INTERNATIONAL
HELLENIC
UNIVERSITY

Data Mining for Smart Cities: Energy Disaggregation and Recommendation System

Iordanis Tourpeslis

SID: 3301200004

SCHOOL OF SCIENCE & TECHNOLOGY

A thesis submitted for the degree of

Master of Science (MSc) in Information and Communication Systems

JANUARY 2022

THESSALONIKI – GREECE

INTERNATIONAL
HELLENIC
UNIVERSITY

Data Mining for Smart Cities: Energy Disaggregation and Recommendation System

Iordanis Tourpeslis

SID: 3301200004

Supervisor:	Dr. Christos Tjortjis
Supervising Committee Members:	Assoc. Prof. A. Papadopoulos
	Assist. Dr S. Stavrinidis

SCHOOL OF SCIENCE & TECHNOLOGY

A thesis submitted for the degree of

Master of Science (MSc) in Information and Communication Systems

JANUARY 2022

THESSALONIKI – GREECE

Abstract

One major factor that can influence sustainable living of people is the understanding of energy data consumption to achieve behavioral change which in further will lead to minimization of energy expenses as well as decreasing the waste footprint to the climate. Energy disaggregation tries to enhance the process of leaning the energy behavior of the user by categorizing the total consumption of a household into appliance level consumption. This study aims to develop an energy disaggregation system that will produce recommendations on which actions could be taken to minimize the energy consumption of the household based on the final output of the disaggregation process.

Acknowledgements

I want to thank deeply my supervisor Dr. Christos Tjortjis for his help and guidance through thought the dissertation process and because he gave me the opportunity to acquire a scientific approach of the problem that my thesis deals with.

Contents

1 Introduction

Unconscious energy consumption contributes to the rise of greenhouse gas emissions forcing to be in the focus of attention of a great number of scientific papers and organizations activities, that are pointing out the consequences that we will face in the near future with the climate as the central recipient. These consequences will create irreversible damage resulting in the formation of unsustainable environments.

Climate change is an international problem that all countries must address by developing and implementing national legislations that specifically will tackle the vital problem of global warming.

European Union (EU) has funded numerous research initiatives in order to investigate possible causes, discovering that building sector is accountable for approximately 40% for energy consumption and 36% for carbon dioxide emissions within EU [1] territory making buildings the largest energy consumer sector which result in high potential of energy saving capabilities reducing the overall energy consumption. The outcome has triggered the creation of legislations and standards in national but also in international level with respect to energy consumption minimization. Specific measures introduced as a means to enhance the performance of the buildings from the perspective of the envelope and technological systems in new constructions and existing building stock. Such measures are the Energy Performance of Building Directive [2] and Energy Efficiency Directive [3] which has been presented by EU between 2010 and 2012.

Furthermore, EU to reinforce the process of energy consumption reduction published the Energy End-Use Efficiency and Energy Services Directive [4] denoting that all member states should make mandatory to all energy providers to incorporate within the electricity bill different types of feedback, as an example detailed analyses of overall energy consumption together with appliance level consumption, frequent billing, current and historic data comparisons and further information that will be used to extract new knowledge and rase the awareness of end user with respect opportunities that might exist to reduce their energy consumption, educate them by making energy consumption data available and understandable through the application of user friendly visualization tools.

The effectiveness of feedback has been under investigation [5], specifically the energy savings that can be made by providing end-users recommendations based upon their energy consumption data. In boarder terms the effectiveness of the feedback does not really only in the contents of the information but also on the frequency, communication medium, the method used to present information, detailed and understandable breakdown in order to give the end-user a deep understanding of their energy consumption and which appliances influence their household expenses. Additionally, the feed-back can be enhanced through the usage of different comparison measures. On the whole, every type of feedback will have a positive impact nevertheless the more detailed is the information the higher the impact will be. The adoption of feedback on a great period of time will not only help end-users to extract new knowledge but also help them to assimilate it with their habits [6] forming a revolutionary culture that will be sensitized on environmental issues equipped with the appropriate supplies in order to contribute to the minimization of energy consumption and from the economical perspective to reduce the expenses of their households.

Instant feedback is the most suitable method to interact and share information with the end-users, the most appropriate ecosystem that can support it is the employment of smart cities. Smart City concept [7] can be described as a digital platform which consists of smart electronic devices that are spread to different locations in a city exploiting different resources with purpose to provide improved living conditions, more desirable management of existing resources and services. Smart city concepts can be facilitated by copious resources that can be found within an urban environment (Figure 1).

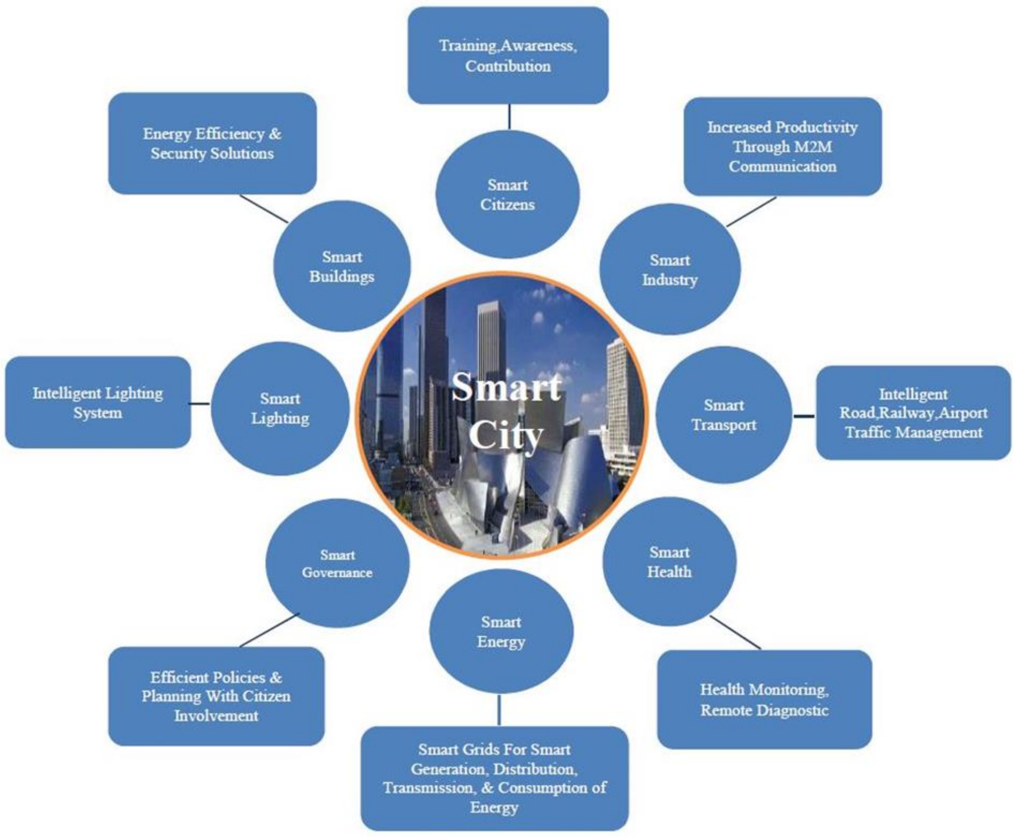

Training,Awareness, Contribution
Energy Efficiency & Security Solutions
Increased Productivity Through M2M Communication
Smart Citizens
Smart Buildings
Smart Industry
Intelligent Lighting System
Smart Lighting
Smart City
Smart Transport
Intelligent Road,Railway,Airport Traffic Management
Smart Governance
Smart Health
Smart Energy
Efficient Policies & Planning With Citizen Involvement
Health Monitoring, Remote Diagnostic
Smart Grids For Smart Generation, Distribution, Transmission, & Consumption of Energy

Figure 1: Smart city applications [1]

The widespread employment of smart city technology has increased the inflow of data, creating enormous amount of data in very small period of time. To take advantage of the incoming information different data mining techniques has been implemented in order to extract knowledge that will add value to the data.

To take advantage of data mining capabilities, numerous industries have incorporated them into their business processes, some of the industries are:

- Financial Industry
- Retail Industry
- Telecommunication Industry
- Research Industry
- Security Industry
- Marketing Industry
- Sales Industry

Energy industry has developed various smart devices and systems that empower end-users to monitor their energy consumption dynamically meaning that they can oversee at any time the total consumption of their household. Smart meters are electronic devices that are installed into the main electrical panel of a household and record the total energy consumption.

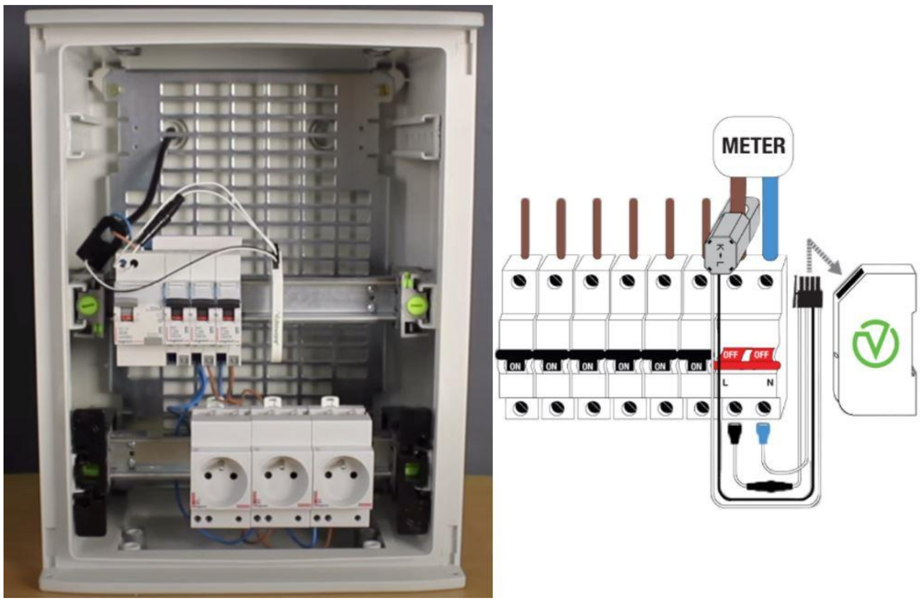


Figure 2: Installation example of a typical smart meter [2]

Because, as described previously the presentation of total consumption is not considered as an energy minimization measure due to the fact that it does not provide detailed break-down of information in order to assist end-user to understand how their energy consumption is consumed by their actions, meaning detailed feedback containing the consumption of each individual appliance that is used within the household. To overcome this problem, devices have been developed that can be attached into the electrical plugs (sockets) and then electrical devices can be connected directly upon these devices.

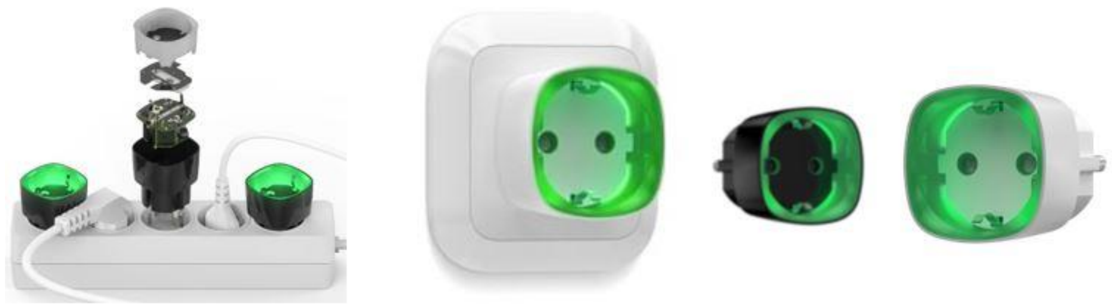


Figure 3: Example of a typical smart plug [3]

They are called smart plug and except from energy consumption monitoring, can provide automation capabilities by turning on and off remotely the appliance that is connected on it. The method of installing devices scattered within a household for the purpose of monitoring total energy consumption alongside the appliances consumption is called Intrusive Load Monitoring (ILM). This methos is high accurately because all the appliances are monitored leading to a detailed breakdown of the energy consumed in a household. A huge disadvantage of the method is the high capital investment that an end-user must do for all the devices, stable and high-speed wireless internet connection is required since all the devices will be connected into a single access point increasing the network traffic and as of their nature wireless connections are not considered as stable as wired connections. Another problem is the formation of tremendous amount of data within a small period of time and by considering that the method can be widely applied to every household common database system cannot support the scale of this kind of projects. Technology providers must invest into their infrastructure to upgrade their database systems if they want to accommodate successfully Intrusive Load Monitoring.

Taking into account the obstacles of Intrusive Load Monitoring the scientific community has turned their attention in more feasible solutions that will not require huge investment from both sides of technology providers and customers. In 1992 George W. Hart has published a method called Nonintrusive Appliance Load Monitoring (NALM) [11] which calculated the appliances energy consumption by using the readings of single energy meter that has been installed into the main circuit panel of a house. His approach uses statistical model and data mining methods in order to categorize the data and then identifies which group of data belongs to the corresponding appliances. NALM is also known as Nonintrusive Load Monitoring (NILM) and as Energy Disaggregation.

With passing of time several papers have been published introducing new methods or introducing improvements to existing methods correcting issues that they haven’t been addressed. The final output of an energy disaggregation process must be user friendly in order to be easily understandable by the end-user thus the total energy consumption side by side with the portion of energy consumed by the appliances exist with-in the household. Fig. 4 illustrates an example that a typical energy disaggregation method might contain.

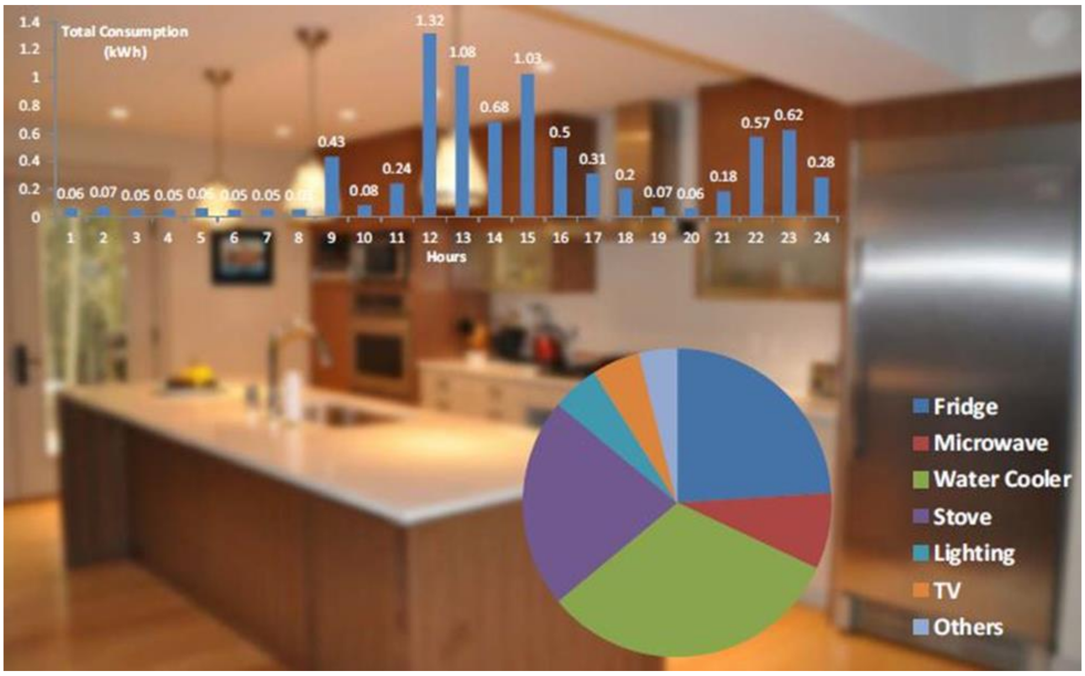


Figure 4: Example of an energy disaggregation output [4]

The enhancement of energy reduction process is not only depended on energy disaggregation considering that the final output of such a model is not sufficient for an end-user to conceive how to act in order to achieve lower energy consumption. The user must dedicate time to search and learn practices on how to lower their household energy consumption and then side by side with the disaggregation output to reduce the electricity expenses. It is very difficult for the end-user to handle it on its own this procedure, for that reason recommendation systems have been developed in order to foster the decision process of the end-user. Recommendation systems are not a new topic, they have been used for quite some time in diverse sectors, Amazon [13] and Netflix [14] use them to suggest products that the user may want to buy or for suggesting movies for the user to see based on their historical data and other users' preferences. In energy sector, recommendation systems can by implemented in combination with the disaggregation output to suggest end-user solutions on how to minimize the energy consumption.

This paper focus to construct an energy disaggregation model that will be based on past papers that have been published for the same topic and on top of that will accompanied by a simple recommendation system that will take into account the number and type of appliances used for energy disaggregation and create consultations on how to achieve energy savings helping the end-user to better understand their consumption and implement the proposed actions of recommendation system to reduce their expenses benefiting them economically but also the environment since lower consumption means lower carbon dioxide emissions.

The result of the disaggregation method developed in this paper can be considered highly accurate for different type of appliances that does not share the same electrical characteristics, with respect to recommendation system it also performed as expected providing suggestions with the highest score on specific appliances.

2 Literature Review

This chapter contains the presentation of existing knowledge for non-intrusive load monitoring methods that have been tested and the basic technologies used to construct a recommendation system.

2.1 Data Mining Methods

Data mining is a processing method that can take advantage of the large quantities of data by investigating them to extract any meaningful pattern that might exits or discover previously unknown information that potentially may appear valuable converting it into new knowledge. In more details, data mining is divided into two main methods prediction and description. As their names indicate prediction methods are used to predict unknown or missing values and description methods detect patterns that describe the dataset. Based on these two methods there have been developed 6 main data mining functions that have different capabilities, implementation procedures and output, Tables 1 and 2 contains sort description of them.

Table 1: Descriptive methods

Descriptive methods	
Clustering	Is the task of dividing data into separate group. Each data within the group is considered to be similar and data that belong to different groups are considered to be dissimilar.
Association Rule Discovery	Is a rule-based method that discovers relations between the data of a database or a dataset using some measures of interestingness.
Sequential Pattern Discovery	Discovers statistically relevant patterns in sequential data.

Table 2: Predictive methods

Predictive methods	
Classification	Is a function that allocates items in a collection of classes. The goal is to accurately predict the target class for each case in the data.
Regression	Is used to predict a range of unknown numeric values given historical data.
Deviation Detection	Is a technique that detects significant deviations from normal behavior.

Varius disaggregation models have been developed based on the principles of data mining methods leading to different performances.

2.2 Event Detection

Non-intrusive load monitoring procedures are composed by four main stages as the authors of the study [15] state. These stages are the power measurement where electrical characteristics (volts, ambers, active power, reactive power, etc.) of the main control panel from a household are collected. In some cases, the measurements of individual appliances might be recorded in order to enhance the overall procedure of a NILM algorithm. Event detection is translated as the spikes that are formed by the electric appliances when they are powered on and off. Different types of appliances form dissimilar events which then are extracted in order to be used during classification. During classifications step the model is trained to be able to distinguish/identify the type of appliance based on the events. The final step is the disaggregation of energy based on aggregated consumption data (identify the portion of energy that each of the appliances has consumed from the total consumption).

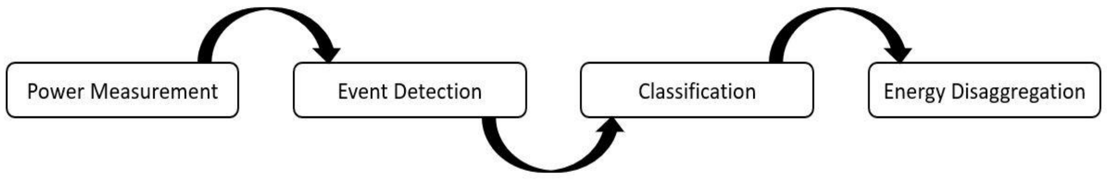


Figure 5: Diagram of NILM main components [5]

They presented four metrics so that event detection algorithms to be evaluated, namely the 1) True Positive Rate, 2) True Positive Percentage, 3) Total Power Change and 4) Average Power Change. Through investigation of literature, they concluded that event detection approaches can be categorized to three categories the Expert Heuristics, Probabilistic Models and Matched-Filters. To evaluate their proposed metrics, they implemented a modification of Generalized Likelihood Ratio (GLR) event detector which belongs to Probabilistic Model category. The GLR uses a log likelihood ratio test and voting. From the five detection parameters that the method uses, four of them was altered to create different combinations of parameters. They concluded that the most efficient evaluation metric is the total power change since it encompasses power characteristic to compute the evaluation.

A hybrid method consisting of a single base algorithm accompanied by two auxiliary algorithms was developed [6] in order to detect the events from aggregated data utilizing one of the electric parameters that can be measured by a smart meter. The base algorithm incorporates the detection of events by calculating the moving average values of the points within a specified time interval, then the subtraction of the absolute values of the points is compared to a threshold and if exceeds it then these points are listed as events. A disadvantage which must be addressed very carefully is the selection of the proper time interval due to the fact that a large time interval may miss the power on or off of an appliance when it occurs in small time interval. After implementing the base algorithm, they have found that alongside of true events detection there was identified many false alarms (events that the algorithms detected but in reality, they are not events but only fluctuations in the power measurement). To overcome it they used the time difference between two events and then they compare it to a specified time threshold, if the difference is below, the two events are considered as one event minimizing the number of false events. Even after the introduction of the time threshold their base algorithm detects false events, this is attributed in appliances that create high fluctuations (heavy consumption appliances) and in the duration change of their state (the duration of power on or off is greater from the time interval specified during the first stage of the algorithm). To eliminate the existence of false events they defined two auxiliary algorithms. The first one uses the calculation of first derivative, then a smoothing algorithm known as LOESS is implemented to reduce the noise created by the first derivative. Then, peaks are identified by comparing a point with their beside points, if the value is not smaller then it is considered as a positive peak, otherwise if the value is not larger then is considered as a negative peak. The last part of the first auxiliary algorithm is the reduction of false events by using derivative threshold and a time limit. By implementing the first auxiliary algorithm they have managed to tackle the false events that are produced by appliances that have long transitions when they change their state. The last part is appliances with high fluctuations, to address this problem the second auxiliary algorithm was introduced.

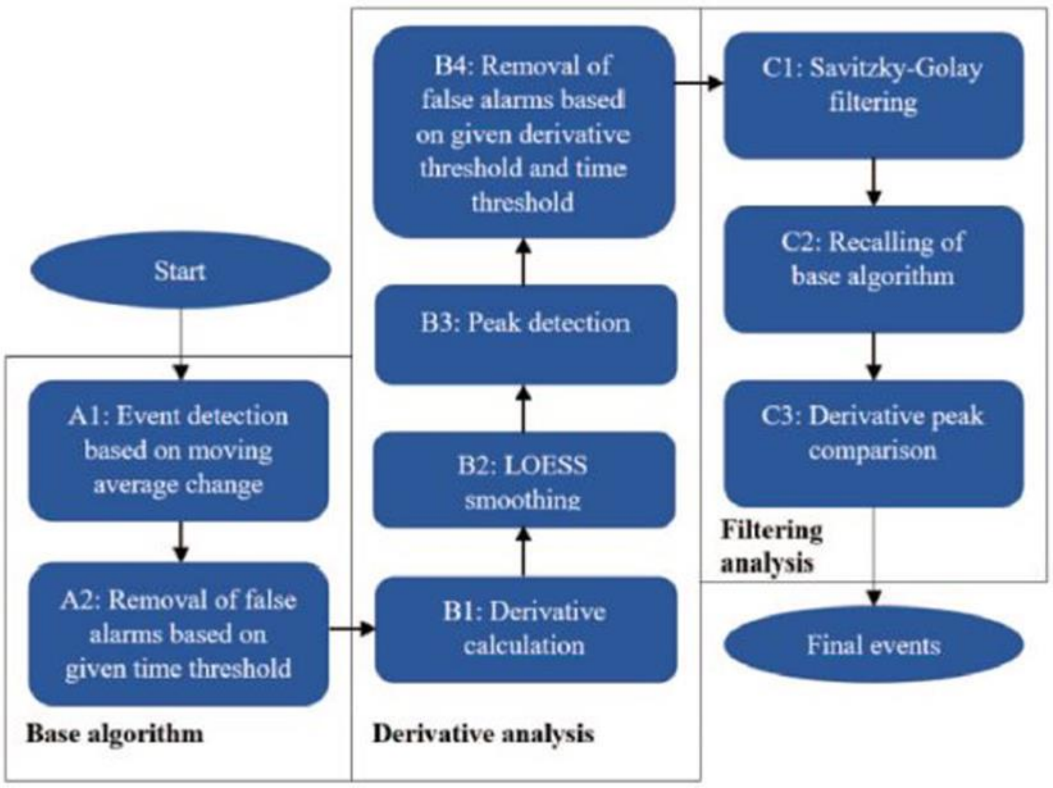

B4: Removal of false alarms based on given derivative threshold and time threshold
C1: Savitzky-Golay filtering
C2: Recalling of base algorithm
C3: Derivative peak comparison
Start
B3: Peak detection
A1: Event detection based on moving average change
B2: LOESS smoothing
Filtering analysis
A2: Removal of false alarms based on given time threshold
B1: Derivative calculation
Final events
Base algorithm
Derivative analysis

Figure 6: Overall hybrid event detection procedure [6]

It's first par is the use of Savitzky-Golay filtering to remove unwanted fluctuations and retain the useful portion of the data. Then the base algorithm is reused for the total re-move of false events. Since the process of filtering may remove and true events, a com-parison between the events and their derivatives is made in order to capture the true events that may have wrongfully removed.

Event detection is one of the main components that must be addressed with great care as it impacts the final output of a non-intrusive load monitoring model, that’s the reason most research develop event detection algorithms consisting of highly complex statistical methods. To overcome the burden of complexity two models based on simple statis-tical features was developed [7], namely the variance and mean absolute deviation. Both algo-rithms developed have as foundation the concept of Sliding Window which is illustrated in Fig. 7.

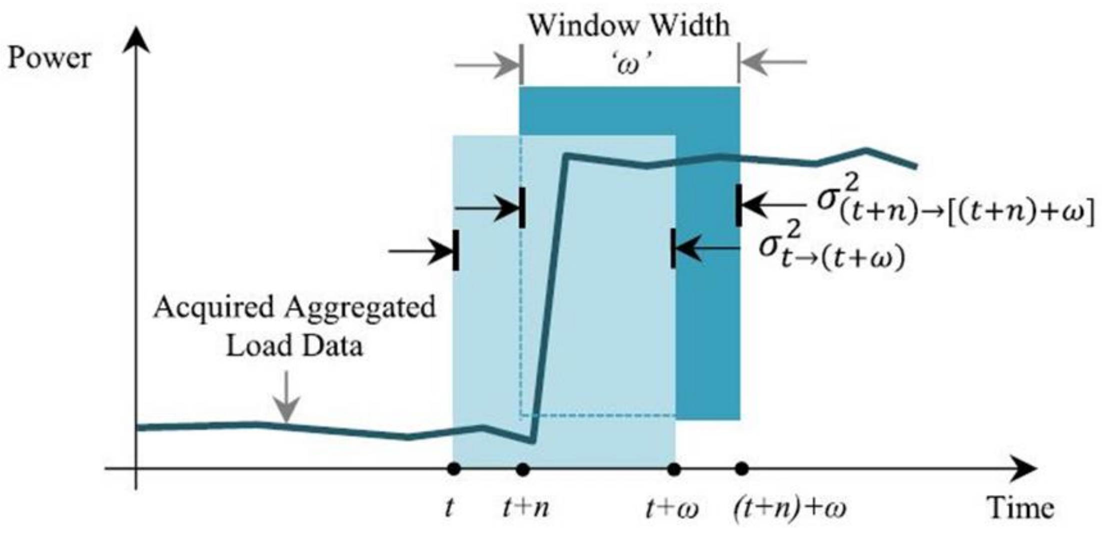

Power
Window Width
'ω'
$\sigma^2_{(t+n)\to[(t+n)+\omega]}$
$\sigma^2_{t\to(t+\omega)}$
Acquired Aggregated Load Data
t
$t+n$
$t+\omega$
$(t+n)+\omega$
Time

Figure 7: Sliding Window representation [7]

Their algorithms were tested by using a public dataset known as Dataport which comprises by household data consumption and appliances consumption data. The first step is the extraction of noise that might exists in the dataset due to extreme fluctuations that the appliances produce. So, median filtering was used as a preprocessing technique. Then the variance and mean absolute deviation is computed. A threshold 'δ' then, was specified in order to reduce the number of errors that are produced because of the dense fluctuations that exist in the dataset. Also, the width of sliding window must be initialized during the initial state of the algorithms. In order to specify the accuracy of the algorithms output they computed the absolute difference of the time the actual evet occurred (this information was provided by the public dataset Dataport) and the time detected evet occurred and compare it with a specified delay tolerance threshold 'Δt'. If the value is below of the delay tolerance threshold, then the event is considered as true positive. The final output of the algorithms is the start and end time indicators of detected events. The algorithms have been applied to the dataset with the same parameters in order to compare them, as evaluation metric Precision and Recall was computed. The variance algorithm achieved a precision of 80.556% and a Recall of 79.573%. The mean absolute deviation algorithm achieved 87.464% and 91.185% respectively for Precision and Recall. Those numbers indicate that the mean absolute deviation algorithm has better performance. To investigate the parameter that has major impact in the performance of the algorithms, a sensitivity analyses was conducted by altering the width of the sliding window and set constant the remainder of the parameters. Their analyses showed that by increasing the width of the sliding window, above a specific limit, the performance of both algorithms diminish dramatically. Therefore, must be treated with care meaning that the best possible value must be selected in order for the algorithms to achieve a desirable performance.

The encumbrance of complexity during the stage of event detection has started to attract the interest of the scientific society in order to engineer a method that will give high score of accuracy with minimal computation and complexity effort. A method that satisfied the aforementioned criteria has been introduced, it is called High Accuracy NILM Detector [8] or for abbreviation HAND. The model uses sliding window technique and calculated the variation of standard deviation for each time interval (window) and then compares it with a specified threshold to distinguish the high amplitude varia-tion events (transient state event) from the low amplitude variation events (steady state events). Their method accepts as input the HSF current signal to detect the events of the appliances. For evaluation purposes they used simulated and real data, computing precision and recall. During the simulations they introduced a delay tolerance threshold to identify the true positive events and the false positive. For the simulated data they used an SNR of 50 dB and two thresholds of 200 ms and 500 ms of delay tolerance. Their model performed poorly for the delay of 200 ms, but for the 500 ms the model achieved 98.19% and 86.20% for precision and recall respectively. To investigate the impact of SNR to their model they computed a sensitivity analysis having stable the delay and al-ternate the SNR. The final verdict of the sensitivity analysis was that their model is vulnerable to noise (SNR) fluctuations and by the authors this was expected since their model is based on standard deviation.

Table 3: True Positive (TP), False Negative (FN), False Positive (FP), Precision and Recall for different SNR values [8]

SNR	**TP**	**FN**	**FP**	**Precision (%)**	**Recall (%)**
50 dB	7399	219	0	100.00	97.13
40 dB	7432	202	15	99.80	97.35
30 dB	7490	186	399	94.94	97.58

Regarding real dataset simulations revealed that the model has detected all events, denoting a high-performance event detection model.

2.3 Energy Disaggregation Model

A generic energy disaggregation method has been tested [9] lacking any prior knowledge concerning appliances specification, usage time, number of devices in use and how long they have been used. The model will be able to identify the number of devices that are used, the consumption power of each device and the amount of time that they have been in usage. To implement the developed model a public dataset has been used which encompass both the aggregated consumption of a household as well as the consumptions of each electric appliance. Sort circuit data has been used mainly to validate the accuracy of the model. The first step was to analyze the raw data in order to detect any anomalies that may lead to inaccurate results. After the preprocessing phase the next step was to discover the starting point and the stopping point of each appliance in order to form the usage period of each appliance.

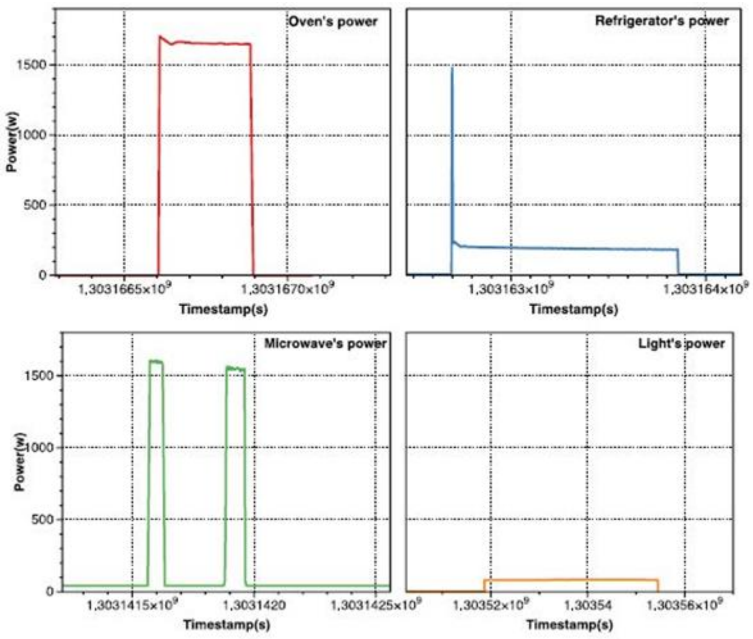


Figure 8: Power data for different electric devices [9]

Fig. 8 illustrates the starting and stopping edges that are formed when a device will start to operate up until it stops. Then, either of the edges can be used to get the appliance level power information. Gaussian Model (GM) was used into the stopping or decreasing edges to model them as each aggregated power value can be modeled after Gaussian Mixture Model (GMM). Expectation-Maximization (EM) algorithms were used to cluster the decreasing edges and get the number of appliances, the number of clusters is denoting the number of appliances. After the estimation of the clusters, the EM algorithm was implemented once again to calculate the model vector. Following the effect of EM algorithm, they proposed a model to optimize the output of the cluster based on the steady operation of the electric devices during the interval of the staring and the stopping edges (the steady operation can be seen from Fig. 8). The final step was to cluster the edges that belong to the same cluster and then to pair the edges that belong to the same appliance in order to discover the duration that each device operate. After following the aforementioned procedure, they evaluated the model based on the measures of the public dataset. They calculated a fluctuation parameter δ which was used to cast out invalid clusters and each unmatching pair (starting and stopping points) was also eliminated. Finally, to determine the performance of their method actual and detected appliance energy signals was compared and found that they are almost identical with a very small portion of fluctuation.

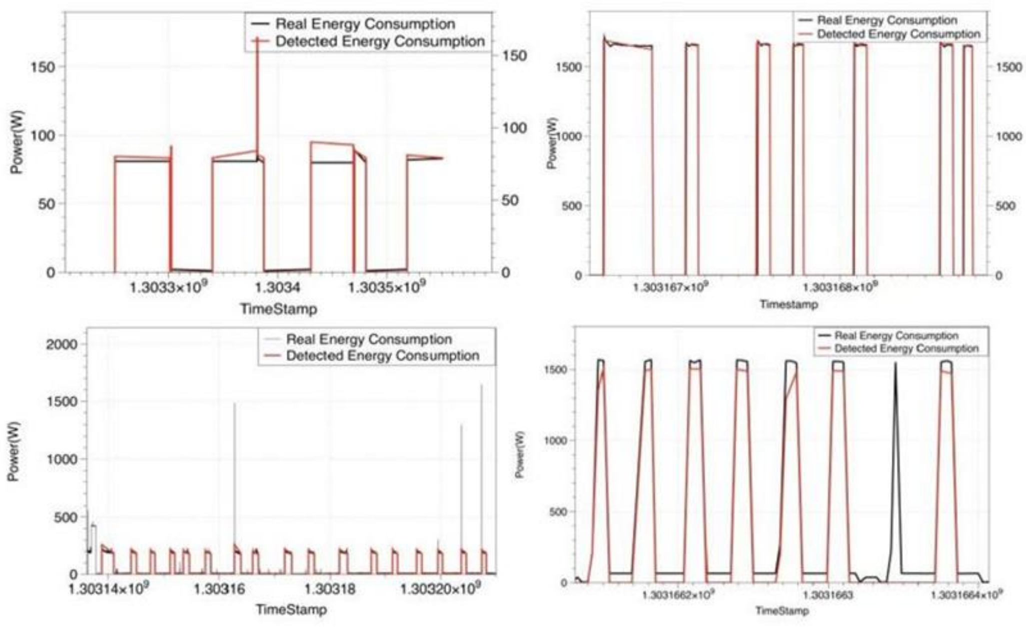


Figure 9: Comparison of actual and detected energy consumption of four appliances [9]

The usage time of each appliance has been modeled and evaluated through the use of false positive and false negative. False positive is the time period that has not identified by their model but actually is a useful time period where some devices was on, and false negative is the time period that the model has identified as a useful period but actually was not. The model has efficiently detected the usage time period with the only exception to be the microwave which has a 13% of misleading accuracy. Similarly, the portion of energy consumed for each appliance was conducted and the real and the detected values was compared showing that the model is highly accurate.

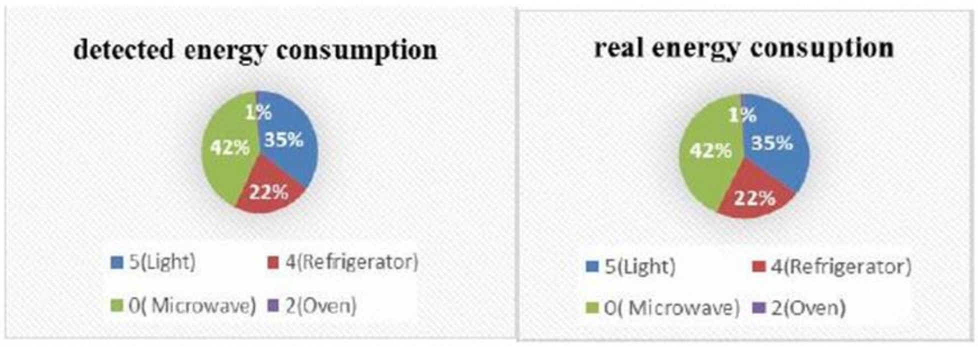


Figure 10: Detected and real energy consumption per appliance [9]

Besides the exceptional results of the model they, expressed that their model has three major limitations that can affect drastically the accuracy of the model. The first one is the frequency of the measures in the dataset. They noticed that as the frequency of the energy consumption data increase the accuracy of the model and generally the performance is depreciated leading to wrong results. The second has to do with the state of the appliance, their model only can identify when an appliance is on and off but in general there are some devices that have more states. The last limitation concerns the fact that the model can perfectly detect the usage period, energy consumption and the number of appliances used but cannot correlate these characteristics with the type of the device which is the major drawback of the model.

A disaggregation technique was introduced [10] using PyCharm, a well-known integrated development environment for python programming language alongside with all the necessary extensions and libraries. Their model was based on events and more specifically in energy consumption changes that are formed when appliances change their state.

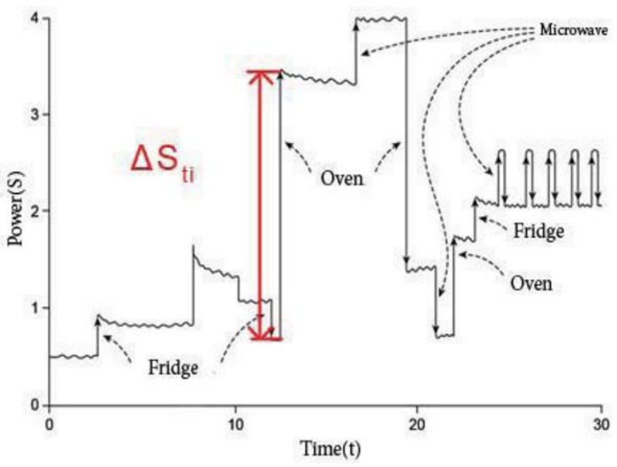


Figure 11: Changes in consumption per appliance [10]

Any statistically significant variation in consumption data that was exceeding a specified threshold was considered as a change. Those changes can be positive (when the consumption of a device is increasing) and negative (when the consumption is decreasing). After the establishment of the cluster pairs the threshold was redefined and the whole process was repeated up until all data was pared. Subsequently, to peer the positive and the negative changes with the records from the database their algorithm creates two graphs one for the measured values and the other for the database and therefore matches are created based on the distance of the two graph nodes. A dataset containing the consumption of six appliances of one day was used to assess the algorithm.

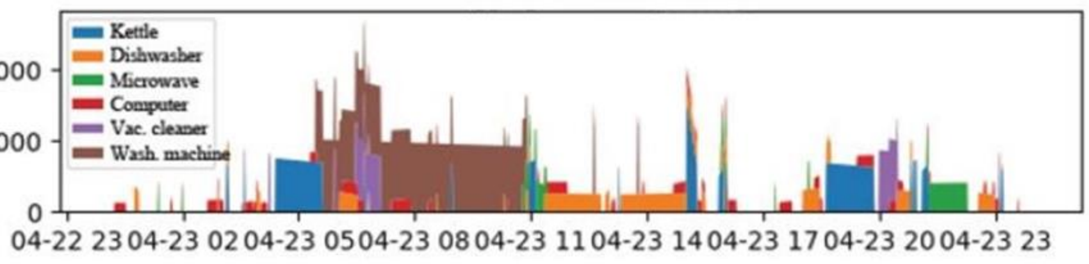


Figure 12: Disaggregation output [10]

Except from real data records, artificial data was used during the evaluation procedure. The output of the algorithm was considered poor for the artificial dataset otherwise in real data the performance was characterized as good with the possibility of improvements.

A cloud-based system was developed [4] called ‘Smart Saver’, which is capable to compute the energy consumption per appliance alongside the aggregated energy consumption in two modes. The first one is the online mode where smart meters can be connected to the system and dynamically or online to calculate the energy disaggregation and the second one is in offline mode where the user can upload energy consumption data and receive back the disaggregated information. Their system is based on three parameters the stand-by power, rated power and power deviation that can be easily found in manuals or through a search on the web for an average electric appliance. Appliance’s state (on/off) was considered as separate variables and thus they created a state matrix and associated it with the pre-defined power parameters. Through the association a sparse switching event recovery (SSER) optimization model was constructed to recover the state matrix for each appliance alongside with the timeline. Based on the rated power and the power deviation the energy consumption with the upper and lower edges for all appliances was established. To implement their SSER model they used a parallel local optimization algorithm (PLOA) that will return to the final user the disaggregation of energy consumption per appliance. The system performed efficiently in online and offline mode and the final output of the system is depicted in Fig. 13.

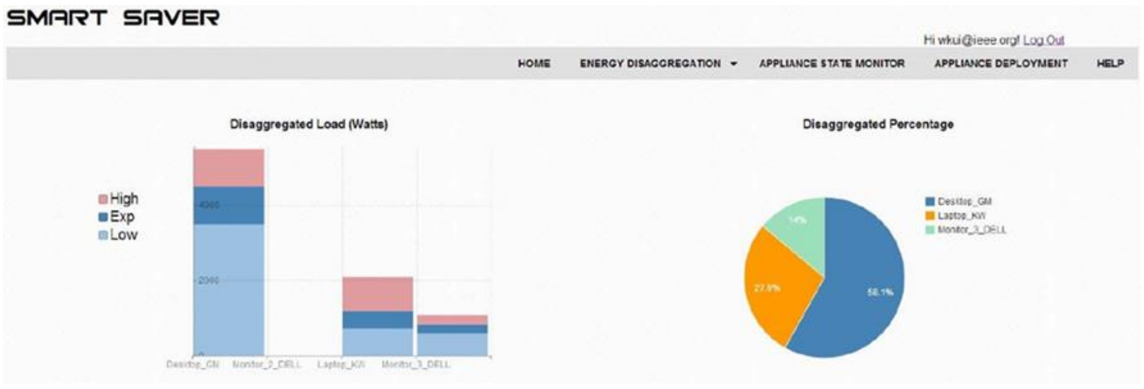
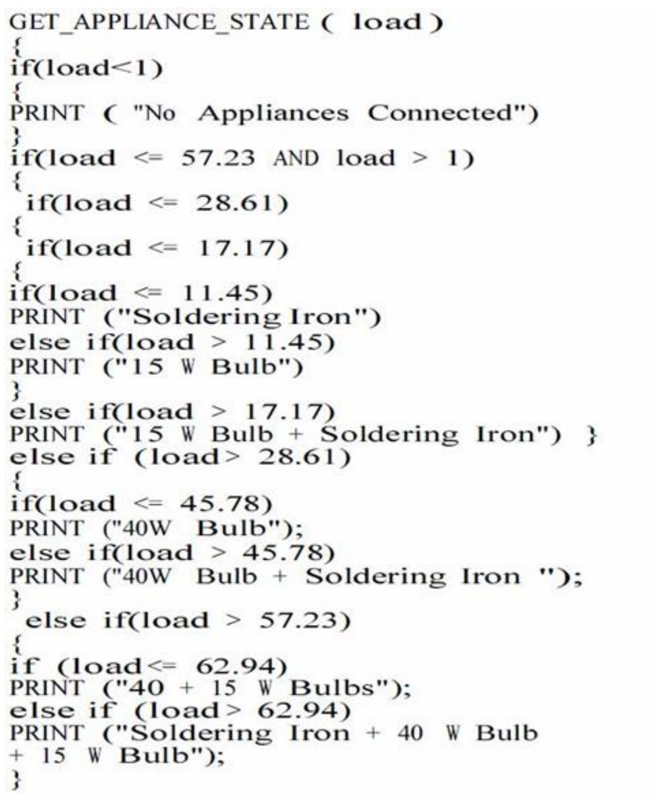


Figure 13: Smart Saver output [4]

A non-complex model has been tested to conduct energy disaggregation [11]. To gather consumption data, they have constructed a smart system that encompasses three sockets that their aggregated consumption is monitored by a smart meter that sends power (Watt) and timestamp into a central server (computer) that stores and computes appliance classification. Two bulbs of 15W and 40W and a soldering iron was used as a load in the sockets. The grid of the system is not staple from the perspective of voltage varying from 220 to 250V leading to unstable consumption. To address this problem efficiently they used machine learning algorithm known as C4.5 Decision Tree. To train the model consumption data of appliances and their aggregated consumption was used, furthermore, to overcome the problem of fluctuations the normal deviation of specific appliances was considered in order for the model to able to perform under these conditions sufficiently. The C4.5 algorithm creates a tree based on the attributes of the dataset and their corresponding values. Information gain is calculated for all attributed and then the highest value is selected as a node. This process is repeated until there is any attribute left on the dataset. WEKA software tool was used and more specifically the J48 algorithm (is the implementation of C4.5 algorithm in WEKA). The model was implemented using a 10-fold cross validation where the dataset is apportionment in 10 parts, 9 of them are used to produce the model and the remaining 1 is used to test the model.



```
GET_APPLIANCE_STATE ( load )
{
if(load<1)
{
PRINT ( "No Appliances Connected")
}
if(load <= 57.23 AND load > 1)
{
 if(load <= 28.61)
{
 if(load <= 17.17)
{
if(load <= 11.45)
PRINT ("Soldering Iron")
else if(load > 11.45)
PRINT ("15 W Bulb")
}
else if(load > 17.17)
PRINT ("15 W Bulb + Soldering Iron") }
else if (load> 28.61)
{
if(load <= 45.78)
PRINT ("40W Bulb");
else if(load > 45.78)
PRINT ("40W Bulb + Soldering Iron ");
}
 else if(load > 57.23)
{
if (load<= 62.94)
PRINT ("40 + 15 W Bulbs");
else if (load> 62.94)
PRINT ("Soldering Iron + 40 W Bulb
+ 15 W Bulb");
}
```

Figure 14: Pseudo code of appliance classification [11]

Fig. 14 illustrates the pseudo code that was produced by the model, it is based on threshold on which the appliances are classified. The accuracy of the model reached 74% leading to low performance, the authors justified that the result of the model is due to three parameters. The first one is that the chosen algorithm was inappropriate to handle the complexity of appliance classification. The second parameter has to do with the frequency that energy consumption data was recorded. During the 4 second interval some appliances might be turned on and off resulting in low resolution of the data and finally the creation of sparks due to small damages in the electrical circuit of the building, unstable voltage of the grid and during the initiation of appliances cause noise in the dataset.

Two-layer classification based on k-nearest neighbors' algorithm (k-NN) was proposed [12]. For the requirements of their study, energy consumption data has been collected from 50 households with a sampling frequency of 800Hz with a duration of 30 days. From the collected data only the active, reactive power and the third and fifth current harmonics where extracted. During the first layer the active and reactive power will be used to classify the appliances, for appliances that share same characteristics it is very difficult to classify them leading to overlaps (two deferent devices are clustered together). To address this problem, they propose the use of a second layer which will use the third and fifth current harmonics of the devices. They explain that despite the similarities of active and reactive power of several appliances the currents harmonics are not similar and thus they selected to utilize the third and the fifth since the odd harmonics fulfil the requirement as can be seen from Fig. 15.

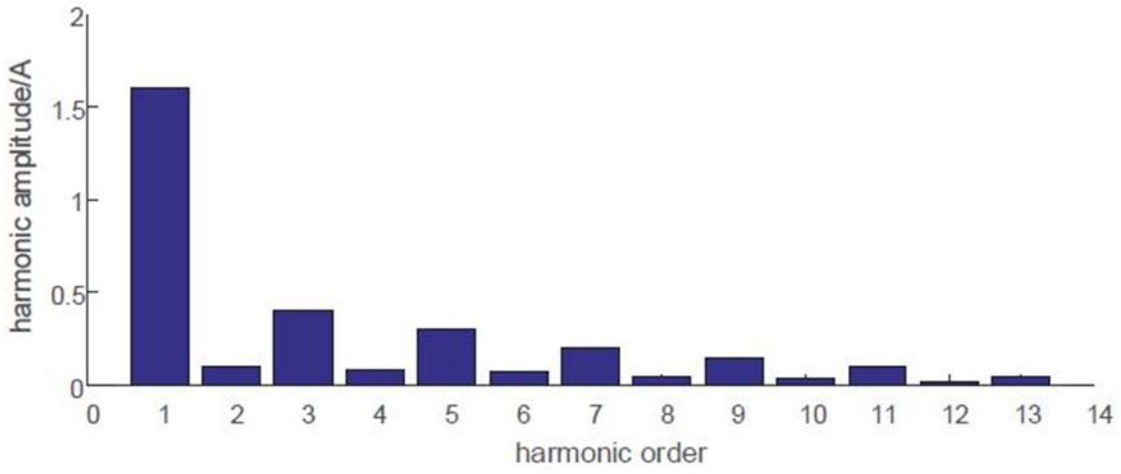


Figure 15: Current harmonics [12]

K-NN calculates the distance of each data point and classifies them based on the nearest point thus for a sample data point x, the nearest point of x in the dataset is denotes as x′, meaning that the x′ belongs to the category of x. To determine the value of k they used an exhaustive method for all possible k values. The highest accuracy was found when k was equal to 20 and they have set a threshold to 15 in order to be used to classify the data. The algorithm generates two clustering graphs, one is for active power and reactive power and the second one is for the third and the fifth harmonics. Each appliance belongs to a single cluster which further belong to one of the quadrants. In the first quadrant takes place the clustering for the devices that operate and in the third quadrant is taking place the clustering for the devices that do not operate. The points of the first and the third quadrant that are part of the same category they are pared and the portion of consumption for each appliance is defined by the power and the period of time that is on and off. In more details they explain that in first layer they extracted the rising of active and reactive power and compare it with a limit point of 50-Watt, if the rising values overcome the 50-Watt limit then the devices are considered to operate, and the timestamp is recorded while if the rising value does not exceed the limit it is considered to operate at a steady-state. After extracting the rising values of active and reactive power the values are searched in the corresponding graph for k neighbors.

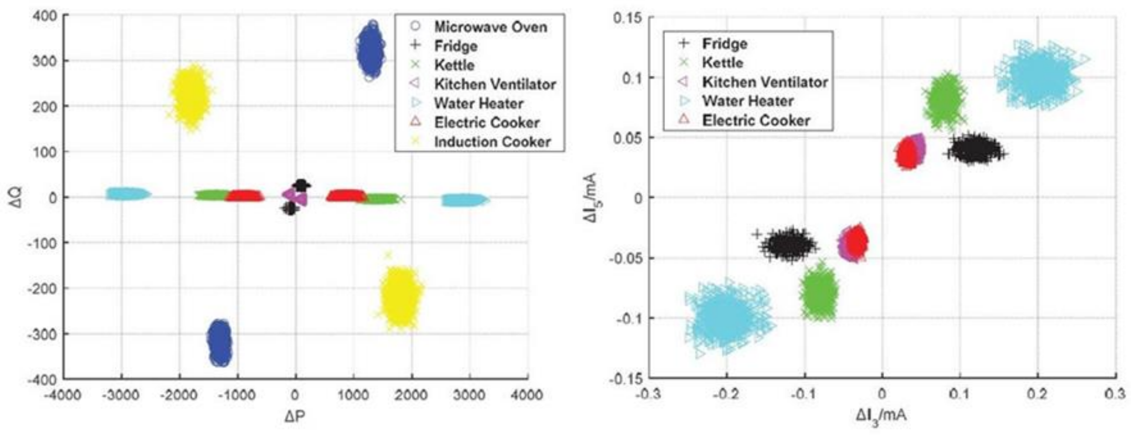


Figure 16: Cluster of active and reactive power (left), cluster of third and fifth current harmonics (right) [12]

If the maximum number of members of the same class is greater of 15 then the datapoint belongs to the same category otherwise if the maximum number of members of the same class is less than 15, then the second layer takes place. The second layer is the same as the first but with the only difference that now the third and the fifth current harmonics are used. For the members that their value is greater than 15 means that the datapoint belongs in the same category and for the members that their values don't exceed the limit of 15 are assigned to the category with the highest sum of neighbors for both layers.

To evaluate the effectiveness of the model they implemented recall, precision, accuracy and F_1 into two steps, the first step considers the first forty households in order to assess model accuracy and the second step which considers the last 10 households in order to test the generalization of the model. Their model reaches acceptable accuracy levels leading to generally pleasant performance.

Table 4: Accuracy of the first 40 household [12]

Household number	**1-10**	**11-20**	**21-30**	**31-40**
precision	0.754	0.714	0.732	0.745
recall	0.841	0.846	0.815	0.826
accuracy	0.902	0.885	0.887	0.895
F_1	0.789	0.775	0.771	0.787

In [13], investigated an alteration of a supervised non-intrusive load monitoring (NILM) model that it is based on dynamic fuzzy c-mean event clustering and k-NN label matching. Over a period of eight months data has been collected from 10 most popular appliances that a household is equipped. Then they have constructed a signature database which is composed from the real power and reactive power events for each of the appliances and their timestamp. To exclude undesired values that may lead to noise, they have conducted several filtering methods and they have concluded that the most appropriate is the total variation denoising.

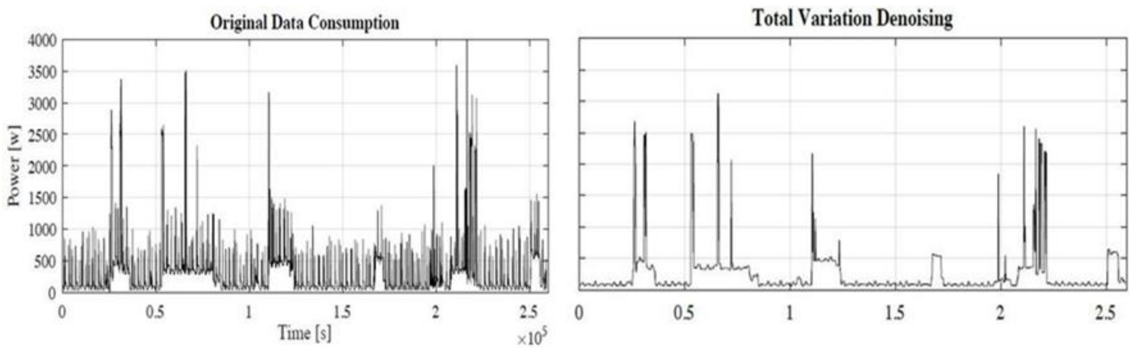


Figure 17: Original data consumption (left), total variation denoising (right) [13]

To identify the edges, they set a threshold, if the real power exceeds it then it is considered that the appliance change state. The reactive power for each appliance event was then calculated and presented into (dQ, dP) space where dQ is the reactive power and dP is the real power.

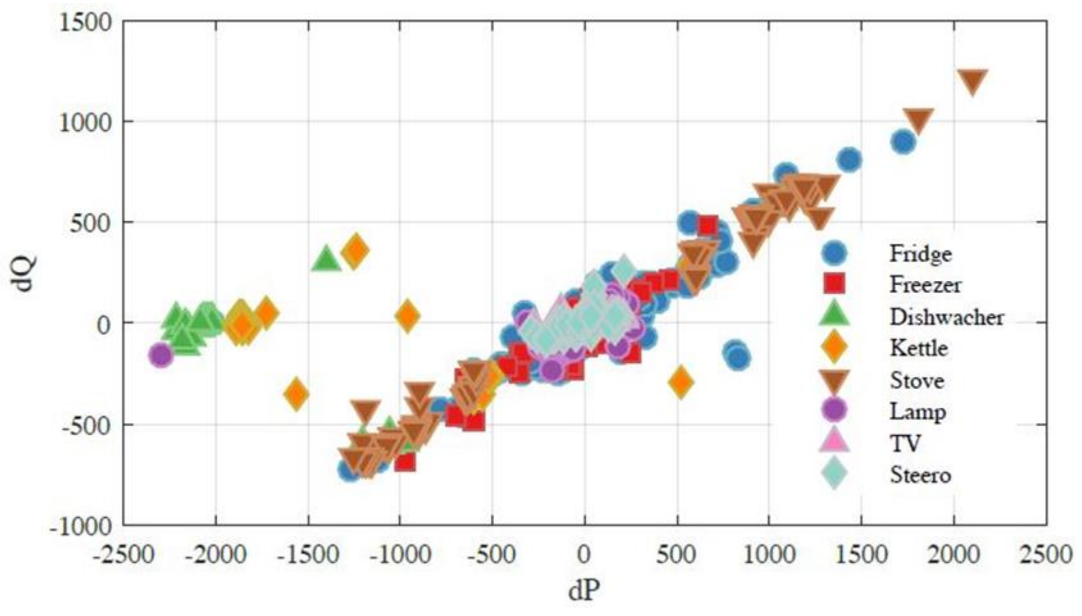


Figure 18: Appliance events in (dQ, dP) space [13]

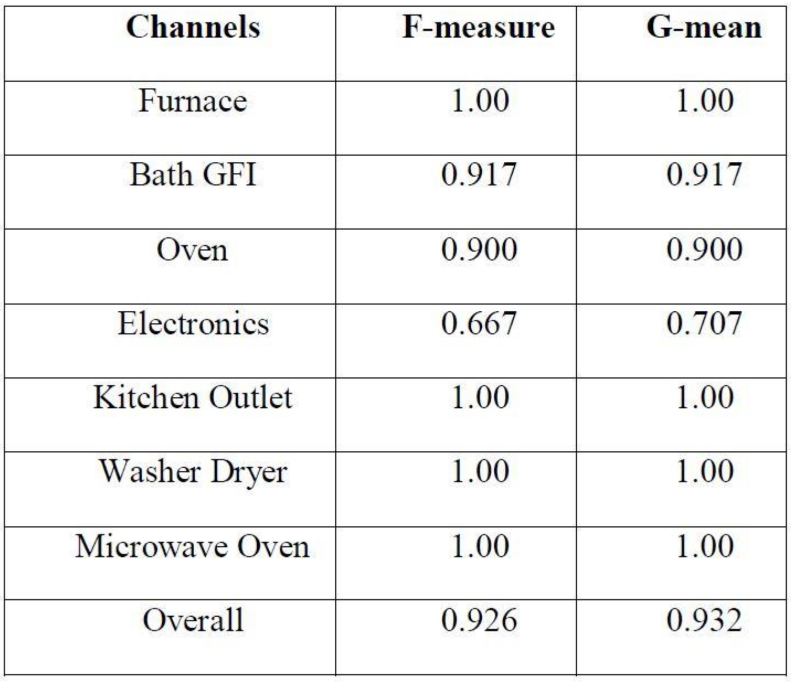
The completion of the signature database was done by performing a fuzzy c-means clustering method in order to identify and cluster the events that best represent each appliance operation. The only drawback of the method is that the number of clusters must be initialized before algorithm implementation. To get through the problem they implemented Akaike Information Criterion and the Bayesian Information Criterion to specify the number of clusters. After the completion of the signature database, they applied k-NN algorithm in order to assign all the events with the most probable appliance signature. Their model performed effectively mainly for heavy consumption appliances reaching a score between 92% and 97% for recall and precision. On the contrary for low consumption appliances their model recorded low values of accuracy leading to values between 20% and 57%, resulting to incorrectly classification of this type of appliances.

K-nearest neighbors (KNN) classification was implemented [14] in order to examine the effectiveness of the algorithm upon the field of energy disaggregation. KNN is a non-parametric supervised classification method that stores the available classes (training set) and then uses them to classify new test points based on a majority vote of its neighbors, the test point is assigned to the class most common amongst its k nearest neighbors measured by a distance function. There are several distance functions such as Euclidian, Manhattan etc., the authors of the study choose to use the Euclidian distance which is the length of line segment between two points.

$$Euclidian\ Distance = \sqrt{\sum_{i=1}^{k}(x_i - y_i)^2}$$

To test the algorithm, they used a well-known dataset which is called REDD and has been constructed in order to be used for energy disaggregation models. From the dataset they have extracted and preprocessed the consumption of seven appliances (furnace, bathroom ground fault interrupters outlets (plugs), oven, electronics (laptop, TV, speakers etc.), kitchen plugs, washing machine, dryer, microwave-oven), the aggregation of these appliances data constitute the final dataset which they used. F-measure and G-mean was computed to evaluate the accuracy of the KNN algorithm.

Table 5: F-measure and G-mean for KNN classifier [14]

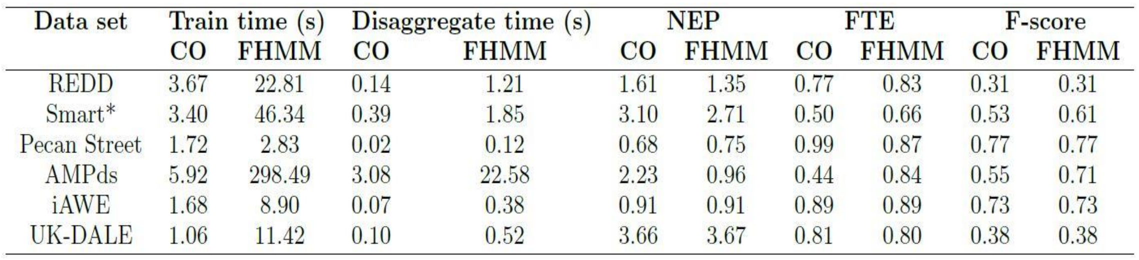


Channels	F-measure	G-mean
Furnace	1.00	1.00
Bath GFI	0.917	0.917
Oven	0.900	0.900
Electronics	0.667	0.707
Kitchen Outlet	1.00	1.00
Washer Dryer	1.00	1.00
Microwave Oven	1.00	1.00
Overall	0.926	0.932

For heavy consumption appliances the model achieved outstanding results reaching a remarkable accuracy of 100% which indicated that the model classifies correctly heavy appliance, contrary for appliances with low consumption such as TV, speakers, PC that constitute electronics appliance in a typical household the model performed poorly reaching an accuracy of 50%, meaning that half of the electronics devices are misclassified.

The creation of a benchmark model for implementing energy disaggregation was developed, namely the Non-intrusive Load Monitoring Tool Kit (NILMTK) [15]. The tool kit is publicly available and accommodated by documentation to help researchers to study it and use it as benchmark. In more details the NILMTK was created in python since it is already providing tools and modules that supports data mining and machine learning topics. The tool kit was tested by the use of several publicly available datasets. Since these datasets have different characteristics (e.g., sampling rate, power features) the tool kit provides a universal data format which can be diagnosed and applied to diverse statistic model in order for the user to be able to evaluate the characteristics of the dataset and explore the appliance usage. Diving more deeply in tool kit capabilities, the next feature is the preprocessing which provides normalization to clean the dataset. The tool kit has the option to perform two disaggregation algorithms, the Combinatorial Optimization (CO) and the Factorial Hidden Markov Model (FHMM), the choice of the algorithms was made by the authors considering the fact that their popularity and extensive research upon energy disaggregation had a fruitful background. Considering that the tool kit can be manipulated as a benchmark evaluation tool, they integrated several evaluation metrics to give the user the ability to perform their load classification but also to assess the performance in order to be able to investigate the features that impact their model.

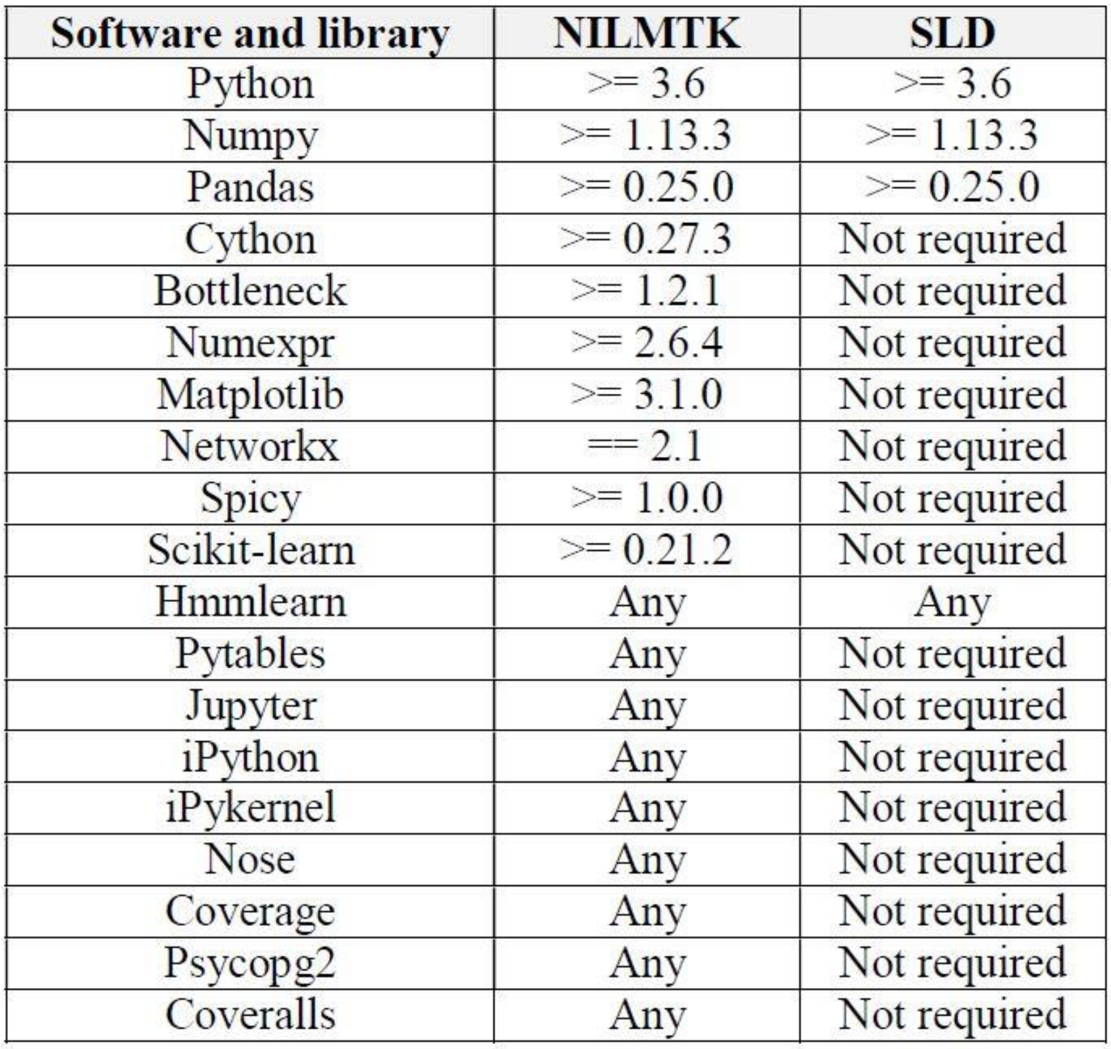
Table 6: Comparison between Combinatorial Optimization (CO) and Factorial Hidden Markov Model (FHMM) for different public datasets [15]



Data set	Train time (s)		Disaggregate time (s)		NEP		FTE		F-score	
	CO	FHMM	CO	FHMM	CO	FHMM	CO	FHMM	CO	FHMM
REDD	3.67	22.81	0.14	1.21	1.61	1.35	0.77	0.83	0.31	0.31
Smart*	3.40	46.34	0.39	1.85	3.10	2.71	0.50	0.66	0.53	0.61
Pecan Street	1.72	2.83	0.02	0.12	0.68	0.75	0.99	0.87	0.77	0.77
AMPds	5.92	298.49	3.08	22.58	2.23	0.96	0.44	0.84	0.55	0.71
iAWE	1.68	8.90	0.07	0.38	0.91	0.91	0.89	0.89	0.73	0.73
UK-DALE	1.06	11.42	0.10	0.52	3.66	3.67	0.81	0.80	0.38	0.38

Authors of [16] pointed out that Non-Intrusive Load Monitoring Toolkit (NILMTK) is the only software tool that can be utilized as a starting point or rather as a reference point to evaluate the accuracy of different NILM models. They state that the NILMTK needs a lot of time and effort to be understood meaning that the implementation of the toolkit is demanding but at the same time it offers a vast majority of capabilities. To tackle the complexity of NILMTK they produced a Simple Load Disaggregation (SLD) library which is based on NILMTK however they have removed all unnecessary tools and functionalities and retain only the part of energy disaggregation segment. SLD uses Factorial Hidden Markov Model (FHMM) and Combinatorial Optimization (CO) model which are inherited from the NILMTK with an exception that their implementation in the SLD was modified in order the model to use python dataframe and not Hierarchical Data Format (HDF5) that it is used by the NILMTK. The preprocessing process was kept simple and as a supervised model it contains the training and the test sets. The procedure of the training part requires a dataframe that contains the total consumption of a household, some appliances consumption, and timestamp. Accordingly, the test set contains the timestamp and the total consumption of a household. The model has the functionality to be trained for a variable number of appliances leading to an equivalent number of disaggregated appliances. The overall employment of the model is straightforward with a few lines of code and with not as much libraries as the NILMTK requires, with respect to the accuracy of the model, authors point out that both of the models share the same level of accuracy making the SLD library a user-friendly version of NILMTK for energy disaggregation.

Table 7: Comparison between the software requirements of NILMTK and SLD [16]



Software and library	NILMTK	SLD
Python	>= 3.6	>= 3.6
Numpy	>= 1.13.3	>= 1.13.3
Pandas	>= 0.25.0	>= 0.25.0
Cython	>= 0.27.3	Not required
Bottleneck	>= 1.2.1	Not required
Numexpr	>= 2.6.4	Not required
Matplotlib	>= 3.1.0	Not required
Networkx	== 2.1	Not required
Spicy	>= 1.0.0	Not required
Scikit-learn	>= 0.21.2	Not required
Hmmlearn	Any	Any
Pytables	Any	Not required
Jupyter	Any	Not required
iPython	Any	Not required
iPykernel	Any	Not required
Nose	Any	Not required
Coverage	Any	Not required
Psycopg2	Any	Not required
Coveralls	Any	Not required

An alteration of non-intrusive load monitoring model was tested [17] based on event detection and random forest optimized by particle swarm optimization. The identification of events was done by comparing the power difference of two points between a threshold, if the power difference is above or below the specified threshold value, then an event is spotted. The event detection algorithm consists of two parameters, the number of sampling points per cycle and the threshold. The most interesting part is that in order to accomplish the energy disaggregation / classification they used Random Forest algorithm, this algorithm constructs multiple decision trees that are produced from subsets and features of the main dataset. Multiple classifiers of low week accuracy are produced with the ultimate goal to form a universal highly accurate classification. The main problem of Random Forest is the fact that a number of parameters have great impact in the final output and there are only initialized by experience and background knowledge. To address the issue Particle Swarm optimization algorithm was used. All parameters that have high magnitude formed a space vector which in further was fed into the Particle Swarm algorithm as a particle. Through a repetitive process the Particle Swarm algorithm will select the best solution of parameters to fed back in the Random Forest algorithm improving the final accuracy of the model. A dataset that includes the power consumption of thirteen appliances was used to implement the overall model. From the whole dataset the 80% was manipulated for training the model and the remaining 20% used from testing the model. The final output of the model is to classify the state of the appliance and its type, for appliances that their state is on the model achieved an accuracy of 98.9%, on appliance that their state is off the accuracy was slightly decreased reaching 97.5%.

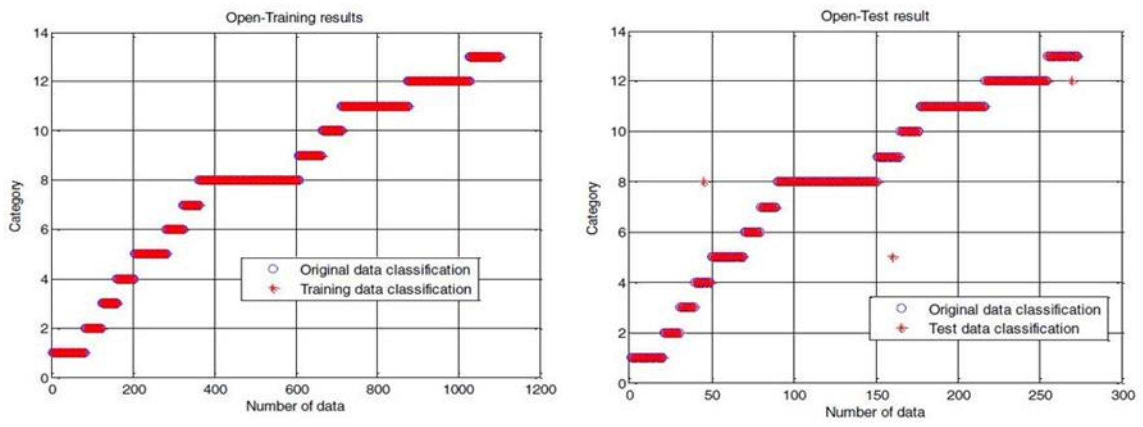


Figure 19: Train (left) and test (right) result for power-on appliances [17]

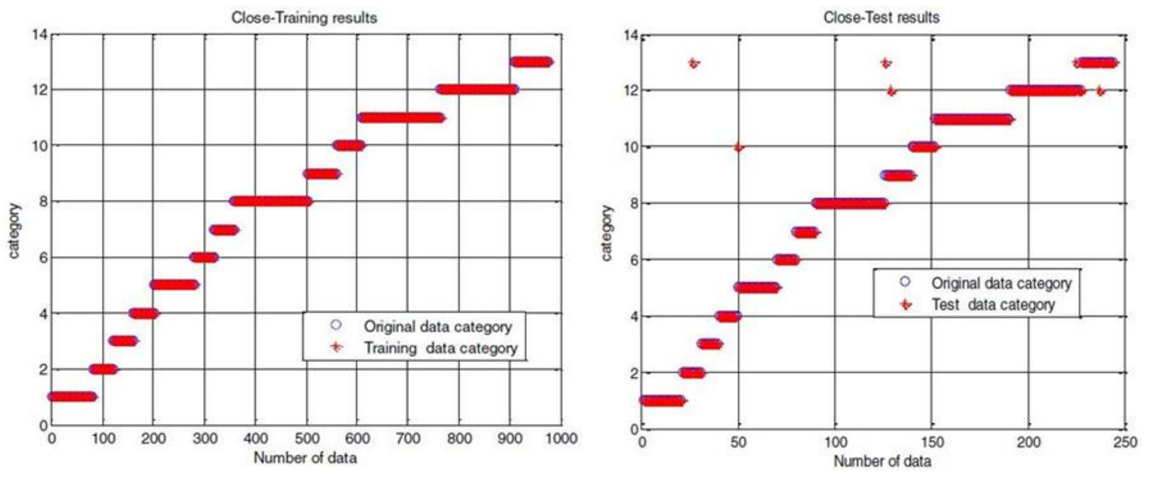


Figure 20: Train (left) and test (right) result for power-off appliances [17]

Another non-intrusive load monitoring model made [18] to identify the type of appliances and the number that exist in an office. They installed a smart meter in the central panel of the office and extracted the current, voltage and power with a resolution of one minute. To construct the load feature for all the appliances and consequently for the whole office the characteristics measured from the smart meter used to calculate the active power and the inactive current which form the load feature.

After the construction of load future vectors, they proceeded with the normalization of the data based in time domain then the implementation of the fuzzy c-means clustering took place in order to cluster the appliances based on similarities on load feature characteristics creating a specified number of clusters which was initialized taking into account the inter-cluster entropy. To achieve the best possible value for the number of clusters they computed the inter-cluster entropy for a diverse number of appliance categories. After plotting their findings, they concluded that the optimal value is to use 5 clusters since was gave them the highest score for the inter-cluster entropy.

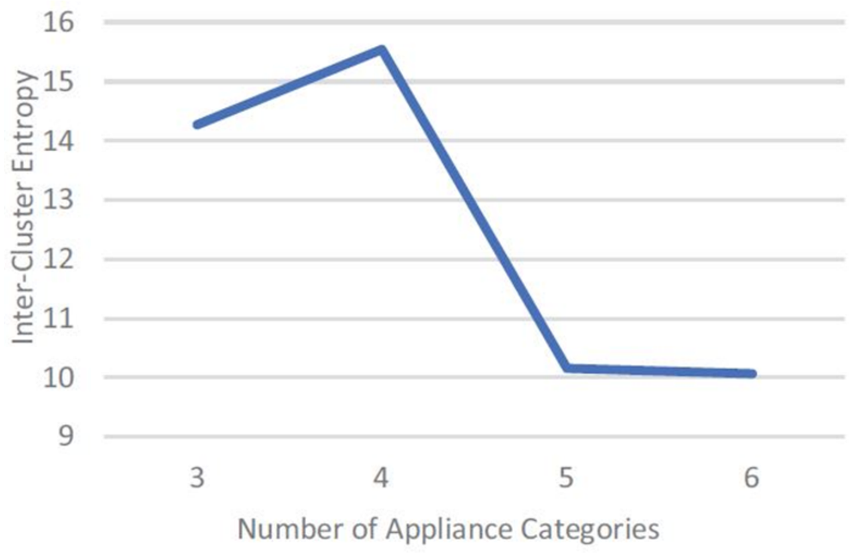


Figure 21: Plot of inter-cluster entropy for different number of categories (appliance type) [18]

To evaluate their model performance with respect to appliance categorization accuracy they investigated the number of appliances that exist in the office and found that there are 6 appliances, resulting in five out of six estimations for their model.

The efficiency of Decision Tree classification [19] when it is implemented in energy disaggregation was investigated. This study does not use a publicly available dataset not either, data from smart meters. They have used a software tool to simulate the circuits of four electric devices and extracted the voltage and current and afterward they used the data to calculate the active, reactive and apparent power, then they applied Discrete Fourier Transform to obtain the fundamental frequency component. The resulted parameters form the power component that was employed as features in the disaggregation procedure. They choose to utilize Decision Tree classification based on two scenarios, in the first one the parameters used are the power components they calculated and in the second one, they used the change in power components in more details these changes are variations applied in voltage, frequency and harmonic. In order to identify the best split for each of the attributes in the nodes they calculated the Gini index, the attribute with the lowest value was used to split the node. They pointed out that because the Decision Tree is utilized mainly for binary classification that means that their method can classify data into two classes which raised a problem since energy disaggregation has multiple appliances to be identified. To overcome this barrier, they implemented a solution which they call it the ‘one against rest’, they set a class as positive and the rest as negative this gave them the ability to execute binary Decision Tree. The dataset was split in two parts, half was used to train the model and the other half to test it. To extract the accuracy for both scenarios, index μ was utilized.

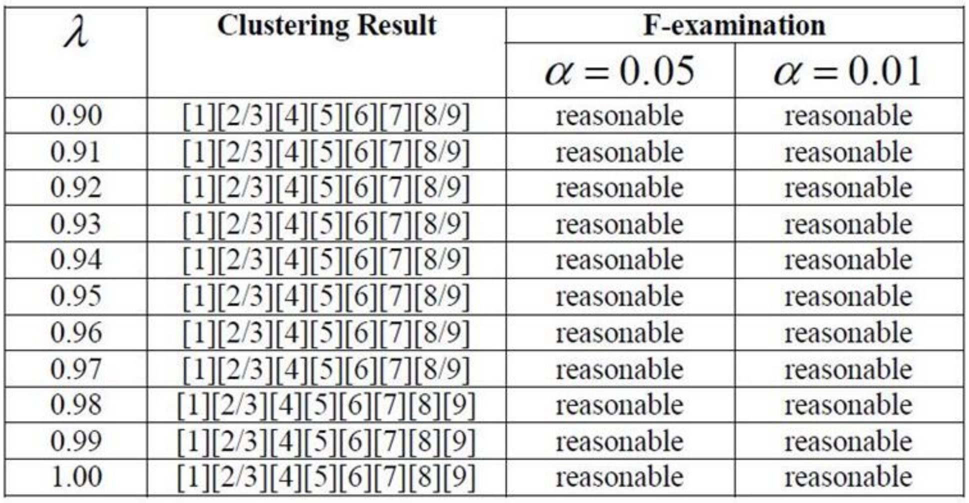
Table 8: Classification accuracy of two scenarios [19]

Positive class	Accuracy using actual power	Accuracy using change in power
Battery	68.29	100
CFL	73.97	98.83
PC	78.08	99.41
LB	72.40	99.61
Mean	73.19	99.46

Table 9 illustrates the impact that the change in power has in the accuracy, for all appliances a difference of 26.27% was recorded compared to the actual power data.

Active power readings [20] harnessed to construct a model to classify the appliances based on Fuzzy Cluster analysis. In order to categorize the appliances correctly they utilized the nonlinear curve graphics of loads due to the fact that appliances sharing the same type have identical values, on the contrary appliances with different type present major dissimilar nonlinear curve graphics. Then they brought the data to the same unit and build the similarity matrix, the similarity matrix presents the statistical similarity measure between the data to by classified. Then the cluster was constructed via the transformation of the matrix into an equivalent fuzzy matrix. The transformation was made using transitive closure method. For different values of λ the output of the cluster analysis is different, in order to evaluate the output of the cluster analysis they computed the F-examination method. If the F-examination is greater to some extent, denotes that the value of dissimilarity is wide among the clusters.

Table 9: Cluster output based on different λ values [20]



λ	Clustering Result	F-examination	
		$\alpha = 0.05$	$\alpha = 0.01$
0.90	[1][2/3][4][5][6][7][8/9]	reasonable	reasonable
0.91	[1][2/3][4][5][6][7][8/9]	reasonable	reasonable
0.92	[1][2/3][4][5][6][7][8/9]	reasonable	reasonable
0.93	[1][2/3][4][5][6][7][8/9]	reasonable	reasonable
0.94	[1][2/3][4][5][6][7][8/9]	reasonable	reasonable
0.95	[1][2/3][4][5][6][7][8/9]	reasonable	reasonable
0.96	[1][2/3][4][5][6][7][8/9]	reasonable	reasonable
0.97	[1][2/3][4][5][6][7][8/9]	reasonable	reasonable
0.98	[1][2/3][4][5][6][7][8][9]	reasonable	reasonable
0.99	[1][2/3][4][5][6][7][8][9]	reasonable	reasonable
1.00	[1][2/3][4][5][6][7][8][9]	reasonable	reasonable

Table 10 illustrates the impact that the number of memberships λ has on the number of clusters. As value of λ increases the number of clusters differs for example for λ = 0.90 the number of clusters is 7 while for λ = 1.00 the number of clusters is 8. To classify the data based on the clusters that they have constructed they used fuzzy pattern recognition based on maximum membership degree principle and used Euclidian distance to compare the membership degree. Their model was highly accurately for appliances of different type even if their consumption was the same. For appliance of the same type with the same consumption their model was poorly performed, but this behavior was expected by the authors of the study since they used only the active power to build the model.

Factorial Hidden Markov Model (FHMM) was used [21] to classify the appliances based on single point measurements from a smart meter. FHMM is an alteration of Hidden Markov Model (HMM) where each appliance is expressed by a single HMM model and the aggregation of multiple HMM models at specific time constitute the FHMM model. For that reason, they pointed out that in order for their model to be effective they need prior knowledge of appliances type, number, and consumption. Since most of energy disaggregation research deal with binary data, they constructed two models based on binary and transient state data and compared the efficiency of the model between the two state approaches. To construct the dataset, they installed smart plugs in their research offices in all appliances that was installed and aggregated this data into a central server. To demonstrate the online capabilities of their model they have built a smartphone application which reports the energy disaggregation for each appliance of the offices. They computed 10 cases of diverse appliance combination alongside with a case where all appliances are considered for evaluation purpose. For the binary state they also modeled different state combinations for each of the cases. With respect to load characteristics, from the consumption readings they calculated 5 features and used them to construct 5 FHMM models of different features.

Table 10: Different FHMM models and their features [21]

Number of models	Features
F1	Average of real power consumption
F2	Average of reactive power alongside with real power
F3	Average of reactive power, real power, and power factor
F4	Average of reactive power, real power, power factor and standard deviation of real power
F5	Average of reactive power, real power, power factor, standard deviation of real power and standard deviation of reactive power

The evaluation of their model was done by computing the F-measure and comparing their approach with a well-known event detection approach that uses the Generalized Likelihood Ratio (GLR). With respect of the binary state their model illustrated that by increasing the number of features their model achieved higher values of F-measure except from the F5 model which the standard deviation of reactive power had a negative impact. Generally, the greater the number of features used the greater the accuracy of the model was.

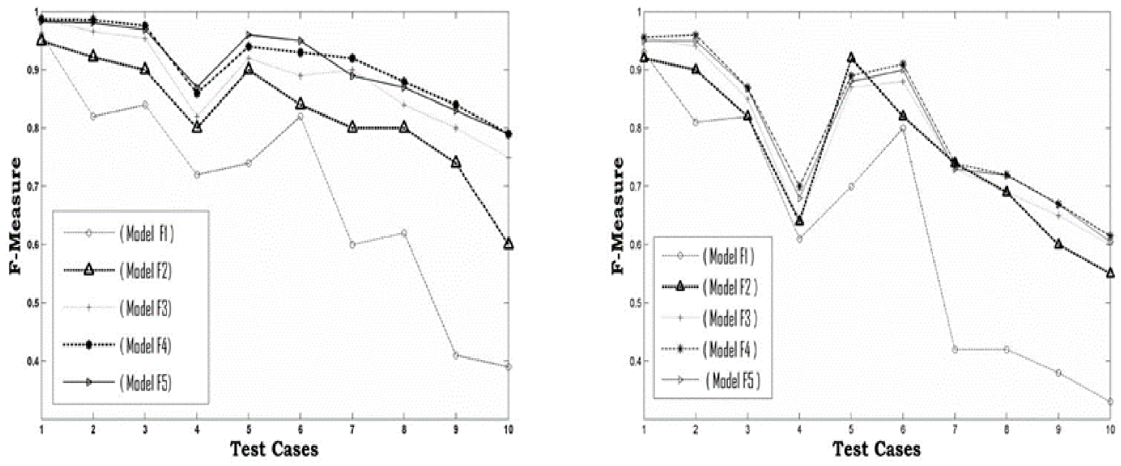


Figure 22: F-measure comparison of five models for binary start (left) and transient state (right) [21]

Their proposed approach with respect to GLR showed that their model is greatly efficient in both scenarios of binary and transient state. For transient state the output of the model was inverse compared to binary state. As the number of features was increased the accuracy of the model was getting decreased. Overall, the authors reach an F-measure of 0.906 for the binary state and 0.804 for transient state appliance operations.

2.4 Recommendation System

Recommendation systems can by implemented to suggest the end-user solutions or to provide suggestions based on user or diverse user's preference. Two well-known techniques are implemented to construct recommendation systems content based filtering and collaborative filtering. Content based filtering takes into consideration only the preference / feedback of a single user that has been extracted during previous transactions. Then products that have highly similar features with the user preference are recommended by the system.

Collaborative filtering is not restricted only on single user data, it accumulates user's preferences and searches for similarities between the preferences and the items to provide recommendations. Furthermore, collaborative filtering is divided into three categories neighborhood-based, model-based and hybrid method.

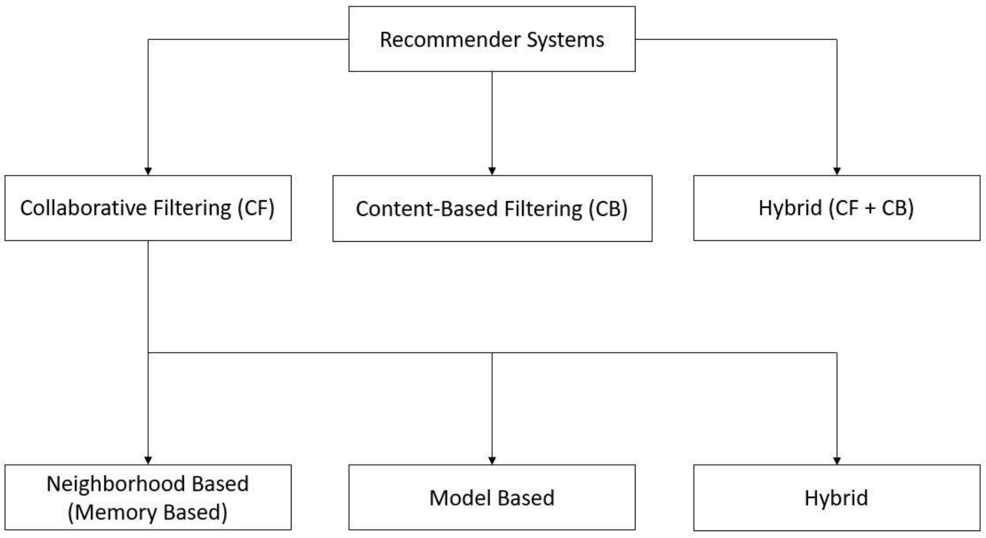

Recommender Systems
Collaborative Filtering (CF)
Content-Based Filtering (CB)
Hybrid (CF + CB)
Neighborhood Based (Memory Based)
Model Based
Hybrid

Figure 23: Recommender Systems Techniques [22]

Neighborhood-based collaborative filtering exploits the behavior of multiple users in order to predict what a candidate buyer (user) might want, by finding similar users with the candidate user based on previous preferences and recommend items that similar users have already expressed their preferences. Model-based collaborative filtering models a system that is established on the ratings that have been created from past user item transactions and utilizes them for predicting new ratings. The hybrid method incorporates the use of both neighborhood-based and model-based methods.

Based on collaborative filtering an electricity plan recommendation system [23] was built to suggest their user with the most suitable energy plan (plan having the lower charge price) based on weekly appliance data. First, they extracted training rating set and training feature set, rating set is considered as the normalized price charged by a plan on a user and the feature set is composed of the usage pattern of several appliances. Any missing feature from the sets will automatically make the implementation of similarity metrics inappropriate. To re-solve this problem a hybrid similarity metric composed of Jaccard, and weighted Euclidean similarity was implemented.

$$Jaccard\ Similarity: S_{mn}^{J} = |A_m \cap A_n| / |A_m \cup A_n|$$

$$Weighted\ Euclidean: S_{mn}^{\omega E} = 1 - \sqrt{\left(\sum_{a \in A_m \cap A_n} |A| \omega_a (f_{ma} - f_{na})^2\right) / |A_m \cap A_n}$$

Users with highest similarity are selected and their ratings are used to estimate potential ratings (recommendation). During last part, they present recommendations to the final user with the plans having the highest scores presented first.

An additional recommendation system that utilized the capabilities of collaboration filtering is the Personalized Residential Energy Usage Recommendation System (REURS) [24]. The aim of REURS is to group residential users based on two categories, those that are highly responsive and those that are not. Then the system will search to find similarities between the two groups based on the number of appliances that they operate and the usage pattern, if those conditions are met then the consumption of a high responsive user is compared with the consumption of the non-highly responsive user and the analogues recommendation is provided to the non-highly responsive user.

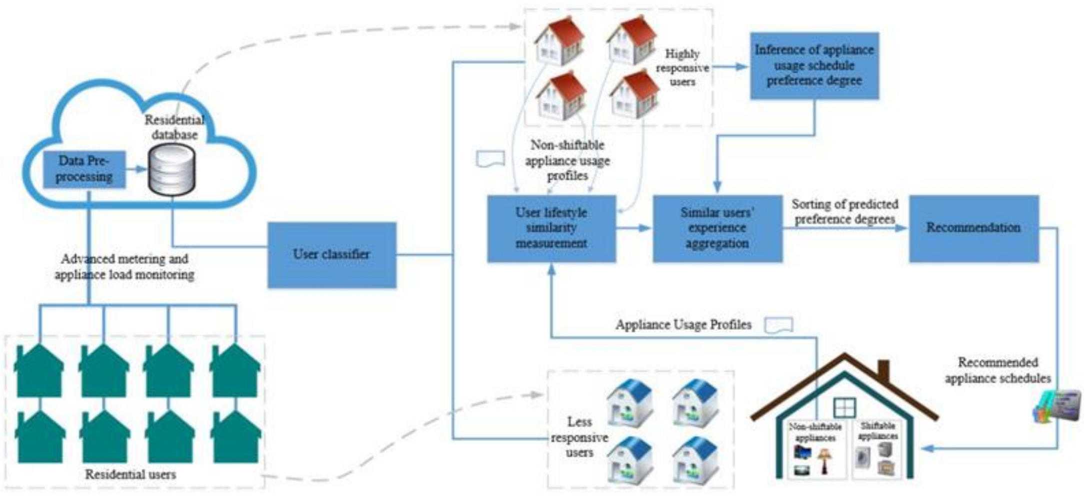

Residential database
Data Pre-processing
Advanced metering and appliance load monitoring
User classifier
Residential users
Highly responsive users
Inference of appliance usage schedule preference degree
Non-shiftable appliance usage profiles
User lifestyle similarity measurement
Similar users' experience aggregation
Sorting of predicted preference degrees
Recommendation
Appliance Usage Profiles
Recommended appliance schedules
Less responsive users
Non-shiftable appliances
Shiftable appliances

Figure 24: Overview of REURS system [24]

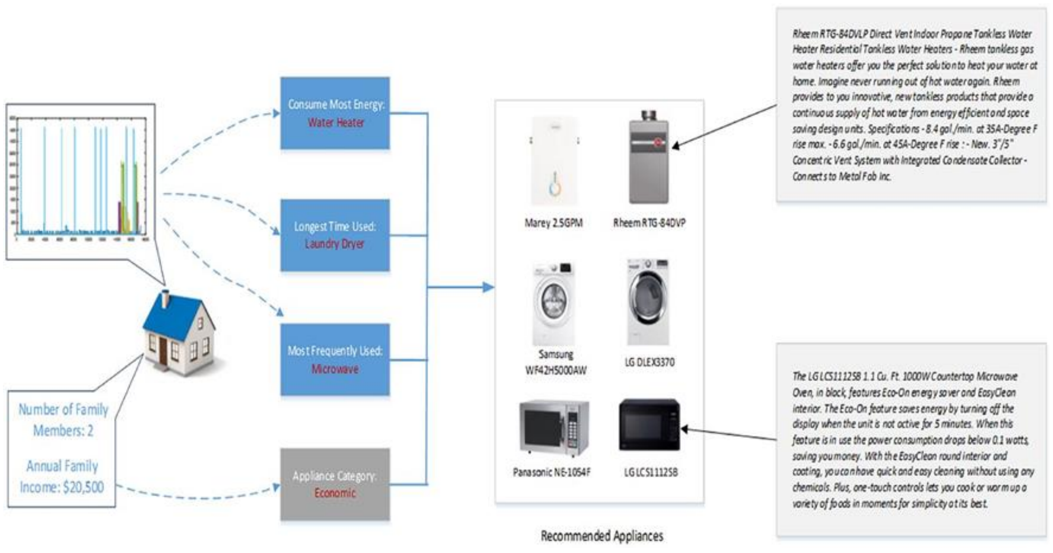
In details, the system uses accumulated daily appliance usage profiles and time-of-use tariff. To categorize the data into groups of highly responsive and non-highly responsive users they need first to identify the feature that will be used to group the data. First, they split the data into multiple groups on similar type of appliances and time-of-use tar-iff and clusters the data using fuzzy C-means clustering method based on mean monthly energy consumption and cost of electricity. Highly responsive are considered those with low energy expenditures and conversely non-highly responsive those with high energy expenditures. The connection link of the two main groups is the lifestyle of each user which is translated to the operation patterns of their appliances. They consider non-shift-able (appliances that the user operates at specific time of a day without the ability to change operation time) and shiftable (appliances that the user can schedule and specify the time of use within a day) appliances to build common profiles within users that be-long to different group. To identify common profiles, they implemented the Cosine sim-ilarity metric.

$$Cosine\ Similarity(A,B) = \frac{A \times B}{||A|| \times ||B||} = \frac{\sum_{i=1}^{n} A_i \times\ B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \times \sqrt{\sum_{i=1}^{n} B_i^2}}$$

Following, their system aggregates the usage pattern of shiftable appliances of high re-sponsive users and implements implicit collaborative filtering (calculates ratings based on users' behavior). Lastly, after the identification of common usage pattern between the high shiftable and the non-high shiftable users the system calculates the ratings (the pref-erence degree of the user on a shiftable appliance usage pattern) and recommends the corresponding plans with the highest rating value.

A system tailored to reduce the expenditures of electricity in a residential building by providing recommendations to the users was developed [25], by using an agent-based architecture. They implemented a central system composed of multiple agents that each one of them had to perform a different task and by aggregating their results the system can recommend actions that have as a result the reduction of electrical expenses. Consumption data of appliances are recorded every 6 seconds, moreover the system receives information from external recourses. External information is the hourly price of electricity and more specifically cost of kilowatt hour (kWh). Following the acquisition of in-formation’s, the next part of the system was to preprocess the data in order to reduce the noise that might exist since originate from diverse resources. Appliances then are classified based on their ability to shift their use. Thus, two categories are formed shift-able and non-shiftable. Usage pattern of the appliances is extracted for weekly and hourly base to understand how each user operates their appliances, to gain that knowledge they implemented knowledge-based technique. Also, two types of recommendation are performed, long-term and sort-term. Long-term take place based on the appliance use for a period of a week and compared with the cost of electricity the ap-propriate recommendations are proposed. Short-term are made within the period of a single day and considered more active. The recommendations are performed by using utility-based technique.

A system was developed following the framework of content-based filtering [26], particularly they firstly collected samples of representative electrical appliances Ads and separated them into groups of expensive and inexpensive appliances. The difference between the groups in not only the price, but also appliances which categorized as expensive have numerous functionalities and greater energy performance compared with inexpensive appliances. The appliances Ads was used to represent the users’ interests on appliances and generate the user profiles. To acquire energy consumption data for each appliance they implemented non-intrusive load monitoring based on Hidden Markov Model method. NILM gave them which appliance is operating the most and the usage pattern. Then users’ interest and needs are calculated by using a rule-based method, taking into account NILM output, and considering additionally two fac-tors, the number of members that live in each household and average income. After the construction of user's interests and needs the user’s profile and items profile are developed. For all Ads they calculated the weighted keyword vector by implementing the term frequency/inverse document frequency technique and for user’s profile three randomly weighted keyword vector was extracted by the Ads of the corresponding category and Rocchio algorithm used to shape an average weighted keyword vector from the three Ads. The recommendation of the appropriate appliance to the user was accomplished by estimating the similarity of the appliance (item) with the user, cosine similarity method was adopted to achieve this task.



Rheem RTG-84DVLP Direct Vent Indoor Propane Tankless Water Heater Residential Tankless Water Heaters - Rheem tankless gas water heaters offer you the perfect solution to heat your water at home. Imagine never running out of hot water again. Rheem provides to you innovative, new tankless products that provide a continuous supply of hot water from energy efficient and space saving design units. Specifications - 8.4 gal./min. at 35A-Degree F rise max. - 6.6 gal./min. at 45A-Degree F rise : - New. 3"/5" Concentric Vent System with Integrated Condensate Collector - Connects to Metal Fab Inc.

The LG LCS1112SB 1.1 Cu. Ft. 1000W Countertop Microwave Oven, in black, features Eco-On energy saver and EasyClean interior. The Eco-On feature saves energy by turning off the display when the unit is not active for 5 minutes. When this feature is in use the power consumption drops below 0.1 watts, saving you money. With the EasyClean round interior and coating, you can have quick and easy cleaning without using any chemicals. Plus, one-touch controls lets you cook or warm up a variety of foods in moments for simplicity at its best.

Figure 25: Appliance recommendation for a given user [26]

3 Dataset

Non-Intrusive Load Monitoring (NILM) model performance is dependent on the quality of dataset used. One of difficulties to implement a NILM model is to find a dataset that will have the appropriate records of electrical parameters and detailed information not only of the total household consumption but also of individual appliances and from specific heavy electrical devices. Moreover, records of a satisfactory period of time should be considered since historical data are manipulated for the construction of training and test set, dependent on the chosen algorithm. A candidate researcher, except of the burden to develop the disaggregation algorithm needs to consider plan of actions to acquire the appropriate dataset. One solution is the installation of suitable sensors in the central electrical panel of a household to record total consumption and in individual appliances. Another solution is to contact electrical suppliers in order to ask for the electrical records they store which is very difficult because of the privacy issues that might arise. To tackle the problem of available datasets, several institutes and researchers have constructed datasets with the aim to extract the burden of data acquisition and open the field of NILM into candidate stakeholders of diverse sectors.

Reference Energy Disaggregation Dataset (REDD) [27] is a publicly available dataset that contains AC waveform and voltage records for 6 households. Except from the total consumption of each household they have collected data for several appliances and sub-meter circuit-level data such as (oven, refrigerator, dishwasher, kitchen outlets, lighting, washer dryer, microwave, bathroom ground fault interrupters, electric heat, stove). Data are logged at a frequency of about one second for the total consumption of the households and once every three seconds for the sub-meter circuits and individual appliances. Also, data are accompanied by their logged time stamp in the format of Universal Time Coordinated (UTC).

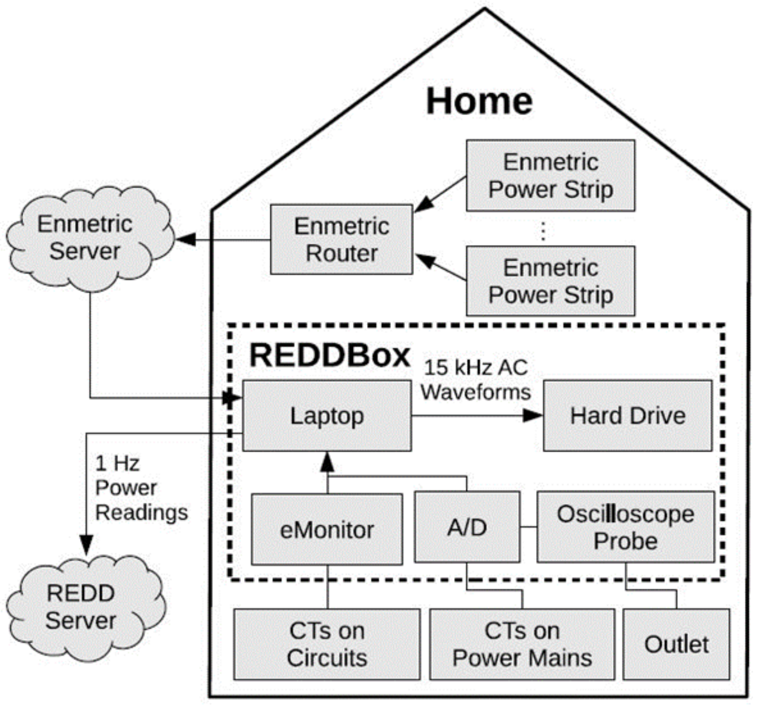


Figure 26: Architecture of REDD recording system [27]

Smart* [28] dataset comprises of two types of datasets, the first one consists of several parameters that are recorded with the use of sensors, and it is called UMass Smart* Home Data Set. In more details the UMass Smart* Home Data Set extracts data from three households where two of them are identical in size. On the first house they have installed a central meter into the main electrical panel and record the electricity data of the house, sub-meters have been installed for each of the circuits that the main panel has in order to record electricity data for each of them separately. The logged frequency of the main and the sub-circuits is once every second. Additionally, they have installed smart switches that record the state on/off or dimed for each circuit, smart plugs have been installed into the sockets of the house to record appliances that are not connected directly in the main electrical circuit of the house. Except from the electric energy, they installed sensors to record the internal temperature and humidity of the house. Motion sensors along with door sensors was installed to monitor the movement of the occupants in order to corelate that data with the time of use for each appliance. Apart from data that are produced by the house they have built a small weather station that record outside temperature, humidity, luminance etc. For the second house they have installed only meters to record the main electrical data of the house and the sub-circuits in the electrical panel. Internal and external weather data are also recorded. In third house the electrical equipment that was installed was the same as the second with the only difference that they also record power generation due to the fact that in third house it is installed two micro wind turbines and a solar panel system. The second dataset is UMass Smart* Microgrid Data Set and contains electric data from 400 houses with a log frequency of one minute.

Electric, natural gas, water, weather, and utility bills data have been collected over the period of one year in Canada and formed the AMPds [29] dataset. Electrical data have been recorded from a single household, data was collected for the whole household and for each circuit of the electrical panel and recorded every sixty seconds. After lifetime duration of the project data have been processed to drop missing values and records that have minimal consumption or infinitesimal activity.

Similarly with the above-mentioned datasets, UK Domestic Appliance-Level Electricity (UK-DALE) dataset [30] was constructed by recording electric data of total household and each sub-circuit of it. For the requirements of the project five households was used for a period ranging from 36 up to 655 days. The total records were logged at a sample rate of 16 kHz and for individual sub-circuits 1/6 Hz.

In 2013 electricity, water, and ambient parameters was recorded in a three-story building in India for a period of 73 days from May to August of 2013 creating the iAWE [31] dataset. Sensors has been used to record total consumption of the building using the main meter. For devices that are connected directly with the electric circuit installation and have their own circuit they recorded data from the main electric panel of the house. Appliances that are not connected directly with the electric installation and use plugs to power up have been monitored through the installation of smart plugs.

Table 11: Household energy datasets

Dataset Name	**Dataset Duration**	**Number of Houses**	**Appliances Sample Frequency**	**Total Sample Frequency**
REDD	3 to 19 days	6	3 sec.	1 sec.
Smart*	3 months	3	1 sec.	1 sec.
AMPds	1 year	1	1 min.	1 min.
UK-DALE	36 to 655 days	5	1/6 Hz	16 kHz
iAWE	73 days	1	1 sec.	1 sec.

4 Methodology

This chapter describes the model used for disaggregation based on public available dataset. Then a simplified recommendation system is described which incorporates the appliances used for disaggregation and different characteristics of building type and climate.

4.1 Energy Disaggregation

Based on literature review Factorial Hidden Markov Models FHMM algorithm has been widely tested and marked as an efficient method for its implementation in disaggregation models. To implement FHMM it is necessary to have consumption data of appliances and not only the total consumption of a household since FHMM is an extension of HMM and each appliance is treated as single hidden Markov model. The hidden component of these HMMs are the states of the appliances. Energy disaggregation involves jointly decoding the power draw of n appliances and hence a factorial HMM is well suited [15]. A FHMM can be represented by an equivalent HMM in which each state corresponds to a different combination of states of each appliance. A FHMM model has three parameters:

i. prior probability (π) containing K^N entries
ii. transition matrix (A) containing K^N x K^N or K^{2N} entries
iii. emission matrix (B) containing $2K^N$ entries

The combination of each HMM constitutes the final FHMM model.

FHMM can be implemented for disaggregation through the use of non-intrusive load monitoring toolkit, a publicly available python library that has been developed [15] with the concept to be used as a benchmark algorithm. Except from disaggregation part the toolkit provides basic statistics functionalities on the dataset. The only drawback of the toolkit is the ease of use since the end user must devote considerable time to learn it and start taking results. For that reason, in this work we used the library developed in [16] which has been developed based on NILMTK [15], but only incorporates the disaggregation part excluding all other functionalities making it very user friendly.

For our disaggregation we used iAWE [31] dataset because it contains a wealth of data from the appliances of the building. The dataset contains consumption of 10 appliances namely two air conditioners, refrigerator, washing machine, kitchen outlets, television, iron, laptop, water filter, water motor. For each device and also for the total consumption has been recorded different electrical parameters such us Watts (W), Reactive Power (VAR), Apparent Power (VA), Frequency (F), Phase Voltage (VLN), Power Factor (PF) and Amber (A). From the electric parameters we choose to work with Watts since this parameter can be easily recorded from diverse electric meters and smart plugs. From these appliances only the fridge, iron, air conditioner and washing machine were used for the disaggregation because they have the heaviest consumption for the specific household.

Table 12: Consumption of the 4 heaviest appliances [31]

	Fridge	**AC**	**Washing Machine**	**Iron**
Consumption in Watt	270113625	689668541	4850056	2606377

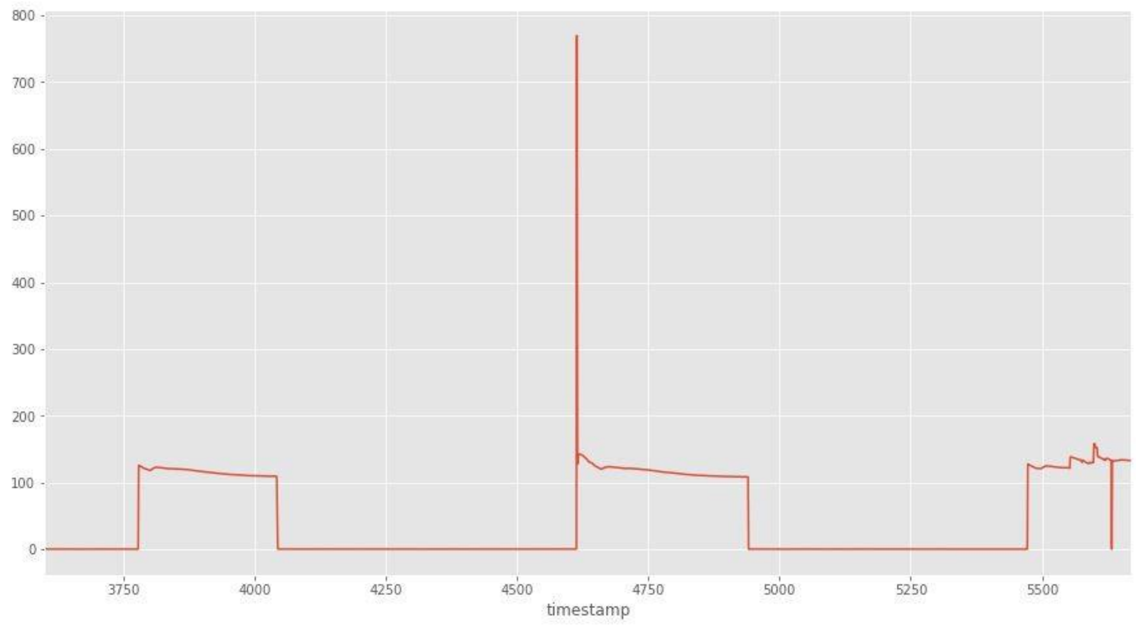


Figure 27: Fridge consumption [31]

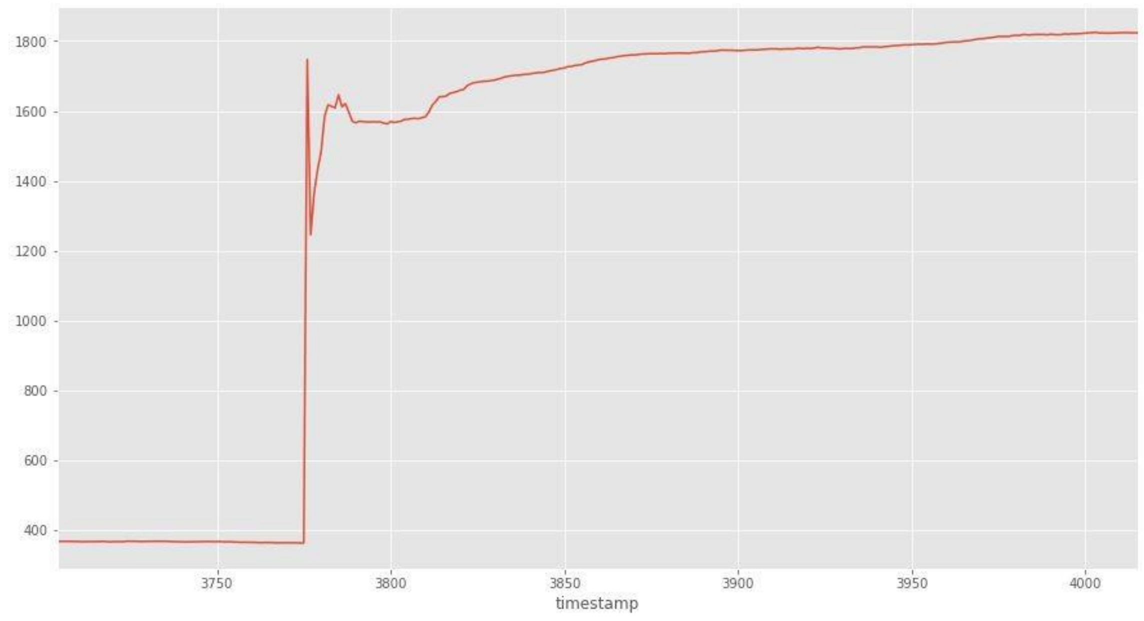


Figure 18: AC consumption [31]

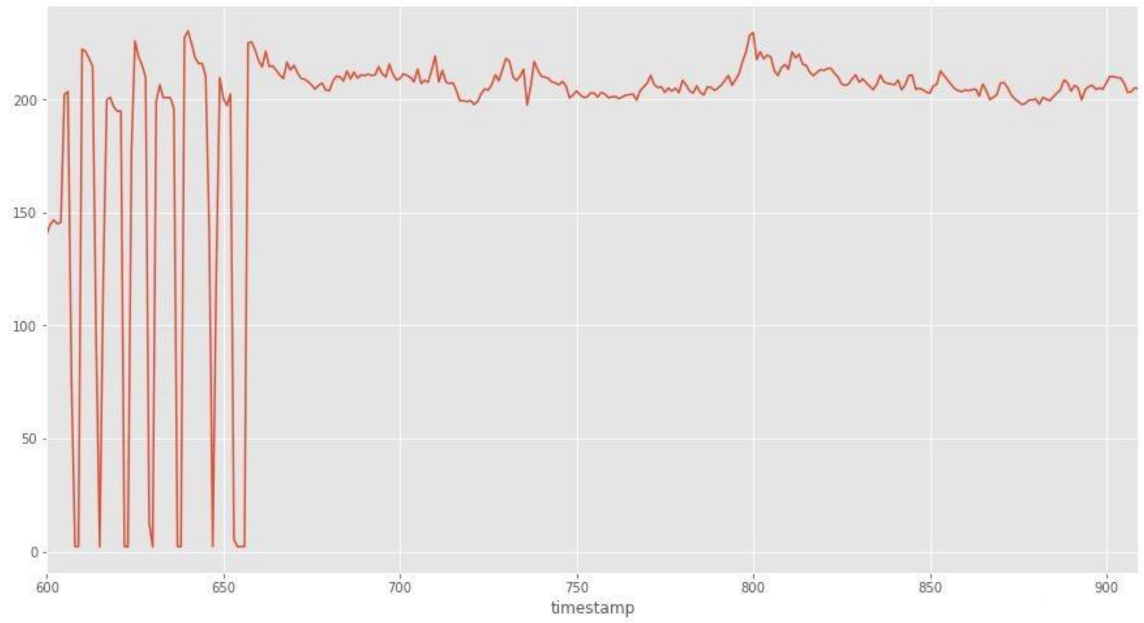


Figure 29: Washing Machine consumption [31]

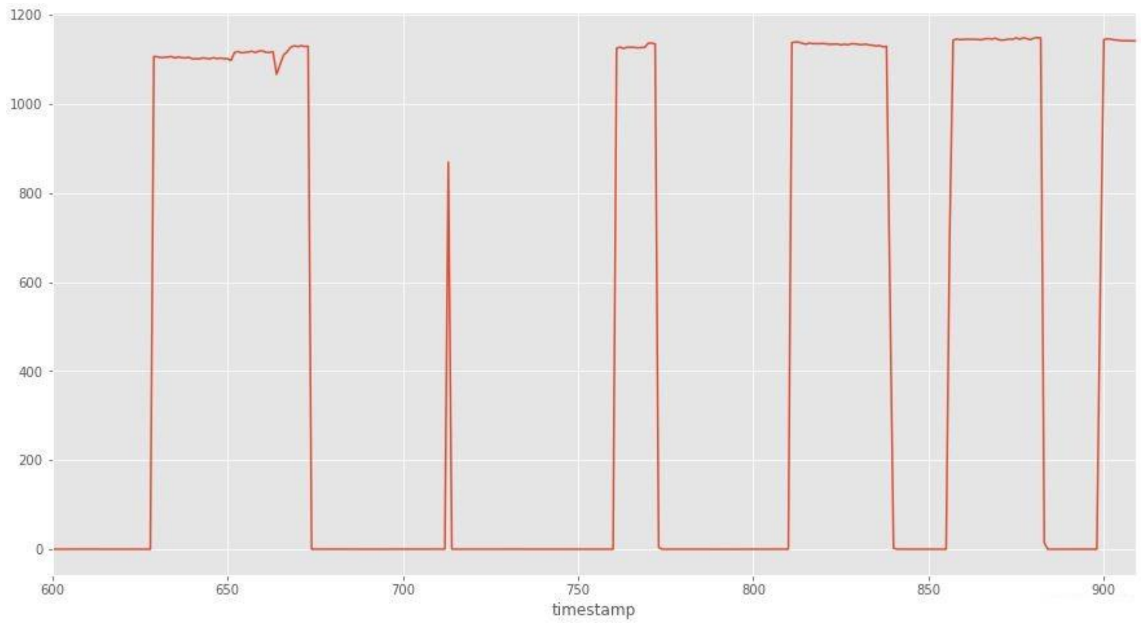


Figure 30: Iron consumption [31]

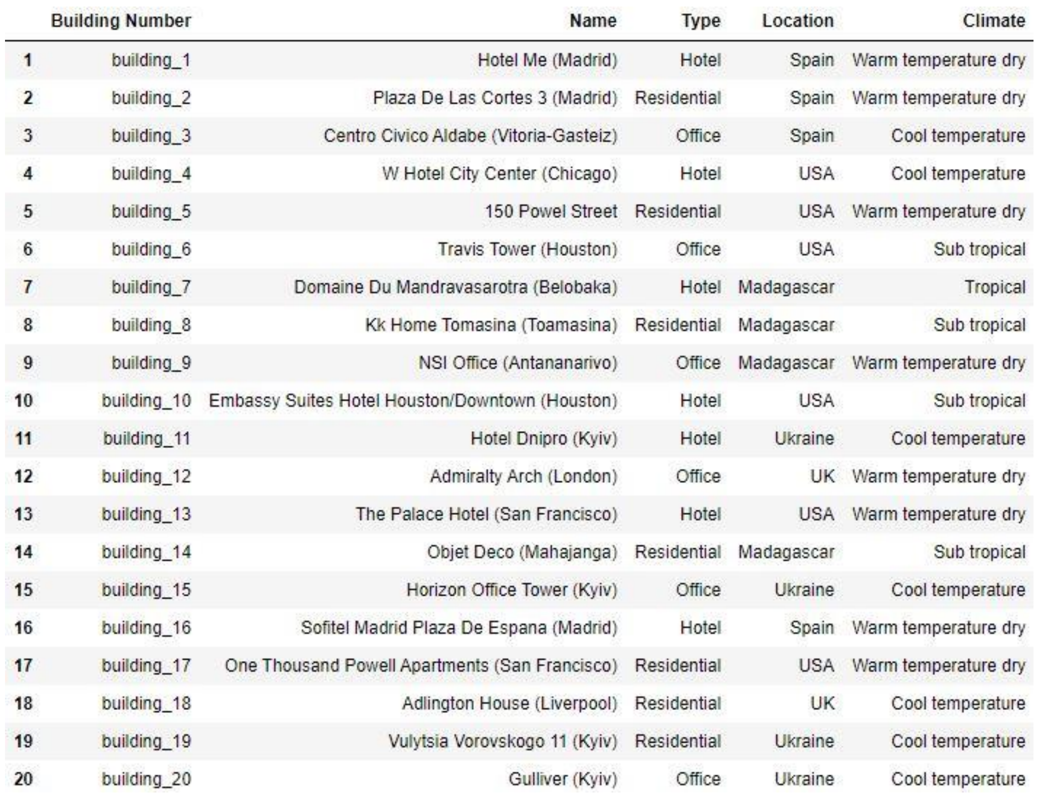
The frequency of data collection is one second for the appliances and the main building consumption. The datasets have been split into two parts, the first one has been used to train the model and the second one used to test it.

The implementation was made through jupyter notebook under anaconda package since it contains all the required software tools for the library [16].

4.2 Recommendation System

Recommendation systems generally rely on similarity between target user and similar other users or the similarity between the preferences of a specific user. To construct a recommendation system, ratings of appliances are of high importance since the recommendation that will be provided to the end user will take into account the appliances used during the disaggregation. To identify a dataset that will have ratings based on the appliance's consumption was very challenging and for that reason the characteristics that was used to construct recommendations are the type of the building and the climate since they influence the operation pattern of the appliances. The dataset used [32] contains information's for 20 buildings such as their location, type, and climate. For each building ratings for 7 appliances are included, these appliances are refrigerator, lights, microwave, stove, air conditioner, washing machine and water heater.

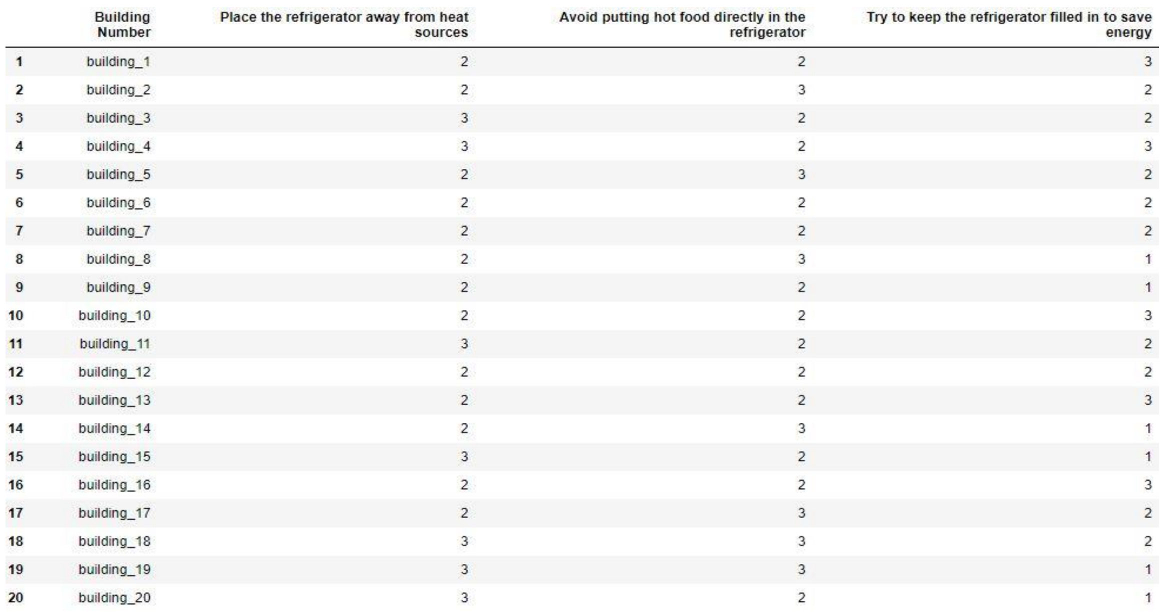
Table 13: Buildings data [32]



	Building Number	Name	Type	Location	Climate
1	building_1	Hotel Me (Madrid)	Hotel	Spain	Warm temperature dry
2	building_2	Plaza De Las Cortes 3 (Madrid)	Residential	Spain	Warm temperature dry
3	building_3	Centro Civico Aldabe (Vitoria-Gasteiz)	Office	Spain	Cool temperature
4	building_4	W Hotel City Center (Chicago)	Hotel	USA	Cool temperature
5	building_5	150 Powel Street	Residential	USA	Warm temperature dry
6	building_6	Travis Tower (Houston)	Office	USA	Sub tropical
7	building_7	Domaine Du Mandravasarotra (Belobaka)	Hotel	Madagascar	Tropical
8	building_8	Kk Home Tomasina (Toamasina)	Residential	Madagascar	Sub tropical
9	building_9	NSI Office (Antananarivo)	Office	Madagascar	Warm temperature dry
10	building_10	Embassy Suites Hotel Houston/Downtown (Houston)	Hotel	USA	Sub tropical
11	building_11	Hotel Dnipro (Kyiv)	Hotel	Ukraine	Cool temperature
12	building_12	Admiralty Arch (London)	Office	UK	Warm temperature dry
13	building_13	The Palace Hotel (San Francisco)	Hotel	USA	Warm temperature dry
14	building_14	Objet Deco (Mahajanga)	Residential	Madagascar	Sub tropical
15	building_15	Horizon Office Tower (Kyiv)	Office	Ukraine	Cool temperature
16	building_16	Sofitel Madrid Plaza De Espana (Madrid)	Hotel	Spain	Warm temperature dry
17	building_17	One Thousand Powell Apartments (San Francisco)	Residential	USA	Warm temperature dry
18	building_18	Adlington House (Liverpool)	Residential	UK	Cool temperature
19	building_19	Vulytsia Vorovskogo 11 (Kyiv)	Residential	Ukraine	Cool temperature
20	building_20	Gulliver (Kyiv)	Office	Ukraine	Cool temperature

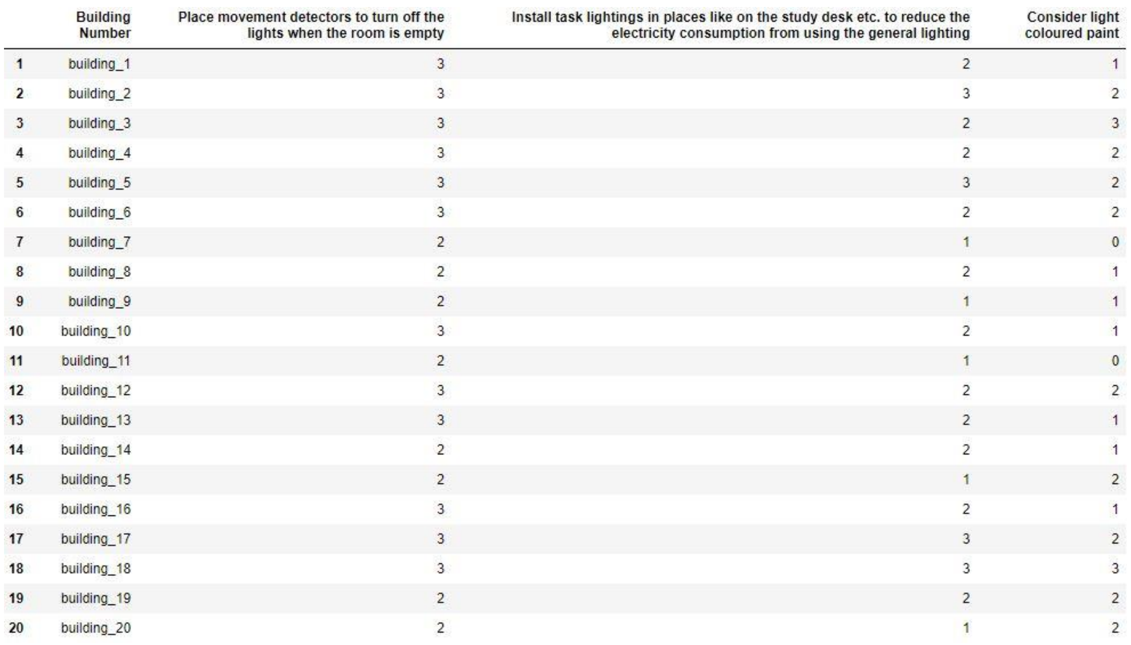
The building used to construct the iAWE dataset is a residential building located in India and more specifically in Delhi. The climate of India is characterized as Sub tropical. Those two characteristics was used to identify the most relevant buildings between the disaggregated building and the buildings data (Table 14). After the identification of the most similar buildings their ratings was extracted based on the building number field.

Table 14: Fridge ratings [32]



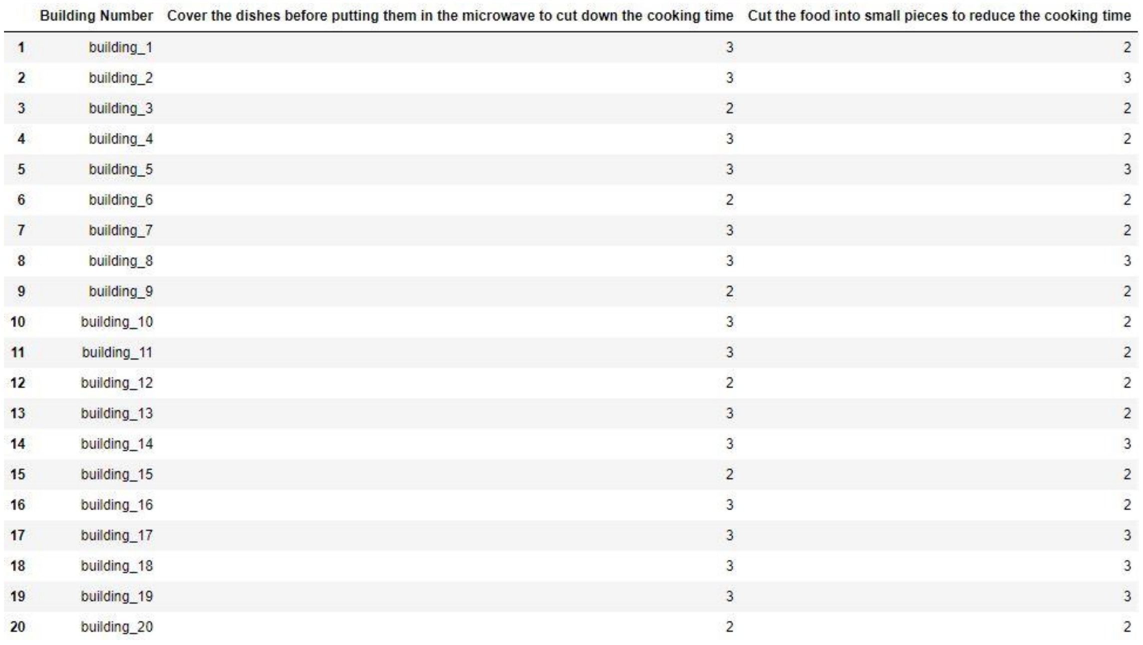
	Building Number	Place the refrigerator away from heat sources	Avoid putting hot food directly in the refrigerator	Try to keep the refrigerator filled in to save energy
1	building_1	2	2	3
2	building_2	2	3	2
3	building_3	3	2	2
4	building_4	3	2	3
5	building_5	2	3	2
6	building_6	2	2	2
7	building_7	2	2	2
8	building_8	2	3	1
9	building_9	2	2	1
10	building_10	2	2	3
11	building_11	3	2	2
12	building_12	2	2	2
13	building_13	2	2	3
14	building_14	2	3	1
15	building_15	3	2	1
16	building_16	2	2	3
17	building_17	2	3	2
18	building_18	3	3	2
19	building_19	3	3	1
20	building_20	3	2	1

Table 15: Light ratings [32]



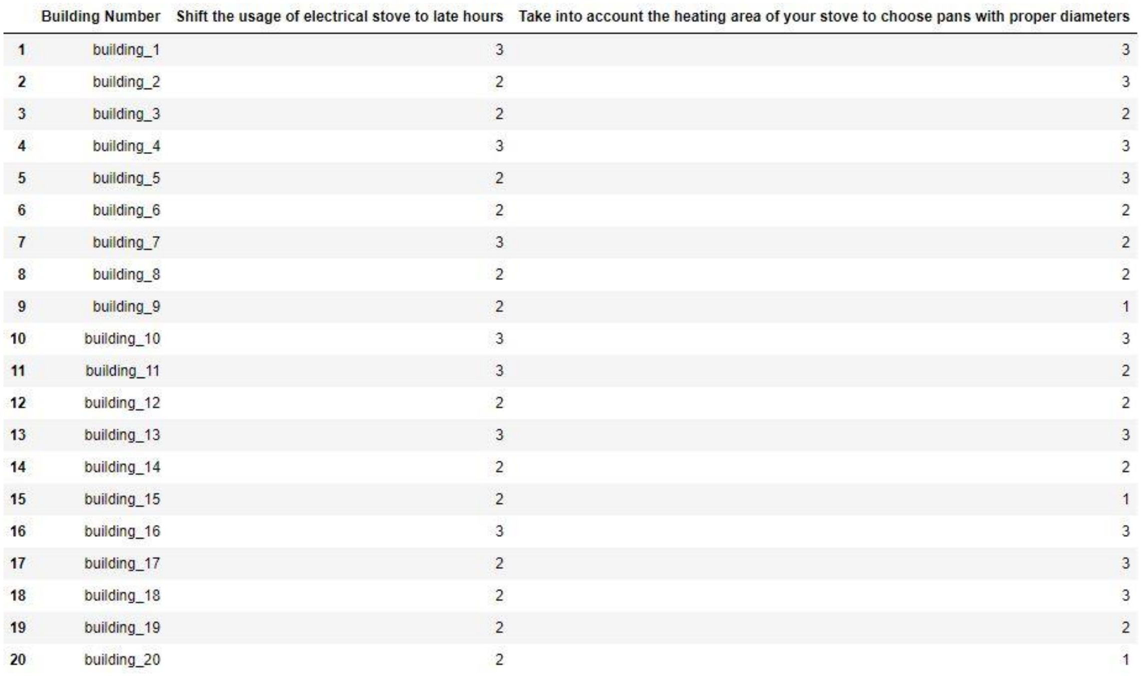
	Building Number	Place movement detectors to turn off the lights when the room is empty	Install task lightings in places like on the study desk etc. to reduce the electricity consumption from using the general lighting	Consider light coloured paint
1	building_1	3	2	1
2	building_2	3	3	2
3	building_3	3	2	3
4	building_4	3	2	2
5	building_5	3	3	2
6	building_6	3	2	2
7	building_7	2	1	0
8	building_8	2	2	1
9	building_9	2	1	1
10	building_10	3	2	1
11	building_11	2	1	0
12	building_12	3	2	2
13	building_13	3	2	1
14	building_14	2	2	1
15	building_15	2	1	2
16	building_16	3	2	1
17	building_17	3	3	2
18	building_18	3	3	3
19	building_19	2	2	2
20	building_20	2	1	2

Table 16: Microwave ratings [32]



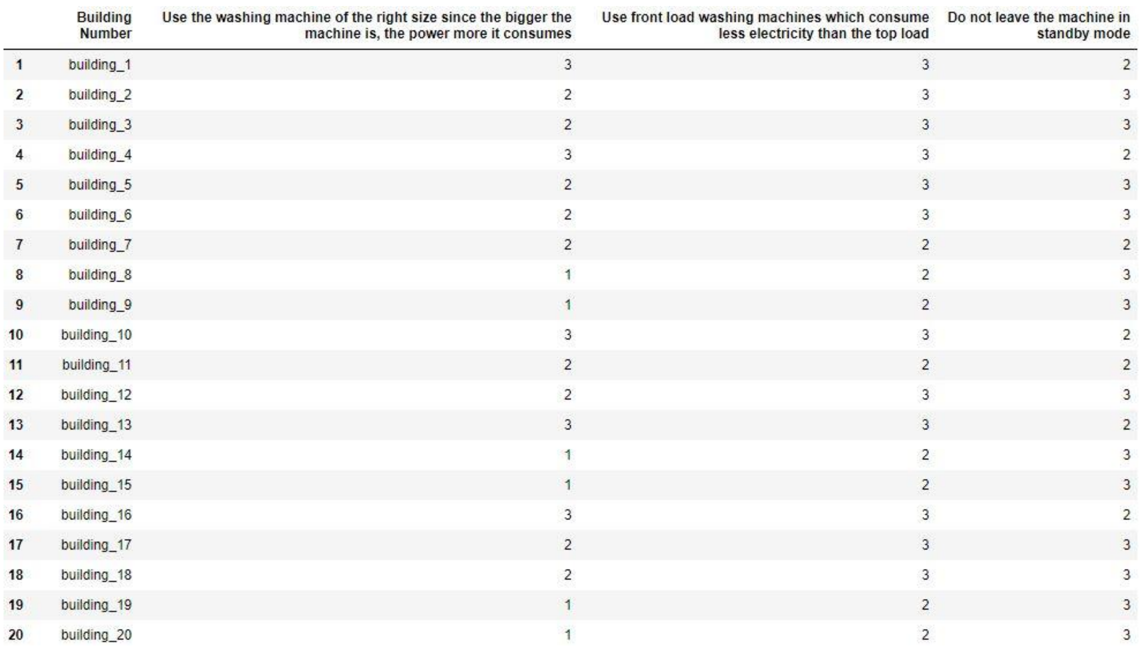
	Building Number	Cover the dishes before putting them in the microwave to cut down the cooking time	Cut the food into small pieces to reduce the cooking time
1	building_1	3	2
2	building_2	3	3
3	building_3	2	2
4	building_4	3	2
5	building_5	3	3
6	building_6	2	2
7	building_7	3	2
8	building_8	3	3
9	building_9	2	2
10	building_10	3	2
11	building_11	3	2
12	building_12	2	2
13	building_13	3	2
14	building_14	3	3
15	building_15	2	2
16	building_16	3	2
17	building_17	3	3
18	building_18	3	3
19	building_19	3	3
20	building_20	2	2

Table 17: Stove ratings [32]



	Building Number	Shift the usage of electrical stove to late hours	Take into account the heating area of your stove to choose pans with proper diameters
1	building_1	3	3
2	building_2	2	3
3	building_3	2	2
4	building_4	3	3
5	building_5	2	3
6	building_6	2	2
7	building_7	3	2
8	building_8	2	2
9	building_9	2	1
10	building_10	3	3
11	building_11	3	2
12	building_12	2	2
13	building_13	3	3
14	building_14	2	2
15	building_15	2	1
16	building_16	3	3
17	building_17	2	3
18	building_18	2	3
19	building_19	2	2
20	building_20	2	1

Table 18: Washing Machine ratings [32]



	Building Number	Use the washing machine of the right size since the bigger the machine is, the power more it consumes	Use front load washing machines which consume less electricity than the top load	Do not leave the machine in standby mode
1	building_1	3	3	2
2	building_2	2	3	3
3	building_3	2	3	3
4	building_4	3	3	2
5	building_5	2	3	3
6	building_6	2	3	3
7	building_7	2	2	2
8	building_8	1	2	3
9	building_9	1	2	3
10	building_10	3	3	2
11	building_11	2	2	2
12	building_12	2	3	3
13	building_13	3	3	2
14	building_14	1	2	3
15	building_15	1	2	3
16	building_16	3	3	2
17	building_17	2	3	3
18	building_18	2	3	3
19	building_19	1	2	3
20	building_20	1	2	3

Table 19: Water Heater ratings [32]

	Building Number	Insulate the pipes connected to the heater	Prefer taking a sort shower instead of a bath	Consider installing heat traps on the water heater
1	building_1	2	3	2
2	building_2	2	3	2
3	building_3	3	2	3
4	building_4	3	3	3
5	building_5	2	3	2
6	building_6	2	2	2
7	building_7	2	3	1
8	building_8	2	3	1
9	building_9	2	2	1
10	building_10	2	3	2
11	building_11	3	3	2
12	building_12	2	2	2
13	building_13	2	3	2
14	building_14	2	3	1
15	building_15	3	2	2
16	building_16	2	3	2
17	building_17	2	3	2
18	building_18	3	3	3
19	building_19	3	3	2
20	building_20	3	2	2

Table 20: Air Conditioner ratings [32]

	Building Number	Keep the curtains and blinds closed to reduce the space from heating up	Use a programmable thermostat that turns off the AC when the space is empty
1	building_1	2	3
2	building_2	3	3
3	building_3	1	3
4	building_4	1	3
5	building_5	3	3
6	building_6	2	3
7	building_7	2	2
8	building_8	3	2
9	building_9	2	2
10	building_10	2	3
11	building_11	1	2
12	building_12	2	3
13	building_13	2	3
14	building_14	3	2
15	building_15	1	2
16	building_16	2	3
17	building_17	3	3
18	building_18	2	3
19	building_19	2	2
20	building_20	1	2

The next step is to compare the appliances that was used for the disaggregation with the appliances that we have the ratings and take out the appliances that does not pair. Final part was to calculate the average rating for each recommendation category and extract only the recommendation with the highest value for each appliance.

5 Results

In this section will be illustrated the output of the disaggregation method alongside with the output of the recommendation system.

5.1 Energy Disaggregation

After the iAWE dataset has been fed into the NILM algorithm the prediction of the appliance's consumption has been calculated. In Fig. 31 is illustrated the consumption of the four appliances before the disaggregation. The biggest share of consumption is made by air conditioner, and it is very logical considering that the period the data extracted is between May and August, which is summer period.

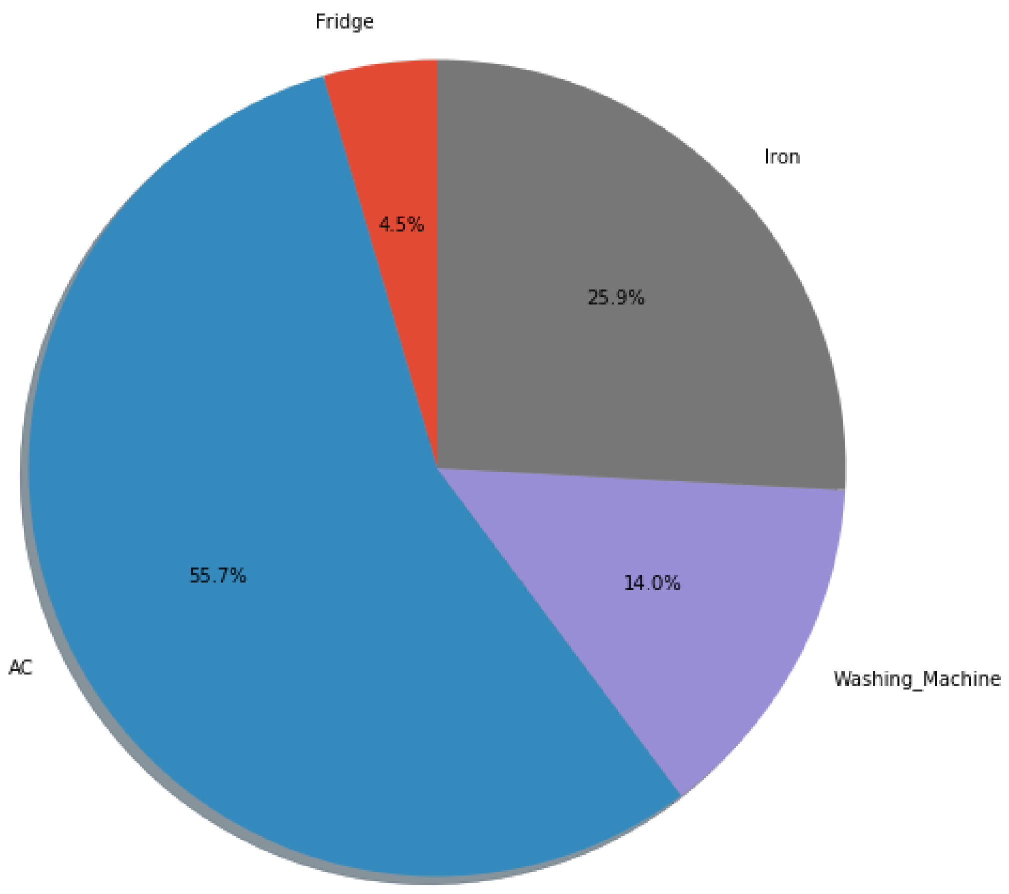


Figure 31: Ground truth of appliances

In figure 32 is illustrated the appliances predictions, as can be seen the model is very accurate for the fridge and washing machine having a deviation from the ground truth outmost to 3,5 percent. The major difference can be seen for air conditioner and iron. For the case of the air conditioner there is an increase while for the iron can be seen a decrease. Observing the predictions can be seen that the algorithm has misclassified the air conditioner with the iron. This large gap can be explained if we assume on average a medium air conditioner has between 1100 to 1300 watts and an average iron has 900 to 1200 watts. Since both appliances share compatible characteristics with respect to watts the algorithm could not classify them accurately.

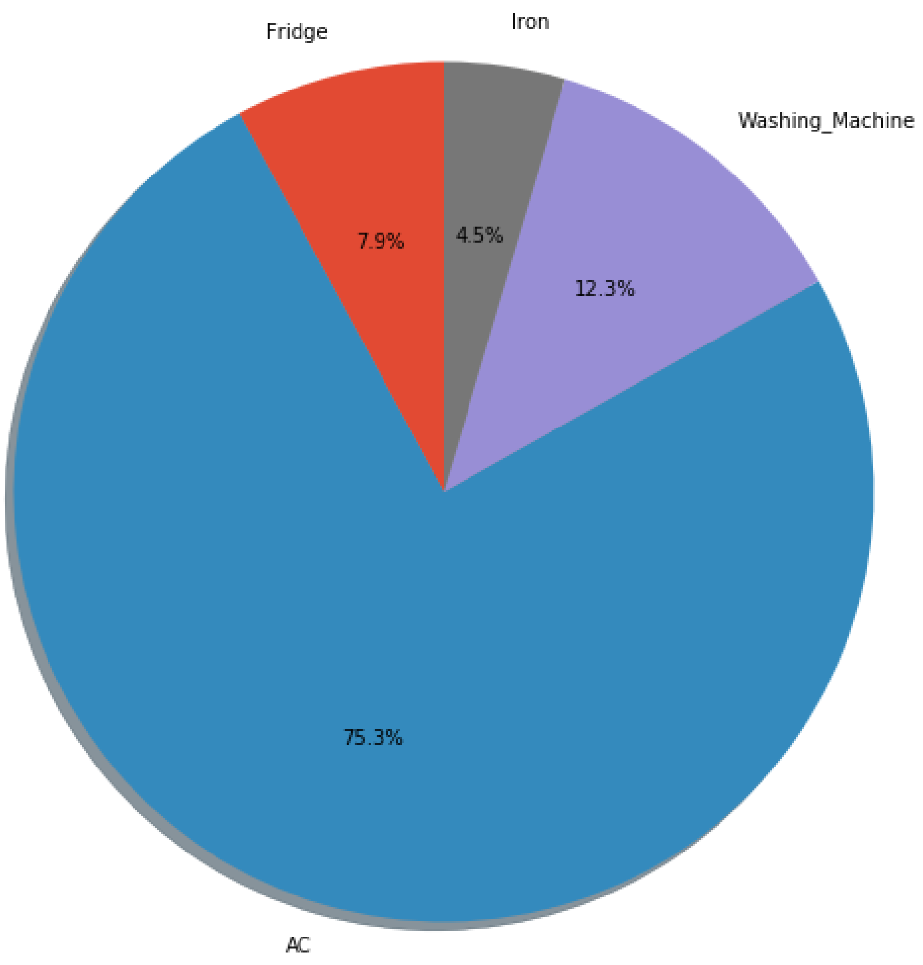


Figure 32: Prediction of appliances

For appliances that does not share same electrical characteristics the algorithm is performing at satisfactory levels considering that the difference of the ground truth with the predicted values of fridge and washing machine does not exceed 2%. Since a typical household is constituted by diverse appliances that does not share the same electrical characteristics the algorithm will correctly classify most of the appliances. The drawback is only for appliances that have same electrical characteristics and can be seen from the predictions of air-condition and iron were the algorithm has misclassified their consumption leading to poor performance.

5.2 Recommendation System

Next, after the completion of disaggregation, the building characteristics were used alongside with the appliances fed into the NILM algorithm. More specifically to extract identical buildings was use the type which is residential and the general climate of India which is subtropical. Then the appliances that have ratings have been filtered with the appliances used during disaggregation and remove those that does not match.

Table 21: Appliances used to extract recommendations

Appliances Ratings	Fridge	Microwave	Lights	Stove	AC	Washing Machine	Water heater	-
Disaggregation Appliances	Fridge	-	-	-	AC	Washing Machine	-	Iron

As can be seen from table 22 recommendations was provided only for fridge, air conditioner and washing machine. The recommendations for all the appliances are 18, table 23 illustrates the recommendations for each appliance.

Table 22: Recommendations [32]

Appliances	**Recommendations**
Fridge	Place the refrigerator away from heat sources.
	Avoid putting hot food directly in the refrigerator.
	Try to keep the refrigerator filled in to save energy.
Microwave	Cover the dishes before putting them in the microwave to cut down the cooking time.
	Cut the food into small pieces to reduce the cooking time.

Lights	Place movement detectors to turn off the lights when the room is empty.
	Install task lightings in places like on the study desk etc. to reduce the electricity consumption from using the general lighting.
	Consider light colored paint.
Stove	Shift the usage of electrical stove to late hours.
	Take into account the heating area of your stove to choose pans with proper diameters.
Air conditioner	Keep the curtains and blinds closed to reduce the space from heating up.
	Use a programmable thermostat that turns off the AC when the space is empty.
Washing Machine	Use the washing machine of the right size since the bigger the machine is, the power more it consumes.
	Use front load washing machines which consume less electricity than the top load.
	Do not leave the machine in standby mode.
Water Heater	Insulate the pipes connected to the heater.
	Prefer taking a sort shower instead of a bath.
	Consider installing heat traps on the water heater

Last part of the simplified recommendation system was to calculate the average rating for each recommendation and extract only the recommendation with the highest value. After the calculations our method identifies the row based on index where the average ratings have been stored and extracts the recommendation with the maximum value, if in a row exist recommendations with the same score then we choose the first of the maximum scores, thus the expected output must be three recommendations one for each appliance for fridge, air conditioner and washing machine.

In table 24 can be seen the out of the algorithm for the three appliances with their corresponding score.

Table 23: Output recommendations of the model

Appliance	Recommendation	Score
Fridge	Avoid putting hot food directly in the refrigerator.	3
AC	Keep the curtains and blinds closed to reduce the space from heating up.	3
Washing Machine	Do not leave the machine in standby mode.	3

The recommendation system performed as was expected since recommendation have been provided only for the appliances that was used in the disaggregation. Except from accurate appliance filtering only the methods with the highest score were provided.

6 Conclusions

The disaggregated method presented in this dissertation can be considered highly accurate for appliances that do not share the same characteristics, like fridge, electric stove, dishwasher, and electric boiler which have different power consumption (250 Watts, 2000 Watts, 1350 Watts, and 5000 watts respectively). As they have totally different power consumption, the algorithm will efficiently classify them, even if they operate at the same time. In the model we used the fridge and washing machine were classified accurately as they have different power consumption.

Contrariwise for the iron and the air conditioner disaggregation did not perform well. Evaluating predictions against the ground truth we see 20% for these appliances. This can be explained by the compatibility in power consumption (ranging from 900 to 1200 watts and from 1100 to 1300 watts, respectively). Furthermore, if usage times for the two appliances coincide, an extra burden is added on the algorithm to classify them with a small deviation from ground truth values.

A solution to this weakness could be the use of multiple or different type of electrical parameters. In the literature the usage of current harmonics has been examined [12]. Current harmonics could be different for appliances that have the same electric parameters leading to better classification predictions.

A final drawback of the algorithm proposed can be considered that it requires the recordings of each appliance that the user wants to disaggregate. From one point this could be considered as beneficial, since the algorithm does not have to identify the number of the appliances used, but it is difficult to acquire consumption data for each appliance.

Regarding the recommendation system, it performed efficiently, providing only the recommendations with the highest score for each appliance that was used in the disaggregation. More specifically the disaggregation is performed by using the Factorial Hidden Markov model, then based on the building type and climatic condition of the disaggregated building a filtering is applied in order to extract buildings with the same characteristics. A second layer filtering then take place to extract from the appliances used for the disaggregation only those for which recommendation are available based on the recommendation dataset. Then for each recommendation the average score is calculated and extracted only the recommendations with the highest score for each appliance.

A potential drawback is its simplicity, stemming from the difficulty to find datasets based on individual appliance consumption. Moreover, if it was possible to use a well-constructed dataset with appliance ratings based on their consumption, then a more sophisticated recommendation filtering could be implemented to predict and extract ratings for appliances, such as model and item based collaborative filtering [23] [33] [34].

The result of this study is pointing out that the collaboration of energy disaggregation and recommendation systems can have a strong impact in our economy by minimizing energy expenses of a household decreasing the impact of energy crisis that all households are facing due to the high electricity prices and at the same time minimizing the emissions of carbon dioxide reducing the footprint of each house in the environment.

Bibliography

[1] F. Malik and A. Shah, “Smart City: A Roadmap Towards Implementation.”

[2] “Data Collection - Voltaware sensors.” https://voltaware.com/our-technology/voltaware-sensors (accessed Oct. 16, 2021).

[3] “Socket — Wireless smart plug with energy monitor | Ajax Systems.” https://ajax.systems/products/socket/ (accessed Oct. 16, 2021).

[4] G. Tang, J. Chen, C. Chen, and K. Wu, “Smart saver: A consumer-oriented web service for energy disaggregation,” in *IEEE International Conference on Data Mining Workshops, ICDMW*, Jan. 2015, vol. 2015-January, no. January, pp. 1235–1238. doi: 10.1109/ICDMW.2014.19.

[5] K. D. Anderson, M. E. Berges, A. Ocneanu, D. Benitez, and J. M. F. Moura, “Event detection for Non Intrusive load monitoring,” in *IECON Proceedings (Industrial Electronics Conference)*, 2012, pp. 3312–3317. doi: 10.1109/IECON.2012.6389367.

[6] M. Lu and Z. Li, “A Hybrid Event Detection Approach for Non-Intrusive Load Monitoring,” *IEEE Transactions on Smart Grid*, vol. 11, no. 1, pp. 528–540, Jan. 2020, doi: 10.1109/TSG.2019.2924862.

[7] A. U. Rehman, T. T. Lie, B. Valles, and S. R. Tito, “Event-Detection Algorithms for Low Sampling Nonintrusive Load Monitoring Systems Based on Low Complexity Statistical Features,” *IEEE Transactions on Instrumentation and Measurement*, vol. 69, no. 3, pp. 751–759, Mar. 2020, doi: 10.1109/TIM.2019.2904351.

[8] IEEE Signal Processing Society, *2017 IEEE International Conference on Acoustics, Speech and Signal Processing : proceedings : March 5-9, 2017, Hilton New Orleans Riverside, New Orleans, Louisiana, USA.*

[9] T. Liu, X. Ding, and N. Gu, “A Generic Energy Disaggregation Approach: What and When Electrical Appliances are Used,” in *Proceedings - 15th IEEE International Conference on Data Mining Workshop, ICDMW 2015*, Jan. 2016, pp. 389–397. doi: 10.1109/ICDMW.2015.28.

[10] F. Jakab, IEEE Czechoslovakia Section, and Institute of Electrical and Electronics Engineers, *ICETA 2019 : 17th IEEE International Conference on Emerging eLearning Technologies and Applications : proceedings : November 21-22, 2019, Starý Smokovec, the High Tatras, Slovakia*.

[11] S. Bhattacharjee, A. Kumar, and J. Roychowdhury, "Appliance classification using energy disaggregation in smart homes," in *2014 International Conference on Computation of Power, Energy, Information and Communication, ICCPEIC 2014*, Oct. 2014, pp. 1–6. doi: 10.1109/ICCPEIC.2014.6915330.

[12] P. Tao, X. Liu, Y. Zhang, C. Li, and J. DIng, "Multi-level non-intrusive load identification based on k-NN," in *2019 3rd IEEE Conference on Energy Internet and Energy System Integration: Ubiquitous Energy Network Connecting Everything, EI2 2019*, Nov. 2019, pp. 1905–1910. doi: 10.1109/EI247390.2019.9061896.

[13] M. Azaza and F. Wallin, "Supervised household's loads pattern recognition," Dec. 2016. doi: 10.1109/EPEC.2016.7771718.

[14] M. M. R. Khan, M. A. B. Siddique, and S. Sakib, "Non-Intrusive Electrical Appliances Monitoring and Classification using K-Nearest Neighbors," Dec. 2019. doi: 10.1109/ICIET48527.2019.9290671.

[15] N. Batra *et al.*, "NILMTK: An open source toolkit for non-intrusive load monitoring," in *e-Energy 2014 - Proceedings of the 5th ACM International Conference on Future Energy Systems*, 2014, pp. 265–276. doi: 10.1145/2602044.2602051.

[16] C. Electrical Engineering/Electronics, IEEE Thailand Section., Institute of Electrical and Electronics Engineers, and T. ; O. ECTI-NCON (Conference) (3rd : 2020 : Ban Phatthaya, *DAMT & NCON 2020 : International Conference on Digital Arts, Media and Technology (DAMT) and ECTI Northern Section Conference on Electrical, Electronics, Computer and Telecommunications Engineering (NCON) : 11-14 March 2020 at Pattaya,Thailand.*

[17] F. Gong, C. Liu, L. Jiang, H. Li, J. Y. Lin, and B. Yin, "Load Disaggregation in Non-Intrusive Load Monitoring Based on Random Forest Optimized by Particle Swarm Optimization."

[18] Y. Wang, J. Sun, Q. Sun, R. Wennersten, and J. L. Scartezzini, “Office Appliance Data Classification Based on Non-intrusive Load Monitoring,” in *2020 3rd International Conference on Power and Energy Applications, ICPEA 2020,* Oct. 2020, pp. 113–116. doi: 10.1109/ICPEA49807.2020.9280121.

[19] M. Nguyen, S. Alshareef, A. Gilani, and W. G. Morsi, “A novel feature extraction and classification algorithm based on power components using single-point monitoring for NILM,” in *Canadian Conference on Electrical and Computer Engineering,* Jun. 2015, vol. 2015-June, no. June, pp. 37–40. doi: 10.1109/CCECE.2015.7129156.

[20] Y. Gao and H. Yang, “Non-intrusive load identification by fuzzy cluster analysis based on active power,” 2012. doi: 10.1109/APPEEC.2012.6307566.

[21] M. Palaniswami, Annual IEEE Computer Conference, S. N. and I. P. 8 2013. 04. 02-05 M. IEEE International Conference on Intelligent Sensors, ISSNIP 8 2013.04.02-05 Melbourne, A. and S. 2013. 04. 02-05 M. Workshop on RFID Technology, and Workshop on Sensor Fusion and Tracking 2013.04.02-05 Melbourne, *IEEE Eighth International Conference on Intelligent Sensors, Sensor Networks and Information Processing ISSNIP 2013 ; 2-5 April 2013, Melbourne, Australia ; [including workshop papers].*

[22] I. D. of E. E. National Institute of Technology (Raipur, Institute of Electrical and Electronics Engineers. Bombay Section, I. I. S. B. National Institute of Technology (Raipur, and Institute of Electrical and Electronics Engineers, *2020 First International Conference on Power, Control and Computing Technologies (ICPC2T) : 03 -05 January 2020, Department of Electrical Engineering, National Institute of Technology, Raipur G.E. Road, Raipur, Chhatisgarh-492010, India.*

[23] Y. Zhang, K. Meng, W. Kong, and Z. Y. Dong, “Collaborative Filtering-Based Electricity Plan Recommender System,” *IEEE Transactions on Industrial Informatics,* vol. 15, no. 3, pp. 1393–1404, Mar. 2019, doi: 10.1109/TII.2018.2856842.

[24] F. Luo, G. Ranzi, W. Kong, G. Liang, and Z. Y. Dong, “Personalized Residential Energy Usage Recommendation System Based on Load Monitoring and Collaborative Filtering,” *IEEE Transactions on Industrial Informatics,* vol. 17, no. 2, pp. 1253–1262, Feb. 2021, doi: 10.1109/TII.2020.2983212.

[25] D. M. Jiménez-Bravo, J. Pérez-Marcos, D. H. de La Iglesia, G. V. González, and J. F. de Paz, "Multi-agent recommendation system for electrical energy optimization and cost saving in smart homes," *Energies*, vol. 12, no. 7, 2019, doi: 10.3390/en12071317.

[26] F. Luo, G. Ranzi, W. Kong, Z. Y. Dong, S. Wang, and J. Zhao, "Non-intrusive energy saving appliance recommender system for smart grid residential users," *IET Generation, Transmission and Distribution*, vol. 11, no. 7, pp. 1786–1793, May 2017, doi: 10.1049/iet-gtd.2016.1615.

[27] J. Z. Kolter and M. J. Johnson, *REDD: A Public Data Set for Energy Disaggregation Research*. 2011. [Online]. Available: http://www.picotechnologies.com

[28] S. Barker, A. Mishra, E. Cecchet, D. Irwin, P. Shenoy, and J. Albrecht, *Smart*: An Open Data Set and Tools for Enabling Research in Sustainable Homes*. 2012. [Online]. Available: https://www.researchgate.net/publication/266888035

[29] S. Makonin, B. Ellert, I. v. Bajić, and F. Popowich, "Electricity, water, and natural gas consumption of a residential house in Canada from 2012 to 2014," *Scientific Data*, vol. 3, Jun. 2016, doi: 10.1038/sdata.2016.37.

[30] J. Kelly and W. Knottenbelt, "The UK-DALE dataset, domestic appliance-level electricity demand and whole-house demand from five UK homes," *Scientific Data*, vol. 2, Mar. 2015, doi: 10.1038/sdata.2015.7.

[31] Rasit. Eskicioglu, *Proceedings of the 10th ACM Conference on Embedded Network Sensor Systems*. ACM, 2012.

[32] Y. Al-Dara, "Electricity usage recommender system with limited input data A Master Thesis submitted for the Erasmus Mundus Joint Master Degree on Smart Cities and Communities (SMACCs)," 2021.

[33] G. Linden, B. Smith, and J. York, "Amazon.com Recommendations," IEEE Computer Society. [Online]. Available: http://computer.org/internet/

[34] C. A. Gomez-Uribe and N. Hunt, "The netflix recommender system: Algorithms, business value, and innovation," *ACM Transactions on Management Information Systems*, vol. 6, no. 4, Dec. 2015, doi: 10.1145/2843948.

Appendix 1

Appendix 1 provides the python code that was used to implement the disaggregation model. The disaggregation algorithm has been taken completely without any change from [16] which is publicly available on GitHub, the programming language of the algorithm is Python.

```python
import itertools
from copy import deepcopy
from collections import OrderedDict
from warnings import warn
import pickle


import pandas as pd
import numpy as np
from hmmlearn import hmm


from six import iteritems
from builtins import range

SEED = 42

np.random.seed(SEED)

def sort_startprob(mapping, startprob):
    num_elements = len(startprob)
    new_startprob = np.zeros(num_elements)
    for i in range(len(startprob)):
        new_startprob[i] = startprob[mapping[i]]
    return new_startprob
```

```
def sort_covars(mapping, covars):

    new_covars = np.zeros_like(covars)

    for i in range(len(covars)):

        new_covars[i] = covars[mapping[i]]

    return new_covars




def sort_transition_matrix(mapping, A):
    num_elements = len(A)
    A_new = np.zeros((num_elements, num_elements))
    for i in range(num_elements):
        for j in range(num_elements):
            A_new[i, j] = A[mapping[i], mapping[j]]
    return A_new



def sort_learnt_parameters(startprob, means, covars, transmat):
    mapping = return_sorting_mapping(means)
    means_new = np.sort(means, axis=0)
    startprob_new = sort_startprob(mapping, startprob)
    covars_new = sort_covars(mapping, covars)
    transmat_new = sort_transition_matrix(mapping, transmat)
    assert np.shape(means_new) == np.shape(means)
    assert np.shape(startprob_new) == np.shape(startprob)
    assert np.shape(transmat_new) == np.shape(transmat)

    return [startprob_new, means_new, covars_new, transmat_new]




def compute_A_fhmm(list_A):
    result = list_A[0]
    for i in range(len(list_A) - 1):
        result = np.kron(result, list_A[i + 1])
    return result



def compute_means_fhmm(list_means):
    states_combination = list(itertools.product(*list_means))
    num_combinations = len(states_combination)
    means_stacked = np.array([sum(x) for x in states_combination])
    means = np.reshape(means_stacked, (num_combinations, 1))
    cov = np.tile(5 * np.identity(1), (num_combinations, 1, 1))
    return [means, cov]
```

```
def compute_pi_fhmm(list_pi):
    result = list_pi[0]
    for i in range(len(list_pi) - 1):
        result = np.kron(result, list_pi[i + 1])
    return result




def create_combined_hmm(model):
    list_pi = [model[appliance].startprob_ for appliance in model]
    list_A = [model[appliance].transmat_ for appliance in model]
    list_means = [model[appliance].means_.flatten().tolist()
                  for appliance in model]

    pi_combined = compute_pi_fhmm(list_pi)
    A_combined = compute_A_fhmm(list_A)
    [mean_combined, cov_combined] = compute_means_fhmm(list_means)

    combined_model = hmm.GaussianHMM(n_components=len(pi_combined), co
variance_type='full')
    combined_model.startprob_ = pi_combined
    combined_model.transmat_ = A_combined
    combined_model.covars_ = cov_combined
    combined_model.means_ = mean_combined

    return combined_model




def return_sorting_mapping(means):
    means_copy = deepcopy(means)
    means_copy = np.sort(means_copy, axis=0)

    mapping = {}
    for i, val in enumerate(means_copy):
        mapping[i] = np.where(val == means)[0][0]
    return mapping




def decode_hmm(length_sequence, centroids, appliance_list, states):
    hmm_states = {}
    hmm_power = {}
    total_num_combinations = 1

    for appliance in appliance_list:
        total_num_combinations *= len(centroids[appliance])

    for appliance in appliance_list:
        hmm_states[appliance] = np.zeros(length_sequence, dtype=np.int
)
        hmm_power[appliance] = np.zeros(length_sequence)

    for i in range(length_sequence):

        factor = total_num_combinations
        for appliance in appliance_list:
            factor = factor // len(centroids[appliance])

            temp = int(states[i]) / factor
            hmm_states[appliance][i] = temp % len(centroids[appliance]
)
            hmm_power[appliance][i] = centroids[
                appliance][hmm_states[appliance][i]]
    return [hmm_states, hmm_power]



def cluster(X, max_num_clusters=3, exact_num_clusters=None):
    data = _transform_data(X)

    centroids = _apply_clustering(data, max_num_clusters, exact_num_cl
usters)
    centroids = np.append(centroids, 0)
    centroids = np.round(centroids).astype(np.int32)
    centroids = np.unique(centroids)
    return centroids



def _transform_data(data):
    MAX_NUMBER_OF_SAMPLES = 2000
    MIN_NUMBER_OF_SAMPLES = 20
    DATA_THRESHOLD = 10

    data_above_thresh = data[data > DATA_THRESHOLD].dropna().values
    n_samples = len(data_above_thresh)
    if n_samples < MIN_NUMBER_OF_SAMPLES:
        return np.zeros((MAX_NUMBER_OF_SAMPLES, 1))
    elif n_samples > MAX_NUMBER_OF_SAMPLES:
        random_indices = np.random.randint(0, n_samples, MAX_NUMBER_OF
_SAMPLES)
        resampled = data_above_thresh[random_indices]
        return resampled.reshape(MAX_NUMBER_OF_SAMPLES, 1)
    else:
        return data_above_thresh.reshape(n_samples, 1)


def _apply_clustering_n_clusters(X, n_clusters):
    from sklearn.cluster import KMeans
    k_means = KMeans(init='k-means++', n_clusters=n_clusters)
    k_means.fit(X)
    return k_means.labels_, k_means.cluster_centers_



def _apply_clustering(X, max_num_clusters, exact_num_clusters=None):
    from sklearn import metrics

    import warnings
    warnings.filterwarnings("ignore", category=DeprecationWarning)
    num_clus = -1
    sh = -1
    k_means_labels = {}
    k_means_cluster_centers = {}
    k_means_labels_unique = {}

    if exact_num_clusters is not None:
        labels, centers = _apply_clustering_n_clusters(X, exact_num_cl
usters)
        return centers.flatten()

    for n_clusters in range(1, max_num_clusters):

        try:
            labels, centers = _apply_clustering_n_clusters(X, n_cluste
rs)
            k_means_labels[n_clusters] = labels
            k_means_cluster_centers[n_clusters] = centers
            k_means_labels_unique[n_clusters] = np.unique(labels)
            try:
                sh_n = metrics.silhouette_score(
                    X, k_means_labels[n_clusters], metric='euclidean')

                if sh_n > sh:
                    sh = sh_n
                    num_clus = n_clusters
            except Exception:
                num_clus = n_clusters
        except Exception:
            if num_clus > -1:
                return k_means_cluster_centers[num_clus]
            else:
                return np.array([0])

    return k_means_cluster_centers[num_clus].flatten()




def timeStamptoIndex(df):
    df.index = df['timestamp']
    return df
```

```
class FHMM:
    def __init__(self, debug=False):
        self.model = {}
        self.predictions = pd.DataFrame()
        self.MIN_CHUNK_LENGTH = 100
        self.MODEL_NAME = 'FHMM'
        self.debug = debug
        if self.debug : print("[FHMM Initialised]")



    def train(self, df, appliance_list):
        import warnings
        warnings.filterwarnings("ignore", category=Warning)
        learnt_model = OrderedDict()

        max_num_clusters = 2

        for i, meter in enumerate(appliance_list):
            meter_data = df[meter].dropna()
            X = meter_data.values.reshape((-1, 1))

            if not len(X):
                print(" [train] ERROR Submeter '{}' has no samples, sk
ipping...".format(meter))
                continue

            assert X.ndim == 2
            self.X = X

            states = cluster(meter_data, max_num_clusters)
            num_total_states = len(states)

            if self.debug : print(" [train] Training model for submete
r", meter)
            learnt_model[meter] = hmm.GaussianHMM(num_total_states, "f
ull")

            learnt_model[meter].fit(X)

        self.meters = []
        new_learnt_models = OrderedDict()
        for meter in learnt_model:
            startprob, means, covars, transmat = sort_learnt_parameter
s(
                learnt_model[meter].startprob_, learnt_model[meter].me
ans_,
                learnt_model[meter].covars_, learnt_model[meter].trans
mat_)

            new_learnt_models[meter] = hmm.GaussianHMM(startprob.size,
"full")
```

```
            new_learnt_models[meter].startprob_ = startprob
            new_learnt_models[meter].transmat_ = transmat
            new_learnt_models[meter].means_ = means
            new_learnt_models[meter].covars_ = covars
            self.meters.append(meter)

        learnt_model_combined = create_combined_hmm(new_learnt_models)
        self.individual = new_learnt_models
        self.model = learnt_model_combined



    def disaggregate(self, df):
        if not 'timestamp' in df:
            print("[FHMM_model][disaggregate] Could not detect column
\"timestamp\" in the given dataframe")
            return

        if not 'power' in df:
            print("[FHMM_model][disaggregate] Could not detect column
\"timestamp\" in the given dataframe")
            return

        test_mains = df['power']

        learnt_states_array = []
        test_mains = test_mains.dropna()
        length = len(test_mains.index)
        temp = test_mains.values.reshape(length, 1)
        learnt_states_array.append(self.model.predict(temp))

        means = OrderedDict()
        for elec_meter, model in iteritems(self.individual):
            means[elec_meter] = (
                model.means_.round().astype(int).flatten().tolist())
            means[elec_meter].sort()

        decoded_power_array = []
        decoded_states_array = []

        for learnt_states in learnt_states_array:
            [decoded_states, decoded_power] = decode_hmm(
                len(learnt_states), means, means.keys(), learnt_states
)
            decoded_states_array.append(decoded_states)
            decoded_power_array.append(decoded_power)

        prediction = pd.DataFrame(
            decoded_power_array[0], index=test_mains.index)

        prediction.index = pd.to_datetime(df['timestamp'], unit='s')

        return prediction
    def save(self, filename):
        with open(filename+'.pkl', 'wb') as output:
            pickle.dump(self.model, output, pickle.HIGHEST_PROTOCOL)
            pickle.dump(self.individual, output, pickle.HIGHEST_PROTOC
OL)


    def load(self, filename):
        with open(filename+'.pkl', 'rb') as input:
            self.model = pickle.load(input)
            self.individual = pickle.load(input)
```

Appendix 2

Appendix 2 includes the overall implementation of the disaggregation model and the recommendation system. The programming language that was used is Python and more specifically, the code has been developed in Jyputer Notebooks under Anaconda package. This package was used because it provides a large number of libraries that has been used in only one installation.

```
import pandas as pd
import matplotlib.pyplot as plt

%matplotlib inline

plt.style.use("ggplot")


mains_1_path = r'C:\Users\Jordan\Documents\NILMTK\electricity\1.csv'
mains_2_path = r'C:\Users\Jordan\Documents\NILMTK\electricity\2.csv'
fridge_path = r'C:\Users\Jordan\Documents\NILMTK\electricity\3.csv'
ac_path_1 = r'C:\Users\Jordan\Documents\NILMTK\electricity\4.csv'
ac_path_2 = r'C:\Users\Jordan\Documents\NILMTK\electricity\5.csv'
washing_machine_path = r'C:\Users\Jordan\Documents\NILMTK\electricity\
6.csv'
iron_path = r'C:\Users\Jordan\Documents\NILMTK\electricity\8.csv'


columns_mains = ['timestamp', 'W']
columns_appliances = ['W']
df_mains_1 = pd.read_csv(mains_1_path, sep = ',', error_bad_lines=Fals
e, index_col=False, dtype='unicode', usecols=columns_mains)
df_mains_2 = pd.read_csv(mains_2_path, sep = ',', error_bad_lines=Fals
e, index_col=False, dtype='unicode', usecols=columns_mains)
df_fridge = pd.read_csv(fridge_path, sep = ',', error_bad_lines=False,
index_col=False, dtype='unicode', usecols=columns_appliances)
df_ac = pd.read_csv(ac_path_1, sep = ',', error_bad_lines=False, index
_col=False, dtype='unicode', usecols=columns_appliances)
df_washing_machine = pd.read_csv(washing_machine_path, sep = ',', erro
r_bad_lines=False, index_col=False, dtype='unicode', usecols=columns_a
ppliances)
df_iron = pd.read_csv(iron_path, sep = ',', error_bad_lines=False, ind
ex_col=False, dtype='unicode', usecols=columns_appliances)
```

```
df_mains_1 = df_mains_1.rename({'W': 'power'}, axis=1)

df_mains_1 = df_mains_1.astype({"timestamp": int})

df_mains_1['power'] = pd.to_numeric(df_mains_1['power'],errors='co-

erce')


df_mains_1['power'] = df_mains_1['power'].fillna(0)
df_mains_1['power'].isnull().values.any()

False


df_mains_2 = df_mains_2.rename({'W': 'power'}, axis=1)

df_mains_2 = df_mains_2.astype({"timestamp": int})

df_mains_2['power'] = pd.to_numeric(df_mains_2['power'],errors='co-

erce')


df_mains_2['power'] = df_mains_2['power'].fillna(0)
df_mains_2['power'].isnull().values.any()

False


df_fridge = df_fridge.rename({'W': 'app1'}, axis=1)
df_fridge['app1'] = pd.to_numeric(df_fridge['app1'],errors='coerce')
df_fridge['app1'].isnull().values.any()

False


fridge = df_fridge['app1'].sum()
round(fridge)

270113625


df_ac = df_ac.rename({'W': 'app2'}, axis=1)
df_ac['app2'] = pd.to_numeric(df_ac['app2'],errors='coerce')
df_ac['app2'].isnull().values.any()

False


ac = df_ac['app2'].sum()
round(ac)

689668541
```

```
df_washing_machine = df_washing_machine.rename({'W': 'app4'}, axis=1)
df_washing_machine['app4'] = pd.to_numeric(df_washing_ma-
chine['app4'],errors='coerce')


df_washing_machine['app4'].isnull().values.any()

False

washing_machine = df_washing_machine['app4'].sum()
round(washing_machine)

4850056


df_iron = df_iron.rename({'W': 'app6'}, axis=1)

df_iron['app6'] = pd.to_numeric(df_iron['app6'],errors='coerce')

df_iron['app6'].isnull().values.any()

False

iron = df_iron['app6'].sum()
round(iron)

2606377


df_train = pd.concat([df_mains_1, df_fridge, df_ac, df_washing_machine
, df_iron,], axis=1, join="inner")


list_of_appliance = ['app1', 'app2', 'app4', 'app6']
fhmm = FHMM()
fhmm.train(df_train, list_of_appliance)

df_test=df_mains_2

prediction = fhmm.disaggregate(df_test)


df_prediction = prediction.rename(columns={'app1': 'Fridge', 'app2':
'AC','app4':'Washing_Machine','app6':'Iron'})


df_gt = df_train.rename(columns={'app1': 'Fridge', 'app2': 'AC',
'app4':'Washing_Machine','app6':'Iron'})

df_gt = df_gt.set_index('timestamp')

df_gt = df_gt[['Fridge','AC', 'Washing_Machine', 'Iron']]
```

```
building_appliances = pd.DataFrame(df_prediction.columns.values, col-
umns=['Appliances'])
appliances_list=building_appliances.values.tolist()
appliances_list = [ item for elem in appliances_list for item in elem]


appliances_data = ['Fridge', 'Light', 'Microwave', 'Stove', 'AC',
'Washing_Machine','Water_Heater']
appliances_list_com = list(set(appliances_list).intersection(appli-
ances_data))


buildings_path = r'C:\Users\Jordan\Documents\NILMTK\Buildings.csv'
fridge_path = r'C:\Users\Jordan\Documents\NILMTK\Fridge.csv'
light_path = r'C:\Users\Jordan\Documents\NILMTK\Light.csv'
microwave_path = r'C:\Users\Jordan\Documents\NILMTK\Microwave.csv'
stove_path = r'C:\Users\Jordan\Documents\NILMTK\Stove.csv'
washing_machine_path = r'C:\Users\Jordan\Documents\NILMTK\Washing_mach
ine.csv'
water_heater_path = r'C:\Users\Jordan\Documents\NILMTK\Water_heater.cs
v'
ac_path = r'C:\Users\Jordan\Documents\NILMTK\AC.csv'


df_buildings_data = pd.read_csv(buildings_path, delimiter = ';')
df_buildings_data.index+=1


df_fridge_ratings = pd.read_csv(fridge_path, delimiter = ';')
df_fridge_ratings.index+=1


df_light_ratings = pd.read_csv(light_path, delimiter = ';')
df_light_ratings.index+=1


df_microwave_ratings = pd.read_csv(microwave_path, delimiter = ';')
df_microwave_ratings.index+=1


df_stove_ratings = pd.read_csv(stove_path, delimiter = ';')
df_stove_ratings.index+=1
```

```
df_washing_machine_ratings = pd.read_csv(washing_machine_path, delim-
iter=';')
df_washing_machine_ratings.index+=1

df_water_heater_ratings = pd.read_csv(water_heater_path, delimiter =
';')
df_water_heater_ratings.index+=1

df_ac_ratings = pd.read_csv(ac_path, delimiter = ';')
df_ac_ratings.index+=1

df_equal = df_buildings_data.loc[(df_buildings_data['Type'] == 'Resi-
dential') & (df_buildings_data['Climate'] == 'Sub tropical')]

df_light_equal=df_light_ratings[df_equal.eq(df_light_rat-
ings).any(axis=1)]

df_microwave_equal=df_microwave_ratings[df_equal.eq(df_microwave_rat-
ings).any(axis=1)]

df_stove_equal=df_stove_ratings[df_equal.eq(df_stove_rat-
ings).any(axis=1)]

df_water_heater_equal=df_water_heater_ratings[df_equal.eq(df_wa-
ter_heater_ratings).any(axis=1)]

df_washing_machine_equal=df_washing_machine_rat-
ings[df_equal.eq(df_washing_machine_ratings).any(axis=1)]

df_ac_equal=df_ac_ratings[df_equal.eq(df_ac_ratings).any(axis=1)]
```

```
recommendations={}


def recommendation_extraction(df_appliance):
    df_appliance = df_appliance.drop(['Building Number'], axis = 1)
    df_appliance.loc['avg'] = df_appliance.mean()
    df_avg = df_appliance.loc[['avg']]
    rec = df_avg.idxmax(axis=1).values[0]
    max_rec = df_avg.max(axis=1).values[0]
    recommendations[rec] = max_rec



if 'Fridge' in appliances_list_com:
    recommendation_extraction(df_fridge_equal)

if 'Light' in appliances_list_com:
    recommendation_extraction(df_light_equal)

if 'Microwave' in appliances_list_com:
    recommendation_extraction(df_microwave_equal)

if 'Stove' in appliances_list_com:
    recommendation_extraction(df_stove_equal)

if 'AC' in appliances_list_com:
    recommendation_extraction(df_ac_equal)

if 'Washing_Machine' in appliances_list_com:
    recommendation_extraction(df_washing_machine_equal)

if 'Water_Heater' in appliances_list_com:
    recommendation_extraction(df_water_heater_equal)
```