MARMARA UNIVERSITY

INSTITUTE FOR GRADUATE STUDIES
IN PURE AND APPLIED SCIENCES

INTERNET OF THINGS BASED WASTE AND SERVICE MANAGEMENT IN QUICK SERVICE RESTAURANTS

KEREM AYTAÇ

MASTER THESIS
Department of Computer Engineering

ADVISOR
Asst. Prof. Ömer KORÇAK

ISTANBUL, 2018

MARMARA UNIVERSITY
INSTITUTE FOR GRADUATE STUDIES
IN PURE AND APPLIED SCIENCES

INTERNET OF THINGS BASED WASTE AND SERVICE MANAGEMENT IN QUICK SERVICE RESTAURANTS

KEREM AYTAÇ
(524112905)

MASTER THESIS
Department of Computer Engineering

ADVISOR
Asst. Prof. Ömer KORÇAK

ISTANBUL, 2018

MARMARA UNIVERSITY

INSTITUTE FOR GRADUATE STUDIES

IN PURE AND APPLIED SCIENCES

Kerem AYTAÇ, a Master of Science student of Marmara University Institute for Graduate Studies in Pure and Applied Sciences, defended his thesis entitled "INTERNET OF THINGS BASED WASTE AND SERVICE MANAGEMENT IN QUICK SERVICE RESTAURANTS", on …/…/2018 and has been found to be satisfactory by the jury members.

Jury Members

Asst. Prof. Dr. Ömer KORÇAK	(Advisor)	
Marmara University		....
Asst. Prof. Dr. Müjdat SOYTÜRK	(Jury Member)	
Marmara University		....
Asst. Prof. Dr. Ayşegül TÜYSÜZ ERMAN	(Jury Member)	
Işık University		....

APPROVAL

Marmara University Institute for Graduate Studies in Pure and Applied Sciences Executive Committee approves that Kerem AYTAÇ be granted the degree of Master of Science in Department of Computer Engineering, Computer Engineering Program on …/…/2018. (Resolution no: ………………)

Director of Institute

Prof. Dr. Bülent EKİCİ

ACKNOWLEDGMENTS

I would like to thank all those who supported me and made this research project possible. At the top and the first, I would like to thank to my advisor Asst. Prof. Dr. Ömer KORÇAK for his patience, full-support, complementary actions, neutral feedbacks and all suggestions which makes this thesis to improve. Moreover, I would like to call a little sorry for making him sleepless while building the thesis up at some nights by hoping that it will be accepted.

I also would like to present my one of the greatest thanks to my company Ata Technology Platforms, which enlightens me about vital gaps of the sector and provides many possibilities including an environment to play, many tools, human-power and advisory to fix and implement this thesis.

The last but never the least, I would like to thank to my lovely wife, Eda AYTAÇ who constructs me a peaceful working area in home and adopts a "thesis police" role that follows strict working rules and coordination.

And my whole family and friends who makes me feel so good at anytime and anywhere.

The research was supported by Ata Technology Platforms for the conferences and for many expenses.

May, 2018 **Kerem AYTAÇ**

TABLE OF CONTENTS

ÖZET

Hızlı Servis Restoranlarında Nesnelerin İnterneti Tabanlı Atık ve Hizmet Yönetimi

Nesnelerin Interneti; elektronik cihazlar, yazılımlar, sensörler, işleticiler ile gömülü nesneler topluluğudur ve internet üzerinden birbirleri arasında veri alışverişi yaparlar. Yıllardır fazlasıyla trend olan bir konu olup bir çok yerde bir çok şeyin yerini almış ve bunları daha kolay ve akıllı otomasyonlarla uzaktan kontrol edilebilir hale getirmiştir. Bu gibi yerlerde insanın rolü gitgide yok olmakta ve sensörler, işleticiler, geçit cihazları bu yükü insanlardan alıp ölçülmesi istenen değerleri otomatik olarak ölçüp bunu ağ içerisinde iletebilir duruma geçmektedir. Bu cihazlar bazı ön-tanımlı kurallar, yapay zeka ve makine öğrenme yöntemleriyle insanların yerine karar alabilirler. Uç hesaplama; (çiy hesaplama) düşük gecikme, ekstra kaynak, ağ kısıtları, problemli bağlantı ortamları ve gerçek zamanlı karar verme ihtiyaçları sebebiyle bu alanın vazgeçilmezi haline gelmiştir. Hızlı servis restoranlarında, bir çok atık yönetimi ve hizmet optimizasyonları insanlar ya da önceden hesaplanıp, simüle edilmiş kağıt üstü bilgiler tarafından yönetiliyor. Bu tez çalışmasında hızlı servis restoranları için, pil ömrü, performans, gecikme, güvenlik, yerinde erişilebilirlik gibi hem yazılım hem de donanımla ilgili sektör bazlı veya bazı genel iyileştirmeleri nesnelerin interneti mimarisi kurmak için göstereceğiz. Ayrıca atık azaltma ve hizmet optimizasyonu sağlayabilmek için uç hesaplama uygulaması olarak; sensor değerlerini işleyerek, anlamlı bilgiler çıkardıktan sonra veri bütünlüğü sağlayıp, hatta daha da önemlisi bu verilerin desenlerini öğrenip bir takım tahminler, alarmlar ve akıllı kararlar verilecektir.

May, 2018 **Kerem AYTAÇ**

ABSTRACT

Internet of Things Based Waste And Service Management In Quick Service Restaurants

INTERNET OF THINGS (IoT) is the population of “things” which is embedded with electronics, sensors, software, actuators, and tunneled into the internet to retrieve and transfer data with each other [1] and also has been hyped for years, and takes so many thing’s places in everywhere and makes many things easily and remotely controllable with smart automations. Human role in such areas are about to be vanished and sensors, actuators, gateways take over the workloads from human-being by generating the values which are desired to be measured, and transferring them within the network. They also take decisions with some preset rules, artificial intelligence or machine learning methods on behalf of humans. Edge computing (a.k.a Dew Computing) became a vital as there is a huge amount of requirement of low-latency, extra resources, network restrictions, loose connections, real-time decisions, etc. In quick service restaurants, many waste management and service optimizations are human or paper-based which contains pre-calculated or pre-simulated values. In this thesis, we propose an IoT architecture for quick service restaurants with some both general and sector-specific improvements in terms of either hardware and software such as battery-life, performance, latency, security, in-place accessibility and describe various edge computing applications including processing the sensor values, extracting meaningful information, providing data integrity and more importantly learning the data patterns to present predictions, create alerts, or make some intelligent decisions to provide waste minimization and service optimization.

May, 2018 **Kerem AYTAÇ**

SYMBOLS

μ	: Mean
C^{o}	: Celcius
$\boldsymbol{D_b}$	: Bhattacharya Distance
BC	: Bhattacharya Coefficient
$\boldsymbol{\sigma}$	: Variance
$\sum_i$	: Sum
$\boldsymbol{\mathcal{N}}$	: Normal Distribution
det	: Determination

ABBREVIATIONS

IoT	: Internet of Things
QSR	: Quick Service Restaurant
ZB	: Zettabytes
DMZ	: Demilitarized zone
CPU	: Central Processing Unit
GPU	: Graphic Processing Unit
IEEE	: Institute of Electrical and Electronics Engineers
MAC	: Media Access Control
QoS	: Quality of Service
WLAN	: Wireless Local Area Network
GHz	: Gigahertz
Wi-fi	:Wireless Fidelity
LAN	: Local Area Network
Mbit	: Megabits
PHY	: Physical
Kbps	: Kilobits per second
LE	:Low Energy
ACK	: Acknowledge
Pub-Ack	: Publish Acknowledged
RFID	: Radio Frequency Identification
API	: Application Programming Interface
gr	: Grams
TLS	: Transport Layer Security
Regex	: Regular Expression
MD5	: Message Digest 5th version
SHA	: Secure Hash Algorithm
Nonce	: Number of once

LIST OF FIGURES

LIST OF TABLES

PAGE

1. INTRODUCTION

1.1. Introduction To Internet of Things

As in the abstract mentioned, Internet of Things allows objects to be interacted and/or controlled remotely from an existing network architecture, [2] by creating good and catchy opportunities for more direct integration of the physical world into computer-based systems, also results with enhanced efficiency, accuracy and economic benefit besides to human intervention avoidance [3]. In IoT World, there is 6.4 billions of connected IoT objects in 2016, which is %30 more than 2015 and estimated that it will grow up to 20.8 billion objects by 2020. In 2016, 5.5 million brand new things will be connected every day [4]**.** IoT objects are generating massive amount of data per day, and it is estimated that annual generated data by IoT objects will reach to 600 ZB(Zettabytes) by 2020, corresponding to 600 Billion of Terabyte. This is 275 times higher than projected traffic flowing from data centers to end users or devices which is 2.2 ZB, and also 39 times higher than total projected data center traffic corresponds to 15.3 ZB [5]**.** Assuming those massive numbers of objects and their data traffic, there must be remarkably security and data efficiency considerations.

1.2. Introduction To Quick Service Restaurants

Quick Service Restaurants (a.k.a Fast-Food) are attracting much more customers because of its low-cost and time-efficiency for people and its number increasing remarkably year by year, and in USA this annual growth occurs with a ratio of nearly 2% [6] The global revenue of quick service restaurants is more than $570 billion, which is bigger than the economic value of most countries. Only in the United States, the revenue was approximately $200 billion in 2015. There are more than 200,000 quick service restaurants in the United States and it is estimated that 50 million Americans eat at one of them every single day. Fast food industry employs more than 4 million people which means a significant amount of human source is dedicated to this industry. [7]

In such a massively growing and mostly preferred sector, quick service and the least waste policy is strictly followed. Here, “waste” can be split into two definitions. First one is, food waste that is, waste of any raw materials included by food product or consumable materials by cooking the food and disposal of a ready-to-consume product on the service platform. The other one is, package waste, that is, disposal of the product that is bought by a consumer without consuming. QSRs should pay more attention to food waste and keep as much as low that they could. Every year, 1.3 Billion of Tons are wasted or 1 of 3 of the foods are lost or disposed [8] .A report by Natural Resources Defense Council in 2012 asserts that 40% of the foods are lost or wasted in USA [9] . Moreover, some researches reveal that size of the portions and human food-consuming capacity affects directly these waste ratios. Averagely, 17% of foods left without eaten due to more food service than demanded. In addition to this, because of company policies like McDonald’s, any product or raw material that is not served or consumed within 7 minutes should be disposed, thus, causing a disposal of the foods by 10% [10].

In quick service restaurants, production and consumption sometimes cannot be monitored and handled by being out of control especially within the rush hours. Lots of customer queue for eating and restaurant skips to a fast production phase to meet the demand either quickly or efficiently. Because unlike the ordinary table-serviced restaurants, customers expect to get their food within a couple of minutes, not much more. That is why it is called “quick” service restaurant, and the product, which is sold there, is called “fast” food.

1.3. History Of Drill Down To Fog From Cloud Networking For IoT

At the very beginning, Cloud Networking was set as destination for storing and processing data. The main subject mentioned here is all data produced by things sent to cloud via various methods. All the data are stored here and any clients can consume as much as they want them by applying algorithms where needed. However, this need has started to be inevitable to change from now on. Storing all required/redundant data in cloud caused a requirement for either huge amount of resources or mass data traffics. Besides, a need for communication between things and because any endpoints or sinks (users, computers, databases, reporting tools etc.) has to wait for the data to arrive at cloud and to be processed, Fog Networking method appeared to provide these needs and take very important place in terms of resolving the issues rises from cloud networking.

Providing this, any nodes located between things and cloud have the ability of filtering and bypassing the data directly to clients without sending to cloud and having more advantages. Amongst these advantages, we can count less latency, shorter paths between client and server, better handling of security methods, mobility, efficiency for real-time processes, lower possibility of having a man-in-the-middle attack [11]. On the contrary of cloud networking device population, fog networking devices are more of them. If we instance cloud devices in terms of count, that would be thousands of them like data centers, servers, firewalls in DMZ etc. So, to clarify fog devices, it would be millions of them like mid-servers, nodes, routers, base stations, relay servers, main/regional gateways etc. As long as we drill down from cloud to ground, devices count increment dramatically.

1.4. History Of Drill Down To Edge From Fog Networking For IoT

Cloud or Fog computing is the most common way to store or process the generated data. However, in many areas, especially in our case, utilizing the power of sensor-side or gateway-side edge computing will have significant advantages as described below.

Firstly, communicating with the cloud before any action may cause high delay, especially if there exist some connectivity problems. On the other hand, edge computing would reduce delay, minimize the effects of connectivity problems and enable real time analysis and responses. For example, a camera can count and classify people count in its processor or values received from weight-meter may trigger some alerts when an extraordinary material is disposed to it. This way it would be possible to intervene quickly if something is going wrong, and quick actions can be triggered to confront the problems.

Secondly, most of the sensors (such as temperature or humidity meters) are tended to produce lots of values most of which are the same with the previous ones. Sending all these values to cloud significantly increases the traffic intensity and also fills the storage in the cloud. Another example is camera, if camera sends all the captured frames to cloud to count how many people are in/out, it would kill the network bandwidth. Therefore, data should be processed before sending to the cloud.

Moreover, with the recent technological advancements, there exists remarkable amount of resource powers in gateways, and even in sensors. By using these resources via edge computing, it can be assumed that overall amount of resources in the ecosystem would increase.

Edge computing can also provide high-level security by implementing some edge-side security functions between sensors and the cloud. To talk about it detailed, values can easily be secured at the cloud side. But, if you do not apply any security at the edge side, but only cloud side; it could be instanced with this story: You have a child with bags of golds on its back. You want to send your child to a well-secured bank from your home on foot on his own via kilometers of way. Golds can be isolated in bank, it is a good thing. But, what about transfer? Your child is very vulnerable to any effects outside. Even a little puppy on the way can cause the bags left at the side of the way and make itself pet by your child. To sum up, you have to secure your value from the very beginning to the end.

With all these advantages, it is evident that intelligent edge computing functions will boost the system performance in many aspects. Many smart and heavyweight jobs can be done in the gateway. Some little corrections and little algorithms can be applied in basic sensors. Also, heavy image processing jobs can be done in camera as the cameras are specialized for these jobs. Some security protocols will be applied at the network hardware.

If we categorize what we benefit from with general titles to sum up:

- Low-Latency
- Less redundant data
- Real-time analysis and responses
- Extra resources for ecosystem
- High-Level security

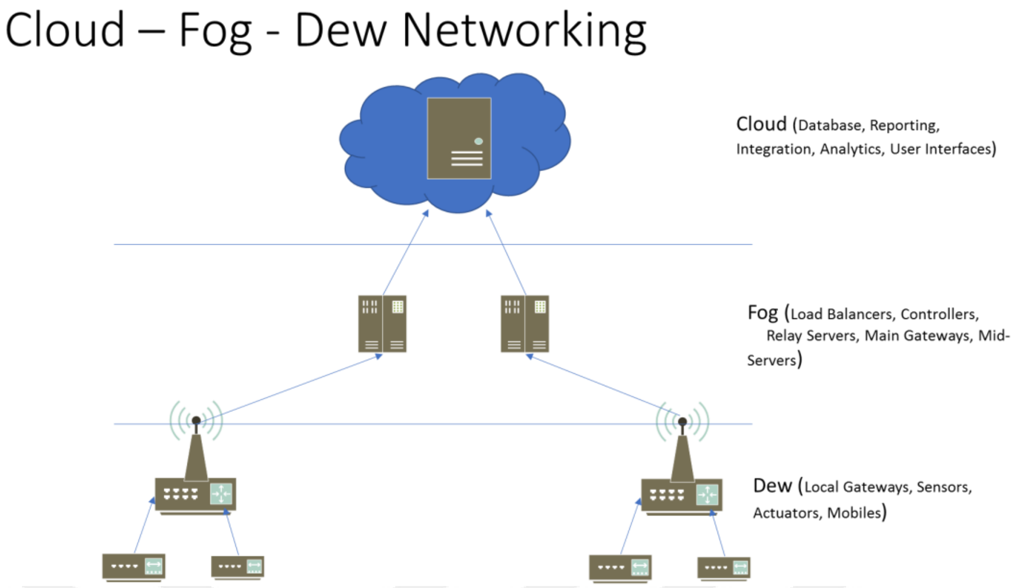

Figure 1.1. From cloud to edge networking, representing the layers

1.5. Back To IoT Fundamentals

Many basic IoT methods will be implemented in this architecture. Sensors are the vital organs of an IoT organism. Little, primitive sensors are managed by smarter microchips which can make sense the outputs of the sensors in terms of voltage or something else.

A gateway is a must in a local area which plays role as the brain of that area. This gateway is a gate to the way goes through cloud. This gateway is the smartest one within the local area and runs a high-level operating system on it containing some random-access memory, hard-disk, a CPU, maybe a GPU as hardware specification. Having these make anyone ride a horse at gallop easily on this architecture with many possibilities.

Communication between sensor and gateway could be via Wi-fi, Ethernet, Bluetooth, ZigBee protocol or else. Wireless protocols are the most preferable ones as it adapts and lives in many changing environments which makes it sustainable.

A Wi-Fi uses wireless standards by IEEE 802.11 as WLAN. It originally uses 2.4 GHz frequency band where later versions of WLAN types, it may vary to 5 GHz. As the years gone, better standard types appeared in terms of distance, bandwidth, interruption-proof etc. Those types are can be listed in Table 1.1.

It is very important which communication method is preferred. Because there are lots of things to consider in IoT world, such as battery-life, bitrate, distance etc. So we have to be mean as much as possible, and select the feasible one. If we don't have wired system, so we have three good wireless system. Wi-Fi, ZigBee and Bluetooth.

WiFi means a local area network (a.k.a LAN) which tunnels to internet with a limited range of area. It can be installed for your home network or in public places like airports which is the most common one. WiFi is more like star network which can be assumed as a star network where all nodes or devices connected to a single hub. This star topology creates an opportunity to add or remove any nodes without affecting the others that resides in the network. [12]

802.11 protocol	**Release date**	**Fre-quency**	**Stream data rate**	**Approximate range**	
		(GHz)	**(Mbit/s)**	**Indoor**	**Outdoor**
802.11-1997	Jun 1997	2.4	Up to 2	20 m (66 ft)	100 m (330 ft)
a	Sep 1999	5	Up to 54	35 m (115 ft)	120 m (390 ft)
		3.7			5,000 m (16,000 ft)
b	Sep 1999	2.4	Up to 11	35 m (115 ft)	140 m (460 ft)
g	Jun 2003	2.4	Up to 54	38 m (125 ft)	140 m (460 ft)
n	Oct 2009	2.4/5	Up to 288.8	70 m (230 ft)	250 m (820 ft)
			Up to 600		
ac	Dec 2013	5	Up to 346.8	35 m (115 ft)	
			Up to 800		
			Up to 1733.2		
			Up to 3466.8		

Table 1.1. 802.11 Network PHY Standards **[13]**

ZigBee technology is a protocol designed for carrying tiny amounts of data over a short distance where it also consumes very little power. Its being battery-life friendliness makes this protocol also inevitable. On the contrary of WiFi, it’s a mesh networking standard, meaning each node in the network is connected to each other easily. Unlike WiFi, it implements the standards of IEEE 802.15.4.

Bluetooth is also very popular communication method known by being a low energy consumer, but with a short distance of travel. Bluetooth is widely used on our mobile phones.

Comparing them each other makes our way to some results that we can overview and decide how, where and when to use them.

	Low Energy Bluetooth	**ZigBee**	**Low Power WiFi**
Frequency(MHz)	2402-2482	868-868.8, 902-928	2400-2500
Channels	3	16	3
Max Potential Data Rate	1 Mbps	250 Kbps	54 Mbps
Range	10m	100+m	30m
Complexity	Complex	Simple	Complex
Extendibility	No	Yes	Yes
Power Profile	Days	Months/Years	Hours

Table 1.2. Specification Comparison Table of LE Bluetooth vs. Zigbee vs. Low Power Wifi

Sensors communicate with gateway via these protocols and transfer anything they want. However, at that point, these things must be transferred to cloud as well. Thus, a lightweight and a secure method, MQTT (Message Queueing Telemetry Transport) protocol is a good choice. MQTT stands for Message Queueing Telemetry Transport. It is a publish/subscribe, very simple and lightweight messaging protocol, designed for constrained in other words disabled devices with low-bandwidth, high-latency or unreliable networks. This protocol is especially designed for minimizing the network throughput and bandwidth where it also ensures reliability and some degree of assurance of delivery.

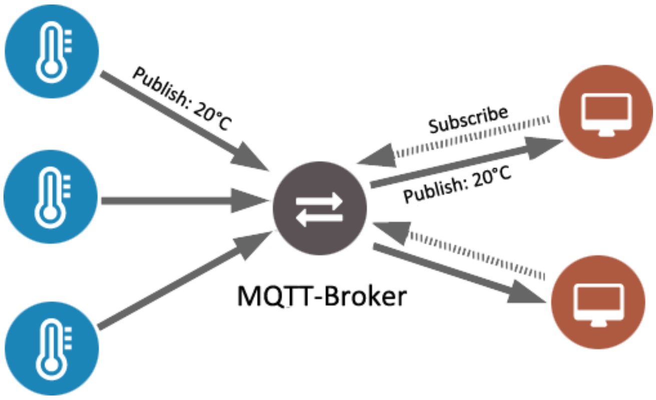

Figure 1.2. How MQTT works

Gateway **publishes** the data to MQTT-Broker (a server), and regarding of what Quality of Service mode selected, MQTT-Broker ACKs or not. This is a send action of gateway to cloud.

It **subscribes** to a topic to retrieve a data from desired sets of domains. If MQTT-Broker needs to send a data to an IoT device, it sends the data according to subscriptions. This is a receive action of gateway from cloud.

While publishing data to cloud, many Quality of Service methods can be used according to preferences within performance – reliability tradeoffs. Three levels of QoS can be used.

QoS 0 level: Publish and Forget, no acks. This will increase performance, and lower bandwidth, but decreases the reliability in case of packet loss.

QoS 1 level: Publish and only receive pub-ack, which verifies that broker received publish

QoS 2 level: Publish and receive pub-ack, tell broker "I received pub-ack", then broker tells pub-completed. This will max up the reliability, where creates overhead on low network bandwidth.

1.6. Motivation

In quick service restaurants, they cannot adopt a reactive service approach, which you only produce when an order arrives. If they adopt this approach, service time becomes longer and longer, causing a loss of customers or low-rating reviews full of "we have been put hungry", "I was brought here by my mother just after my primary school, now I'm leaving here to catch my meeting with my boss to consolidate the current fiscal year." comments in social media. This is not a desired thing for the chain-restaurant brand owners. Thus, most of them have a primitive predictive production approach like using past sale data by day and hour, and extracting “we have sold 100 of A product last Thursday at 3.p.m, so today is Thursday, and the time is about to be 3 p.m. So, let’s produce some A product and keep them in the chute”. However there is something they predicted wrong, it was sunny 30 C° and official holiday last Thursday, but now it’s snowing and storming and everybody trapped in their home. So, this kind of approaches are highly prone to vast amounts of wastes for unable to be sold products, and indirectly waste of money. Here to optimize this approach, restaurant supervisor or manager always monitors the weather, entrance, queue length, people density, and another condition occur in surroundings like football match or concert etc. So, they try to fix the last Thursday’s prediction manually, and decides for a production. But again, this causes a VIP staff like supervisor or manager, are nearly dedicated to this kind of duty, and prevents him/her doing their real and more productive, useful duties. And also, human is naturally prone to mistakes easily. Moreover, it might be very hard to grow such a staff because it requires a lot of experience, and work years, so if that staff resigns, you always have a risk of starting at the very beginning to grow another staff.

Moreover, there are many factors to measure that a key staff (manager/supervisor etc.) cannot handle easily, such as, deciding eating court temperature or humidity is at optimal value for customers, or weather is going to turn into a catastrophic state from a lovely day, or surrounding activities such as football match, concert will affect the density of people. These factors are the major things that can affect the restaurant service level. No key staff can wait until they get sweaty to decide if eating court is not at optimal value or check the internet how the weather is and will be or check activity sites to find out if there are popular activities around restaurant.

All the inputs told above results with an amount of waste and an interval of service time.

1.7. Related Works

Some waste management applications have been applied in IoT domain. For example, IOT Based Smart Garbage and Waste Collection bins (SGWC). This idea is based on creating some social awareness to reduce food waste through measuring and displaying the amount of food wasted and recycling the wasted food using some embedded systems. In this idea, human-beings are aimed to diminish the food waste by some social effects by their initiative senses and waste is focused to be converted into a reusable object like fertilizer [14], where we will directly decrease the waste amounts directly and won't let any human decide if they waste or not at that moment. System will handle many factors, which make them waste, and remove those factors by making human not to waste. Another work is arming a garbage vehicle with some sensors, and measure RFID-tagged weight bin while disposing into the vehicle, and send the related data to a center for catering companies. This idea aims to detecting and tracking the companies how much they disposed [15] . On the contrary, in this work, we directly focus on how to decrease the amount of waste. We design a system to handle many factors that increase the amount of waste, and remove those factors by making human not to waste. So far, we could not find any similar work that uses IoT to resolve waste problem in restaurants. So far, we could not find any specific works by using IoT to resolve waste problem in restaurants.

1.8. Our Contributions

Our solution builds a quick service restaurant oriented IoT architecture and creates many processes that can solve waste and service problems and bottlenecks. Many of the methods are currently used in different or similar areas, all of them specifically evaluated and eliminated to useful ones; and is adopted to our area by altering with some improvements and patching the wounded sides.

1.9. Organization

A brief overview of this thesis follows. First, in the next chapter, we start with giving relevant information and point out the related works fulfilled in IoT works in Quick Service Restaurants, again in the same chapter, materials and test domain explained during the development and test phase. We define the current workflow and problems which is experienced in QSRs. Using the current workflow, we propose an IoT architecture which takes over many human workloads, automatizes, heals and excels. Algorithms defined and showed how they used. How a physical IoT architecture can be installed to a QSR will be explained. Many of the solutions are applied and have the results. Then, in the Chapter 3, algorithms are compared with their results and methodologies. We conclude the thesis with further discussion and future work in Chapter 4 the last.

2. MATERIAL AND METHOD

2.1. The Experiment Domain

As a test domain, we have chosen a well-known quick service restaurant chain that produce hamburgers in Turkey with more than 500 restaurants spread to country. This will provide us a good play-around area.

2.1.1. How the experiment domain lives

They have many manual and human-supported workflows to provide some optimizations for waste and service like also mentioned at the challenges and obstacles title. The main method used in restaurants depend on an instant service level. This service level determines how many products (such as burgers, raw materials, side products etc.) will be pre-produced regardless of what is ordered by customers and kept in the “chutes” (food storage boxes).

One of the major challenges for key staff is querying for the latest sales data like told at previous title as well.

But of course, it's not that simple. In addition to this, they apply some formulas and it results with some service level intervals from 1 to 7. The first level (1) means everything is safe and sound, just calm down and sit back and wait for customers with a great relief. The last level (7) means, you have no time to breath, keep producing for the most popular and raw materials before ordered by any customer as much as calculated in the formulation paper. As long as you produce, shoot them to chute to stock. If approximately 10 minutes left per material, and still not sold, then pray and get ready for a mourning, because you have to dispose them. Now here is another bad news, key staffs are forced to inform what they have just disposed gram by gram. If they disposed half of meat, they should enter it to an excel sheet by grams.

But there are some drawbacks here. Such as, if so much of materials are disposed, they might want to hide it from the company, or they might miss to inform the real disposed values by human mistakes. So many side effects can disrupt a model like this and the outputs have very high error values.

2.1.2. How the experiment domain aimed to evolve

In this paper, we propose a fully automated and self-learning IoT architecture for Quick Service Restaurants. The proposed architecture consists of various sensors, actuators and data sources connected to cloud infrastructure via a gateway. Since most of the data is produced in the edge of the network, it would be more efficient and valuable to process the data at the edge-side. Previous work such as micro datacenter [16], [17], cloudlet [18], and fog computing [19] has been introduced to the community because cloud computing may not be the best option for data processing when the data appears at the edge-side firstly. [20]. Processing all the data at cloud level causes very high latency, connectivity dependency, requires remarkable amount of network bandwidth, data redundancy and waste of resource as the resource power of edge level devices are dismissed. In this study, we describe several benefits of edge computing in the proposed IoT architecture. We study several edge applications including data processing, security and machine learning. We provide some results obtained by deployment of proposed architecture in a real test domain, a well-known quick service restaurant chain.

2.2. IoT Architecture Installation Fundamentals In QSR

A well-decided and smartly-implemented IoT design should be installed within restaurant with low-cost as possible, and high-end results. We should define any requirements within that restaurant and workflows implemented for that brand. The design concept should vary from restaurant to restaurant, brand to brand and company to company, it is basically covered by Quick Service Restaurant essentials, though. So, these variations should be overviewed at the top, and any method that work in a specific sub-domain should not be generalized. In this architecture, any full approach is oriented nearly for specific single restaurant. This approach can be inherited from, modified, altered for others, but better not to implement verbatim. This can be assumed as human-being creation. Any human-being is similar to each other in terms of organism schema, but not the same either physically or logically. Human neuro-systems, briefly the brain is the main factor which creates the diversity.

In IoT, this diversity will be provided by edge computing, the brain of organism. Any restaurant will have their own brains within their borders and this brain will adapt the conditions and limitations. Herein this brain will be called as gateway, and its organs, receptors will be sensors.

2.2.1. IoT architecture installation - materials

As gateway, Raspberry Pi 2 or 3 is preferred as they are good micro-computers with GHzs of CPU, GBs of RAM which will not make us try to seek for giant machines like server computers and make itself useful. Windows IoT Core operating system is installed. This is a good advantage of programming at high-level, like C#. Any object-oriented, service-oriented methods can be easily implemented, even any enterprise level architectures are available. High-level programming increases the creativity and development skills while decreases the maintenance cost. It also provides a comfortable working area for developers as C# is supported by a high-end IDE Visual Studio which has a powerful debugging, testing, supporting mods and more. High-level programming languages also have powerful frameworks, libraries and API, that means we will profit them all. Basic sensors are supported by many microchips like Arduino so that sensor-level developments become possible as well. Communication between sensors and gateway fulfilled via ZigBee protocol where possible as it is battery-friend, and high-performance wireless communication method. It’s actually IoT oriented communication method. For some sensors which requires too much energy consumption, will be supported by ethernet energized by POE, moreover for some sensors which requires energy, but can not be handled with direct physical connection uses WiFi.

MQTT protocol is also used as a communication method between cloud and edge. This protocol also implements TLS 1.2 and username - password authentication security methods. A local network, which contains a switch for ethernet inputs, and also a wireless router which “pipes-out” to internet by creating a local wireless network area as well. This wireless network is protected with ordinary and up-to-date security methods.

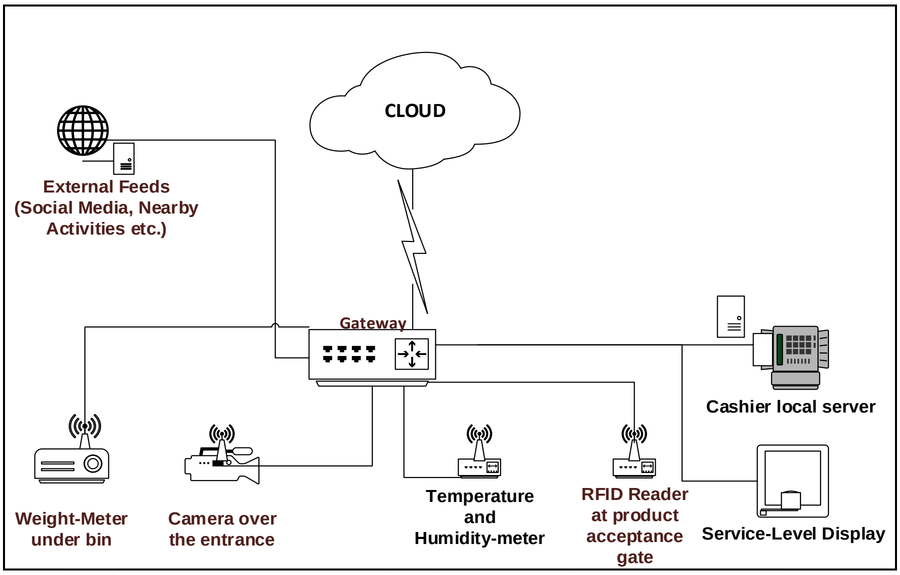

Figure 2.1. Overview of the general IoT architecture in a restaurant

To talk about the sensors included in the architecture is like below:

- **Temperature-meter** to measure the real temperature in the eating-area if it is bearable by the customers especially in corresponding peak times of summer or in winter. Another temperature-meter is in the cold storage room such as +4 C^o for short-term storages or –18 C^o for long term and non-durable materials. Because even 1 degree is very vital for any area mentioned here.
- **Humidity-meter** is also very important in such areas to keep the wellness of the materials or customers.
- **Weight-meter** is placed under the waste bin to weigh the bin with a high precision. This cannot classify the wastes but it can figure out to watch how much wastes are disposed.
- **Smart Camera** to count how many customers get in or out from the entrance, and how busy the queue line in front of the cashiers.
- **RFID-Reader** to watch and read the tags mounted on the product boxes while they are unloading from the delivery lorry into restaurant from the product-acceptance door, and another located reader at some key points to draw the life cycle and waypoints of the product boxes within the restaurant.

- **Cashier-Agent** is a cashier specific software, which can stream the instant sales data to desired domain.
- **Equipment maintenance sensors** are the sensors, which can identify the equipment, check the health status, and track the whole lifecycle of the equipment. These sensors can foresee any possible breakdowns in advance, and can warn the center that it may require a fix.
- **Service level indicator** is a simple one-digit indicator, which shows a digit from one to seven to tell the staff that service level production method should be adopted by then.
- **External Data Sources** like weather forecasts, surrounding activities, social media data etc.

2.3. Edge Computing In QSR

2.3.1. Sensor-side edge computing

Basic sensors have little memory to store some values. In temperature and humidity sensors, values are compared with previous ones. If any outlier value appears because of some physical effects, it will be ignored and disposed before sending to gateway. Let's say temperature meter creates 25 C^o value for 100 times consecutively and sensor has affected by an electromagnetic microwave in a very short time, and created 50 C^o and then again 25 C^o which is barely seen that it is an outlier. So we can ignore 50 C^o value. Advanced sensor like camera will do image processing algorithms to its captured frames, and will try to figure out how many people is in/out with a high precision and send the results to gateway. So, gateway don't have to bother. In addition, all sensors prepend the MAC address to the values before sending it to gateway. Therefore, that gateway could understand what the source of that value is.

2.3.2. Gateway-side edge computing

All gateways download a configuration file from cloud to identify the surrounding sensors and apply many rules on received values. In this paper, we will focus on three different gateway-side edge functions: Data processing, data integrity and machine learning.

2.3.2.1. Data processing

Gateway does not send raw data to cloud as it comes from sensors directly. It recognizes the sensor, extracts the value, and sends only the necessary value to cloud. Sensors generally create a string which includes an address part identifying the sensor (mostly a mac address), a sensor identifier part if it sends multiple different values from single sensor such as a combined temperature and humidity sensor in one device, and the value. Some sensors generate more complicated string and gateway needs to use some regular expressions to extract the values from these strings. To give an example, Figure 2.2 shows a string produced by a people-counting camera. This camera can produce multiple information at once like time-intervals, incoming or outgoing counts, address information, some id etc.

<FieldSeparator>SerialID=010101262626<FieldSeparator>SendACK=2<FieldSeparator>IncomingHumanCount=0<FieldSeparator>MeasurementEndTime=2018-01-10 16:39:59<FieldSeparator>MeasurementStartTime=2018-01-10 16:37:26<FieldSeparator>OutgoingHumanCount=0<FieldSeparator>PublicIP=NONE<FieldSeparator>UUID=70edd846-b31c-439d-97e6-fb6c2ebd548a

Figure 2.2. Example string generated by a people-counting camera

To extract only the incoming human count, the following regular expression can be used.

([\s\S]*IncomingHumanCount|<Field[\s\S]*?$)|[^\d]

where "\s" matches any white-space character, "\S" matches any character except a white-space character, "*" matches previous element zero or more times. [^\d] captures all characters but digits. _<Field[\s\S]*?$_ will also capture any characters just after "<Field".

“?” matches previous element zero or one times. “$” matches the end of the input, or the point before a final ‘\n’ at the end of the input. [21]

Using the power of regular expressions, gateway can extract any value from the received strings. Then, it analyzes these values and eliminates the redundant ones. It only sends valuable information to the cloud via MQTT protocol.

Now it is time to send to cloud destinations via Message Queuing Telemetry Transport (MQTT) protocol which is an asynchronous lightweight transfer protocol for IoT. Any first coming value queues with the highest priority and waits for their turn. A task scheduler pools the queue, and collects the values and send them to cloud and waits for the success response from cloud. If value fails to send to cloud due to some temporary connection issues or some limitations from cloud such as object size, valid object limitations etc., that value requeues with a lower priority and stamped with a near datetime value such as 1 minute later, if it is the first failure. If there is no queued item, low priority values can be retried to send. As long as high prioritized values exist, no lower ones will be processed. However, when that datetime comes, it will raise as high-priority again, and dequeues and requeues with high-priority rank. The more it fails, the later timestamp will be, increasing exponentially, until dequeued and disposed with a defined limit of timestamp.

In addition to this, a storage engine simply stores any data in the disk with some other information like cloud status (sent, retrying, failed, ignored etc.). Any data which is in in-memory queue flushes into disk with regular intervals. This makes the system robust, and if any device shutdown or failure occurs, in the next boot, the device retrieves any unsent data from disk, and retries. The data older than 7 days are purged to earn some space. This also renders possible edge-side data retrieval by any device.

2.3.2.2. Data integrity

Any messages sent from sensors to gateway, encrypted or not, could be exploited by man-in-the-middle. Figure 3 illustrates a simple scenario, where an attacker intercepts the messages sent from a sensor to the gateway and may modify, re-sent or replay these packets. To avoid this, nonce and digest can be used with a lightweight hashing algorithm.

With nonce, no message can be replayed; with a digest no message can be modified. Hashing process will be done at the edge side. Therefore this can be assumed as edge computing as well.

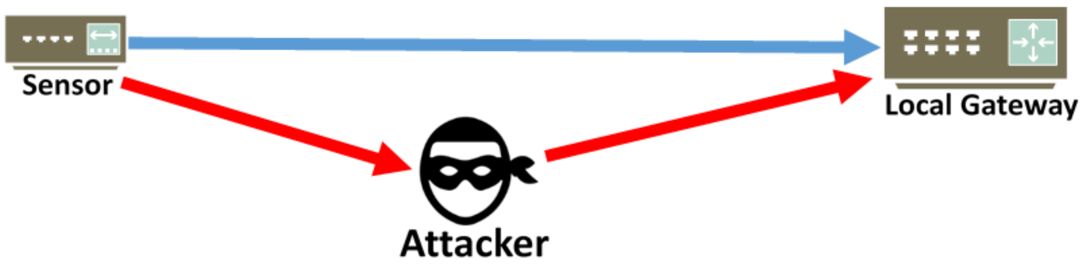

Figure 2.3. Man-in-the-middle between a sensor and local gateway

Unique Guid Nonce is created with the current datetime. A pre-defined or shared by gateway pin has been taken. All of them (nonce, datetime and pin) are hashed with an algorithm known by both sensor and gateway. This could be MD5 or SHA hashing algorithms. The resulting digest is sent together with Nonce and Nonce_CreatedOn to the gateway as shown in Figure 2.4. Gateway applies the same hash algorithm to validate the digest. Hence, the following rules at the gateway will prevent intervention of a man-in-the-middle.

- Any message which the nonce is manipulated will be ignored, because the digests won't meet at the gateway side.
- Any message with digest which isn't met with the sent one and re-created one in gateway, will be ignored.
- Any message with a Nonce_CreatedOn which is before more than, lets say 10 minute, will be counted as expired.
- Any message with the same headers (Nonce, NonceCreatedOn, digest etc.) will be counted as replay attack, as same nonce already cached by gateway and assumed as used, and will be ignored.

Nonce: {Unique Guid}
Nonce_CreatedOn: {Datetime.Now}
Message body (Encrypted or not)
Digest: Hash (Pin + Nonce + Nonce_CreatedOn + Data)

Figure 2.4. Message format between sensor and gateway

2.3.2.3. Machine learning

After extracting valuable data sent by the sensors, the next important and intelligent job is to learn data patterns and try to catch any events, anomalies or try to figure out some information about the state of the restaurant system. For this purpose, various machine learning algorithms can be applied. We will try to concretize this idea by describing two example applications: a) Intelligent weight-meter and b) Service Level Estimator and c) Smart Staff Allocator

a-) <u>Intelligent Weight-meter:</u> Without any machine learning, weight-meter collects data, processes them as described above and send them to cloud. However, after applying the learning algorithms over the collected data, it would be possible to see the patterns and recognize the outliers, anomalies, etc. This will enable the gateway to:

- Create an alert immediately if an outlier value comes from weight-meter such as longer than usual bin removals, or too heavy objects. Weight-meter now knows how long the average time of the bin removed to be cleaned and then replace; or what is the usual wastes and if any heavy objects or too many products are wasted; it creates an alert to authorized people.
- Find the lowest error of average of waste by time-intervals as it knows the outliers, and dismiss those values.

- Store a learning formula in the disk in exchange of all the values. An approximate value in a specific time in a specific day can be estimated by using this little piece of formula instead of a large dataset, in case a little error value is acceptable.

Weight-meter has been allowed to collect data for a week. Just after one week, data has been scattered onto a graph by hour: minute and value in terms of grams. Empty bin is approximately 2000 grams. Most of the values are gathered around 2000 to 5000. Some values seem to be outliers like the ones less than 2000 grams or greater than 5000 grams. These can be bin removals or some extraordinary wastes. In order to catch such events, it would be appropriate to apply a clustering algorithm such as K-Means clustering. Five clusters would be enough to identify the sets. Two of them are upper and lower outliers, and three of them are low, moderate and high values for ordinary values.

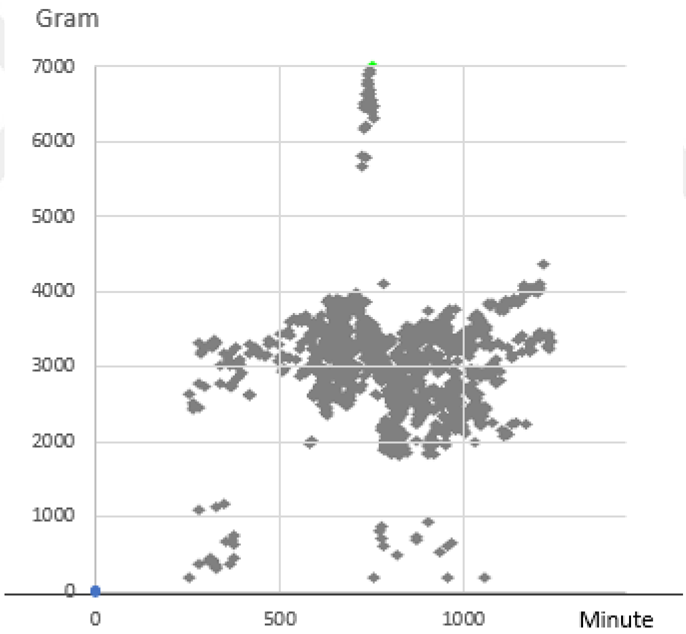

Figure 2.5. Weight-meter values (Grams by Day Time) collected within the given time.

As it can be barely seen, most of the values are gathered around 2000 to 5000. Some values seem to be outliers like the ones at <2000 grams or >5000 grams. These can be bin removals or some extraordinary wastes. But here is one thing to focus, machines don’t have eyes. But have brains stronger than us. We can apply a K-Means Clustering algorithm to cluster the values. 5 Clusters would be enough to identify the sets. Two of them upper and lower outliers, one of them the lowest floor of the ordinary sets where the other one upper floor and the last one is the most ordinary ones.

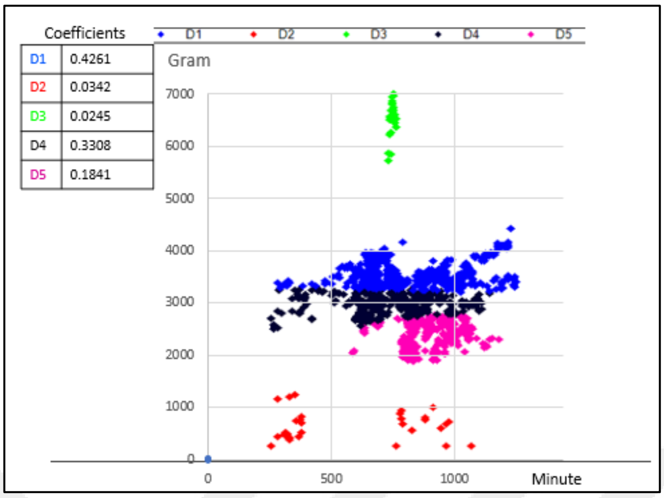

Figure 2.6. Weight-meter values clustered into 5 sets and colorized. Time is represented by minutes.

Figure 2.6 shows the obtained weight-meter values by time and the resulting clusters. Coefficients indicate the density of values per cluster. Red and green (uppermost and lowermost) clusters are the outliers; others can be assumed ordinary sets. For any new coming values, our K-Means model decides in-which cluster the value should be in. If any value falls into outlier set, it can create an alert or trigger an algorithm at the edge side or cloud side or anywhere.

In K-Means Clustering, we used Bhattacharyya distance to test how successful we clustered our dataset, and to detect the outlier clusters. In statistics, the Bhattacharyya distance scores the similarity for two discrete or continuous probability distributions. Bhattacharyya coefficient is the determinist fact which determines overlap amount between two populations. This coefficient determines the relative closeness of two samples and it shows the success rate of the similarity of classes. In other words, it tells you how successful you separated the classes while classifying a set of data.

For probability distributions p and q over the same domain X, the Bhattacharyya distance is defined as

$$D_B(p,q) = -\ln(BC(p,q)) \tag{2.1}$$

where

$$BC(p,q) = \sum_{x \in X} (p(x)q(x))^{\frac{1}{2}} \tag{2.2}$$

is the Bhattacharyya coefficient for discrete probability distributions. In our domain, x-axis is the sensor value as a floating number, y-axis is the time in terms of minutes, in some cases seconds or hours.

The Bhattacharyya distance, $D_B(p,q)$, between two classes (*p* and *q*) under the normal distribution can be calculated by extracting the mean and variances of two separate distributions or classes (assuming that the sensors generate non-faulted and precise values):

$$D_B(p,q) = \frac{1}{4}\ln\left(\frac{1}{4}\left(\frac{\sigma_p^2}{\sigma_q^2} + \frac{\sigma_q^2}{\sigma_p^2} + 2\right)\right) + \frac{1}{4}\left(\frac{(\mu_p - \mu_q)^2}{\sigma_p^2 + \sigma_q^2}\right) \tag{2.3}$$

where σ_p^2 is the variance and μ_p is the mean of the p-th distribution [15].

For multivariate normal distributions $p_i = \mathcal{N}(\boldsymbol{\mu_i}, \Sigma_i)$,

$$D_B = \frac{1}{8}(\boldsymbol{\mu_1} - \boldsymbol{\mu_2})^T \Sigma^{-1}(\boldsymbol{\mu_1} - \boldsymbol{\mu_2}) + \frac{1}{2}\ln\left(\frac{\det \Sigma}{\sqrt{\det \Sigma_1 \det \Sigma_2}}\right) \tag{2.4}$$

where $\boldsymbol{\mu_i}$ and Σ_i are the means and covariances of the distributions, and

$$\Sigma = \frac{\Sigma_1 + \Sigma_2}{2} \tag{2.5}$$

For detecting whether a cluster is outlier or not, we calculate Bhattacharyya distance between every cluster. We do not care only the distance between the classes, but we also consider the similarity of the classes. That is why, we use Bhattacharyya distance, rather than Euclidian or Manhattan distance. Mahalanobis distance is a particular case when standard deviations are the same for two classes you work on. Evaluating Bhattacharya distance is more general method for any scenario regardless of how the standard deviations are, and more reliable than Mahalanobis distance. Therefore we use Bhattacharyya distance instead of Mahalanobis distance, which also reflects the difference between standard deviation among classes. [22] [23]

In our case, sensors are tended to be disrupted by environmental affects like electronic microwaves, temperature, signals outputs from wireless devices etc. So, they can mis-measure or be broken down for a while. So, there will not be an architectural excellence in the pattern or each cluster, and standard deviations could be very different amongst each cluster. Therefore Mahalanobis distance is dismissed in this method, as told above.

Just after clustering, all clusters are compared with each other by Bhattacharyya Distance and scored to figure out whether we are successful. Within a cluster, by randomly selecting sub-clusters, if a distance is close to zero, then it is OK. However, if a distance is non-acceptably far from zero in our data space, then our clustering might not have worked well. Moreover, if two separate clusters haven't acceptably far from zero value, again we might have been failed. In this case, we should throw another try with different way or algorithm, or create notification. Moreover, distance between ordinary and outlier clusters should be much more than distance among ordinary clusters. If there is no such distant cluster, there may not exist any outlier.

	D1	**D2**	**D3**	**D4**	**D5**
D1		16.1	16.8	**1.3**	3.7
D2	16.1		51.1	12.7	**6.4**
D3	**16.8**	51.1		28	35.4
D4	**1.3**	12.7	28		1.48
D5	3.7	6.4	35.4	**1.48**	

Table 2.1. Bhattacharyya Distances between each cluster.

Table 2.1. illustrates Bhattacharyya distances among every cluster pair shown in Figure 5. For each cluster, distance to the nearest cluster is shown in bold. As we can see in the table, none of the cluster values are close to zero, which indicates a separation among clusters. Ordinary clusters (D1, D4 and D5) are closer to each other (their distance to nearest cluster is not more than 1.5), while outliers (D2 and D3) can easily be detected by observing long distances to the nearest clusters.

The logic described in this subsection can be used for any sensors such as camera people counters. In case of an extraordinary people entrance, it can immediately detect the busyness and trigger any workflow. Similarly, this method can also be used to detect anomalies resulting from any type of attack that impersonate our sensor.

b-) <u>Service-Level Estimator:</u>

As described before, production service level is a value between 0 and 7 which determines how many products will be pre-produced regardless of what is ordered by customers and kept in the chutes. It is related to many factors such as date, time of day, outside temperature, nearby activities, previous sales, number of customers, change in the customer density, etc. However, it is not directly determined by any of these factors alone. For example, if a large group of customers arrive in a silent time period, this may not mean that new customers will continue to arrive. Here, we want the gateway to decide on the production service-level according to all these information. For this purpose, it is appropriate to use a classification algorithm like Naïve Bayes Classifier [24] which consists of two steps:

- Training step: Using a set of training data, the algorithm finds out the parameters of a probability distribution (assuming conditional independence between the features).
- Prediction step: Just after training and learning step, any test data can be classified after computing the probability belonging to each class, and choosing the class with largest probability.

Such a classifier will enable using many different kind of sensor data, training with them, and giving an output in terms of production service-level. In the training step, a key staff is required to take care the service-level device, and enter the service level manually after counting the customers, checking the formula based on previous sales, getting information from the call center that there is a football match nearby, etc. Gateway use this production service level information as a training input, but it knows more since it always gets information from internal IoT system, knows the acceleration in the customer density in advance, and incremental sales data within that hour. It also knows that waste-bin always fills up and recycles and also it checks the Internet and social media for the external events. However, gateway falls back and just watches. Its day will come later. It is now being trained.

After seven days, gateway can decide immediately from the inputs, and predict the production service-level without any human intervention and effort. It manages the staff like an orchestra chef. Raises the stick when necessary and increase the production, lowers the stick if it comes to a peaceful moment.

Here is an experiment performed in a different restaurant which has weight-meter, people-counter and a production service-level. Although the other sensors and external data are missing in the experiment (which lowers the accuracy of the training), we will show that the results are acceptably good. We have used weight, people incoming count within 10 mins and time as an input, and a production service-level sensor value had been collected for a week. We created a simple formula within that week interval and set service-level counter automatically according to these sensor values to simulate a real restaurant operation. To briefly explain about the formula, any sensor value directly effects to level with the same trend, meaning that while sensor value increases, level is tended to increase by then. This formula also does not ignore some restaurant working style facts. We tried to fit it to real scenarios. So, we have joined all the values and spread onto a timeline. Time is also an input and can be considered as a sensor value, because in real scenario people do not mind eating fast food at 10 or 11 a.m. But, after 12 a.m., such as noon break for companies or evening times are starvation time of the people.

Time is again represented as minutes. We obtained nearly 9K data for training set and 1K data for testing. Sample row from the training data is as follows.

Weight	People Coming	Time	Service-Level
4200gr	50	1000 (4:40 pm)	**7**

Table 2.2. An example row for training data

Further information about our dataset:

Total Count	Service-Level
3	0
7	1
63	2
1226	3
1078	4
5142	5
664	6
816	7

Table 2.3. A pivot table of our dataset

According to Naïve Bayes algorithm with Gaussian Distribution which is an extension method of Naïve Bayes. When coping with continuous data, the continuous values associated with each class are distributed according to a Gaussian distribution. [25] Also, according to this distribution we find the averages and the variances of each inputs for specific service-level. Average is represented with μ and variance is represented with σ.

Here are the means and variance values of each class.

Class	People Count	Weight	Time
G0	N(x; μ = 6, σ^2 = 1E-12)	N(x; μ = 76,66, σ^2 = 409,33)	N(x; μ = 79,66, σ^2 = 316,33)
G1	N(x; μ = 12,42, σ^2 = 5,28)	N(x; μ = 872,14, σ^2 = 1015320,14)	N(x; μ = 304,28, σ^2 = 85781,57)
G2	N(x; μ = 19,26, σ^2 = 7,78)	N(x; μ = 851,80, σ^2 = 700556,44)	N(x; μ = 472,12, σ^2 = 93435,17)
G3	N(x; μ = 20,95, σ^2 = 15,9504131571062)	N(x; μ = 2202,10, σ^2 = 115023,64)	N(x; μ = 646,84, σ^2 = 10339,98)
G4	N(x; μ = 22,68, σ^2 = 26,52)	N(x; μ = 2666,16, σ^2 = 110498,13)	N(x; μ = 700,94, σ^2 = 18424,60)
G5	N(x; μ = 24,09, σ^2 = 48,77)	N(x; μ = 3249,74, σ^2 = 337610,54)	N(x; μ = 823,32, σ^2 = 31385,21)
G6	N(x; μ = 36,56, σ^2 = 16,34)	N(x; μ = 2972,0, σ^2 = 478729,99)	N(x; μ = 792,13, σ^2 = 35442,57)
G7	N(x; μ = 32,18, σ^2 = 146,43)	N(x; μ = 3917,34, σ^2 = 1920377,71)	N(x; μ = 817,06, σ^2 = 42051,15)

Table 2.4. Average or variance values for each class, each input

We apply Naïve Bayes with Gaussian distribution as a classification algorithm. We allow one-level tolerance for mispredictions, and obtain 91.3% accuracy level. Misses and hits for various people count – weight pairs are shown below (see Figure 2.7). Time is ignored here to keep the graph simplified at 2-dimentional, not 3-dimentional.

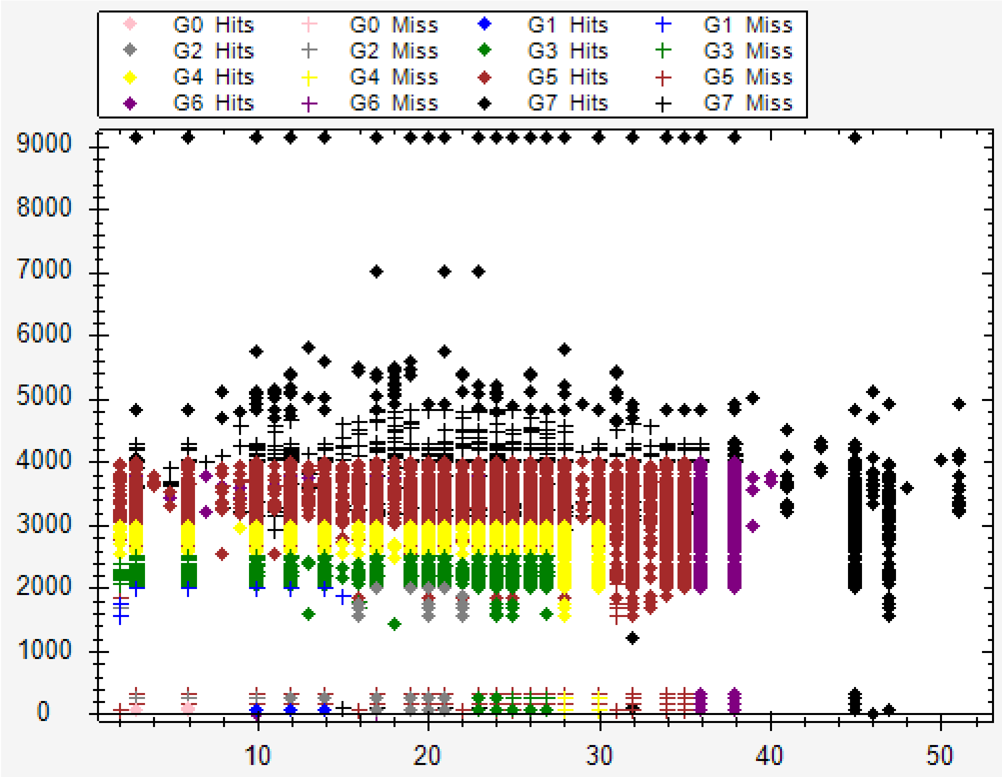

Figure 2.7. Weight in terms of grams (Y-Axis) and people count (X-Axis) are split into their regarding classes, where hits and misses are represented after a prediction process with Naïve Bayes.

Accuracy	Total Miss	Total Hit
0,913	866	9135

Table 2.5. Results of tolerated Naïve Bayesian process

<u>To improve the result</u>, **Categorical Naïve Bayesian** method can be preferred.

On the contrary of weight approach for the first Naïve Bayes method, here we will not evaluate the instant weight of the bin, but only the acceleration of the weight. In other words, weight increment/decrement by unit time (minute). For example; sensor produced a weight value 3000gr, and the next one produced as 3200gr and this value is retrieved 2 minutes later than preceding. Then, difference between two values divided into total minutes. (3200 – 3000)/2 =100.

So here, we briefly assume that, if waste is continuously and rapidly increases, it might mean that a vast amount of production can be processed in restaurant due to order density. We will try to find out if there is a pattern with this input.

As previous work, we have implemented Naïve Bayes method with quantative inputs and also quantative classes. Here we will also examine this method with categorical inputs and classes. By doing this we aim to improve the results of predictions.

To clarify what we mean about categorical; we defined some categories for the same input intervals, such as time reduced to definitions like “MORNING”, “AFTERNOON” representing the specific time interval of day.

Here is the full definition of other input category reductions:

	People Count	**Category**	**Weight Acc.(gr)**	**Category**	**Time**	**Category**	**Service Level**	**Category**
0	<10	EMPTY	<10	LITTLE	<420	NIGHT	0	SLEEP
1	10-20	COOL	10-50	NORMAL	420-720	MORNING	1	WAKEUP
2	20-30	NORMAL	50-100	ACTIVE	720-810	NOON	2	HMM!
3	30-40	BUSY	100-250	WASTEFUL	810-1000	AFTERNOON	3	WORK
4	40+	OVERCROWD	250+	OVERWASTE	1000-1200	EVENING	4	WORK HARDER
5					1200+	POSTEVENING	5	WATCH-OUT
6							6	NO-BREAK
7							7	NO EYEBLINK

Table 2.6. Input category reductions from their integer values

These intervals will keep the things easier and trainable. Categories can also be represented as integer values corresponding to their string values. Such as, (EMPTY = 0, WASTEFUL = 3) in terms of their row order number. As the same with preceding method, service-level prediction error rate with one service level can be ignored, and counted as successful. This ignoring should be applied, because our output service level data is not fully accurate. For the same input, it should output +/-1 service level. Such as 1-0-3 -> 2; 1-0-3 -> 3. This actually happens due to human decisions.

Accuracy	Total Miss	Total Hit
0,99	93	9907

Table 2.7. Results of tolerated *Categorical* Naïve Bayesian process

Not enough to improve? Then **Decision Tree** method can improve the results with categorical, discrete data sets.

Decision tree learning contains decision tree as a predictive model, which uses observations represented as branch, which is our inputs like people count, and concludes to target values represented as leaf. Decision tree can branch with “yes” or “no” questions for discrete inputs whilst “greater than”, “less than” for continuous inputs. If we had used the first numeric dataset, it would have been assumed as continuous. But, for the first previous dataset, categorical one, we can assume it discrete.

Decision tree can work well for discrete categorical dataset. So here, we are not going to tolerate +/-1 level for this learning method like we do for Naïve Bayesian. ID3 (**Iterative Dichotomiser 3**) algorithm is the core one for decision tree. It’s a predecessor of C4.5 algorithm, and briefly used in machine learning or natural language processing works. This algorithm adopts a top to down and greedy search through set of possible branches without backtracking. ID3 basically uses **Entropy** and **Information Gain** to build a decision tree. Entropy is used by ID3 to find out the homogeneity of sample and if it is fully-homogeneous, then entropy is 0 and if equally split into two, the entropy is 1. [26]

To define what **entropy** is, here is an example of basic and reduced dataset for our case. Entropy using the frequency table of one attribute:

Set Service Level to Max?	
YES	9
NO	5

Table 2.8. Should service level be max? 9 inputs says yes, 5 inputs says no.

$$E(S) = \sum_{i=1}^{c} -p_i log_2 p_i \tag{2.6}$$

$$Entropy(SetMax) = Entropy(5,9) = Entropy(0.36,0.64) = -(0.36log_2 0.36) - (0.64log_2 0.64) = 0.94 \tag{2.7}$$

Entropy by using two attributes' frequency table:

$$E(T,X) = \sum_{c \in X} P(c)E(c) \tag{2.8}$$

		Level Max?		
		Yes	No	
People Count	BUSY	3	2	5
	OVERCROWDED	4	0	4
	NORMAL	2	3	5
				14

Table 2.9. Should service level be max by people count.

$$E(Set\ Level\ Max, People\ Cnt) = P(Busy) * E(3,2) + P(Overcrowded) * E(4,0) + P(Rainy) * E(2,3) = \left(\frac{5}{14}\right) * 0.971 + \left(\frac{4}{14}\right) * 0 + \left(\frac{5}{14}\right) * 0.971 = 0.69 \tag{2.9}$$

To define what **Information Gain** is:

The information gain depends on the decrement of entropy after a dataset is split on an attribute. Decision tree is based on finding attribute which has the highest information gain and makes you construct tree. [27]

Gain(T,X) = Entropy(T) – Entropy(T,X) (2.10)

As conclusion, when we use this learning method, we got good results, even we do not tolerate the model in terms of level.

Accuracy	Total Miss	Total Hit
0,986	134	9866

Table 2.10. Results of **non-tolerated** *Decision Tree Learning* process

Here is the semi-expanded tree visual where we can understand the results clearly.

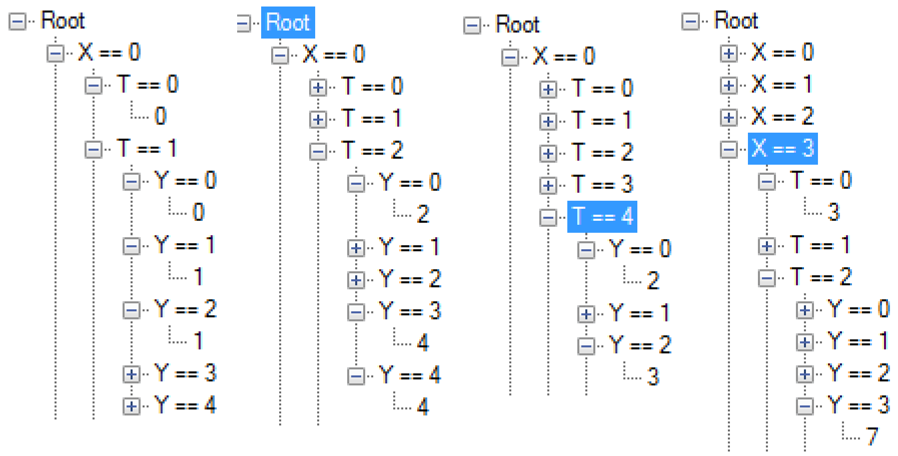

Figure 2.8. Semi-expanded tree visual of decision tree learning result

c-) <u>Smart Staff Allocator:</u>

This is another proficiency of our gateway. This module is another plug-in which can allocate the staff and give them orders to prepare. Especially within busy times, these staffs have no time to blink their eyes. They have an order queue screen, and they assign the orders to themselves or assigned by a manager. However, these assignments happen in a blink of an eye, and decided by human. So, allocation would not be optimal. In addition to this, it would be allocation waste. Sometime, these assignments create unfairness due to heterogeneous distribution, that is, some staff have five orders to prepare where another one have only two.

So, in terms of being fair and optimal, orders will be allocated to staffs with **Genetic Algorithm** which is an optimization method. Genetic algorithm can be assumed how human DNA matches, crosses over and create another gene, and with this gene, a child is born. Good DNA chains mixed up with other DNA chains where some of them bad. The better DNA included, the better human-being is born.

Here in our scenario, order collection screen is connected with the gateway, and gateway collects any orders from the cashier. Order collection screen has also an RFID reader or integrated with fingerprint device which knows who is working on that shift. So that, our screen knows the capabilities of the staffs. Staffs can be classified and scored by managers within regular intervals such as, a specific staff is good at preparing product A, but not product B or if a staff is newbie, he/she can prepare easily product C, but not fast and good at the others. Besides this classification by managers, gateway can also find out how he/she is good at preparing that product from a time on. Because, staffs mark as complete the order when finished. So gateway compare the performance with the others, and if it is a slow preparation, staff will be low-scored for that product. These scores can be evaluated and manipulated by managers in anytime. If a staff has a low-score for product, then that products preparation by that staff incurs high-cost.

Here is a test case:

We have five types of products. Twenty of orders are queued and will be distributed to five of staffs that are ready to welcome the orders. Here is the capabilities:

Staff Number	**Capability**
1	Normal for all product types
2	Bad at Product #4 (2x slower), normal for others
3	Very bad at Product #1 and #2 (3x slower), good at the others (2x faster)
4	Good at Product #0 and #1 (2x faster), fairly bad at others (0.25x slower)
5	Excels at Product #0 (4x faster), very bad at others (3x slower)

Table 2.11. Staff proficiencies at products

In addition to these costs, a delivery type also affects and have a penalty on it due to their importance and multiplies the cost. Because an “In Restaurant” order is very important and have to be produced immediately. So if a low-profile staff tries to produce it, its penalty multiplies it with a higher number than the others, and cost results higher than the experienced staff.

Delivery Type	**Penalty (Applied on total cost)**
0 (In Restaurant)	3x
1 (Home Delivery)	2x
2 (Proactive Production)	1x

Table 2.12. Delivery penalties

Product production difficulty is the same with product number. For example, Product 1 is the easiest to produce with cost one, Product 5 is the hardest to produce with cost five.

All staff should be distributed with same amount of products if possible. Otherwise, unfairness occurs.

Therefore, a genetic algorithm with elite selection should work for us. Elite selection is the gene selection which is the last and best solution within a iteration of genetic algorithm. That gene is inherited to successors while cross-over or mutation implementations. Because a good gene can create better gene with other genes.

Iteration number shouldn't be so low or so high. If it is low, then your final result might not be the best one, if it is high, your result might be the best solution but it takes too much time to find it. So in our case, 200 iterations gave us good results and our population size is limited with 40 and uses greedy crossover.

Order No	Product Type	Delivery Type
0	3	2
1	1	1
2	3	0
3	1	0
4	4	0
5	0	1
6	0	2
7	4	2
8	3	2
9	3	2
10	1	2
11	2	1
12	3	2
13	3	0
14	1	1
15	3	0
16	3	2
17	0	2
18	4	0
19	2	1

Table 2.13. Orders and its details

In the Table 2.13. the chromosome for order no and staff match that we need to find.

Staff 1	Staff 2	Staff 3	Staff 4	Staff 5	Staff 1	Staff 2	Staff 3	Staff 4	Staff 5
?	?	?	?	?	?	?	?	?	?
Staff 1	Staff 2	Staff 3	Staff 4	Staff 5	Staff 1	Staff 2	Staff 3	Staff 4	Staff 5
?	?	?	?	?	?	?	?	?	?

Table 2.14. Question marks will be filled after genetic algorithm appliance respectively on this order. So How should the **(?)** be filled respectively?

As in the table 2.14. all the question marks will be replaced respectively with order numbers with the best chromosome.

Just after execution of algorithm, here is the best **chromosome** which will be also inserted into (?) represented in the previous table:

4 – 8 – 15 – 12 – 10 – 11 – 19 – 13 – 1 – 17 – 0 – 16 – 2 – 3 – 6 – 7 – 9 – 18 – 14 - 5

Thus, for example staff 1 will take care of 4th order and the others where staff 5 will take care of 10th order and so on so forth…

Here is the total cost **77,5**.

Now we now the best possible cost is this value, so we can distribute the orders to our staffs easily. When the staffs are deployed with these orders, and they complete it, they mark it as completed on the screen. So that, we can save how quickly he completed that product and as a result, we can understand if that staff excels on that product, or need more training, or going worse by the days gone. This will impact their overall score for each product.

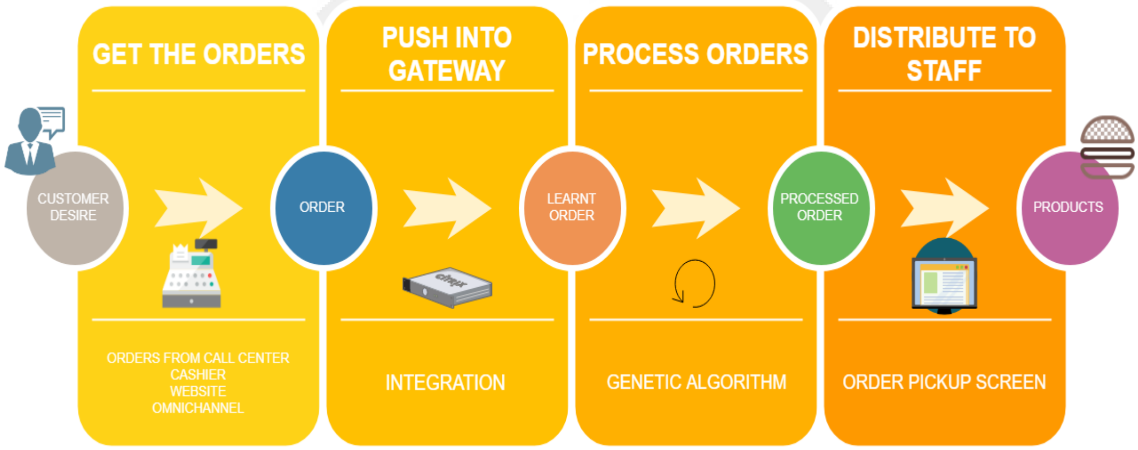

Figure 2.9. Full overview of smart staff allocation workflow from customer to product.

We can overview the full workflow like in the Figure 2.9. All lifecycle can be seen at a glance.

2.3.2.4.Social network analysis for QSR

Until now, we had such a great movie, and major things have been mentioned, but not at all. Here, let us talk about an external resource of information, social media which is the royal of the information resource. For future work, any social media data can be streamed into the learning algorithm as an input. Basically we can explain how it may work and produce these data.

The idea of ‘Social Network’ has been built when many popular social media environments hyped and grew up rapidly. “Facebook” [28], “Twitter” [29], “Instagram” [30], “Foursquare” [31] and similar media make everyone build their virtual mirror of their existence in the web [32]. Each of them builds specific part of us. “Facebook” is mostly related about “what we are, what we like”, “Twitter” is like “What we would like to say at that moment”, “Instagram” is more about “how we look, how we do what we do” and “Foursquare” is “Where we are and how much we are interested in there”. When we combine each of them, they create the “Voltron”. Moreover, most of them also answers the question of “where” nowadays. All of these create a massive, nearly infinite network which can be illustrated with graph. Graph has nodes which are the objects like people, a school, a road, a district, etc.; has edges which are the connections between nodes; has weights which fortify or weaken the connectivities of nodes, a property of the edge like friendship, marriage, owning etc.; and has labels which identifies either node or edge like name of someone or school or a book.

We can make use many of answers for Quick Service Restaurants. All restaurants have their own addresses and also their precise points with latitudes and longitudes, moreover have their names or split names which is also called “tags”. This information are the deterministic parts in social network. Thanks to users, our restaurant is also connected to other objects in network with many kinds of edges. Our restaurant might not be so important for users, it is just a quick service restaurant, just go, eat and leave. So, they might not mention about it. But, surroundings or neighborhoods can contain many key places. For example, a walkway area near coast, a stadium, a concert hall, a picnic area, a running contest, a show etc. These places have very high attraction to be shown in social media by people. These places have their own importance by evaluating in-degree, or out-degree values of edges. In-degree is defined as “centrality focuses on a specific individual node; centrality of all other individuals is based on their relation to that focused node of the "in-degree" individual”. Moreover, out-degree is defined as “a measure of a centrality which still focuses on a single individual node as well, but the degree score is related with the outgoing interactions of that individual; to sum up, the score of out-degree centrality is about how many times the focused individual interacts with other nodes” [33].

Here neighborhood nodes have to be scored, because not all events have the same effect to that area. Besides, we do not want to bother the gateway with low scored, unimportant events. So, questions like “how a score can be given to nodes”, “which node is the most central in a network”, should be answered by some Social Network Analysis algorithms.

Social Media Engine

Social media engine is located at the cloud side, opposed to the other engines. This is because employing gateway to analyze the social data can kill its resources, and it would be too time consuming to handle a very crowded and raw data. So, this time cloud will handle it. Bad news for cloud, but enough for sleeping so far.

This engine dedicates itself to track any social occurrences and applying some algorithms and process the social data and then stream it to gateway. Centrality and scoring will be handled by social network analysis metrics like degree centrality, closeness centrality, betweenness centrality and eigenvalue centrality.

i- **Degree Centrality:** Simplest and the most convenient way of finding the most central node which is the highest scored at the same time. It is the node which has the most edges leaves from or penetrates into. As it is a social network, it is a directed graph, number of in-degree and out-degree edges will define its centrality.

ii- **Closeness Centrality:** Degree centrality is more about quantity. But quantity is not everything. Closeness takes account into being close to other nodes by creating the shortest path to other nodes. A node, which is the closest to the others, is the most central one.

iii- **Betweenness Centrality:** It is more structured algorithm than closeness. Because, being close might not be an answer if the others do not use that node as a path. In betweenness, the most central node is used by other nodes to reach each other as a bridge. So, this node is the most called on node by other nodes while traversing the network.

iv- **Eigenvector Centrality:** This algorithm is one more step ahead. It defines the node’s influence in a network map. Any nodes that resides in this network are scored regarding to their connection number. If it has many connections to itself, than it will be high-scored, otherwise low-scored.

Katz centrality and Google's PageRank are derived from eigenvector centrality. [34]

Especially the last one is a suitable metric to measure the influence of a node in a network.

Our social engine can use social media APIs and third party social media engines in order to extract the detailed network graph with all the weights and directions of the edges. We also check for the traffic congestion on the road passing in front of the restaurant, which may give some idea of the busyness around restaurant. A simplified version of example social network graph is given below (see Figure 2.9). All the nearby places and also a "traffic congestion" node is connected to the restaurant node via a "road" node.

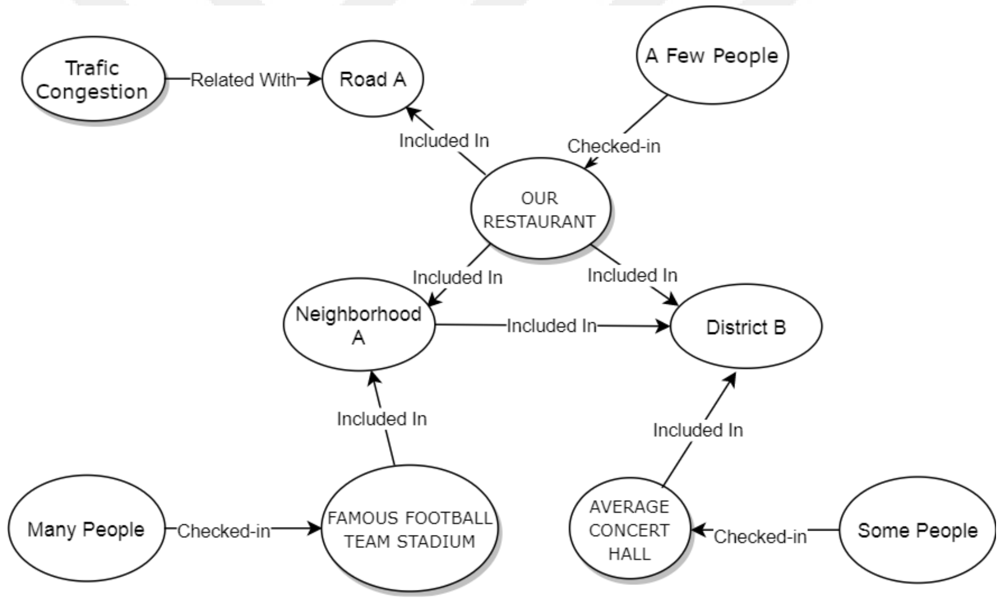

Figure 2.10. Social network graph of a restaurant with its neighboring nodes.

Then we define some grades for every node of the graph between 0 and 10. These grades are determined by the eigenvector centrality for the nearby places. For the "traffic congestion" node, grade is determined by the level of congestion, i.e. 0 means empty road and 10 means a pinned traffic which is caused by a huge accident or very high demand to the road. The grades assigned to nearby places would then converted to a single value (again between 0 and 10) which determines the social impact level, i.e. how much people are attracted to our surroundings.

To sum up, the social media engine obtains the social network graph, evaluates the nodes in the graph, determines the social impact level between 0 and 10, and pushes this information to related gateway. This value is also used as a sensor value named “social impact sensor” which is fed from cloud.

3. RESULTS AND DISCUSSION

We have run many algorithms to cluster or learn the data we owned by sensors. Some of them worked OK where other ones worked like charm. Here we will compare the results we run, and will see which learning method work for which circumstances. This will make us create some hybrid structures, which adapts to any conditions. Besides to them, we will see how the parameters can be efficiently set and used for each algorithm to carry it to the top.

3.1. K-Means Clustering – (Weight-Meter)

In our case, we have created five clusters for our dataset. Our dataset most probably have to outliers for any kind of sensor where one of them is an uppermost and the other one lowermost. Because our sensor can create disrupted values which converges to zero (default) or diverges to max due to some physical problems So that, we should have at least 2 clusters to gather them separately from normal values.

So what about three clusters? Here in the Figure 3.1. three cluster for weight-meter with same dataset.

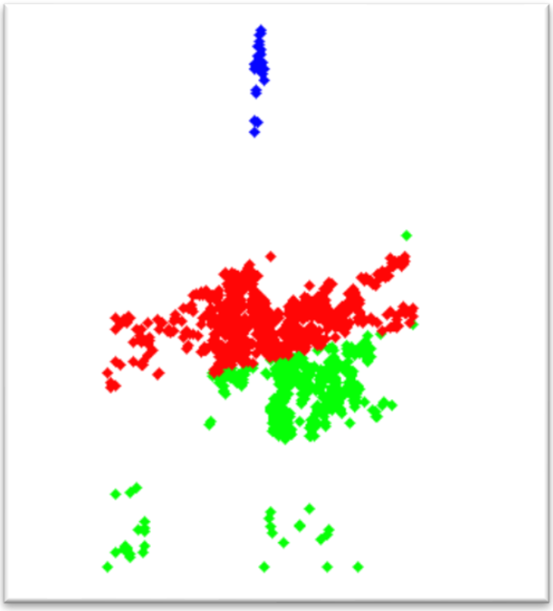

Figure 3.1. Weight-Meter values clustered into 3 classes.

Figure 3.1. shows results with lowermost cluster not split from the lower bound of normal values. So that, regardless of how we use Bhattacharya distance, which may resulted as valid, our outliers would be hidden from us. So here, we should keep the cluster number high as possible, but also not too much not to increase the overhead of computing process.

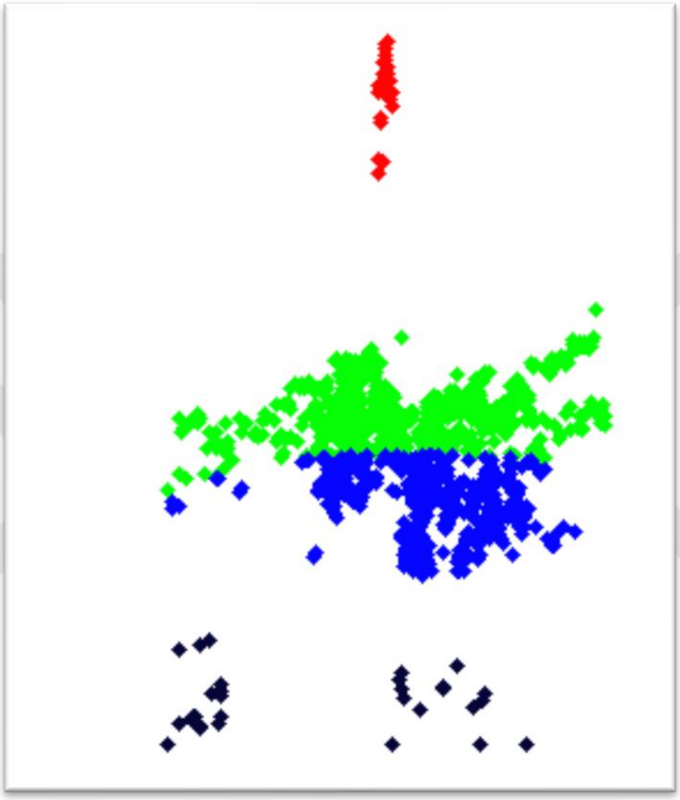

Figure 3.2. Weight-Meter values clustered into 4 classes.

Figure 3.2. shows clustering of weight-meter into 4 classes and it is plausible, but still risky if outlier values close to our normal clusters. In our case, that would be no problem. We have used 5 classes to be at the safe-side. Using 5 classes made us to show what are the lower and upper boundaries of our normal clusters.

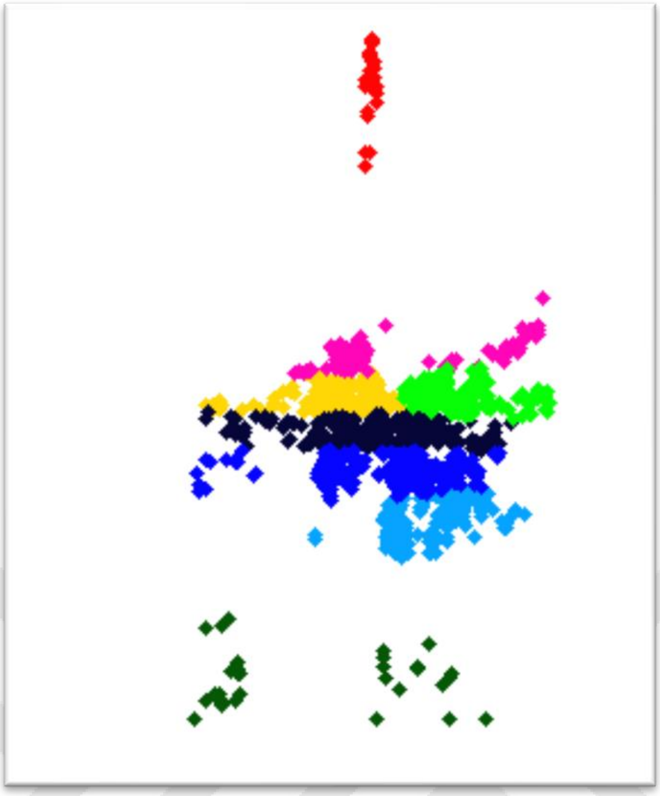

Figure 3.3. Weight-Meter values clustered into 8 classes.

Figure 3.3. is guaranteed! The more clusters we have, the more successfully we have separate outliers. Still risky if we keep the number of classes high in terms of performance.

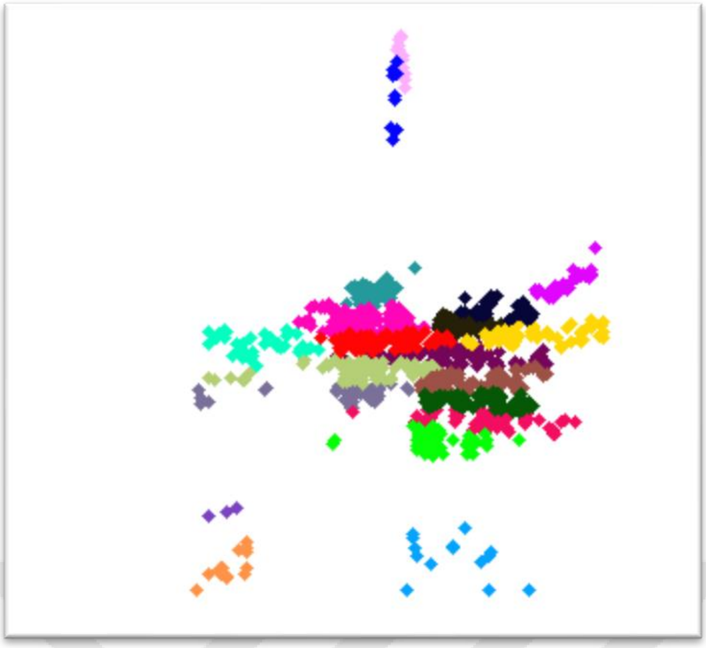

Figure 3.4. Weight-Meter values clustered into 20 classes.

We have given the number of classes as high as possible in Figure 3.4. Our outlier sets are also split into sub cluster sets. Acceptable, but better be not preferred. Because, computational time and resource would increase due to class quantity, and also time will increase to detect the Bhattacharya Distance between each other.

Nevertheless, it should be examined and then decided for each sensors. Like weight-meter which has continuous and incremental/decremental, in other words, distributed value dataset, work with little number of clusters. On the other hand, like temperature or humidity sensors which is linear and has stabile and non-changed values, (i.e. under normal circumstances, a temperature in restaurant is between 24 – 27 not less, not more) may need lots of clusters.

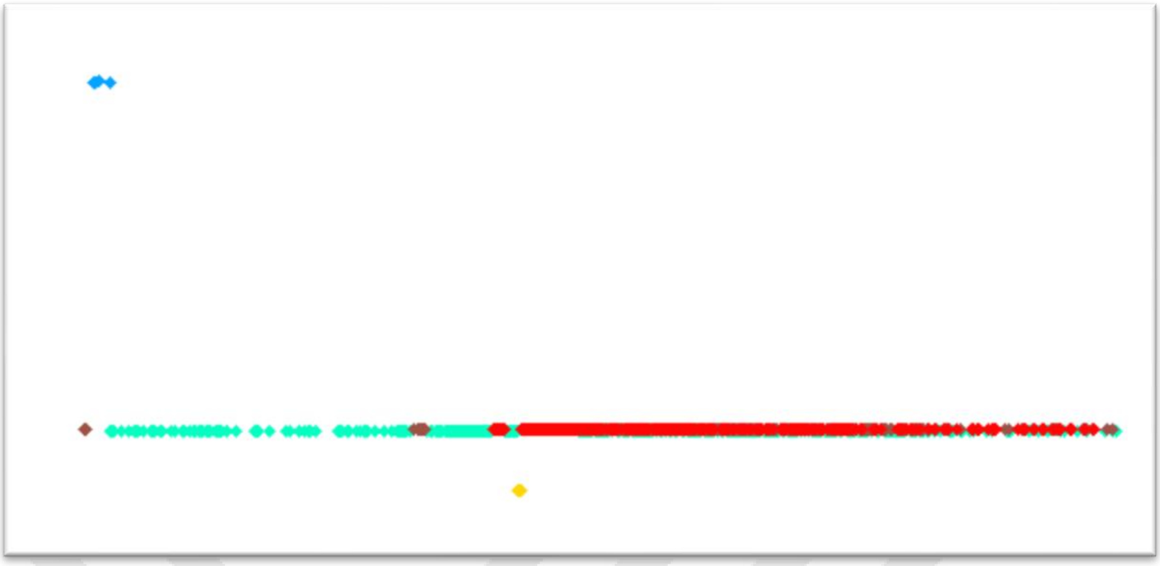

Figure 3.5. Temperature values(Y-Axis) between [24-27 C^o] by time of day (X-Axis) clustered into 10 classes.

As it can be seen in Figure 3.5., this value dataset is tend to linear and horizontal distribution and prone to clustering into horizontal clusters. Some outliers exist which is the lowest degrees (-32 C^o) and disrupted values like (342 C^o). By requesting many clusters like 10, will make us have some vertical clusters so that we can detect outliers like the yellow at lowermost and light blue at uppermost.

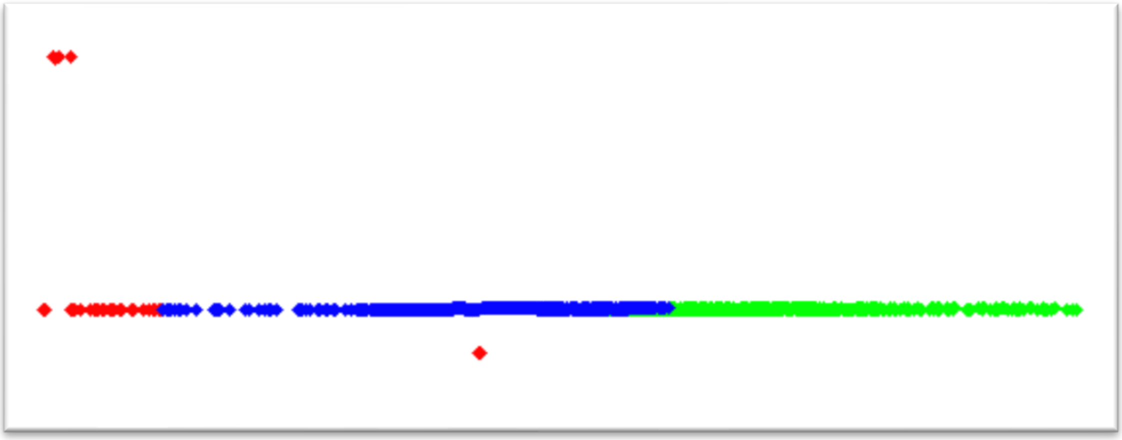

Figure 3.6. Temperature values(Y-Axis) between [24-27 C] by time of day (X-Axis) clustered into 3 classes.

Unfortunately for Figure 3.6., low amount of cluster caused our outliers joined to our normal distribution. Therefore, we have missed them. We can also double-check our model by checking their mean values. For example, here in this example given in Figure 3.6., the mean is approximately 24 C^{o} . But our red cluster have a mean like 40 C^{o} So here, we can understand that our model did not fit how we want. So we break the learning method, and recalculate with new higher number of clusters.

Another way to understand is to get the highest value/value set of a cluster, and measure the Bhattacharya distance to its owning cluster. If it has a greater Bhattacharya distance than we can accept, then cluster may not be separated well. So Bhattacharya distance is our validation and guarantee method to apply.

3.2. Naïve Bayesian Vs. Decision Tree (Service-Level)

In our machine learning applications, we had better results for categorical inputs. So, where possible, categorizing the inputs from their numerical values might increase the results of the model. But, nevertheless, no matter if it is categorized, it works well with vice versa. All of them have more than %90 success rates.

Method	Success Rate
Naïve Bayes – Tolerated – Numeric	%91
Naïve Bayes – Tolerated – Categorized	%99
Decision Tree- Tolerated - Categorized	%99
Decision Tree - Non-Tolerated – Categorized	%98
Decision Tree – Tolerated – Numeric (C4.5)	%94

Table 3.1. All tests are compared with success rates.

For decision tree with numerical input, a well-known decision tree algorithm C4.5 algorithm used. C4.5 is an extended version of ID3 which is mentioned above. Because ID3 Algorithm can only process discrete datasets.

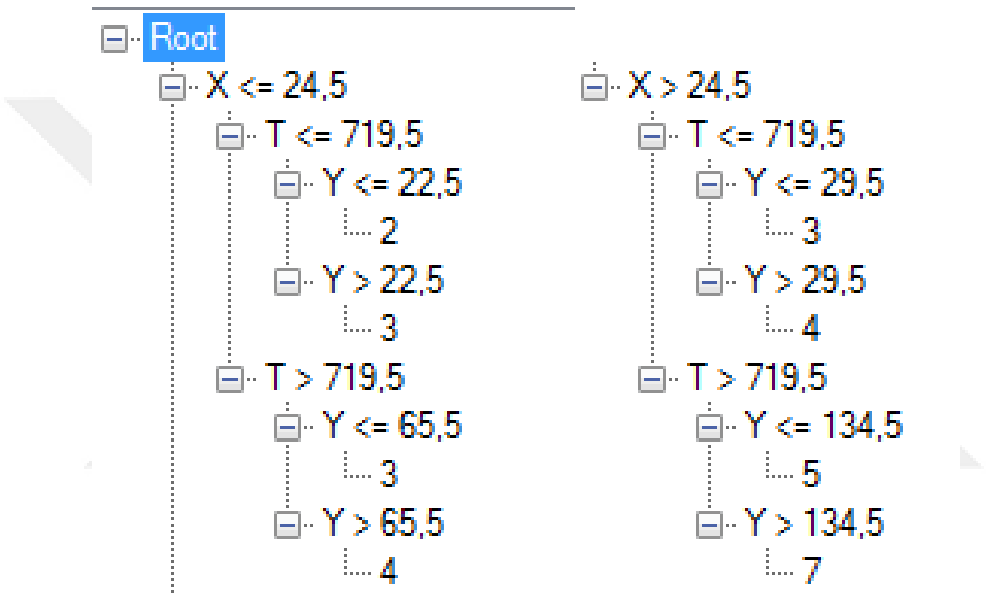

Figure 3.7. Decision Tree with C 4.5 algorithm.

With C4.5 Decision Tree Learning algorithm, we have been supported with ability to process continuous data set like in Figure 3.7. As in that figure, X (People Count) is divided into two subtrees. It is greater than 24 or less than or equal 24.

Moreover, T (Time) is also have two branches for each X branches. Both of them are the same for each branch. Similar for this, Y (Weight Acceleration) have branched into two for each branches. But it varies from branch to branch. Here Service Level 6 seem to be missing. Level 6 is more like shifting to up-level or down-level regarding to their inputs. This may cause an error, but if we tolerate this shifting to another level like we accepted at the very beginning, that would be no problem. Level 6 may be a transit level with a very narrow limit.

3.3. Genetic Algorithm (Staff Allocator)

For the same dataset and conditions which is given above in the thesis, we will change the selection methods and iteration numbers and see how it results.

First scenario is to use **roulette selection** method which is differs from elite selection method. Roulette selection is just random picks for each iteration, and never evaluates and inherits the previous iteration. It just saves the latest best result in the memory, and if any better result comes, disposes it and re-writes on it with the new better result. We will use same iteration number (200) and population size (40).

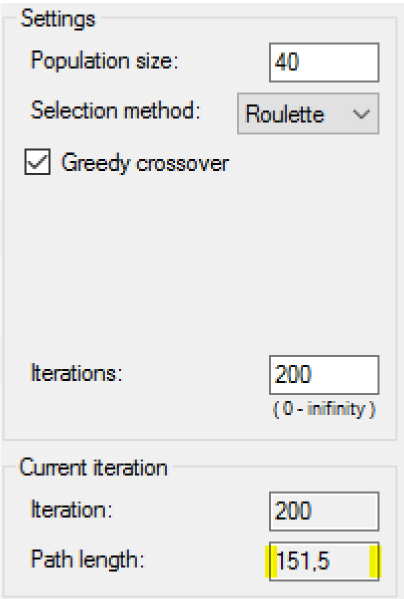

Figure 3.8. The result is 151,5 which is far away from our best solution

Another execution with same conditions and different result.

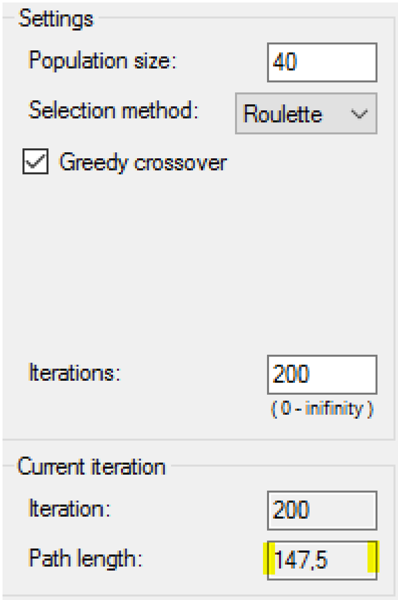

Figure 3.9. The result is 147,5 which is far away from our best solution and different from previous one.

As we can see that, roulette selection method is just a matter of luck, not much more. Every time we try, it results different. We can assume this method as a brute force method, as it tries all possibilities without using any optimization or heuristic methods.

Another test is, back to the elite selection method with 10 iterations with same population size.

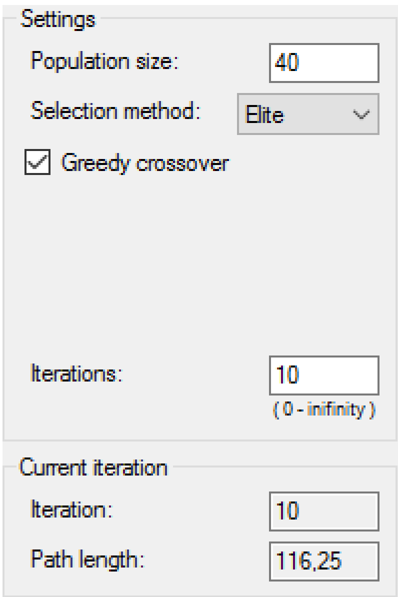

Figure 3.10. 10 iterated elite selection method resulted 116,25.

As it is barely seen, very less iteration number interrupts the model prematurely and blocks us to find better results. So, what about keeping iteration number at 10000;

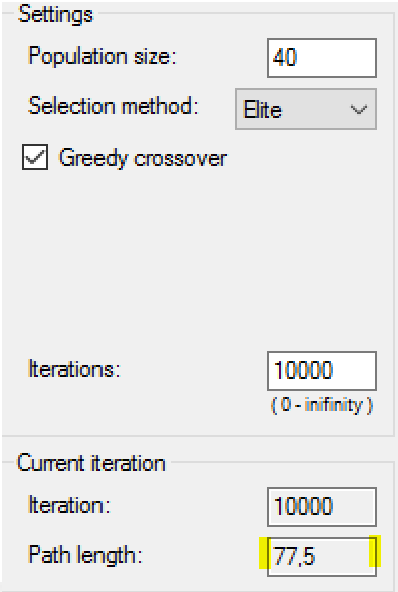

Figure 3.11. 10k iteration with same result

It resulted same with 200 iterations. But it took 15 – 20 seconds to calculate which is a little bit far from real-time calculation.

As a conclusion we get what we want with 200 iterations within a short time by using elite selection method.

4. CONCLUSIONS

As a conclusion, we installed a good IoT architecture in a test quick service restaurant, and armed with many machine learning algorithms, and engine layers. A raw data from the sensor-side to cloud-side has shown through the whole pipeline where it is processed, transformed, learnt. For a limited test domain within a sampled time and in a selected restaurant, we provided an intelligence to replace key staff. Mentioned key staff found several times to take care other things to do while our intelligence was doing many things on behalf of him. At the same time, quick actions made other staff well organized. Production capability and quality reasonably increased.

We also tried to optimize the waste under the name of this thesis topic, and for the waste-bin, a good amount of waste has been restored thanks to well-pointed predictions. Still have many works to converge waste amount to zero. But for the beginning any waste avoidance is something.

Moreover, many processes are now under record and open to tracking and monitoring. In terms of reporting, headquarters of the company have any information at once with a single look. Also, restaurant or region managers have ability to track their place and alerted by key and important scenarios.

4.1. Future Work

For the future work, besides to applied methods, other machine learning methods like random forests, boosted trees, and some deep learning methods by neural networks is planning to be used. Moreover, genetic algorithm will be enhanced with another cost inputs like awaiting time of product and also supported with dynamic genetic algorithm, and another resource planning algorithms. This learning method will carry the architecture to further points.

This thesis is a good start for the restaurant chains to make it smarter. Not only for restaurants, but for other fields can apply this architecture.

References

[1] Y. Yang, L. Wu, G. Yin and L. L. a. H. Zhao, "A Survey on Security and Privacy Issues in Internet-of- Things," *IEEE Internet of Things Journal*, pp. 1250 - 1258, 2017.

[2] "Internet of Things: Science Ficton or Business Fact?," Harvard Business Review, 2014.

[3] O. Vermesan and P. Friess, Internet of Things: Converging Technologies for Smart Environments and Integrated Ecosystems, Denmark: River Publishers, 2013.

[4] Gartner, "Gartner Says 6.4 Billion Connected "Things" Will Be in Use in 2016, Up 30 Percent From 2015," Gartner, 2015.

[5] M. Joe, "With Internet Of Things And Big Data, 92% Of Everything We Do Will Be In The Cloud," Forbes, 13 November 2016. [Online]. Available: Forbes.com. [Accessed 1 November 2017].

[6] "Statista," Number of quick service restaurant (QSR) franchise establishments in the United States, 2017. [Online]. Available: Statista.com. [Accessed 22 November 2017].

[7] "Franchisehelp," Fast Food Industry Analysis 2018 - Cost & Trends, 2018. [Online]. Available: Franchisehelp.com. [Accessed 5 January 2018].

[8] "Unep," Food Waste Facts.- United Nations Environmental Programme, 2013. [Online]. Available: Unep.org. [Accessed 8 November 2017].

[9] D. Gunders, "Wasted: How America is losing Up to 40 Percent of Its Food from Farm to Fork to Landfill," 2012.

[10] B. Plumer, "The Washington Post," How the U.S. manages to waste $165 billion in food each year, 2012. [Online]. Available: washingtonpost.com. [Accessed 30 November 2017].

[11] M. Abdelshkour, "IoT, from Cloud to Fog Computing," Cisco, 25 March 2015. [Online]. Available: blog.cisco.com. [Accessed 4 November 2017].

[12] B. Ray, "The ZigBee Vs WiFi Battle For M2M Communication," 2 November 2015. [Online]. Available: link-labs.com. [Accessed 25 March 2018].

[13] Litepoint, "IEEE 802.11ac: What Does it Mean for Test?," October 2013. [Online]. [Accessed 25 March 2018].

[14] K. S. P. a. C. A. Jayalakshmi, "Waste to Wealth- A Novel Approach for Food Waste Management," *IEEE International Conference on Electrical, Instrumentation and Communication Engineering (ICEICE),* 2017.

[15] Z. Wen, S. Hu, D. D. Clercq, M. B. Beck, H. Zhang, H. Zhang, F. Fei and J. Liu, "Design, implementation, and evaluation of an Internet of Things (IoT) network system for restaurant food waste management," *Waste Management Science Direct,* 2017.

[16] A. Greenberg, J. Hamilton, D. A. Maltz and a. P. Patel, "The cost of a cloud: Research problems in data center networks,," *ACM SIGCOMM Comput. Commun. Rev.,* vol. 39, no. 1, pp. 68-73, 2008.

[17] E. C. e. al., "MAUI: Making smartphones last longer with code offload,," *Proc. 8th Int. Conf. Mobile Syst. Appl. Services,* pp. 49-62, 2010.

[18] M. Satyanarayanan, P. Bahl, R. Caceres and a. N. Davies, "The case for VM-based cloudlets in mobile computing," *IEEE Pervasive Computing,* vol. 8, no. 4, pp. 14-23, 2009.

[19] F. Bonomi, R. Milito, J. Zhu and a. S. Addepalli, "Fog computing and its role in the Internet of things," *in Proc. 1st Edition MCC Workshop Mobile Cloud Comput., Helsinki, Finland,* pp. 13-16, 2012.

[20] W. Shi, J. Cao, Q. Zhang, Y. Li and a. L. Xu, "Edge Computing: Vision and Challenges," *IEEE Internet of Things Journal,* pp. 637-646, 2016.

[21] "Regex Reference," 2010. [Online]. Available: Regexstorm.net. [Accessed 10 January 2018].

[22] A. M. Zoubir and J. F. Böhme, "Bootstrap multiple tests applied to sensor location," *IEEE Transactions on Signal Processing,* vol. 43, pp. 1386-1396, 1995.

[23] G. B. Coleman and H. C. Andrews, "Image Segmentation by Clustering," *Proc IEEE,* vol. 67, no. 5, pp. 773-785, 1979.

[24] H. Zhang, "The optimality of Naive Bayes," *Proc. of the Int. Florida Artificial Intelligence Research Society (FLAIRS) Conference,* 2004.

[25] D. J. Hand and K. Yu, "Idiot's Bayes — not so stupid after all?," *International Statistical Review,* p. 385–399, 2001.

[26] J. R. Quinlan, " Induction of Decision Trees," vol. 1, no. 1, p. 81–106, 1986.

[27] saedsayad, "Decision Tree - Classification," [Online]. Available: saedsayad.com. [Accessed 1 April 2018].

[28] "Facebook," [Online]. Available: facebook.com. [Accessed 1 April 2018].

[29] "Twitter," [Online]. Available: twitter.com. [Accessed 2018 April 1].

[30] "Instagram," [Online]. Available: instagram.com. [Accessed 1 April 2018].

[31] "Foursquare," [Online]. Available: foursquare.com. [Accessed 1 April 2018].

[32] J. Scott, "Social Network Analysis," Sage, 2017.

[33] M. d. e. a. Laat, "Investigating patterns of interaction in networked learning and computer-supported collaborative learning: A role for Social Network Analysis," *International Journal of Computer-Supported Collaborative Learning*, vol. 2, no. 1, pp. 87-103, 2007.

[34] D. Austin, "How Google Finds Your Needle in the Web's Haystack," [Online]. Available: ams.org. [Accessed 21 February 2018].

AUTOBIOGRAPHY

Name Surname: Kerem AYTAÇ

Place and Date of Birth: İstanbul / Oct 11, 1990

Foreign Language: English

E-Mail : keremaytac@gmail.com

Educational Status

Grade	Department/Programme	University/High School	Graduation Year
High School	Physical Sciences	Haydarpasa Anadolu Lisesi	2008
University	Mathematics	Yıldız Teknik University	2012

Work Experience

Year	Company/Corporation	Position
2013-…	Ata Technology Platforms – R&D Department	Senior Software Developer

Academic Publications

1. K.Aytaç and Ö. Korçak, “IoT Edge Computing in Quick Service Restaurants” in **The International Workshop on Edge and Fog Computing for Intelligent IoT Applications (EFC-IoT)**, Shangai, China, May. 2018. [Published]
2. K.Aytaç and Ö. Korçak, “IoT Brain for Quick Service Restaurants”, **IEEE Pervasive Computing Magazine**, April. 2018 [Submitted]