens who shift their consumption volume to off-peak hours, by reducing the kWh price. The homes under examination on this work are located in Massachusetts. Indeed, according to [73] those kinds of smart pricings are already increasing in the USA and more specifically in states like California and Massachusetts, "this is actually being mandated by the state legislatures".

This work analyses an electricity consumption dataset with the primary purpose of predicting the total consumption in a house through installed meters while attempting to mine deeper into the data in order to extract any useful information that could be used to lead in previously unknown knowledge. Forecasting of consumption is not only conducted to inform citizens but also can give insights on power providers about habits and consumer profiles or aid governments to reform strategies in order to apply more *eco-*

friendly regulations. The energy consumption predicting in a particular building is usually influenced by many factors, such as electrical appliances or devices inside the building, geographical location of the building, and as well as the time range of building operation [56]. Occupancy is also such a factor but is not easy to record it in highly operating buildings. The prediction of total electricity consumption through different subsets of meters might not seem reasonable, as most of the times there is a central meter for this purpose. However, it is necessary to provide a secondary way of accessing total electricity consumption (in terms of recovery), as the central meter may collapse or being hacked by cunning consumers. In general, citizens have a decent understanding about the appliances that consume higher rates of electricity; however, the constant change in price prevents the creation of a clear consumption plan. For example, air-conditioning represents the biggest part of electrical energy consumption in residential buildings [74], while according to [75] electrical energy consumption increases on summer months over 2.5 % because of the rise on temperature.

5.3 Dataset Description

Two interrelated sets of data are used in this work from the *Smart** project. The *Smart** project, as stated on their website, seeks to optimize energy consumption in homes, with specific attention to modern 'smart homes' and the new opportunities made available by such homes. As already explained in introduction, initially there were recorded data for seven homes, however the structure was not matching for all of them. The data concerned electricity consumption for three consecutive years, however, two homes of them contain values only for the last year and two of them do not provide the total consumption, so it is impossible to serve this work's goal. As a result, all four of them were excluded from the procedure and only three were kept. These homes are labelled from the project alphabetically and specifically the remaining ones were Home B, Home C and Home F. For the rest of this work the homes will be distinct based on these names. The first dataset contains consuming data from several sensors around the home every thirty minutes that record the consumption in kilowatts. Those sensors are separated into two sets –meter one and meter two. Those meters have some sensors in common. The goal out of this dataset is to compare the two meters where this is possible in order to understand which of them contains the more reasonable sensors. It is also desired to understand if there are sensors that even if they do not represent a big part of the

total consumption, their deviation affects the prediction model. The second dataset contains hourly weather data for the specific homes. Although it is a well-structured set of data, only the data that match the homes from the first are taken into consideration. Below in table 4 all the available weather data are listed:

Table 3: Available weather metrics

Temperature	Icon	Visibility
Summary	Apparent Temperature	Pressure
Wind Speed	Cloud Cover	Wind Bearing
Dew Point	Precipitation Intensity	Precipitation Probability

It is worth noting that in the first dataset the installed sensors are not the same and actually differ in every single home. For that reason, they will be explained individually during the process. The most significant part of the analysis in Chapter 7 is that since the available data was limited it was decided to rebuild the target of the model. In essence, it is not expected for such a small dataset (1096rows x 12columns) to achieve any significant results in a regression task. Thus, the problem is turning into binary classification. Indeed, the consumption is labelled around a mean value as *High* or *Low*. This is a closer approach to the real world where citizens are mainly interested in more simple predictions.

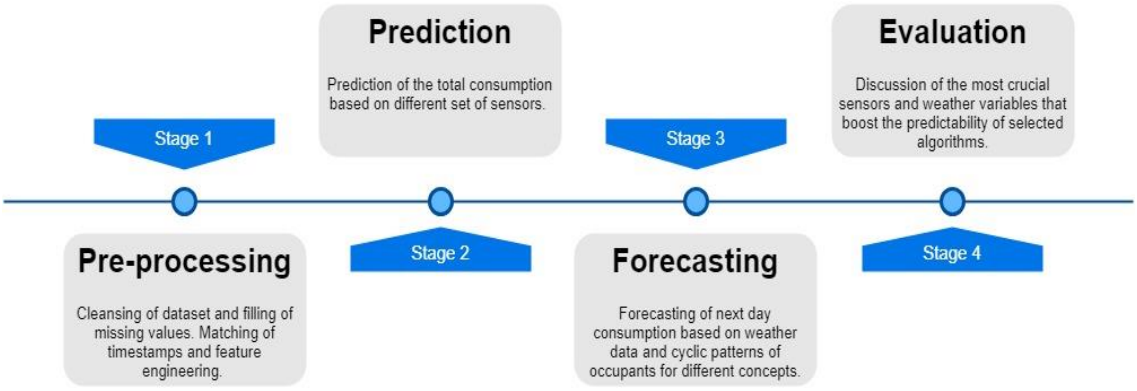


Figure 3 Problem's flow chart

6 Smart metering

As described in Chapter 4, datasets most often require a preparation to meet all the validation criteria. If these are resulting from the union of some smaller sets, it is easy to understand that there is a need for perfect match between them. Usually datasets suffer from missing values and redundant data.

6.1 Pre-processing

As in any project, the pre-processing is the initial task that largely determines its success. The datasets that are being used in that case *do* contain missing values, while contain only numerical values besides the Date & Time attribute (a data sample can be seen in Appendices). The strategy for missing values was not unified for all purposes. More specifically, for that part of the work it was decided to exclude the whole instances out of the dataset if a sensor had failed. All these numerical values regard the recorded consumption of each installed sensor at a specific timestamp. In addition, all the redundant attributes such as *Generated* power, *Grid* load or *Solar* panels were removed.

6.2 Home B

Home B as was described in [76] is a huge house across two stories with eight rooms and four full-time occupants. It is roughly 1700 square feet and it contains a central A/C as well as a gas-powered heating system. In addition, the implementations of all algorithms come from *Scikit-Learn* on a predefined form.

6.2.1 Consumption patterns

The first step in analysis of every home is to understand the profile of residents and their consuming behaviour patterns. Similarly, in Home B the first approach targets to spot the peaks in demanding throughout the specific time intervals during the day. Below, Figure 3 gives a clear indication about the total consumption in each year individually.

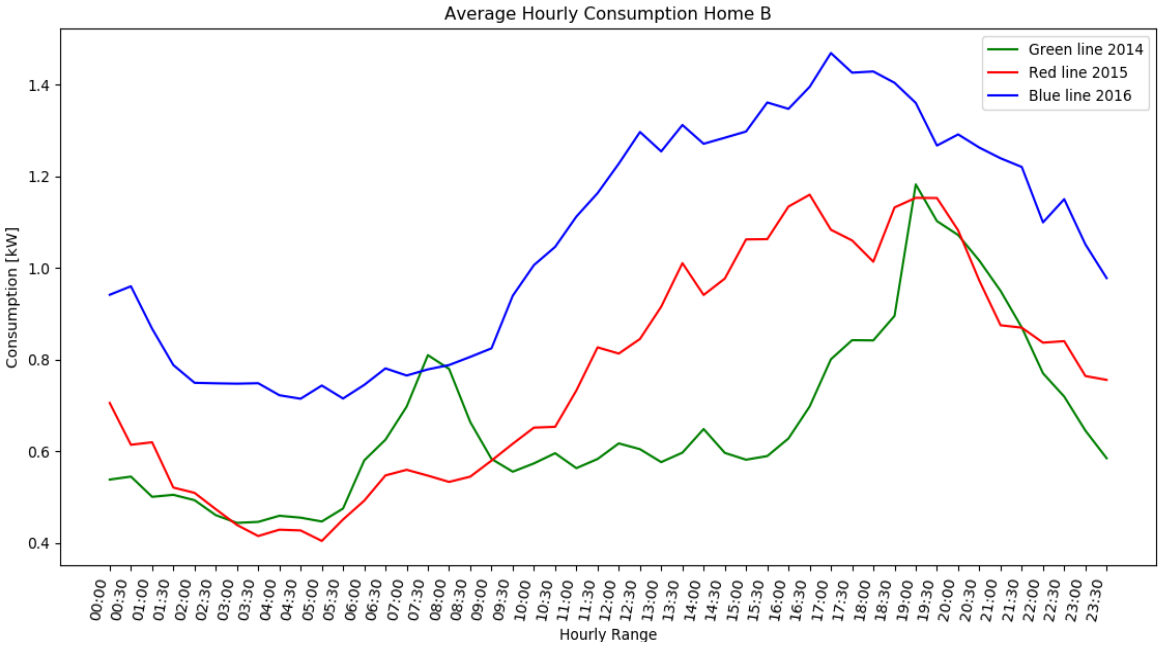


Figure 4: Averaged consumption throughout each year on half-hourly timestamps – Home B

Undoubtedly, there is not a very similar habit amongst the three years. During the second year, it is obvious that there is a significant decrease in the morning hours, which may be due to several reasons. Moreover, in-between from early morning to late afternoon there is also a constant increase in consumption, which indicates that there is probably a new resident or one of the existed changed daily routines and shifted the consumption load. Besides that, the total consumption itself is higher in year 2015 and even higher in 2016 while it is clear that apart from the first year the rest have a very similar pattern. That information even if do not affect directly the predictions, can however explain different phenomena and deviations from the goal. Nevertheless, this is probably in the lowest level of information that the smart home should inform the citizen.

In order to justify the Figure 3 even further, the Figure 4 explains the consumption during the year as it might be a case that the average hourly consumption can be affected by extreme situations in weather or inside the home. Below, Figure 4 gives a clearer picture of consumption, averaged per month that ensures that possible extreme phenomena are captured. If strange phenomena (e.g. heat wave during a summer month individually) occurred, which could possibly affect the consumption in an unpredicted way a new strategy might be adopted. Consumptions of all years carry similar paths of different magnitude and especially show an expected rise on summer months. That could

possibly imply that the sensors around appliances that control the home's microclimate affect the total consumption more than the rest and thus are crucial in recording.

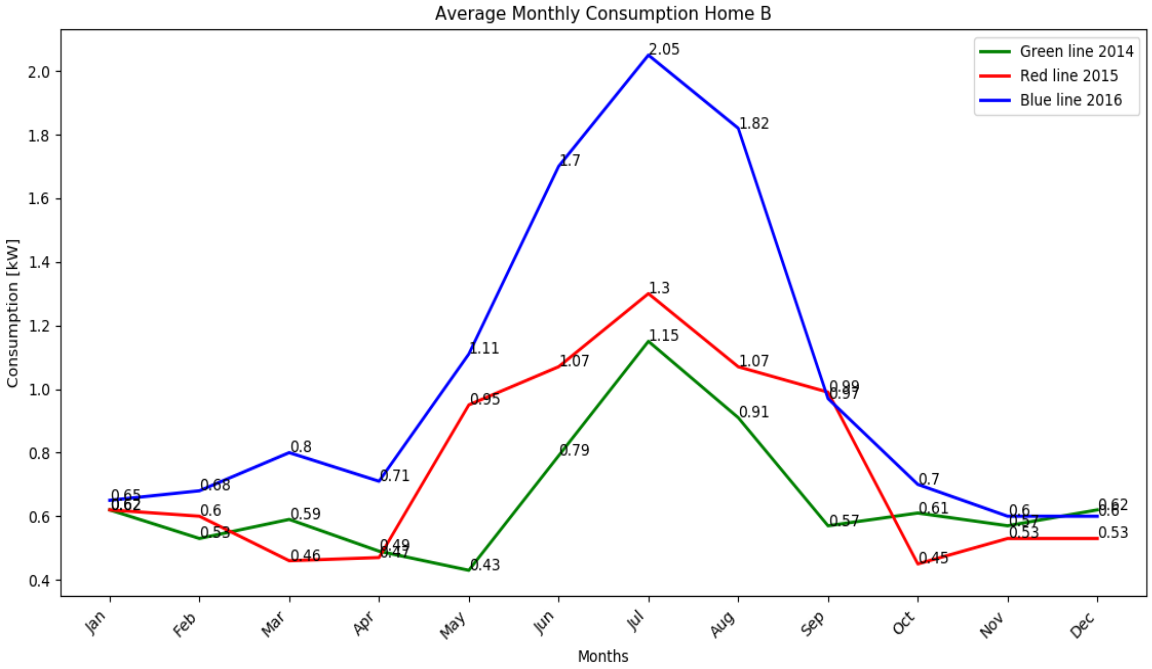


Figure 5: Averaged consumption throughout each year for every month – Home B

On this figure, the most obvious conclusion is that a much higher averaged consumption was observed on summer and that the first two years show a greater similarity in terms of pattern.

6.2.2 Prediction

Through regression on the 'raw' data of Home B's- first meter, the following results are achieved (Table 5). The first algorithm to be examined is Random Forest, which is considered the most dominant across the Decision Trees family. In addition, the most popular in the area of Machine Learning, 'Linear Regression' as long as two boosting algorithms, which by experience outperform the rest in many projects that are included. An important issue to be clarified is the fact that intentionally it was decided for classic algorithms in prediction such as ANN's and Naïve Bayes to be excluded from this thesis, as there are numerous implementations around the bibliography. Another important issue is that the following experiments do not contain the factor of '*Time'* as it was removed during the pre-processing step. So initially, Table 4 below, which only concerns data from 2014, shows that while no effort was spared the results were already impres-

sively high. Of course, RMSE as already explained is a negatively oriented score varying from zero to infinity and the lower the value the better.

Table 4: Initial regression results for 2014 data - Home B

B1	Initial Experiments			
	RandForest	Linear Regression	Gradient Boosting	Xgboost
RMSE	0,128	0,187	0,137	0,138
R2	0,964	0,922	0,958	0,957
Ad R2	0,963	0,921	0,957	0,957

Boosting algorithms perform almost identical while even the metrics R-squared and adjusted R-squared do not diverge at all. Clearly, Random Forest has a significant higher score, but the aim is the highest possible to be achieved. The relative big number of sensors encourages further development as it is thought that certain techniques can lead on predicting *Usage* with nearly hundred percent confidence. At this point, the validation of the model is using as a testing set a random one third of the initial. Generally, besides the problem looks like a time series problem and thus a different validation method is proposed, it is not and so; is treated as static. For time series problems, the split around testing and training sets should not be random in terms of sequence and the testing set should not include instances prior to those in training set.

In addition, by aggregating the data of two consecutive years and by applying a 5-fold cross-validation to ensure validity, a slightly better and more reliable performance is achieved. Table 5 below illustrates the results from processing over 35.000 half-hourly stamps of electricity consumption.

Table 5: Regression results on processing 2014's and 2015's data – Home B

B1	2 Years 5-FOLD CV			
	RandForest	Linear Regression	Gradient Boosting	Xgboost
RMSE	0.114	0.15	0.125	0.126
R2	0.975	0.952	0.97	0.97
Ad R2	0.976	0.953	0.971	0.971

Not surprisingly, it is clear that processing more data leads to better performance. Furthermore, the single split of the data into two predefined sets, the training and the testing, raises suspicion that results may not reach a specific level of reliability. Thus the 5-fold cross validation technique actually shuffles the dataset randomly while split-

ting it into five groups equal in size. At that point, each group is selected to be the testing set of the model that will evaluate the performance of it based on the rest four groups of data. Averaging the score of all five fits gives a less biased and less optimistic estimation of the model. Similarly, the same experiments were executed by feeding all the available data for Home B's first meter. It is very important to note that the number of instances at this stage is much larger as there are available data per minute; however, everything that is available is used. After the results of the second meter, a new analysis of the first meter will be conducted as some redundant data will have been excluded. For now, the results are again encouraging and start to get closer to the desired level.

Table 6: Regression results on processing all available data – Home B

B1	3 Years 5-FOLD CV			
	RandForest	Linear Regression	Gradient Boosting	Xgboost
RMSE	0.077	0.206	0.117	0.117
R2	0.993	0.956	0.986	0.985
Ad R2	0.994	0.956	0.986	0.986

In predictive analytics, tuning the hyper-parameters of each algorithm is considered as an essential part of the process. Hyper-parameter tuning mostly leads to implementations that are more accurate. Generally, the default values perform well; however, there is always room for improvement. By reaching the documentation of each algorithm, it is easily observed which the default values of hyper-parameters are, and which can be set to a specific bound. The next step of prediction analysis is to perform some brief test on arguments for the selected algorithms in order to increase the performance. The examined algorithm in this task is Random Forest as it turned out to be the most dominant algorithm. It was decided after some identification efforts that the only parameter to be examined and analysed would be '*n_estimators*' that expresses the number of trees on the so-called forest. That happens due to the limited space for improvement, and so parameters such as max_depth and min_samples_split are performing the max on their default values. Figure 5 gives a clear indication on how the raise of trees affects the performance.

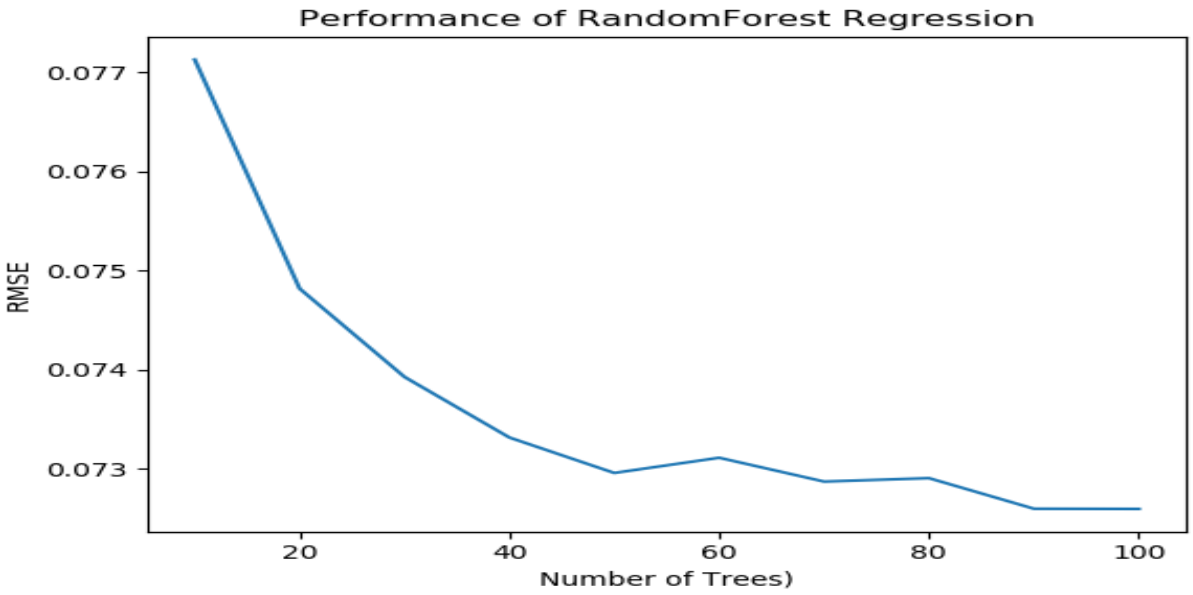


Figure 6: RMSE against the addition of extra trees

For the default number of ten estimators, the RMSE is around 0.077 exactly as shown at the Table 7. Adding more trees in general aids to achieve better results however; the improvement is so small at a point that computation cost is much higher than the achieved benefit. A very small positive change also occurs in R-squared however it is practically insignificant and is not as clear as on RMSE, given that may occur due to the randomness of sampling. In addition, Figure 6 illustrates the linear correlation between execution time and number of trees.

In Figure 6, numbers in y-axis are not indicative and do not capture seconds but are intended to show the percentage increase. In real numbers, executing the regression on a 5-fold cross validation with 110 trees in the available machine requires almost twenty-five minutes while using the default value of ten requires only two. The percentage of improvement is barely 0.0007%.

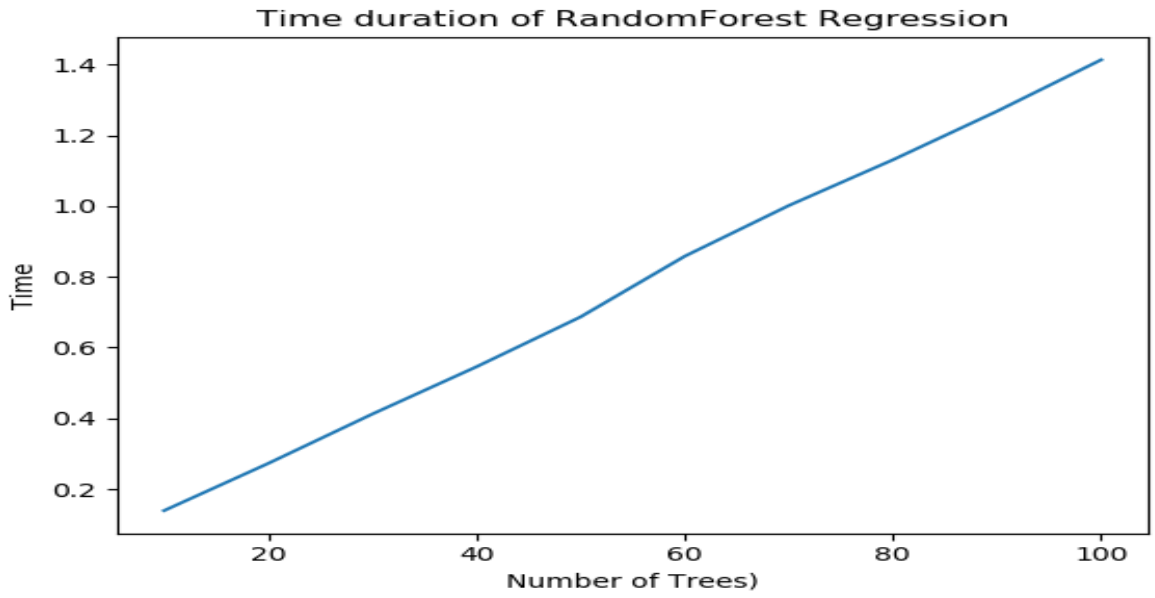


Figure 7: Execution time against the addition of extra trees

As in every project, data usually do not obey the same rules and in order to be processed, a clear strategy is needed. In that case, the second meter of Home B produced more data than the expected. More specifically the available data for the last fifteen days of the year 2015 were recorded minute by minute however; there was no information for the *Usage* at that timestamps. Thus, it was decided to be excluded even if it was confirmed earlier that higher amount of data leads to more precise predictions.

Another decision was to keep comparing only two of the four algorithms, as it was obvious that Linear Regression could not compete the rest due to complexity and the XGBRegressor had almost identical results to GradientBoosting. Moreover, on the same purpose it was decided to stop examining the simple R-squared metric as it was considered a form of redundancy since on that level, there was any differentiation to adjusted R-squared and the number of features (sensors) is constant.

In addition to the strategy that was explained above, Meter 2 needed a much more intensive pre-process, as there were obstacles that disrupted the desired structure. To combine the *Usage* column from the first meter to the second for all the three available years was a time-consuming process. First, the part of duplicates should be settled for both meters, which was a relatively easy task to be done. Duplicates might not affect the regression task of a single meter individually but in order to conclude on a perfect match, obviously that was an obstacle. Next, as mentioned earlier on Meter 2 the avail-

able data for the last fifteen days of December came per minute instead of a thirty-minute gap. That resulted in filtering the whole of available data and keeping only those whose form was ##:00:00 or ##:30:00. Although that was initially considered as the final step, the dimensions still were not matching, resulting in an unavoidable manual search through the data. The intent behind all these steps was to automatically perform such data cleansing so that it is repeatable in possible future datasets. Nevertheless, even if it was more or less expected that some timestamps were initially missing, there was no efficient automated way to deal with. Of course, this is not the recommended way to deal with such problems but sometimes it has to be done in the 'dirty' way. Finally, after an exhaustive search, the missing timestamps were located, and it was decided to be filled in, instead of being excluded from the corresponding time on the other meter. Through this manual intervention, therefore, the missing instances were added simply by mirroring adjacent instances.

For example, if the missing data were about 17:00:00 and 17:30:00, the first value was recorded identical to 16:30:00 while the second to 18:00:00. However, that approach was not expected to capture any differentiation on the results in contrast to others, since the number of those missing values was approximately 1 over 7000.

Once the second Meter was 'synchronized' to the target column, the experiments below were conducted. Initially, the results of the first year were unexpectedly low, both in terms of RMSE and in terms of adjusted R-squared. Both models showed an unexpected decrease, which however can be attributed to the fact that there was no sensor that had a high correlation with total consumption. The following tables contain the exact results of both models just as for the Meter 1 earlier. It is worth mentioning that 5-fold cross validation was again used for each experiment and that the experiments were stacking annual data, as well.

Table 7: Regression results - Second meter - Home B

B_2	1st year – 5-fold CV	
	Random Forest	Gradient Boosting
RMSE	0.387	0.408
Ad R2	0.679	0.641

Table 8: Regression results – Second Meter – Home B

B_2	2 years – 5-fold CV	
	Random Forest	Gradient Boosting
RMSE	0.377	0.42
Ad R2	0.726	0.665

Table 9: Regression results – Second Meter – Home B

B_2	3 years – 5-fold CV	
	Random Forest	Gradient Boosting
RMSE	0.408	0.473
Ad R2	0.754	0.674

At this point, the same experiments are conducted again; however, the data that are being processed are only those that can guarantee a total match between the two meters. It is also important to clarify that there is no hyper-parameter tuning at this level as the purpose of these experiments is just to highlight the importance in predicting, of reasonable sensors.

Therefore, up to this point the minute-by-minute data as explained earlier were excluded, while also the last *sixteen* days of the last year were not taken into consideration. In addition, first meter does not contain the desired data of four sensors for forty-two days; however, no matter excluding the corresponding instances, the results remain similar.

The following tables contain those results:

Table 10: Results of matched data – First meter – Home B

B_1	Matched data – 10-fold CV	
	Random Forest	Gradient Boosting
RMSE	0.1220	0.1419
Ad R2	0.9798	0.9728

Table 11: Results of matched data – Second meter – Home B

B2	Matched data – 10-fold CV	
	Random Forest	Gradient Boosting
RMSE	0.4413	0.5071
Ad R2	0.7208	0.6278

The Tables 10 and 11 above include matched data, as they will be used in the assessment below in order to examine the final step of sensor selection.

6.2.3 Sensor selection

The selection of the core sensors in order to examine the percentage of the accuracy that they can bring as a subset to this problem is made through a correlation matrix. For that reason, in Figure 7, all the *distinct* sensors of Home B are listed as well as the correlation between each of them.

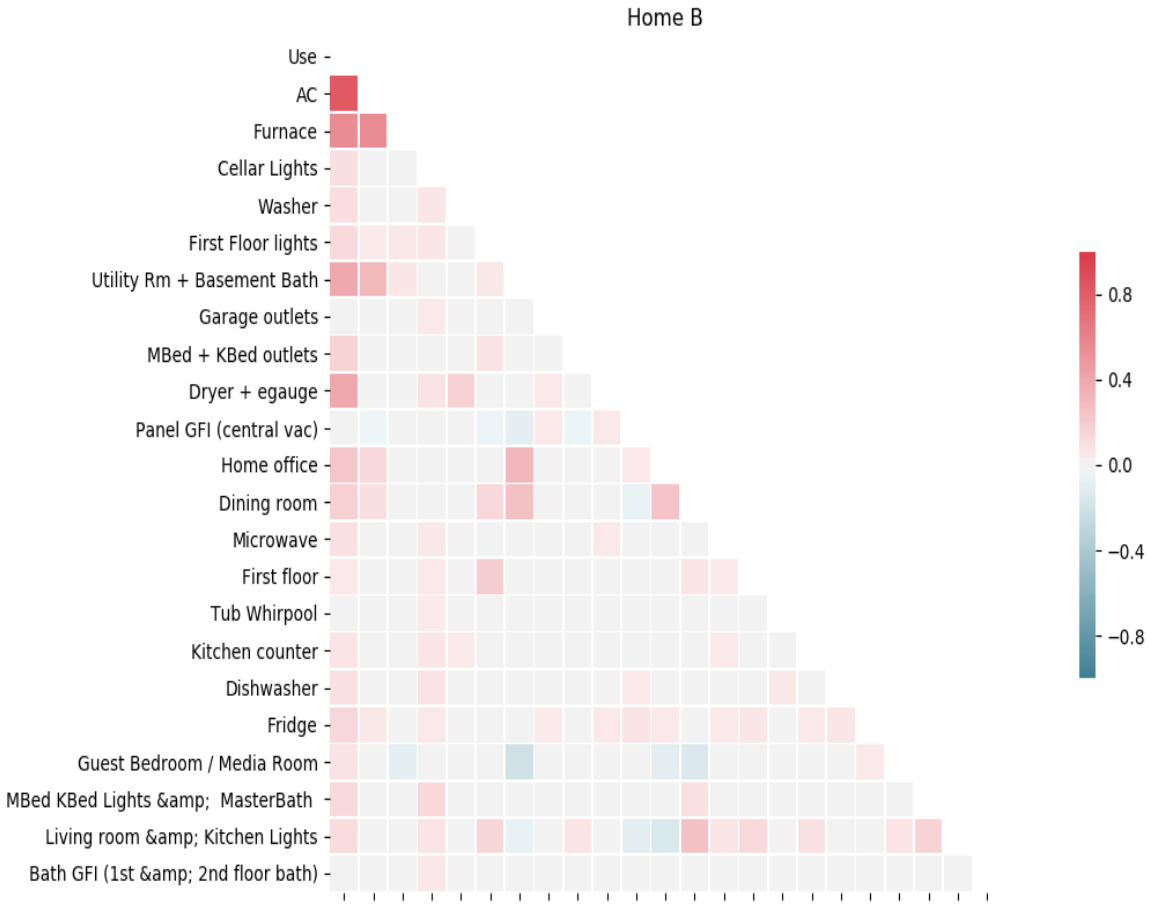


Figure 8: Heat map of Home B on overall set of distinct sensors

More precisely, the exact correlation of the top-6 values around the total consumption is as follows:

Table 12: Top correlated sensors for Home B

	Correlation with total consumption
A/C	0.833791
Furnace	0.552492
Utility Rm + Basement Bath	0.401344
Dryer + Egauge	0.393643
Home Office	0.234851
Dining Room	0.187463

The air condition sensor essentially depicts a huge part of the total consumption, fact that can be easily derived from Figure 4, which shows a rapid increase in the summer months. Subsequently, the contribution to the problem of the top-6 sensors is examined in two steps. In the first, only the top-3 are considered, while in the second step, the total of them.

Table 13: Top-3 sensors – Home B

B	**3 Top – 10-fold CV**	
	Random Forest	**Gradient Boosting**
RMSE	0.3516	0.3763
Ad R2	0.8225	0.7962

The top-3 sensors, indeed, do manage to capture some significant results ,Table 13, but the error is not good enough to rely on. The concept is to ensure that total consumption can be reached on a side way if the central meter fail. Thus, it is necessary to stay as closer to zero as possible. For Home B it is clear that besides the A/C has a very strong correlation with the total consumption, the top-3 sensor concept should be examined even further. On Table 14, the top-6 trial could be said to achieve a significant decrease on error.

Table 14: Top-6 sensors – Home B

B	6 Top – 10-fold CV	
	Random Forest	Gradient Boosting
RMSE	0.1483	0.1756
Ad R2	0.9690	0.9556

Summing up, Home B completely confirms the initial estimate that although the full set of sensors gave a fault of 0.1030, a subset of only six of them gave an error just 0.0453 greater.

6.3 Home C

Home C is almost double the size of Home B, around 3500 square feet again across two stories. Unfortunately, the real number of occupants is unknown. Home C also generates power which not only covers some of the electricity demands but is also possible to 'reverse direction when the home's generation exceeds its consumption'.

6.3.1 Consumption patterns

Fortunately, the three Homes under examination differ a lot in consuming behaviour and therefore it is expected to observe the reason that prediction might deviate amongst them.

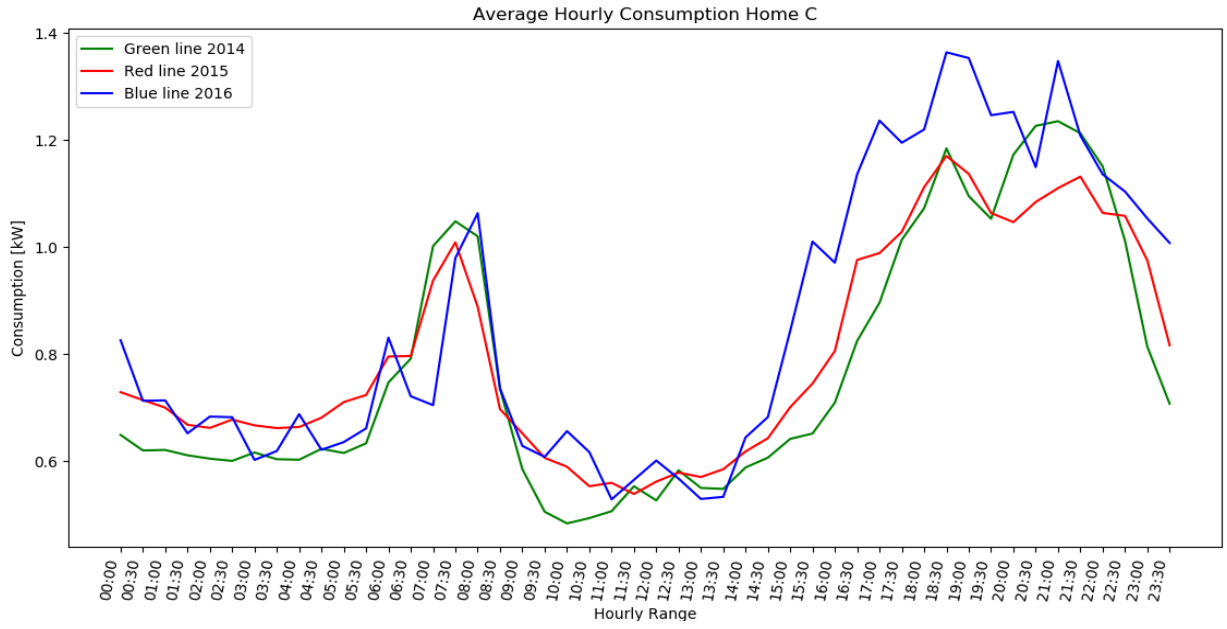


Figure 9: Averaged consumption throughout each year on half-hourly timestamps – Home C

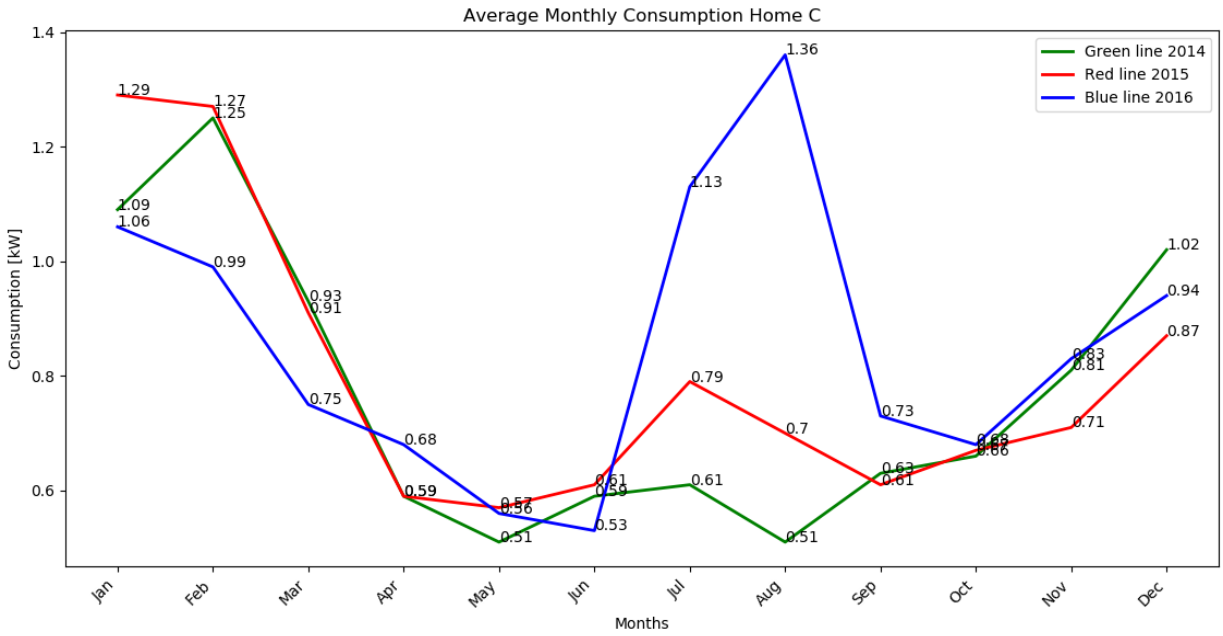


Figure 10: Averaged consumption throughout each year for every month – Home C

The above figures clearly indicate that Home C is very different regarding the consuming behaviour. More specifically, there is a surprisingly similar behaviour in consumption during the day. In contrast to Home B, where only summer months led to peak consumption, Home C in the winter period is experiencing an equally high consumption. The last year's data indicate a very dissimilar behaviour which was actually a case also for Home B. Unlike Home B, the other two Homes are not divided into two sets (meters) of different sensors. In fact, the second meter is a subset of the first and so it makes no sense to compare the two of them. Nevertheless, the same procedure will be followed with respect to the correlation matrix and the examination of the top-3 and top-6 sensors.

6.3.2 Sensor Selection

Home C contained a huge gap between 24th of December 2014 and 19th of March 2015 on several sensors in terms of missing values. Thus, inevitably all these half-hourly instances were ignored and since Figure 9 shows an increase on winter months it is very interesting to examine how the regression algorithms perform.

It is important to be clarified that Home C's sensors are regarding also external sensors such as Well; Barn and Wine cellar and generally the installed sensors are majorly

different. In addition, Home C does not contain neither an A/C sensor nor a Dining room, which was amongst the top-6 sensors of Home B.

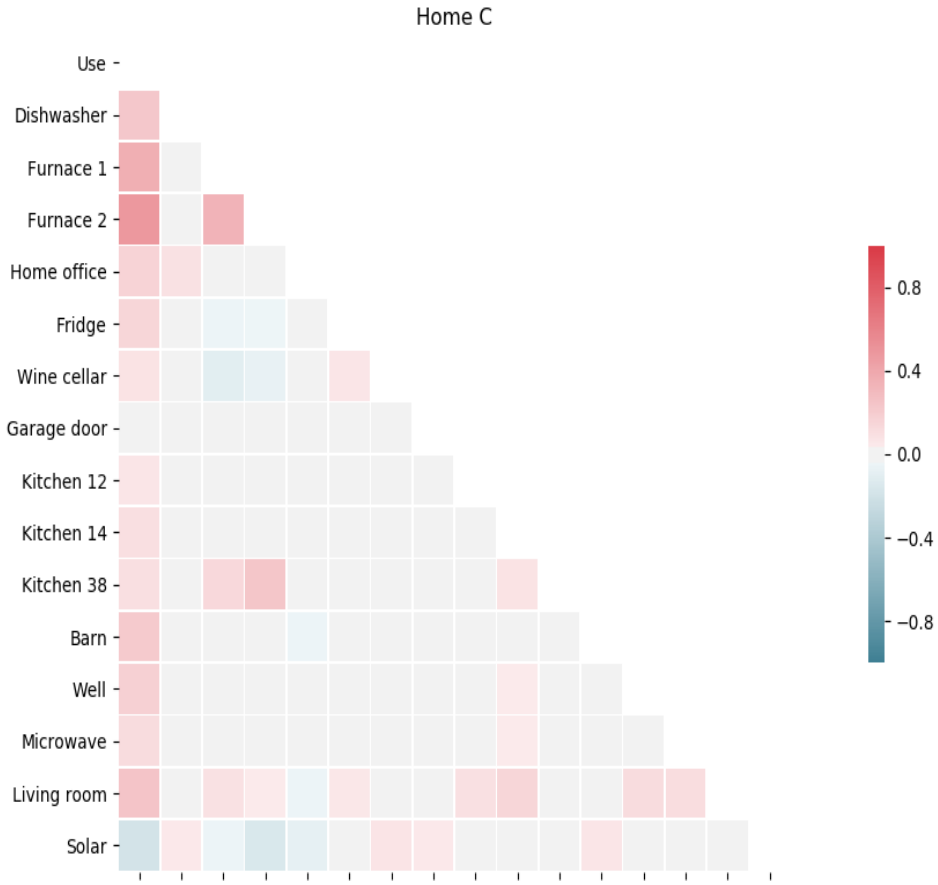


Figure 11: Heat map of Home C on overall set of distinct sensors

Similar to sub-chapter 7.2.3 the top sensors are listed below in Table 15.

Table 15: Top correlated sensors for Home C

	Correlation with total consumption
Furnace 2	0.487334
Furnace 1	0.366014
Living room	0.245231
Dishwasher	0.228602
Barn	0.212722
Well	0.180083

Both Furnaces inside the house as well as two outdoor sensors are in the top-6. This justifies fully the heating habits of the residents who tend to consume severe amount of electricity during winter. Table 16 shows the performance of the two algorithms based on the top-3 sensors of Home C.

Table 16: Top-3 sensors – Home C

C	3 Top – 10-fold CV	
	Random Forest	Gradient Boosting
RMSE	0.5189	0.5049
Ad R2	0.5901	0.6129

Because Home C is almost double in size in comparison with Home B, it is observed that it is impossible to get a satisfying enough predictability. This was expected based on the heat-map in Figure 10 as it clearly shows that unlike Home B there are no sensors so intimate with total consumption. It is worth noting that after the pre-processing, the average total consumption of Home C is less than B despite the difference in size. Specifically, the average total consumption of Home C is 0.777 while Home's B is 0.826. Since the top-3 concept failed completely it is not expected neither from top-6 concept in Table 17 to approximate the results that the whole of sensors returns.

Table 17: Top-6 sensors – Home C

C	6 Top – 10-fold CV	
	Random Forest	Gradient Boosting
RMSE	0.4068	0.4173
Ad R2	0.7527	0.7358

The results of all the available sensors are; round mean squared error- 0.318 and adjusted R-squared -0.8466. Therefore, as shown in Table 17, the results are within the acceptable range; however, it is not safe to draw conclusions based on those sensor sets. Another interesting fact is that Gradient boosting algorithm seems to overcome for the first time the Random forest regarding both the concept of top-3 and the summation of the sensors.

6.4 Home F

Unfortunately, Home F does not come with a description as it is included on the dataset as an update. Information about Home F is expected to be published later in 2018. Like Home C, there is only one general set of sensors and therefore no comparison can be conducted.

6.4.1 Consumption patterns

The first interesting thing that can be seen in Figure 11 is that Home F in general shows a much greater consumption. Homes B and C at peak hours had an averaged consumption around 1300 – 1400 Watts. Home F late in the afternoon, when it is considered peak network hours, consumes an average of about 3000 Watts. Besides that, there is no significant change in consuming behaviour over the years apart from the fact that consumption in 2016 is again higher (even slightly) than the rest.

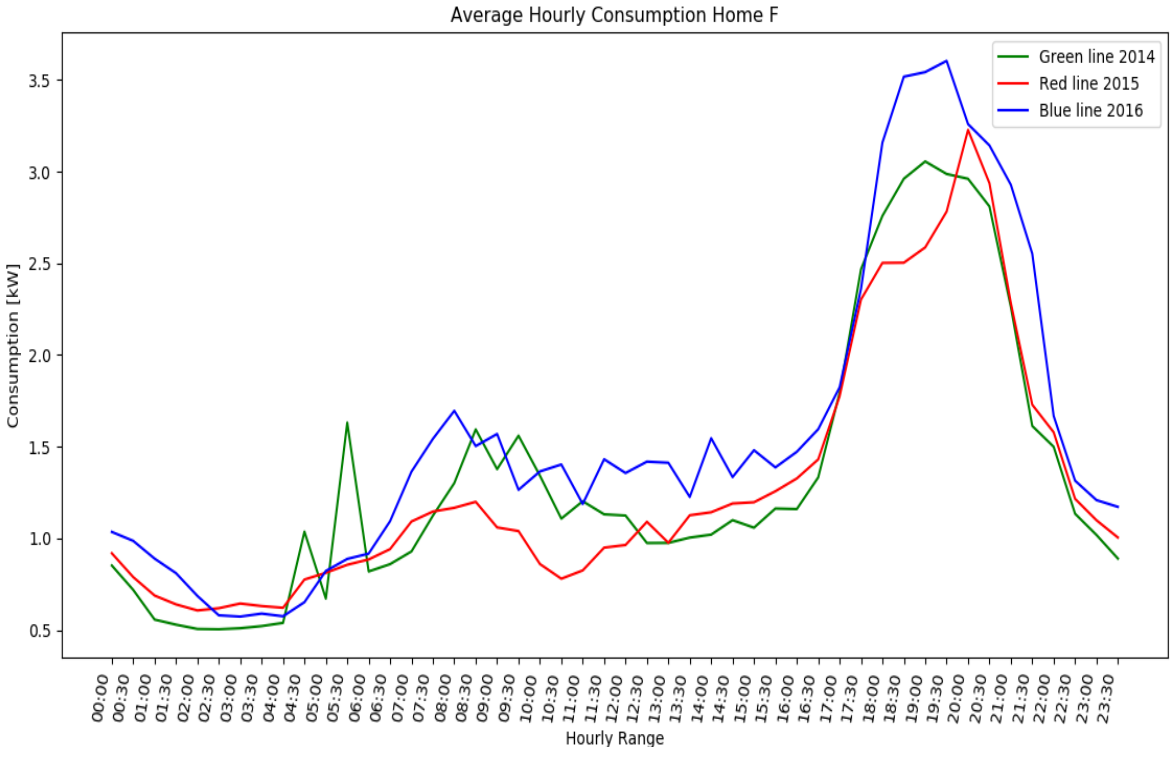


Figure 12: Averaged consumption throughout each year on half-hourly timestamps – Home F

Another phenomenon is that like Home C's Figure 8 the last year shows several spikes and is not as smooth as the rest years. That might indicate unstable weather phenomena or partly change on the number of occupants.

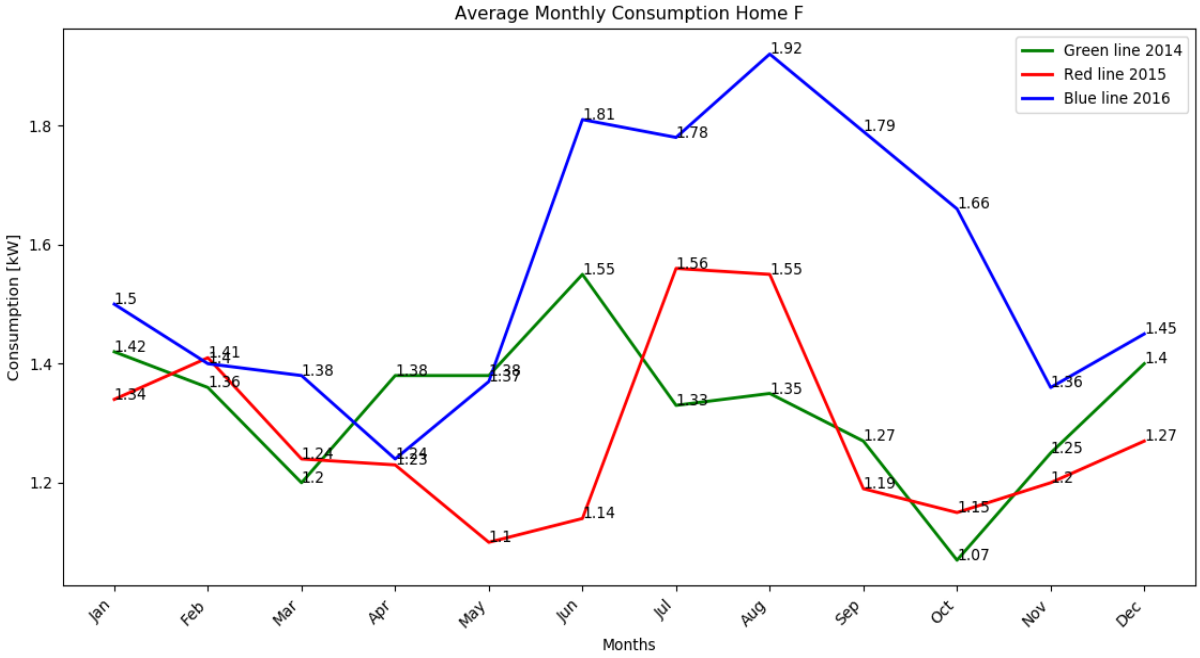


Figure 13: Averaged consumption throughout each year for every month – Home F

It is quite discouraging the fact that Home F has a very unstable consumption over months, however some important patterns can still be observed. First, relatively low consumption on winter months while not significant differentiation throughout the rest of the first two years. In last year, 2016, total consumption again explodes on summer months, while on the rest months has not remarkable difference.

6.4.2 Sensor selection

In Home F, fortunately, there are not special demands regarding the pre-processing as the formats totally match and the dataset is not suffering from missing values such as Home C.

The dataset of Home F contains sensors in three different wash machines; however, it was decided to be aggregated as one because it does not make sense to be taken into consideration as different. Furthermore, the second meter contained two sensors about the consumption of garage, but their real meaning was not clear and since the data description is missing it was decided for both to be excluded. Figure 13 shows only one sensor with negative correlation, the solar, while interestingly enough there is a relatively high correlation of half-bath foyer. Half-baths contain only a toilet and washbasin.

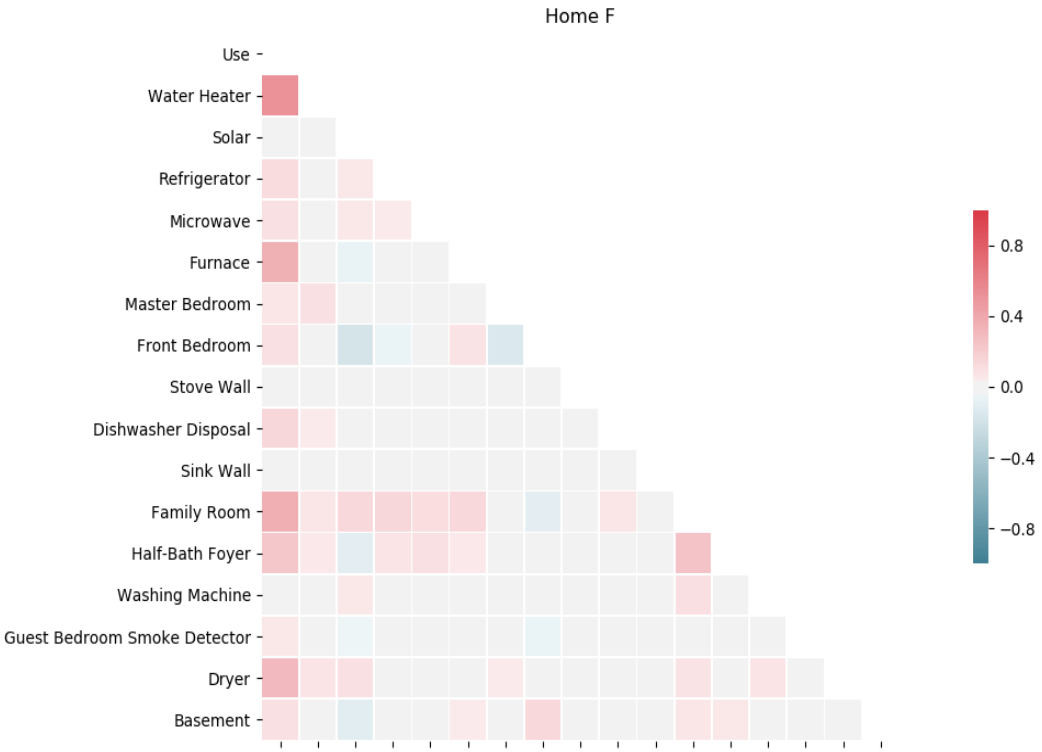


Figure 14: Heat map of Home F on overall set of distinct sensors

Table 18: Top correlated sensors for Home F

	Correlation with total consumption
Water Heater	0.518500
Family Room	0.365200
Furnace	0.356082
Dryer	0.298531
Half-bath Foyer	0.232953
Dishwasher Disposal	0.143392

In this case, the furnace is again in the top-3 sensors, while surprisingly the dish-washer disposal is amongst the top correlated sensors. Family room, which is similar to the living room concept, shows a relative high correlation while the two bedrooms (master and front) do not make it to the top-6. Below Table 19 ‘inspects’ the capability of the two algorithms to predict the total consumption.

Table 19: Top-3 sensors – Home F

F	3 Top – 10-fold CV	
	Random Forest	Gradient Boosting
RMSE	1.0829	1.0506
Ad R2	0.5411	0.5660

Undoubtedly, none of the algorithms has any reasonable predictability with very poor results, which could be compared only with a baseline model. However, again Gradient boosting algorithm achieves better results against Random forest when the attributes are just three. Besides the fact that the average consumption is 1.3939 and thus much higher than the rest two homes the RMSE is unacceptable.

Table 20: Top-6 sensors – Home F

F	6 Top – 10-fold CV	
	Random Forest	Gradient Boosting
RMSE	0.8831	0.9618
Ad R2	0.6955	0.6362

The top-6 sensors of course return better results, but they are also not enough to say that they form a trustworthy subset to reach the total consumption. This can be attributed to the low correlation as a summation of the top-6 sensors and the undetermined change in overall consumption as it was illustrated in Figure 12. In general, none of the two subsets is reliable; however, the results that the total number of the sensors returns are not so disappointing. In particular, the Random forest gives RMSE equal to 0.6023 and adjusted R-squared equal to 0.8584. On the contrary, Gradient boosting performs worse with RMSE equal to 0.6985 and adjusted R-squared equal to 0.8091.

Summarizing is safe to conclude that furnaces and living rooms are those that show high correlation in all cases with the total consumption. Of course, as expected the A/C was the major sensor for Home B, but unfortunately there was not any for the rest two homes. In terms of predictability based on the top sensors, it was clear that there was no validity for Home F, while for Home C the results were 'marginally' acceptable. Home B, on the other hand, gave optimistic results and this was not only attributed to the A/C sensor. Home B showed a smoother and more similar consumption patterns across the years.

7 Load Forecasting

In the need of forecasting the consumption or the upcoming price of [kW], it was decided for a simple model to be built in order to highlight some of the factors that affect consumption. Beside every installed sensor and the recorded consumption, the new model focuses on external factors and residents' habits. Prior consumption as it was presented on Figures 3, 8 and 11 shows that consumption rises after the noon and does not stabilize until late afternoon. That is a strong indication that residents have a morning job and probably children. So far, the constructed models were aiming to predict consumption based on smaller fraction of installed sensors.

For a real-time prediction, it is necessary to record data and analyse them in nearly zero time. However, citizens are not only interested in learning a prediction of their instant consumption to adopt their consuming behaviour but also for prior knowledge of the upcoming consumption. Providers who need to adjust their production plans and form a more competitive price policy show similar interest. By building a reliable model that can forecast the demands of electricity and the loads that grid could face, it could create a larger profit margin by reducing production costs and a more environmentally friendly profile.

7.1 Strategy

Forecasting electricity consumption is a particularly complex and difficult process. The factors that affect consumption are many. In [65] and [67] the forecasting is on a larger scale and thus follow a very different approach. In this case, the consumption forecasting process is done on a much smaller scale of individual residence for day ahead. Initially, what should be clear is that such a model includes only variables that are set to be known in advance.

The model includes mostly weather data. From the available weather data that match the previous Homes, only the following were selected:

- Temperature (°F)

- Apparent Temperature (°F)
- Wind Speed (Mph)
- Wind Direction (Bearing)
- Humidity (%)
- Dew Point (°F)
- Weather Icon (Categorical)

In addition, a simplistic dummy-like variable is created that indicates if a day fell on a weekend. The last variable shows similarly to weekends if a day fell on a national holiday as they were listed here [77].

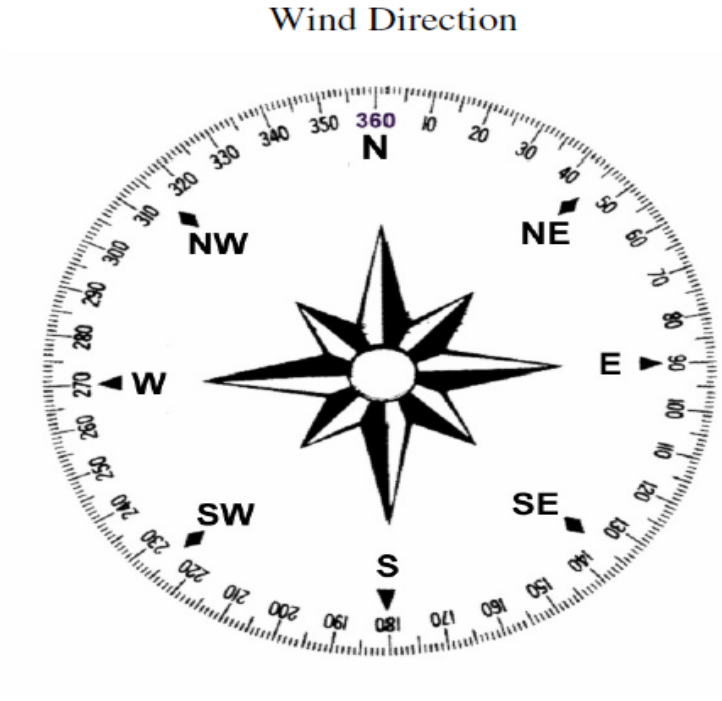
Based on the previous steps it was decided that the model should focus on two different time intervals, on and off-peak hours. Obviously, as in most residences, on Home B the peak consumption hours were between 15:00 and 21:00 so it was decided for the *Usage* to be averaged under that time-interval. The off-peak consumption hours was a tricky part as the intention was to be of equal time as the first and not overlapping each other. The rest three 6-hour intervals are expected to have similar behaviour however only one should be chosen.

Besides the most common approach for providers is to introduce night tariffs and rates, it was decided that as long as the two variables of this model were referring on Weekends and Holidays the wiser approach should be to choose the 09:00-15:00 interval. The on and off-peak hours is a concept that should be separately adjusted for each Home, based on the occupants' habits. If the purpose is the load forecasting of Grid, then of course this separation is unified. There are studies that choose different time intervals. For example, the peak period in [78] was defined as 7am to 7pm, Monday to Friday, while all the other times and public holidays were considered as off-peak. Consumer behaviour is estimated to be more accurate described in the mornings on holidays and weekends than with nightly habits that are pretty much the same. Therefore, the scale of our data is daily, and the same process is followed for the weather data. Despite the supplementary available dataset contains hourly information for the years 2014-2015-2016 the targets are turning into daily stamps. For example, the average temperature of 01/01/2014 equals to twenty, which is calculated through the half-hourly values between 15:00 and 21:00.

It is worth noting that dew point temperature [79] is "the temperature at which the air can no longer hold all of the water vapor which is mixed with it, and some of the water vapor must condense into liquid water". Thus, it is obvious that air temperature is always equal or higher than dew point.

7.2 Pre-processing

The effort on pre-processing for this dataset was in a very different perspective. There was only one missing value as it was by mistake transmitted through the averaging process described above. The missing value was filled by the average of the previous and next day. Unlike the regression part in Chapter 6, there was some feature engineering as the variable of *Wind Direction* was categorized based on their orientation to the points of horizon [80].



Common	Degree
N	350, 360, 010
N/NE	20, 30
NE	40, 50
E/NE	60, 70
E	80, 90, 100
E/SE	110, 120
SE	130, 140
S/SE	150, 160
S	170, 180, 190
S/SW	200, 210
SW	220, 230
W/SW	240, 250
W	260, 270, 280
W/NW	290, 300
NW	310, 320
N/NW	330, 340

Figure 15: A compass for transforming the Wind direction data from numerical to categorical

More specifically, if the wind direction was amongst 315-45 it was labelled as *North*, if it was between 45-135 it was labelled as *East* etc. Then all the four categories were turned into dummy variables.

Regarding, the weather icon the pre-processing contained two parts. Since many algorithms cannot handle categorical variables, the most common approach as explained

above is to split all those icons into separate dummy variables. Nevertheless, the categories were too many and indeed were not the same for each Home, so it was not possible to examine the exact same model (in terms of structure) to all the available houses. For example, some of contained icons were Rain, Snow, and Flurries etc. Since the most dominant icon for all the houses was *Clear*, it was decided to turn this variable into a dummy variable only for *Clear* icon.

The last variable that was processed in a similar way is the sunset time. As explained in Chapter 6, in [64] it was used a variable regarding the hours of daylight during each day. In this case, it was taken into consideration the hour of sunset on each day and was transformed into an integer. For example, if the sunset time was 19:23 this was turned into 1923 and then similar to *Wind direction* all the integers were grouped into five different categories per hour. Finally, the weather dataset also needs pre-processing on the time variable, which comes as a 10-digit Unix base.

7.3 Implementation

The algorithms that are tested on this task are SVM, Random Forest, Stochastic Gradient Descent and Logistic Regression. The structure of results will not be the same as in Chapter 7. On the following sub-chapters, there are three different technical stages of classification while there are three different classification implementations. The first one splits the total consumption instances around the general mean value. The second one splits the total consumption instances regarding each month's mean value, while the last one regarding each season's mean value. For each step, there are some indicative visualisations of the best algorithm, highlighting the behaviour of it, as its hyper-parameters are changing.

7.3.1 Home B

To begin with, below in Table 21 it is presented the performance of each algorithm in terms of accuracy.

Table 21: Results for total mean in Home B

B	15:00-21:00 / 2 classes over total mean			
	SVM	RandForest	SGD	LogReg
Stage 1	0.8869	0.8773	0.5986	0.8746
Stage 2	0.8705	0.8773	0.7752	0.8760
Stage 3	0.8910	0.8950	0.8801	0.8828

On this chapter, each algorithm will be examined for each of the following stages:

Stage 1 represents the results that are given by calling the default algorithms as they are set on Scikit-Learn package

Stage 2 reflects what happens when the training data are scaled. The data due to the different units and ranges it is wiser to be scaled. For some classifiers such as SGD and ANN this is a crucial step, while others are not affected. Scaling was not conducted in the Chapter 7, as data were of the same unit.

Stage 3 reflects a wide hyper-parameter tuning as it already mentioned that can boost the performance of every algorithm. For each algorithm, the tuned parameters are chosen empirical, so not all of them are set to be tuned.

It is important to be clarified that for SVM the different kernels delay the process. This happens mostly because *linear* kernel is an almost identical implementation with SGD's *hinge* kernel. Moreover, the *polynomial* kernel requires the data to be scaled and so cannot be examined at the same time with the rest.

Table 22: Selected hyper-parameters for total mean in Home B

Best Parameters -- Peak demand			
SVM		**Random Forest**	
Kernel	RBF	Estimators	150
Gamma	0.01	Min Split/Leaf	2/4
C	0.5	Max Features	Auto
SGD		**Logistic Regression**	
Alpha	0.001	Max Iterations	2
Iterations	1000	C	0.1
Loss Func	hinge	Penalty	l1

Transforming the consumption values into classes, based on the total average consumption might not be of great interest. Therefore, Table 23 and Table 24 show the results of classifying over a set of mean values, based on the Month.

Table 23: Results for monthly mean in Home B

B	15:00-21:00 / 2 classes over monthly mean			
	SVM	RandForest	SGD	LogReg
Stage 1	0.6825	0.6430	0.6021	0.6798
Stage 2	0.7043	0.6430	0.6267	0.6771
Stage 3	0.7166	0.6934	0.7029	0.7029

Clearly, the results are much worse than expected. In reality, that differentiation in transformation affects 278 out of the 1096 target values. However, on a technical perspective it is also the first time on this thesis that Random Forest performs worse than all the rest classifiers.

Table 24: Selected hyper-parameters for monthly mean in Home B

Best Parameters -- Peak demand			
SVM		Random Forest	
Kernel	RBF	Estimators	50
Gamma	Auto	Min Split/Leaf	2/4
C	3	Max Features	Log2
SGD		Logistic Regression	
Alpha	0.001	Max Iterations	2
Iterations	2000	C	0.1
Loss Func	Hinge	Penalty	l1

On the other hand, that split suffers from the effect of having a month with totally abnormal consumption due to an unknown situation in one of the three years and thus affect the result. Probably a more conservative approach would be to split into seasonal means. In this regard, the citizen could be informed about his consumption based on what he had in recent years. Table 25 and Table 26, below, contain those results.

Table 25: Results for seasonal mean in Home B

B	15:00-21:00 / 2 classes over seasonal mean			
	SVM	RandForest	SGD	LogReg
Stage 1	0.7138	0.7043	0.6035	0.7356
Stage 2	0.7411	0.7043	0.6457	0.7288
Stage 3	0.7493	0.7561	0.7397	0.7411

Table 26: Selected hyper-parameters for season mean in Home B

Best Parameters -- Peak demand			
SVM		**Random Forest**	
Kernel	RBF	Estimators	150
Gamma	0.01	Min Split/Leaf	2/4
C	5	Max Features	Auto
SGD		**Logistic Regression**	
Alpha	0.001	Max Iterations	50
Iterations	2000	C	2
Loss Func	Hinge	Penalty	l1

The results for all algorithms are significantly higher, but still closer to the monthly mean. The classes as in any real-world problem are not balanced. In the first case, there is a split of 735-361 in favour of *Low*, while in second and third case the splits are 665-431 and 718-378.

Obviously, the split of consumptions varies for each house and the time interval under examination. In this case, there is no clear preference in any of the two classes. It seems that depending on the side of interest there is a different perspective on the problem. For providers, as already discussed, the storage of excess electricity is extremely costly and thus predicting both classes is of equal importance. For citizens, from our perspective, it is preferable that the error concerns the higher consumption forecasts, rather than the opposite. Anyhow this impression is not a rule and might not apply in all cases, resulting therefore on choosing the standard accuracy metric for evaluating the classification performance of each algorithm.

Below, Table 27 shows the size of the two classes for each of the cases under consideration.

Table 27: Number of instances regarding each class after transformation around different means

Homes-Time Interval	Consumption Per Total		Consumption Per Month		Consumption Per Season	
	Classes		Classes		Classes	
	High	Low	High	Low	High	Low
Home B OFF-peak	372	724	400	696	393	703
Home C ON-peak	395	685	382	698	380	700
Home C OFF-peak	408	672	390	690	398	682
Home F ON-peak	553	543	553	543	547	549
Home F OFF-peak	438	658	447	649	452	644

The classes are not that balanced for Home B and Home C, as opposed to Home F, however, this is not considered prohibitive to continue the process. Interestingly, in most of the cases the most dominant class is *Low*. Table 28 is the most important of this work as it summarizes the results of all the basic experiments. Then, the rest of this Chapter focuses on some different approaches of the problem.

Algorithms	Homes Time Inr	STAGE 1	STAGE 2	STAGE 3	STAGE 1	STAGE 2	STAGE 3	STAGE 1	STAGE 2	STAGE 3
		Consumption Per Total			**Consumption Per Month**			**Consumption Per Season**		
SVM	**HOME B OFF PEAK**	0.7833	0.7915	0.8024	0.6839	0.7002	0.7084	0.6975	0.7057	0.7125
RF		0.7724	0.7724	0.7973	0.6675	0.6675	0.7152	0.6811	0.6811	0.7179
SGD		0.6811	0.7152	0.7915	0.5572	0.598	0.6975	0.5054	0.6117	0.7057
LR		0.797	0.7847	0.797	0.6961	0.692	0.6989	0.6907	0.6893	0.6989
SVM	**HOME C ON PEAK**	0.7441	0.7339	0.7759	0.6652	0.6929	0.697	0.6694	0.6984	0.7233
RF		0.7261	0.7261	0.7676	0.65	0.65	0.7123	0.6556	0.6556	0.7109
SGD		0.5532	0.6846	0.7897	0.5311	0.5975	0.6915	0.6334	0.6639	0.7192
LR		0.7842	0.7773	0.7869	0.6929	0.6929	0.6984	0.7095	0.7081	0.715
SVM	**HOME C OFF PEAK**	0.7233	0.7634	0.7731	0.6307	0.6957	0.7136	0.6237	0.7178	0.7385
RF		0.7219	0.7219	0.7593	0.6666	0.6666	0.7136	0.668	0.668	0.7247
SGD		0.6915	0.6777	0.7717	0.6071	0.6071	0.715	0.6002	0.6559	0.7289
LR		0.7745	0.7662	0.7745	0.7067	0.7109	0.7136	0.7206	0.7192	0.7316
SVM	**HOME F ON PEAK**	0.564	0.6784	0.6839	0.5217	0.6512	0.6621	0.5299	0.6798	0.6852
RF		0.643	0.643	0.6866	0.5912	0.5912	0.6512	0.6212	0.6212	0.6716
SGD		0.5027	0.5855	0.6757	0.5068	0.5871	0.6607	0.4986	0.5994	0.6662
LR		0.6825	0.6716	0.6852	0.6662	0.6457	0.6716	0.673	0.6634	0.6811
SVM	**HOME F OFF PEAK**	0.643	0.6757	0.6811	0.6185	0.6294	0.6416	0.6008	0.6253	0.6471
RF		0.6267	0.6267	0.6893	0.6076	0.6076	0.628	0.5953	0.5953	0.6348
SGD		0.5313	0.6335	0.6771	0.5027	0.5871	0.6376	0.5231	0.5912	0.6362
LR		0.6716	0.6689	0.6784	0.6294	0.6294	0.6416	0.6294	0.6376	0.6485

Table 28: Accuracy results for each Home's period for all different means

The results between the houses differ and it is obvious that there is not algorithm that performs significantly better from the others. In fact, in this case, Random forest has a serious competition and it is not as dominant as in Chapter 7. Next, there is an indicative visualisation of hyper-parameters performance for the best of each Home-Interval Time combination on Figures 15-20.

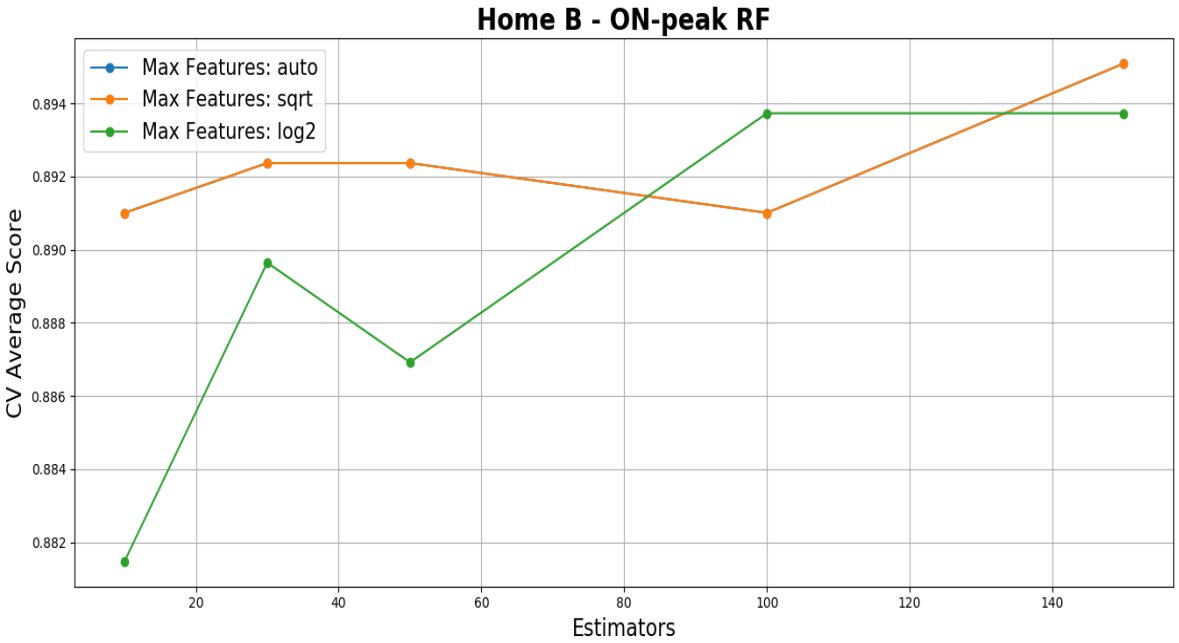


Figure 16: Home B, hyper-parameters of Random Forest for On-peak period

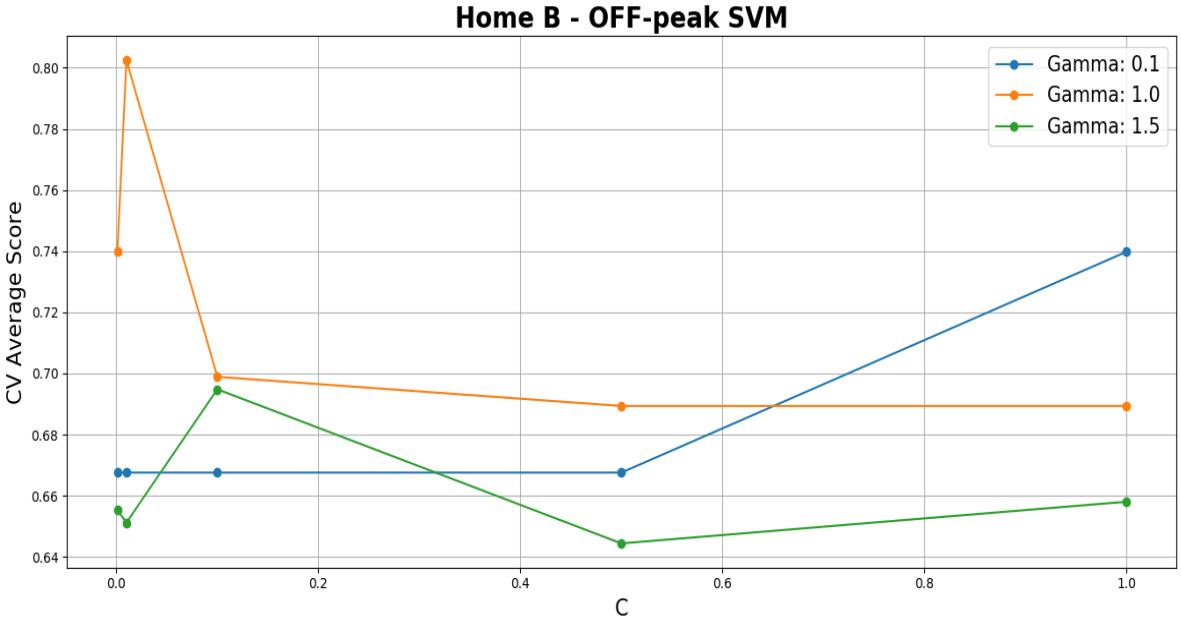


Figure 17: Home B, hyper-parameters of Support Vector Machines for Off-peak period

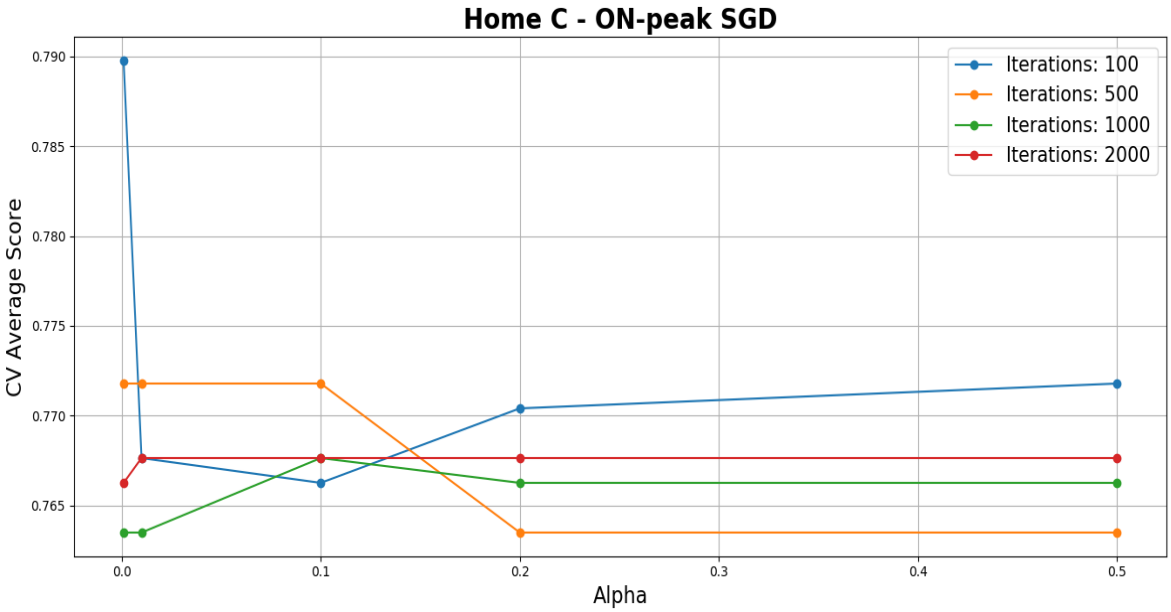


Figure 18: Home C, hyper-parameters of Stochastic Gradient Descent for On-peak period

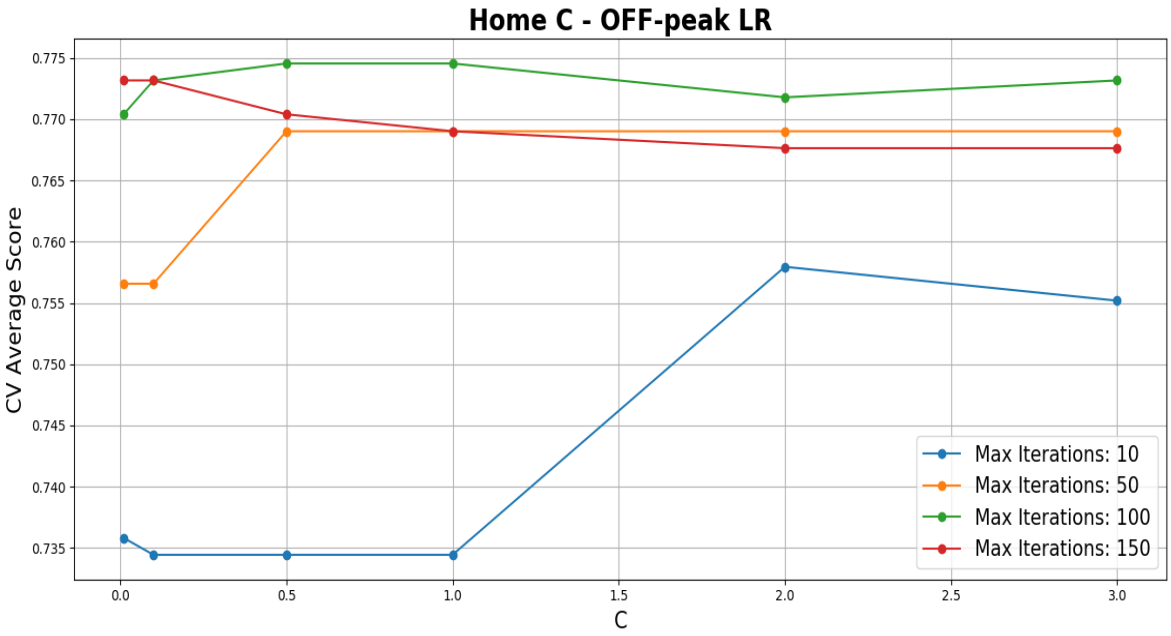


Figure 19: Home C, hyper-parameters of Logistic Regression for Off-peak period

For the sake of clarity, all the diagrams show the correlation of only two of the hyper-parameters that were examined. For the same reason, the test values over each hyper-parameter have deliberately decreased.

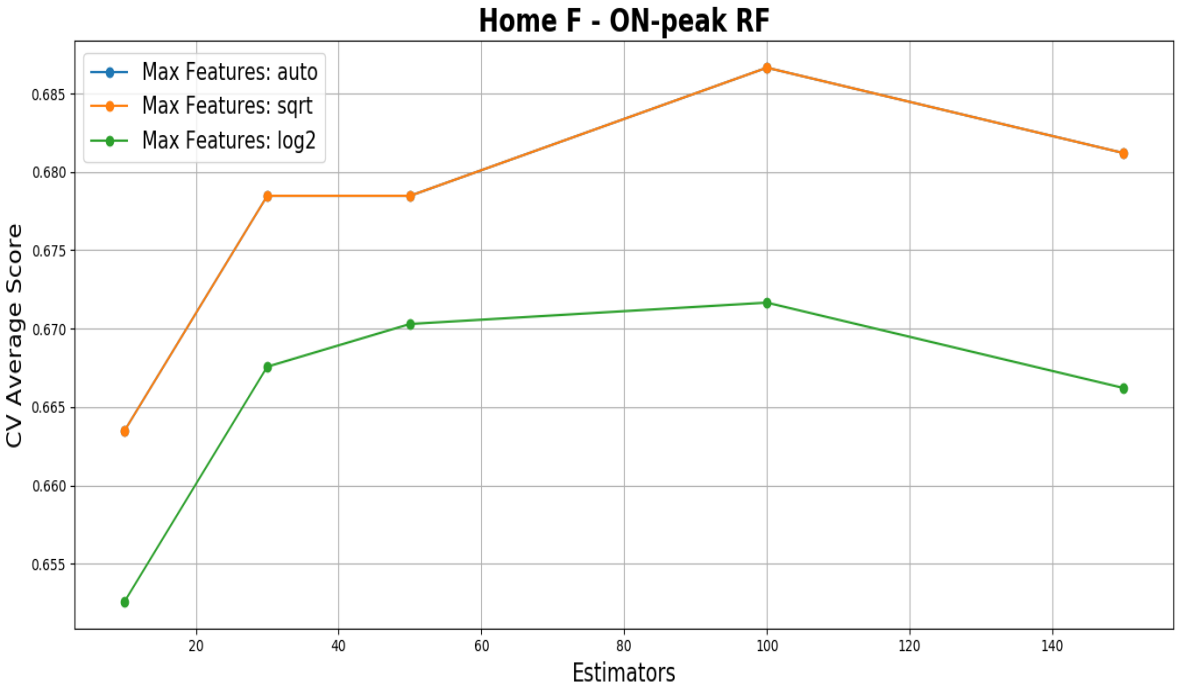


Figure 20: Home F, hyper-parameters of Random Forest for ON-peak period

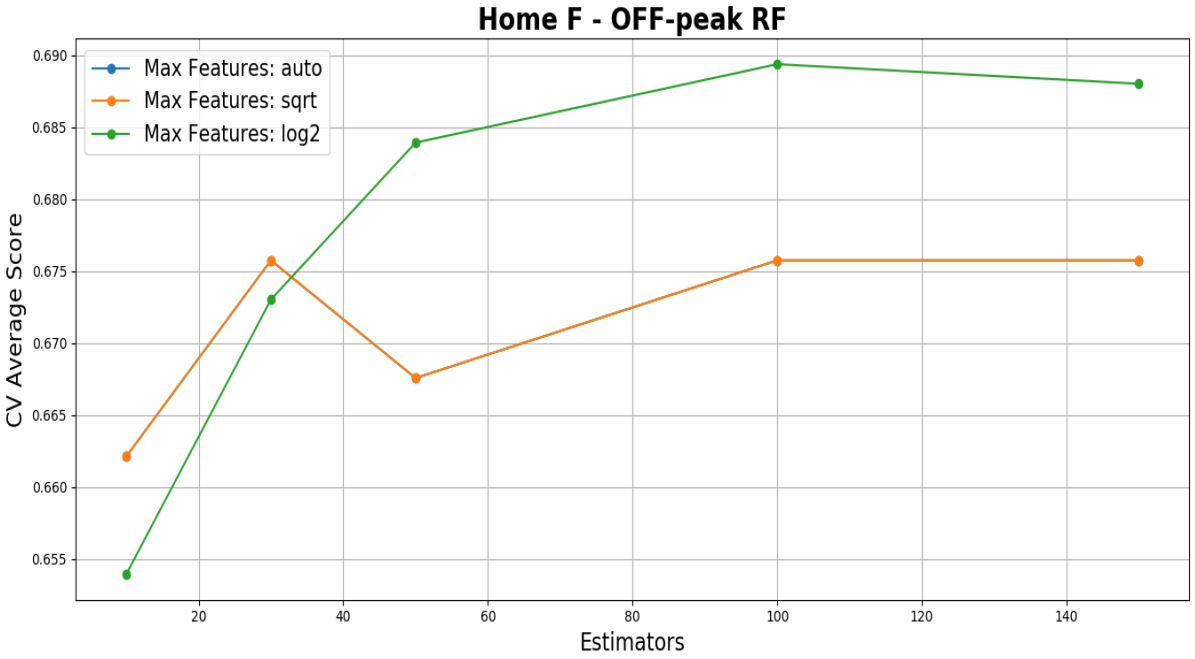


Figure 21: Home F, hyper-parameters of Random Forest for Off-peak period

It is worth mentioning that in Figures 15,19 and 20 where the random forest algorithm is examined, for the hyper-parameter 'Max_Features', the values *auto* and *sqrt* are performing equally and so the one overlaps the other, resulting in two only lines.

7.4 Grid load

As a mentioned earlier, the rest of this chapter focuses on different approaches of the same problem in order to investigate possible new aspects. At this point, there is a merging of the data that each home brings, with the ultimate purpose of making a simplistic representation of the grid's load. Since all the houses are located on the same region the weather data are very similar, thus an average value for each weather metric is calculated. Regarding the electrical consumptions, the values are summed both for Yesterday and Past week load. The target values again are transformed into two classes, however since previous analysis resulted in better performance for the general mean value, it was decided to be the only to be examined. The split of classes for On-peak period is 474 High-606 Low, while for Off-peak period it is 433 High-647 Low. It is worth nothing that the merging of data caused the resulted size of our matrix to be the same with Home C (1080 instances).

The structure of results remains the same and it is presented on the following Tables 29 and 30.

Table 29: Results for On-peak period of Grid

G	On-peak / General mean value			
	SVM	RandForest	SGD	LogReg
Stage 1	0.6846	0.6680	0.5892	0.7026
Stage 2	0.7358	0.6680	0.6154	0.7150
Stage 3	0.7358	0.7178	0.7247	0.7192

Table 30: Results for Off-peak period of Grid

G	Off-peak / General mean value			
	SVM	RandForest	SGD	LogReg
Stage 1	0.6376	0.6860	0.5767	0.7302
Stage 2	0.7219	0.6860	0.6528	0.7275
Stage 3	0.7275	0.7495	0.7247	0.7302

As seen in Tables 29 and 30, the performance of the model does not change drastically. However, at this point through merging, a different perspective of generalisation can be examined. The accuracy for both periods remain similar however, for Off-peak period a slightly higher accuracy is achieved.

7.5 Disaggregation

The last concept that has been examined on this work is the load forecasting for Home B based on an hourly level. This concept was applied only for Home B, because it was the one that achieved the most promising results and also showed an unexpected difference between On and Off-peak period. This level of disaggregation is the penultimate of the main concept in 7.3. The chosen input variables slightly differ since the period of forecasting is hourly. After the feature selection step, the variables regarding consumption patterns are the consumption of previous hour, the consumption of the same hour the previous day and the average consumption at the specific hour over the last week. For both the On and Off-peak periods the starting hours were deleted since there were not available at that stage. To regain prior values of the starting hours we stepped one more stage back and this was not expected to reasonable affect the model.

Additionally, the strategy for splitting data into training and testing set was different. More specifically, instead of applying cross-validation it was decided to split manually the training set after two years of data and use the last year for testing, since the size of dataset was bigger. Due to time limitations, it was not feasible to examine all the algorithms and thus it was decided to choose only Linear Regression. This particular algorithm was very accurate in comparison to the rest as regards its default implementation.

As the problem is significantly imbalanced (split regularly around total mean), the corresponding metric was not selected to be accuracy. In this case, both precision and recall are calculated and the main class of interest is *High*. The results are summarized below on Table 31.

Table 31: Results of Home B for the disaggregated form

Home B	On-peak		Off-peak	
	Precision	Recall	Precision	Recall
High	0.8	0.66	0.85	0.45
Low	0.84	0.92	0.71	0.94

Clearly the precision for *High* class is in both cases satisfying, however recall is much lower. Usually there is a trade-off between precision and recall. It is worth noting that for these two cases the accuracy equals to 0.83 on On-peak and 0.73 on Off-peak period.

8 Discussion

Electrical load forecasting is a complex problem, which needs detailed design and a deep understanding of the domain. The purpose of this work was not to build a state-of-the-art model, but to analyse as many different aspects as possible. The bibliography generally suggests several factors that could positively affect the electrical load predictability; however, due to the limited available data, the effect of weather conditions on the electrical load forecasting was examined in this study.

Load forecasting based on weather data is a topic that already has gained attention among researchers. The overall aim of this study is to examine this general problem within the concept of smart city. Unlike many studies, which attempt to predict electricity consumption of the grid or large blocks of apartments, this work is focused on single households. In contrast with commercial buildings, where electricity consumptions follow a pattern (for example 08:00 to 17:00 with a decline around 13:00 during the lunch break), occupant behaviour has a major effect on single households. Thus, in single household scale, a thoroughly prediction model is hard to be established.

The model accuracy was over 75% for two out of the three houses. Predicting accuracy of more than 90% was not expected in this analysis, as factors of major importance (such as occupancy or activity inputs) were not available and thus were not incorporated into the model. In general, it would be easier to extract safer conclusions if the data from all the seven houses were met the criteria. Furthermore, the decomposition of this time-series problem into explanatory input variables gives more space for creativity and understanding of the problem itself.

For smart cities administrators and electricity providers, such an approach, would face the common data-mining problem of cold start. For a citizen to learn information and forecasts about his home, it is necessary to record and process data for a long time. However, the approach, as described earlier in Chapter 7, which decomposes the time factor and assumes that same weather conditions induce same consuming behaviour irrespective of the day or season, and can be implemented much faster.

Regarding the first part of this work, the sensors that were measuring the consumption of several appliances proved that through a well-defined set of them one could calculate the total consumption with very small error. Thus, it is recommended for smart homes to install smart meters to key appliances over the house, besides the main electricity meter, which facilitate data availability and analysis. These meters could also be utilised for remote and more efficient control, under the smart city concept.

9 Conclusions and Future Research Directions

9.1 Conclusions

This last chapter summarizes the results. This research was conducted mostly with a citizen-centric approach; and thus, it was focused on the short-term load forecasting of individually homes. As two different data sets were used, and two different parts were examined (prediction through sensors and forecasting based on weather conditions) the conclusions are summarised in the following issues:

1. Data Acquisition
2. Pre-processing and input variables
3. Machine learning algorithms
4. Extracted knowledge

9.1.1 Data Acquisition

Data acquisition is a critical component of the whole process. Although, an existing data repository was used, and data acquisition was not part of this study, the importance of an adequate database has been revealed. The data collection process should be constant and clearly defined in advance based on the scope of the study. This is achieved by detailed infrastructure design. During the data analysis conducted in this work, a great amount of unexploited valuable data was identified. This was because either data were not consistently monitored, or their structure was not clearly defined. This resulted in a smaller dataset available for the analysis than the one that has been initially considered given the number of variables.

Only three of the seven houses contained a complete dataset, which can be utilised for data analysis, prediction models development and assessment. For this reason, it was decided that the first part of the work should focus on a secondary way of acquiring the total electricity consumption for each house. In this way, for the second part (load forecasting), the dataset was in an almost perfect shape for all timestamps; however, this was not enough due to the incomplete form of the Home dataset.

9.1.2 Pre-processing and input variables

As explained above, a major pre-processing step was to match the timestamps of each Home for every dataset due to insufficient data acquisition. The classic pre-processing, however, is not an appropriate solution on that kind of problems. For load forecasting, besides the missing values, outlier detection has no meaning. A trustworthy model should also be able to explain and predict extreme situations, since it is associated with individual houses. Moreover, data aggregation for On-peak and Off-peak periods during a day, gave unexpectedly different results for Home B, while the rest of the houses demonstrated similar predictability.

The effect that two subsets of the total number of variables could induce in terms of RMSE and adjusted R-squared was examined in the first part. Results showed that two homes gave reasonably accepted results in comparison with the original full set of available sensors and only for Home F this was not a case.

The input variables of the weather dataset were already specific, and it was nearly impossible to access different valid weather data for these houses, since the exact location is unknown. Not all the weather variables were taken into consideration though, due to the increased complexity that they would cause, as well as the very low standard deviation. For this reason, only the most common weather variables were included, such as air temperature and humidity. In addition to the available weather data, two more variables were introduced in order to counter the seasonality which is associated with the electricity consumption. Since this study is focused on day ahead forecast, each day was characterized as holiday or weekday. The dummying process was followed for all the categorical variables. The most contributing was the weather icon that was used both as one-condition variable for the *Clear* icon and for the whole of icons, however, the results were almost identical. The two variables used to interpret two of the most common consumption cycles, daily and weekly, increased the accuracy for all the algorithms, but not to a large extent. It is expected though that adding more of them could boost the accuracy even further.

Finally, a brief analysis of the variables showed that there is none that contributes to the model heavier than others; yet the feature importance was not examined and therefore no reliable conclusion can be made.

9.1.3 Machine learning algorithms

The different machine learning algorithms (Random Forest, SVM, Logistic Regression etc.) that were examined in this study exhibit different characteristics and strengths. Generally, in the first stage Random Forest was much more stable than Gradient Boosting, while on the second and main stage SVM was also very competitive. Classic regression models such as linear, which was indicatively used, cannot compete with the more sophisticated machine learning methods.

The basic conclusion is that the complexity of the model was not so large as to give the margin to one of the algorithms to stand out. All the results are very close and sometimes the same. Specific algorithms though have specific requirements. For Random forest, data scaling is ineffective while hyper-parameter tuning can boost predictability significantly. SVM is a traditionally powerful algorithm that is preferred from many researchers for short or medium-term load forecasting. This algorithm gives plenty of options however not a rule in scaling data, while hyper-parameter tuning is much more intense and actually performs slower in comparison with others.

9.1.4 Extracted knowledge

The most important results of this work are summarized below:

- The accuracy of the model increases when the consumption follows similar paths through months. For Home F, although it has, a smooth and similar average consuming daily behaviour; it is not the same for average monthly consumption.
- Home C is twice as big as Home B, and this may indicate that the bigger a home is, the more difficult it is to predict its consuming behaviour. This can be associated also with the higher and thus more diverse occupancy patterns that are usually expected to bigger houses. Unfortunately, there is no information for Home F, however through its sensors on two garages and three water heaters it is expected to be larger than that of Home B.
- The most significant sensors through Homes are those applied to HVAC appliances, as the main electricity consumption is associated with them.
- The top-3 concept of sensors achieves acceptable results only for Home B, while the top-6 for both Home B and Home C.

- Transforming the consumption from a real number into a binary class achieves better results when the point of division is just the mean of all instances. Next in performance is when the division is based on a seasonal mean and finally on a monthly mean value; the first two seems to be the more reasonable.
- Unexpectedly, Home F and Home C have no differentiation between the predictions around On and Off-peak periods. In Home B there is a significant decrease on Off-peak period, possibly due to the fact that in Home B none of the periods is actually of low consumption.
- The most important part of the analysis is the proper feature selection, as the selection of algorithms or the evaluation technique (i.e. Cross-validation, Manual split) do not affect that much the result.

9.2 Threats to validity

Every research and scientific work 'suffer' from several threats that might question the validity of the approach and thus the results. In this case, a major threat is that of sufficiently deseasonalising data. The factor of time could be possible analysed to more explanatory variables.

Another significant factor is the access to accurate weather data. Since the model examines the information that weather data can accommodate forecasting, those data should be accurate. However, obviously forecasting on day-ahead period also the weather data themselves are achieved through prediction. Since the problem is not treated as a time-series one, it is not clear which method against overfitting is more appropriate. In general, in time-series problems there is a different way to use cross-validation than the classic, which was described in 6.2.2.

In addition, the size of the homes is much bigger than a typical house or a regular apartment. Since the electricity consumption is also affected from the size, may in smaller residences the results would be different.

Finally, the last threat is the selected metric to compare the models against. Accuracy is not always the more appropriate metric especially when the classes are imbalanced. Generally, the recall for *Low* class was much higher while for *High* class it was much lower.

9.3 Future research directions

Such work and research process require time-consuming analysis and different approaches. For that reason, a big effort was spared to cover as many aspects as possible. However, it is never possible to explore every aspect, especially when the time of delivery is limited.

The biggest missing part of this work is the experiments of the constructed model on different houses. The *UMass* Trace repository is expected to release new data for different homes by the end of 2018 and thus it could bring extra knowledge about the model by applying the same techniques for those. Regarding features selection, it was desired to introduce the concept of the *week of the month*, which is increasingly used by such efforts. More specifically, it is claimed that people tend reasonably or not to consume higher amounts of electricity a specific week of the month. Such a distinction between days could bring probably better results. In addition, another feature could also regard the summation of consumption if the provider follows an escalated pricing policy. It is common in such cases, that the consumers reduce their consumption when they pass a threshold, thus reaching a new level of pricing during a month. Occupancy or activity monitoring could also be recorded and examined as an input in the electricity forecasting models. This could have a significant impact on model's accuracy given the scale of the application.

Regarding the machine learning algorithms under examination, it was desired to also test neural networks, which however needs a wide hyper-parameter tuning and it was decided to be excluded from this work.

The perspective of merging the data of each house in order to simulate a tiny grid could also gather more interest, but as such, it could also eliminate the factor of personalisation, which is the biggest challenge of this work.

References

[1] Khan Z, Kiani SL (2012) A cloud-based architecture for citizen services in smart cities. In: ITAAC Workshop 2012. IEEE Fifth International Conference on Utility and Cloud Computing (UCC), Chicago, IL, USA. pp 315–320. IEEE

[2] A.Alkandari, M. Alnasheet, I. Alshekhly,(2012) Smart Cities: Survey

[3] S. Dirks, M. Keeling, J. Dencik. (2009) How smart is your city? Helping cities measure progress

[4] M. Mazhar Rathore , Anand Paul , Awais Ahmad , Suengmin Rho , Urban planning and building smart cities based on the internet of things using big data analytics, Computer Networks (2016), DOI: 10.1016/j.comnet.2015.12.023

[5] Al Nuaimi et al. (2015) Journal of Internet Services and Applications 6:25 DOI 10.1186/s13174-015-0041-5

[6] United Nations Environment Programme. (2015) The Global Initiative for Resource Efficient Cities

[7] Edward L. Glaeser and Bruce Sacerdote "Why Is There More Crime in Cities?" Journal of Political Economy, Vol. 107, no. 6, part 2 (December 1999) 225

[8] L. Edwards (2016) Privacy, security and data protection in smart cities: a critical EU law perspective

[9] N. Akhtar (2018) How the new GDPR helps smart cities adoption in the EU https://technology.ihs.com/603153/how-the-new-gdpr-helps-smart-cities-adoption-in-the-eu

[10] Privacy regulators study finds Internet of Things shortfalls (2016) https://ico.org.uk/about-the-ico/news-and-events/news-and-blogs/2016/09/privacy-regulators-study-finds-internet-of-things-shortfalls/

[11] HP study finds alarming vulnerabilities with IoT home security systems (2015) http://www8.hp.com/us/en/hp-news/press-release.html?id=1909050

[12] How the EU's GDPR will be a game changer for cities using open data (2018) https://eu.smartcitiescouncil.com/article/how-eus-gdpr-will-be-game-changer-cities-using-open-data

[13] A. Atwood, M. Merabti, P. Fergus, O. Abuelmaatti (2011) SCCIR: Smart cities critical infrastructure response framework

[14] X. Feng, T. Yu-Chu, L. Yanjun, Sun Y. (2007) Wireless Sensor/Actuator Network Design for Mobile Control applications

[15] X. Zhang, H. Bao, K. Yan, H. Zhang, J. Ye. (2013) A data gathering scheme for WSN/WSAN based on partitioning algorithm and mobile sinks

[16] S. Tadic, A. Favenza, C. Kavadias, V. Tsagaris (2016) GHOST: A novel approach to smart city infrastructures monitoring through GNSS precise positioning

[17] H. Ghorbanpanah, A. Saeedi, S. Alishahi, N. Nakhodchi (2016) A competitive model for apartment complexes energy management with the aim of creating the infrastructure of smart city

[18] P. Sotres, J. R. Santana, L. Sanchez, J. Lanza, L. Munoz (2017) Practical lessons from the deployment and management of a smart city IoT infrastructure: The SmartSantander Testbed case

[19] A. Karkouch, H. Mousannif, H. Al Moattasime, T. Noel (2016) Data quality in internet of things: A state-of-the-art survey

[20] P. Berrone, J. E. Ricart (2018) IESE Cities in motion index

[21] G. Vaseekaran (2017) Big Data Battle: Batch processing vs Stream processing https://medium.com/@gowthamy/big-data-battle-batch-processing-vs-stream-processing-5d94600d8103

[22] B. Marr (2014) Big Data: The 5 Vs Everyone Must Know https://www.linkedin.com/pulse/20140306073407-64875646-big-data-the-5-vs-everyone-must-know

[23] A. Devan (2016) The 7 V's of Big Data https://impact.com/marketing-intelligence/7-vs-big-data/

[24] J.M Cavanillas, E. Curry, W. Wahlster (2016) New Horizons for a Data-Driven Economy [A roadmap for usage and exploitation of Big Data in Europe]

[25] Formerly Booz & company, "Benefiting from big data: A new approach for the telecom industry," in 2013

[26] C.L. Philip Chen, C.-Y. Zhang, Data-intensive applications, challenges, techniques and technologies: A survey on Big Data, Inform. Sci. (2014), http://dx.doi.org/10.1016/j.ins.2014.01.015

[27] D. Wang, Z. Zhiwei, Big data analysis and parallel load forecasting of electric power user side

[28] V. Peristeras (2017) Data Science for Business [Theory and Practice]

[29] S.P. Mohanty, U. Choppali, E. Kougianos (2016). Everything you wanted to know about smart cities

[30] Big Data is coming to a Government near You (2015). https://www.nec.com/en/global/ad/insite/article/bigdata03.html

[31] J. Han, M. Kamber, J. Pei (2012) Data Mining Concepts and Techniques 3rd Edition

[32] Ian H. Witten, Eibe Frank, Mark A. Hall, Christopher J. Pal, Data Mining: Practical Machine Learning Tools and Techniques 4th edition

[33] Data Mining Functionalities http://www.lastnightstudy.com/Show?id=37/Data-Mining-Functionalities

[34] S. Santoyo (2017) A brief overview of outlier detection techniques https://towardsdatascience.com/a-brief-overview-of-outlier-detection-techniques-1e0b2c19e561

[35] H. Blockeel, M. Sebag (2004) Scalability and Efficiency in multi-relational data mining

[36] D. K. Singh, V. Swaroop (2013) Data Security and Privacy in Data Mining: Research Issues and Preparation

[37] http://www.statisticssolutions.com/what-is-linear-regression/

[38] https://www.statisticshowto.datasciencecentral.com/rmse/

[39] https://people.duke.edu/~rnau/compare.htm

[40] T.Chai, R.R Draxler (2014) Root mean square error (RMSE) or mean absolute error (MAE)? – Arguments against avoiding RMSE in the literature

[41] https://towardsdatascience.com/coefficient-of-determination-r-squared-explained-db32700d924e

[42] https://towardsdatascience.com/coefficient-of-determination-r-squared-explained-db32700d924e

[43] Adjusted R-squared in Python (2017) https://statinfer.com/204-1-7-adjusted-r-squared-in-python/

[44] https://people.duke.edu/~rnau/411regou.htm

[45] Alan Marshall, 2001, Regression Analysis – Time series Analysis ppt available at *www.yorku.ca/amarshal/MGTS5120L05.ppt*

[46] Kecman, Vojislav. (2005). Support Vector Machines – An Introduction. Support Vector Machines: Theory and Applications. 177. 605-605

[47] https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/

[48] https://machinelearningmastery.com/logistic-regression-for-machine-learning/

[49] http://www.chioka.in/differences-between-l1-and-l2-as-loss-function-and-regularization/

[50] https://towardsdatascience.com/l1-and-l2-regularization-methods-ce25e7fc831c

[51] D. Chen, D. Irwin (2017) Weatherman: Exposing Weather-based Privacy Threats in Big Energy Data

[52] A. Mishra, D. Irwin, P. Shenoy, J. Kurose, T. Zhu (2012) SmartCharge: Cutting the Electricity Bill in Smart Homes with Energy Storage

[53] S. Barker, A. Mishra, D. Irwin, P. Shenoy, J. Albrecht (2012) SmartCap: Flattening Peak Electricity Demand in Smart Homes

[54] O.M. Longe, K. Ouahada, S. Rimer, H. Zhu, H.C Ferreira (2015) Effective Energy Consumption Scheduling in Smart Homes

[55] A. Azadech, S.F Ghaderi, S. Sohrabkhami (2007) Forecasting electrical consumption by integration of Neural Network, time-series and ANOVA

[56] A. Ahmad, M. Hassan, M. Abdullah, H. Rahman, F. Hussin, H Abdullah, R. Saidur (2014) A review on applications of ANN and SVM for building electrical energy consumption forecasting

[57] X. Li, C.P. Bowers, T. Schnier (2010) Classification of Energy Consumption in Buildings With Outlier Detection

[58] M. Espinoza, C. Joye, R. Belmans, B.D Moor (2005) Short-Term Load Forecasting, Profile Identification, and Customer Segmentation: A Methodology Based on Periodic Time Series

[59] Q. Wang (2009) Grey Prediction Model and Multivariate Statistical Techniques Forecasting Electrical Energy Consumption in Wenzhou, China

[60] A. Sauhats, R. Varfolomejeva, O. Linkevics, R. Petrecenko, M. Kunickis, M. Balodis (2015) Analysis and prediction of electricity consumption using smart meter data

[61] J.Contreras, R. Espinola, F. Nogales, A. I. Conejo (2003) ARIMA Models to Predict Next-Day Electricity Prices

[62] C. Xia, J. Wang, K. McMenemy (2010) Short, medium and long term load forecasting model and virtual load forecaster based of radial basis function neural networks

[63] R. Torkzadeh ET. Al. (2014) Medium Term Load Forecasting in Distribution Systems Based on Multi Linear Regression & Principal Component Analysis: A Novel Approach

[64] A. D. Amato, M. Ruth, P. Kirshen, J. Horwitz (2005) Regional energy demand responses to climate change: Methodology and Application to the commonwealth of Massachusetts

[65] V. Bianco, O. Manca, S. Nardini (2009) Electricity consumption forecasting in Italy using linear regression models

[66] G. Jannuzzi, L. Schipper (1991) The structure of electricity demand in the Brazilian household sector

[67] D. Jose, M. Mathew, A. Krishman (2016) Weather Dependency of Electricity Demand: A case study in Warm Humid Tropical Climate

[68] E. Almeshaiei, H. Soltan (2011) A methodology for Electric Power Load Forecasting

[69] Y. Mo, T. Hyun Jim Kim, K. Brancik, D. Dickinson, H. Lee, A. Perrig, B. Sinopoli (2011) Cyber-Physical security of a Smart Grid Infrastructure

[70] S. Zeng, J. Li, Y. Ren, Research of Time-of-Use Electricity Pricing Models in China: A survey

[71] https://www.eia.gov/energyexplained/index.php?page=electricity_factors_affecting_prices

[72] R. Huisman, C. Huurman, R. Mahieu (2006) Hourly Electricity prices in day-ahead markets

[73] http://www.whatissmartgrid.org/featured-article/what-you-need-to-know-about-dynamic-electricity-pricing
[74] H. M Fadzili, S. Hazlina, Y. Rubiyah, B. Salinda, F. S. Ismail (2013) Review of HVAC scheduling techniques for buildings towards energy-efficient and cost-effective operations
[75] S. Mirasgedis, Y. Sarafidis, E. Georgopoulou, DP. Lalas, M. Moschovits, F. Karagiannis et al (2006) Models for mid term electricity demand forecasting incorporating weather influences
[76] S. Barker, A. Mishra, D. Irwin, E. Cecchet, P. Shenoy (2012) Smart*: An open data set and tools for enabling research in sustainable homes
[77] https://www.sec.state.ma.us/cis/cishol/holidx.htm
[78] R. Lawson, P. Thorsnes, J. Williams (2011) Consumer Response to Timen Varying Prices for Electricity
[79] http://www.weatherquestions.com/What_is_dewpoint_temperature.htm
[80] https://uni.edu/storm/Wind%20Direction%20slide.pdf

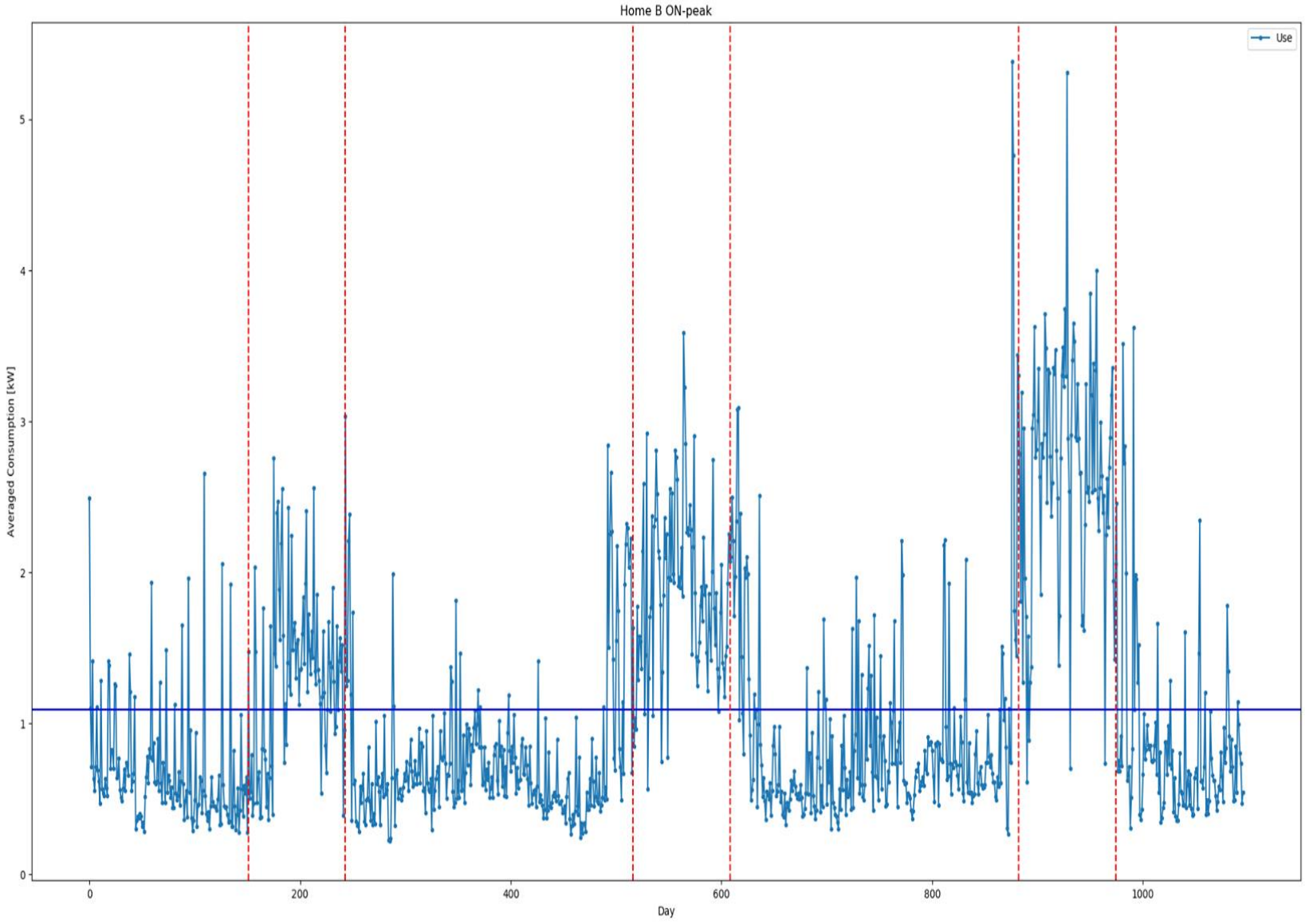
Figures

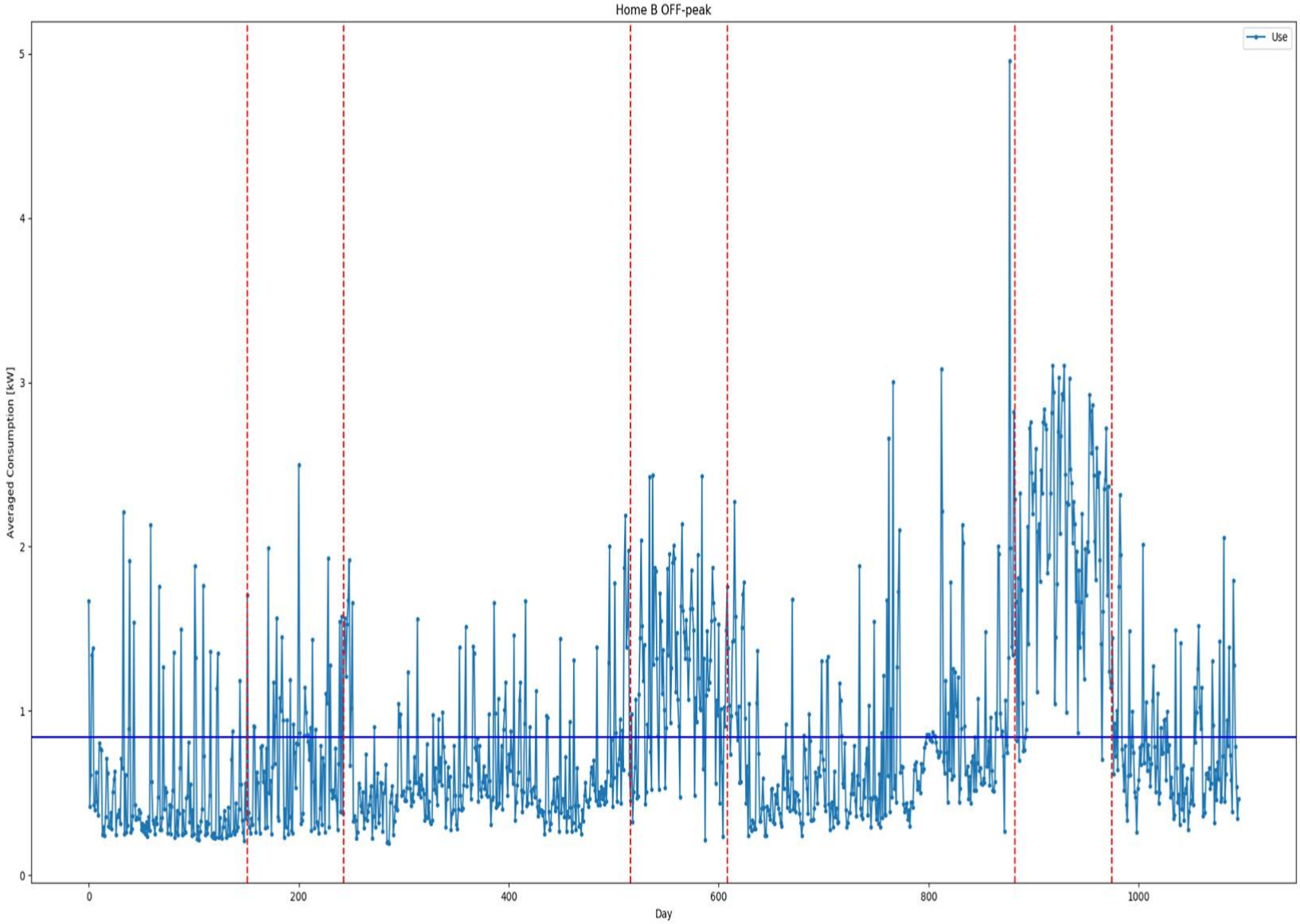
[1] https://www.rfpage.com/what-are-the-major-components-of-internet-of-things/
[2] https://www.mushroomnetworks.com/infographics/the-landscape-of-big-data-infographic/

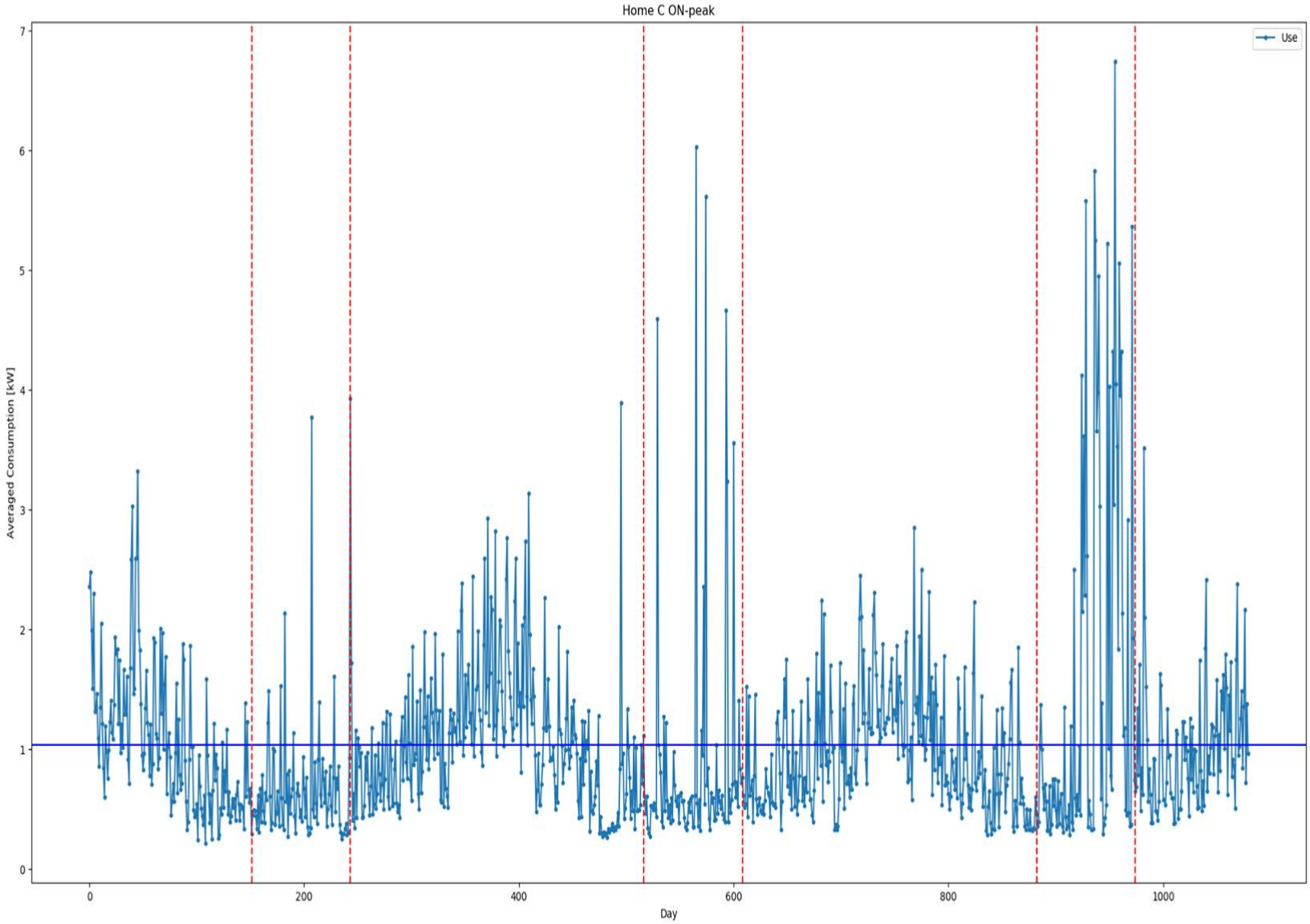
Appendix

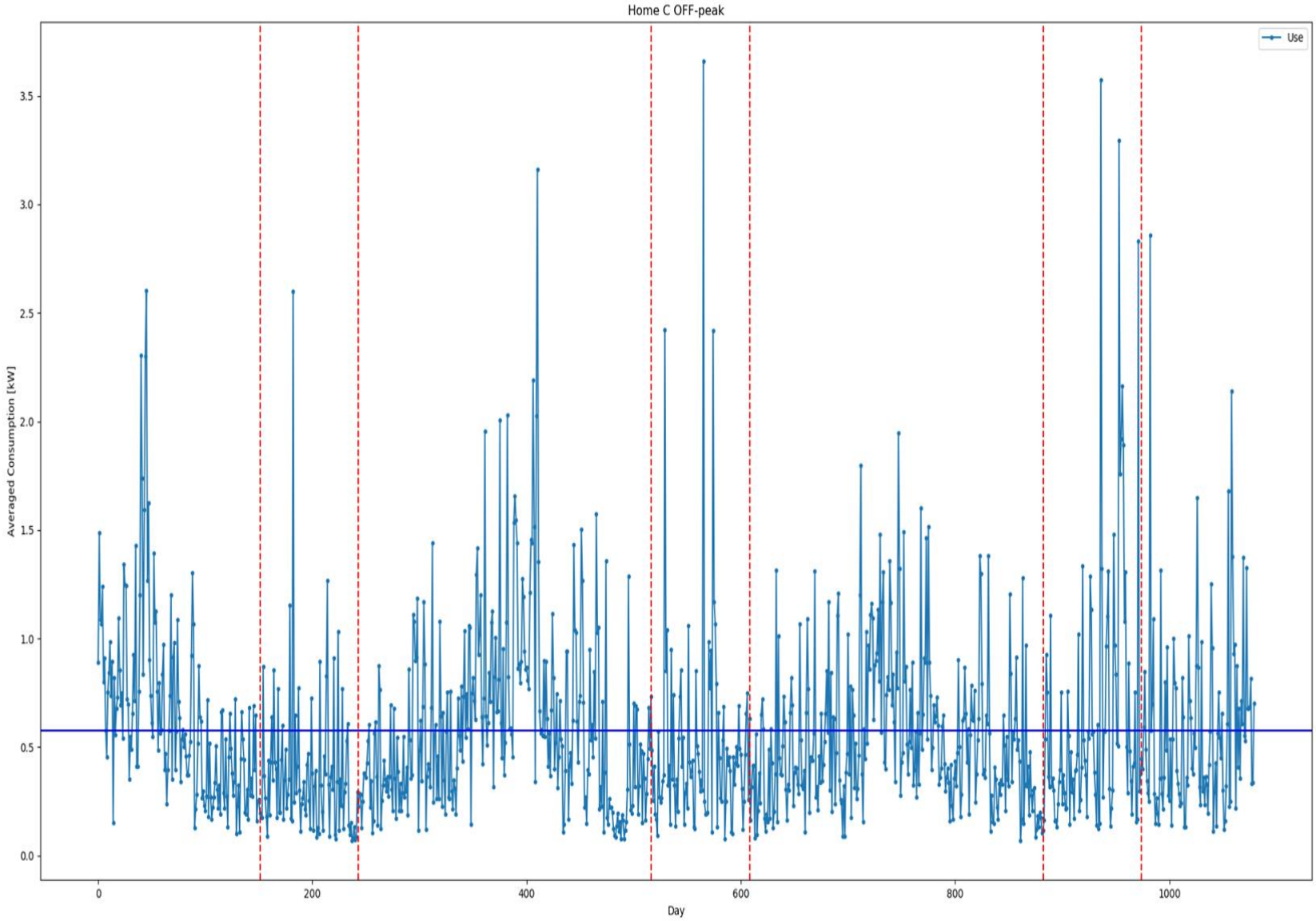
Below one can find some indicative figures of consumption for each Home throughout the completely available period. Besides that, there are some data samples for each dataset as long as the scripts written in Python.

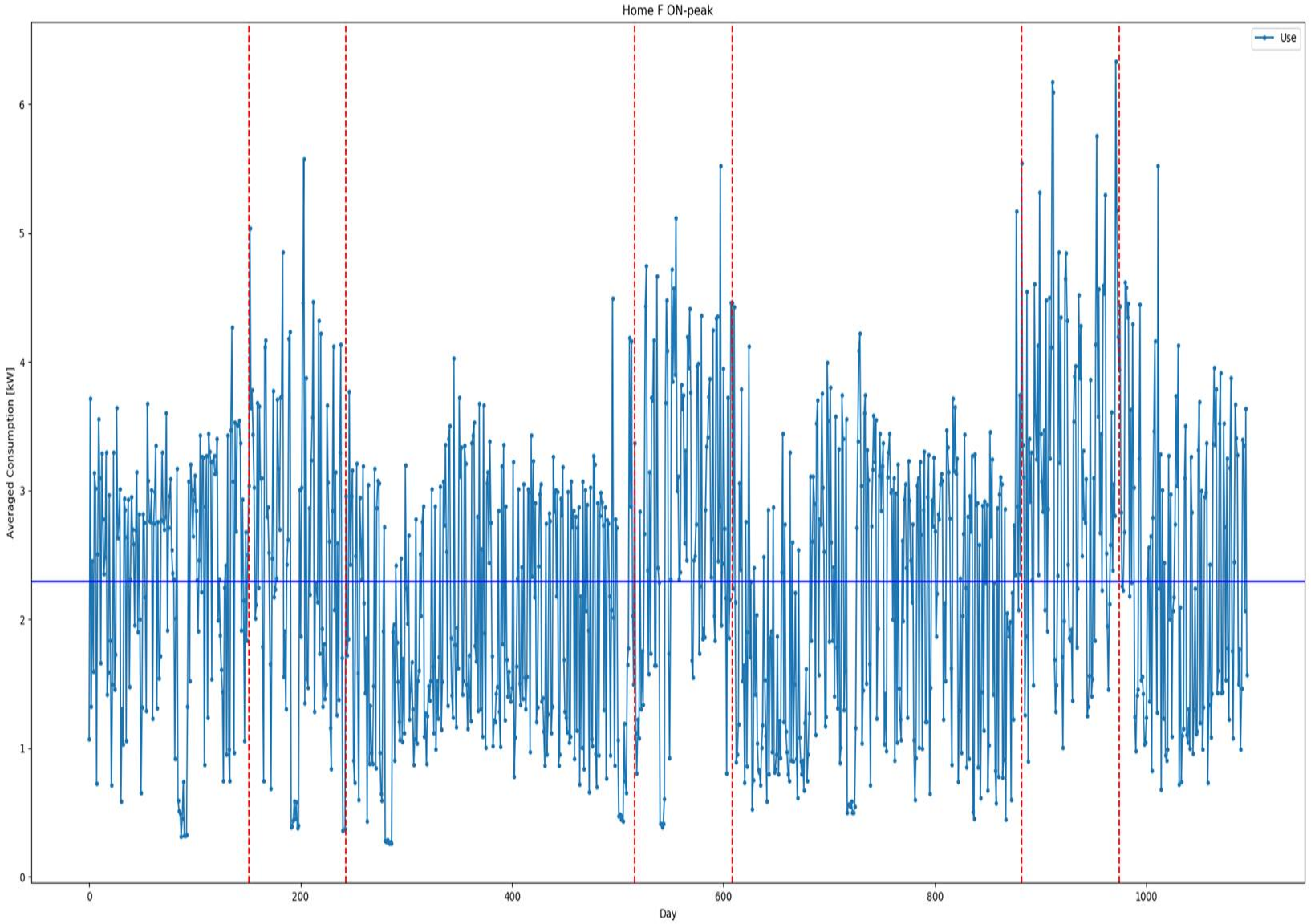
On the figures of electricity consumption per day, the deep blue horizontal line indicates the total average consumption, while the red dotted lines define the summer periods.

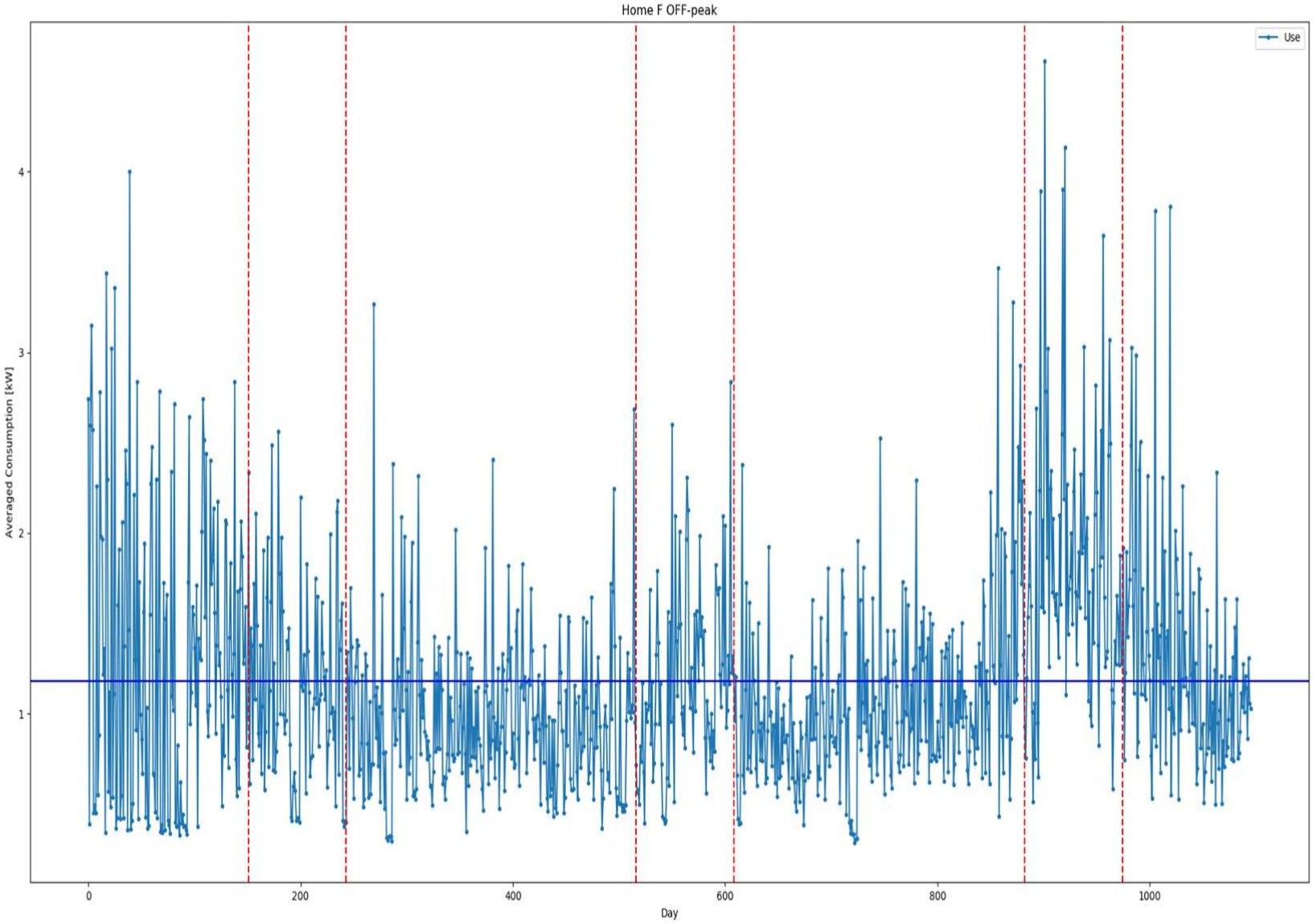

Home B ON-peak
Use
Averaged Consumption [kW]
Day
0
1
2
3
4
5
0
200
400
600
800
1000


Home B OFF-peak
Use
Averaged Consumption [kW]
Day
0
1
2
3
4
5
0
200
400
600
800
1000


Home C ON-peak
Use
Averaged Consumption [kW]
Day
0
1
2
3
4
5
6
7
0
200
400
600
800
1000


Home C OFF-peak
Use
Averaged Consumption [kW]
Day
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
0
200
400
600
800
1000


Home F ON-peak
Use
Averaged Consumption [kW]
Day
0
1
2
3
4
5
6
0
200
400
600
800
1000


Home F OFF-peak
Use
Averaged Consumption [kW]
Day
1
2
3
4
0
200
400
600
800
1000

DATA SAMPLES

Home B - Meter 1

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	
Date & Time	use [kW]	gen [kW]	Grid [kW]	AC [kW]	Furnace [kW]	Cellar Lights [kW]	Washer [kW]	First Floor lig	Utility Rm	Garage ou	MBed + KE	Dryer + eg	Panel GFI	Home Off	Dining roc	Microwav	Fr
01/01/2014 00:00	0.304439	0	0.304439	5.78E-05	0.009531111	0.005335556	0.000125556	0.011175	0.003836	0.004836	0.002132	8.89E-06	0.007159	0.063666	0.004299	0.004733	0
01/01/2014 00:30	0.656771	0	0.656771	0.001534	0.364338333	0.005522222	4.33E-05	0.00351444	0.003512	0.004888	0.002137	0.000107	0.007221	0.064698	0.003589	0.004445	0
01/01/2014 01:00	0.612895	0	0.612895	0.001847	0.417988889	0.005503889	4.44E-05	0.00352778	0.003484	0.004929	0.002052	0.00017	0.007197	0.065109	0.003522	0.004396	0
01/01/2014 01:30	0.683979	0	0.683979	0.001744	0.410652778	0.005556111	5.94E-05	0.00349889	0.003476	0.004911	0.002068	0.000121	0.007236	0.065032	0.003404	0.004262	0
01/01/2014 02:00	0.197809	0	0.197809	3.00E-05	0.017152222	0.005301667	0.000118889	0.00369389	0.003865	0.004876	0.002087	5.22E-05	0.007133	0.062451	0.003915	0.004407	0
01/01/2014 02:30	0.397099	0	0.397099	0.000442	0.12696	0.005415	5.44E-05	0.00362667	0.003749	0.004891	0.002133	2.44E-05	0.007187	0.063814	0.003813	0.004398	0
01/01/2014 03:00	0.590319	0	0.590319	0.001858	0.420358333	0.005508889	4.33E-05	0.00356222	0.003541	0.005007	0.002072	0.000197	0.007208	0.065277	0.003008	0.004008	0
01/01/2014 03:30	0.538266	0	0.538266	0.001071	0.257654444	0.005507222	2.00E-05	0.00358167	0.003549	0.004837	0.002136	4.67E-05	0.007198	0.063142	0.004142	0.004565	0

Home B - Weather data

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
temperatu	icon	humidity	visibility	summary	apparentT	pressure	windSpee	cloudCove	time	windBeari	precipInte	dewPoint	precipProbability		Date
20.31	clear-nigh	0.47	10	Clear	7.74	1023.25	12.65	0	1388552400	283	0	3.3	0		01/01/2014
19.22	clear-nigh	0.48	10	Clear	7.89	1023.21	9.99	0	1388556000	283	0	2.69	0		01/01/2014
18.1	clear-nigh	0.52	10	Clear	8.11	1024.25	7.76	0	1388559600	255	0	3.42	0		01/01/2014
17.47	clear-nigh	0.55	10	Clear	7.36	1025.02	7.73	0	1388563200	245	0	4	0		01/01/2014
15.63	clear-nigh	0.61	10	Clear	6.98	1025.16	5.76	0	1388566800	220	0	4.45	0		01/01/2014
15.33	clear-nigh	0.63	10	Clear	6.21	1025.48	6.16	0	1388570400	219	0	4.87	0		01/01/2014
15.33	clear-nigh	0.65	10	Clear	6.09	1026.24	6.28	0	1388574000	222	0	5.68	0		01/01/2014
15.6	clear-nigh	0.65	10	Clear	6.73	1026.76	5.97	0.04	1388577600	211	0	5.86	0		01/01/2014
15.31	clear-day	0.67	10	Clear	6.4	1027.55	5.96	0	1388581200	228	0	6.25	0		01/01/2014
18.8	clear-day	0.63	9.73	Clear	10.31	1028.36	6.21	0	1388584800	228	0	8.19	0		01/01/2014

Meters Pre-processing

```
import numpy as np
import pandas as pd
import time
from datetime import datetime


# Import datasets of first meter & remove duplicates (remdup)

df1_2014 = pd.read_csv('HomeF-meter1_2015.csv')
print('Meter 1 - 2014', df1_2014.shape)
remdup_df1_2014 = df1_2014.drop_duplicates(subset='Date & Time')
print(remdup_df1_2014.shape)

df1_2015 = pd.read_csv('HomeF-meter2_2015.csv')
print('Meter 1 - 2015', df1_2015.shape)
remdup_df1_2015 = df1_2015.drop_duplicates(subset='Date & Time')
print(remdup_df1_2015.shape)

df1_2016 = pd.read_csv('HomeF-meter2_2016.csv')
print('Meter 1 - 2016', df1_2016.shape)
remdup_df1_2016 = df1_2016.drop_duplicates(subset='Date & Time')
print(remdup_df1_2016.shape)
#remdup df1 2016.to csv('HomeB-meter1 2016 pp.csv', index=False)


# Import datasets of second meter & remove duplicates (remdup)

df2_2014 = pd.read_csv('HomeF-meter3_2014.csv')
print('Meter 2 - 2014', df2_2014.shape)
remdup_df2_2014 = df2_2014.drop_duplicates(subset='Date & Time')
print(remdup_df2_2014.shape)

df2_2015 = pd.read_csv('HomeF-meter3_2015.csv')
print('Meter 2 - 2015', df2_2015.shape)
remdup_df2_2015 = df2_2015.drop_duplicates(subset='Date & Time')
print(remdup_df2_2015.shape)

df2_2016 = pd.read_csv('HomeF-meter3_2016.csv')
print('Meter 2 - 2016', df2_2016.shape)
remdup_df2_2016 = df2_2016.drop_duplicates(subset='Date & Time')
print(remdup_df2_2016.shape)


# Renaming the Date & Time column for convenience

remdup_df1_2014.columns = ['DateTime' if x == 'Date & Time' else x for
x in remdup_df1_2014.columns]
remdup_df1_2015.columns = ['DateTime' if x == 'Date & Time' else x for
x in remdup_df1_2015.columns]
remdup_df1_2016.columns = ['DateTime' if x == 'Date & Time' else x for
x in remdup_df1_2016.columns]

remdup_df2_2014.columns = ['DateTime' if x == 'Date & Time' else x for
x in remdup_df2_2014.columns]
remdup_df2_2015.columns = ['DateTime' if x == 'Date & Time' else x for
x in remdup_df2_2015.columns]
remdup_df2_2016.columns = ['DateTime' if x == 'Date & Time' else x for
x in remdup_df2_2016.columns]
```

```
# Transforming DateTime column into datetime object so as to be manip-
ulated easier

remdup_df1_2014['DateTime'] =
pd.to_datetime(remdup_df1_2014['DateTime'])
remdup_df1_2015['DateTime'] =
pd.to_datetime(remdup_df1_2015['DateTime'])
remdup_df1_2016['DateTime'] =
pd.to_datetime(remdup_df1_2016['DateTime'])

remdup_df2_2014['DateTime'] =
pd.to_datetime(remdup_df2_2014['DateTime'])
remdup_df2_2015['DateTime'] =
pd.to_datetime(remdup_df2_2015['DateTime'])
remdup_df2_2016['DateTime'] =
pd.to_datetime(remdup_df2_2016['DateTime'])

# Keeping only instances that reflect hourly and half hourly consump-
tions

remdup_df1_2014 = rem-
dup_df1_2014[(remdup_df1_2014['DateTime'].dt.minute == 0) |
                                    (rem-
dup_df1_2014['DateTime'].dt.minute == 30)]
remdup_df1_2015 = rem-
dup_df1_2015[(remdup_df1_2015['DateTime'].dt.minute == 0) |
                                    (rem-
dup_df1_2015['DateTime'].dt.minute == 30)]
remdup_df1_2016 = rem-
dup_df1_2016[(remdup_df1_2016['DateTime'].dt.minute == 0) |
                                    (rem-
dup_df1_2016['DateTime'].dt.minute == 30)]

remdup_df2_2014 = rem-
dup_df2_2014[(remdup_df2_2014['DateTime'].dt.minute == 0) |
                                    (rem-
dup_df2_2014['DateTime'].dt.minute == 30)]
remdup_df2_2015 = rem-
dup_df2_2015[(remdup_df2_2015['DateTime'].dt.minute == 0) |
                                    (rem-
dup_df2_2015['DateTime'].dt.minute == 30)]
remdup_df2_2016 = rem-
dup_df2_2016[(remdup_df2_2016['DateTime'].dt.minute == 0) |
                                    (rem-
dup_df2_2016['DateTime'].dt.minute == 30)]

# Reseting index and checking if dimensions match

remdup_df1_2014.reset_index()
#print(remdup_df1_2014['DateTime'].tail())
print('Meter 1 - 2014', remdup_df1_2014['DateTime'].shape)
print('-----')

remdup_df1_2015.reset_index()
#print(remdup_df1_2015['DateTime'].tail())
print('Meter 1 - 2015', remdup_df1_2015['DateTime'].shape)
print('-----')

remdup_df1_2016.reset_index()
#print(remdup_df1_2016['DateTime'].tail())
```

```
# Since Meter 2 -2016 has not information about the last 16 days of
year, the last 48*16 = 768 rows are dropped for
# Meter 1 as well
# remdup_df1_2016.drop(remdup_df1_2016.tail(768).index, inplace=True)
print('Meter 1 - 2016', remdup_df1_2016['DateTime'].shape)
print('-----')


remdup_df2_2014.reset_index()
#print(remdup_df2_2014['DateTime'].tail())
print('Meter 2 - 2014', remdup_df2_2014['DateTime'].shape)
print('-----')

remdup_df2_2015.reset_index()
#print(remdup_df2_2015['DateTime'].tail())
print('Meter 2 - 2015', remdup_df2_2015['DateTime'].shape)
print('-----')

remdup_df2_2016.reset_index()
#print(remdup_df2_2016['DateTime'].tail())
print('Meter 2 - 2016', remdup_df2_2016['DateTime'].shape)
print('-----')


# Extracting data into new form ready to analyse

# remdup_df1_2014.to_csv('F-meter1_2014.csv', index=False)
# remdup_df1_2015.to_csv('F-meter1_2015.csv', index=False)
# remdup_df1_2016.to_csv('F-meter1_2016.csv', index=False)

# remdup_df2_2014.to_csv('F-meter2_2014.csv', index=False)
# remdup_df2_2015.to_csv('F-meter2_2015.csv', index=False)
# remdup_df2_2016.to_csv('F-meter2_2016.csv', index=False)
```

Sensors Regression

```
import numpy as np
import pandas as pd
from sklearn.model_selection import GridSearchCV
from sklearn.ensemble import RandomForestRegressor
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.model_selection import train_test_split
from sklearn import linear_model
from math import sqrt

df1 = pd.read csv('HomeB-meter1.csv')
df1 = df1.drop('gen [kW]', 1)
df1 = df1.drop('Grid [kW]', 1)
df1 = df1.drop('DateTime', 1)
print(df1.shape)
df1 = df1[(df1 != 0).all(1)]
print(df1.shape)
df2 = pd.read_csv('HomeB-meter2.csv')
df2 = df2.drop('gen [kW]', 1)
# df2 = df2.drop('old usage [kW]', 1)
df2 = df2.drop('DateTime', 1)
```

```
# ----------------------------------------------------------------------

Y = df1['use [kW]']
X = df1.drop('use [kW]', 1)

X_train, X_test, y_train, y_test = train_test_split(X, Y,
test_size=0.33, random_state=42)

# Regression = GradientBoostingRegressor()
Regression = RandomForestRegressor()
n_estimators = [10, 20, 30,  40, 50]
max_depth = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
param_grid = {'n_estimators': n_estimators, 'max_depth': max_depth}

# REG = GridSearchCV(Regression, param_grid={}, cv=10, verbose=10,
scoring='r2')
REG = GridSearchCV(Regression, param_grid={}, cv=10, verbose=10, scor-
ing='neg_mean_squared_error')
REG.fit(X_train, y_train)

print('RMSE =', sqrt(abs(REG.cv_results_['mean_test_score'])))
# print('R-squared =', REG.cv_results_['mean_test_score'])
# print('Adjusted R-squared =', 1-(1-
REG.cv_results_['mean_test_score']*(len(y_test)-1)/(len(y_test)-
X_test.shape[1]-1)))

# ----------------------------------------------------------------------

Y = df2['use [kW]']
X = df2.drop('use [kW]', 1)

X_train, X_test, y_train, y_test = train_test_split(X, Y,
test_size=0.33, random_state=42)

# Regression = GradientBoostingRegressor()
Regression = RandomForestRegressor()
n_estimators = [10, 20, 30,  40, 50]
max_depth = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
# param_grid = {'n_estimators': n_estimators, 'max_depth': max_depth}

# REG = GridSearchCV(Regression, param_grid={}, cv=10, verbose=10,
scoring='r2')
REG = GridSearchCV(Regression, param_grid={}, cv=10, verbose=10, scor-
ing='neg_mean_squared_error')
REG.fit(X_train, y_train)

print('RMSE =', sqrt(abs(REG.cv_results_['mean_test_score'])))
# print('R-squared =', REG.cv results ['mean test score'])
# print('Adjusted R-squared =', 1-(1-
REG.cv_results_['mean_test_score']*(len(y_test)-1)/(len(y_test)-
X_test.shape[1]-1)))
```

Aggregated Plots

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA
import csv

dfHour14 = pd.read_csv('C-meter1_2014_hour.csv')
dfHour15 = pd.read_csv('C-meter1_2015_hour.csv')
dfHour16 = pd.read_csv('C-meter1_2016_hour.csv')

dfMonth14 = pd.read_csv('C-meter1_2014_month.csv')
dfMonth15 = pd.read_csv('C-meter1_2015_month.csv')
dfMonth16 = pd.read_csv('C-meter1_2016_month.csv')

# dfHour14['DateTime'] = pd.to_datetime(dfHour14['DateTime'])
# dfHour15['DateTime'] = pd.to_datetime(dfHour15['DateTime'])
# dfHour16['DateTime'] = pd.to_datetime(dfHour16['DateTime'])

# dfHour14 = dfHour14[(dfHour14['DateTime'].dt.hour)]
# dfHour15 = dfHour15[(dfHour14['DateTime'].dt.hour)]
# dfHour16 = dfHour16[(dfHour14['DateTime'].dt.hour)]

# print(dfHour14.head())

def group_by_hour(a):
    dataagg = a.groupby(['DateTime'], as_index=False)['Use'].mean()
    dataagg.reset_index()
    return dataagg

def group_by_month(a):
    dataagg_sup = a.groupby(['DateTime'],
as_index=False)['Use'].mean()
    dataagg_sup.reset_index()
    dataagg = dataagg_sup.reindex([4, 3, 7, 0, 8, 6, 5, 1, 11, 10, 9,
2])
    return dataagg


Hour14 = group_by_hour(dfHour14)
Hour15 = group_by_hour(dfHour15)
Hour16 = group_by_hour(dfHour16)

# print(type(Hour14))
# print(Hour14.shape)
# print(Hour14.head())


Month14 = group_by_month(dfMonth14)
Month15 = group_by_month(dfMonth15)
Month16 = group_by_month(dfMonth16)
dataframe = dfHour14.iloc[:48]

print(Month14)
print(Month15)
print(Month16)

# Figure for average monthly consumption
# Consumptiond data are manually rounded on 2nd decimal
```

```
"""
# datalists for Home B
datalist1 = [('Jan', 0.62), ('Feb', 0.53), ('Mar', 0.59), ('Apr',
0.49), ('May', 0.43),
             ('Jun', 0.79), ('Jul', 1.15), ('Aug', 0.91), ('Sep',
0.57), ('Oct', 0.61),
             ('Nov', 0.57), ('Dec', 0.62)]
datalist2 = [('Jan', 0.62), ('Feb', 0.60), ('Mar', 0.46), ('Apr',
0.47), ('May', 0.95),
             ('Jun', 1.07), ('Jul', 1.3), ('Aug', 1.07), ('Sep',
0.99), ('Oct', 0.45),
             ('Nov', 0.53), ('Dec', 0.53)]
datalist3 = [('Jan', 0.65), ('Feb', 0.68), ('Mar', 0.80), ('Apr',
0.71), ('May', 1.11),
             ('Jun', 1.70), ('Jul', 2.05), ('Aug', 1.82), ('Sep',
0.97), ('Oct', 0.70),
             ('Nov', 0.60), ('Dec', 0.60)]
"""

# datalists for Home C
datalist1 = [('Jan', 1.09), ('Feb', 1.25), ('Mar', 0.93), ('Apr',
0.59), ('May', 0.51),
             ('Jun', 0.59), ('Jul', 0.61), ('Aug', 0.51), ('Sep',
0.63), ('Oct', 0.66),
             ('Nov', 0.81), ('Dec', 1.02)]
datalist2 = [('Jan', 1.29), ('Feb', 1.27), ('Mar', 0.91), ('Apr',
0.59), ('May', 0.57),
             ('Jun', 0.61), ('Jul', 0.79), ('Aug', 0.70), ('Sep',
0.61), ('Oct', 0.67),
             ('Nov', 0.71), ('Dec', 0.87)]
datalist3 = [('Jan', 1.06), ('Feb', 0.99), ('Mar', 0.75), ('Apr',
0.68), ('May', 0.56),
             ('Jun', 0.53), ('Jul', 1.13), ('Aug', 1.36), ('Sep',
0.73), ('Oct', 0.68),
             ('Nov', 0.83), ('Dec', 0.94)]

"""
# datalists for Home F
datalist1 = [('Jan', 1.42), ('Feb', 1.36), ('Mar', 1.20), ('Apr',
1.38), ('May', 1.38),
             ('Jun', 1.55), ('Jul', 1.33), ('Aug', 1.35), ('Sep',
1.27), ('Oct', 1.07),
             ('Nov', 1.25), ('Dec', 1.40)]
datalist2 = [('Jan', 1.34), ('Feb', 1.41), ('Mar', 1.24), ('Apr',
1.23), ('May', 1.10),
             ('Jun', 1.14), ('Jul', 1.56), ('Aug', 1.55), ('Sep',
1.19), ('Oct', 1.15),
             ('Nov', 1.20), ('Dec', 1.27)]
datalist3 = [('Jan', 1.50), ('Feb', 1.40), ('Mar', 1.38), ('Apr',
1.24), ('May', 1.37),
             ('Jun', 1.81), ('Jul', 1.78), ('Aug', 1.92), ('Sep',
1.79), ('Oct', 1.66),
             ('Nov', 1.36), ('Dec', 1.45)]
"""

months, y = zip(*datalist1)
plt.plot(range(len(months)), y, '-g', lw=2)
for i, j in enumerate(y):
    plt.annotate(str(j), xy=(i, j))
months, y = zip(*datalist2)
```

```
plt.plot(range(len(months)), y, '-r', lw=2)
for i, j in enumerate(y):
    plt.annotate(str(j), xy=(i, j))
months, y = zip(*datalist3)
plt.plot(range(len(months)), y, '-b', lw=2)
for i, j in enumerate(y):
    plt.annotate(str(j), xy=(i, j))
plt.title('Average Monthly Consumption Home C')
plt.xlabel('Months')
plt.ylabel('Consumption [kW]')
plt.legend(["Green line 2014", 'Red line 2015', 'Blue line 2016'])
axes = plt.gca()  # Get the Current Axes
axes.get_yaxis().get_major_formatter().set_scientific(False)
axes.set_xticks(range(len(months)))
axes.set_xticklabels(months, rotation=45, ha="right")
plt.show()

# Figure for average hourly consumption
plt.plot(dataframe['DateTime'], Hour14['Use'], '-g')
plt.plot(dataframe['DateTime'], Hour15['Use'], '-r')
plt.plot(dataframe['DateTime'], Hour16['Use'], '-b')
plt.title('Average Hourly Consumption Home C')
plt.xlabel('Hourly Range')
plt.ylabel('Consumption [kW]')
plt.legend(["Green line 2014", 'Red line 2015', 'Blue line 2016'])
axes = plt.gca() # Get the Current Axes
axes.get_yaxis().get_major_formatter().set_scientific(False)
axes.set_xticks(range(len(dataframe['DateTime'])))
axes.set_xticklabels(dataframe['DateTime'], rotation=80, ha="right")
plt.show()
```

Heat-maps

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns

"""
df1 = pd.read_csv('HomeB-meter1.csv')
# drops for Home B-1
df1 = df1.drop('Generated', 1)
df1 = df1.drop('Grid', 1)
df1 = df1.drop('DateTime', 1)
print('-----')
df2 = pd.read_csv('HomeB-meter2.csv')
# drops for Home B-2
df2 = df2.drop('Generated', 1)
df2 = df2.drop('Old usage', 1)
df2 = df2.drop('DateTime', 1)
df2 = df2.drop('Use', 1)

df = pd.concat([df1, df2], axis=1)
df = df.drop('Home Office (R)', 1)
df = df.drop('Fridge (R)', 1)
df = df.drop('Dining room (R)', 1)
df = df.drop('Microwave (R)', 1)
print(df.shape)
"""
"""
```

```
# Home C sensors
df = pd.read_csv('HomeC_sensors.csv')
print(df.shape)
print(df.tail())
df = df.drop('DateTime', 1)
df = df.drop('House overall', 1)
df = df.drop('Generated', 1)
"""
# Home F sensors
df = pd.read_csv('HomeF_sensors.csv')
print(df.shape)
df = df.drop('DateTime', 1)
df = df.drop('Generated', 1)
df = df.drop('WaterHeater1', 1)
df = df.drop('WaterHeater2', 1)
df = df.drop('WaterHeater3', 1)
df = df.drop('Phase_A', 1)
df = df.drop('Phase_B', 1)

"""
print(df.corr())
plt.matshow(df.corr())
plt.show()

Y = df['use [kW]']
X = df.drop('use [kW]', 1)

#Y.reset_index()
X['Use'] = Y.values
X.reset_index()
X = X.drop(['index'], axis=1)
print(X.tail())
print(X.describe().loc[['min', 'max', 'std', 'mean'], 'Use'])

X.columns = ['Home office', 'Dining room', 'Microwave', '1st floor',
'Tub whirpool', 'Kitchen counter', 'Dishwasher',
             'Fridge', 'Guest Bedroom/Media Room', 'MBed-KBed lights &
MasterBath', 'Living room & Kitchen lights',
              'Bath GFI 1st-2nd', 'Use']
"""

corr = df.corr()
mask = np.zeros_like(corr, dtype=np.bool)
mask[np.triu_indices_from(mask)] = True
f, ax = plt.subplots(figsize=(40, 25), tight_layout=True)
cmap = sns.diverging_palette(220, 10, as_cmap=True)
sns.heatmap(corr, mask=mask, cmap=cmap, vmax=1, vmin=-1, center=0,
square=True, linewidths=.5, cbar_kws={"shrink": .5})
ax.axes.xaxis.set_ticklabels([])
plt.yticks(rotation=0)
plt.title('Home F')
plt.show()

print(df.corr()['Use'].sort_values(ascending=False))
```

Top sensors regression

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.model_selection import GridSearchCV
from sklearn.ensemble import RandomForestRegressor
from sklearn.ensemble import GradientBoostingRegressor
from math import sqrt

"""
df1 = pd.read_csv('HomeB-meter1.csv')
# drops for Home B-1
df1 = df1.drop('Generated', 1)
df1 = df1.drop('Grid', 1)
df1 = df1.drop('DateTime', 1)
print('-----')
df2 = pd.read_csv('HomeB-meter2.csv')
# drops for Home B-2
df2 = df2.drop('Generated', 1)
df2 = df2.drop('Old usage', 1)
df2 = df2.drop('DateTime', 1)
df2 = df2.drop('Use', 1)

df = pd.concat([df1, df2], axis=1)
df = df.drop('Home Office (R)', 1)
df = df.drop('Fridge (R)', 1)
df = df.drop('Dining room (R)', 1)
df = df.drop('Microwave (R)', 1)

# df = df[['Use', 'AC', 'Furnace', 'Utility Rm + Basement Bath', 'Dry-
er + egauge', 'Home office', 'Dining room']]
# df = df[['Use', 'AC', 'Furnace', 'Utility Rm + Basement Bath']]

"""

"""
# Home C
df = pd.read_csv('HomeC_sensors.csv')
df = df.drop('DateTime', 1)
df = df.drop('Generated', 1)
df = df.drop('House overall', 1)
df = df.drop('Kitchen 38', 1)

# print(df.tail())
# df = df[['Use', 'Furnace 2', 'Furnace 1', 'Living room', 'Dishwash-
er', 'Barn', 'Well']]
# df = df[['Use', 'Furnace 2', 'Furnace 1', 'Living room']]
"""

# Home F
df = pd.read_csv('HomeF_sensors.csv')
df = df.drop('DateTime', 1)
df = df.drop('Generated', 1)
df = df.drop('WaterHeater1', 1)
df = df.drop('WaterHeater2', 1)
df = df.drop('WaterHeater3', 1)
df = df.drop('Phase_A', 1)
df = df.drop('Phase_B', 1)
```

```
# df = df[['Use', 'Water Heater', 'Family Room', 'Furnace', 'Dryer',
'Half-Bath Foyer', 'Dishwasher Disposal']]
# df = df[['Use', 'Water Heater', 'Family Room', 'Furnace']]

print(df.tail())
Y = df['Use']
X = df.drop('Use', 1)

X_train, X_test, y_train, y_test = train_test_split(X, Y,
test_size=0.33, random_state=42)

# Regression = GradientBoostingRegressor()
Regression = RandomForestRegressor()
# n_estimators = [10, 20, 30,  40, 50]
# max_depth = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
# param_grid = {'n_estimators': n_estimators, 'max_depth': max_depth}

# REG = GridSearchCV(Regression, param_grid={}, cv=10, verbose=10,
scoring='r2')
REG = GridSearchCV(Regression, param_grid={}, cv=10, verbose=10, scor-
ing='neg_mean_squared_error')
REG.fit(X_train, y_train)

print('RMSE =', sqrt(abs(REG.cv_results_['mean_test_score'])))
# print('R-squared =', REG.cv_results_['mean_test_score'])
# print('Adjusted R-squared =', 1-(1-
REG.cv_results_['mean_test_score']*(len(y_test)-1)/(len(y_test)-
X_test.shape[1]-1)))
```

Weather Pre-processing

```
import numpy as np
import pandas as pd
from datetime import datetime

# import dataset into a pandas dataframe
weather1 = pd.read_csv('Weather_HF_2014.csv')
weather2 = pd.read_csv('Weather_HF_2015.csv')
weather3 = pd.read_csv('Weather_HF_2016.csv')

weather = pd.concat([weather1, weather2, weather3], axis=0)
weather = weather[['time', 'temperature', 'humidity', 'apparentTemper-
ature', 'windSpeed', 'windBearing', 'summary',
'dewPoint']]

print(weather.isnull().sum())
print(weather.shape)
weather['time'] = pd.to_datetime(weather['time'])
# keeping only values between 15:00 - 21:00. *7 values in total each
day
# weather = weather[(weather['time'].dt.hour <= 21) & (weath-
er['time'].dt.hour >= 15)]
weather = weather[(weather['time'].dt.hour <= 15) & (weath-
er['time'].dt.hour >= 9)]

print(weather.shape)

# weather.to_csv('Weather_F_ON.csv', index=False)
# weather.to_csv('Weather_F_OFF.csv', index=False)
```

Weather Aggregation

```
import numpy as np
import pandas as pd
from datetime import datetime

# weather = pd.read_csv('Weather_F_ON.csv')
weather = pd.read_csv('Weather_F_OFF.csv')

summary_values = weather['summary'].tolist()
chunks = [summary_values[x:x+7] for x in range(0, len(summary_values),
7)]
max_chunks = []
for item in chunks:
    max_chunks.append(max(item, key=item.count))

# reseting the index
# weather = weather.reset_index()

temperature_values = weather['temperature'].values
temperature_mean = np.mean(temperature_values.reshape(-1, 7), axis=1)

humidity_values = weather['humidity'].values
humidity_mean = np.mean(humidity_values.reshape(-1, 7), axis=1)

apparentTemperature_values = weather['apparentTemperature'].values
apparentTemperature_mean =
np.mean(apparentTemperature_values.reshape(-1, 7), axis=1)

windSpeed_values = weather['windSpeed'].values
windSpeed_mean = np.mean(windSpeed_values.reshape(-1, 7), axis=1)

windBearing_values = weather['windBearing'].values
windBearing_mean = np.mean(windBearing_values.reshape(-1, 7), axis=1)

dewPoint_values = weather['dewPoint'].values
dewPoint_mean = np.mean(dewPoint_values.reshape(-1, 7), axis=1)

#Sunset creation
sunset_df = pd.read_excel('Sunset.xlsx')
sunset = sunset_df['Sunset'].values
#sunset = sunset[:-16]
print(len(sunset))


FinalWeather = pd.DataFrame({'temperature': temperature_mean, 'humidi-
ty': humidity_mean,
                             'apparentTemperature': apparentTempera-
ture_mean, 'windSpeed': windSpeed_mean,
                             'windBearing': windBearing_mean, 'sta-
tus': max_chunks, 'dewPoint': dewPoint_mean,
                             'sunset': sunset})

FinalWeather = FinalWeather[['temperature', 'apparentTemperature',
'humidity', 'windSpeed', 'windBearing', 'status',
                             'dewPoint', 'sunset']]
# FinalWeather = FinalWeather[:-15]

print(FinalWeather.isnull().sum())
print(FinalWeather.shape)
print(FinalWeather.tail(10))
```

```
# FinalWeather.to_csv('Final_Weather_F_ON.csv', index=False)
# FinalWeather.to_csv('Final_Weather_F_OFF.csv', index=False)

# FinalWeather.to_csv('WeatherAgg_F_ON.csv', index=False)
# FinalWeather.to_csv('WeatherAgg_F_OFF.csv', index=False)
```

Date Usage – peak periods

```
import pandas as pd
from datetime import datetime

# importing dataset into a pandas dataframe
dataframe1 = pd.read_csv('HomeB_2014_pp.csv')
dataframe2 = pd.read_csv('HomeB_2015_pp.csv')
dataframe3 = pd.read_csv('HomeB_2016_pp.csv')
# merging the datasets
dataframe = pd.concat([dataframe1, dataframe2, dataframe3], axis=0)

print(dataframe.shape)

# keeping only the two columns of interest
dataframe = dataframe[['DateTime', 'use [kW]']]

print(dataframe.shape)

# turning the 'DateTime' column into datetime object so as to be more
easily manipulated
dataframe['DateTime'] = pd.to_datetime(dataframe['DateTime'])

# keeping only values between 15:00 - 20:30. *12 values in total each
day
# dataframe = dataframe[(dataframe['DateTime'].dt.hour <= 21) & (data-
frame['DateTime'].dt.hour >= 15)]
dataframe = dataframe[(dataframe['DateTime'].dt.hour <= 15) & (data-
frame['DateTime'].dt.hour >= 9)]
print(dataframe.shape)
print(dataframe.head(15))

# dataframe.to_csv('DateUsage_B_ON.csv', index=False)
dataframe.to_csv('DateUsage_B_OFF.csv', index=False)
```

Usage Aggregation

```
import numpy as np
import pandas as pd
from math import sqrt
from datetime import datetime
import matplotlib.pyplot as plt
from pandas.tseries.holiday import USFederalHolidayCalendar as calen-
dar

dataframe = pd.read_csv('DateUsage_F_OFF.csv')

# creating an array with consumption values
consumption = dataframe['use [kW]'].values
```

```
consumption_rescaled = np.mean(consumption.reshape(-1, 12), axis=1)
# dataframe['DateTime'] = dataframe['DateTime'].dt.weekday_name
print(len(consumption_rescaled))


Days_year = ['Wednesday', 'Thursday', 'Friday', 'Saturday', 'Sunday',
'Monday', 'Tuesday']
Days_final = Days_year * 157
del Days_final[-3:]
print(len(Days_final))

datevalues = dataframe['DateTime'].values
datevalues_rescaled = datevalues[::12]

# Creating a new dataframe with the same name 'dataframe' containing
the 2 arrays above.
dataframe = pd.DataFrame({'DateTime': Days_final, 'Use': consump-
tion_rescaled})

# Creating a new column 'Weekends' that contains 1 if it's Saturday or
Sunday, else 0.
dataframe['Weekends'] = np.where((dataframe['DateTime'] == 'Saturday')
| (dataframe['DateTime'] == 'Sunday'), 1, 0)

# Creating a new column 'Holidays' that contains 1 if it's a legal
holiday, else 0.
dr = pd.date_range(start='2014-01-01', end='2016-12-31')
dataframe['Date'] = dr
cal = calendar()
holidays = cal.holidays(start=dr.min(), end=dr.max())
dataframe['Holidays'] = dataframe['Date'].isin(holidays)
dataframe['Holidays'] = [1 if x is True else 0 for x in data-
frame['Holidays']]

print(dataframe.tail(20))

# However, the built in Calendar does not contain Patriot's day -
Third Monday of April so it was applied manually.
dataframe.at[110, 'Holidays'] = 1
dataframe.at[474, 'Holidays'] = 1
dataframe.at[838, 'Holidays'] = 1

# dataframe.to_csv('UseAgg_F_ON.csv', index=False)
dataframe.to_csv('UseAgg_F_OFF.csv', index=False)
```

Combine weather – consumption

```
import numpy as np
import pandas as pd
from math import sqrt
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler

# Use_data = pd.read_csv('UseAgg_F_ON.csv')
Use_data = pd.read_csv('UseAgg_F_OFF.csv')

# Weather_data = pd.read_csv('WeatherAgg_F_ON.csv')
Weather_data = pd.read_csv('WeatherAgg_F_OFF.csv')
```

```
Final_df = pd.concat([Use_data, Weather_data], axis=1)

del Final_df['DateTime']
del Final_df['Date']

print(Final_df.isnull().values.any())

# wind bearing to categories
BearingList = Final_df['windBearing'].tolist()
Final_df = Final_df.drop('windBearing', 1)
EmptyList = []
for i, elem in enumerate(BearingList):
    if elem < 45.0 or elem > 315.0:
        EmptyList.append('North')
    elif elem < 135.0:
        EmptyList.append('East')
    elif elem < 225.0:
        EmptyList.append('South')
    else:
        EmptyList.append('West')

dfentry = pd.Series(EmptyList)
Final_df['windBearing'] = dfentry.values

# dummies for wind bearing categories
Final_df1 = pd.get_dummies(Final_df['windBearing'])
Final_df = pd.concat([Final_df, Final_df1], axis=1)
del Final_df['windBearing']

# dummies for combined weather
df_status = pd.get_dummies(Final_df['status'])
Final_df = pd.concat([Final_df, df_status], axis=1)
del Final_df['status']

# Creation of last day use column
Previous_use = Final_df['Use'].tolist()
cut = Final_df['Use'].values.mean()
print('Mean value of consumption =', cut)
Yest_Use = [cut]
for i in range(0, len(Final_df['Use'])):
    Yest_Use.append(Previous_use[i])
del Yest_Use[-1]
# Final_df = Final_df.drop('Use', 1)
Final_df['Yesterday'] = Yest_Use

# Sunset into categories
sunset list = Final df['sunset'].tolist()
Final_df = Final_df.drop('sunset', 1)
empty_sunset = []
for i, elem in enumerate(sunset_list):
    if elem < 1700:
        empty_sunset.append('4-hour')
    elif elem < 1800:
        empty_sunset.append('5-hour')
    elif elem < 1900:
        empty_sunset.append('6-hour')
    elif elem < 2000:
        empty_sunset.append('7-hour')
    else:
        empty_sunset.append('8-hour')
```

```
dfentry = pd.Series(empty_sunset)
Final_df['sunset'] = dfentry.values

df_sunset = pd.get_dummies(Final_df['sunset'])
Final_df = pd.concat([Final_df, df_sunset], axis=1)
del Final_df['sunset']

consumption_list = Final_df['Use'].tolist()
#Final_df = Final_df.drop('Use', 1)
empty_list = []
for i, elem in enumerate(consumption_list):
    if elem < cut:
        empty_list.append('Low')
    else:
        empty_list.append('High')

dfentry = pd.Series(empty_list)
Final_df['UseCategorized'] = dfentry.values
print(Final_df['UseCategorized'].value_counts())

cols_at_end = ['UseCategorized']
Final_df = Final_df[[c for c in Final_df if c not in cols_at_end] + [c
for c in cols_at_end if c in Final_df]]

print(Final_df.shape)
print(Final_df.tail(5))


# Final_df.to_csv('FINAL_B_ON.csv', index=False)
# Final_df.to_csv('FINAL_B_OFF.csv', index=False)

# Final_df.to_csv('FINAL_F_ON_use.csv', index=False)
Final_df.to_csv('FINAL_F_OFF_use.csv', index=False)
```

Classification on finals and parameters plotting

```
import pandas as pd
import numpy as np
from sklearn.ensemble import RandomForestRegressor
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import GridSearchCV
from sklearn.svm import SVC
from sklearn import linear_model
from sklearn.model_selection import train_test_split
from math import sqrt
import matplotlib.pyplot as plt
from sklearn import linear_model
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report
from sklearn.ensemble import GradientBoostingClassifier

#df = pd.read_csv('Complete_HomeF_OFF.csv')
df = pd.read_csv('Merging-OFF.csv')

# Consumption per general mean value
del df['Use']
del df['Month']
```

```
#del df['Season']
del df['ConsPerMonth']
del df['ConsPerSeason']
# del df['ConsPerTotal']

# Split data into training and testing set
Y = df['ConsPerTotal']
X = df.drop('ConsPerTotal', 1)
X_train, X_test, y_train, y_test = train_test_split(X, Y,
test_size=0.33, random_state=42)

# Scaling the data
#scaler = StandardScaler()
#scaler.fit(X_train)
#X_train = scaler.transform(X_train)
#X_test = scaler.transform(X_test)

#Classification = RandomForestClassifier(random_state=42,
min_samples_leaf=6, min_samples_split=2)
#Classification = RandomForestClassifier(random_state=42)
#n_estimators = [5, 10, 30, 50, 100, 150, 200, 250]
#min_samples_split = [2, 3, 4, 5]
#min_samples_leaf = [2, 3, 4, 5, 6]
#max_features = ['auto', 'sqrt', 'log2']
#param_grid = {'n_estimators': n_estimators, 'max_features':
max_features, 'min_samples_split': min_samples_leaf,
#               'min_samples_leaf': min_samples_leaf}
#param_grid = {'n_estimators': n_estimators, 'max_features':
max_features}

#Classification = linear_model.SGDClassifier(random_state=42)
#alpha = [0.001, 0.01, 0.1, 0.2, 0.5, 1, 2]
#n_iter = [100, 500, 1000, 2000]
#loss = ['hinge', 'squared_hinge', 'perceptron']
# learning_rate = ['optimal']
# param_grid = {'alpha': alpha, 'n_iter': n_iter, 'loss': loss}
#param_grid = {'alpha': alpha, 'n_iter': n_iter}

#Classification = SVC(random_state=42)
#C = [0.001, 0.01, 0.1, 0.5, 1, 2]
#kernel = ['poly', 'sigmoid', 'linear', 'rbf']
#gamma = [0.001, 0.01, 0.1, 0.5, 1.0, 1.5]
#param_grid = {'C': C, 'gamma': gamma, 'kernel': kernel}

Classification = LogisticRegression(penalty='l1', random_state=42)
max_iter = [10, 50, 100, 150]
C = [0.01, 0.1, 0.5, 1, 2, 3]
# penalty = ['l1', 'l2']
param_grid = {'max_iter': max_iter, 'C': C}


CLF = GridSearchCV(Classification, param_grid=param_grid, cv=10, ver-
bose=10)
CLF.fit(X_train, y_train)
print('Best parameters: ', CLF.best_params_)
print('Average score: ', CLF.best_score_)
print(CLF.cv_results_['mean_test_score'])

y_true, y_pred = y_test, CLF.predict(X_test)
print(classification_report(y_true, y_pred))
```

```
# plt.plot(combined_weather['Date'], Y, '-g', lw=2)
# plt.show()

def plot_grid_search(cv_results, grid_param_1, grid_param_2,
name_param_1, name_param_2):
    # Get Test Scores Mean and std for each grid search
    scores mean = cv results['mean test score']
    scores_mean = np.array(scores_mean).reshape(len(grid_param_2),
len(grid_param_1))

    scores_sd = cv_results['std_test_score']
    scores_sd = np.array(scores_sd).reshape(len(grid_param_2),
len(grid_param_1))

    # Plot Grid search scores
    _, ax = plt.subplots(1, 1, tight_layout=True)

    # Param1 is the X-axis, Param 2 is represented as a different
curve (color line)
    for idx, val in enumerate(grid_param_2):
        ax.plot(grid_param_1, scores_mean[idx, :], '-o', la-
bel=name_param_2 + ': ' + str(val))

    ax.set_title("Home F - OFF-peak RF", fontsize=20, font-
weight='bold')
    ax.set_xlabel(name_param_1, fontsize=16)
    ax.set_ylabel('CV Average Score', fontsize=16)
    ax.legend(loc="best", fontsize=15)
    ax.grid('on')
    plt.show()


# Calling Method
#plot_grid_search(CLF.cv_results_, n_estimators, max_features, 'Esti-
mators', 'Max Features')
```

No aggregation part

```
import numpy as np
import pandas as pd
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import GridSearchCV
from sklearn.svm import SVC
from sklearn import linear_model
import matplotlib.pyplot as plt
from sklearn import linear_model
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import classification_report
from sklearn.model_selection import TimeSeriesSplit
from sklearn.preprocessing import MinMaxScaler
from sklearn.linear_model import LogisticRegression
from xgboost import XGBClassifier
from sklearn.model_selection import train_test_split

"""
date_use = pd.read_csv('DateUsage_C_ON.csv')
print(date_use.shape)
```

```
date_use = date_use.iloc[::2]
date_use = date_use.reset_index(drop=True)
print(date_use.shape)
weather = pd.read_csv('Weather_C_ON.csv')
weather['Use'] = date_use['use [kW]'].astype(float)

data = weather

BearingList = data['windBearing'].tolist()
data = data.drop('windBearing', 1)
EmptyList = []
for i, elem in enumerate(BearingList):
    if elem < 45.0 or elem > 315.0:
        EmptyList.append('North')
    elif elem < 135.0:
        EmptyList.append('East')
    elif elem < 225.0:
        EmptyList.append('South')
    else:
        EmptyList.append('West')

dfentry = pd.Series(EmptyList)
data['windBearing'] = dfentry.values

print(data.head())
print(data.shape)

# data.to_csv('noaggregation_B_OFF.csv', index=False)
"""


df = pd.read_csv('noaggregation_B_OFF.csv')

del df['summary']
del df['windBearing']
# del df['Use']

df = df.drop_duplicates(subset='time')
df['time'] = pd.to_datetime(df['time'])

print(df.groupby('Hour')['Use'].mean())

emptylist1 = []
for index, row in df.iterrows():
    if row['Hour'] == '09:00':
        if row['Use'] > 0.663451:
            emptylist1.append('High')
        else:
            emptylist1.append('Low')
    elif row['Hour'] == '10:00':
        if row['Use'] > 0.741325:
            emptylist1.append('High')
        else:
            emptylist1.append('Low')
    elif row['Hour'] == '11:00':
        if row['Use'] > 0.797954:
            emptylist1.append('High')
        else:
            emptylist1.append('Low')
    elif row['Hour'] == '12:00':
        if row['Use'] > 0.886245:
```

```
            emptylist1.append('High')
        else:
            emptylist1.append('Low')
    elif row['Hour'] == '13:00':
        if row['Use'] > 0.906011:
            emptylist1.append('High')
        else:
            emptylist1.append('Low')
    elif row['Hour'] == '14:00':
        if row['Use'] > 0.946573:
            emptylist1.append('High')
        else:
            emptylist1.append('Low')
    elif row['Hour'] == '15:00':
        if row['Use'] > 0.979634:
            emptylist1.append('High')
        else:
            emptylist1.append('Low')
    else:
        print(row['Use'])

dfentry = pd.Series(emptylist1)
print(len(dfentry))
print(dfentry)
print(df.shape)
df['Target'] = dfentry.values

# On-peak
#df = df[df['time'].dt.hour >= 16]
# Off-peak
df = df[df['time'].dt.hour >= 10]
print(df.head(10))
# del df['time']
#df.to_csv('NoAgg-B-OFF.csv', index=False)

del df['ConsPerTotal']
del df['Use']

df1 = df.iloc[:4381]
df2 = df.iloc[4381:]
print(df1.shape)
print(df2.shape)
del df1['time']
del df1['Hour']
del df2['time']
del df2['Hour']

Y_train = df1['Target']
Y_test = df2['Target']
X_train = df1.drop('Target', 1)
X_test = df2.drop('Target', 1)

scaler = MinMaxScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

CLF = LogisticRegression()
CLF.fit(X_train, Y_train)
print(CLF.score(X_test, Y_test))
y_true, y_pred = Y_test, CLF.predict(X_test)
print(classification_report(y_true, y_pred))
```