User behaviour analysis and prediction based on device logs

Zhang, H.

Award date: 2016

Disclaimer
This document contains a student thesis (bachelor's or master's), as authored by a student at Eindhoven University of Technology. Student theses are made available in the TU/e repository upon obtaining the required degree. The grade received is not published on the document as presented in the repository. The required complexity or quality of research of student theses may vary by program, and the required minimum study period may vary in duration.

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

• Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
• You may not further distribute the material or use it for any profit-making activity or commercial gain
User behaviour analysis and prediction based on device logs

Master Thesis

Hao Zhang

Supervisors:
TU/e, W&I: prof. dr. Mykola Pechenizkiy
Philips: dr.Qi Gao
TU/e, IE&IS: M.Sc. Mark Graus

Committee Members:
TU/e, W&I: prof. dr. Mykola Pechenizkiy
Philips: dr.Qi Gao
TU/e, EE: dr. Bart Mesman

Public version

Eindhoven, 28 November 2016
Abstract

Nowadays, with the increasing usage of IoT (Internet of Things) devices in daily lives, the data collected from those connected devices can help understanding user behaviours and delivering better services to individual users. In this thesis, we explored the device log data collected from connected air purifier devices to analyze usage behavior patterns. Further, we integrated the device log data with external data set to invesitgate the impact of external factors on the user behavior. We studied different user behaviour patterns from two aspects. The first one only focuses on the ON/OFF state of the device and four daily usage patterns based on the power state are discovered. Then those patterns are utilized in the prediction model as extra features to improve the performance of prediction. The second one splits power state ON as two operating modes AUTO and OVERRIDE, and the focus is put on the three operating modes: AUTO/OVERRIDE/OFF. Then three metrics are proposed to evaluate user behaviours from different aspects: aggregated operating mode ratio, histogram and distribution of operating mode ratio. Then we study potential factors that may have impact on user behaviours. These analyses produce more insights about how people use their devices and know how these factors affect user behaviours.

Then two prediction models predicting power state and operating modes separately are established using those factors in analysis part as features with two purposes. The first one is to validate the observations in analysis part by comparing prediction performance with and without those factors. Results show that improvement of prediction performance caused by those factors are significant. The second one is to facilitate the further goals like operating mode automatization by evaluating the prediction model. Results show that the final two predictors taking analysed factors as features perform pretty well with average weighted F1 score 0.990 and 0.977 separately.

Finally main findings of this thesis are generalized as well as limitations and possible future jobs. This thesis shows the feasibility of using IoT device logs to analyse user behaviours and the possibility to supply better services to users based on the understanding of user behaviours.
Contents

List of Figures vii
List of Tables ix

1 Introduction 1
  1.1 Motivation and goal ........................................ 1
  1.2 Research questions ........................................ 1
    1.2.1 Research question for data exploration .................. 2
    1.2.2 Research question for operating mode automatization .......... 2
  1.3 Overall workflow ........................................ 2
  1.4 Organization of this thesis ................................ 3

2 Literature review 5
  2.1 Log data related research ................................ 5
  2.2 User behaviour pattern related research ..................... 6
  2.3 Time series prediction related research ..................... 6
  2.4 Operating mode automation related research .................. 7

3 Methodology 9
  3.1 Problem formulations ...................................... 9
    3.1.1 Problem formulation for data exploration ................. 9
    3.1.2 Problem formulation for operating mode automatization .... 10
  3.2 Work summary ........................................... 12
    3.2.1 Work summary for data exploration ....................... 12
    3.2.2 Work summary for operating mode automatization .......... 13
  3.3 Project framework ....................................... 13

4 Analysis of user behaviour patterns 17
  4.1 General data description and analysis ....................... 17
    4.1.1 Operating mode description ................................ 17
    4.1.2 Data set from CNEMC ..................................... 18
  4.2 Daily usage pattern mining based on power state ............. 19
    4.2.1 Daily pattern metric definition ......................... 19
    4.2.2 Daily pattern mining method discussion ................... 19
    4.2.3 Daily pattern mining experiments setting ................. 22
    4.2.4 Daily pattern mining result ................................ 23
  4.3 Pattern exploration based on three operating modes .......... 26
    4.3.1 Pattern metric definitions ............................... 27
    4.3.1.1 Metric 1: aggregated ratio of operating modes .......... 27
    4.3.1.2 Metric 2: Histogram of ratio of operating modes ......... 28
    4.3.1.3 Metric 3: Cumulative distribution of ratio of operating modes . 30
## CONTENTS

4.3.2 Pattern influence factors study .............................................. 31

5 Operating mode prediction for automatization ................................. 33
  5.1 Prediction form discussion ..................................................... 33
      5.1.1 Prediction form 1: direct operating mode prediction .................. 33
      5.1.2 Prediction form 2: operating mode switch prediction ................... 34
      5.1.3 Prediction form 3: time to next mode switch .......................... 34
  5.2 Machine learning algorithms discussion .................................... 35
      5.2.1 Logistic regression ....................................................... 35
      5.2.2 Neural network .......................................................... 36
      5.2.3 Random forest .......................................................... 36
      5.2.4 Gradient boosting classifier ............................................ 37
      5.2.5 Other available machine learning algorithms .......................... 37
  5.3 Evaluation metric discussion .................................................. 38
  5.4 Experiment settings ................................................................ 39
      5.4.1 Prediction goal ............................................................ 39
      5.4.2 Model building ............................................................ 39
      5.4.3 Evaluation metric ........................................................ 40
      5.4.4 Feature engineering ....................................................... 40
      5.4.5 Algorithm selection ....................................................... 40
      5.4.6 Parameters tuning ......................................................... 40
  5.5 Experiment results and analysis ............................................... 41
      5.5.1 Baseline predictors ......................................................... 41
      5.5.2 Predictor with different external factor as additional feature ........ 42

6 Conclusions ............................................................................. 45
  6.1 Main findings ........................................................................ 45
  6.2 Limitations ........................................................................... 46
  6.3 Guidelines for future work ..................................................... 46

Bibliography ............................................................................. 47
# List of Figures

1.1 Whole workflow of the project ............................................. 3
3.1 Whole framework of project .................................................. 15

4.1 Number of days in different air quality levels ...................... 18
4.2 Silhouette analysis for K-means experiments ......................... 24
4.3 Visualization for "always off" daily pattern .......................... 25
4.4 Visualization for "always on" daily pattern ............................ 25
4.5 Visualization for "sleep" daily pattern ................................ 26
4.6 Visualization for "daytime" daily pattern .............................. 26
4.7 Example visualization of aggregated ratio of different modes ........ 28
4.8 Example histogram of ratio of mode Off ............................... 29
4.9 Example of cumulative distribution curve regarding mode Off .......... 31

5.1 Diagram of direct operating mode prediction ......................... 34
5.2 Diagram of operating mode switch prediction ......................... 34
5.3 Diagram of operating mode switch prediction ......................... 35
5.4 Architecture of Neural network[28] ...................................... 36
5.5 F1 score distribution of baseline predictor for Goal1 ............... 41
5.6 F1 score distribution of baseline predictor for Goal2 ............... 41
5.7 Top 10 important features of baseline predictor for Goal1 .......... 42
5.8 Top 10 important features of baseline predictor for Goal2 .......... 42
5.9 F1 score distribution of final predictor for Goal .......................... 43
5.10 F1 score distribution of final predictor for Goal2 .................... 43
5.11 Top 10 important features of final predictor Goal1 ................... 43
5.12 Top 10 important features of final predictor Goal2 ................... 43

User behaviour analysis and prediction based on device logs
List of Tables

4.1  Mapping from AQI index to air quality level ........................................ 18
4.2  K-means experiment configurations ..................................................... 22
4.3  DBSCAN experiment configurations ....................................................... 23

5.1  Confusion matrix for binary classification ............................................. 38
5.2  Experiment result for baseline predictor ................................................. 41
5.3  Experiment results with each external factor added as well as the final predictor for On/Off prediction ......................................................... 42
5.4  Experiment results with each external factor added as well as the final predictor for three modes prediction .............................................. 43
Chapter 1

Introduction

In this chapter, an elaborate introduction will be given regarding different aspects about this project. In Section 1.1, firstly a brief background is provided regarding the environment in which this project is set up, then we give the motivation and specific goals that we are going to achieve. In Section 1.2, several research questions regarding the business goals are proposed. Then a whole workflow of this project representing the informed data mining process is explained in Section 1.3. And finally the overall structure of this thesis is given in Section 1.4.

1.1 Motivation and goal

Nowadays, with the increasing usage of IoT (Internet of Things) devices in daily lives, the data collected from those connected devices can help understanding user behaviours and delivering better services to individual users.

The Smart Air Purifier is an Internet connected product. Because of its ability of cleaning and improving indoor air quality, it has become more and more popular in Chinese families.

In order to have a better understanding of the usage behavior of users and improve user experience, anonymized log data generated by air purifiers have been collected in which a number of parameters about the device are included. Besides, another additional data set regarding the external air quality (referred as EAQ later in this report) is also used in order to help us study the user behaviours more broadly and deeply.

So in conclusion, there are mainly two goals in this project:

1. We firstly want to have more insights about the data sets to get better understanding about how people use their devices and what the factors really affecting user behaviours are combining the external air quality data set.

2. Secondly, we try to utilize the data set for the use case of operating mode automatization on the basis of first part, i.e. predicting what mode users will use based on their historical usage behaviours. In this way, manual interaction with devices can be possibly reduced in future, which helps to improve user experiences by setting the correct state automatically.

1.2 Research questions

From Section 1.1, motivation and goals have been elaborated. And the goals of this project can be generalized simply as two parts: data exploration and operating mode automatization. The first goal can be regarded as a basis for second one to some degree while second one can be regarded as
validation and extension of first part. However the specific research questions in achieving these two goals are still not clear for us. So in this section I will present the detailed research questions that we are going to focus on and dedicated to solving in this project. In Section 1.2.1, research questions regarding first business goal will be given. And in Section 1.2.2, research questions regarding the second business goal will be given.

1.2.1 Research question for data exploration

As explained in Section 1.1, the first goal is to get better understanding about how people use their devices and what the factors affecting user behaviours are. In order to achieve this goal, we must firstly know how to measure or evaluate the user behaviours based on device logs. After getting the metric that can be used as the representation of user behaviours, possible factors that may affect user behaviours have to be studied. So secondly we must determine the potential factors. At last, so as to validate the impact of external factors on user behaviours, comparison experiments need to be designed and conducted to see whether these factors actually affect the user behaviours.

So in conclusion the research questions in first part can be generalized as:
1. How can we model the user behaviour/usage pattern based on the device logs?
2. What is the impact of external factors on user behaviours?
3. How can we design the comparison experiments to show the impact of external factors?

1.2.2 Research question for operating mode automatization

As explained in Section 1.1, the second goal of our project is to achieve the operating mode automatization. This part will be conducted on the basis of first part and can be regarded as a extension and validation of first part. Extension means the obtained insights in first part can be used to establish more appropriate predictive model while validation means the results of prediction can be used to see whether the insights we get in the first part are valid or not. The main strategy that we agree to establish the predictive model is using machine learning techniques to achieve this goal. However specific questions still need to be defined. The most important one is to know what the expected implementation type of operating mode automatization is i.e. what the prediction goal is considering the application in reality. After determining the goal, then we need to understand what information is useful for prediction. In this project, we will mainly see whether the factors we study in first part are helpful for prediction and study impact of those contextual information on prediction performance.

So in conclusion, the research questions in this part can be generalized as:
1. What is the expected implementation of achieving operating mode automatization considering the application in reality?
2. Can we use those contextual information mined in first part to build our prediction model?
3. Does those information indeed improve the prediction performance?

1.3 Overall workflow

Now in the above Section 1.2, we clarify the research questions corresponding to the two goals proposed in Section 1.1. These research questions are mainly focused in our project and we are dedicated to solving them in later steps. In this section, the top-level workflow will be illustrated from an overall view, in which we can see the whole process of the project, the general structure as well as the interconnection between different parts. The workflow is a very abstract generalization...
of what we are going to do corresponding to the goals and specific research questions.

The whole structure of the project follows Figure 1.1. It is an illustration of informed data mining process. The two main data sets in the centre are the focus in this project. Domain knowledge and predictive modelling are the two big parts in our project, which just correspond to the research questions in two parts proposed before. Firstly, based on the two data sets we have, we need to obtain the domain knowledge. Specifically, it is understanding user behaviours as well as studying the impact of external factors on user behaviours. Secondly, we utilize the domain knowledge to form our predictive model based on the main data sets, which corresponds to the use case of operating mode automatization. The predictive modelling is a continuously refined process. And it in turn can be utilized to validate the knowledge we have discovered.

Figure 1.1: Whole workflow of the project

1.4 Organization of this thesis

The remainder of the thesis is organized as follows: In Chapter 2, literature reviews are given from four aspects that are highly correlated with this project. In Chapter 3, we formulate the problems according to the research questions proposed in Section 1.2 after referring to the relevant works in Chapter 2. Then simple work summaries based on the formulations are given as well as the positions of the implementations in this report. And finally the whole framework of this project is given. In Chapter 4, analysis on user behaviours is done which includes daily pattern mining and studying possible factors on how they affect the user behaviours. In Chapter 5, how factors affect the prediction results is studied, through which we confirm the possible features and finally two predictors predicting power state and operating mode separately are established and their performances are evaluated. In Chapter 6, main findings of this thesis are given, as well as the limitations and some future works.
Chapter 2

Literature review

In Chapter 1, business goals and corresponding research questions have been proposed. In order to solve these questions, referring to similar researches that have been done by other researchers or engineers makes much sense for us to come up with the methodologies to the problems in this project. Considering the business goals, research questions and characteristics of the data set, we can find that there are three highly-correlated aspects in our project: log data mining, user behaviour pattern and prediction model on time-series data. As a result, we will firstly expound specific techniques adopted popularly in relevant researches regarding the above three aspects and do some analysis of the feasibilities in this project. Literature reviews for these three aspects correspond to Section 2.1, Section 2.2 and Section 2.3. Secondly we will refer to researches that have similar goals as this project especially to those dedicated to achieving operating mode automatization, then present some analysis and generalization of their methods. These form the Section 2.4. Notice that since it is really hard to categorise one paper just into one of the three aspects, some papers introduced in one subsection may be also related to other subsections.

2.1 Log data related research

Since the collected data in this project comes from the device logs and at the same time log data has been one of important sources of information and log monitoring techniques to characterize system and user behaviour have gained significant popularity\cite{49}, it is necessary to do some survey about the log data mining related techniques.

In \cite{49} Chinghway Lim et al. describe a log mining approach to predict anomalies of enterprise telephony systems using a combination of data mining and statistical analysis techniques, specifically speaking, individual message frequencies are used to characterize the system behaviour and domain-specific knowledge is incorporated through user feedback. However, I find the techniques used in this paper seem not so appropriate in my project, because descriptive information about the failures extracted from these logs which is used for failure prediction is not applicable for predicting operating mode. In \cite{65, 64}, Qiang Yang et al. propose a novelty that is using the n-gram based association-rule predictive algorithm to improve the cache performance from web log mining. This is one method that we can consider later in our project for predicting future operating mode from historical device log. Similarly, in \cite{63}, Qiang Yang et al. again propose an association-rule based predictive algorithm to predict next web page visiting. Their method incorporates a probable temporal region in which the next event will happen as well as a confidence estimate on when the next event will occur. This method seems to match the operating mode prediction in this project in which possible operating mode changing can be regarded as one visiting event.

Above papers mainly introduce some techniques for predictions based on web logs, and their
implementations are primarily on the basis of association rules mining, which may be a candidate for our project.

2.2 User behaviour pattern related research

Concretely, what we are going to do is one kind of user behaviour pattern mining, so other researches regarding user behaviour pattern may be helpful for this project. I will simply introduce some relevant papers in following paragraph.

In [47] Jung-Jin Lee studies learning user pattern of using particular resources through time-series action analysis and abstraction and a final probabilistic user model is constructed that could predict the resources that one user will use in the near future. And the key part of the prediction system is based on the HMM. In [44], Mobasher et al. study web usage mining which servers for the web recommendation through clustering technique. The key point is representing every user as a vector of URL references. A similar study on web usage pattern prediction through clustering technique is also done in [41]. This reminds us that we can also utilize the similar idea, for example we can represent every device usage pattern as a vector of time slots and the value i.e. operating mode can be encoded into numerical numbers, which will be explained in Section 4.2. And in further steps, we can do some clustering from the training set and then do classification for every unseen device which can be used for later prediction. In [33], Rakesh Agrawal studies sequential pattern mining in web transactions which is used for further prediction and three algorithms are proposed, tested and compared. Although the idea they proposed works for web transaction, it seems not very appropriate for our project because there are only 6 operating modes and most users in our data set just run the device in 2 modes. The sequence mining that does not take time information into account is not capable enough of distinguishing different patterns from my intuition.

Above papers are mainly regarding the user pattern mining. There are some papers that are valuable to our project in which representation of pattern needs to be constructed firstly and then mining process is finished by using some clustering techniques. Other techniques like sequence mining is referred too, but it seems not so appropriate.

2.3 Time series prediction related research

No doubt, the time stamp along with each record indicates that the problem I am going to solve belongs to the time series prediction category. Also, nowadays a lot of methods have been proposed to solve such kind of problem. However, I find that most researches are intended to solve the continuous time series prediction problem, for example, [40] primarily models the problem as a stochastic process and gives some solutions from pure statistical perspectives; [44] introduces a Dynamic Evolving Neural-Fuzzy Inference System for time-series prediction. [58] proposes a method for time-series prediction based on Neural network. Besides, it studies the time-series more from a signal processing perspective, which is similar to some ideas in [40]. [50, 45] mainly handle time-series problem based on the so-called grey system theory. These kinds of methods are apparently suitable for regression problem rather than the discrete operating mode prediction in this project.

The most relevant method is from [60], it is dedicated to the rare event prediction in a time series event sequence. A generic algorithm for identifying prediction patterns and a greedy algorithm for creating prediction rules are proposed. It fits our project to some degree because we can regard the operating mode changing in the log data as a rare event.

Actually, after some careful considerations, this kind of time-series discrete prediction can be
regarded as a classification problem rather than a true time series prediction problem, which I mean the temporal information can be ignored or extracted as some explicit features, then traditional algorithms can be used to train the classifier. Many papers have been proposed to solve such kind of problems, for example, [55] to [61] are intended to solve a similar problem from Recsys2015 challenge [10] that is predicting user behaviour based on log data with time stamps. The general idea in these papers can be concluded as two steps:

1. Feature engineering. This step serves for feature extraction consisting of temporal features and statistical features.

2. Classifier training. This step is training needed classifier to do prediction.

A lot of algorithms have been tried in these papers, like neural network, logistic regression, random forest, support vector machine, as well as some advanced methods like gradient boosting algorithms, and ensemble methods. Similar structure and method can be found in winner solutions in the KDD CUP 2015 [7] for Coursera course dropouts prediction. A miniature of all good solutions can be found in [32, 13]. The successful predictions from these two competitions lead me to the idea that the operating mode prediction can also be solved in a similar way i.e. feature engineering and classifier training based on traditional machine learning algorithms.

In this section, I generalized some relevant researches already done by others as references. Although some techniques adopted by them like the ones for continuous time series prediction problem seem not so applicable in this project, most of them can still give us some ideas and guidance like the methods discussed in [60] and strategies to solve time series prediction by general-purpose machine learning techniques in those papers dedicated to Recsys2015 challenge and KDD CUP 2015.

2.4 Operating mode automation related research

Above three sections are literature review related to three aspects: log data mining, user behaviour pattern and prediction model on time-series data respectively. In this section I will give some introductions about papers that directly achieve operating mode automatization.

Device operating mode automation has been proposed for many years by researchers, however, most papers are dedicated to solving this problem from the perspective of control theory [38] or through a predefined schedule program [52, 53, 36, 56, 37], which seems to deviate from what we are going to do through a machine learning method.

Moreover, researches on operating mode automation of air purifier is nearly non-existent, which makes it difficult to refer to others’ results. Fortunately, nowadays similar researches on home thermostat automation to increase occupants’ comfort and decrease energy consumption have been conducted by many researchers in academia and even some mature products have been promoted to the market by companies like Nest Labs [5] and Honeywell [4]. They often only require the user to enter a pre-set temperature and the schedule will be learned automatically while maximizing user’s comfort which is usually evaluated by the so-called miss time and minimizing the energy cost [35]. The unit usually tries to learn the optimal schedule over time based on a user’s historical manual adjustment.

However, the principles like architecture and algorithms behind the products are not made public. In [35], Enda Barrett and Stephen Paul Linder propose an architecture which can support occupancy prediction and HVAC control and optimize the heating and cooling of a space automatically without prior knowledge. Specifically, Bayesian Learning approach is utilized to predict the room occupancy and this predicting result is again used by a Reinforcement Learning method (Q-learning) to learn a control policy. In [51], a similar study is done by Jiakang Lu et al.
which is intended to make thermostat smart enough to turn on and/or off automatically to save energy. There are mainly two steps of their implementation, the first one is using the so-called fast reaction algorithm which is supported by a probabilistic model named Hidden Markov Model to predict the state of occupants like active, sleep and away using sensor data. The second step is to decide whether to preheat the home or not combining the historical pattern with on-line sensor data. In [57], James Scott et al. almost do the same thing, i.e. controlling home temperature through preheating based on occupancy prediction. The difference is that they used a KNN-variant method to predict the occupancy of different times in a day and occupancy situation of each day is represented as a vector. In [39], P.J. Boait et al. demonstrate a fully automatic operation of domestic heating in which the occupancy prediction is achieved through electrical load and hot water usage using Bayes theorem.

All above are relevant researches on operating mode automation of home thermostat which seem very similar to what we are going to do for second part in this project except that the devices are different. And we can also find that the operating mode prediction are all based on the occupancy prediction in these papers. However, occupancy information seems not accessible in this project, actually after doing some analysis and comparison we can easily see that the operating mode prediction in our project is more similar to the occupancy prediction in these mentioned papers rather than operating mode prediction because occupancy prediction is more about the essence of time series based prediction and the so-called operating mode prediction is mainly done on the basis of occupancy prediction. Thus, we can use the methods for occupancy prediction proposed in these papers for reference.

In this section, we present the literature reviews taking into the business goals and research questions into account. Related work are mainly exhibited from four aspects that are high-correlated with this project. Simple analysis as well as discussion of feasibility for these techniques are also included. In next section, we will show the methodologies that we think are suitable to work out research questions in our project based on the references in this chapter.
Chapter 3

Methodology

In previous chapter, we present the relevant literature reviews regarding business goals and research questions from four different aspects. Now combining these related works conducted by other researchers and research questions we proposed in Section 1.2, we need to form our own solutions in this project. In Section 3.1, we formulate the problems in a formal way corresponding to the research questions proposed for two parts in this project. In Section 3.2, according to the problem formulations given in Section 3.1 general work summaries are presented. In Section 3.3, a diagram of the whole framework for our project is given, and simple descriptions for each component as well as their relations are explained.

3.1 Problem formulations

We already know the research questions from the business perspective in Section 1.2, in this section we will refine those questions in a formal way at a high level after referring to other researches which have been explained in Chapter 2. These formulations can be regraded as the guidance and blueprint for our later works. Again, we will present the problem formulations for two different parts respectively.

3.1.1 Problem formulation for data exploration

From Section 1.2.1, we explicated the research questions in data exploration part. Firstly we have to know:

How can we model the user behaviour/usage pattern based on the device logs?

It is necessary to discovery some usage patterns in this multivariate, categorical and time-series based data set. The found patterns then can be regarded as representations of how people use their devices. However, user behaviour pattern is a quite broad concept and is highly dependent on the specific question. For example, even within the same research topic like smart phone usage pattern, Joon-Myung Kang et al. study usage pattern as sequence of smart phone states like voice call and data communication[42] while Heejune Ahn et al. consider smart phone usage pattern as distribution of popularity of applications usage[34]. In other fields like web usage pattern, a vector of URL references is used as the representation of the pattern in [44]. So from these stories we can see that usage pattern is highly correlated with specific question and also can be represented as a metric. For clarity, here we define a pattern as one abstract metric:

$$M(v_1, v_2, v_3...v_n)$$  \hspace{1cm} (3.1)

where:

$M$ = the symbol representing the pattern as an abstracted metric. It can be a numerical value, a
vector and so on.

\( v_i \) = one attribute in which we are interested and contributes to the pattern.

\( n \) = the number of the attributes and is \( \geq 1 \).

Now the problem in this part becomes defining the appropriate and concrete \( M \) from our data set according to what we are going to study.

After defining the usage patterns, the next two questions are both regarding studying how those external and contextual information affect user behaviours. Specifically, the two questions are listed here again:

**What is the impact of external factors on user behaviours?**

**How can we design the comparison experiments to show the impact of external factors?**

In terms of studying the factors affecting use behaviours, we can firstly extend the Formula 3.1 as 3.2 by adding the factors as arguments:

\[
M(v_1, v_2, v_3...v_n) = D(f_1, f_2, f_3...f_m)
\]

(3.2)

where:

\( f_i \) = \( i_{th} \) external factor that affects the pattern \( M \).

\( m \) = number of factors and is \( \geq 0 \).

\( D \) = the map from combination of factors to a specific pattern \( M \).

Now the problem is to find those factors that indeed have impact on \( M \). One example is the so-called control variate method. In this way, a factor \( f_i \) can be considered as valid if following function is valid:

\[
CP(D(f_1, f_2, f_3...f_m), D(f_1, f_2, f_3...f_i + \delta f_i...f_m))
\]

(3.3)

where:

\( CP \) = an abstract comparison method. Appropriate comparison method should be defined based on concrete situation.

\( \delta f_i \) = variance on factor \( f_i \)

If above function is valid which means difference is found, then we can say that \( f_i \) is indeed a determinant of pattern \( M \). This is just an explanation of control variate method. Other methods could also be adopted if appropriate.

Now the problem in the data exploration part can be generalized as making some abstract symbols concrete. Specifically, defining appropriate pattern metric \( M \) including determining attribute \( v_i \), determining argument \( f_i \) in which we are interested and adopting appropriate comparison method \( CP \).

### 3.1.2 Problem formulation for operating mode automatization

It makes sense to automate the operating mode of the air purifier without users’ manual adjustment based on the understanding of user behaviour patterns. In Section 2.4, a lot of relevant researches that are dedicated to achieving the operating mode automatization of thermostat are depicted in details. From all these papers we can generalize that all their implementations of operating mode automatization can be abstracted as three steps:

1. Learn the real room occupancy pattern from historical usage data.
2. Preset a desired environment state e.g the room temperature.
3. Achieve the operating mode automatization based on the previous two items.

A similar use case occurring in our mind is that we can imitate their ideas to achieve our own operating mode automatization of air purifiers. A possible solution would be:

1. Learn the real room occupancy pattern from device logs of the owner.
2. A desired indoor environment is set by the owner beforehand.
3. Achieve the operating mode automatization based on the occupancy pattern and objective environment.

However, after studying the current data set we have, occupancy information is not accessible as well as users’ desired environment state which could be indoor air quality index (referred as IAQ later in this report), which makes it really difficult to utilize the same idea of those papers to achieve such kind of operating mode automatization. But luckily, we can achieve a simplified counterpart, that is learning underlying rules regarding user behaviour on historical data just like the occupancy learning in these papers. This actually has been explained in Section 2.4. This idea is based on the assumption that is logged usage pattern always represents users’ preferences, in other words, we do not distinguish good ways or bad ways of how to use the devices because we lack such kind of information. Then the problem becomes building a model that could output something which could guide the device to switch operating mode automatically, which can be represented as:

\[ y = P([f_1, f_2, f_3, ..., f_n]) \] (3.4)

where:
- \( y \) = output of the prediction model.
- \( f_i \) = \( i \)th element of the feature vector and represents one feature variable.
- \( P \) = a map from a feature vector to an desired output.

Then considering the research question for second part in Section 1.2.2:

**What is the correct form of achieving operating mode automatization considering the application in reality?**

It becomes concretizing the output \( y \), in Formula 3.4, for example, it could be ON/OFF state of the device. Then let us consider the second research question that is:

**Can we use those contextual information to build our prediction model?**

This question now can be achieved by converting those external contextual information to features in Formula 3.4, then we need to evaluate the prediction performance to see whether we can get a satisfied result.

The third research question is:

**Does those information indeed improve the prediction performance?**

In order to study the above question, we define an abstract comparison function \( CP \) which could compare the performance of two predictors. Then so as to decide whether the contextual information improve the prediction performance, we have to check whether the following function is valid:

\[ CP(P([f_{i_1}^b, f_{i_2}^b, f_{i_3}^b, ..., f_{i_m}^b]), P([f_{i_1}^b, f_{i_2}^b, f_{i_3}^b, ..., f_{i_m}^b])) \] (3.5)

where:
- \( CP \) = an abstract comparison function taking into two predictors as arguments.
- \( f_{i_i}^b \) = the \( i \)th baseline feature. Baseline features are those features that are always included in
comparison.

$m$ = the number of baseline features.

$f_t$ = the target feature which we are interested in and it corresponds to the contextual information.

If above function is valid, which means that the performance of predictor with the $f_t$ is better than the predictor without the feature $f_t$, then we can say that the contextual information corresponding to the feature $f_t$ indeed improves the prediction performance. Now the problem mainly becomes defining an appropriate comparison function $CP$.

In this Section 3.1, we formulate the problems in a formal way according to the research questions that are proposed in Section 1.2, and convert those questions to specific tasks based on the formulas raised in this section that is making corresponding abstract symbols concrete. In next two sections, we will discuss specific questions and elaborate the future works based on the formulations and finally a whole framework of this project will be given.

3.2 Work summary

In above section, we have got the formal problem formulations according to the research questions proposed in Section 1.2. However, those problem formulations are at a very high level and can only be regarded as the blueprint for our future works. And no specific tasks have been explained yet. In this section we will give the specific question definitions and summarize the related work that we have done in this project. Again we will organize the content into two parts separately.

3.2.1 Work summary for data exploration

For the data exploration part, in Section 3.1.1, the problem formulations have been explained according to the three research questions, that is:

1. Find an appropriate pattern representation $M$ according to what we are going to study.

2. Determine external factors that we are going to study which correspond to the factors $f_1, f_2, f_3, ..., f_n$ in Formula 3.2.

3. Find an appropriate comparison function method which is used to study the impact of those external information.

In this section, we will define the specific questions according to above listed guidances. Specifically, we mainly did two sub parts in the data exploration part.

1. We study daily usage pattern based on the power state and find four meaningful patterns.

2. We study usage pattern based on the three specific operating modes, and study whether external factors affect the usage pattern.

For the first sub goal, we define one pattern metric in Section 4.2.1 corresponding to the $M$ in Formula 3.2 and we got four patterns through some pattern mining techniques which is discussed in Section 4.2.2. And we did not study the impact of external factors on these patterns. But these patterns are finally utilized by us to build the prediction model.

For the second sub goal, we define three pattern metrics in Section 4.3.1 corresponding to $M$ in Formula 3.2 which describe the usage pattern from three different aspects. Then we study the impact of external factors on defined usage patterns.

Then comparison methods for these three metrics are also discussed in Section 4.3.1 which correspond to the $CP$ in Formula 3.3.
In this section, we describe the works that we are going to do in first data exploration part. And for those concrete implementations based on the problem formulations in Section 3.1.1, we give the section numbers in which they are implemented instead of attaching those details here.

### 3.2.2 Work summary for operating mode automatization

For the operating mode automatization part, the problem formulations have been given in Section 3.1.2 according to research questions. The work mainly consists of:

1. Decide the appropriate output of the prediction model, which corresponds to the $y$ in Formula 3.4.
2. Convert those contextual information into features adopted in Formula 3.4.
3. Find an appropriate comparison method to study whether the contextual information helps for prediction. The comparison method corresponds to the $CP$ in Formula 3.5.

In order to achieve this goal, specifically, the discussion and determination of $y$ is described in Section 5.1 and Section 5.4.1. Generating features from those external information is discussed in Section 5.4.4 and described in Section 5.5. Comparisons are also described and conducted in Section 5.5.

In this Subsection 3.2.2, we describe the works that we are going to do in second part of operating mode automatization. And for those concrete implementations based on the problem formulations in Section 3.1.2, we give the specific positions where these implementations are achieved instead of attaching those details here.

In this Section 3.2, we summarize the works that we are going to do for both two parts according to the problem formulations in Section 3.1 and give references in this report to the detailed implementations. Next, a whole framework of this project will be illustrated.

### 3.3 Project framework

In above Section 3.2, we materialize the problem formulations by giving simple descriptions of the future implementations. We can get a whole framework of this project by putting them together, which is shown in Figure 3.1. From this picture we can see that there are mainly three components in our project: Data collection and preprocessing, data exploration and operating mode automatization. A simplified structure is shown as:

- **Data collection and preprocessing.**
  - Device log data.
  - External air quality data.

- **Data exploration.**
  - Daily usage pattern mining on power state.
  - Analysis of usage patterns based on three operating modes.

- **Operating mode automatization.**
  - Feature engineering.
  - Final predictor for operating mode automatization.
In the data collection and preprocessing part, we collected device log data as well as external air quality data. Then we combine the two data sets together through some preprocessing works for example obtaining the external air quality information for each device.

Then the whole project mainly consists of two parts: data exploration and operating mode automatization, which has been explained in previous sections. In the data exploration part we can see from Figure 3.1 that there are two sub parts. In the first part, we focus on the daily pattern mining based on power state. We define one metric and find four common patterns using clustering techniques. In the second part, we define three pattern metrics and they are used to study the impact of external factors on usage patterns. Those external factors are listed in the figure.

In the operating mode automatization, we also have two blocks. In the feature engineering block, we mainly study whether those external factors and the daily patterns we found in data exploration part could improve prediction performance. And we study two prediction goals, that is, power state and operating modes. After getting the factors that can indeed contribute to the prediction, operating mode automatization is achieved by building two predictors using all features corresponding to two prediction goals respectively.

Notice that the factors that we study in part named ”Analysis of usage pattern based on three operating modes” are same as the contextual information we use in the ”Feature engineering”. So the analysis part is the basis for the feature engineering part and the feature engineering part in turn is also used to validate the analysis.

In this section, we put all different blocks together and form the whole framework, in which the main tasks and correlation between different blocks are illustrated. In next chapters, detailed work in two parts will be presented.
Figure 3.1: Whole framework of project
Chapter 4

Analysis of user behaviour patterns

In the previous chapter, we have made research questions and corresponding problem formulas explicit and clear. Now next step should be obtaining the domain knowledge through data analysis and understanding.

In this chapter, we will present the data exploration part in which we analyse the user behaviour from different aspects and study the factors that may affect this. Specifically, we will go through this process by three steps. In Section 4.1, we will give a general description about the device log data set as well as a simple depiction about EAQ data. In Section 4.2, we are only interested in the power state of the air purifier, that is, ON and OFF from a daily pattern perspective. One metric is proposed to model the daily behaviour and four regular patterns are discovered through clustering. In Section 4.3, our interests in the state of the air purifier are expanded to three modes by splitting the ON mode into two sub modes: AUTO and OVERRIDE. In this way, we refine the user behaviour and take the users’ initiative on operating mode into consideration. Three valid metrics measuring the usage pattern are proposed, based on which potential factors that may affect the pattern are studied.

4.1 General data description and analysis

In this section, I will firstly give the general description about the operation modes of the devices, then some introduction about the EAQ data set is given.

4.1.1 Operating mode description

In this section, we will simply explain the modes of the air purifier. From official description, there are four power states of the device in total: On; Off; Error; Cover_missing. When the power state is On, there are totally six operating modes: Auto; Mode1; Mode2; Mode3; Silent; Turbo. And Auto means the device automatically changes its fan speed within the range from Mode1 to Mode3 according to the indoor air quality index. The fan speed of other five modes is fixed. Mode1, Mode2 and Mode3 are ordinary modes, and Silent mode means the fan speed is slowest and generates lowest noise while Turbo mode is the strongest mode and produces highest noise. So in general, there are 9 different modes: Auto; Mode1; Mode2; Mode3; Silent; Turbo; Off; Error; Cover_missing;
CHAPTER 4. ANALYSIS OF USER BEHAVIOUR PATTERNS

4.1.2 Data set from CNEMC

In this section, we will introduce the EAQ data set. The data set is crawled from the website of CNMEC[1]. It includes in total 286 cities and collection period is different for different cities. One record consists of six fields: Current record number; Date; City; Air quality index(AQI); Air quality levels; Primary pollutant.

What we are mainly focusing on is AQI index which ranges from 0 to 500. According to the AQI, air quality level is decided according to table 4.1.

<table>
<thead>
<tr>
<th>AQI</th>
<th>Air quality levels</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-50</td>
<td>Excellent</td>
</tr>
<tr>
<td>51-100</td>
<td>Good</td>
</tr>
<tr>
<td>101-150</td>
<td>Light pollution</td>
</tr>
<tr>
<td>151-200</td>
<td>Moderately polluted</td>
</tr>
<tr>
<td>201-300</td>
<td>Severe pollution</td>
</tr>
<tr>
<td>301-500</td>
<td>Serious pollution</td>
</tr>
</tbody>
</table>

Table 4.1: Mapping from AQI index to air quality level

After counting number of days in all recorded cities for different air quality levels, we get the following distribution:

![Figure 4.1: Number of days in different air quality levels](image)

From above figure, we can see the distribution between different air quality levels is quite imbalanced and good air quality is most commonly seen.

In this section, we mainly present the statistics of the external air quality data. In conclusion, we find:

1. There are in total 286 cities covered by this data set. Collection period is different for different cites.
2. Different air quality level can be decided according to AQI. And there are six levels suggested by CNEMC[1].
3. There are different number of days in different air quality levels. Good occupies most and the second place is light pollution.
4.2 Daily usage pattern mining based on power state

In this section we primarily focus on the power state to analyse user behaviours, that is except mode Error and Cover_missing we regard any other operation mode as the same mode: On. In this section I will propose a metric corresponding to the Formula 3.1 to look for some common daily usage patterns through clustering techniques. In this way, we want to know whether there exist fixed habits of user behaviours considering the power state. And those patterns can give us more insights about how people use their devices in a day and can also be used by us to characterize user behaviours and improve the performance of prediction model.

4.2.1 Daily pattern metric definition

In order to find common patterns, it is essential and necessary to define a quantitative metric that can represent the daily usage. Also finding an appropriate pattern metric $M$ is guided by the Formula 3.1. Apparently we need two attributes: time slot and power state to represent how the device is used during the entire day. So this reminds us of discretizing the 24 hours by a fixed interval, for example, 10 minutes, then we can use a vector with length 144 to represent the whole day. The $i_{th}$ element $v_i$ represents the time interval from $[10 \times (i - 1)]_{th}$ minute to $[10 \times i]_{th}$ minute and there are in total $24 \times 60 = 1440$ minutes in a day.

$$[v_1, v_2, v_3, v_4, ..., v_i, ..., v_{144}] \quad (4.1)$$

Now we need to come up with a method to let the element of the vector $v_i$ represent the power states. Since there are only 2 modes (On and Off) which we are interested in here, so the state is a binary value. And for the time period in which there is no recorded data, we regard the power state as Off. Now we can simply encode them into 0 and 1 and 0 represents state Off and 1 represents state On. Now the question is how to decide whether the value of element of $v_i$ is 0 or 1. The strategy is we firstly calculate the duration of both modes in that interval indicated by element $v_i$. Here we use $D_i(On)$ and $D_i(Off)$ represent the duration of On and Off respectively in the interval $[10 \times (i - 1)]_{th} - [10 \times i]_{th}$ minute corresponding to $v_i$. Then $v_i$ is:

$$v_i = \begin{cases} 
1, & \text{if } D_i(On) > D_i(Off) \\
0, & \text{if } D_i(On) < D_i(Off) \\
\text{randomly chosen from } 0,1, & \text{if } D_i(On) = D_i(Off) 
\end{cases} \quad (4.2)$$

In such a way now the power state of the device in an entire day can be represented as a vector consisting of 0s and 1s.

4.2.2 Daily pattern mining method discussion

From Section 4.2.1 we know that metric representing a daily usage can be defined as Formula 4.1 whose element value can be calculated based on Formula 4.2. Now we need to find whether there are several common patterns inside these recorded days, which reminds us of the clustering technique. Clustering is a task of grouping a set of objects in a way that the objects in the same group are much more similar to each other than to objects in other groups[16]. Now it has become a main task in exploratory data mining, statistical data analysis and so on[16] and it has also been adopted frequently by other researchers to do pattern mining in other fields.

For example, it has been used by Amro Khasawneh, Sergio A Alvarez et al. to study sleep patterns and group people based on similar sleep behaviour[46]. And in [66], Yanchu Zhang, Guandong Xu et al. use clustering technique to study users’ task-oriented behavior patterns and build user profiles according to users’ web transaction data. In [59], Gang Wang, Xinyi Zhang et al. try to build user behaviour models from click stream data using clustering techniques. Similar research can also be found in [48], that is, a partitional clustering algorithm with interaction-based
CHAPTER 4. ANALYSIS OF USER BEHAVIOUR PATTERNS

features as input vectors is adopted by Luis A. Leiva to find interaction profiles according to how users behave while browsing. All these relevant papers demonstrate that using clustering to find similar patterns according to users' behaviour can be regarded as a valid technique.

Technically speaking, nowadays a lot of clustering models have been proposed such as connectivity models e.g. hierarchical clustering, centroid models e.g. k-means algorithm, density models e.g. DBSCAN and so on[16]. All these models try to group similar items together while separate different items based on a similarity measurement. Below I will give some brief introduction about two potential algorithms as well as several similarity measurements.

**K-means clustering algorithm[25]**

K-means clustering algorithm aims to partition $n$ items into $k$ groups in which each item belongs to the group with the nearest mean[25]. Mathematically, this can be described as given $n$ items $(x_1, x_2, x_3, ... x_n)$ where each item is a $d$-dimensional vector, this algorithm aims to partition the $n$ items into $k (k \leq n)$ cluster set $C = \{C_1, C_2, ..., C_k\}$ so as to minimize the within-cluster sum of distances to the mean point, which can be represented as[25]:

$$\arg\min_C \sum_{i=1}^{k} \sum_{x \in C_i} \|x - \mu_i\|^2 \quad (4.3)$$

where $\mu_i$ is the mean point of cluster $C_i$, that is:

$$\mu_i = \frac{\sum_{x \in C_i} x}{|C_i|} \quad (4.4)$$

Although the problem is computationally difficult, there exists an efficient heuristic algorithm[25] which is often used and can output a local optimum:

1. Firstly initialize $k$ mean points, $\{m^1_1, m^1_2, ..., m^1_k\}$ where the superscript represents the iteration number. Then the algorithm goes by alternating next two steps.

2. For each item $x$, assign it to cluster $C_i$ if it is closest to the mean point $m^t_i$. Then new cluster $C_i$ can be represented as:

$$C^t_i = \{x_p : \|x_p - m^t_i\|^2 \leq \|x_p - m^t_j\|^2 \forall j, 1 \leq j \leq k\}, \quad (4.5)$$

3. Update the new mean point of cluster $C_i$:

$$m^{t+1}_i = \frac{\sum_{x \in C^t_i} x}{|C^t_i|} \quad (4.6)$$

The algorithm terminates when $m^{t+1}_i = m^t_i \forall i, 1 \leq i \leq k$. This algorithm usually converges in a fast speed. And just as Formula 4.3 shows, it is quite often that Euclidean distance shown as Formula 4.2.2 is used to measure the distance between the item and the mean point. And the number of cluster $k$ has to be specified beforehand. This has to be determined carefully because improper cluster number may output a seriously bad result.

**DBSCAN clustering algorithm**

DBSCAN is short for density-based spatial clustering of applications with noise. It is a density-based algorithm which groups together points that are closely packed together while making points that are located in low-density area as outliers[19]. What is different from K-means algorithm is
that DBSCAN does not require a specified number of clusters i.e. the parameter $k$, and this algorithm can automatically decide the number of needed clusters based on the other two given parameters: $Eps$ and $MinPts$. And the so-called density equals number of points within a specifies radius $Eps$. All points can be classified into three categories based on these two parameters[2]:

1. A core point has more than a specified number of points ($MinPts$) within $Eps$.
2. A border point has fewer than $MinPts$ within $Eps$, but is in the neighborhood of a core point.
3. A noise point is any point that is not a core point or a border point.

And the algorithm works as follows[14]:

1. Label all points as core, border and noise according to the above definitions.
2. Remove all noise points.
3. Do clustering on the rest points:
   (a) Put an connection between all core points that are within $Eps$ of each other.
   (b) Make each group of connected core points into a separate group.
   (c) Assign each border point to the cluster of its associated core point.

Not like K-means in which the Euclidean distance (Formula 4.2.2) is often used. In DBSCAN we can adopt other different distance metrics. Below I will give some introduction that can be adopted in our project considering that the daily usage is represented as a vector explained in Section 4.2.1.

**Euclidean distance**[21]

Euclidean distance is quite often used in K-means algorithm and also can be adopted in DBSCAN. Given two vectors: $A = [a_1, a_2, a_3, ..., a_n]$ and $B = [b_1, b_2, b_3, ..., b_n]$, the Euclidean distance is calculated as:

$$d(A, B) = \|A - B\|^2 = \sqrt{(a_1 - b_1)^2 + (a_2 - b_2)^2 + (a_3 - b_3)^2 + \cdots + (a_n - b_n)^2}$$

(4.7)

The calculation shows the distance between two points. The smaller the value is, the more similar the two vectors are to each other.

**Cosine similarity**[17]

Cosine similarity is used to measure similarity of two non-zero vectors by calculating the cosine of the angle between them. Given two vectors: $A = [a_1, a_2, a_3, ..., a_n]$ and $B = [b_1, b_2, b_3, ..., b_n]$, it is calculated as:

$$s(A, B) = \frac{A \cdot B}{\|A\| \|B\|} = \frac{\sum_{i=1}^{n} a_i b_i}{\sqrt{\sum_{i=1}^{n} a_i^2} \sqrt{\sum_{i=1}^{n} b_i^2}}$$

(4.8)

The resulting similarity ranges from $-1$ meaning exactly opposite, to $1$ meaning exactly the same, with $0$ indicating orthogonality[17]. So the larger the value is, the more similar the two vectors are to each other.
Manhattan Distance

Manhattan distance is a measurement in which the metric between two points is the sum of the absolute differences of their coordinates. Given two vectors: \( A = [a_1, a_2, a_3, \ldots, a_n] \) and \( B = [b_1, b_2, b_3, \ldots, b_n] \), it is calculated as:

\[
m(A, B) = \sum_{i=1}^{n} |a_i - b_i|
\] (4.9)

For example, if we set the interval as 10 minutes, then we get a vector with length 144 consisting of 0s and 1s. If the two vectors are totally same, then according to Formula 4.2.2 the distance is 0. If the two vectors are totally different, then we can know that the distance is 144. So we know that the smaller the value is, the more similar two points are to each other.

Above we discuss two potential algorithms and three distance metrics that can be used in our project to mine common daily patterns. Usually, distance metric is combined with the algorithm. And just as mentioned above, K-means often adopts Euclidean distance while different distance metrics can be tried in DBSCAN. Next, I will give some descriptions about the pattern mining experiments through clustering.

4.2.3 Daily pattern mining experiments setting

From above Section 4.2.2, we know that clustering is a valid technique to find common patterns. In this section I will show experiment settings proposed based on above section.

Experiment with K-means clustering

In terms of the algorithm implementation, we directly use the API provided by scikit-learn which is a machine learning toolbox for Python programming language. Since it is not available to change distance function using the supplied API, we just utilize the default Euclidean distance to do experiments. And we already know that the number of cluster \( k \) should be specified beforehand. But because we do not know what the appropriate number is, we tried from 3 to 6. In terms of the daily pattern vector, we know that it is determined by the interval according to the explanations in Section 4.2.1. Since we do not know what the best interval is, we tried three options, that is, 10 minutes, 20 minutes and 30 minutes. So there are in total \( 4 \times 3 = 12 \) experiments and their configurations are listed in Table 4.2:

<table>
<thead>
<tr>
<th>Pattern metric vector interval</th>
<th>Cluster number</th>
<th>Distance function</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 mins</td>
<td>3</td>
<td>Euclidean distance</td>
</tr>
<tr>
<td>10 mins</td>
<td>4</td>
<td>Euclidean distance</td>
</tr>
<tr>
<td>10 mins</td>
<td>5</td>
<td>Euclidean distance</td>
</tr>
<tr>
<td>10 mins</td>
<td>6</td>
<td>Euclidean distance</td>
</tr>
<tr>
<td>20 mins</td>
<td>3</td>
<td>Euclidean distance</td>
</tr>
<tr>
<td>20 mins</td>
<td>4</td>
<td>Euclidean distance</td>
</tr>
<tr>
<td>20 mins</td>
<td>5</td>
<td>Euclidean distance</td>
</tr>
<tr>
<td>20 mins</td>
<td>6</td>
<td>Euclidean distance</td>
</tr>
<tr>
<td>30 mins</td>
<td>3</td>
<td>Euclidean distance</td>
</tr>
<tr>
<td>30 mins</td>
<td>4</td>
<td>Euclidean distance</td>
</tr>
<tr>
<td>30 mins</td>
<td>5</td>
<td>Euclidean distance</td>
</tr>
<tr>
<td>30 mins</td>
<td>6</td>
<td>Euclidean distance</td>
</tr>
</tbody>
</table>

Table 4.2: K-means experiment configurations
For experiments, we filter out those days that have more than 10% miss data and the algorithm is applied on remaining valid days.

**Experiment with DBSCAN clustering**

Just as the K-means algorithm, the API in scikit-learn[11] for DBSCAN is used. And three distance calculation methods are tried. We know that for DBSCAN we need to specify two parameters $Eps$ and $MinPts$ rather than the number of cluster $k$. Since DBSCAN is really slow and consumes huge memory, we do experiments by down sampling the data in half from days we used in experiments of K-means. The algorithm is applied on these valid days. For daily pattern vector, we fix the interval as 10 minutes such that the vector has 144 elements. In terms of the parameter choosing, although there exist some automatic methods like the differential evolution method in [43], it is too complicated to use it in this project. So for each distance metric we tried several configurations. We tried values $\{2000, 4000, 6000, 8000, 10000, 12000\}$ for $MinPts$ according to the total number of days.

Because all elements are binary, so the Euclidean distance between two points ranges from 0 to 12. We tried the values $\{4.0, 5.0, 6.0, 7.0, 8.0, 9.0\}$ for $Eps$. With regard to the Cosine similarity, the value ranges from 0 to 1. Since the larger number indicates smaller distance, then for intuitive understanding the Formula 4.2.2 is modified as:

$$s(A, B) = 1 - \frac{A \cdot B}{\|A\| \|B\|} = \frac{\sum_{i=1}^{n} a_i b_i}{\sqrt{\sum_{i=1}^{n} a_i^2} \sqrt{\sum_{i=1}^{n} b_i^2}}$$  \hspace{1cm} (4.10)

In this way, the values still ranges from 0 to 1 but 0 means closest while 1 means farthest. Thus, we tried values $\{0.3, 0.5, 0.7\}$ for $Eps$. In terms of the Manhattan distance, it ranges from 0 to 144, so we tried values for $\{30, 60, 90\}$ for $Eps$. Now we have in total 72 experiment configurations.

<table>
<thead>
<tr>
<th>$(Eps, MinPts)$</th>
<th>Distance function</th>
</tr>
</thead>
<tbody>
<tr>
<td>${4.0, 5.0, 6.0, 7.0, 8.0, 9.0} \times (2000, 4000, 6000, 8000, 10000, 12000)$</td>
<td>Euclidean distance</td>
</tr>
<tr>
<td>${0.3, 0.5, 0.7} \times (2000, 4000, 6000, 8000, 10000, 12000)$</td>
<td>Cosine similarity</td>
</tr>
<tr>
<td>${30, 60, 90} \times (2000, 4000, 6000, 8000, 10000, 12000)$</td>
<td>Manhattan distance</td>
</tr>
</tbody>
</table>

Table 4.3: DBSCAN experiment configurations

**4.2.4 Daily pattern mining result**

According to settings listed in Table 4.2 and 4.3, we tried all these experiments. In order to find an optimal result, we need to compare the clustering result. And we use Silhouette analysis to evaluate the clustering result as well as visualizing all the clustering result. And finally we find that the clustering result of K-means algorithm outperforms DBSCAN, so here we only present the results of K-means.

**Silhouette analysis**

Silhouette analysis is often used to evaluate the clustering result and helps us to choose parameters because it can study the separation distance between the resulting clusters[15]. It is based on the so-called mean intra-cluster distance($a$) and the mean nearest-cluster distance($b$). Specifically, $a$ means the mean distance between a sample and all other points within the same cluster and $b$...
CHAPTER 4. ANALYSIS OF USER BEHAVIOUR PATTERNS

is the mean distance between a sample and the nearest cluster that the sample is not a part of[6]. Then the Silhouette score for a sample is calculated as[6]:

$$Sc = \frac{b - a}{max(a, b)}$$  \hspace{1cm} (4.11)

It has a range from -1 to 1. Score near +1 indicates the sample is far away from neighbouring clusters, a value near 0 means the sample is very close to the boundary between two neighbouring clusters and a negative score might indicate the sample is assigned to the wrong cluster.

The API supplied by scikit-learn[11] asks for a sample size, then the mean of Silhouette score of all sampled points will be returned as the overall Silhouette score. To eliminate the noise and randomness introduced by the sampling, we do 10 round of calculations, and the mean value of these 10 rounds is regarded as the final Silhouette score. For the K-means experiments, the Silhouette score is shown in Figure 4.2.

![Silhouette score Comparison](image)

Figure 4.2: Silhouette analysis for K-means experiments

From above Figure 4.2, we can see for all interval settings we always get the highest score when the cluster number is set as 4. And the highest score is achieved when setting vector interval as 20 minutes and cluster number as 4. Then we fetch 250 days from each cluster and visualize them to manually label each cluster. And finally we have four different daily patterns regarding on the power states. They are listed from Figure 4.3 to Figure 4.6. The X-axis represents individual days and Y-axis represents the 24 hours in a day from bottom to top. The different colors represent different operating modes while blank represents missing data. Since we only care about power state, blue means \textit{Off} and all other colors representing different operating modes can be regarded as \textit{On}.

**Always off**
CHAPTER 4. ANALYSIS OF USER BEHAVIOUR PATTERNS

Figure 4.3: Visualization for “always off” daily pattern

From above figure, we can find the device is nearly turned off in a whole day.

**Always on**

Figure 4.4: Visualization for “always on” daily pattern

From above figure, we can see the device is nearly turned on in a whole day.

**Sleep**
CHAPTER 4. ANALYSIS OF USER BEHAVIOUR PATTERNS

Figure 4.5: Visualization for "sleep" daily pattern

From above figure, we can see the device is mainly turned on from about 22:00 to about 8:30 next morning. That is the time in which people often sleep.

Daytime

Figure 4.6: Visualization for "daytime" daily pattern

From above figure, we can see that the device is turned on from about 13:00 to about 23:00.

In this section named as daily usage pattern mining, we mainly focus on the daily pattern of the device based on power state. We firstly give the definition of daily pattern metric, and then discuss the way that is clustering to mine daily patterns, and finally we get four meaningful daily patterns through experiments. The visualizations of the clustering results are finally given, which show the result is acceptable at least from manual observations.

4.3 Pattern exploration based on three operating modes

From the data set, we found that air purifiers are more likely to be turned off or set as the Auto mode. In terms of other manual operating modes, they are used in a very small proportion. And based on these observations, in Section 4.2, we study the user behaviours with regard to the power
CHAPTER 4. ANALYSIS OF USER BEHAVIOUR PATTERNS

state of air purifiers at the level of days.

However, it is apparent not enough only to know how people turn on or off the device, and knowing further about how these operating modes are utilized is also interesting, especially for the second part i.e. operating mode automatization. So in this section, we will put more focus on the specific operating modes when the device is on. And we know that there are in total 6 operating modes, that is, Auto, Mode1, Mode2, Mode3, Silent and Turbo, and except Auto, other manual operating modes are seldom used. So for convenience, here we combine other 5 modes as a special mode called Override, which means it is set by the user himself/herself rather than automatically decided by the device itself. When the device is running on this mode, at least it shows that users are not quite satisfied with the Auto mode, that is why the user overrides the operating mode.

In this section, we will mainly study the factors that can affect users’ patterns considering the mode Off, Auto and Override. Specifically we will firstly give some metric definitions that can describe users’ patterns from different aspects, and then finally several experiments are done in which the proposed metrics are utilized to study the factors that could affect users’ behaviour patterns.

4.3.1 Pattern metric definitions

It is important to have some metrics to evaluate users’ patterns, which is the ground for the later processing. The first idea to represent usage pattern when taking Auto and Override into consideration is extending the Formula 4.1 in Section 4.2.1. That is we can use the one-hot encoding method[29] to let each element of the vector be a three-element vector again instead of one numerical value. For example we can let \([1, 0, 0]\) represent Off, \([0, 1, 0]\) represent Auto and \([0, 0, 1]\) represent Override and then the vector representing one day could be a vector with length \(144 \times 3 = 432\). Then we can again utilize the clustering method to get some daily patterns, and finally we can utilize those daily patterns to analyze different factors. However, after some experiment we find that it is really hard to get meaningful and concise results, that is, it is hard to group those days into just several common labels. So in this section, the same method is given up, and we just try to utilize several simple methods to try to present the user behaviour pattern from different aspects at a higher level, which corresponds to the \(M\) in Formula 3.2.

4.3.1.1 Metric 1: aggregated ratio of operating modes

To see how people within a certain group use the devices, this is the most direct and handy method to calculate the ratio of different operating modes. Assume a group consists of a set of \(n\) people that is \(G = \{p_1, p_2, p_3, \ldots, p_n\}\). For the person indexed with \(i\), its device logs consist of a period of \(m\) days and \(m\) is a function of \(i\) and can be presented as \(func(i)\). Then the days of person \(i\) can be represented as: \(D(p_i) = \{d_{i,1}, d_{i,2}, d_{i,3}, \ldots, d_{i,func(i)}\}\). Taking a day indicated by person \(i\) and index \(j\), and one of the three modes Off, Auto and Override as arguments, we define a function \(T\) that outputs the absolute time of the mode in that day. That is, for example \(T(d_{i,j}, \text{Off})\) means the time in which the \(i_{th}\) person puts the device in\(\text{Off}\) mode on his/her \(j_{th}\) day. Now for mode \(m\), its aggregated ratio is calculated as:

\[
AgRatio(m) = \frac{\sum_{i=1}^{n} \sum_{j=1}^{\text{func}(i)} T(d_{i,j}, m)}{\sum_{m \in \{\text{Off}, \text{Auto}, \text{Override}\}} \sum_{i=1}^{n} \sum_{j=1}^{\text{func}(i)} T(d_{i,j}, m)}
\]

(4.12)

After getting the ratio of these operating modes, we can draw them as a stacked bar graph. An visualization example of aggregated ratio for all devices in some city is shown in Figure 4.7
CHAPTER 4. ANALYSIS OF USER BEHAVIOUR PATTERNS

Figure 4.7: Example visualization of aggregated ratio of different modes

The Y-axis in the stacked ratio ranging from 0.0 to 1.0. Different colours represents different
operating modes. The bottom blue is the Off mode, the middle green is Auto mode and the top
red represents the Override mode. When comparing this between different groups, we can draw
stacked graph of all groups in the same figure to do manual observation.

This metric represents the user behaviour at a very high level and from an overall view. This
combines all users together and ignore the difference and variety between individual persons. For
convenience, this metric is referred as $M_1$ later on.

4.3.1.2 Metric 2: Histogram of ratio of operating modes

Just as mentioned in last section, metric $M_1$ ignores the variety of different users. To compensate
for this, we can use the histogram of ratio of operating modes. A histogram is a representation
of the distribution of numerical data[23]. Just as the assumption and rules given in Section
4.3.1.1, the histogram is obtained in following ways. For each person $p_i$, we firstly calculate the
average ratio over days of all operating modes. Thus, the ratio of operating mode $m$ belonging to
\{Off, Auto, Override\} for person $i$ is calculated as:

$$\text{Ratio}(m, i) = \frac{1}{\text{func}(i)} \times \sum_{j=1}^{\text{func}(i)} \frac{T(d_j, m)}{\sum_{m \in \{\text{Off, Auto, Override}\}} T(d_j, m)}$$  \hspace{1cm} (4.13)

And for $n$ persons and one given specific operating mode $m$, we can calculate the ratio and get a
vector:

$$[\text{Ratio}(m, 1), \text{Ratio}(m, 2), \text{Ratio}(m, 3), ..., \text{Ratio}(m, n)]$$

Then we need to divide the entire range of values into a series of intervals, and then count
how many values fall into each interval[23]. So histogram is mainly featured by two vectors.
The first vector indicates the boundaries of the intervals, that is, $[b_1, b_2, b_3, ..., b_n]$. The second
vector indicates the number(or ratio) of values located in the corresponding intervals, that is, \([c_1, c_2, \ldots, c_{n-1}]\). Element \(c_i\) means the number(the ratio) of values in the range \([b_i, b_{i+1})\). A typical visualization is drawing bar graphs for each count value(or ratio) whose axis includes the value intervals. An example of the histogram of ratio of mode \(Off\) for users in some city is shown in Figure 4.8.

The X-axis means the value range that is from 0 to 1 here because it is the ratio of operating mode. The Y-axis indicates the proportion of persons. From each bar, we can know the ratio of persons whose usage ratio of mode \(m\) is in that certain interval.

From Figure 4.8, we can see how people use the device in \(Off\) mode is quite variable because for each interval from small to large, there are always a certain number of persons. Also we can estimate the probability of a user using one specific mode in a certain ratio of time. For example, according to the first column, we can estimate that the probability of a user in this city using the \(Off\) mode in less than 5% time is about 14.5%.

In terms of comparing two different histograms, we need to firstly fix the boundary vector and then there are several ways to calculate the difference between two different histograms. Suppose that the count(or the ratio) vectors for \(n\) bins are separately: \(A = [Ra_1, Ra_2, Ra_3, \ldots, Ra_n]\) and \(B = [Rb_1, Rb_2, Rb_3, \ldots, Rb_n]\), then there are four calculation methods according to the histogram comparison documentation in OpenCV[3], which is an open source library for computer vision[30]. However, some original distance calculations(Correlation and Intersection) do not satisfy the intuition, that is 0 distance should intuitively means totally same, and the larger the distance is, the more different two cities are. So some modifications are done, then we get following four distance calculations.

**Correlation**

\[
d(A, B) = 1 - \frac{\sum_{i=1}^{n}(Ra_i - \bar{A})(Rb_i - \bar{B})}{\sqrt{\sum_{i=1}^{n}(Ra_i - \bar{A})^2 \sum_{i=1}^{n}(Rb_i - \bar{B})^2}}
\]  

(4.14)

where \(\bar{A} = \frac{\sum_{i=1}^{n}Ra_i}{n}\) and \(\bar{B} = \frac{\sum_{i=1}^{n}Rb_i}{n}\). And this correlation ranges from 0 to 2 where 0 is the perfect match and 2 is the worst.
CHAPTER 4.  ANALYSIS OF USER BEHAVIOUR PATTERNS

Chi-Square

\[ d(A, B) = \sum_{i}^{n} \frac{(R_{a_i} - R_{b_i})^2}{R_{a_i} + 0.00001} \]  

(4.15)

And this score is \( \geq 0 \), where 0 is the perfect match and mismatch is unbounded. The 0.00001 in the denominator is to prevent the divisor from being 0.

Intersection

\[ d(A, B) = 1 - \sum_{i}^{n} \min(R_{a_i}, R_{b_i}) \]  

(4.16)

And this score ranges from 0 to 1, where 0 is the perfect match and 1 is mismatch if the histogram is normalized.

Bhattacharyya distance

\[ d(A, B) = \sqrt{1 - \frac{1}{\sqrt{ABn^2}} \sum_{i}^{n} \sqrt{R_{a_i} \cdot R_{b_i}}} \]  

(4.17)

where \( \bar{A} \) and \( \bar{B} \) are mean value and explained in Correlation calculation. This score ranges from 0 to 1 where 0 is perfect match and 1 is mismatch.

Above are four histogram comparison methods, each method uses a different distance function. And any of them can be used to test the histogram for equality.

Now the metric proposed in this section measures the user behaviours considering the variety of individual users within a certain group. And for convenience, this metric is referred as \( M^2 \) later.

4.3.1.3  Metric 3: Cumulative distribution of ratio of operating modes

In Section 4.3.1.2, we propose using histogram to measure user behaviours within a certain group, which can describe variety of individual users. Also four distance calculation methods are proposed to test the equality of two histogram. We already know from Formula 4.13 that for mode \( m \) and \( i_{th} \) person, \( Ratio(m, i) \) means the average ratio over days of mode \( m \) for this person. In this section, from another perspective, we can regard the ratio of mode \( m \) for \( n \) persons, that is the vector:

\[ [Ratio(m, 1), Ratio(m, 2), Ratio(m, 3), ..., Ratio(m, n)] \]

as data points sampled from an inner undiscovered distribution. And in order to show this distribution, we can get an estimated cumulative distribution\[18\] curve from these points by the following way. For the above vector, we can reorder it in an ascending way and get a new vector:

\[ [AsRatio(m, 1), AsRatio(m, 2), AsRatio(m, 3), ..., AsRatio(m, n)] \]  

(4.18)

such that for each pair index \((i, j)\), where \( i \leq j \), we have \( AsRatio(m, i) \leq AsRatio(m, j) \).

Then corresponding to each index \( j \), we calculate the position ratio of that index, that is \( \frac{j}{n} \). Then we get another vector:

\[ \left[ \frac{1}{n}, \frac{2}{n}, \frac{3}{n}, ..., 1.0 \right] \]  

(4.19)
CHAPTER 4. ANALYSIS OF USER BEHAVIOUR PATTERNS

Then the estimated cumulative distribution curve can be drew by letting the vector 4.19 as X-axis and vector 4.18 as Y-axis. A visualization example for some city regarding mode Off is given in Figure 4.9.

![Cumulative distribution curve](image)

Figure 4.9: Example of cumulative distribution curve regarding mode Off

The X-axis represents ratio of devices and Y-axis represents ratio of mode Off. From Figure 4.9, we can know how ratio of mode Off distributes in this city. And for example, from the point located at (0.2, 0.2), we can estimate that there are less than 20% persons who use the Off mode in less than 20% time.

In order to compare the difference of two distributions, we can use the so-called KS test\[26\], which is a nonparametric test of the equality of continuous, one-dimensional probability distributions that can be used to compare a sample with a reference probability distribution (one-sample KS test), or to compare two samples (two-sample KS test)\[26\]. Here in our project we use the so-called two-sample K-S test because we do not have a reference probability. And the API in scipy library\[12\] is used in our project for simplicity, which returns two outputs: \(\text{ks}\) statistic and \(p\) value. In this project, we set the significance level as 5%, that is, only when the \(p\) is smaller than 5%, we can say that the two distributions under test can be regarded as different.

This metric is just like \(M2\) considering the variety of individual persons, and also supplies a guidance to judge whether two distributions differ to each other or not given a significance level, while we can only get distances from \(M2\) based on which it is hard to do judgement but suitable to see the degree to which they are different.

Till now, we have proposed three pattern metrics that can be used to analyze user behaviours from different aspects as well as the corresponding comparison methods. In next part, we will study what factors affecting users’ behaviours are and how they affect users’ behaviours based on the proposed metrics.

4.3.2 Pattern influence factors study

In Section 4.3.1, we propose three metrics and comparison methods which can represent user behaviours from different aspects and help us to see the difference between them. Based on these, in this section we want to know whether there exist some factors that may have impact on the user behaviour patterns. While studying one factor, the strategy is that we split the data set into several sub groups according to different conditions of that factor, and then the metric and comparison method in Section 4.3.1 are utilized to do experiments whose results are then analyzed.
CHAPTER 4. ANALYSIS OF USER BEHAVIOUR PATTERNS

to form conclusions.

Notice that not all three above metrics need to be used every time because for some factors and some metric, it is not suitable to do comparison experiments between sub groups split by that factor. Also which metric is used is also dependent on what we are interested in. Below, main findings of experiments regarding the external factor EAQ will be given.

Based on EAQ

Certainly, external air quality has a high probability to have impact on users’ behaviours from intuition and it is interesting to investigate this aspect. So in this section, we will study how EAQ affects user behaviour patterns. By employing the metric defined previously \(M_1\) and \(M_3\), we study the impact of EAQ on users’ behaviour. The main finding are summarized as follows.

1. In terms of metric \(M_1\):
   - Users are indeed more likely to turn on the device and manually adjust it when EAQ becomes serious.
   - And this observation happens in all three ways of splitting EAQ. But it is more apparent when the EAQ is split into six or five levels than when the EAQ is split into only two levels.

2. In terms of metric \(M_3\):
   - No significant difference is found between \textit{Good} and \textit{Bad} EAQ for each mode.
   - Still no significant difference is found between days with same-level engagement.
   - Significant difference is found between \textit{Good} and \textit{Bad} EAQ in \textit{CityB}.
   - The same difference is again found between \textit{Good} and \textit{Bad} EAQ in \textit{CityB} for high-engagement days.

From above generalization, we can see that difference is more apparent when using \(M_1\) because we can find it on three modes and also it is apparent in the figure especially when splitting the EAQ into more levels. However, no much difference was found when using \(M_3\). What we can know is that people in \textit{CityB} are more sensitive to the EAQ especially on high-engagement days. But this sensitivity is only embodied in the \textit{Off} mode. The reason we guess resulting in this phenomenon is we split the EAQ only into \textit{Good} and \textit{Bad}, because we can also find that the difference fades in \(M_1\) when only considering the two levels. Anyway, what we have found now tells us that EAQ can be used as an extra feature to build the prediction model and also hints us that we can also try different refinement ways to present the EAQ as an extra feature.

In summary, in this Section 4.3.2, we firstly define three metrics to measure the user behaviour patterns from different aspects. Then we use appropriate metrics to study some potential factors that may affect usage patterns and present the main findings of EAQ.
Chapter 5

Operating mode prediction for automatization

In above chapter, analysis about the data set regarding user behaviour patterns is performed, through which we discover some factors that can affect user behaviours. In this chapter, we focus on the second part of the project, that is operating mode automatization based on what we have concluded till now. The limitation of this project to achieve this goal has been explained in Section 3.1.2, and also a formal problem formulation is proposed that is we need to build a prediction model in a way that could give guidance for device automatic operation. On the one hand, the prediction model can be regarded as a further step based on our previous understanding, on the other hand the prediction model can be used to validate our precious findings especially for those discoveries about factors being determinants of user behaviours.

Regarding the prediction model, in Section 5.1 feasible forms of predictions will be discussed for the purpose of automatization. Then in Section 5.2 available prediction algorithms will be talked over from an technical perspective. Then possible evaluation metrics regarding prediction performance are given in Section 5.3. In Section 5.4, settings about the experiments that are to be done are given. Finally we give analysis and results of the experiments in Section 5.5.

5.1 Prediction form discussion

From Section 3.1.2, we know that it is not available for us to achieve operating mode automatization based on users’ occupancy pattern and desired environment state that are often utilized by other researchers to achieve similar goals. We have to think of a prediction form that can be used to service automatization according to what we already have now. This corresponds to the first research question in Section 1.2.2 and just as explained in Section 3.1.2, we need to determine the output $y$ in Formula 3.4. The automatization is nothing but letting the device choose a proper mode automatically. Thus, we propose three possible ways to achieve this.

5.1.1 Prediction form 1: direct operating mode prediction

The simplest one that occurs to our mind firstly is predicting the operating mode according to conditions measured at a certain time point, which is shown in Figure 5.1. From the historical observation, the model learns what the operating mode should be in a certain environment. In practice, the device can work in following way. It could detect the state including external environment situation, current device state and configurations, for example, every 10 minutes. Then combing these measurements with the time information at that point, the device could make a decision about the proper operating mode. If it is different from the current one, then it switches to the target automatically. Otherwise, it stays on the current operating mode until next decision.
5.1.2 Prediction form 2: operating mode switch prediction

The second form for prediction is in Figure 5.2. In this way, the device tries to predict whether
there is a mode switch in a fixed time window. This usually could be done through two steps:

1. First step is predicting whether there will be a mode switch from now on in a time window,
which is a binary classification problem.

2. Second step is predicting the destination mode if the answer of first step is yes, which is a
multi-class classification problem.

This is also a valid way for practical use. For example the device can detect the necessary in-
formation to make a decision. Then the device could execute the determination when the time
window expires and make another prediction. In this way the time window can be adjusted to
achieve a better performance.

5.1.3 Prediction form 3: time to next mode switch

The third form for prediction is in Figure 5.3. In this way, the device tries to predict the time to
the next mode switch. This usually could be done through two steps:

1. First step is predicting time to the next switch. This is in essence a regression problem. We
can categorized the time into several ranges to make it become a classification problem.

2. The second step is predicting the destination operating mode, which is a multi-class problem.
CHAPTER 5. OPERATING MODE PREDICTION FOR AUTOMATIZATION

This form is also a valid way for practical use. For example, the device can detect necessary information to make a decision. Then when the waiting time expires, the device switch to a new mode and make another prediction.

Above are three forms that we can use to do prediction and each method could help us get closer to the goal of operating mode automatization. However, considering the complexity of implementation and future practical deployment we find that the first form is the proper one. Firstly, it is the easiest one in terms of model building because it consists of only one multi-class problem. Secondly, considering quality of our data set I do not think the model could perform well in the latter two methods in which we need to predict next mode switch and time to next switch respectively. Thirdly, the form 1 is also quite easy, efficient and error-proof in practical application. Because it can adjust operating mode in a fixed frequency and the current prediction is not related with previous one. However in the form 3, the prediction moment is decided by the previous prediction. If the previous one is not correct then the later one may be affected due to chain effect. So in our project, we choose the first form.

5.2 Machine learning algorithms discussion

Nowadays a lot of prediction algorithms have been proposed. And many of them are quite mature nowadays. In this section, I will give some introduction to the commonly used machine learning algorithms. These algorithms can be used in a large number of fields. In principle, they are all methods that try to firstly discover the underlying rules between input and output from observed data called training set, and then try to give an appropriate output for an unseen input. The input usually consists of a set of features which normally need to be tuned by so-called feature engineering. And according to the analysis in Section 2.3, we can find that many general-purpose machine learning algorithms have been adopted by other researchers to solve time series data prediction. Next, I will explain some famous algorithms.

5.2.1 Logistic regression

Logistic regression[8] is one kind of classification method which estimates the probability of a binary response based on one or more predictor (or independent) variables (features)[27]. The probability is the output of a logistic function, which is defined as follows:

\[ g(z) = \frac{1}{1 + e^{-z}} \]  

(5.1)

For a training set \( \{(X(1), y(1)), (X(2), y(2)), \ldots, (X(m), y(m))\} \), assume \( X(i) \) are features with \( N \) dimensions, \( y(i) \) are labels where 0 indicates negative and 1 indicates positive. Then the training
problem is to find parameters $\theta_0, \theta_1, \theta_2, \ldots, \theta_N$ such that the cost function is minimized which is defined as:

$$-\frac{1}{m}\left[\sum_{i=1}^{m}(y'_i \log(h_{\theta}(x'_i)) + (1 - y'_i) \log(1 - h_{\theta}(x'_i)))\right] \quad (5.2)$$

The definition of $h_{\theta}(x')$ is:

$$h_{\theta}(x') = g(\theta_0 + \theta_1 x'_1 + \theta_2 x'_2 + \cdots + \theta_N x'_N) \quad (5.3)$$

When a new sample feature is given, then we calculate $h_{\theta}(x')$, if it is greater than 0.5, then we label it as positive otherwise negative. A more advanced version is so-called regularized logistic regression whose cost function is defined as:

$$-\frac{1}{m}\left[\sum_{i=1}^{m}(y'_i \log(h_{\theta}(x'_i)) + (1 - y'_i) \log(1 - h_{\theta}(x'_i)))\right] + \frac{\gamma}{2m} \sum_{j=1}^{N} \theta_j^2 \quad (5.4)$$

$\gamma$ is the so-called regularized factor, which is mainly used to prevent overfitting. For the multi-class classification, we can use the so-called one-vs-all method, that is training one classifier for each class in such a way that is labelling the current class as positive and all other classes as negative. Then the final prediction can be done by combing all individual classifiers.

### 5.2.2 Neural network

Neural network[28] is one kind of classification which mimics the humans’ central nervous systems by connecting processing elements to form a network. And a typical architecture looks like Figure 5.4. Every unit in the hidden layer is a weighted sum of the input features and the output unit is also another weighted sum of the hidden units. In this way even complex non-linear relations could be expressed.

### 5.2.3 Random forest

Random forest classifier is an ensemble of decision trees[20] which offers various sources of randomness to break correlations among predictor variables and also the trees in the ensemble[62]. Typically, each tree works on a bagged version of the input data and each node split in each tree is determined according to a random subset of features rather than the whole feature set. The expectation is that the variance of the final classifier is lower than individual classifiers while its bias is compensated. And the final output is usually decided by vote of individual classifiers.
5.2.4 Gradient boosting classifier

Gradient Boosting Classifier is another ensemble model which is a weighted combination of weak classifiers (which is typically decision tree as well) and is built successively based on the residual error of preceding weak classifiers\(^\text{[54]}\). Assume that we have input variable \(X = x_1, x_2, \cdots, x_n\) and output variable \(y\). Given training data \(\{x_i, y_i\}_{i=1}^n\), the goal is to find \(F^*\) that minimizes the expectation of the given loss function \(L\):

\[
F^* = \arg\min_F E_{x,y}[L(y, F(x))]
\] (5.5)

Gradient boosting assumes \(F\) in the form of weighted sum of weak learners \(h_i(x)\):

\[
F(x) = \sum_{i=1}^{n} \gamma_i h_i(x) + \text{const}
\] (5.6)

Where \(\gamma_i\) is the weight of \(h_i(x)\). It starts with a model consisting of a constant function \(F_0(x)\) and expands it in forward-stage manner.

\[
F_0(x) = \arg\min_{\gamma} \sum_{i=1}^{n} L(y_i, \gamma)
\] (5.7)

\[
F_m(x) = F_{m-1}(x) + \arg\min_{f} \sum_{i=1}^{n} L(y_i, F_{m-1}(x_i) + f(x_i))
\] (5.8)

Then the standard steepest gradient descent method is applied to solve the minimization problem.

General-purpose algorithms may rely primarily on feature selections and parameters tuning. Although these algorithms are general, but they are often difficult to interpret, that is, their internal structure might not have any correspondence to the real-world process that is generating the training data\(^\text{[47]}\).

5.2.5 Other available machine learning algorithms

Besides the above mentioned commonly used algorithms, there are other algorithms as well according to the analysis in Chapter 2. As explained in Section 2.2, HMM (Hidden Markov Model) is adopted by Jung-Jin to predict the resources that are to be used based on time series data in \([47]\). Since the record in our data is time stamped, it can be regarded as time-series data. This gives us hints to use HMM to establish a predictor on our dataset. Hidden Markov Model is one kind of Markov process based on the Bayesian theory. It is represented as a tuple of initial state probabilities, transition probabilities and emission probabilities. Learning problem in HMM is to find, given an output sequence or a set of such sequences, the best set of state transition and emission probabilities. The task is usually to derive the maximum likelihood estimate of the parameters of the HMM given the set of output sequences\(^\text{[24]}\). And Viterbi algorithm\(^\text{[24]}\) is often used in prediction using HMM. Also, some other strategies based on statistical methods like the association-rules in \([65, 64, 63]\) can be tried, which has been explained in Section 2.1 in the literature review part.

Although our data can be regarded as time sequences, HMM or association-rule only considers the time information and it is hard to utilize other vital elements like EAQ, season and IAQ. So adopting HMM or association-rule is not a good choice. In terms of the four algorithms we discussed above, all of them have simple API to use from scikit-learn\(^\text{[11]}\) and considering the speed and performance according to our experience, finally random forest is adopted.
5.3 Evaluation metric discussion

It is important to evaluate how the prediction model performs on an unseen data set i.e. its generalization ability. Nowadays many metrics have been proposed to help us choose a better model. Take a binary classification problem as an example, the performance of the model is mainly evaluated by the counts of test instances that are correctly and incorrectly predicted by the model. And a very effective visualization of the performance is putting those counts in a specific table layout called confusion matrix. This is illustrated in Table 5.1.

<table>
<thead>
<tr>
<th>Predicted condition</th>
<th>True condition</th>
<th>False condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>positive</td>
<td>True positive</td>
<td>False negative</td>
</tr>
<tr>
<td>negative</td>
<td>False positive</td>
<td>True negative</td>
</tr>
</tbody>
</table>

Table 5.1: Confusion matrix for binary classification

In the table, \( tp \) means the counts of positive instance that are correctly predicted, \( fp \) means the counts of negative instance that are incorrectly predicted, \( fn \) means the counts of positive instance that are incorrectly predicted and \( tn \) means the counts of negative instance that are correctly predicted. Based upon above four counts, some metrics can be introduced.

**Accuracy**

Accuracy means the ratio of correctly predicted instances over all test instances. This is can be defined as:

\[
\text{accuracy} = \frac{tp + tn}{tp + fp + fn + tn}
\]

(5.9)

Normally, accuracy truly shows how well the performance is. However, sometimes it may let us get a biased model if the instances are imbalanced. For example, there are 90 positive and 10 negative instances, then a predictor always outputing positive could get a 90% accuracy, which is not a bad result.

**Precision**

In an classification problem, precision for label \( i \) means the fraction of instances we correctly predict as \( i \) out of all instances that are predicted as \( i \) by the model. In our confusion matrix, precision for positive is defined as:

\[
\text{positivePrecision} = \frac{tp}{tp + fp}
\]

(5.10)

And precision for negative is defined as:

\[
\text{negativePrecision} = \frac{tn}{tn + fn}
\]

(5.11)

**Recall**

In an classification problem, recall for label \( i \) means the fraction of instances we correctly predict as \( i \) out of all instances that are actually with label \( i \). The recall for positive is defined as:

\[
\text{positiveRecall} = \frac{tp}{tp + fn}
\]

(5.12)

The recall for negative is defined as:

\[
\text{negativeRecall} = \frac{tn}{fp + tn}
\]

(5.13)
F1 score
In practice, balancing precision and recall is very necessary, based on which the F1 score is proposed. It is defined as:

$$F1 = \frac{2 \times precision \times recall}{precision + recall}$$

(5.14)

F1 score reaches best at 1 and worst and 0. In a multi-class classification problem, quite often the weighted F1 score is used. It is the average of all labels’ F1 score, weighted by support (the number of true instances for each label).

Above we give some formal definitions of all kinds of metrics for evaluating the predictor performance. In our project, we use the weighted F1 score as our final standard.

5.4 Experiment settings

In above several sections, we present the general discussion about different aspects of prediction model, which can help us decompose the related tasks regarding operating mode prediction into several sub problems like device selection, model building and so on. In this section I will give details about those specific problems, which are concluded as experiment settings here.

5.4.1 Prediction goal

In Section 5.1, we discuss three possible prediction forms that can to some degree achieve the goal of automatization, and some reasons are given to the selection of form 1 that is predicting operating mode at a certain time point.

Just as we have analysed in Chapter 4.1, power state is firstly focused on followed by three operating modes. With regard to the specific prediction goal, we also practice in two aspects. The first one is only considering the power state and try to predict whether the device should be turned on or not. The second one is considering three modes: Off, Auto and Override by combing other manual modes together.

Although both two methods are not real operating mode automatization because specific modes still to be decided, this could lay a ground for later implementation to see whether it is possible to achieve the real automatization.

5.4.2 Model building

Considering the current situation, we can have two options to build the prediction model.

Build one model for all devices

The first option is building one model for all devices based on the whole data set. In this way, we put more focus on the general pattern mining and try to find the rules that are suitable for all users. This way is very easy in practical application. We only need to deploy the model on all devices after we get it.

Build one model for each device

The second option is building one model for each device. This is also mostly adopted by other researchers and products. In this way, we put more focus on the individuality of users and try to find the rules that are exclusively suitable for one user. This way is a little complicated in practical application. For each air purifier, we have to install one machine learning system on it.
CHAPTER 5. OPERATING MODE PREDICTION FOR AUTOMATIZATION

and let it learn the specific pattern.

Above, we talked about two possible ways to build the model. Considering the cost of implementation, we will employ the first option in our project.

Specifically, for each device we will split the data into two parts: training set and test set. Then we combine all training sets together as a new big one, and the same for test sets. Finally the model is built on the whole training set and evaluation is done on the whole test set.

5.4.3 Evaluation metric

Just as explained in Section 5.3 we will use the weighted F1 score as the evaluation metric. According to above section, now the test set consists of all individual devices. Instead of obtaining F1 score on the whole test set, we will calculating F1 score for each device. Then through distribution of F1 scores, we can see how this model behaves on different devices for example how many are well predicted and how many are poorly predicted. However, the question is it seems hard to compare two models. Here we focus on the mean F1 of all devices when doing comparison. And under the help of t-Test using API from Scipy[12], we can get the significance of the difference. Here we regard the average values of two sample sets as different only if the $p$ value is $< 0.05$.

5.4.4 Feature engineering

Feature engineering is quite important and fundamental in machine learning problems. It is the process of using domain knowledge of the data to create features that make machine learning algorithms work better[22]. In Section 4.3.2, the analysis of impact on user behaviours of different factors is the basis for feature selection in this part. We will investigate whether these external factors indeed improve the prediction performance. So we firstly need a baseline predictor that only takes those original information in the records as features. And information that is not useful or directly indicates operating mode is not considered. Notice that IAQ is included as the original feature since it is one field of the record.

Then in order to validate that those external factors actually have impact on user behaviours, we need to add one external factor to the features each time to see whether the performance is indeed improved. If so, we can say that the factor indeed improve the prediction performance.

5.4.5 Algorithm selection

Just as explained in Section 5.2, only Random forest will be tried using the API supplied by scikit-learn[11].

5.4.6 Parameters tuning

However, there are a lot of parameters that we can change using the API. So in order to get a well performed model, the parameters should be carefully selected. However, if we set candidate values for all parameters, then the parameter space will expand to a large extent such that it becomes extremely slow and expensive to find the best combination. So here we mainly focus on three parameters.

- **max_features.** This is used to decide the number of features to consider when looking for the best split[9]. Two potential values for this parameter are: 'sqrt' and 'log2'.
- **min_samples_split.** The minimum number of samples required to split an internal node[9]. Two potential values for this parameter are: 2 and 10.
- **min_samples_leaf.** The minimum number of samples in newly created leaves[9]. Two potential values for this parameter are: 1 and 3.
Now we have 8 possible combinations for parameter tuning. When doing experiments for each feature combination, we need to find a best parameter combination. For each parameter combination we will do 3-fold cross validation. It means three rounds of experiments will be done in each of which 66.7% of the data will be used as training set and remaining part will be used as test set. The folds are made by preserving the percentage of samples for each class. Then according to the average score of the three rounds, the best parameter combination is decided.

In this section, we give descriptions about the experiment settings to show how our experiments will be conducted and evaluated.

5.5 Experiment results and analysis

In this section, I will give experiment results for the baseline predictor and final predictor.

5.5.1 Baseline predictors

Firstly I give the outcome of the baseline predictor, which is shown in Table 5.2.

<table>
<thead>
<tr>
<th>baseline</th>
<th>On/Off prediction</th>
<th>Three modes prediction</th>
</tr>
</thead>
<tbody>
<tr>
<td>count</td>
<td>1011</td>
<td>1011</td>
</tr>
<tr>
<td>mean</td>
<td>0.616</td>
<td>0.509</td>
</tr>
<tr>
<td>std</td>
<td>0.186</td>
<td>0.262</td>
</tr>
<tr>
<td>min</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>25%</td>
<td>0.499</td>
<td>0.295</td>
</tr>
<tr>
<td>50%</td>
<td>0.629</td>
<td>0.516</td>
</tr>
<tr>
<td>75%</td>
<td>0.757</td>
<td>0.735</td>
</tr>
<tr>
<td>max</td>
<td>0.983</td>
<td>0.994</td>
</tr>
</tbody>
</table>

Table 5.2: Experiment result for baseline predictor

In the table, except the number in the row indexed by ‘count’, all other values represent F1 score. For convenience, from now on On/Off prediction will be referred as Goal1 and Three modes prediction will be referred as Goal2. From the above table, we can see that mean value of Goal1 is larger than Goal2 and this shows prediction of power state is better than three modes. For each prediction goal, we can find the result is very good for some devices while absolutely bad for some others. In order to see how the score of devices distributes, we get two histograms for both two goals. They are shown in Figure 5.5 and 5.6.

From Figure 5.5, we can see the shape is quite similar to the norm distribution for Goal1. And most devices have the score between 0.50 and 0.80. And the device with score less than 0.2 only
occupy a small fraction. From Figure 5.6, we can see the distribution is much flatter for Goal2, and more devices have score less than 0.2. So from these two pictures, we can see that prediction for Goal1 performs better than the one for Goal2.

Now let us look at the feature importances of these two models. The top 10 ones are shown in Figure 5.7 and 5.8.

From these two pictures, we can see IAQ is most important for the baseline predictor to predict whether Goal1 and Goal2.

### 5.5.2 Predictor with different external factor as additional feature

Then for each external factor, we add one of them as additional feature each time to see whether the added factor improves the prediction performance by using the t-test. And fortunately, we find that all these external factors can improve the prediction performance significantly. Also we have tried different ways of transforming those external factors to features, and we find the most appropriate way for each factor, which gives us the most improvement and is adopted by us finally. Then the prediction results for each additional feature as well as the final predictors for two prediction goals are shown in Table 5.3 and 5.4. For confidentiality, the name of features are represented as $f_{1}^{ext}$ to $f_{7}^{ext}$.

<table>
<thead>
<tr>
<th>On/Off prediction</th>
<th>$f_{1}^{ext}$</th>
<th>$f_{2}^{ext}$</th>
<th>$f_{3}^{ext}$</th>
<th>$f_{4}^{ext}$</th>
<th>$f_{5}^{ext}$</th>
<th>$f_{6}^{ext}$</th>
<th>$f_{7}^{ext}$</th>
<th>Final predictor</th>
<th>Baseline predictor</th>
</tr>
</thead>
<tbody>
<tr>
<td>count</td>
<td>1011</td>
<td>1011</td>
<td>1011</td>
<td>1011</td>
<td>1011</td>
<td>1011</td>
<td>1011</td>
<td>1011</td>
<td>1011</td>
</tr>
<tr>
<td>mean</td>
<td>0.656</td>
<td>0.624</td>
<td>0.679</td>
<td>0.643</td>
<td>0.634</td>
<td>0.625</td>
<td>0.843</td>
<td>0.990</td>
<td>0.616</td>
</tr>
<tr>
<td>std</td>
<td>0.001</td>
<td>0.000</td>
<td>0.027</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.119</td>
<td>0.000</td>
<td>0.186</td>
</tr>
<tr>
<td>25%</td>
<td>0.559</td>
<td>0.505</td>
<td>0.591</td>
<td>0.524</td>
<td>0.514</td>
<td>0.520</td>
<td>0.784</td>
<td>0.988</td>
<td>0.499</td>
</tr>
<tr>
<td>50%</td>
<td>0.671</td>
<td>0.635</td>
<td>0.690</td>
<td>0.650</td>
<td>0.646</td>
<td>0.864</td>
<td>0.994</td>
<td>0.629</td>
<td>0.757</td>
</tr>
<tr>
<td>75%</td>
<td>0.778</td>
<td>0.756</td>
<td>0.789</td>
<td>0.778</td>
<td>0.776</td>
<td>0.756</td>
<td>0.925</td>
<td>0.997</td>
<td>0.757</td>
</tr>
<tr>
<td>max</td>
<td>0.987</td>
<td>0.989</td>
<td>0.996</td>
<td>0.991</td>
<td>0.992</td>
<td>0.985</td>
<td>0.997</td>
<td>1.000</td>
<td>0.983</td>
</tr>
<tr>
<td>p value</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>nan</td>
<td>0.000</td>
</tr>
</tbody>
</table>

Table 5.3: Experiment results with each external factor added as well as the final predictor for On/Off prediction
CHAPTER 5. OPERATING MODE PREDICTION FOR AUTOMATIZATION

### Table 5.4: Experiment results with each external factor added as well as the final predictor for three modes prediction

<table>
<thead>
<tr>
<th></th>
<th>( f_1^{ext} )</th>
<th>( f_2^{ext} )</th>
<th>( f_3^{ext} )</th>
<th>( f_4^{ext} )</th>
<th>( f_5^{ext} )</th>
<th>( f_6^{ext} )</th>
<th>( f_7^{ext} )</th>
<th>Final predictor</th>
<th>Baseline predictor</th>
</tr>
</thead>
<tbody>
<tr>
<td>count</td>
<td>1011</td>
<td>1011</td>
<td>1011</td>
<td>1011</td>
<td>1011</td>
<td>1011</td>
<td>1011</td>
<td>1011</td>
<td>1011</td>
</tr>
<tr>
<td>mean</td>
<td>0.544</td>
<td>0.520</td>
<td>0.591</td>
<td>0.529</td>
<td>0.534</td>
<td>0.517</td>
<td>0.697</td>
<td>0.977</td>
<td>0.509</td>
</tr>
<tr>
<td>std</td>
<td>0.243</td>
<td>0.256</td>
<td>0.220</td>
<td>0.255</td>
<td>0.253</td>
<td>0.261</td>
<td>0.188</td>
<td>0.028</td>
<td>0.262</td>
</tr>
<tr>
<td>min</td>
<td>0.004</td>
<td>0.001</td>
<td>0.006</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.011</td>
<td>0.792</td>
</tr>
<tr>
<td>25%</td>
<td>0.359</td>
<td>0.309</td>
<td>0.429</td>
<td>0.328</td>
<td>0.347</td>
<td>0.308</td>
<td>0.582</td>
<td>0.972</td>
<td>0.295</td>
</tr>
<tr>
<td>50%</td>
<td>0.553</td>
<td>0.526</td>
<td>0.613</td>
<td>0.529</td>
<td>0.547</td>
<td>0.532</td>
<td>0.721</td>
<td>0.987</td>
<td>0.516</td>
</tr>
<tr>
<td>75%</td>
<td>0.753</td>
<td>0.737</td>
<td>0.766</td>
<td>0.745</td>
<td>0.747</td>
<td>0.739</td>
<td>0.849</td>
<td>0.994</td>
<td>0.735</td>
</tr>
<tr>
<td>max</td>
<td>0.986</td>
<td>0.994</td>
<td>0.995</td>
<td>0.995</td>
<td>0.994</td>
<td>0.994</td>
<td>0.995</td>
<td>1.000</td>
<td>0.994</td>
</tr>
<tr>
<td>p value</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>nan</td>
</tr>
</tbody>
</table>

From above two tables we can find that each of these external factors improves the prediction performance whether for On/Off prediction or three modes prediction. And the improvement is significant supported by the p values that are all less than 5%. And finally we get two predictors for two prediction goals with all external factors added as additional features, which gives us satisfactory score with 0.990 and 0.977 separately. Then the F1 score distribution is shown as Figure 5.9 and 5.10.

**Figure 5.9: F1 score distribution of final predictor for Goal**

**Figure 5.10: F1 score distribution of final predictor for Goal 2**

Compared with the Figure 5.5 and 5.6, we can find the histograms are shifted a lot to the right, as we can find for most devices we get an acceptable result (weighted F1 > 0.9)

Now we look at the feature importance shown in Figure 5.11 and 5.12 for the two final predictors.

**Figure 5.11: Top 10 important features of final predictor Goal 1**

**Figure 5.12: Top 10 important features of final predictor Goal 2**
From above two pictures, we can see that there are four external factors in the top 10 most
important features for On/Off prediction and six external factors in the top 10 for three modes
prediction. Notice that for external factor 7, we have two indicated by suffix _1 and _2 because of
the one-hot encoding.

In this section, we mainly study the impact of the extra feature on prediction performance
and validate its positive impact in polishing the predictor through t-test. Through these feature
engineering jobs, we finally get two predictors for Goal1 and Goal2 with acceptable prediction
performance (average weighted F1 score 0.990 and 0.977 respectively).
Chapter 6

Conclusions

In this chapter, we draw main findings in Section 6.1. Discussion of limitation and deficiency of our project in terms of the data and methodology are given in Section 6.2. Suggestions and possible future researches are given in Section 6.3.

6.1 Main findings

In this thesis, we mainly studied the device log data combing external air quality data from CNEMC[1] through two parts according to the informed data mining process. The first part as data understanding is intended to know more and better about user behaviours of using this product from the device logs. The second part as model building is intended to establish a prediction model using machine-learning techniques based on previous understanding.

Specifically, in data understanding part:

- Four daily patterns regarding power state of the device are discovered by using clustering technique.
- We defined three metrics to analyse user behaviours from different aspects:
  - Aggregated ratio of different operating modes.
  - Histogram of ratio of different operating modes.
  - Distribution of ratio of different operating modes.
- We analyzed the impact of external factors on user behaviours using above different metrics.

These findings show that logs of IoT devices can be used to study user behaviours, the user behaviours can be measured in different metrics that are dependent on the interests and the type of the device. Also the experiments in this project can be extended to deal with other similar devices to get deeper insights about how people use the device in different conditions.

Then in model building part, domain knowledge from first part is utilized to build the model. For each factors analysed in first part, we set up an pair of comparison experiments to study its impact on prediction performance, which is also regarded as validation of our previous analysis. After confirming the impact of each factor, two final prediction models that predict power state and operating mode respectively are established using random forest algorithm. And both predictors give us acceptable results and lay solid ground for future operating mode automatization.

Above are the main findings and achievements in our project. This project shows that the informed data mining process is a valid and appropriate method to study user behaviours and achieve useful use case through IoT device logs.
6.2 Limitations

There are several limitations regarding the data set as well as methodology we adopted here.

- Log collection period for most device is not long enough, which makes analysis may suffer from lack of persuasiveness because small data may not be an adequate representation of the user behaviours.

- We cannot distinguish power off and Internet down where missing data occurs.

- It is hard to distinguish between good usage and bad usage since there is no other background information, which makes us have to assume the log always reflects real usage situation.

- The analysis is not complete for some factors in Section 4.3.2. For example EAQ could be split in a more refined way when using $M3$.

- It is hard to tightly link the previous analysis target to the later prediction goal. For example it is not quite visible to see the relation between mode ratio distribution and direct mode predicting.

6.3 Guidelines for future work

Besides the strategies that we can do to avoid the limitation proposed in Section 6.2, there are some other works to be done.

- Some other prediction forms explained in Section 5.1 could be tried to see how they perform.

- In terms of current prediction goal, other algorithms could be tried to see how they perform.

- We could take the variety of user behaviour over time into account, and try some streaming data prediction model which serves for the better application in reality.
Bibliography


