Predictive Analysis and Load Identification for Smart Energy Meter

Undergraduate Student Project

by

Nadine Danielle Reyes
2015-07586
BS Computer Engineering

Russtin Tolentino
2015-08760
BS Computer Engineering

Adviser:

Nestor Michael Tiglao, PhD

University of the Philippines, Diliman
January 2022

Abstract

Predictive Analysis and Load Identification for Smart Energy Meter

Smart Energy Meter devices allow consumers to be aware of their consumption and make smarter decisions based on this information. Due to the pandemic, the energy consumption for every household has continued to rise as people are being forced to stay at home. With this, the need to increase awareness of one's own consumption has also risen. The purpose of this study is to present a system capable of presenting essential appliance information in an easily accessible and understandable form. More specifically, this system is capable of electricity consumption prediction alongside precise automatic load detection that adapts to the user's needs. The system gathers data based on specific appliances used by a user with the intention of comparing data per device. The said system is scalable based on the number of users and implemented using a SONOFF POW R2, a short extension outlet, a virtual machine, and present output features through a mobile phone. The results of the study show that the created models with an accuracy of 0.64 for the UK-DALE dataset, 0.97 for the first model that used the power readings in the created dataset and 0.77 for the second model which utilized an implementation of PCA in the created dataset, are capable of identifying appliances connected to the smart meter and giving a consumption prediction.

Contents

List of Figures

Chapter 1

Introduction

1.1 Background of the Project

COVID-19 pandemic is still not over and a lot of people are still staying at home most of the time, students are still on a virtual setup and the working class are on a work-from-home/hybrid setup. With this, the trend for residential electricity consumption continues to increase. Even the secretary of the Department of Energy (DoE), Secretary Alfonso Cusi, encourages the people to be mindful of their consumption, especially as there are a lot of people who lost their jobs [1]. In addition to this, the Philippines is experiencing hot weather that may result in residents using air conditioning units more often than before, thus increasing their consumption [2].

Smart Energy Meters use are not that prominent here in the Philippines and residents only rely on their electricity provider to inform them of their consumption on a monthly basis. It is also difficult to monitor the consumption of individual appliances. However, this may change in the near future as the Manila Electric Co. (Meralco) plans to gradually transition to Smart Meter for their postpaid subscribers to allow them to monitor their consumption in real-time [3]. With this change, the over-estimation of consumers' bills last year during the enhanced community quarantine (ECQ) may be avoided.

In order to have an idea of how each of the appliances contributes to the whole consumption of the house, load disaggregation can be used. Load disaggregation predicts the appliances that are present in an aggregate data from a smart meter, as well as their consumption. To be able to do this, there must be data for each appliance to be trained. With the lack of dataset here in the country, the researchers proposed a way to collect data from a number of appliances and use it to test an existing load disaggregation algorithm.

The proposed way to collect the data is to have a SONOFF POW R2, flashed with Tasmota, connected to a convenience outlet. Appliances that were considered are the ones that the students commonly use. Home Assistant is used to be able to get the data from the device and a Telegram bot to send notifications to the users of their current consumption.

1.2 Significance of the Project

The goal of the project is to create a usable system that is easy to understand and implement to existing homes and establishments. Due to the pandemic, making homes and establishments smarter through smart devices have been a way to adapt to our situation. The researchers were able to come up with a model capable of detecting the presence of specific appliances connected to the same power strip or extension outlet. This model can be used in homes and establishments such as coffee shops. In coffee shops, owners will be able to detect what kind of appliances are connected to their public outlets and an estimated total cost of the usage of these appliances. This can be advantageous for both safety and cost as their awareness allows them to control which extension outlets can be used as the Tasmota device can be remotely switched off. In addition, with improvements such as detecting the specific time, it can be used for forensic purposes. By detecting the specific time when an appliance was present, it could give more information about the case. For practicality on own home usage, the model is easily repeatable. It allows remote control of the power strip or extension outlet while giving information such as estimated cost for the day. More importantly, the appliance detection can help homeowners have greater control on electricity consumption in their homes. In our country, using extension outlets is quite prevalent. With the implementation of the model, important information regarding the devices connected to the connection outlet becomes readily available. Some direct application of the model on homes would be knowing which combination of appliances would cost more based on simultaneous usage. By knowing which appliances are connected to the extension outlet, users will be able to have an idea of their own consumption. Another application of the model to our country would be the usage of multiple extension outlets or power strips wherein the devices connected to each extension outlet cannot be easily identified. The presence prediction capability of the model provides the necessary information such that the user can safely unplug the specific appliance he or she intends to unplug.

1.3 Documentation Flow and Organization

Review of Related Works will be discussed in Chapter 2, this will include studies done with relation to electricity consumption prediction and load identification. Chapter 3 will be focusing on the problem statement, objectives, and scope and limitations of the study. How the project and testing are made is presented in Chapter 4: Methodology. The discussion of results will be on Chapter 5. And lastly, Chapter 6 will discuss the conclusion and recommendations for the project.

Chapter 2

Related Work

2.1 Load Identification based on Electricity Parameters

To be able to identify the loads in an aggregate data, load disaggregation is used. This can also predict the contribution of each load to the total consumption. In this section, related works about load disaggregation and dataset used will be presented.

2.1.1 Load Disaggregation

In an attempt to compare existing load disaggregation algorithms, Batra et al. created a Non-Intrusive Load Monitoring Toolkit (NILMTK) that will be able to compare multiple algorithms and have it reproducible by making it open-source [4]. Emphasizing the benefits of being able to identify the appliances that contribute to the total consumption of the household and how much each appliance contributes by providing three motivations for the project. The three motivations mentioned are appliance level information regarding consumption, taking actions in reducing energy consumption, and providing advice as to what appliances can be used during off-peak/peak times, if the timestamp is available.

The authors from [4] created the NILMTK using Python using its machine learning library called ***scikit-learn***. The toolkit is made in a modular manner so that the modifications can be made easily without the need to modify the whole program. It contains several load disaggregation algorithms, dataset preprocessing to their proposed NILMTK-DF (data format), and accuracy metrics to test each algorithm to a specific test case. There were several publicly available datasets that were used in this research, namely, Reference Energy Disaggregation Dataset (REDD), Building-Level Fully-Labeled Dataset (BLUED), Smart Dataset, Pecan Street Dataset, Household Electricity Survey Dataset, Almanac of Minutely Power Dataset (AMPds), Indian data for Ambient Water and Electricity Sensing (iAWE), and UK Domestic Appliance-Level Electricity Dataset (UK-DALE). After preprocessing the datasets, which includes downsampling, voltage normalization, and selecting top-k appliances, each of them went through two disaggregation algorithms, Combinatorial Optimization (CO) and Factorial Hidden Markov Model (FHMM). CO gets the total power of all the possible combinations of appliances and then gets the nearest predicted power from the observed aggregate power. While the FHMM uses the appliances' states as hidden components of the model. The time complexity of the FHMM algorithm is $O(TK^{2N})$ while CO's is $O(TK^{N})$. After going through the disaggregation algorithms mentioned, the results are then evaluated by the metrics such as the error in total energy assigned, the fraction of total energy assigned correctly, the normalized error in assigned power, the RMS error in assigned power, the confusion matrix, the true/false positive rate, the precision and recall, the F1-score, and the hamming loss.

In the NILMTK paper [4], the authors showed the comparison between the two algorithms using the iAWE dataset. It can be seen that the two algorithms show almost the same F-Score and NEP. The source-code of the toolkit can be found here [5].

In another paper by Kitisak Osathanunkul and Khukrit Osanthanunkul [6], they created a load disaggregation library that is a simplified version of the NILMTK. The authors mentioned that NILMTK contains a lot of features that it requires a steep learning curve especially for those who are just starting on load disaggregation and machine learning. With this, they created the Simple Load Disaggregation Library (SLD) that will be able to do load disaggregation but in a way that it will be easier for beginner researchers to understand and use.

SLD uses the same load disaggregation algorithms as the NILMTK, FHMM and CO. FHMM, it learns the load patterns of all the appliances listed. While the CO gets the sum of the combination of appliances and compares it to the aggregated power data. The steps in using the library are listed below:

- Ensuring that the software and library requirements are satisfied. This includes that Python, Numpy, Pandas, and HMMLEARN are properly installed.
- Data are preprocessed as follows:
  - Training Data. Prepare a dataframe that contains the timestamp, total power, and

individual power consumption of appliances to be trained.

  - Testing Data. A dataframe that contains the timestamp and the power reading of the whole house/aggregated data.

- Create and Train the Model. A list of appliances is needed as an input together with the training data to the ***train*** function. After calling the ***train*** function, the trained model can now be saved and loaded for future use.

- Lastly, the load disaggregation function can now be called.

The source code of this library can be found here [7].

2.1.2 Dataset

UK Domestic Appliance - Level Electricity (UK-DALE) dataset contains a record of the power consumption of five houses. In each house, there are submeters connected to individual appliances that will be able to record each power consumption. In some appliances, the switch times are also recorded [8]. Each house has its own folder that contains the recordings of the appliances present in each household, one appliance is to one file. In each file there are timestamps and the power reading that corresponds to it. The dataset can be found here [9].

2.2 Electricity Bill Prediction

A study [10] published in the 2017 International Electrical Engineering Congress focused on smart meters being utilized for electricity bill forecasting. In this paper, an API was developed so that consumers can obtain data from a database. An Electricity Bill Forecasting Application used the average daily energy consumption in the last 7 days to predict the energy consumption and electricity cost for the billing period. The application flow starts with getting the Energy consumption in the last 7 days and saves it as $EWeek$. Secondly, the average energy consumption $Eavg$ is computed by dividing $EWeek$ by 7. Third, the energy consumption from the last due date until current day is saved as $Eactual$. The application then starts calculating the energy consumption from the current day until the next due date and saves it as $DayRemain$. Lastly, the predicted energy consumption is solved as $Eforecast = Eactual + (Eavg * DayRemain)$. For billing prediction, the computed $Eforecast$ is used alongside the electricity company set tariff to produce an estimation on the cost of that billing period. The data is then sent out to an application and shows up as a notification. A problem that arose during the testing was having data loss due to WiFi signal loss that could be attributed to electricity outage and hardware malfunction. The electricity parameters were measured and sent every 1 minute. After testing the system for 40 days, the recorded available data only amounted to 87% of the possible available data. The Forecasting accuracy of the system showed an average of 96% with a minimum of 94% accuracy. To check the accuracy, daily predicted electricity cost for 10 days was compared with the actual electricity cost found in the electricity bill. The recommendations of the paper were to make the system a learning tool and not simply a monitoring system. Another recommendation was to make the system tailor-designed to the specific needs and requirements of the consumer.

A tool that can be used to implement the consumption prediction is Home Assistant. Home Assistant OS supports numerous built-in integrations[11]. One of such is the utility meter which provides the user the capability to track consumption of various utilities.

To present the output of the prediction, *Telegram* and with utilization of its bots can be used. Telegram Bots are third-party applications that can be controlled to provide a needed service such as notifications within the messaging application[12]. Home Assistant also allows the integration of Telegram.

Another tool is Tasmota. Tasmota provides more control over ESP devices. It also adds flexibility with its MQTT integration[13].

Chapter 3

Problem Statement and Objectives

3.1 Problem Statement and Objectives

Electricity bills rise as consumption increases. Smart Energy Meter monitors and tracks the consumption per household that allows consumers to view their consumption through their smartphones. The goal of this project is to provide consumer data analysis, descriptive and predictive, and load identification for better consumption planning.

In order to reach the set goal, the following objectives are set:

- Create a data set that contains commonly used devices.
- Determine the devices that are connected to the Smart Energy Meter.
- Provide prediction of future consumption based on the recorded number of users, length of use, and devices used.
- Utilize Home Assistant features to use as dashboard for all project's features.

3.2 Scope and Limitations

Given the circumstances due to the pandemic, the experiment will only take place in the household of the researchers.
Here are the scope and limitations of the project:

- The setup made can only accommodate up to a maximum of 4 appliances at the same time. And only up to a maximum load of 15 A.

- Manual declaration of the list of appliances when using the load disaggregation model.
- In the dataset used by the researchers as their own, only 3 appliances are present.
- Effect of outside factors, such as the weather, on the consumption will be little to zero since the experiment was only conducted for a few months.
- Due to time constraints, and a few problems encountered with the setup, the web application for the load disaggregation model was not made.

Chapter 4

Methodology

4.1 System Setup

The Smart Energy Monitoring Set-up used a virtual machine for storing and transmission of data. It was also used for communication with the host consumer appliance. The Sonoff POW R2 was used for obtaining the electricity parameters per appliance.

4.1.1 Design

The SONOFF POW R2 collected the important data and analyzed the appliance load power (current effective power load), power factor (ratio of real power flowing to load to apparent power in circuit), voltage, current and energy usage each day. All gathered data were stored as CSV files. The SONOFF POW R2 then sent telemetry data to a local server via MQTT. The collected data was analyzed for patterns and features and were used to develop the load identification Machine Learning models. The created Machine Learning model was used by the system for dynamic identification of the various loads connected to the SONOFF POW R2.

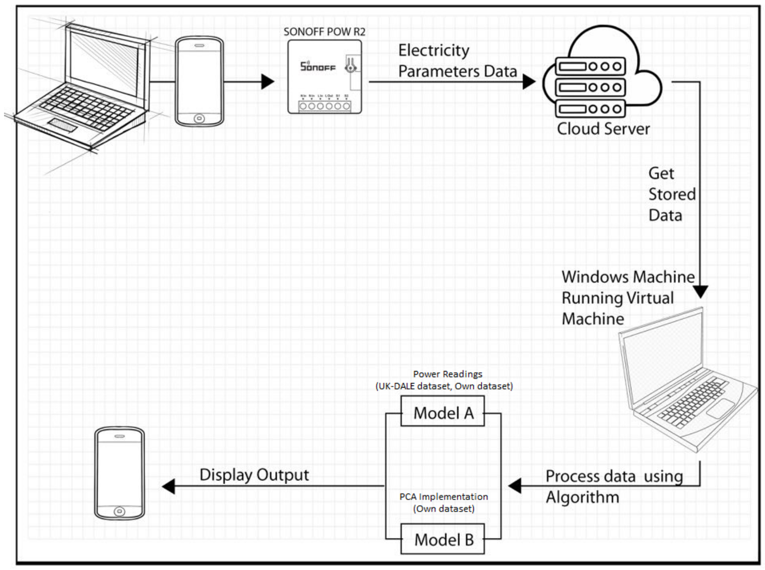

SONOFF POW R2
Electricity Parameters Data
Cloud Server
Get Stored Data
Windows Machine Running Virtual Machine
Power Readings (UK-DALE dataset, Own dataset)
Model A
Display Output
Process data using Algorithm
PCA Implementation (Own dataset)
Model B

Figure 4.1: System Design

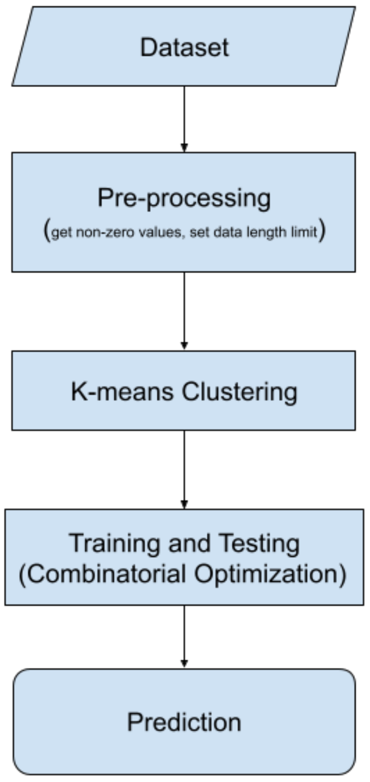

Dataset
Pre-processing
(get non-zero values, set data length limit)
K-means Clustering
Training and Testing
(Combinatorial Optimization)
Prediction

Figure 4.2: Machine Learning Model Design Flowchart

4.1.2 Implementation

SONOFF POW R2 device was utilized in the created Smart Energy Monitoring system per household. The created system used a custom Tasmota firmware. To emulate a branch circuit and for ease of electrical connection, the SONOFF POW R2 was connected to a convenience outlet and the various loads were then connected to it. Due to this, the appliances were limited to a maximum current of 15A and power of 3500W. These devices were identified by the algorithm.

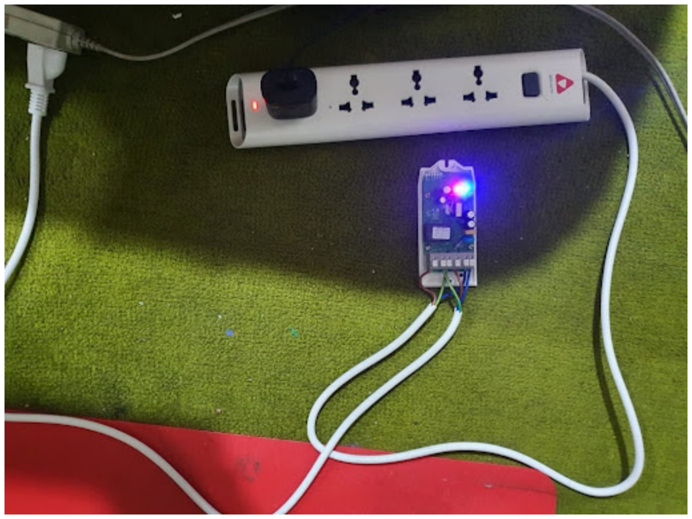


Figure 4.3: Full Setup

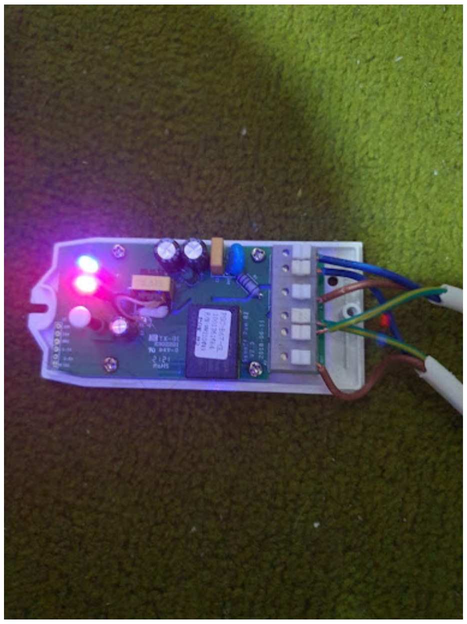


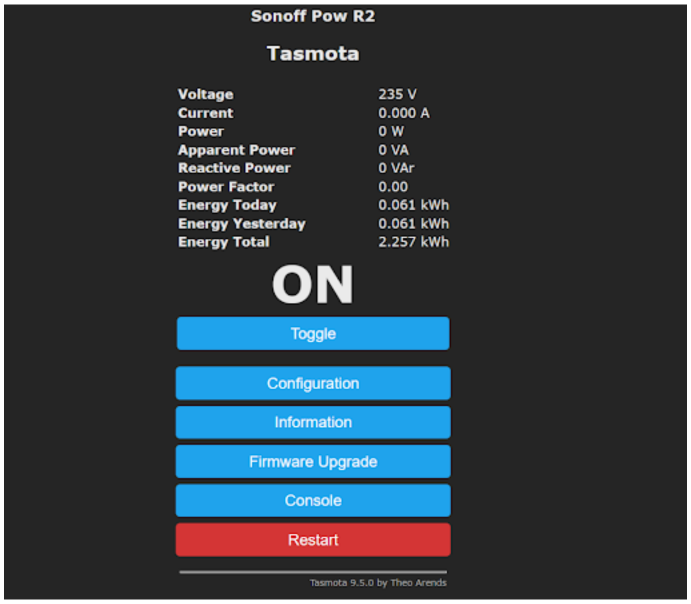
Figure 4.4: Sonoff Pow R2 (Tasmota Device)

4.2 Data Collection

In this section, how the data that was collected by the setup mentioned above is transferred to the Home Assistant. The installation and the modules used are also explained.

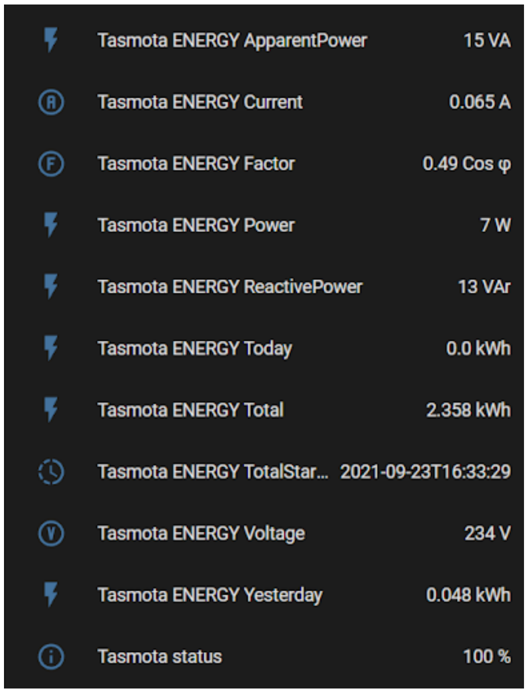
4.2.1 Tasmota

Tasmota is an open-source firmware that allows greater control over ESP devices (”Espressif” Shanghai-based Chinese manufacturer) Tasmota version 9.5.0 was flashed into the Sonoff Pow R2 device.



Voltage	235 V
Current	0.000 A
Power	0 W
Apparent Power	0 VA
Reactive Power	0 VAr
Power Factor	0.00
Energy Today	0.061 kWh
Energy Yesterday	0.061 kWh
Energy Total	2.257 kWh

Figure 4.5: Tasmota Web Interface



Tasmota ENERGY ApparentPower	15 VA
Tasmota ENERGY Current	0.065 A
Tasmota ENERGY Factor	0.49 Cos φ
Tasmota ENERGY Power	7 W
Tasmota ENERGY ReactivePower	13 VAr
Tasmota ENERGY Today	0.0 kWh
Tasmota ENERGY Total	2.358 kWh
Tasmota ENERGY TotalStar...	2021-09-23T16:33:29
Tasmota ENERGY Voltage	234 V
Tasmota ENERGY Yesterday	0.048 kWh
Tasmota status	100 %

Figure 4.6: Tasmota Readings in Home Assistant

Installation and Set-up process:
Using a FT232RL FTDI mini USB to TTL Serial Adapter, the Sonoff POW R2 was connected to a PC. The Tasmota firmware was then flashed using ESP Easy Flasher. The Tasmota device was then configured with the help of Termite.

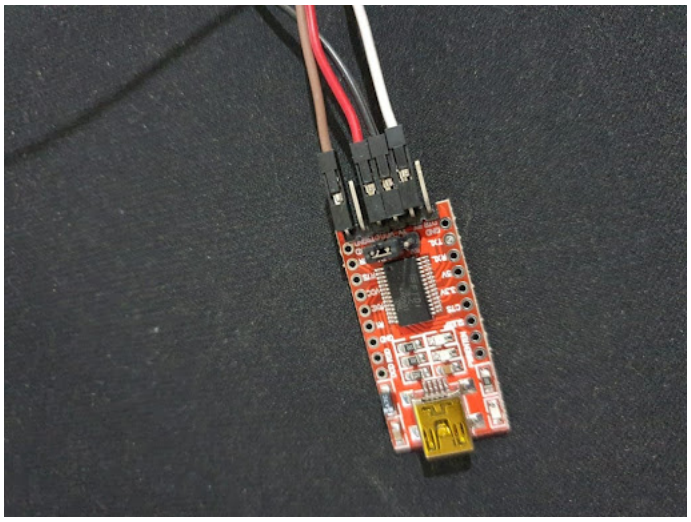


Figure 4.7: USB to TTL Serial Adapter

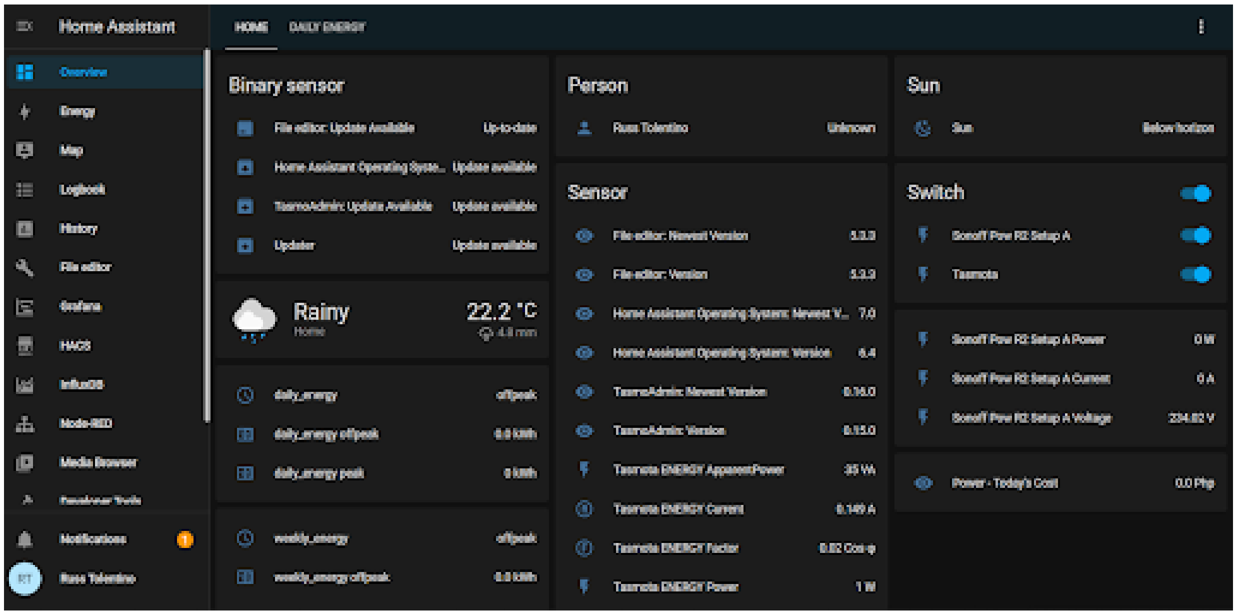
4.2.2 Home Assistant

Home Assistant is a free, open-source software used for controlling smart devices and home automation.

Installation and Set-up Process:
Home Assistant OS version installed is 6.4 with core-2021.9.7.

Modules Installed:

- File Editor was used to edit the configuration files of Home Assistant OS
- Mosquitto broker was used to bridge the connection between the Tasmota device and Home Assistant
- TasmoAdmin is an integration in Home Assistant for controlling Tasmota devices
- Samba share was used for accessing the internal Home Assistant files such as configuration.yaml
- InfluxDB was used as the database for storing the data obtained from the Tasmota device
- Grafana was used for displaying the data inside the database



Binary sensor

File editor: Update Available	Up-to-date
Home Assistant Operating Syste...	Update available
TasmoAdmin: Update Available	Update available
Updater	Update available

Rainy 22.2 °C
Home 4.8 mm

daily_energy	offpeak
daily_energy offpeak	0.0 kWh
daily_energy peak	0 kWh

weekly_energy	offpeak
weekly_energy offpeak	0.0 kWh

Person

Russ Tolentino	Unknown

Sensor

File editor: Newest Version	5.3.3
File editor: Version	5.3.3
Home Assistant Operating System: Newest V...	7.0
Home Assistant Operating System: Version	6.4
TasmoAdmin: Newest Version	0.16.0
TasmoAdmin: Version	0.15.0
Tasmota ENERGY ApparentPower	35 VA
Tasmota ENERGY Current	0.149 A
Tasmota ENERGY Factor	0.02 Cos φ
Tasmota ENERGY Power	1 W

Sun

Sun	Below horizon

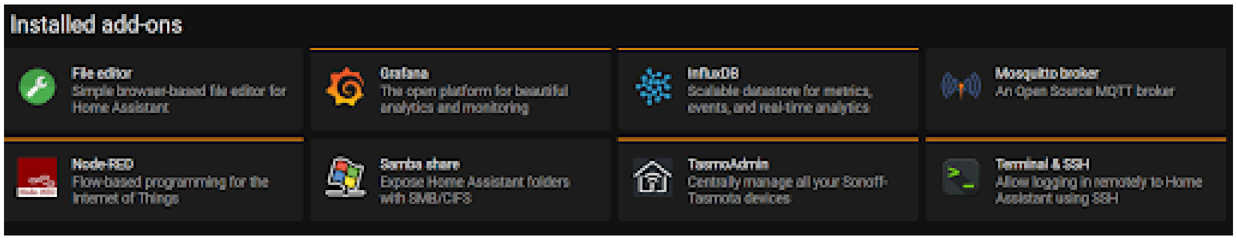
Switch

Sonoff Pow R2 Setup A	
Tasmota	

Sonoff Pow R2 Setup A Power	0 W
Sonoff Pow R2 Setup A Current	0 A
Sonoff Pow R2 Setup A Voltage	234.82 V

Power - Today's Cost	0.0 Php

Figure 4.8: Home Assistant Web Interface


Installed add-ons

File editor	Simple browser-based file editor for Home Assistant
Grafana	The open platform for beautiful analytics and monitoring
InfluxDB	Scalable datastore for metrics, events, and real-time analytics
Mosquitto broker	An Open Source MQTT broker
Node-RED	Flow-based programming for the Internet of Things
Samba share	Expose Home Assistant folders with SMB/CIFS
TasmoAdmin	Centrally manage all your Sonoff-Tasmota devices
Terminal & SSH	Allow logging in remotely to Home Assistant using SSH

Figure 4.9: Modules

4.3 Load Identification

The whole process in getting the predicted loads in an input of aggregated power and combined features is presented in this section. This includes the preprocessing of each data, load disaggregation itself, and the prediction of the appliance.

4.3.1 Data Preprocessing

In Machine Learning, the data is divided into two, the training and test sets. Python has a library called ***sklearn*** that automatically splits the data according to the ratio set by the user. For this model, the ratio is 80-20, training and test set respectively. But before splitting the data, it needs to be prepared according to the data format of the model to be used. The format is similar to what is indicated in [6], where the first column is the total power of the combination of appliances, and the next columns are the power reading of each appliance. Preprocessing for each dataset is different because of the characteristics of the data and how they were collected.

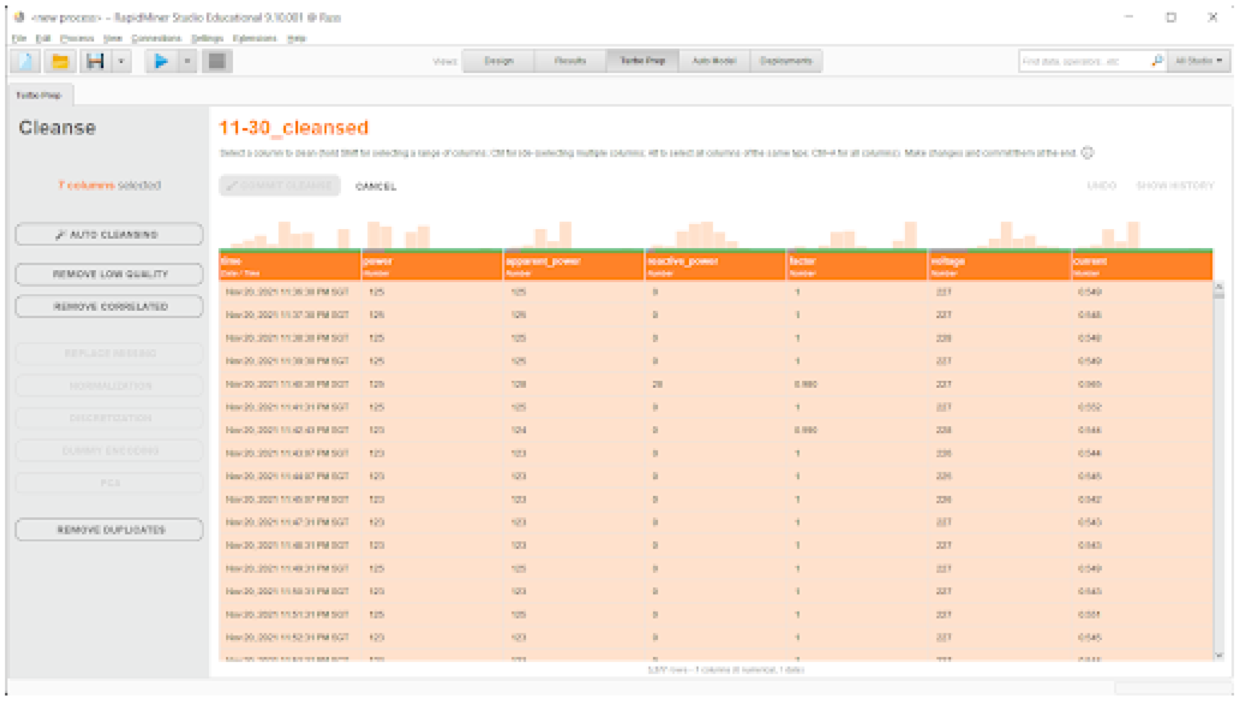
4.3.1.1 UK-DALE Dataset

The researchers wanted to set the number of appliances to four (4) as it is the maximum number of appliances allowed in their setup. With this, they decided to use four (4) appliances from House 1 namely: LCD Office (channel 14), fridge (channel 12), TV (channel 7), and Home Theatre PC (channel 9). The appliances mentioned were chosen as they are the appliances that are similar to commonly used appliances here in the country. Since the data for each appliances is already separated per file, clustering the data per appliance is not a problem anymore. Only the first one thousand (1000) non-zero values of the file were considered, to only get the readings where the appliance is at its ON state. During this preprocessing stage, the task is to get the total power of all possible combinations of appliances, then combine everything in one dataframe.

Once the dataframe is made, the researchers made sure that the minimum power reading of each appliance is 10 W and no NaN in the dataframe. This is because, upon observation, the model has problems predicting power values less than 10 W. The appliances does not contain the same number of data points, might not also reach 1000, so the researchers made sure that there is no NaN in the data. The last step of this stage is to label the data according to the combination made.

4.3.1.2 Created Dataset

The data exported from the Home Assistant was preprocessed using RapidMiner. From the TurboPrep feature, Auto-cleansing was applied to the data which primarily replaced missing values to zero and set the format of the readings to real numbers. The cleansed data then was subjected to KMeans to determine the different appliances and cluster the data accordingly. For the data used there are three appliances, namely: laptop, TV, and electric fan.



time (Date / Time)	power (Number)	apparent_power (Number)	reactive_power (Number)	factor (Number)	voltage (Number)	current (Number)
Nov 20, 2021 11:36:38 PM SGT	125	125	0	1	227	0.549
Nov 20, 2021 11:37:38 PM SGT	125	125	0	1	227	0.548
Nov 20, 2021 11:38:38 PM SGT	125	125	0	1	228	0.548
Nov 20, 2021 11:39:38 PM SGT	125	125	0	1	227	0.549
Nov 20, 2021 11:40:38 PM SGT	125	128	28	0.980	227	0.560
Nov 20, 2021 11:41:31 PM SGT	125	125	0	1	227	0.552
Nov 20, 2021 11:42:43 PM SGT	123	124	0	0.990	228	0.544
Nov 20, 2021 11:43:57 PM SGT	123	123	0	1	226	0.544
Nov 20, 2021 11:44:57 PM SGT	123	123	0	1	226	0.545
Nov 20, 2021 11:45:57 PM SGT	123	123	0	1	226	0.542
Nov 20, 2021 11:47:31 PM SGT	123	123	0	1	227	0.543
Nov 20, 2021 11:48:31 PM SGT	123	123	0	1	227	0.543
Nov 20, 2021 11:49:31 PM SGT	125	125	0	1	227	0.549
Nov 20, 2021 11:50:31 PM SGT	123	123	0	1	227	0.543
Nov 20, 2021 11:51:31 PM SGT	125	125	0	1	227	0.551
Nov 20, 2021 11:52:31 PM SGT	123	123	0	1	227	0.545

Figure 4.10: Rapid Miner Turbo Prep

For the model that uses the power readings as input, the clustered data is then separated according to clusters. Similar to the UK-DALE dataset, the total power of all possible combinations were obtained to create a new dataset after getting the first one thousand (1000) non-zero values of each cluster.

The data consisted of numerous features, namely power, apparent power, reactive power, power factor, and current. These features were then transformed using PCA (Principal Component Analysis). Principal Component Analysis is a technique used for dimensionality reduction. It aims to make data more interpretable while minimizing information loss by creating new variables that maximize variance and are uncorrelated. The method of PCA in data reduction involves creating an index variable, called a component, based on a larger set of measured variables. This is done through a linear combination of the set of variables. For the project, the measured variables were power, apparent power, reactive power, power factor, and current. Through the implementation of PCA, the goal was to combine these measured variables into a single or fewer components. The combination of the measured variables are weighted, meaning that some are more emphasized than others. The PCA function was imported from ***sklearn*** as well.

The output of the PCA function is then used as an input for the same KMeans function used for the power readings model.

4.3.2 Load Disaggregation Model

The load disaggregation model that the researchers used is the open-source toolkit from the Simple Load Disaggregation Library [6] that can be found in [7]. It was modified to cater to the datasets that were used. Some modifications include limiting the length of data of each appliance, the threshold of the data, and removing the dependency of the model to the timestamp of each data. The timestamp was removed because only one setup was used to collect data for the researchers' own dataset, as compared to other publicly available datasets that use 1 submeter for each appliance [8].

According to [4], the time complexity of the Combinatorial Optimization (CO) model is better than the Factorial Hidden Markov Model (FHMM), with $O(TK^N)$ and $O(TK^{2N})$ respectively. And as per the accuracy metrics, both models achieved almost the same score. For this reason, the researchers decided to use CO as the load disaggregation model for this project.

Once the training and test data is ready, from the previous steps, it can now be used to train the CO model. Before calling the train function, a list of appliances needs to be declared. The function requires the dataframe and list of appliances as inputs. After training the model, it needs to be saved and then the load disaggregation function can be called, with the test data as the input. The output of the load disaggregation function will be an array that contains the prediction of the model as to what appliances are there in the input of the function.

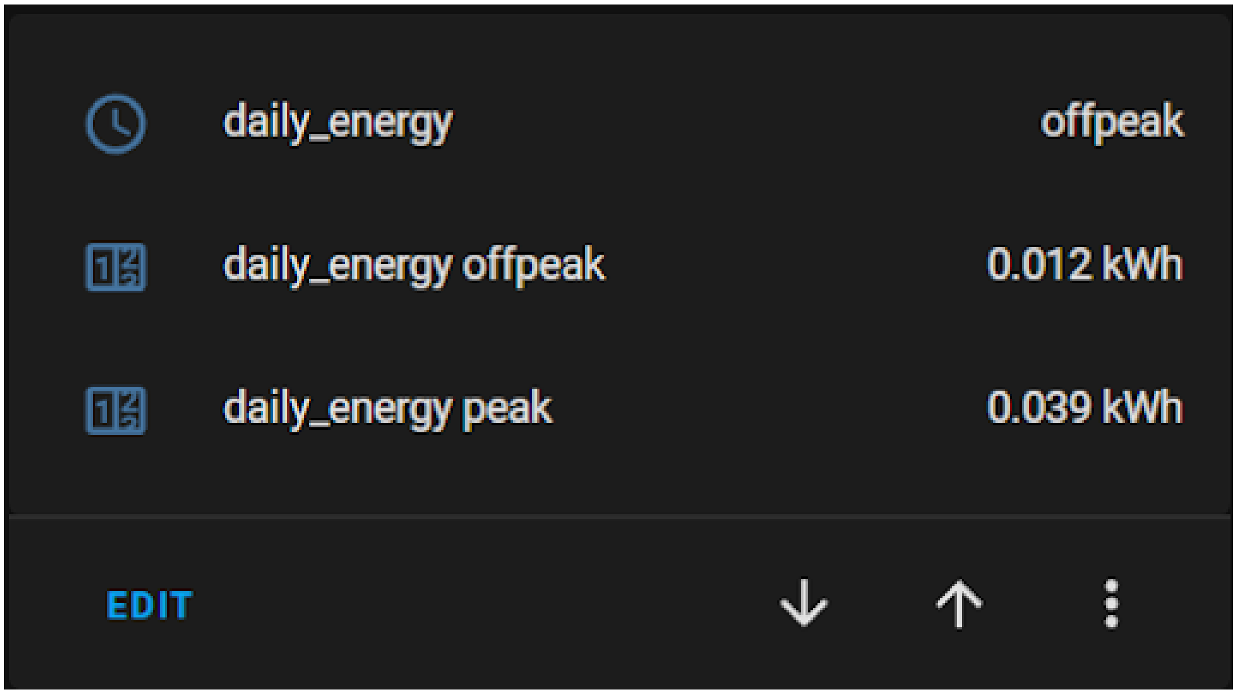
The test and prediction data are then labeled as a preparation for the functions that will compute for the accuracy, precision, recall, and the F1-score of the output. Python's sklearn library also has built-in functions for accuracy_score and classification_report. The classification_report output contains the precision, recall, and F1-score of each combination and the overall accuracy of the prediction. The accuracy score is computed by comparing the predicted set of labels for a sample to a set of labels indicated as the ground truth or correct labels. In this instance, the set of predicted labels must exactly match the compared correct labels since the model is designed to solve a Multiclass Classification task. Accordingly, Multiclass Classification implied that each sample is assigned to only one label.

4.4 Consumption Prediction (Descriptive and Predictive Data Analysis)

Consumption prediction was based on the *Tasmota Energy Today* readings shown in Figure 4.12. ***Custom sensors***, which are shown in Figure 4.11, were then created in Home Assistant OS to track the energy consumption based on the peak and offpeak times on the Meralco website as seen in Figure 4.16. The custom sensor utilizes a *utility meter* in Home Assistant alongside a ***custom automation*** that automatically switches between peak and off-peak times based on the Meralco site. The custom automation adjusts the prediction to peak rates from 8am to 9pm during Mondays to Saturdays and from 6pm to 8pm on Sundays. The predicted cost for the day, that is shown in Figure 4.13, takes advantage of this custom automation to adjust the appropriate tariff based on the previously mentioned peak and offpeak times. As an output, the predicted cost in Philippine Peso is displayed through *Telegram* as a message or reminder every 11:59 pm. A custom bot and integration in Home Assistant was designed for this to be implemented. Multiple users can subscribe to ***ha-tasmo*** bot if they wish to receive the predicted cost that day. This feature of multiple users being able to subscribe to the bot is shown in Figures 4.14 and 4.15.

For the peak and off peak rates, Wet Season (July - December) was used for the computation. The rates were also rounded to the nearest hundredths.

The researchers also used the created model to predict the cost for one day as shown in Figure 4.17. Based on the Power readings dataset, the mean of the energy consumption of each appliance was used to calculate the total cost. These values were then scaled to 24 hours to simulate a whole day of usage. The computed values were then multiplied to the peak and off peak rates with the appropriate ratios to reflect the time period (Peak: 13/24 hrs, Off-peak: 11/24 hrs).

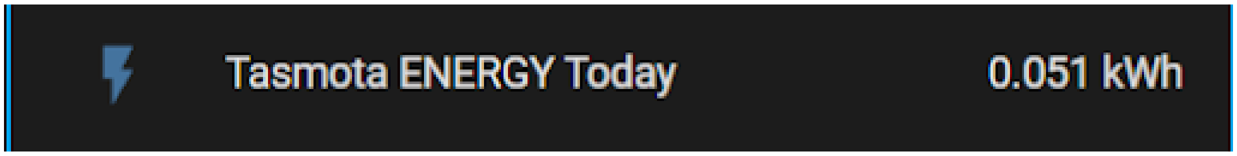


daily_energy offpeak

daily_energy offpeak 0.012 kWh

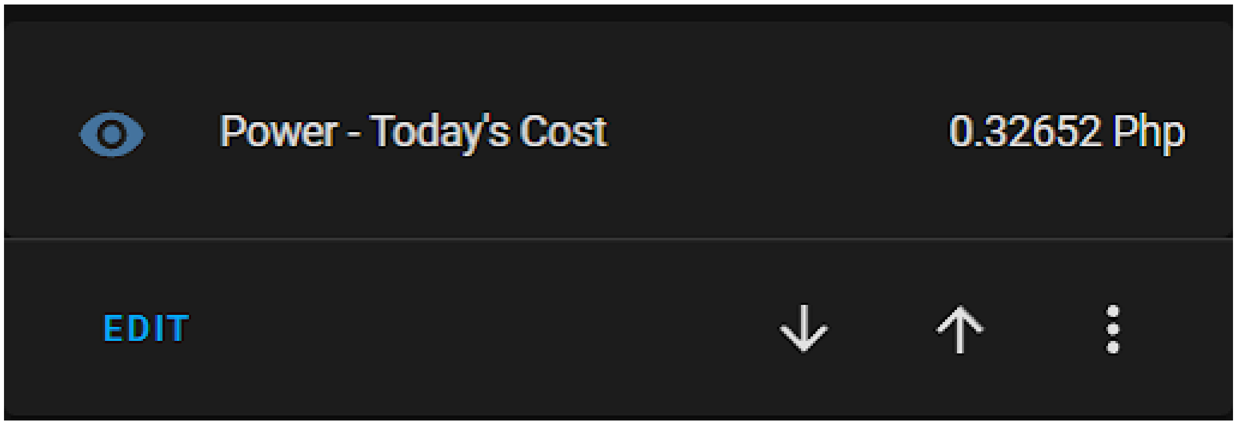
daily_energy peak 0.039 kWh

Figure 4.11: Daily Energy Custom Sensor Output



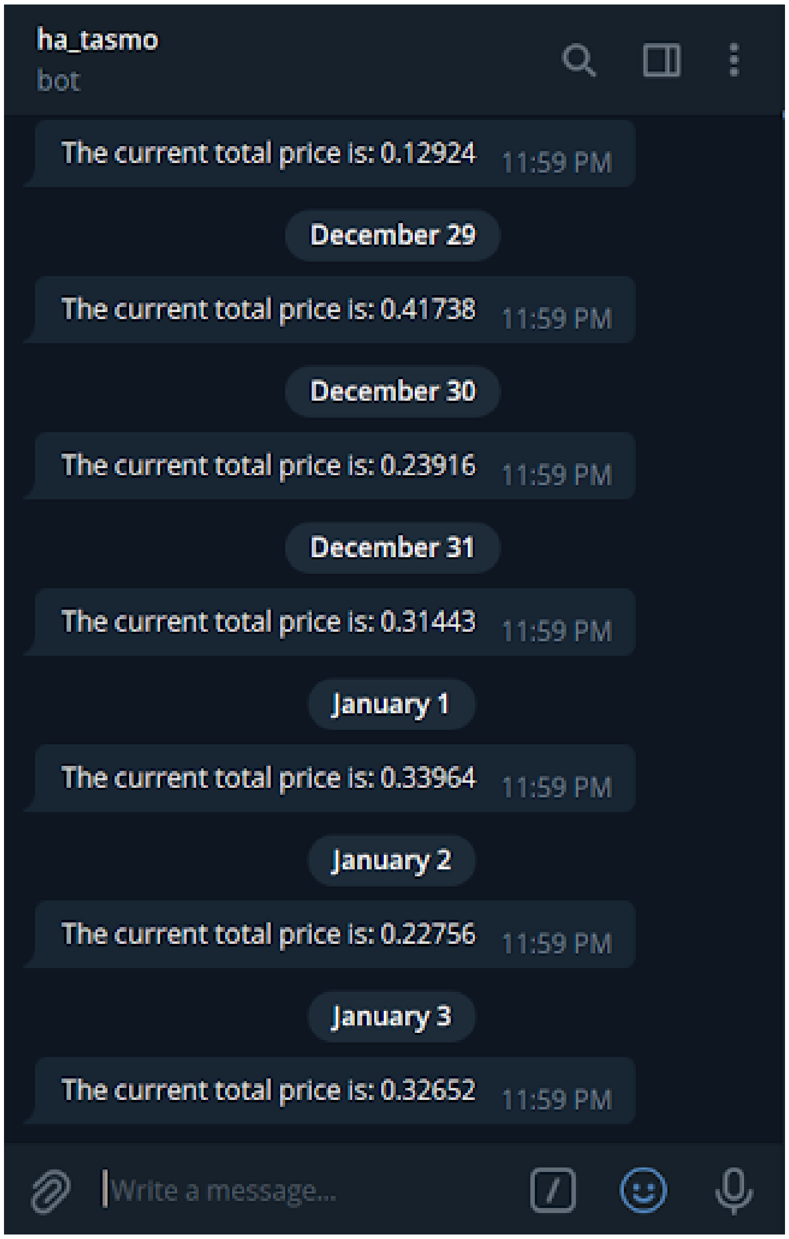
Tasmota ENERGY Today 0.051 kWh

Figure 4.12: Tasmota Energy Reading



Power - Today's Cost 0.32652 Php

Figure 4.13: Daily Energy Consumption Cost


ha_tasmo
bot

The current total price is: 0.12924 11:59 PM

December 29

The current total price is: 0.41738 11:59 PM

December 30

The current total price is: 0.23916 11:59 PM

December 31

The current total price is: 0.31443 11:59 PM

January 1

The current total price is: 0.33964 11:59 PM

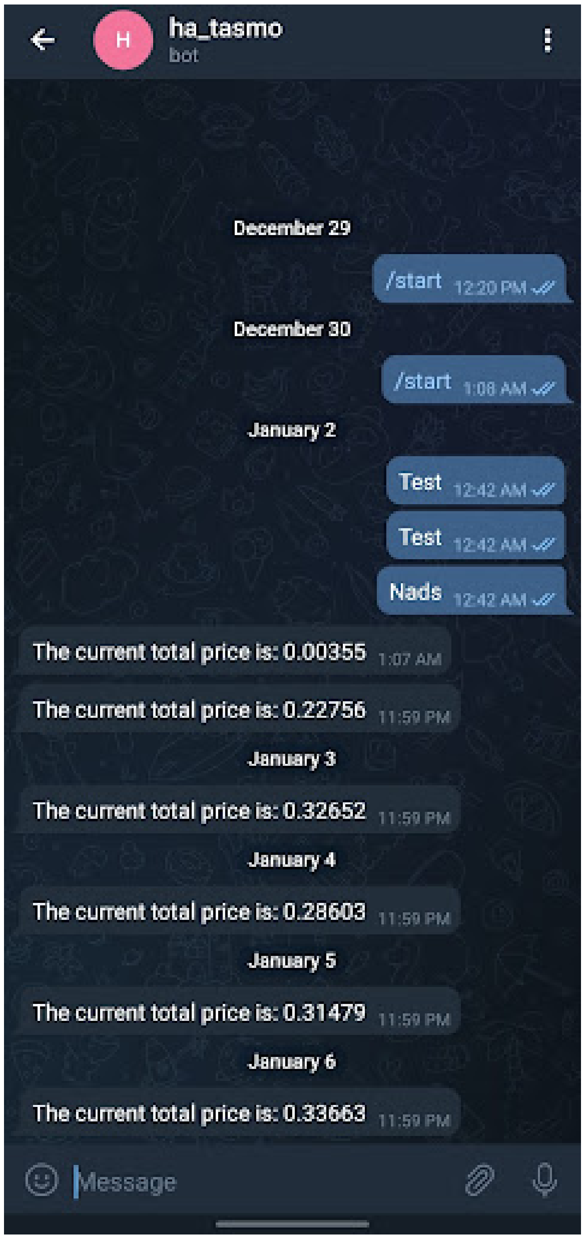
January 2

The current total price is: 0.22756 11:59 PM

January 3

The current total price is: 0.32652 11:59 PM

Figure 4.14: Telegram Bot Predicted Cost Message (Russ)



ha_tasmo
bot

December 29

/start 12:20 PM

December 30

/start 1:08 AM

January 2

Test 12:42 AM

Test 12:42 AM

Nads 12:42 AM

The current total price is: 0.00355 1:07 AM

The current total price is: 0.22756 11:59 PM

January 3

The current total price is: 0.32652 11:59 PM

January 4

The current total price is: 0.28603 11:59 PM

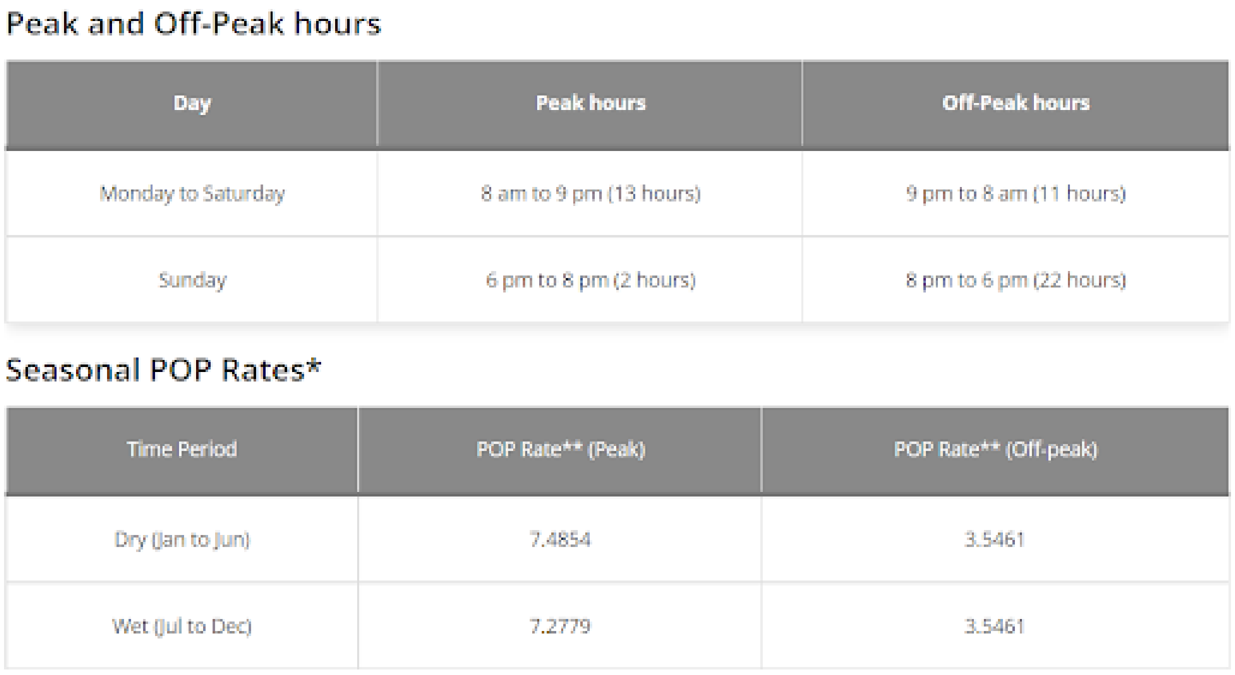
January 5

The current total price is: 0.31479 11:59 PM

January 6

The current total price is: 0.33663 11:59 PM

Figure 4.15: Telegram Bot Predicted Cost Message (Nadine)



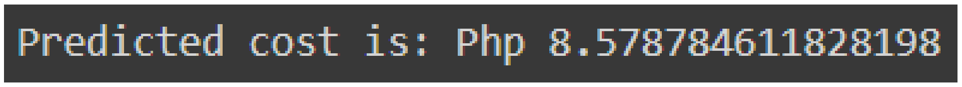
Peak and Off-Peak hours

Day	Peak hours	Off-Peak hours
Monday to Saturday	8 am to 9 pm (13 hours)	9 pm to 8 am (11 hours)
Sunday	6 pm to 8 pm (2 hours)	8 pm to 6 pm (22 hours)

Seasonal POP Rates*

Time Period	POP Rate** (Peak)	POP Rate** (Off-peak)
Dry (Jan to Jun)	7.4854	3.5461
Wet (Jul to Dec)	7.2779	3.5461

Figure 4.16: Meralco Tariff Rates



```
Predicted cost is: Php 8.578784611828198
```

Figure 4.17: Model Predicted Cost

Chapter 5

Results and Analysis

5.1 Load Identfication

There were three sets of data that were used to train the model and test its functionality, the UK-DALE dataset, the researchers' dataset but using only the power readings, and the combined values of all of the features of the researchers' dataset using PCA.

As mentioned in the discussion of the model in the methodology, the ***accuracy_score*** and ***classification_report*** functions from ***sklearn*** were used to check for the performance of the whole algorithm. When using the ***classification_report*** function, it returns the precision and recall of each class that is then used for the F1 score and the overall accuracy of the model. Precision is the score of the model computed using the number of correct predictions over the overall number of predictions. And recall is computed using the number of all true positives over the combination of the true and false negatives. The F1 score is commonly and must be used when dealing with imbalanced data that have no equal distribution in each class [14]. For this case, imbalance data may occur if there are a lot of zero data points in each cluster and from splitting the data into train and test sets.

The plots that are included in this chapter will show the test data used for prediction and the prediction result of the model.

5.1.1 UK-DALE Dataset

Shown in Figure 5.1 is the plot of the power readings of the UK-DALE test dataset in each index of data points. The data points in the plot were limited to only 100 indices for better visualization of the data.

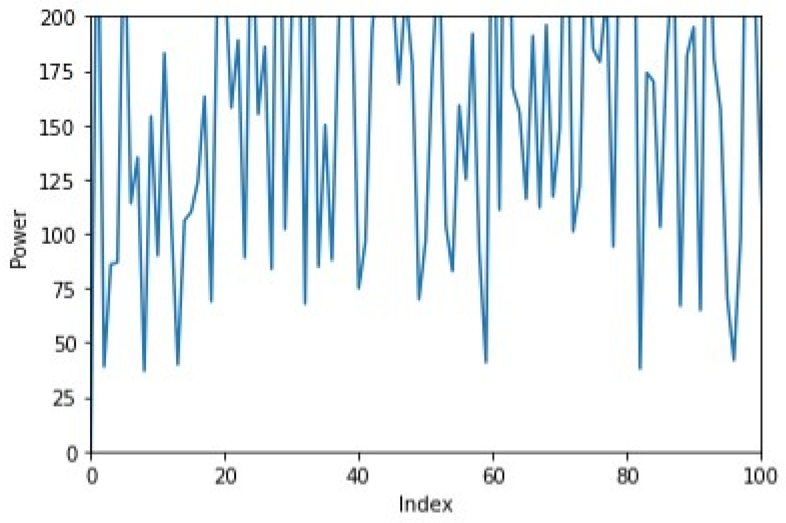


Figure 5.1: UK-DALE Test Data

Since the timestamp of each power readings was not used, you will see below are the colored plots, in Figure 5.2 that represent each channel or appliance present in each index and its power consumption of the y-axis. If the value of the channel is at 0 on the y-axis, it means that the appliance is off or not in use. The data points in the plot were limited to only 100 indices for better visualization of the data.

Lastly, in Figure 5.3, you will see the score of each label or combination of appliances scores in all of the metrics used. The label for each appliances combinations is shown on the leftmost column. Macro average refers to the average F1 score that did not take into consideration the proportion for each label in the dataset. On the other hand, weighted average is the average F1 score that considered the proportion for each label in the dataset. The total accuracy of the model is also shown below.

5.1.2 Created Dataset

How the created dataset of the researchers' performed in the model in predicting the appliances present and their consumption is presented in this section.

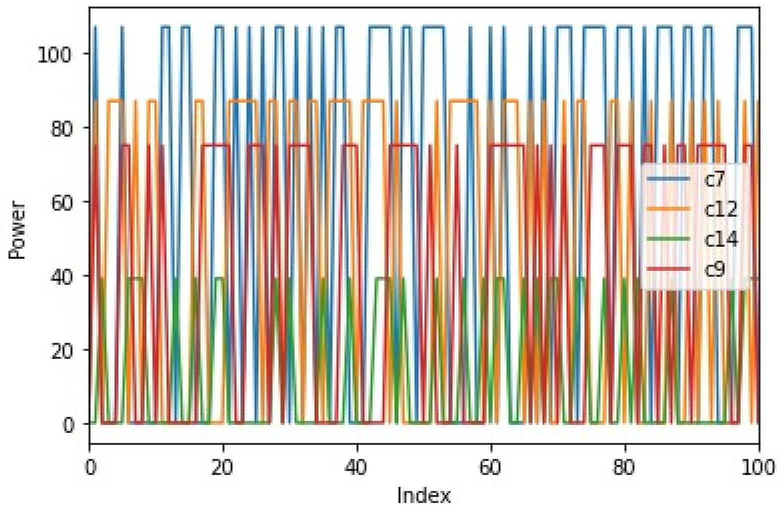
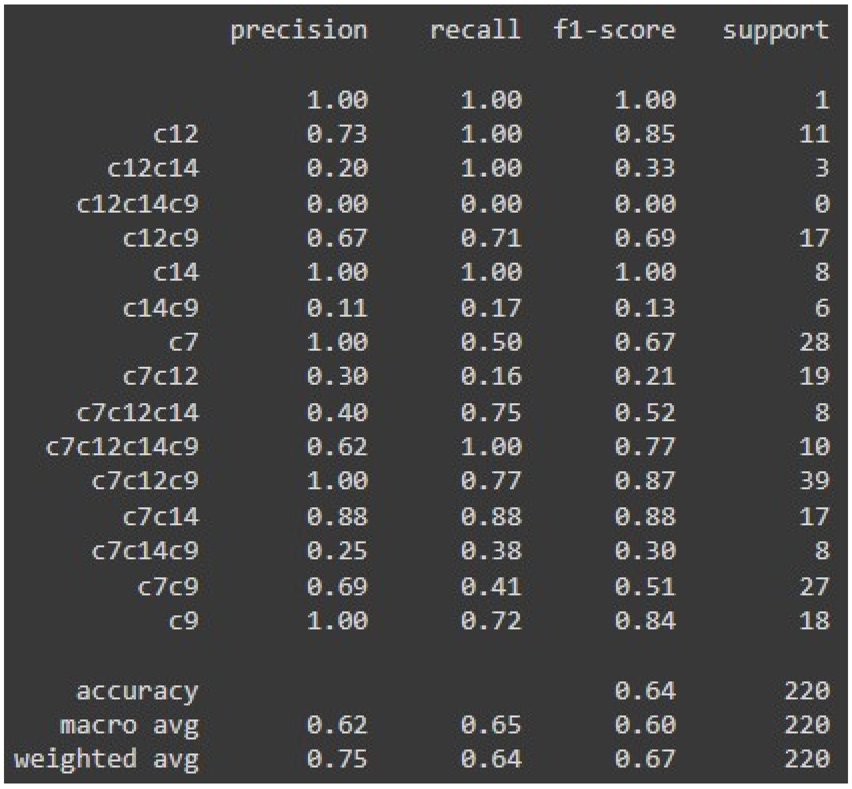


Figure 5.2: UK-DALE Prediction Results



```
              precision    recall  f1-score   support

                   1.00      1.00      1.00         1
         c12       0.73      1.00      0.85        11
      c12c14       0.20      1.00      0.33         3
    c12c14c9       0.00      0.00      0.00         0
       c12c9       0.67      0.71      0.69        17
         c14       1.00      1.00      1.00         8
        c14c9      0.11      0.17      0.13         6
          c7       1.00      0.50      0.67        28
       c7c12       0.30      0.16      0.21        19
    c7c12c14       0.40      0.75      0.52         8
  c7c12c14c9       0.62      1.00      0.77        10
     c7c12c9       1.00      0.77      0.87        39
       c7c14       0.88      0.88      0.88        17
      c7c14c9      0.25      0.38      0.30         8
        c7c9       0.69      0.41      0.51        27
          c9       1.00      0.72      0.84        18

    accuracy                           0.64       220
   macro avg       0.62      0.65      0.60       220
weighted avg       0.75      0.64      0.67       220
```

Figure 5.3: UK-DALE Classification Report

5.1.2.1 Power Readings

Shown in figure 5.4 is the plot of the test dataset of the power readings from the researchers' own dataset. You can see that the data points are lesser compared to the UK-DALE plot as it was only collected in a much shorter period. And in Figure 5.5, the colored plots represent the appliances present in the dataset. The y-axis is the power readings and the x-axis is the index of each data point.

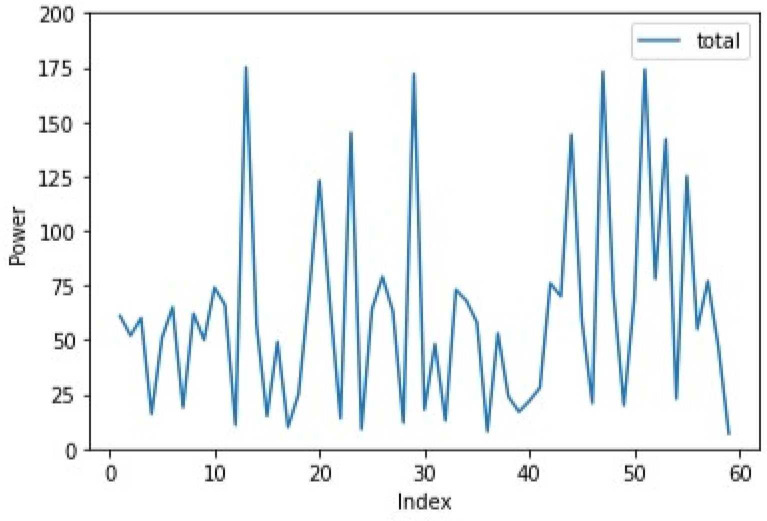


Figure 5.4: Created Dataset Power Readings Test Data

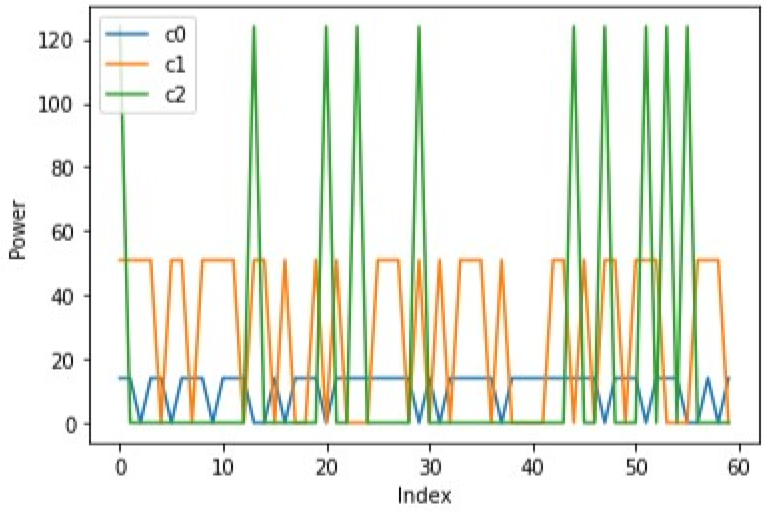
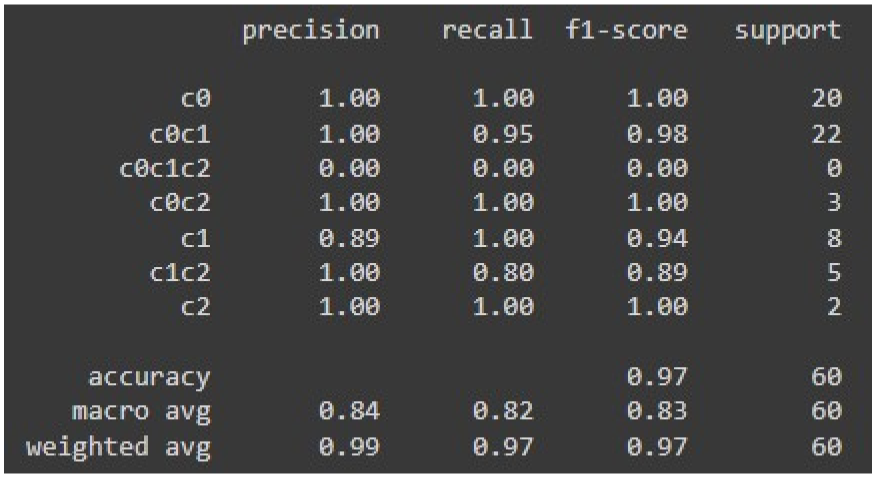


Figure 5.5: Created Dataset Power Readings Prediction Results

Similar to the UK-DALE dataset section, Figure 5.6 shows the score of each label in all of the metrics used.



```
              precision    recall  f1-score   support

          c0       1.00      1.00      1.00        20
        c0c1       1.00      0.95      0.98        22
      c0c1c2       0.00      0.00      0.00         0
        c0c2       1.00      1.00      1.00         3
          c1       0.89      1.00      0.94         8
        c1c2       1.00      0.80      0.89         5
          c2       1.00      1.00      1.00         2

    accuracy                           0.97        60
   macro avg       0.84      0.82      0.83        60
weighted avg       0.99      0.97      0.97        60
```

Figure 5.6: Created Dataset Power Readings Classification Report

5.1.2.2 PCA

The difference of the plots in this section from the two previous ones is that instead of the power readings the combined features is the y-axis. Figure 5.7 is the plot of the test data and the prediction results is shown in Figure 5.8.

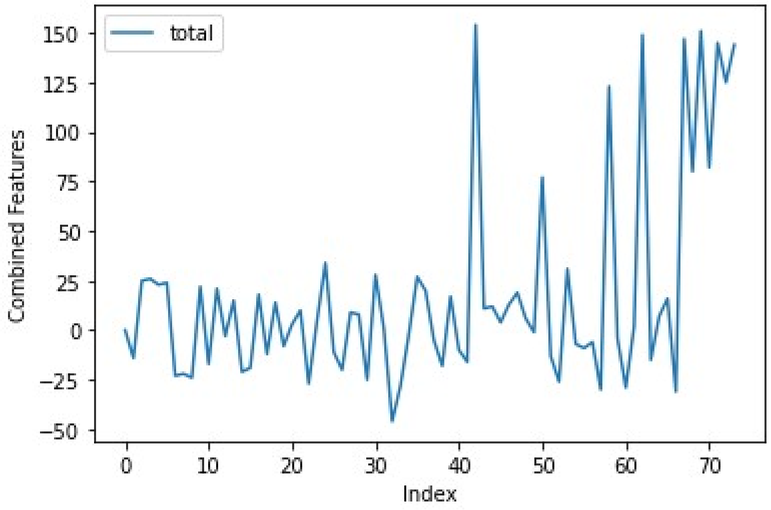


Figure 5.7: Created Dataset PCA Test Data

The prediction result scores are shown in Figure 5.9 for each combination of appliances. Accuracy is lower compared to the model that used the power readings only. The reason for this

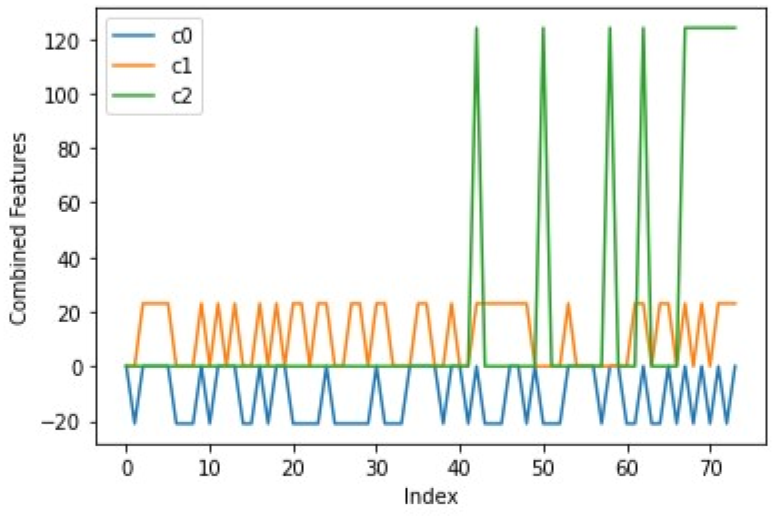
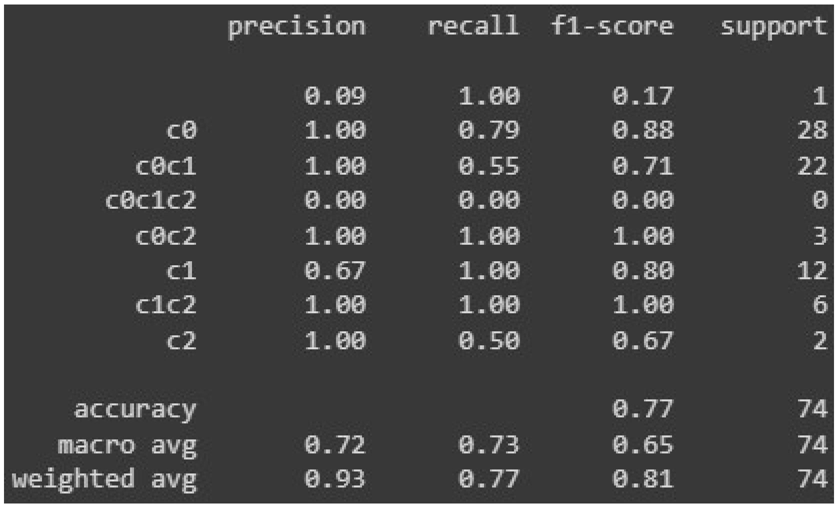


Figure 5.8: Created Dataset PCA Prediction Result

is that the model which utilized PCA combined more features into a single component and this process caused a loss of data.



```
              precision    recall  f1-score   support

                   0.09      1.00      0.17         1
          c0       1.00      0.79      0.88        28
        c0c1       1.00      0.55      0.71        22
      c0c1c2       0.00      0.00      0.00         0
        c0c2       1.00      1.00      1.00         3
          c1       0.67      1.00      0.80        12
        c1c2       1.00      1.00      1.00         6
          c2       1.00      0.50      0.67         2

    accuracy                           0.77        74
   macro avg       0.72      0.73      0.65        74
weighted avg       0.93      0.77      0.81        74
```

Figure 5.9: Created Dataset PCA Classification Report

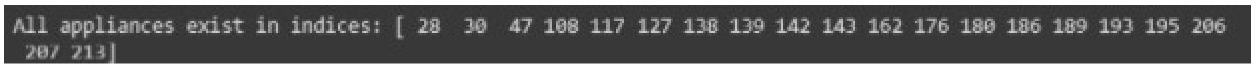
5.2 Appliance Presence Prediction

One of the main objectives of the model was to be able to identify which appliances were present in a given unlabeled dataset. In terms of real-world usage, the model, once implemented, should be able to distinguish and label the specific appliances from the extension outlet containing simultaneously connected appliances.

As an output, the researchers were able to identify and show the presence of each appliance based on a specific index from the given dataset.

5.2.1 UK-DALE Dataset

As seen in Figure 5.10, all appliances were detected to be present in certain indices.



```
All appliances exist in indices: [ 28  30  47 108 117 127 138 139 142 143 162 176 180 186 189 193 195 206
 207 213]
```

Figure 5.10: UK Dale Dataset Presence Detection for All Appliances

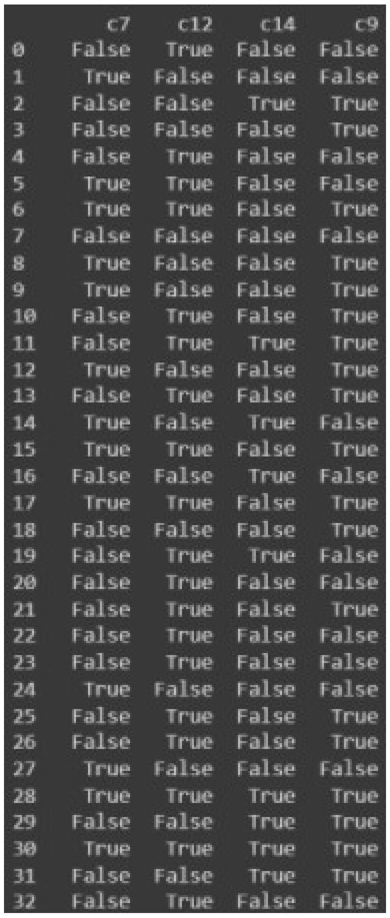
These results show that for certain indices in a given dataset, the created model was able to individually detect each appliance. The ones seen in Figure 5.10 and 5.11 were set by the researcher to detect the presence of all 4 appliances when they are connected simultaneously. More specifically, the indices shown in Figure 5.10 are the indices wherein all 4 appliances are simultaneously connected. Figure 5.11 shows which specific appliance is present in a given index. As an example, in indices 28 and 30, all appliances are detected by the model as seen in both Figures 5.10 and 5.11.

The model can also be used to detect the presence of only one appliance. In Figure 5.12, the researchers set the detection to be limited to only one appliance.

Figures 5.13, 5.14 and 5.11 show that the model is able to detect each of the 4 appliances individually and separately.

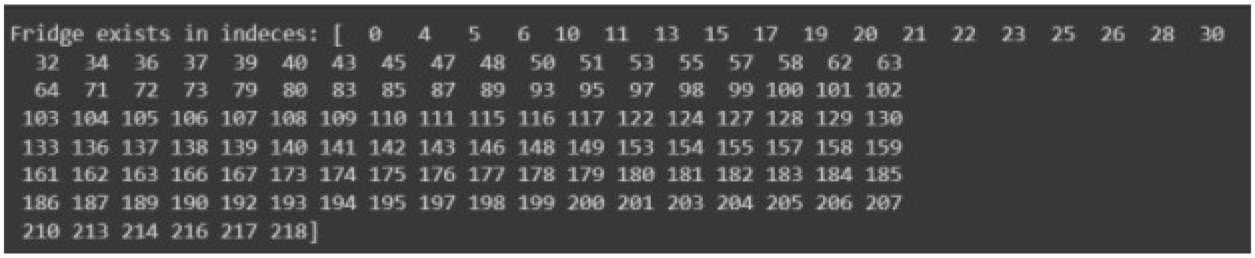
5.2.2 Created Dataset

How the created dataset of the researchers' performed in determining the indices on which each appliance is present is presented in this section.



```
      c7    c12    c14     c9
0   False   True  False  False
1    True  False  False  False
2   False  False   True   True
3   False  False  False   True
4   False   True  False  False
5    True   True  False  False
6    True   True  False   True
7   False  False  False  False
8    True  False  False   True
9    True  False  False   True
10  False   True  False   True
11  False   True   True   True
12   True  False  False   True
13  False   True  False   True
14   True  False   True  False
15   True   True  False   True
16  False  False   True  False
17   True   True  False   True
18  False  False  False   True
19  False   True   True  False
20  False   True  False  False
21  False   True  False   True
22  False   True  False  False
23  False   True  False  False
24   True  False  False  False
25  False   True  False   True
26  False   True  False   True
27   True  False  False  False
28   True   True   True   True
29  False  False   True   True
30   True   True   True   True
31  False  False   True   True
32  False   True  False  False
```

Figure 5.11: UK Dale Dataset Presence Detection for All Appliances

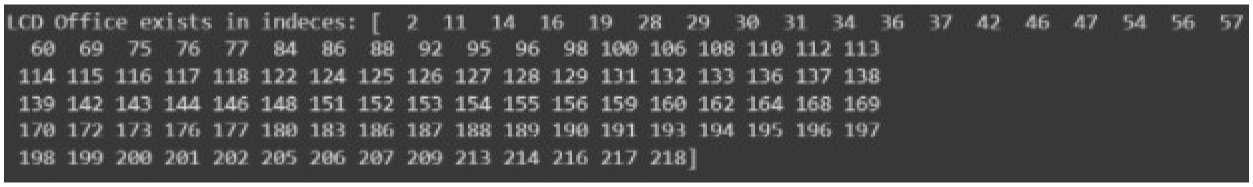


```
Fridge exists in indeces: [  0   4   5   6  10  11  13  15  17  19  20  21  22  23  25  26  28  30
  32  34  36  37  39  40  43  45  47  48  50  51  53  55  57  58  62  63
  64  71  72  73  79  80  83  85  87  89  93  95  97  98  99 100 101 102
 103 104 105 106 107 108 109 110 111 115 116 117 122 124 127 128 129 130
 133 136 137 138 139 140 141 142 143 146 148 149 153 154 155 157 158 159
 161 162 163 166 167 173 174 175 176 177 178 179 180 181 182 183 184 185
 186 187 189 190 192 193 194 195 197 198 199 200 201 203 204 205 206 207
 210 213 214 216 217 218]
```

Figure 5.12: Single Appliance Presence Detection (Fridge)

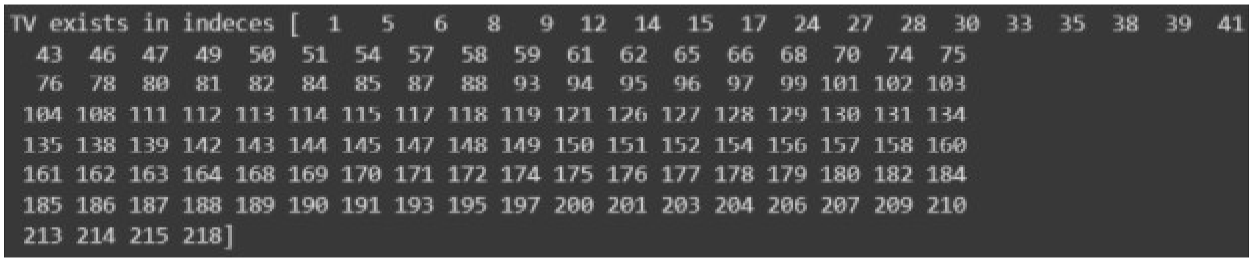
5.2.2.1 Power Readings

Similarly with what was shown in Figures 5.10 and 5.11, the researchers set the detection to all appliances being connected simultaneously in Figures 5.16 and 5.17. It can be seen based on the Figures that all 3 appliances were only detected to be connected simultaneously in index 0.



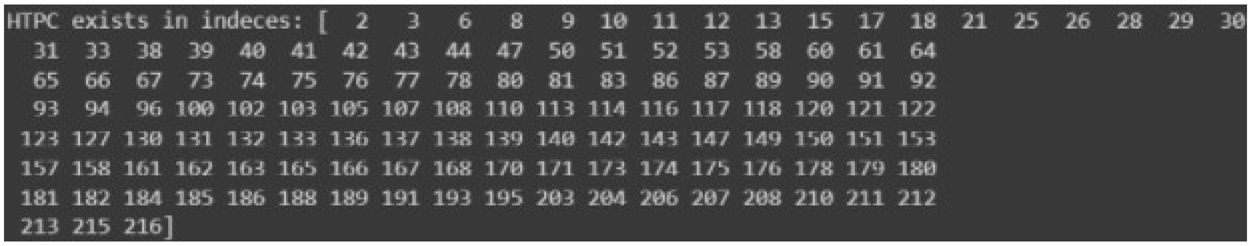
```
LCD Office exists in indeces: [  2  11  14  16  19  28  29  30  31  34  36  37  42  46  47  54  56  57
  60  69  75  76  77  84  86  88  92  95  96  98 100 106 108 110 112 113
 114 115 116 117 118 122 124 125 126 127 128 129 131 132 133 136 137 138
 139 142 143 144 146 148 151 152 153 154 155 156 159 160 162 164 168 169
 170 172 173 176 177 180 183 186 187 188 189 190 191 193 194 195 196 197
 198 199 200 201 202 205 206 207 209 213 214 216 217 218]
```

Figure 5.13: Single Appliance Presence Detection (LCD Office)



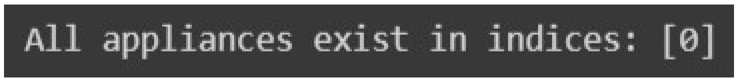
```
TV exists in indeces [  1   5   6   8   9  12  14  15  17  24  27  28  30  33  35  38  39  41
  43  46  47  49  50  51  54  57  58  59  61  62  65  66  68  70  74  75
  76  78  80  81  82  84  85  87  88  93  94  95  96  97  99 101 102 103
 104 108 111 112 113 114 115 117 118 119 121 126 127 128 129 130 131 134
 135 138 139 142 143 144 145 147 148 149 150 151 152 154 156 157 158 160
 161 162 163 164 168 169 170 171 172 174 175 176 177 178 179 180 182 184
 185 186 187 188 189 190 191 193 195 197 200 201 203 204 206 207 209 210
 213 214 215 218]
```

Figure 5.14: Single Appliance Presence Detection (TV)



```
HTPC exists in indeces: [  2   3   6   8   9  10  11  12  13  15  17  18  21  25  26  28  29  30
  31  33  38  39  40  41  42  43  44  47  50  51  52  53  58  60  61  64
  65  66  67  73  74  75  76  77  78  80  81  83  86  87  89  90  91  92
  93  94  96 100 102 103 105 107 108 110 113 114 116 117 118 120 121 122
 123 127 130 131 132 133 136 137 138 139 140 142 143 147 149 150 151 153
 157 158 161 162 163 165 166 167 168 170 171 173 174 175 176 178 179 180
 181 182 184 185 186 188 189 191 193 195 203 204 206 207 208 210 211 212
 213 215 216]
```

Figure 5.15: Single Appliance Presence Detection (HTPC)



```
All appliances exist in indices: [0]
```

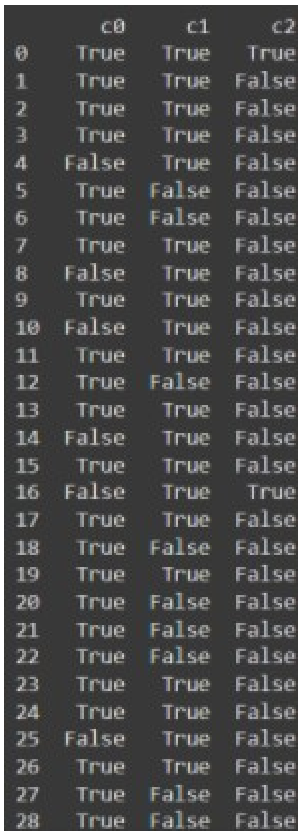
Figure 5.16: Power Readings Presence Detection for All Appliances

The researchers also set the detection capability to only a single appliance. The results are shown in Figures 5.18, 5.19, and 5.20.

5.2.2.2 PCA

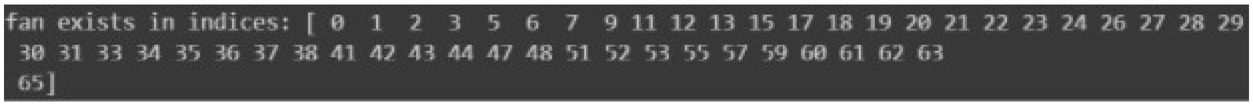
The same method of appliance detection was applied to the model utilizing PCA. Seen in Figure 5.21 are the indices where all appliances were detected to be simultaneously present.

Single appliance detection was also done by the researchers in the model utilizing PCA. The results can be seen in Figures 5.22, 5.23, and 5.24.



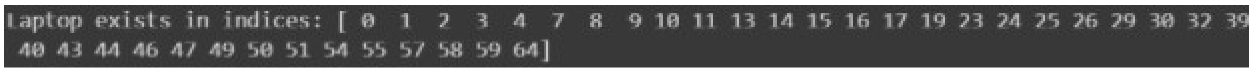
```
       c0     c1     c2
0    True   True   True
1    True   True  False
2    True   True  False
3    True   True  False
4   False   True  False
5    True  False  False
6    True  False  False
7    True   True  False
8   False   True  False
9    True   True  False
10  False   True  False
11   True   True  False
12   True  False  False
13   True   True  False
14  False   True  False
15   True   True  False
16  False   True   True
17   True   True  False
18   True  False  False
19   True   True  False
20   True  False  False
21   True  False  False
22   True  False  False
23   True   True  False
24   True   True  False
25  False   True  False
26   True   True  False
27   True  False  False
28   True  False  False
```

Figure 5.17: Power Readings Presence Detection for All Appliances



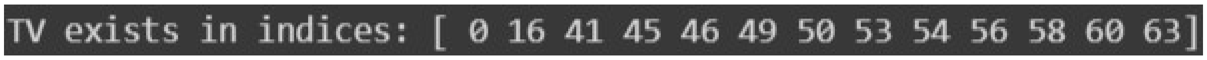
```
fan exists in indices: [ 0  1  2  3  5  6  7  9 11 12 13 15 17 18 19 20 21 22 23 24 26 27 28 29
 30 31 33 34 35 36 37 38 41 42 43 44 47 48 51 52 53 55 57 59 60 61 62 63
 65]
```

Figure 5.18: Single Appliance Presence Detection (Fan)



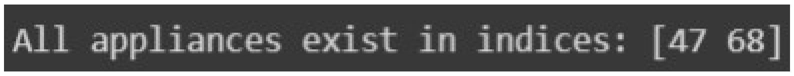
```
Laptop exists in indices: [ 0  1  2  3  4  7  8  9 10 11 13 14 15 16 17 19 23 24 25 26 29 30 32 39
 40 43 44 46 47 49 50 51 54 55 57 58 59 64]
```

Figure 5.19: Single Appliance Presence Detection (Laptop)



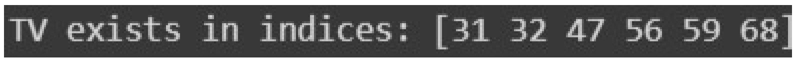
```
TV exists in indices: [ 0 16 41 45 46 49 50 53 54 56 58 60 63]
```

Figure 5.20: Single Appliance Presence Detection (TV)



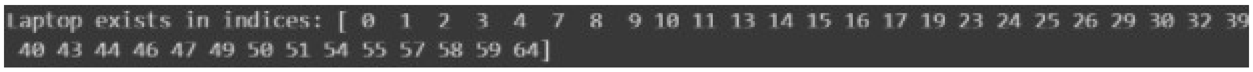
```
All appliances exist in indices: [47 68]
```

Figure 5.21: PCA Presence Detection for All Appliances



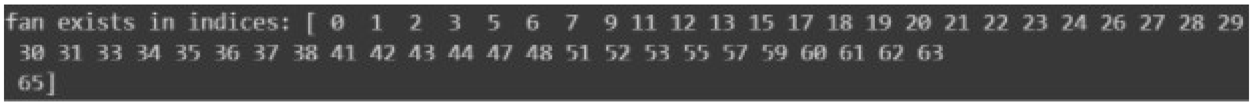
```
TV exists in indices: [31 32 47 56 59 68]
```

Figure 5.22: Single Appliance Presence Detection (TV)



```
Laptop exists in indices: [ 0  1  2  3  4  7  8  9 10 11 13 14 15 16 17 19 23 24 25 26 29 30 32 39
 40 43 44 46 47 49 50 51 54 55 57 58 59 64]
```

Figure 5.23: Single Appliance Presence Detection (Laptop)



```
fan exists in indices: [ 0  1  2  3  5  6  7  9 11 12 13 15 17 18 19 20 21 22 23 24 26 27 28 29
 30 31 33 34 35 36 37 38 41 42 43 44 47 48 51 52 53 55 57 59 60 61 62 63
 65]
```

Figure 5.24: Single Appliance Presence Detection (Fan)

The results shown in Figures 5.2, 5.5 and 5.8 are presented in a usable form through the Appliance Presence Prediction. Hence, the Appliance Presence Prediction section is an extension of the Prediction shown in the graphs by indicating the specific indices where each appliance is present based on the predicted power reading.

In summary, the Appliance Presence Prediction capability of the created model shows which appliances are working based on a specific index. It also shows that the model is capable of determining if the appliance was actually present at a given index. Consequently, the created model can give the index where the appliance is detected to be present. This shows the versatility of the model as it can detect the presence of each appliance whether it be simultaneously or separately. It can adapt based on the user's needs when it comes to Appliance Presence Prediction as well.

5.3 Consumption Prediction

The descriptive analysis provides a way such that the user is able to understand the data presented to them, data that are processed from the gathered measurements from the device. The Smart Energy Meter has a built-in feature for this. It presents the measurement of voltage, current, wattage, and power consumption of the device connected to it via the smartphone application.

The predictive data analysis was centered around the goal of reducing electricity cost through electricity consumption prediction based on the consumer's usage patterns. The prediction used the appliance electricity parameter, energy consumption.

The researchers created a custom sensor in Home Assistant to be able to predict the cost of the energy consumption of all the appliances connected. The custom sensor was based on the *Tasmota Energy Today Readings.* The researchers created a custom sensor that was capable of giving an estimated cost based on the Peak and Off peak rates stated in the Meralco Website. Moreover, the researchers were able to create a custom automation such that the tariff costs were properly followed based on the time period stated in the Meralco Website. Besides the peak and off peak hours, Meralco implements Seasonal POP (Peak/Off peak) rates. The Wet Season (Jul - Dec) POP rates were used for the custom sensor. The full integration of both the custom sensor and custom automation resulted in being able to separate the energy consumption for peak and off peak hours as seen in Figure 5.26. The panel also indicated whether the consumption tracking is at its peak or off peak period. Figure 5.27 shows the automated switching of the tariff rates from peak to off peak and vice versa. Furthermore, according to the Meralco Website, there are different peak and off peak hours on Sundays as compared to other days of the week. Another custom automation was created to accommodate this.

The predicted cost is shown in Figure 5.25. As an output, the researchers set up a custom bot in Telegram, which is a messaging application, that sends the predicted cost every 11:59 pm. Using Home Assistant Custom Automations, Sensors and Integrations, a Predicted Daily Cost is given through Telegram as shown in Figure 5.28. This Predicted Cost is based on the Tasmota Energy Today Readings. The readings are based on the length of use and total number of devices connected to the SONOFF POW R2. The tariff rates of the Predicted Daily Cost is based on the rates seen in the Meralco Website.

Based on the created dataset, the load disaggregation model was used for cost prediction also. The scaled whole-day values of the mean energy consumption of each appliance was used in the calculation of the predicted cost. The appropriate ratios to reflect the tariff time period were then applied such that it followed the near real time cost prediction design. This Predicted Consumption Cost, which used the machine learning algorithm, is shown in Figure 5.29. The Predicted Consumption Cost is based on the mean of the Predicted Power of each identified appliance in the created dataset. The mentioned Predicted Power is the output of the prediction done by the machine learning algorithm. The prediction assumes whole-day usage to give a future daily consumption prediction based on the number of devices identified.

As a result, the stated objective of providing a prediction of future consumption based on the recorded number of users, length of use, and devices used was met. Furthermore, Home Assistant features were utilized and used as a dashboard for the project's features. The Home Assistant Dashboard shows Smart Energy Readings and Predicted Daily Cost. Home Assistant Automations were also used to set-up the Telegram Bot which sends the predicted cost. This Dashboard can be viewed through Mobile Home Assistant as well. The Mobile Home Assistant Dashboard containing the Consumption Prediction and device readings are shown in Figures 5.30 and 5.31.

In summary, a near real time cost prediction system based on the user's usage patterns was successfully created. Utilizing the device parameter *Tasmota Energy Today*, alongside the integration of the *custom sensor* and *custom automations*, the system was able to provide and present data to a user in a convenient and understandable form through the messaging application, *Telegram* and Mobile Home Assistant. Using the created model, the descriptive analysis is implemented and explored even deeper. The system is not only capable of providing near real time cost estimation but estimations based on unlabeled power readings dataset as well. The system is also custom tailored for the Philippines as cost predictions are given in Philippine Peso and based on Meralco Tariff rates.

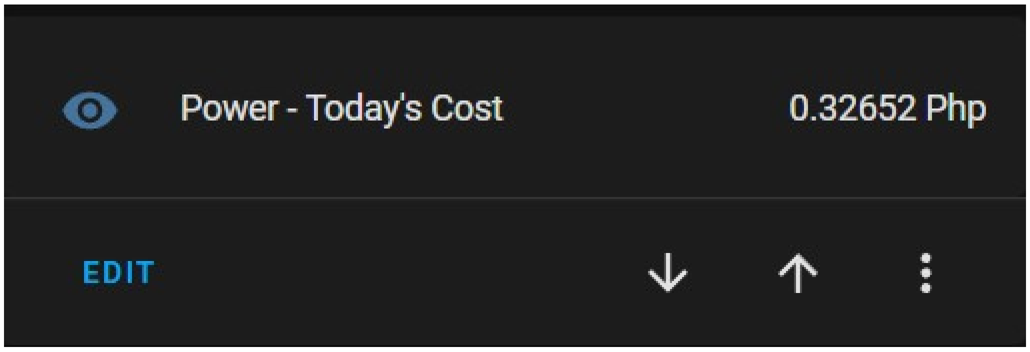
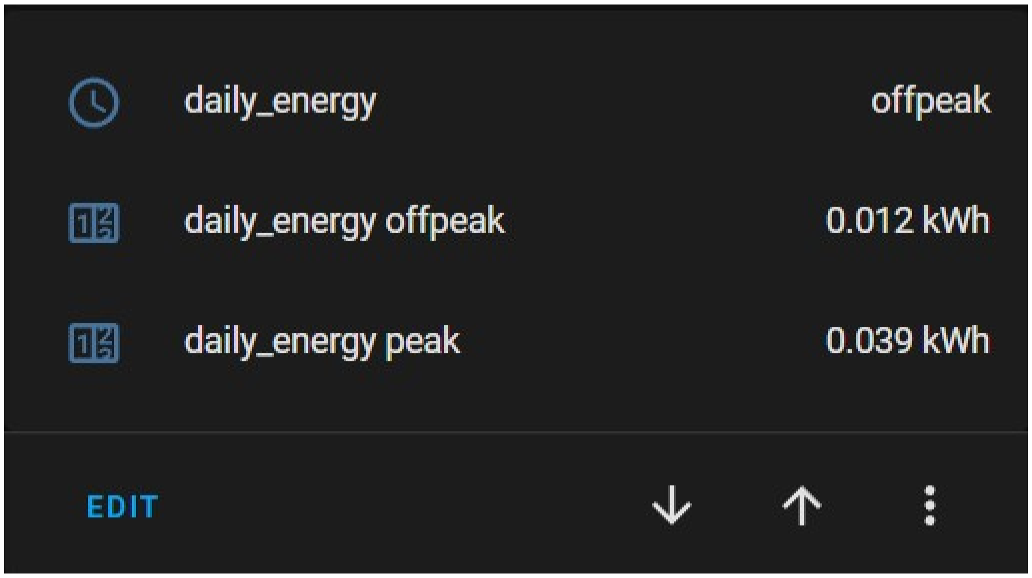
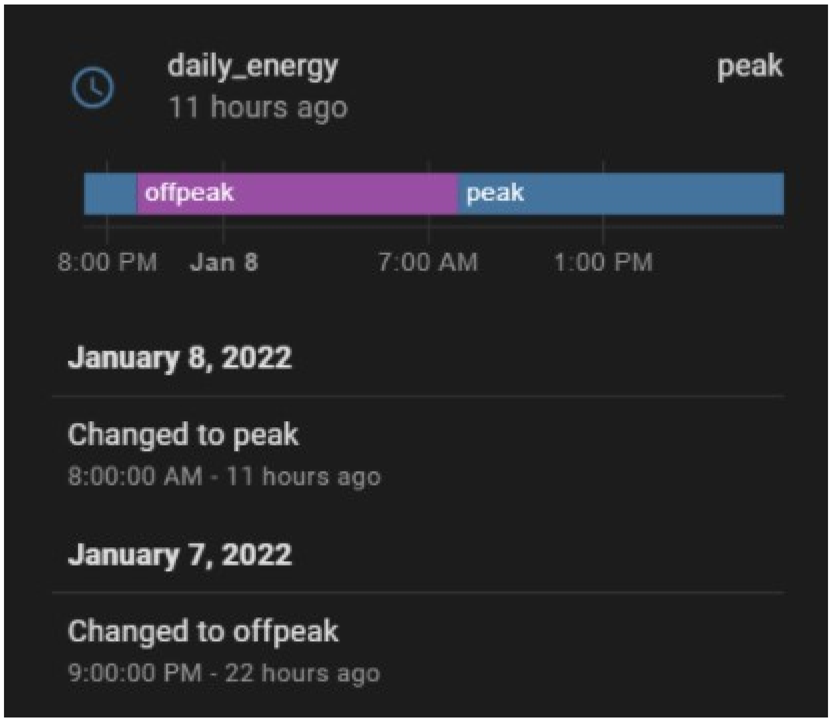

Power - Today's Cost 0.32652 Php

Figure 5.25: Predicted Cost for Today Sample



daily_energy	offpeak
daily_energy offpeak	0.012 kWh
daily_energy peak	0.039 kWh

Figure 5.26: Peak/Off-Peak Energy Consumption



daily_energy
11 hours ago
peak

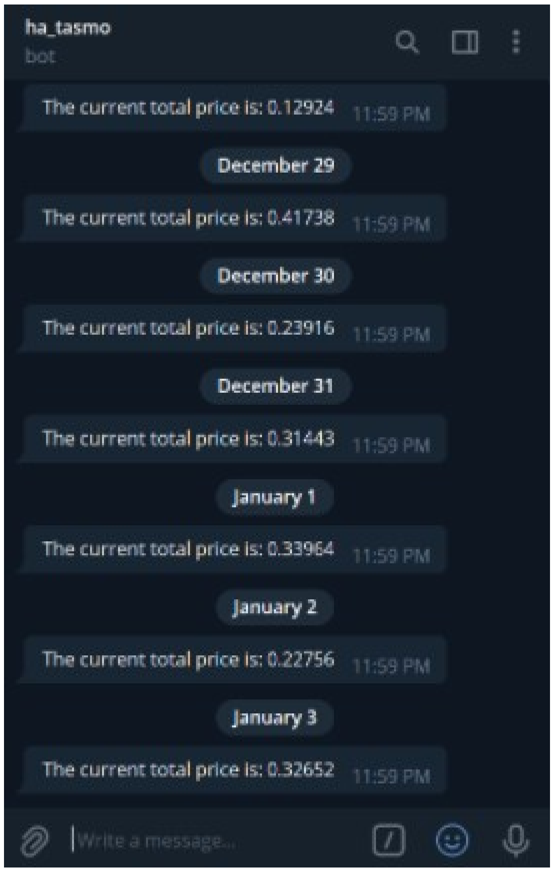
January 8, 2022

Changed to peak
8:00:00 AM - 11 hours ago

January 7, 2022

Changed to offpeak
9:00:00 PM - 22 hours ago

Figure 5.27: Custom Automation Peak/Off-Peak Mon-Sat



ha_tasmo
bot

The current total price is: 0.12924 11:59 PM

December 29

The current total price is: 0.41738 11:59 PM

December 30

The current total price is: 0.23916 11:59 PM

December 31

The current total price is: 0.31443 11:59 PM

January 1

The current total price is: 0.33964 11:59 PM

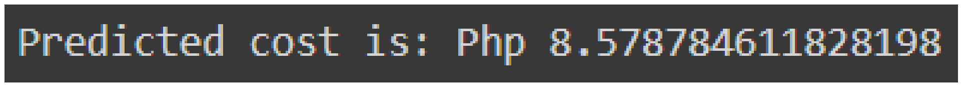
January 2

The current total price is: 0.22756 11:59 PM

January 3

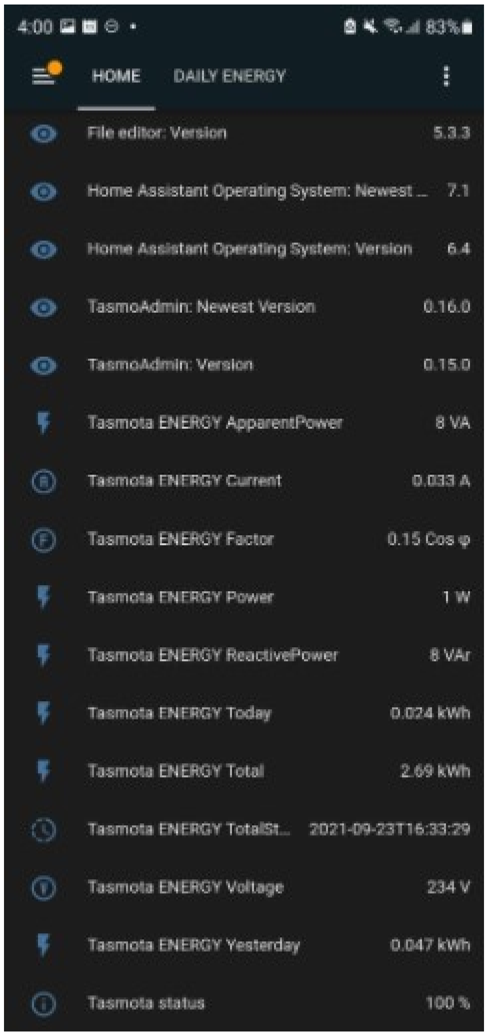
The current total price is: 0.32652 11:59 PM

Figure 5.28: Telegram Predicted Cost Bot



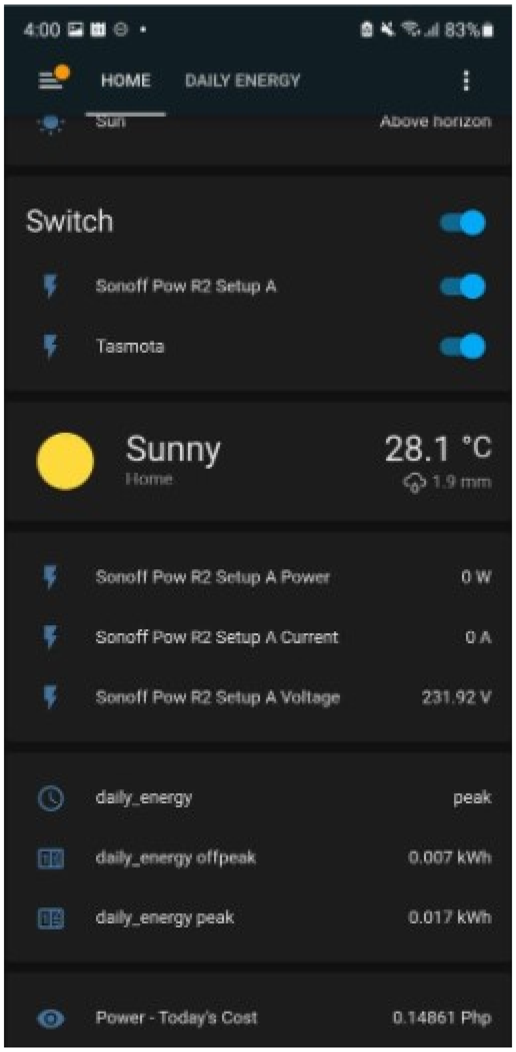
```
Predicted cost is: Php 8.578784611828198
```

Figure 5.29: Model Predicted Cost



File editor: Version	5.3.3
Home Assistant Operating System: Newest ...	7.1
Home Assistant Operating System: Version	6.4
TasmoAdmin: Newest Version	0.16.0
TasmoAdmin: Version	0.15.0
Tasmota ENERGY ApparentPower	8 VA
Tasmota ENERGY Current	0.033 A
Tasmota ENERGY Factor	0.15 Cos φ
Tasmota ENERGY Power	1 W
Tasmota ENERGY ReactivePower	8 VAr
Tasmota ENERGY Today	0.024 kWh
Tasmota ENERGY Total	2.69 kWh
Tasmota ENERGY TotalSt...	2021-09-23T16:33:29
Tasmota ENERGY Voltage	234 V
Tasmota ENERGY Yesterday	0.047 kWh
Tasmota status	100 %

Figure 5.30: Mobile Home Assistant Tasmota Interface


4:00 83%

HOME DAILY ENERGY

Sun Above horizon

Switch

Sonoff Pow R2 Setup A	
Tasmota	

Sunny 28.1 °C
Home 1.9 mm

Sonoff Pow R2 Setup A Power	0 W
Sonoff Pow R2 Setup A Current	0 A
Sonoff Pow R2 Setup A Voltage	231.92 V
daily_energy	peak
daily_energy offpeak	0.007 kWh
daily_energy peak	0.017 kWh
Power - Today's Cost	0.14861 Php

Figure 5.31: Mobile Home Assistant Consumption Prediction

Chapter 6

Conclusion and Future Works

6.1 Conclusion

The results of the study show that the created models with an accuracy of 0.64 for the UK-DALE dataset, 0.97 for the first model that used the power readings in the created dataset and 0.77 for the second model which utilized an implementation of PCA in the created dataset, are capable of identifying appliances connected to the smart meter and giving a consumption prediction. The said consumption prediction can be viewed in the Home Assistant dashboard and by subscribing to the Telegram bot created.

Given the constraints, such as time, a few things can be done to further improve the project made. The researchers recommend deploying the model in a simple web application such that it can be easily accessible even to those who do not have a background in machine learning. The simple web application should be capable of training the model to a new appliance, load identification of an input of aggregated data, and cost prediction of the current consumption. And to increase the performance and accuracy of the model, more data need to be collected. The collected data should span for at least a year and it should contain varying types of appliances. Lastly, to be able to identify more appliances in the future, the system should be trained with more distinct types of appliances.

6.2 Future Works

To make load disaggregation and cost prediction more accessible to many, a web application is recommended for future work. Deploy the models to an API and have the web application communicate with models through the API. Possible use cases are training the model to a new appliance, load identification of an input of aggregated data, and cost prediction of the current consumption.

In order to be able to cater to one of the features mentioned above, the list of appliances in the model should be able to automatically update once a new appliance is to be trained.

Also, increase the number of appliances in the dataset. This is to cater to a variety of appliances that can be identified during the disaggregation stage.

Another improvement to the existing model would be to increase the sensitivity and specificity of detection. The model was limited to different types of appliances. This was caused by the power values of similar devices being very close to each other. By being able to detect appliances of the same type such as phone chargers with different brands, the model will be more usable.

Lastly, detecting the specific time when certain appliances were connected would be beneficial to the model. It gives more important information to the user which can be translated into more options.

Bibliography

[1] Doe sec. cusi calls for 'Conscious Consumption' of energy as ph plunges into recession. *Department of Energy*, Aug 2020.

[2] Melissa Luz Lopez. Blackouts hit parts of luzon amid thin power supply. *CNN Philippines*, May 2021.

[3] Meralco to install 3.3 m smart meters to prevent 'billing errors'. *Manila Bulletin*, May 2020.

[4] Nipun Batra, Jack Kelly, Oliver Parson, Haimonti Dutta, William Knottenbelt, Alex Rogers, Amarjeet Singh, and Mani Srivastava. An open source toolkit for non-intrusive load monitoring. In *5th International Conference on Future Energy Systems (ACM e-Energy), Cambridge, UK*, 2014.

[5] github - nilmtk/nilmtk: non-intrusive load monitoring toolkit (nilmtk), 2014.

[6] Kitisak Osathanunkul and Khukrit Osathanunkul. Simple load disaggregation library based on nilmtk. In *2020 Joint International Conference on Digital Arts, Media and Technology with ECTI Northern Section Conference on Electrical, Electronics, Computer and Telecommunications Engineering (ECTI DAMT NCON)*, pages 141–144, 2020.

[7] github - amzkit/load-disaggregation: load disaggregation python library.

[8] Jack Kelly and William Knottenbelt. The UK-DALE dataset, domestic appliance-level electricity demand and whole-house demand from five UK homes. 2(150007), 2015.

[9] Uk domestic appliance-level electricity (uk-dale) dataset.

[10] Charnon Chupong and Boonyang Plangklang. Electricity bill forecasting application by home energy monitoring system. In *2017 International Electrical Engineering Congress (iEECON)*, pages 1–4, 2017.

[11] Home assistant integrations, 2021.

[12] Telegram bot, 2021.

[13] Tasmota, 2021.

[14] Joos Korstanje. The f1 score, 2021.

[15] Smart home automation products, Apr 2021.

[16] S.S. Kuruppu and N. Athula. Kulatunga. Smart meter based non-intrusive appliance detection algorithm for local real time feedback purposes. In *IEEE PES Innovative Smart Grid Technologies*, pages 1–5, 2012.

[17] Marius Marcu and Cosmin Cernazanu. Applications of smart metering and home appliances' power signatures. In *2014 IEEE International Instrumentation and Measurement Technology Conference (I2MTC) Proceedings*, pages 331–335, 2014.

[18] Artur F. da S. Veloso, Regenildo G. de Oliveira, Antonio A. Rodrigues, Ricardo A. L. Rabelo, and Joel J. P. C. Rodrigues. Cognitive smart plugs for signature identification of residential home appliance load using machine learning: From theory to practice. In *2019 IEEE International Conference on Communications Workshops (ICC Workshops)*, pages 1–6, 2019.

[19] Zehua Lan, Bo Yin, Tao Wang, and Gengren Zuo. A non-intrusive load identification method based on convolution neural network. In *2017 IEEE Conference on Energy Internet and Energy System Integration (EI2)*, pages 1–5, 2017.

[20] Andreas Reinhardt, Paul Baumann, Daniel Burgstahler, Matthias Hollick, Hristo Chonov, Marc Werner, and Ralf Steinmetz. On the accuracy of appliance identification based on distributed load metering data, 2012.

[21] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.