Moka Digital Twin

A Small-Scale Study for Process Engineering Applications

Siddanth Nayak Bairampalli

Master of Science in Embedded Computing Systems
Submission date: July 2020
Supervisor: Pierluigi Salvo Rossi
Co-supervisor: Nicola Paltrinieri

Norwegian University of Science and Technology
Department of Electronic Systems

Acknowledgement

I would like to extend my thanks to my supervisors Pierluigi Salvo Rossi and Nicola Paltrinieri who gave me an opportunity to work on this project. Their invaluable suggestions and continuous guidance was of paramount importance for the implementation of this project. I would also like to take this opportunity to thank the staff of the EMECS program for their support throughout the last two years. I would also like to thank all my friends back home and also the ones I made during my stay in Germany and Norway. The importance of the experience gained and the memories made during the last couple of years cannot be understated. Last but not the least, I would like to thank my family whose constant support and encouragement has enabled me to navigate through the challenges of studying in two countries.

Abstract

Digital twins have emerged as an important tool being used in the industry for monitoring equipment. Data generated from a physical device or a process provides an opportunity to apply machine learning algorithms for anomaly detection and prediction. The insights gained by this analysis can be very useful in monitoring the physical condition of the device. The main goal of this thesis is to implement a data driven condition monitoring system for a moka pot and detect anomalies in the coffee preparation process. A data acquisition system was set up to generate data from the brewing process. A comprehensive dataset was generated which included data from ideal and anomalous iterations. Both supervised and unsupervised machine learning algorithms were trained and tested on the dataset for detecting anomalies in the process. With an accuracy score of 88%, an anomaly detection system with a reasonable performance has been implemented, demonstrating the use of the generated dataset.

Contents

List of Figures

List of Tables

Abbreviations

CSV	Comma Separated Values
DAQ	Data Acquisition
DTW	Dynamic Time Warping
HC	Hierarchical Clustering
HH	High Heat
IH	Inconsistent Heat
IT	Internal Temperature
ET	External Temperature
NN	Nearest Neighbour
PC	Personal Computer
RBF	Radial Basis Function
SVM	Support Vector Machine
VI	Virtual Instrument

Chapter 1

Introduction

1.1 Motivation

Digital Twins are becoming increasingly important in many industries such as manufacturing, healthcare etc. They enable the use of data coupled with machine learning methods to provide solutions to engineering problems such as anomaly detection and condition monitoring. This study aims to demonstrate a data driven approach to detect anomalies in a process. The process chosen for this study is the preparation of coffee in a moka pot. The moka pot or simply known as the moka, is the most popular instrument used to brew coffee in Italy. Bialetti industries, the inventor and a leading manufacturer of the moka, claims to have manufactured more than 200 million moka pots [1]. This stands as a testament to the popularity of this coffee brewing method. The preparation of coffee in a moka machine is a complex thermodynamic process and the output greatly depends on multiple parameters such as heat supply, initial water content, initial coffee weight and the condition of the pot itself. A significant variation in any of these parameters have a considerable impact on the quality of coffee produced. The existence of such anomalies in the process and the convenience with which required data can be generated from the moka system, makes it an ideal candidate to carry out this study. Moreover, there have been very few studies on the moka. Experimental analysis and use of machine learning methods may lead to improvements in the process and may pave the way for some optimisations in the pot itself. However, there is no readily available dataset related to the operation and functioning of the moka. This thesis aims to provide a structured dataset which could be used to develop anomaly detection systems for the moka. Faults arising from the sensor system itself can be hard to deal with and these faults interfere with the performance of anomaly detection algorithms. This experimental setup can be used to produce such behaviour and appropriate detection schemes can be tested to detect such faults in addition to the existing process anomalies. Exploratory work on anomaly detection in the coffee preparation process could provide insights for applying similar detection methods to monitor larger and more complex industrial processes.

1.2 Thesis Scope

The goal of this thesis is to implement a digital twin of a moka pot. Specifically, it is to explore the possibility of applying machine learning methods and develop a data driven system to detect anomalies in the coffee preparation process. The unavailability of data required for building such a system was addressed by producing the dataset. To generate a useful dataset, it was needed to understand the physics behind the functioning of the moka pot and it was also necessary to identify the possible faults in coffee preparation process. A data acquisition system, presented in the specialization project with the same title as this thesis [2], was improved to accommodate more sensors. This work involved interfacing hardware from National Instruments and also developing the virtual instrument on Labview. Using this setup, a labelled time series dataset containing data representing ideal and every anomalous condition was generated. Some machine learning methods that could be used to detect anomalies in the brewing process were identified and applied on the generated dataset. Their performance of detecting anomalies have been summarised, demonstrating a proof of concept for a data driven condition monitoring system for the moka.

1.3 Thesis Structure

The thesis is structured as follows:

- Chapter 2 provides fundamental background information about the moka pot. It gives an overview of digital twins, mentions some use-cases of digital twins in the industry and also explains the methodology followed for implementing the digital twin of a moka.
- Chapter 3 explains the concepts related to time series data and presents theory for the analytical methods applied on the generated dataset.
- Chapter 4 describes the hardware setup and the Labview virtual instrument designed for the experiment. It also describes the procedure followed and the conditions under which the ideal and anomalous data was generated.
- Chapter 5 deals with the time series analysis performed on the data. It provides a detailed description of the dataset generated and also provides some visualizations to understand the data. It also explains each method tried on the dataset to detect anomalies.
- Chapter 6 describes the performance of each method applied and contains a discussion on the results obtained.

- Chapter 7 provides a conclusion for the work and possible future work that can be carried out building upon this thesis.

Chapter 2

The Moka Pot and Digital Twins

This gives some background on the equipment under focus - the moka pot. This chapter also includes sections defining digital twins, mentioning their applications and also providing the implementation methodology followed in this thesis.

2.1 Description of the Moka Pot

The moka is one of the most popular instruments used to brew coffee, especially in the country of Italy where the moka was invented. The most popular model of the moka pot is the "Moka Express" by Bialetti industries [1]. It is produced in the 1, 3, 6, 9 and 12 cup variants. For our experiments, the 12 cup version is used. The bigger size of the 12 cup version enables the easy application of sensors on the pot. The moka structurally consists of 2 chambers. The lower boiler chamber which holds the water before the brewing process and the top chamber which holds the coffee after the brewing process. A funnel equipped with the filter plate fits snugly into the boiling chamber and then the top chamber is screwed onto the boiler chamber. The length of the funnel is such that it does not touch the base. The boiler has a pressure release valve for safety purposes and the top chamber has a spout through which the coffee flows. The figure 2.1 illustrates the parts of the moka.

2.2 The Coffee Preparation Process

The process to prepare coffee in a moka contains the following steps:

1. Water needs to be filled into the bottom chamber such that that safety valve is not covered. In the model used for the experiments, there is a reference line provided in the bottom chamber up to which water must be filled.

2. Coffee grounds should be poured into the funnel and loosely packed and not tamped. Care

needs to be taken not to over fill the coffee and also the grounds should not be too fine, so that the filter plate is not clogged.

3. The top chamber needs to be fixed on to the lower chamber, creating a tight seal so that no steam escapes.
4. A consistent medium heat is supplied to the the pot till enough pressure is created for the water to flow from the bottom chamber, through the funnel and wet the coffee grounds.
5. The process continues until the coffee starts flowing through the spout into the top chamber. The flow of coffee is consistent and finally some residual steam passes through the spout, marking the end of the process.

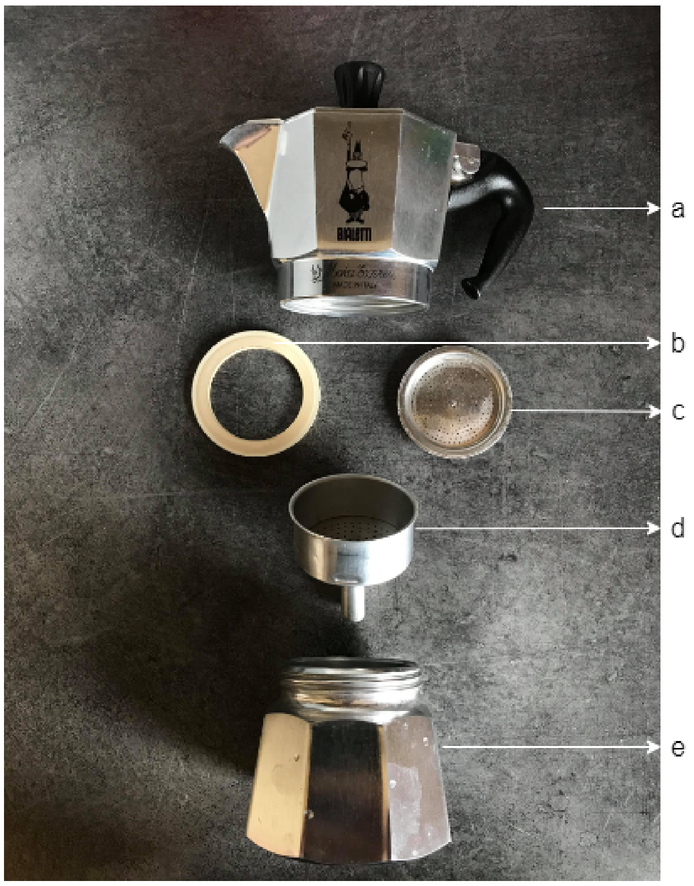


Figure 2.1: Parts of moka pot a. top chamber b. gasket c. filter plate d. funnel e. bottom/boiling chamber

2.3 Moka Pot Physics

To gain intuition about the process and to better interpret data that will be generated in the experiments, it would be useful to understand the physics that governs the brewing process. There has been an attempt to provide a theoretical explanation for the working in [3]. This paper uses Darcy's law of linear filtration to explain the pressure required by the water to pass through the coffee grounds. Saturated water vapour exists above the surface of the water in the boiler chamber. When heat is switched on, the water vapour heats and expands. This applies pressure on the water. When this pressure exceeds atmospheric pressure and the filtration pressure offered by the coffee grounds, coffee starts flowing through the spout, into the top chamber. The water assumes the flavour and aroma of coffee when it comes in contact with the coffee grains and dissolves some of the aromatic oils. Studies [4] have shown the strength and taste of coffee is dictated by parameters like the duration of contact between the water and coffee grains, the pressure and temperature at which the extraction happens, the coffee grain size etc. Darcy's law of linear filtration which governs the filtration pressure is defined by the following equation.

$$P = P_0 + P_f$$

where P_0 is the atmospheric pressure and P_f is filtration pressure which is defined by the equation

$$P_f = \frac{m\eta h}{kS\rho t}$$

where m is mass, η is viscosity coefficient, h is the height of the column through which the water rises, ρ is the density of water and S is the surface area of the funnel, k is the filtration coefficient, t is the time taken by the water to pass through the coffee.

2.4 Prior Work on the Moka Pot

There is very limited literature available on the experimental analysis of the moka. There were two resources that provided some good guidance for the work in this thesis. Some experimental work was performed by Gianino [3] on the moka. The experimental setup described in that paper consists of two thermocouples, one in the spout and one inside the boiling chamber of the moka. By finding the difference in temperatures measured by these thermocouples, the difference in saturated vapour pressure has been calculated. This provides an estimation of the filtration pressure required by the water to flow through the coffee grounds. Further analysis has been performed to find the filtration coefficient of the coffee grounds based on the time the water takes to pass through it. This work provides the explanation to the physics behind the brewing process.

The work of Navarini Et al. [5] inspired the experimental setup used in this thesis. Their setup includes 12 thermocouples, 1 pressure sensor and a level detection circuit. This work highlights the need of collection of temperatures from various locations of the pot in order to correctly model the behaviour of the pot during the brewing process. This study also highlights the role of the dry air in the boiling chamber in the brewing process. A structure has been provided for the brewing process by dividing the process into two phases. The duration where the flow of coffee is smooth and consistent into the top chamber is called the *"regular extraction phase"* and the second phase is named the *"strombolian phase"* which begins when the level of water is below the funnel's bottom tip. This phase is accompanied by immense evaporation and can be identified by the gurgling sound produced by the pot during this phase.

2.5 Possible Faults in the Coffee Preparation Process

To generate a comprehensive dataset to represent the ideal and anomalous conditions of the coffee preparation, it is necessary to identify what constitutes an anomaly. Faults in the physical process reflect as anomalies in the data. A survey on effects of physical parameters like pressure, temperature and coffee grain size on the quality of coffee produced, is provided in [4]. The following section lists all the identified faults in the process which affect the physical parameters that affect the quality of coffee.

1. **Faulty Heating**

   Heat supply is an important component in the coffee preparation using the moka. The heat provided should be consistent and also not too high. High heat is attributed to over extraction of coffee which leads to sub optimal flavours. Another possible fault in the heat supply is an inconsistent supply. When the supply of heat is inconsistent, the smooth flow of coffee through the spout does not occur. The extraction happens irregularly and has a negative impact on the output of coffee.

2. **Incorrect quantity of Water**

   It is highly recommended that the initial amount of water in the boiling chamber should be below the safety release valve. If the water covers the valve, this would make the valve non-functional which could lead to accidents like boiler explosion. If an insufficient amount of water is added, it affects the ratio of water to coffee used and this impacts the final output of the coffee. Less water often leads to underdeveloped flavour in coffee [4].

3. **Coffee based faults**

   As in the case of water, the amount of coffee added should be equal to the recommended amount for the model of moka used. The coffee grounds should fill the funnel but should not be tamped. Having lesser coffee in the funnel than prescribed is also a fault and leads to a lack of flavour in coffee. The grind of coffee should also be carefully chosen, as a super fine grind leads to clogging of the filter plate and may lead to explosion of the boiler if the safety valve malfunctions.

4. **Leak Faults**

   To successfully get the required brew in a moka pot, the condition of the pot should be perfect. A worn out gasket or a malfunctioning safety valve, leads to leak of steam, which reduces the built up pressure and this harms the process. The leak could also arise from not screwing the top chamber on to the bottom chamber tightly. These faults in maintaining a proper seal throughout the brewing process, leads to the extraction taking longer and also sub-optimal output.

5. **Sensor based Faults**

   It is a possibility that anomalies may arise from the sensor system itself. Even if the process may be conducted in perfectly ideal conditions, anomalies in data may occur due to errors in the sensor readings. Such faults can arise from displacement or disorientation of sensors and reduce the performance of anomaly detection algorithms which get all the data from these sensors.

2.6 Digital Twins

This section explains some of the terms related to digital twins, how they are used in the industry. It also explains the some generic steps that can be followed to implement a digital twin and also how it has been applied to the moka pot.

2.6.1 Industry 4.0

Major technological advancements have been mainly observed in leaps, known as 'industrial revolutions' [6]. Increased use of mechanisation and steam power to drive machines was termed the first industrial revolution, use of mass production lines for manufacturing and electrification of these production lines was termed as the second revolution. Automation of manufacturing processes by using programmable devices was the third and finally now , the fourth industrial revolution or *"industry 4.0"* in popular terminology, is the emergence of interconnected cyber-physical systems to make the manufacturing process more intelligent. Digital twin technology is a very important tool in the industry 4.0 paradigm.

2.6.2 Definition

A digital twin is the digital representation of a physical device with a continuous exchange of information between the device and its digital replica. The important ideas defining the digital twin

technology was first described in the works of Dr.Grieves [7]. He defines a digital twin as a collection of all information that can completely describe a product. Any information that can be extracted from the physical device, must be available in its digital twin.
The name *"digital twin"* was coined by him. He also defined other important terms such as *“digital twin prototype”* which contains information required to make the physical entity of the digital model, *“digital twin instance”* which is a digital twin of a specific device or a process. It holds all information of the physical device or the process like a list of components in the device or list of sub-processes in the process. It also holds real time information such as the operational state of the process, typically derived from sensor data. In addition to this, more advanced implementations can also include predictions about the probable future states of the device or process.

2.6.3 Applications

The advent of machine learning algorithms and extensive use of data have made the implementation of digital twins very practical. They are being used in many industries for better visualization of data, condition monitoring and predictive maintenance. Digital twins have been used as tools for condition monitoring in healthcare, aerospace industry [8] and oil and gas industry [9].
The aerospace industry was one of the early adopters of this technology. NASA included digital twins in their technological roadmap. [8]. With high risks involved, there is a need for dynamic diagnostic mechanisms. Having a digital twin of components in a spacecraft, enables the team to obtain information to predict failures and also to diagnose existing problems remotely.
The healthcare industry is evolving to make treatment of patients more tailored to each individual patient [10]. This demands better utilization of patient records and more sophisticated monitoring methods. Digital twins are being considered as a possibility to fulfill these requirements. The real time condition monitoring of patients is possible by having a body worn sensor system on the patient and having a digital twin of the patient. This makes it possible for a doctor to remotely examine a patient by examining the digital twin. Figure shows an architecture for a digital twin based elderly health management system adapted from [10].

2.6.4 Implementation Methodology Adapted for Moka

The methodology used to implement a digital twin is highly dependent on the specific application scenario and the information it needs to capture. Parrott and Warshaw, 2017 [11] have described some general steps to implement a digital twin. The steps have been adapted for the implementation of a digital twin for the moka. The steps are described as follows

1. Building a data acquisition system with a wide variety of sensors to capture relevant data which is useful for the application. In our case, inspired by [5] the pressure and temperature

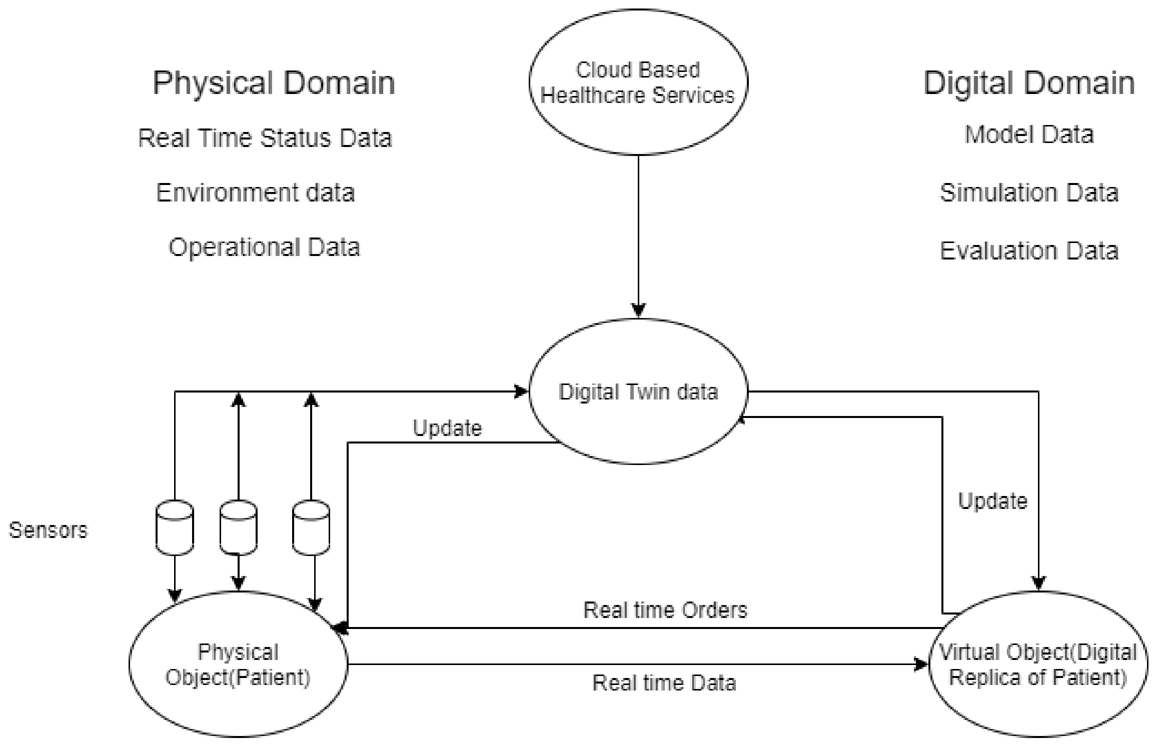


Figure 2.2: A digital twin based healthcare management system

data is identified as critical information. Thus, the data acquisition system is equipped with nine thermocouples and one pressure sensor.

2. Communication channels must be setup between the physical and the virtual domains. Since there is no intention of having a real time feedback to the physical device in this study, there is uni-directional communication from the physical domain to the digital domain via the data acquisition devices and the Labview virtual instrument developed for this experiment.

3. The next step is called *"aggregation"*. This involves data collection, formatting and storage. The data should be suitable for application of analytical tools. This is performed in our Labview virtual instrument.

4. Understanding the data by visualization and application of analytical tools. Some machine learning algorithms were applied on the generated dataset. Our study involves analysing data to detect anomalies in the process.

5. Interpreting the analysis to gain insights into the conditions of the physical device. In our case, to detect if there are any faults occurring in the process and if there is a fault, identifying which fault is affecting the process.

6. This step is named as *"act"* because this step involves taking action based on the insights from the previous step. This typically involves having a feedback loop to the physical device from

the digital domain. Since our study is limited to just anomaly detection, no feedback loop for correction was implemented in this experiment.

Chapter 3

Time Series Analysis

3.1 Introduction

Data is generated in this experiment with nine temperature sensors and one pressure sensor. The data is recorded over multiple iterations of coffee preparation on the moka and each recorded sample has an associated time stamp. The data from each sensor is considered to be a univariate time series and can be analysed independently. The other possibility is that the data from all sensors are grouped together to form a multivariate time series. More formal definitions of univariate and multivariate time series are adapted from [12] and examples for both are provided below.

Univariate Time Series - A univariate time series is a set of real values that have an order defined by their associated time stamp. It can be represented as $X = [x_1, x_2 .... x_N]$ where x_N is the value of X at time N.

The readings from a single pressure sensor can be considered a univariate time series and is visualised in the figure 3.1

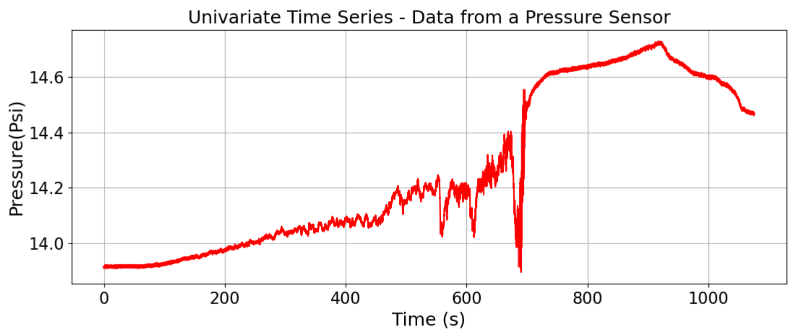


Figure 3.1: An example of univariate time series. Data from a single pressure sensor

Multivariate Time Series - A multidimensional time series of M dimensions consists of M univariate time series where all time series share the same time vector. It can be represented as Y=[X_1, X_2....X_M] where X_1, X_2....X_M are univariate time series. The readings acquired from a data acquisition system equipped with multiple sensors working together can be considered as a multivariate time series and is depicted in the figure 3.2

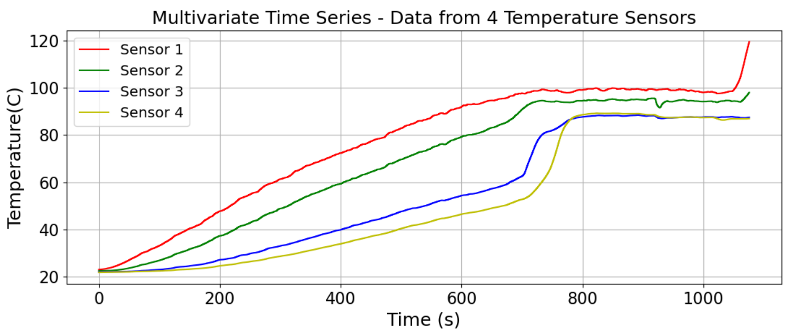


Figure 3.2: An example of multivariate time series. Data from multiple temperature sensors

3.2 Labelling of a Time Series Dataset

Time series can be labelled in multiple ways depending on the application. One method is to label segments in a long time series. Usually, there is a label allocated to the data sample at each time stamp. This type of labelling is useful in activity recognition applications. A very basic example is illustrated in the figure below. A thermocouple is used to determine if the user's hand is resting on a platform. If the hand is detected, the sample is labelled with a '1' and when there is no hand on the platform, the sample is labelled as '0'. The data and its binary label is plotted in the figure 3.3.

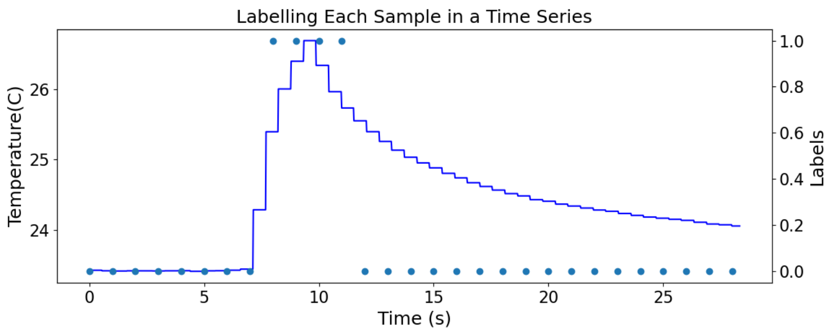


Figure 3.3: Plot of temperature data and binary label. Each recorded sample has a binary label

Another labelling method is that a label is associated with the whole time series. This can be a univariate or a multivariate time series. This method of labelling is followed, to annotate the dataset generated in our experiment. This method is useful for time series classification applications. 3.4 shows an example illustrating the method.

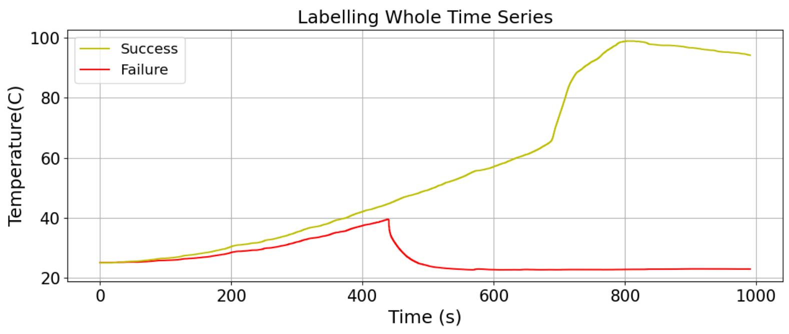


Figure 3.4: Plot of two labelled time series. Each time series has a label

3.3 Anomalies in Time Series Data

An anomalous observation is one which deviates from the expected behaviour. These anomalies can occur due to noise, faulty equipment or it may be generated due to a deviation in the physical behaviour of the device under observation. [13] provides a review on anomaly detection in the context of time series data. The anomalies can be categorised into three types based on their occurrence. They are point anomalies, sub-sequence anomalies and whole time series anomalies and each type is defined and illustrated in this section.

Point anomalies are defined as individual anomalous observations which deviate from the other observations in the same time series. The data at a particular time stamp is very dissimilar when compared to other data points in the time series. The figure 3.5 illustrates a point anomaly.

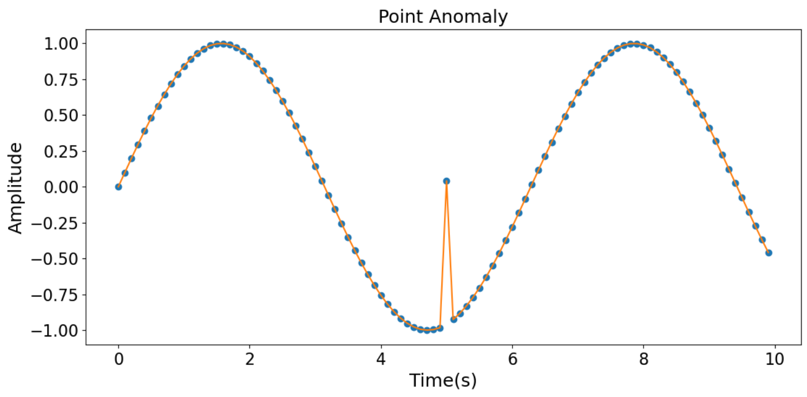


Figure 3.5: Illustration of a point anomaly

Sub-sequence anomalies are said to occur when few consecutive observations put together display a significant deviation in the normal behaviour observed in the time series. Figure 3.6 shows a sub-sequence anomaly where a group of points exhibit anomalous behaviour but each individual point is not an anomalous reading.

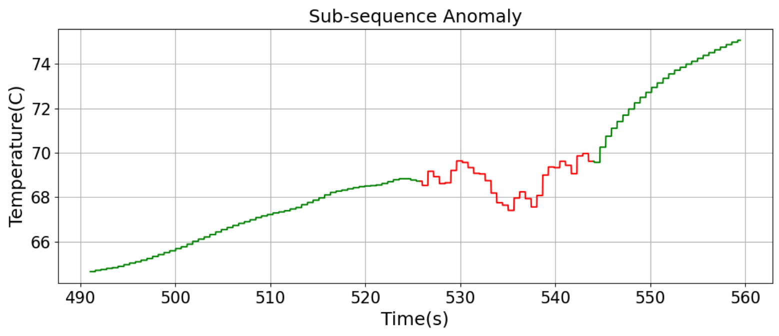


Figure 3.6: Example of sub-sequence anomaly

Whole time series anomaly is when the whole time series is generated in an anomalous setting. An entire time series can be considered as an outlier and classified as an anomaly when compared to other time series data. This is achieved by using similarity metrics which are discussed in the subsequent sections. Whole time series anomaly is depicted in the figure 3.7

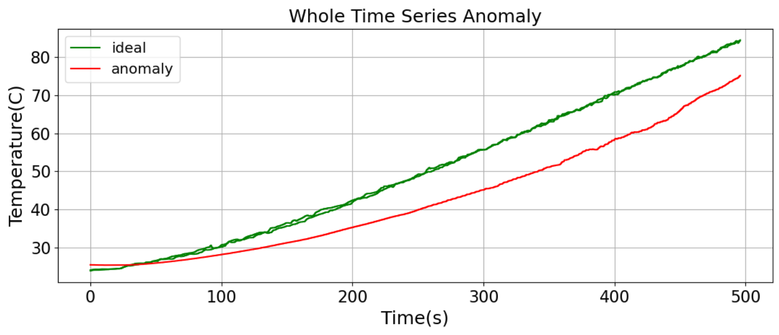


Figure 3.7: Whole time series anomaly

3.4 Distance Measures for Time series

This subsection explains the distance measures used in the anomaly detection methods employed in this thesis.

3.4.1 Euclidean Distance

This is a very common measure of distance between two vectors and is widely used in time series applications as well. The distance is calculated using the following formula

$$Dist(X1, X2) = = \sqrt{\Sigma_i^n (X1_i - X2_i)^2}$$

where X1 and X2 are two univariate time series as defined in section 3.1. This distance measure requires both the vectors to be of the same length.

3.4.2 Dynamic Time Warping

Dynamic time warping is a popular approach and can be considered as an extension to euclidean distance. If there is a misalignment in data, the euclidean distance measure penalises it because distance is calculated only between the samples of two time series at the same time stamp. But with the dynamic time warping approach, small misalignments do not cause a remarkable increase in the distance between the two time series. This measure can be used to measure dissimilarity between two time series of unequal lengths. The method to calculate dynamic time warping distance between two time series X1 and X2 with lengths I and J respectively as mentioned in [14] with euclidean distance as the local cost measure is summarised as follows.

1. Construct a distance matrix with euclidean distances between each pair points of X1 and X2. The dimension of this distance matrix is IxJ and each element of the distance matrix is calculated by the following rule.

   $$C(i, j) = dist(X1_i, X2_j)$$

   where i ϵ [1:I] and j ϵ [1:J].

2. A warping path is found in the distance matrix which reduces the overall distance between the two time series. The warping path is defined as w = (w_1,w_2...w_L) where w_l = (i_l, j_l) with i ϵ [1:I] and j ϵ [1:J]. The warping path calculation is constrained with the following conditions

   - The first constraint is referred to as boundary condition. It constraints the first element in the warping path w_1, to be cell(1,1) in the distance matrix, and the last element in the warping path w_l to be the cell(I,J).
   - The indices of the warp path must be a non decreasing sequence. This constraint is called as monotonic condition.
   - Elements of w are chosen such that there is just one step traversal in the distance matrix for two consecutive elements in w.

3. The total distance is calculated as a sum of the elements on the warping path.

An example of the distance matrix and the calculation of warp path is shown the figure 3.8. Two time series of different lengths are used and to calculate the warping path, every cell of the distance matrix needs to be filled in the classical approach.

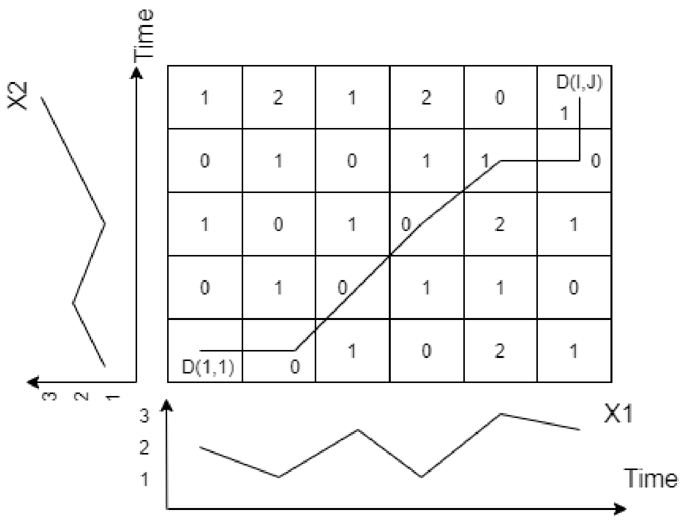


Figure 3.8: Distance matrix with the calculated warp path

3.4.3 FastDTW

The classical DTW approach explained previously, has a quadratic time and memory complexity and the data generated in our experiment is quite sizeable with length of each iteration running in tens of thousands of samples. Thus, it is impractical to use the classical approach to calculate the distance between the time series. FastDTW is an approach developed in [15] which produces an approximate alignment that is close to the optimal alignment provided by the classical approach. This is done at a linear time and space complexity which is much faster and less memory intensive compared to the classical approach. An overview of the FastDTW approach is given below.

1. Reduce the length of the time series by taking the average of two adjacent pairs of points. The new length is half the length of the original time series. Repeatedly run this step to generate time series of multiple resolutions.

2. Calculate the warp path on a lower resolution data and use this to estimate the path in the higher resolution. At the lowest resolution, classical DTW is used to calculate the warp path.

3. Use the projected warp path obtained from the lower resolution and find a more optimum path in the higher resolution by running a constrained DTW algorithm where distance calculations are done only for the cells in the projected warp path and the ones in the neighbourhood of

this path. The neighbourhood is controlled by a parameter which is referred to as radius. For radius equal to the length of the time series, the FastDTW approach generalizes to the classical DTW approach.

4. Step 2 and 3 are recursively run to get the warp path from the lowest resolution to the actual resolution of the time series.

This approach negates the need to fill up the whole cost matrix as required by the classical DTW approach and only the necessary distances are calculated and filled according to the estimations made on the lower resolution data. The figure 3.9 adapted from [15] shows the projection of warp path from the lower resolution to a higher resolution and illustrates the reduction in the number of calculations required to compute the approximate warp path. The dark shaded cells represent the projected path and the lightly shaded squares represent the neighbourhood which are included in the refinement of path calculations. The unshaded cells need not be calculated at all, thereby reducing the number of calculations.

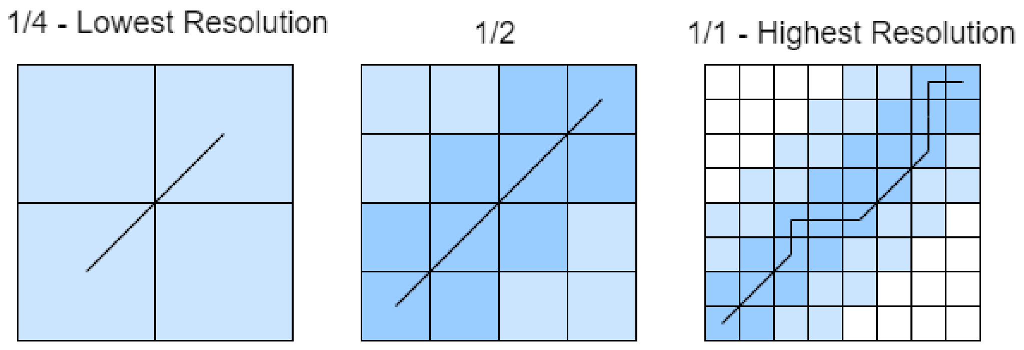


Figure 3.9: The projection of warp path from lower resolution to higher resolution

3.5 Detection of Whole Time Series Anomalies

As mentioned earlier in section 3.2, this project has the goal of generating a dataset with a single label for each iteration of coffee preparation. Thus, detection of whole time series anomalies becomes the topic of interest. There are two main approaches to detect whole time series anomalies as suggested by [13]. They are dissimilarity based methods and feature based methods and are explained as follows.

3.5.1 Dissimilarity Based Methods

The intuition behind this method is that time series data generated under anomalous conditions are dissimilar to data generated under ideal conditions. Some of the measures that can be used to determine the degree of similarity or dissimilarity between two data vectors and that have been used in this thesis have been explained in section 3.4. These methods could be performed in the supervised approach like the nearest neighbour classifier, where labelled data is used for training a model or also in unsupervised methods where data labels need not be provided and the model makes use of the inherent dissimilarity in the data, to group similar data together. The unsupervised approaches include methods such as K-means clustering and Hierarchical Clustering. [16] provides a review on clustering algorithms for time series data. The approaches have been explained in the following subsections.

3.5.1.1 K-means Clustering

The K-means algorithm is one of the most popular and simple clustering algorithms used to assign each data point into one of K clusters. K is an input parameter required for the algorithm.The overview of the algorithm is as follows.

1. K random points are initialised as centroids of the clusters.

2. Distance is calculated between each data point and every centroid.

3. Each data point is assigned to the cluster whose centroid is located closest to the point(distance is minimum). Euclidean distance is a popular distance measure used in this calculation and is used in this thesis as well.

4. After each data point is assigned its cluster, the new cluster centroids are calculated by finding the mean of all points belonging to the cluster.

5. Steps 2,3 and 4 are repeated until the cluster centroids do not show variation.

3.5.1.2 Hierarchical Clustering

Another method of clustering used to cluster raw time series data is hierarchical clustering. As the name suggests, this clusters the data in a hierarchical fashion, grouping most similar data points into smallest of the clusters and then grouping the smaller clusters into bigger clusters. Agglomerative and divisive approaches can be adopted to implement this algorithm. In the Agglomerative approach, each data point is considered to be a cluster and then most similar clusters are merged together as we move higher in the hierarchy. In the divisive approach, the data points are grouped together into one cluster and then divided into smaller clusters as we move lower in the hierarchy. Agglomerative approach is used in this study.

3.5.1.3 Nearest Neighbour Classifier

Nearest neighbour classifier is one of the simplest supervised approaches. In time series classification problems, it is used to test different distance measures and works as a decent baseline method [17]. It works as follows.

1. The distance between the sample under test and every sample in the training data is calculated.
2. The samples with the least distance to the test sample are its neighbours. Generally, an input parameter N decides the number of nearest neighbours under consideration.
3. The most common label among the neighbours is given to the test data point. In case of 1-Nearest Neighbour classifier, only one neighbour is considered and the label associated with that neighbour is assigned to the test data point.

3.5.2 Feature Based Methods

Time series data is generally very high dimensional. The amount of data that needs to be processed depends on various factors such as the sampling rate used to produce data, the duration for which the data was recorded, number of sensors used in the data acquisition system, etc. One way to avoid the high processing times is to extract relevant features which represent the data in a lower dimensional space. These features are fed to a conventional classifier such as a support vector machine. An SVM based classifier is explained below.

3.5.2.1 SVM Classifier

Support vector machine is a very popular tool used for classification. This algorithm finds an optimal hyper-plane which separates two classes. This hyper-plane is found such that the distance is maximum between the two classes of data. Since this is primarily, a binary classifier, in order to classify data into more than two classes, strategies like one vs one approach is used. In this approach, an SVM classifier is trained for each pair of labels. Thus, for a k-class problem, (k)(k-1)/2 binary classifiers are trained. During the testing phase, all the binary classifiers classify the sample between the two classes that they are trained for, and a voting is conducted for obtaining the final class label. An SVM classifier is used for monitoring the condition of a tool in [18] and the SVM algorithm can be briefly described in the following steps with the help of figure 3.10 .

- Given a dataset with labels y ϵ {+1, -1}. Find hyper-planes such that x_i.w+b $\geq$ +1 when y_i=+1 and x_i.w+b $\leq$ -1 when y_i=-1. x_i is a data point and w is the weight vector and b is bias. These hyper-planes are called the support vectors and are named as H1 and H2 in the illustration.

- The goal is to identify a decision function which is a hyper-plane H with equation x_i.w+b=0 which correctly classifies the samples into their respective classes and also maximises the margin. The margin is the sum of d_+ and d_- shown in the figure and is found to be 2/||w|| where ||w|| is the euclidean norm of vector w.
- To maximise the margin, we need to minimise ||w|| with the linear constraint $y_i(x_i.w+b)-1\geq 0$. The problem is typically solved using the Lagrangian method.
- To enable non linear classification, the input space is mapped to a higher dimensional space where the decision function transforms into a linear hyper-plane. This transformation is called the kernel trick and provides flexibility to the SVM algorithm. Some of the typical kernels are linear, polynomial and radial basis functions. The implementation of SVM classifier used in this thesis has an RBF kernel which has the following formula.
  $K(x_1,x_2) = \exp\left(-\frac{||x_1-x_2||^2}{2\sigma^2}\right)$ where x1 and x2 are two samples and σ is a parameter.

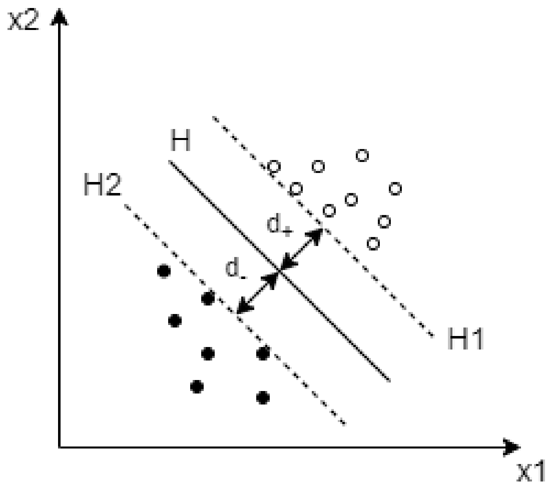


Figure 3.10: Illustration of an SVM classifier

3.5.3 Stacking of Univariate Classifiers

To deal with the multivariate nature of the generated data, a combination of univariate classifiers has been tested. [19] proposes a stacking structure where the multivariate time series is split into its univariate components and univariate classification is used on that data. In their proposed structure, K nearest neighbours with the DTW dissimilarity measure is used as the univariate classifier. The decisions from these classifiers are fed to a Naive Bayes classifier or an SVM classifier to get the final multivariate classification of data. A similar structure has been tried in our study where a univariate classifier is trained on data from a single sensor. Since there are ten sensors in the data acquisition equipment, ten classifiers are trained individually. But instead of having a 2nd level of classification, a majority voting scheme was used to find the final classifier output. Structure of the ensemble tested in the study is illustrated in figure 3.11.

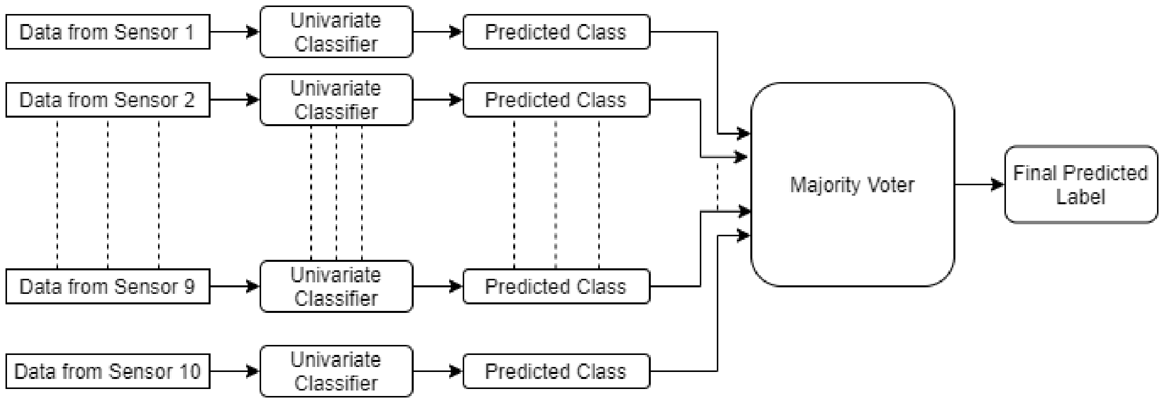


Figure 3.11: Ensemble of univariate classifiers to deal with multivariate data

3.6 Evaluation of Models

Once the models have been trained, they need to be tested with unseen data. With a very limited amount of data available, the leave one out cross validation technique seems to be the most suitable method for the evaluation of approaches on this dataset. The leave one out cross validation technique has been used to obtain a mean accuracy score and F1 score of the models. This measure is used to evaluate how the models have performed over the dataset. In this study, the unsupervised approaches are also evaluated. In such cases, the labels in the dataset are used only for evaluation purposes when testing the unsupervised approaches.

3.6.1 Leave One Out Cross Validation

The leave one out cross validation method is used when the data available is scarce. This applies to our case. In this approach, all data points except one is used as training data to train a model and the remaining data point is used as a test sample. The method is repeated such that each data point in the dataset is the test sample in one of the train-test splits. The performance is evaluated over all the train-test splits and then a mean performance measure is calculated to give an overall performance of the model under evaluation for the given dataset. The figure 3.12 shows an example of the training-test splits generated by LOOCV algorithm. There are four samples in the data and one sample is used as a test sample in each split.

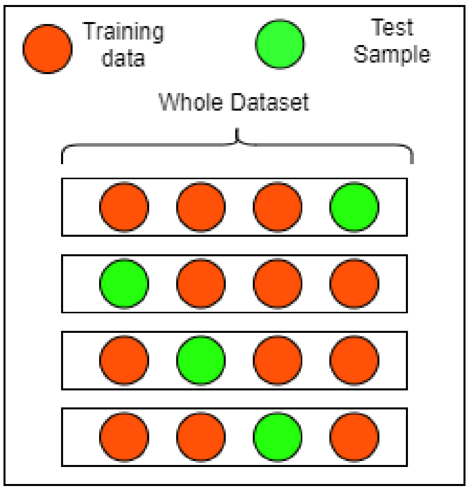


Figure 3.12: Leave one out cross validation over a dataset with four samples

3.6.2 Accuracy Score

Accuracy of a model gives a measure of the correctness of prediction by a classifier. It is calculated by the following formula.

$$\text{Accuracy Score} = \frac{\text{Correct Predictions}}{\text{Total Number of Predictions}}$$

3.6.3 F1 score

When dealing with imbalanced datasets, accuracy score may not be the best performance measure. For example, considering a dataset with 10 samples with 8 negatives and 2 positive samples. A classifier which predicts all samples as negative will have a high accuracy score of 0.8. In such cases it is necessary to quantify the detector's capability to correctly identify the positive samples. This can be achieved by using F1 score as a performance measure.
The F1 score is a harmonic mean of two measures called precision and recall. The precision defines the ability of the classifier to avoid false positives. When the number of false positives in the predictions increase, the precision of the classifier decreases. Similarly, recall defines the ability of the classifier to correctly identify all the positive cases. The formula to calculate precision, recall and the F1 score is given below.

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

$$\text{F1 Score} = 2\ .\ \frac{\text{Precision}\ .\ \text{Recall}}{\text{Precision} + \text{Recall}}$$

Chapter 4

Experiment

4.1 Overview

To generate data required for this study, a suitable data acquisition system was setup. The main motivation for this setup was derived from the study of Navarini et al. [5]. To model the functioning of the moka pot, three main parameters are used, namely, the temperatures on the surface of the moka pot at various locations, the internal temperature of the moka and also pressure inside the boiling chamber. To capture this data, 9 thermocouples and 1 pressure sensor were interfaced with data acquisition devices from National Instruments and a Labview virtual experiment was created to handle the incoming data and store it in the required format in CSV files. This data was used for further analysis which is explained in the subsequent chapter. This chapter describes the hardware setup, the Labview virtual experiment and the methodology used to generate data .

4.2 Hardware Description

This section gives a detailed description of important hardware components used in the setup. The components include the moka pot, the heat source, thermocouples, pressure sensor, data acquisition equipment from National Instruments.

1. **Moka Pot**

   The *Moka Express* from Bialetti Industries is one of the most popular models of moka and is widely used by consumers. Owing to this popularity, this particular model was chosen for the study. A 12 cup moka was chosen for easy application of sensors. The boiler capacity is 775ml and one this pot typically produces 12 shots(1 shot=30ml) of coffee.

2. **Electric Stove**

   A generic electric kitchen stove was used as the source of heat for coffee preparation. This

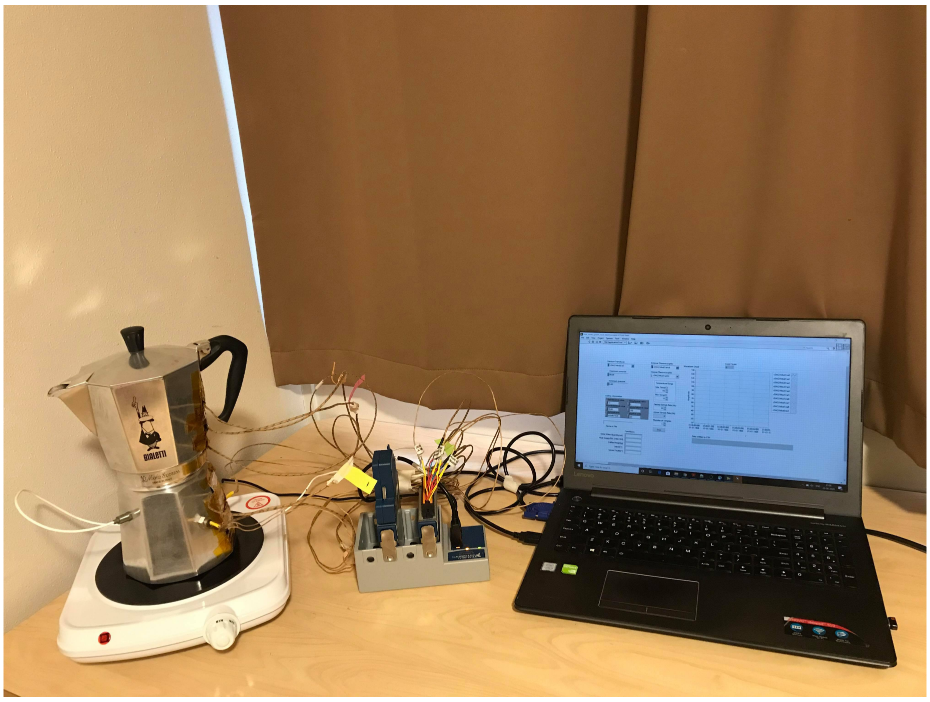


Figure 4.1: The experimental setup

is typically used by a general consumer of the moka. The stove contains multiple heating levels which was convenient to produce data on different heat settings. Another stove was used to generate a class of anomalous readings. This stove maintained a constant stove top temperature, rather than a consistent heat supply as required by the moka.

3. **Thermocouples**

   Thermocouples have a junction formed by two electrical conductors which produce a voltage depending on the temperature. Depending on the combination of metals used to form this junction, they are categorised into different types and have different properties such as range, cost and stability. Two types of thermocouples used in this setup were the type-K and type-T. The external thermocouples were of type T and the internal thermocouples were of type K.
   The junction of T-type thermocouples are made of copper-constantan(a copper-nickel alloy). They have a range of -270 to +370 degrees Celsius. Both conductors used in the preparation of this thermocouple are non magnetic and there is no abrupt change in the characteristics of the thermocouple over the range of use.
   K-type thermocouples contain a junction made of two alloys, they are nickel-chromium and nickel-Alumel alloys. They have a higher range of –270 to +1260 degrees Celsius. Due to their low cost and wide range of temperature, they are the most commonly used general purpose thermocouples.

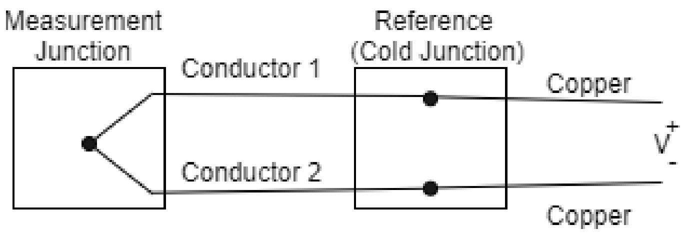


Figure 4.2: Structure of a thermocouple - adapted from [20]

4. **NI-9213 Thermocouple Module**

   The NI-9213 [21] is a data acquisition device specifically designed for conditioning of signals from thermocouples. Data can be acquired from 16 independent channels. There are two timing modes for operation, a high resolution mode with 1 sample/second or the high speed mode with a maximum sample rate of 75 samples/second.

   In order to capture the absolute temperature from a thermocouple, it is necessary to know the temperature of the cold junction so that it can be relatively scaled. This is called cold junction compensation. Figure 4.2 is a diagram of a thermocouple structure for reference. The NI-9213 module provides cold junction compensation internally, thus making it convenient to measure absolute temperature data from thermocouples. The typical error in measurement for temperatures ranging from 0-200 degrees is +/- 1 degree for both types of thermocouples.

5. **Pressure Sensor**

   The pressure sensor used in the experiment belongs to the XTME-190(M) series, produced by Kulite [22]. This transducer can be safely used in temperatures of up to 232 degrees Celsius, which makes it safe to be used inside the moka pot for the experiments as the expected temperature range is well below the rated temperature of the sensor. It has a piezoresistive element to change its resistance according to the pressure applied. The sensor has an inbuilt full-bridge configuration as illustrated in the figure. Thus, to acquire data from the sensor, the activation voltage must be supplied and the voltage must be measured from the other terminals. This is achieved by using the NI-9237 module [23].

6. **NI-9237 Bridge Measurement Module**

   To measure the pressure from a pressure transducer, a Wheatstone bridge setup is used, with one of the resistances being sensitive to pressure. Excitation voltage is provided across two terminals of the bridge and voltage is measured off the other two terminals. This data acquisition module provides the necessary activation voltage for the measurement to occur. If the internal timer is used, the minimum data rate is 1.613k samples/second. The unit of measurement is Volts/Volt. i.e the output is the ratio of measured input voltage and the excitation voltage. The reading needs to be scaled in the user application to get the pressure

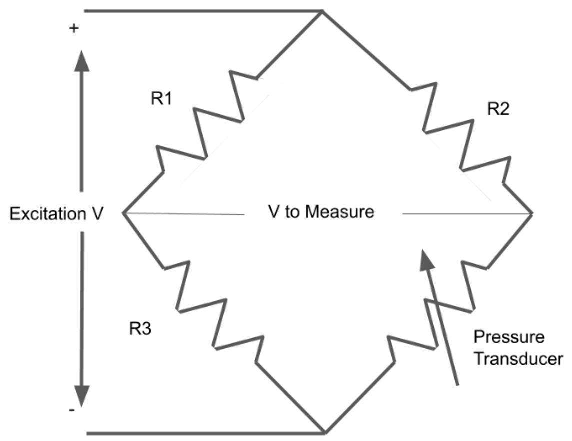


Figure 4.3: A full bridge setup for measurement of pressure

reading. The typical error in measurement is rated at 0.2% of the reading.

7. **NI cDAQ-9174 Chassis**

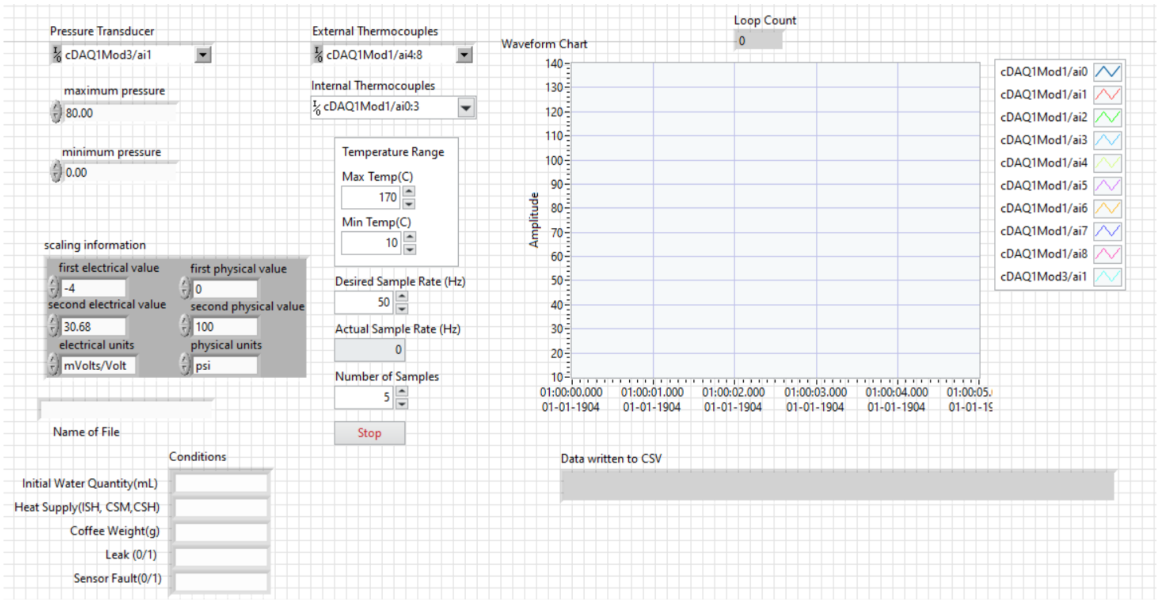
   The NI cDAQ-9174 chassis enables the use of multiple data acquisition modules and connects to a single USB port on the host PC. It provides features for triggering and synchronisation between the modules that are attached to the chassis. It also provides digital routes for managing the data flow from the modules to the host PC.

4.3 Labview Virtual Instrument

A virtual instrument was created in Labview for interfacing the data acquisition hardware with the host PC. This virtual instrument provides a user interface to control data collection. It enables the control of data flow from the sensors through the data acquisition devices and stores them into CSV files in the required format. For each iteration of the coffee preparation process, the virtual instrument creates a CSV file with a name provided by the user and also updates a file containing labels.

4.3.1 Front Panel/User Interface

The front panel provides the user interface to control the hardware and also allows to enter information about the data acquisition. It has a waveform display which allows the user to see the acquired data in real time. Figure 4.4 shows the front panel of the virtual instrument. The user can control sampling frequency, select the data acquisition channels and also enter the expected range of measurements for more accurate analog to digital conversions. The user can provide a name for the file in which the generated data is stored. The user can also type in the details of the conditions under which the data was collected. This information is updated in the label file.


Pressure Transducer: cDAQ1Mod3/ai1

maximum pressure: 80.00

minimum pressure: 0.00

scaling information

first electrical value: -4	first physical value: 0
second electrical value: 30.68	second physical value: 100
electrical units: mVolts/Volt	physical units: psi

Name of File

Conditions

Initial Water Quantity(mL)	
Heat Supply(ISH, CSM,CSH)	
Coffee Weight(g)	
Leak (0/1)	
Sensor Fault(0/1)	

External Thermocouples: cDAQ1Mod1/ai4:8

Internal Thermocouples: cDAQ1Mod1/ai0:3

Temperature Range

Max Temp(C): 170

Min Temp(C): 10

Desired Sample Rate (Hz): 50

Actual Sample Rate (Hz): 0

Number of Samples: 5

Stop

Loop Count: 0

Data written to CSV

Figure 4.4: Front panel of the virtual instrument

4.3.2 Highlights of the Block Diagram

The block diagram represents the flow of data and also the operations performed on the data. It is the graphical source code. The graphical code must be written carefully to get the desired functionality out of the hardware. It also needs to make use of inputs from the front panel which is fed by the user. Some important features of this block diagram is explained in this section.

1. To have a synchronised measurement of all sensors, channels from both the data acquisition modules are placed in the same task. Even though they have different virtual channels(units, scaling information and physical channels), placing them in the same task allows the NI-DAQmx driver to synchronise the modules. The figure 4.5 shows the section of the block diagram where different measurement channels are defined and are associated with the same task.

2. As mentioned in 4.2, the sampling rates of the module used for acquiring thermocouple data and the module used for acquiring pressure data are different. The bridge based DAQ module samples at a minimum rate of 1613Hz, while the thermocouple DAQ module samples at a maximum rate of 75Hz. When two such modules are mounted on the same chassis and are a part of the same task as in our case, the NI-DAQmx driver carries out the data acquisition at the rate of the faster device and simply returns duplicate samples from the slower module. In our case, the whole system runs at a minimum rate of 1613Hz. To overcome the problem of unnecessary logging of data and to log data at 50Hz, an additional data logging loop was created which is shown in figure 4.6.

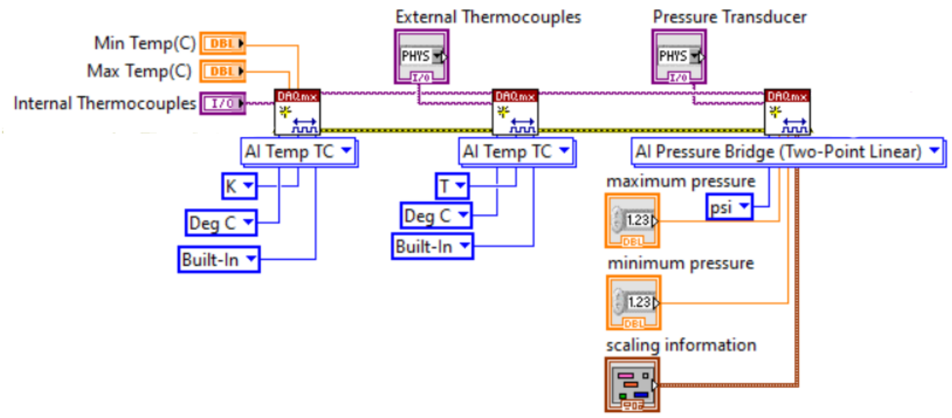


Figure 4.5: Multi-device task to enable synchronization across different hardware modules

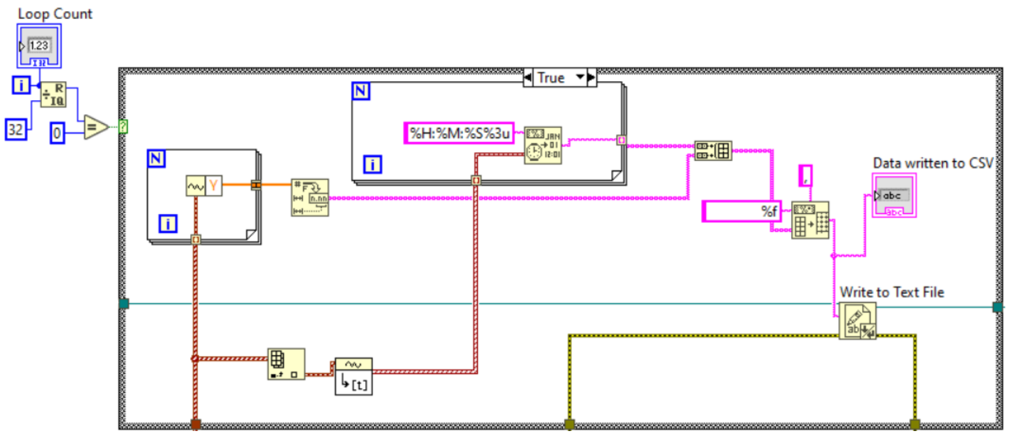


Figure 4.6: The data logging loop

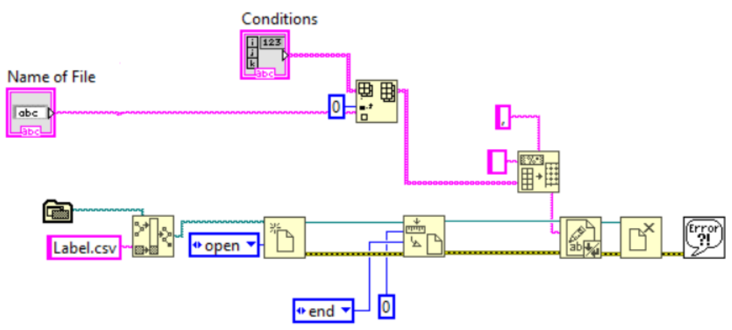


Figure 4.7: Updating the Label.csv file with the conditions under which data was recorded

3. To enable a systematic data collection process, the label file is updated for each coffee preparation with the conditions under which the coffee was prepared.

4.4 Data Collection Methodology

4.4.1 Process Conditions for Data Generation

Preparation of coffee in a moka pot has many process parameters. They are **initial water quantity, the weight of coffee grounds, heat supply and the condition of the pot**. Identifying the ideal process conditions to make the perfect cup of coffee is a challenge. The process is not standardised and an ideal cup of coffee is a subjective definition depending on individual preferences. There is no scientific literature that defines perfect process conditions that produce the ideal cup of coffee. Keeping that in mind, the ideal conditions are considered to be those which do not have any faults mentioned in section 2.5. To produce a fault free iteration, coffee was prepared according to the procedure defined in section 2.2. To generate anomalous data, faults described in section 2.5 were injected one after the other. This fault injection produced data under six different anomalous conditions, namely **insufficient water, insufficient coffee, high heat, inconsistent heat, leaks in the pot, sensor based faults**. The table 4.1 contains the conditions under which data was generated and the corresponding label assigned in the generated dataset.

	ideal	water	coffee	HH	IH	seal	sensor
Water Quantity(ml)	630-640	<550	630-640	630-640	630-640	630-640	630-640
Coffee Grinds Weight(g)	45-50	45-50	<30	45-50	45-50	45-50	45-50
Heat Supply	CSM	CSM	CSM	CSH	IS	CSM	CSM
Leak	No	No	No	No	No	Yes	No
Sensor Faults	No	No	No	No	No	No	Yes

Table 4.1: Conditions under which data was generated and their corresponding labels

To produce data for insufficient water anomaly, the initial amount of water added to the boiling chamber was significantly lesser. Similarly for insufficient coffee anomaly, the amount of coffee grounds added to the funnel was much lesser than the quantities used for ideal iterations. The high heat anomaly was generated by preparing coffee on the highest heat setting of the stove. This setting was named consistent supply high**(CSH)** whereas the lower setting used to produce the ideal scenario was named consistent supply medium**(CSM)**. Similarly the inconsistent heat anomaly was generated by using a second stove which did not provide a continuous supply of heat. This was named inconsistent supply**(IS)**. To have a leak in the pot, the top chamber of the pot was not screwed on properly. This condition of the pot was used to produce data for seal anomaly. And finally, the sensor based anomalies were introduced in the external temperature data by disturbing the sensors and intentionally detaching and reattaching the sensors during the brewing process.

4.4.2 Procedure for Data Collection

To generate a structured dataset from the setup using the designed virtual instrument, a systematic procedure was followed which included the following steps.

1. The name of the file was entered in the user interface of the Labview program. This is the file in which the acquired sensor data is saved. The naming format used to save data is "moka_iteration_number.csv".

2. Weight of water and the coffee grounds was measured and entered into their respective fields in the user interface.

3. Other fields such as the heat setting, leaks in the pot, occurrence of sensor fault were filled. These fields were updated in the label file along with the iteration number by the Labview program.

4. The virtual instrument was started to record data from all sensors till the completion of the brewing process.

Chapter 5

Data Analysis

5.1 Dataset Description

In order to create a data driven anomaly detection system of the coffee making process, a dataset was generated by the experiment described in chapter 4. Every iteration of the coffee making process on the moka generates a multivariate time series with 10 attributes. Each attribute represents data from a single sensor out of 9 temperature sensors and 1 pressure sensor. The data was logged at 50Hz from each sensor and the locations of these sensors are shown in the Figure 5.1.

The data generated for each iteration was saved in a CSV file with the iteration number included in the name of the file. Each file has 11 columns with time stamp as the first column and the rest is the raw sensor data with each column representing data from 1 sensor. The dataset contains 64 such files representing 64 iterations of coffee preparation. There are 40 ideal iterations where no faults were induced in the process and 24 anomalous iterations, with 4 iterations each for 6 types of anomalies . The structure of the dataset is summarised in the table 5.1. The temperature sensor readings have been validated with a standard laboratory thermometer. But the data from the pressure sensor could not be correctly scaled to obtain accurate pressure readings. Thus, to eliminate the uncertainties in the magnitude of the readings, the pressure data was max-min normalised over the whole dataset and saved in the files.

There is also another file called Label.csv which contains the label of each iteration as defined in table 5.1. It also contains the conditions under which the data was generated for each iteration. The format of the raw data and the label.csv file is shown if figure 5.2. The water quantity and coffee quantity columns contain numeric values, leak in pot and sensor faults are denoted by a binary label. The heat supply column holds symbols CSM, CSH or IS representing consistent supply medium, consistent supply high or inconsistent supply respectively. The label column holds one of the labels mentioned in table 5.1 depending on process conditions of the iteration.

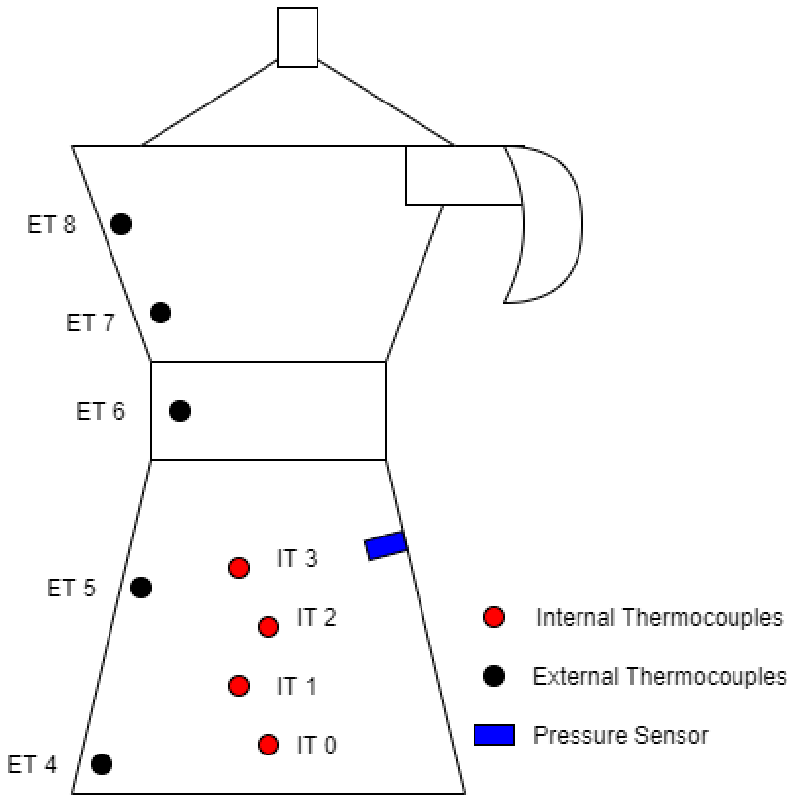
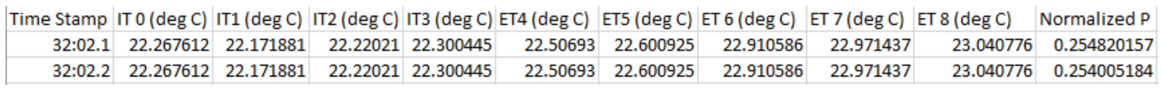


Figure 5.1: Location of sensors on the moka pot

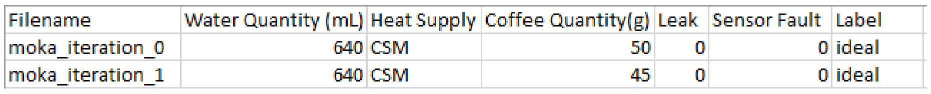
Case	Label	Number of Iterations
Ideal Conditions	ideal	40
Insufficient Water	water	4
Insufficient Coffee	coffee	4
High Heat	HH	4
Inconsistent Heat	IH	4
Leaks in Pot	seal	4
Sensor Faults	sensor	4

Table 5.1: Structure of the generated dataset along with the labels



Time Stamp	IT 0 (deg C)	IT1 (deg C)	IT2 (deg C)	IT3 (deg C)	ET4 (deg C)	ET5 (deg C)	ET 6 (deg C)	ET 7 (deg C)	ET 8 (deg C)	Normalized P
32:02.1	22.267612	22.171881	22.22021	22.300445	22.50693	22.600925	22.910586	22.971437	23.040776	0.254820157
32:02.2	22.267612	22.171881	22.22021	22.300445	22.50693	22.600925	22.910586	22.971437	23.040776	0.254005184

(a)



Filename	Water Quantity (mL)	Heat Supply	Coffee Quantity(g)	Leak	Sensor Fault	Label
moka_iteration_0	640	CSM	50	0	0	ideal
moka_iteration_1	640	CSM	45	0	0	ideal

(b)

Figure 5.2: (a) Sensor data from each iteration (b) Format of Label.csv file

5.2 Data Visualization

For better understanding of the data and to design feature based classifiers for anomaly detection, it is important to visualise the generated data. A python script was written for generating the plots of required data. This section presents some important plots which give a good representation of the generated data and some intuitions behind the brewing process using the moka.

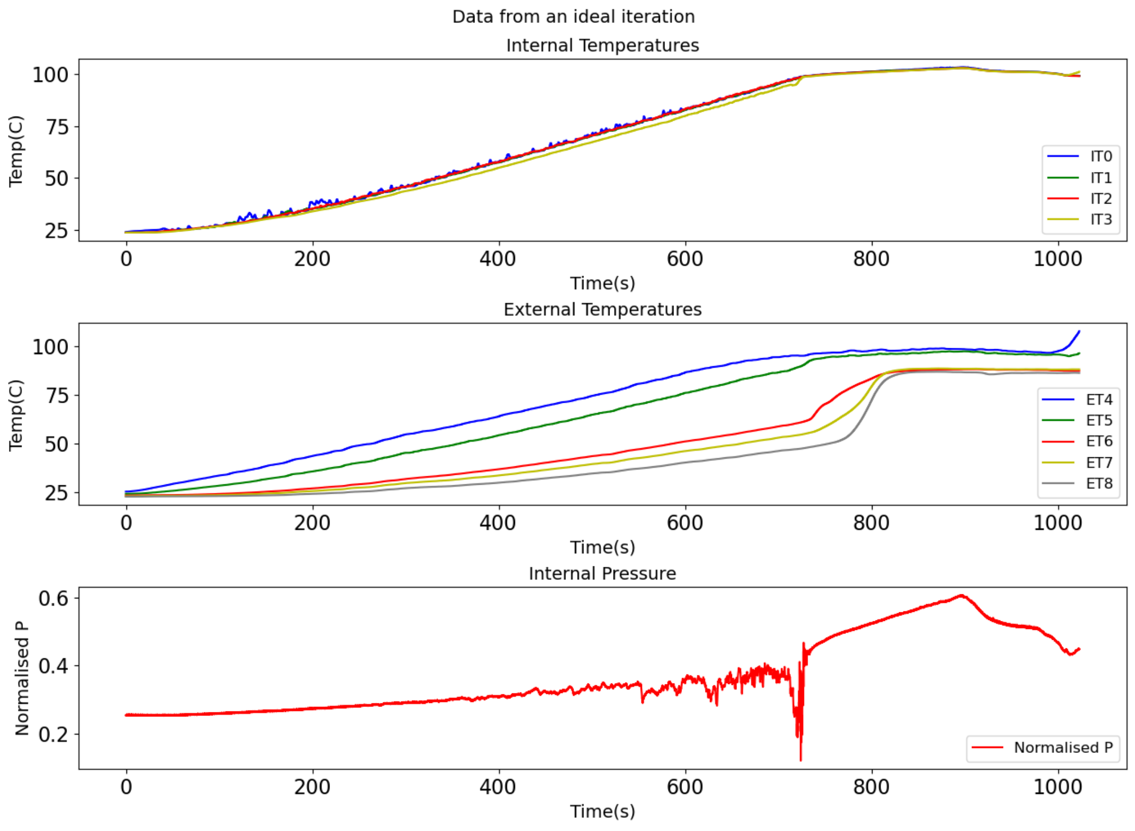


Figure 5.3: Plot of a single ideal iteration of coffee preparation

The Figure 5.3 shows the multivariate time series generated by one coffee preparation process.

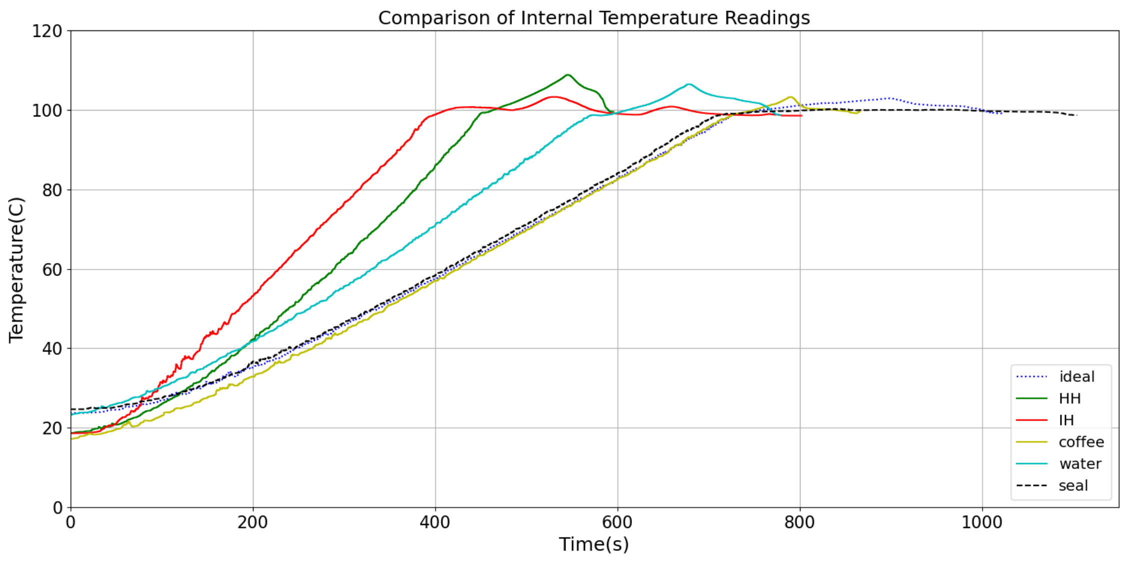


Figure 5.4: Comparison of internal temperatures

The temperature readings are on the Celsius scale and the pressure data is normalised to remove uncertainties in the scaling of readings from the sensor.

More interesting graphs for getting insights on anomaly detection is presented in Figures 5.4, 5.5 and 5.6. These graphs make a comparison between data generated from an ideal iteration and one iteration from each anomalous setting. To represent internal temperature data, one of the internal thermocouples(IT 1) is used. The data from ET6 thermocouple is plotted as external temperature data. The location of the ET6 thermocouple is such that, when hot water enters into the coffee grounds, there is a significant rise in the temperature measured by this sensor.

It needs to be noted that sensor anomalies are not analysed with the other anomalies because they exist in only in one of the sensors. Thus, in most of our univariate analysis, it would not be identified. Sensor anomalies are dealt separately.

Some practical observations from the graphs are listed below.

1. The process duration is different for each iteration and is considerably lesser in the case of HH, IH, water, coffee anomalous conditions. For seal anomaly, it is considerably more. This is justified because, in case of HH and IH, the heat supplied is higher than the ideal condition making the process faster. In case of coffee and water anomalies, since lesser material is used, the process is quicker. But in the case of seal anomaly, there is a leakage. This leads to some steam pressure being wasted and there is a delay in the output.

2. The HH anomalous case has the highest internal temperature, external temperature and pressure.

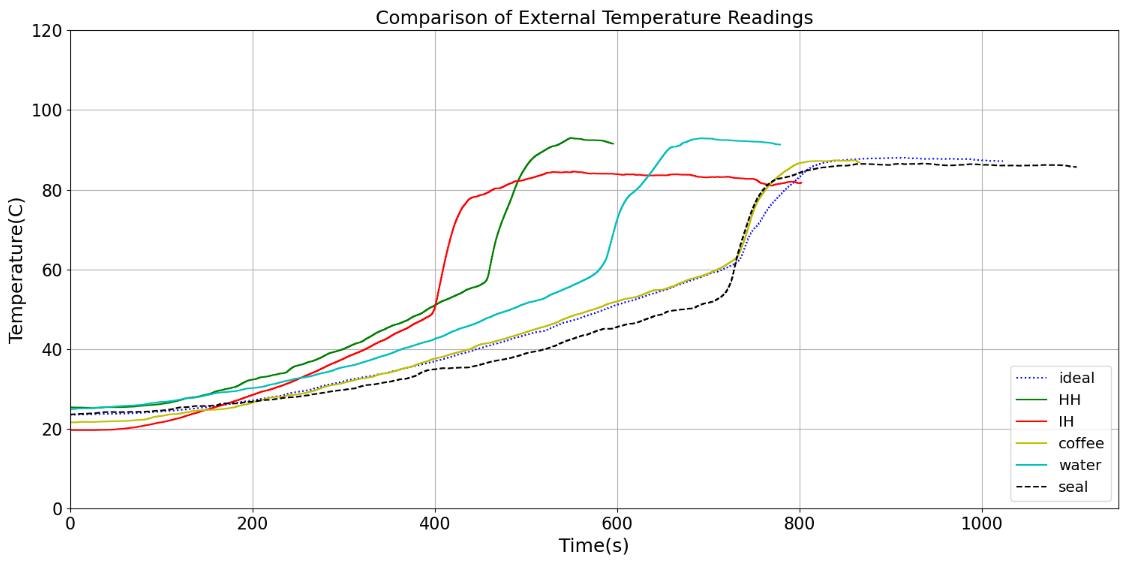


Figure 5.5: Comparison of external temperatures

3. From figure 5.4, we can see that the IH anomalous case reaches a high temperature very fast due to the high heat supply . But once it reaches there, the supply is inconsistent. Thus there are peaks and troughs in the pressure data as well as internal temperature data.

4. When the initial water quantity is less(water anomaly), the internal temperature (figure 5.4) increases rapidly, even though the heat supply used is same as the ideal case.

5. When less coffee is used(coffee anomaly), the process happens as if it was an ideal case till the brewing process starts. i.e the water reaches the coffee chamber. Since there are less coffee grounds to flow through, the water flows faster and the process gets over quickly.

6. When there is a leak in the pot(seal anomaly), the pressure curve is much flatter and the maximum pressure reached is quite low compared to other conditions. This can be seen in figure 5.6.

5.3 Anomaly Detection Methods

This section explains the anomaly detection methods that were applied on the dataset. The sensor anomaly data is not included in this analysis and is dealt with separately and is described in section 5.4. The methods are categorised based on the type of input data used (univariate or multivariate) and also if the labels are used in the training process(supervised and unsupervised). The methods applied are mentioned in table 5.2. To implement these methods standard libraries available in python like sklearn, sci-py were used.

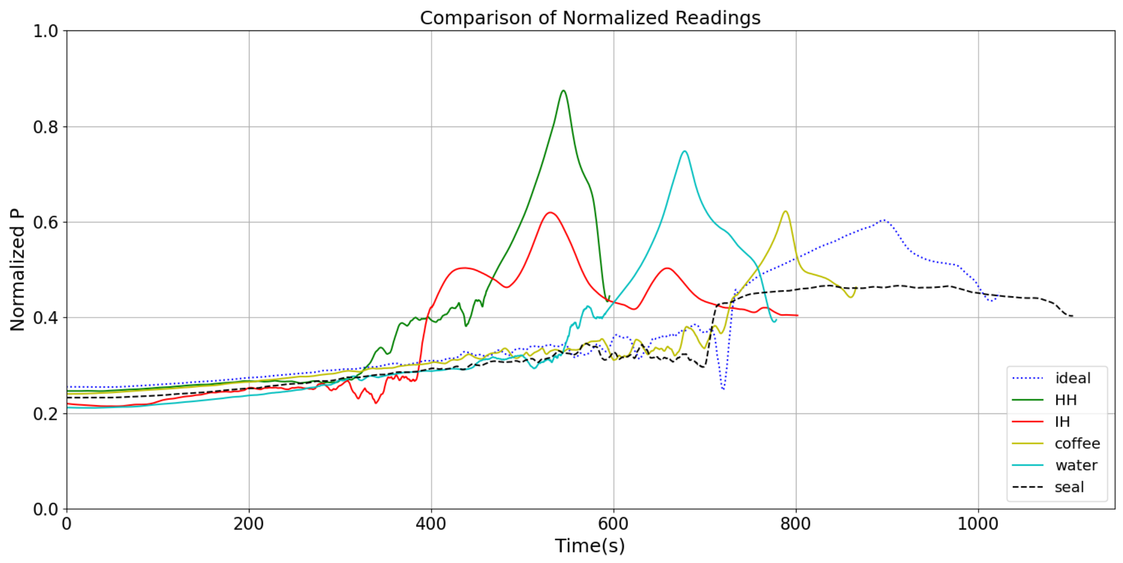


Figure 5.6: Comparison of normalized pressure data

	Univariate Data	Multivariate Data
Supervised Approaches	1-Nearest Neighbour	Ensemble 1-Nearest Neighbour SVM with Engineered Features
Unsupervised Approaches	K-means Clustering Hierarchical Clustering	

Table 5.2: The approaches tried for detecting anomalies

Keeping the same taxonomy mentioned in section 3.5, the dissimilarity based methods are explained first, and then the feature based method is described.

5.3.1 Clustering

1. **K-Means Clustering**

   - The K-Means clustering algorithm was applied on raw sensor data. Data from the pressure sensor was considered for analysis.
   - Data from shorter iterations were appended with 0s to make all time series into equal length.
   - Euclidean distance measure was used for the clustering algorithm.
   - To just detect the anomalies, 2 cluster centers were used to cluster the data. To detect which anomaly had occurred, 6 clusters were used to cluster the data. This was done to

test whether the data from 5 types of anomalies under consideration and data from ideal iterations would form their own clusters.

2. **Hierarchical Clustering**

   - The implementation from the sklearn library was used for analysis.
   - Input data is same as for K-means algorithm. i.e the pressure data was considered from each iteration and data from shorter iterations are appended with 0s to make all data vectors have equal length.
   - Ward linkage method and euclidean distance measure was used for clustering
   - Scipy library was used for constructing a dendrogram to represent the hierarchy of naturally occurring clusters in the dataset.

5.3.2 Nearest Neighbour Classifier

1. **Nearest Neighbour with Euclidean Distance Measure**

   - A nearest neighbour classifier was trained and tested independently on raw data from each sensor. Thus 10 nearest neighbour classifiers were trained.
   - Data was scaled to have zero mean and unit standard deviation.
   - An ensemble of all the univariate classifiers was implemented as mentioned in the Figure 3.11.

2. **Nearest Neighbour with DTW Distance Measure**

   - As mentioned before, the time and memory complexity of calculating DTW distance for long time series data is very high. Thus, the data was downsampled by a factor of 30 before applying the algorithm to calculate the distances.
   - A faster and more approximate approach called FastDTW was used instead of the classical DTW algorithm.
   - To implement the ensemble, univariate classifiers were trained for the pressure data, data from one internal and one external temperature sensor. Then a majority voting between the three classifiers decided the final decision of the combined classifier.

5.3.3 SVM Classifier

To test a feature based classifier on the dataset, features were carefully selected after data visualization and also by using some of the intuition gained during the data generation process. Figure 5.5 shows that the ideal and anomalous iterations vary greatly with respect to the process duration. The maximum temperatures and pressure reached during the process, could also provide some interesting information. After some considerations, the following features were selected and are explained using the figure 5.7.

- **Total Duration** - The total time each iteration takes, is one of the important differentiating factors between the ideal and anomalous iterations.
- **Brew start time** - The time at which the water enters the coffee grains is recorded as the brew start time . We can observe that there is a significant rise in temperature measured by ET6 at this point. To extract this feature, the data from ET6 was differentiated and a threshold was applied for the increase in slope. The first point where a sudden significant increase in slope is detected in the ET6 data is considered to be the brew start point.
- **Brew Duration** - The time duration from the brew start time till the end of iteration. As mentioned before in the observations from the data visualisation, the coffee iteration differs from ideal iteration only after the brewing starts. Thus, this feature was used in the classifier.
- **Maximum Pressure** - The maximum pressure reached in the iteration.
- **Maximum Internal Temperature** - The maximum temperature inside the boiling chamber for each iteration.

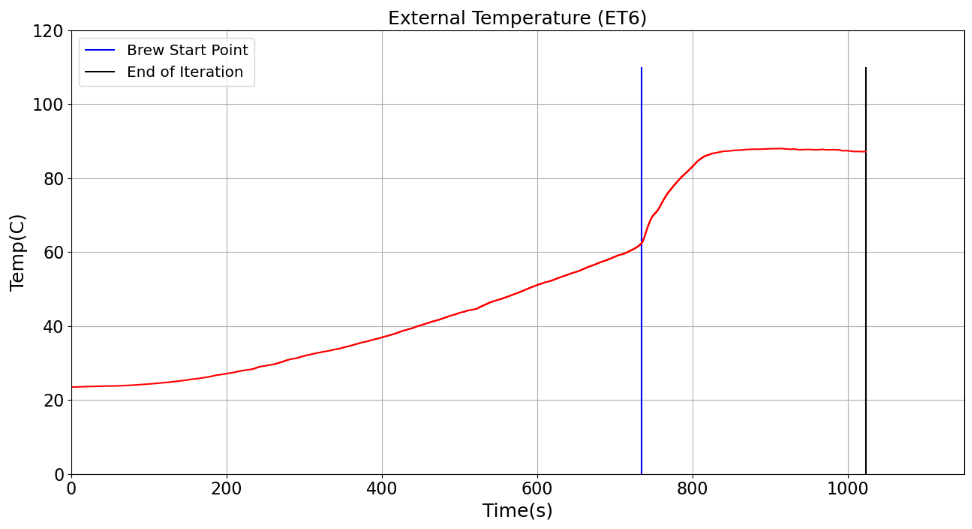


Figure 5.7: Illustration of Brew start and end points

Since the generated dataset is unbalanced with 40 ideal iterations and 24 anomalous iterations(just 20 excluding the sensor faults), higher weights were assigned to anomalous classes to give more importance to classify the anomalous data correctly.

5.4 Detection of Sensor Based Anomalies

Sensor anomalies are those which arise from faults in the sensors or the data acquisition system itself. It is quite challenging to detect these faults as the anomaly detection algorithms rely on the data from the sensors. The sensor anomaly data was excluded from the previous analysis because of the following reasons.

- The sensor faults are injected only in the external thermocouples. Moreover, it is not present in all the senors in a given iteration. Thus, when performing univariate analysis on any sensor other than the faulty sensor, it is impossible to detect such anomalies.

- The feature based classifier depends on the sensor data to be reliable for extracting the brew start time and duration. If faults exist in the data acquisition system, this will severely affect the performance of the algorithm.

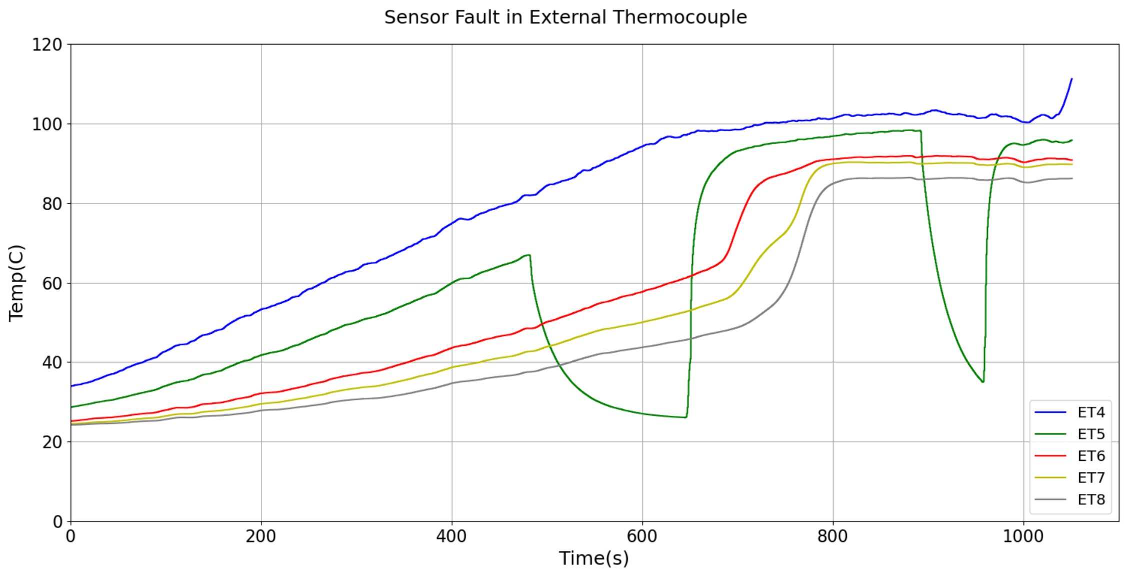


Figure 5.8: Anomalous data due to fault in sensor ET 5

The figure 5.8 shows a plot of the temperature readings of an iteration where faults were introduced in sensor ET 5. It can be seen that such faults may occur at any given point of time and need not have a particular template.
There are four iterations of sensor anomaly data and the anomalies exist only in external thermocouples. Thus, to test a simple detection method, data from the external thermocouples from these iterations were used. Since there are five external thermocouples in the setup, there are 20 univariate time series being considered. Other than the sensor faults, these iterations were performed under ideal conditions. Thus, the detection problem is reduced to a simple binary classification problem. The data from faulty sensors are considered to be anomalous data and the data from other sensors

are considered as ideal.

A simple observation was used to make a detection system for the sensor anomalies. The temperature data from the process in an ideal iteration always has a positive slope and tends to increase. Only when the sensor has a fault, there is significant negative slope in the regions where there is no contact between the thermocouple and the moka pot. This observation was tested and the accuracy of such a simple system was 100% with no false positives or false negatives. But this score is not a good judge of the detection system, since the problem was reduced to a very simple binary classification problem. Due to the lack of time, only a primitive detection system has been tried to detect sensor based anomalies in the generated dataset. More robust methods need to be applied to detect such anomalies along with the anomalies caused by the process faults. This is mentioned as one of the possible future work.

Chapter 6

Results and Discussion

6.1 Performance of Anomaly Detection Methods

This section presents the performance of anomaly detection algorithms tested on the dataset. The leave one out cross validation method was used for evaluating the accuracy score and f1 score of the predictions. In the case of unsupervised methods, the labels were used just for evaluation purposes and not for training the data. There were two types of evaluation undertaken for each method. The first approach is where the algorithm had to identify which anomaly is being detected. i.e each anomaly had its own label. This score is denoted as Accuracy(0-5). The second approach was just to test its anomaly detection performance where all the anomalies were provided the same label. Since it just contains a binary label for data, the accuracy calculated with this approach is denoted as Accuracy(1/0). F1 score is a better performance measure for the anomaly detection system as the generated dataset is imbalanced. The F1 score of each detection method is also included in the table 6.1 and is denoted by F1 Score(1/0).

Method	Accuracy(0-5)	Accuracy(1/0)	F1 Score(1/0)
NN-Euclidean Distance	0.7-0.8	0.73-0.83	0.6-73
NN-DTW Distance	0.55-0.7	0.58-0.73	0.41-0.64
Stacked NN-Euclidean Distance	0.83	0.85	0.756
Stacked NN-DTW Distance	0.73	0.72	0.56
SVM Classifier	0.87	0.88	0.8
K-Means Clustering	0.1	0.85	0.77
Hierarchical Clustering	0.05	0.87	0.79

Table 6.1: Accuracy scores and F1 scores of anomaly detection algorithms

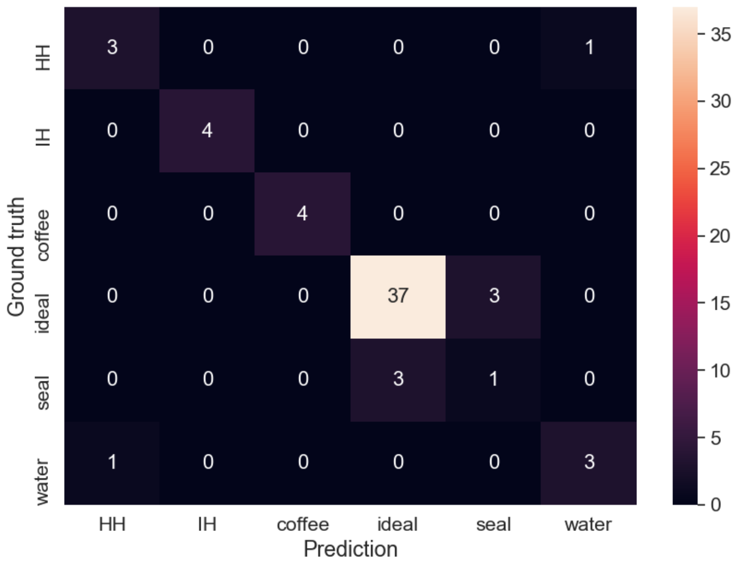


Figure 6.1: Confusion Matrix of SVM classifier output

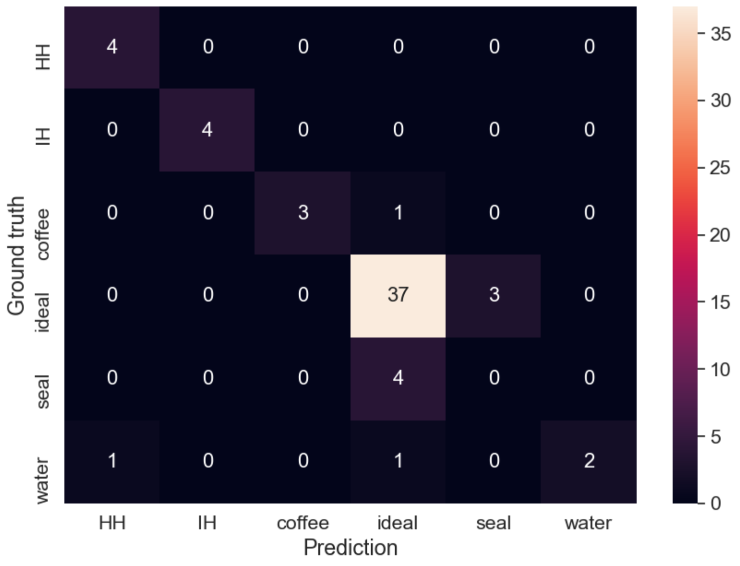


Figure 6.2: Confusion Matrix of Stacked NN-Euclidean Distance classifier output

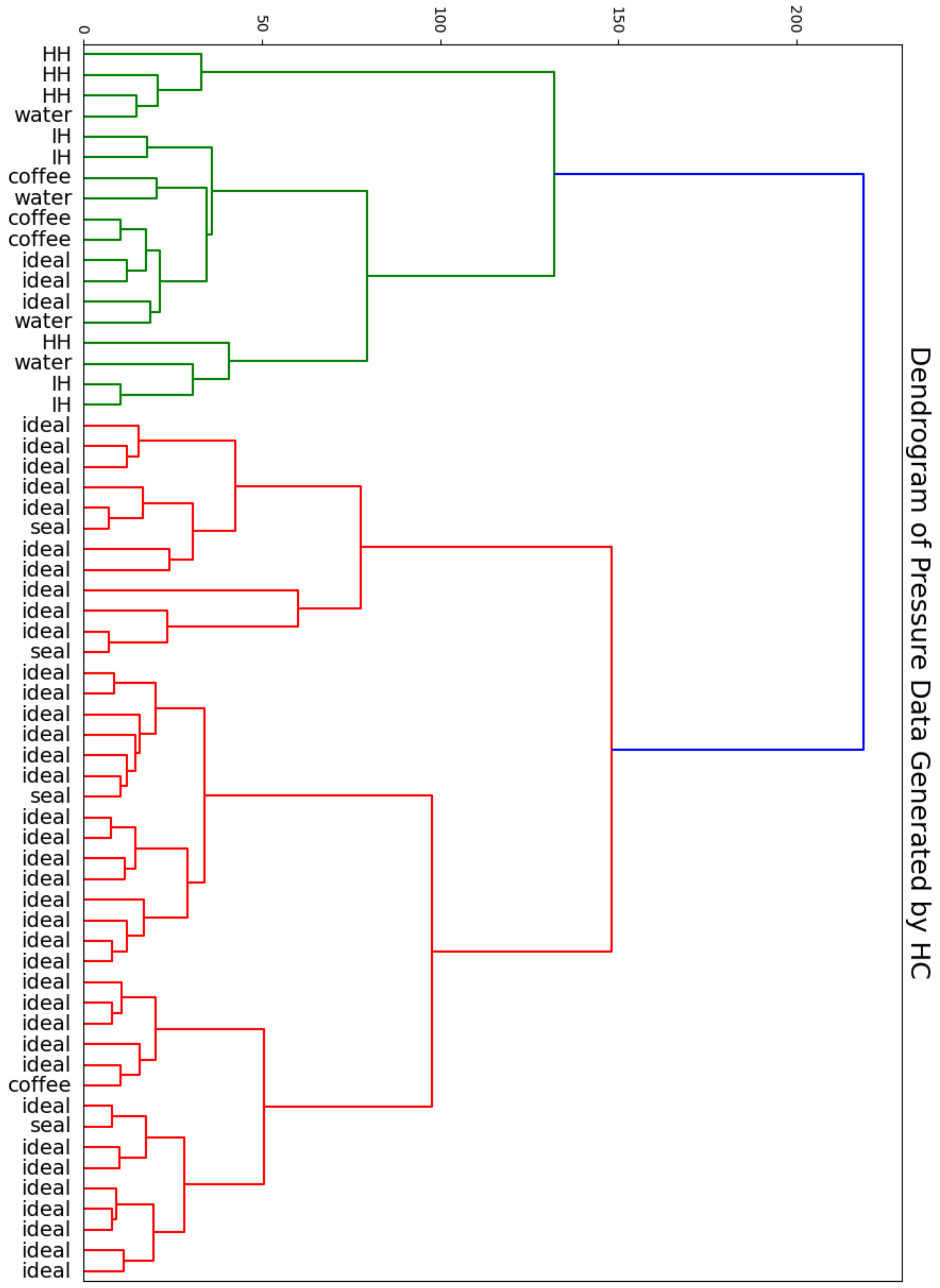


Figure 6.3: Dendrogram of data from pressure sensor

6.2 Discussion

Detection of anomalies in the coffee preparation process was achieved with a maximum accuracy score of 0.88. The SVM classifier with an input of engineered features was found to be the most suitable classifier to detect anomalies. The accuracy scores and the F1 scores of the NN-Euclidean distance and NN-DTW distance classifiers are given as a range because they represent scores from univariate classifiers trained for each sensor data.

The accuracy scores of the clustering methods are highly competitive in case of detecting the anomalies, but when it comes to identifying the anomalies, they perform very poorly. The reason for this poor performance can be seen in the dendrogram of the data shown in figure 6.3. The anomalous iterations are similar to each other , and thus it was possible to get a good anomaly detection accuracy with simple clustering algorithms. But it is clear from the dendrogram that it is difficult to identify which anomaly had occurred in the process.

The SVM classifier performs the best even when the F1 score is used as the performance measure. This is representative of the fact that the classifier has a decent precision score and a recall score. Its predictions have less false positives and false negatives when compared to other methods.

Another interesting observation is that the classifiers employing euclidean distance measure outperform the classifiers employing dynamic time warping distance measure. This could be explained by the fact that in the generated dataset, the ideal iterations and anomalous iterations produce data of similar shape. This can be seen in figures 5.5, 5.4 and 5.6. The main difference between the ideal and anomalous data lies in the duration of the process(the time axis). The dynamic time warping algorithm minimises the penalty which occurs due to small phase shifts in data. This causes the separation between ideal and anomalous readings even lesser, reducing the accuracy of predictions of the methods employing the DTW distance. Thus, the euclidean distance measure is more suitable in this case.

Considering the nearest neighbour classifiers, the multivariate classification achieved by stacking the univariate classifiers outperforms each individual univariate classifier.

The confusion matrix for the two best performing classifiers, the SVM classifier and the stacked nearest neighbour classifier, have been presented in figure 6.1 and figure 6.2 respectively. It can be seen that the biggest weakness of both the classifiers is their inability to identify the seal anomaly data. The SVM classifier identifies only one instance and the nearest neighbour classifier identifies none, and in both the cases there have been three false positives for the seal anomaly where an ideal case has been wrongly classified as seal anomaly data. To generate the seal anomaly data, the top chamber was not screwed on to the bottom chamber tightly, to create some leak. It is hard to control the amount of leak and the behaviour produced by the moka is found to be different in each iteration. This makes it harder for the classifier to identify the seal anomaly. This is also evident in the dendrogram of the data 6.3 where the seal anomaly is found very similar to the ideal cases.

The application of machine learning methods on the generated dataset and their performance to detect physical faults in the process, demonstrates the validity of the generated dataset to be used in building a condition monitoring system for the moka.
The data is generated from sensors on the physical entity i.e the moka. Analysis is performed in the digital domain to gain insights on the condition of the moka. This data driven condition monitoring system can be seen as a part of a digital twin implementation of the moka pot.

Chapter 7

Conclusion and Future Work

7.1 Summary

The main goal of this master thesis was to demonstrate the application of digital twin technology for a moka pot. This was achieved by implementing a data driven anomaly detection system for the coffee preparation process using the moka. The tasks that were involved to complete this project were the setting up a data acquisition system, generation of the required dataset and application of machine learning methods for anomaly detection.

Data acquisition devices from National Instruments were interfaced with the sensors and a Labview virtual instrument was designed to handle data from the data acquisition hardware. After defining the conditions for ideal and faulty conditions of the brewing process, the data acquisition system was used to generate a relevant dataset which represents both ideal and anomalous iterations of coffee preparation.

Both supervised and unsupervised machine learning algorithms were applied on the dataset to detect anomalies and a comparison was made with respect to their performance. By performing this analysis, the utility of the generated dataset has been proved. An SVM classifier which was fed with some engineered features performed the best among the methods that were tried.

7.2 Future Work

This master thesis can be a foundational study for application of data driven methods on the moka. It is focussed on finding anomalies in the coffee making process, but the study does not include the effect of parameters like water quantity, coffee quantity and the heat supply on the quality of the coffee produced. Due to the challenging circumstances, a refractometer could not be procured for

this study. A refractometer measures the total dissolved solids in the coffee. Interesting inferences can be made on the coffee with this measure and is recommended to be included in future studies on the moka. A regression model to predict the total dissolved solids in coffee with the physical parameters of the iteration as the inputs, could be an interesting topic of study.
The implementation of digital twins vary greatly and is mainly dependent on the application. In this study, an offline anomaly detection system was sufficient as real time feedback was not implemented. A study with an online condition monitoring system with real time feedback to make changes during the process is also a possibility.
The detection of sensor faults in this study is done separately and the method used is primitive. A more robust algorithm which would detect the sensor faults along with the other faults would be interesting. Algorithms for early detection of anomalies can also be tested on the dataset and this study could lead interesting results for prediction of failure in the process.
One of the biggest limitations of this study is the improper scaling of the pressure sensor data. With the correct values of pressure, a study can be performed which analyses the risks involved in the process. The boiling chamber in the moka experiences 1-2 bars of pressure according to the experiments carried out by [5]. The safety valve on the moka ensures that an additional pressure in the moka is released to avoid any accidents. But an improper functioning valve could lead to accidents. Successfully predicting the failure of the valve could be an interesting topic of study.

Bibliography

[1] Bialetti industries, moka express. [online]. `https://www.bialetti.us/coffee/stovetop/moka-express-c-1_7_22.html`.

[2] Siddanth Nayak Bairampalli. Moka digital twin-a small scale study for process engineering applications. Unpublished Work, December 2019.

[3] Concetto Gianino. Experimental analysis of the italian coffee pot 'moka'. American Journal of Physics - AMER J PHYS, 75:43–47, 01 2007.

[4] Nancy Cordoba, Mario Fernandez-Alduenda, Fabian L. Moreno, and Yolanda. Ruiz. Coffee extraction: A review of parameters and their influence on the physicochemical characteristics and flavour of coffee brews. Trends in Food Science Technology, 96:45 – 60, 2020.

[5] L. Navarini, E. Nobile, F. Pinto, A. Scheri, and F. Suggi-Liverani. Experimental investigation of steam pressure coffee extraction in a stove-top coffee maker. Applied Thermal Engineering, 29(5):998 – 1004, 2009.

[6] M. Baygin, H. Yetis, M. Karakose, and E. Akin. An effect analysis of industry 4.0 to higher education. In 2016 15th International Conference on Information Technology Based Higher Education and Training (ITHET), pages 1–4, Sep. 2016.

[7] Michael Grieves and John Vickers. Digital Twin: Mitigating Unpredictable, Undesirable Emergent Behavior in Complex Systems, pages 85–113. In: Kahlen FJ., Flumerfelt S., Alves A. (eds) Transdisciplinary Perspectives on Complex Systems, Springer International Publishing, Cham, 2017.

[8] J. Vickers D. Lowry S. Scotti J. Stewart. Piascik, R. and A. Calomino. Technology area 12:materials, structures, mechanical systems, and manufacturing road map, nasa office of chief technologist., 2010.

[9] Subrata Bhowmik. OTC-29455-MS, chapter Digital Twin of Subsea Pipelines: Conceptual Design Integrating IoT, Machine Learning and Data Analytics, page 9. Offshore Technology Conference, Houston, Texas, 2019.

[10] Y. Liu, L. Zhang, Y. Yang, L. Zhou, L. Ren, F. Wang, R. Liu, Z. Pang, and M. J. Deen. A novel cloud-based framework for the elderly healthcare services using digital twin. IEEE Access, 7:49088–49101, 2019.

[11] Parrott and Warshaw. Industry 4.0 and the digital twin. 2017. Retrieved from: `https://www2.deloitte.com/cn/en/pages/consumer-industrial-products/articles/industry-4-0-and-the-digital-twin.html`.

[12] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. Deep learning for time series classification: a review. Data Mining and Knowledge Discovery, 33(4):917–963, Jul 2019.

[13] Ane Blázquez-García, Angel Conde, Usue Mori, and Jose A. Lozano. A review on outlier/anomaly detection in time series data, 2020.

[14] Meinard Müller. Dynamic time warping. Information Retrieval for Music and Motion, Volume 2, pages 69–84, 01 2007.

[15] Stan Salvador and Philip Chan. Fastdtw: Toward accurate dynamic time warping in linear time and space. Workshop on Mining Temporal and Sequential Data, page 11, 2004.

[16] Ali Javed, Byung Suk Lee, and Dona M. Rizzo. A benchmark study on time series clustering, 2020.

[17] Anthony Bagnall, Jason Lines, Aaron Bostrom, James Large, and Eamonn Keogh. The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. Data Mining and Knowledge Discovery, 31(3):606–660, May 2017.

[18] Nagaraj N. Bhat, Samik Dutta, Tarun Vashisth, Srikanta Pal, Surjya K. Pal, and Ranjan Sen. Tool condition monitoring by svm classification of machined surface images in turning. The International Journal of Advanced Manufacturing Technology, 83(9):1487–1502, Apr 2016.

[19] Carlos Alonso, Óscar Prieto, Juan José Rodríguez, and Aníbal Bregón. Multivariate Time Series Classification via Stacking of Univariate Classifiers, pages 135–151. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[20] Cold junction compensation in thermocouples. [online]. `https://www.maximintegrated.com/en/design/technical-documents/app-notes/4/4026.html`.

[21] Datasheet of ni-9213 module. `http://www.ni.com/pdf/manuals/374916a_02.pdf`.

[22] Kulite pressure transducer. `https://kulite.com//assets/media/2018/01/XTME-190.pdf`.

[23] Datasheet of ni-9237 module. `http://www.ni.com/pdf/manuals/374186a_02.pdf`.

Appendix A

Python Code

This chapter contains all the python code used in the thesis. It includes scripts for plotting data and also python code of the machine learning methods applied on the dataset. All the libraries used in codes are listed below.

```python
import numpy as np
import pandas as pd
import seaborn as sn
import matplotlib.pyplot as plt
from numpy import genfromtxt
from datetime import datetime
from sklearn.preprocessing import StandardScaler
from scipy.spatial import distance
from scipy import signal
from fastdtw import fastdtw
from sklearn.neighbors import KNeighborsClassifier
from sklearn.cluster import KMeans
import scipy.cluster.hierarchy as shc
from sklearn.cluster import AgglomerativeClustering
from sklearn.svm import SVC
from sklearn.model_selection import LeaveOneOut
from sklearn.metrics import accuracy_score, f1_score
from sklearn.metrics import confusion_matrix
```

Listing A.1: Libraries imported across all codes

The following function plots the multivariate time series data of the required iteration.

```python
def plot_temp(name_of_csv):
    data = genfromtxt(name_of_csv, delimiter=',')
    data = data[1:, :]
    data = np.array(data)
    datat = genfromtxt(name_of_csv, delimiter=',', dtype='unicode')
    timestamps = []
    datat = datat[1:, :]
    datat = np.array(datat)
    form = "%H:%M:%S.%f"
    for i in datat[:, 0]:
       timestamp = i
       timestamps.append(datetime.strptime(timestamp, form))
    times = []
    for i in timestamps:
       time = i-timestamps[0]
       time = time.total_seconds()
       times.append(time)

    times = np.array(times)
    print(times[0], times[1])
    print(len(timestamps))

    plt.figure()
    plt.suptitle('Data from '+name_of_csv[25:-4], fontsize=14)
    axes = plt.gca()
    axes.set_xlim([0, 1300])
    axes.set_ylim([0, 150])
    plt.grid()
    plt.subplot(311)
    plt.title("Internal Temperatures", fontsize=14)
    plt.ylabel("Temp(C)", fontsize = 14)
    plt.xlabel("Time(s)", fontsize=14)
    axes.set_xlim([0, 3000])
    axes.set_ylim([0, 110])
    line1, = plt.plot(times, data[:, 1], 'b')
    line2, = plt.plot(times, data[:, 2], 'g')
    line3, = plt.plot(times, data[:, 3], 'r')
    line4, = plt.plot(times, data[:, 4], 'y')
    plt.legend([line1, line2, line3, line4],
               ['IT0', 'IT1', 'IT2', 'IT3'], loc=4, fontsize='large')
    plt.subplot(312)
    plt.title("External Temperatures", fontsize=14)
    plt.ylabel("Temp(C)", fontsize=14)
    plt.xlabel("Time(s)", fontsize=14)
    axes.set_xlim([0, 3000])
    axes.set_ylim([0, 110])
    line5, = plt.plot(times, data[:, 5], 'b')
    line6, = plt.plot(times, data[:, 6], 'g')
    line7, = plt.plot(times, data[:, 7], 'r')
    line8, = plt.plot(times, data[:, 8], 'y')
    line9, = plt.plot(times, data[:, 9], 'grey')
    plt.legend([line5, line6, line7, line8, line9],
    ['ET4', 'ET5', 'ET6', 'ET7', 'ET8'], loc=4, fontsize='large')
    plt.subplot(313)
    plt.title('Internal Pressure', fontsize=14)
    plt.ylabel("Normalised P", fontsize=14)
    plt.xlabel("Time(s)", fontsize=14)
    axes.set_xlim([0, 3000])
    axes.set_ylim([0, 110])
    line10, = plt.plot(times, data[:, 10], 'r')
    plt.subplots_adjust(hspace=0.45, top=0.92, right=0.98, bottom=0.07, left=0.07)
    plt.legend([line10], ['Normalised P'], loc=4, fontsize='large')
```

Listing A.2: Code to plot data from a single iteration of coffee preparation

For some of the univariate analysis performed on the dataset, it was required to extract data of a single sensor from the whole dataset. This script also appends the shorter time series with 0s and saves the extracted data in a CSV file. Each row in the file will have data from a single iteration of the required sensor.

```
def extract_univariate_data(sensor_name):
    sensor_num = {'IT0': 1, 'IT1': 2, 'IT2': 3, 'IT3': 4, 'ET4': 5, 'ET5': 6, 'ET6':
    7, 'ET7': 8, 'ET8': 9, 'P': 10}
    X_train = genfromtxt('final_dataset/Moka_iteration_0.csv', delimiter=',')
    X_train = X_train[1:, sensor_num[sensor_name]]
    leng = X_train.shape[0]
    for i in range(1, 64):
        X = np.zeros(leng)
        s = 'final_dataset/Moka_iteration_' + str(i) + '.csv'
        data = genfromtxt(s, delimiter=',')
        data = data[1:, sensor_num[sensor_name]]
        X[:data.shape[0]] = data
        X_train = np.vstack((X_train, X))
        print(i)
    print(X_train.shape)
    file_name = "univariate_dataset/unscaled" + sensor_name + ".csv"
    np.savetxt(file_name, X_train, delimiter=",")
```

Listing A.3: Isolates univariate data of the required sensor from the whole dataset and stores in a new file

The following code is a primitive detection mechanism for sensor faults

```
detection_labels = np.array([])
for i in range(56,60):
    X_data = genfromtxt('final_dataset/moka_iteration_'+str(i)+'.csv', delimiter=',')
    X_data = X_data[1:, 5:10]
```

```python

    for j in range(0,5):
        slope = np.diff(X_data[:,j])
        y = np.array(np.where(slope<(-0.5)))
        if y.size == 0:
            detection_labels = np.append(detection_labels, 0)
        else:
            detection_labels = np.append(detection_labels, 1)
print(detection_labels)
accu = accuracy_score(detection_labels ,labels)
f1 = f1_score(detection_labels ,labels)
print("accuracy=",accu ,"F1score=",f1)
```

Listing A.4: Simple sensor fault detection mechanism

The following code snippet generates the dendrogram showed in figure 6.3

```python
X_data_2 = genfromtxt('univariate_dataset/P.csv', delimiter=',')
X_data_2 = X_data_2[:60, :]
print(X_data_2.shape)
final_label = genfromtxt('final_dataset_rescaled_2/final_label_2.csv', delimiter=',',
     dtype='unicode')
final_label_without_sensor = (final_label[1:61, :])
print(final_label_without_sensor.shape)
plt.figure()
plt.subplots_adjust(hspace=0.45, top=0.92, right=0.98, bottom=0.09, left=0.07)
plt.title("Dendrogram of Pressure Data Generated by HC", fontsize=18)
dend = shc.dendrogram(shc.linkage(X_data_2, method='ward'), leaf_font_size=14, labels
    =final_label_without_sensor[:, -1])
plt.show()
```

Listing A.5: Generate dendrogram of data

The following code snippets are the implementations of classifiers and the clustering methods mentioned in the report.

```python
def euclidean_nearest_neighbour(s):
    X_data = genfromtxt(s, delimiter=',')
    X_data = X_data[:60,:]
    loo = LeaveOneOut()
    loo.get_n_splits(X_data)
    neigh = KNeighborsClassifier(n_neighbors=1)
    accuracy=np.array([])
    predictions = np.array([])
    test_labels = np.array([])
    for train_index, test_index in loo.split(X_data):
    #Split train data and test data to have one test sample
        X_train, X_test = X_data[train_index], X_data[test_index]
        y_train, y_test = Y_bin[train_index], Y_bin[test_index]
        #y_train, y_test = Y[train_index], Y[test_index]
         #Scaling done one only on train data
        trained_scaler = StandardScaler().fit(X_train)
        # Trained scaler is applied on test data point as well
        X_train = trained_scaler.transform(X_train)
        X_test = trained_scaler.transform(X_test)
        neigh.fit(X_train, y_train)
        y_pred = neigh.predict(X_test)
        predictions = np.append(predictions, y_pred)
        test_labels = np.append(test_labels, y_test)
    mean_accuracy = accuracy_score(test_labels, predictions)
    f1 = f1_score(test_labels, predictions)
    print("Accuracy =", mean_accuracy, "F1=", f1)
    return predictions.astype(int), mean_accuracy
#The following code is to stack the predictions of univariate classifiers and perform majority voting

all_pred = np.vstack((Pred_P, Pred_IT0,Pred_IT1,Pred_IT2,Pred_IT3,Pred_ET4
          ,Pred_ET5,Pred_ET6,Pred_ET7,Pred_ET8))
y_ensemble = np.array([])
for i in range (0,60):
    counts = np.bincount(all_pred[:,i])
    y_ensemble = np.append(y_ensemble, (np.argmax(counts)))
print("Accuracy of Ensemble = ", accuracy_score(Y_bin, y_ensemble))
print("F1 of Ensemble = ", f1_score(Y_bin,y_ensemble))
```

Listing A.6: Euclidean distance based NN classifier

```
Distances = np.zeros((64,64))
for i in range(0,63):
    X_data_1 = genfromtxt('final_dataset/moka_iteration_'+str(i)+'.csv', delimiter=',')
    X_data_1 = X_data_1[1:, 7]
    X_data_1 = signal.decimate(X_data_1, q=30)
    print(i)
    for j in range(i, 64):
        X_data_2 = genfromtxt('final_dataset/moka_iteration_'+str(j)+'.csv')
        X_data_2 = X_data_2[1:, 7]
        X_data_2 = signal.decimate(X_data_2, q=30)
        distance, path = fastdtw(X_data_1, X_data_2, dist=euclidean)
        Distances[i][j] = distance
        Distances[j][i] = distance
np.savetxt("Fast_DTW_distances_ET_6_final.txt", Distances, delimiter = ',')

def find_neighbours_DTW(name_distance_file, Y):
    Y_pred = np.array([])
    distance_matrix = np.genfromtxt(name_distance_file, delimiter = ',')
    distance_matrix = np.vstack((distance_matrix[:60, :60]))
    nearest_neighbours = np.array([])
    for i in range (0,60):
        data = distance_matrix[i, :]
        #Find nearest element with non zero distance
        neigh = np.where( data==np.min(data[np.nonzero(data)]))
        nearest_neighbours = np.append(nearest_neighbours, neigh)
    for i in nearest_neighbours.astype(int):
        Y_pred = np.append(Y_pred, Y[i])
    return Y_pred.astype(int), accuracy_score(Y, Y_pred)

# Stacking the decisions and performing majority voting
all_pred = np.vstack((Y_pred_P, Y_pred_IT_0, Y_pred_ET_6))
y_ensemble = np.array([])
for i in range (0,60):
    counts = np.bincount(all_pred[:,i])
    y_ensemble = np.append(y_ensemble, (np.argmax(counts)))
print("Accuracy of Ensemble = ", accuracy_score(y_ensemble, Y_bin))
print("F1 score of Ensemble = ", f1_score(y_ensemble, Y_bin))
```

Listing A.7: DTW distance based NN classifier

```
# Snippet for K-means Clustering of Data

loo = LeaveOneOut()
loo.get_n_splits(X_data)
predictions = np.array([])
test_labels = np.array([])
for train_index, test_index in loo.split(X_data):
    X_train, X_test = X_data[train_index], X_data[test_index]
    y_train, y_test = Y[train_index], Y[test_index]
    kmeans = KMeans(n_clusters=2, random_state=1).fit(X_data)
    y_pred = kmeans.predict(X_test)
    predictions = np.append(predictions, y_pred)
    test_labels = np.append(test_labels, y_test)
    #print(y_test, y_pred)
mean_accuracy = accuracy_score(test_labels, predictions)
f1 = f1_score(test_labels, predictions)
print("Accuracy =", mean_accuracy, "F1=", f1)

# Snippet for Hierarchical Clustering of Data
cluster = AgglomerativeClustering(n_clusters=2, affinity='euclidean', linkage='ward')
y_hc_pred = cluster.fit_predict(X_data_2)
print(y_hc_pred)
```

```python
accu = accuracy_score(y_hc_pred, labels_4)
f1 = f1_score(y_hc_pred, labels_4)
print("Accuracy =", accu,"F1=", f1)
```

Listing A.8: KMeans and Hierarchical Clustering

```python
# Code for Feature Extraction
feature_array = np.array(["Iteration","Total Time(s)","Brew Start Time(s)","Brew
    Duration(s)","Max Pressure","Max Int Temp"])
for i in range(0, 60):
    s = 'final_dataset_rescaled_2/Moka_iteration_' + str(i) + '.csv'
    data = np.genfromtxt(s, delimiter = ",", dtype = "unicode")
    tot_time = (data.shape[0]-1)/50
    ET_6 = data[1:,7]
    ET_6 = ET_6.astype(float)
    ET_6 = np.diff(ET_6)
    y = np.where(ET_6>0.15)
    brew_start = np.array(y)[0][0]/50
    P = data[1:,10]
    P = P.astype(float)
    max_p = np.max(P)
    IT_1 = data[1:,2]
    IT_1 = IT_1.astype(float)
    max_it1 = np.max(IT_1)
    brew_duration = tot_time - brew_start
    feature_array = np.vstack((feature_array, np.array([s[25:-4], tot_time,
    brew_start, brew_duration, max_p, max_it1])))
```

Listing A.9: Feature extraction

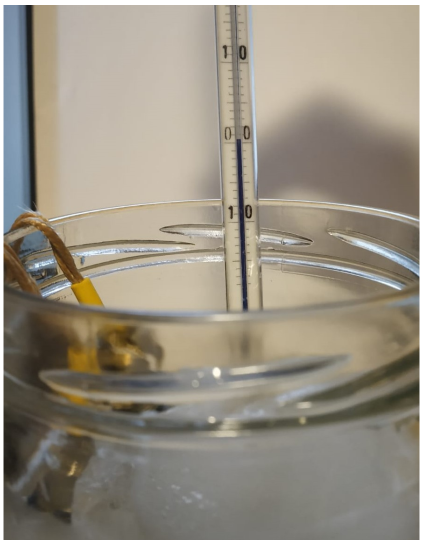
```python
# Snippet for SVM classifier

loo = LeaveOneOut()
loo.get_n_splits(X_feat)
#weights_class = {0: 10, 1: 10, 2: 10, 3: 2, 4: 10, 5: 10}
#clf = SVC(decision_function_shape='ovo', gamma = 'scale', class_weight=weights_class
    )
clf = SVC(decision_function_shape='ovo', gamma = 'scale')
accuracy=np.array([])
predictions = np.array([])
test_labels = np.array([])
for train_index, test_index in loo.split(X_feat):
    #print("TRAIN:", train_index, "TEST:", test_index)
    X_train, X_test = X_feat[train_index], X_feat[test_index]
    y_train, y_test = Y_bin[train_index], Y_bin[test_index]
    #y_train, y_test = Y[train_index], Y[test_index]
    trained_scaler = StandardScaler().fit(X_train)
    X_train_scaled = trained_scaler.transform(X_train)
    X_test_scaled = trained_scaler.transform(X_test)
    clf.fit(X_train_scaled, y_train)
    y_pred = clf.predict(X_test_scaled)
    #print(accuracy_score(y_test, y_pred))
    predictions = np.append(predictions, y_pred)
    test_labels = np.append(test_labels, y_test)
mean_accuracy = accuracy_score(test_labels, predictions)
f1 = f1_score(test_labels, predictions)
```

Listing A.10: SVM classifier

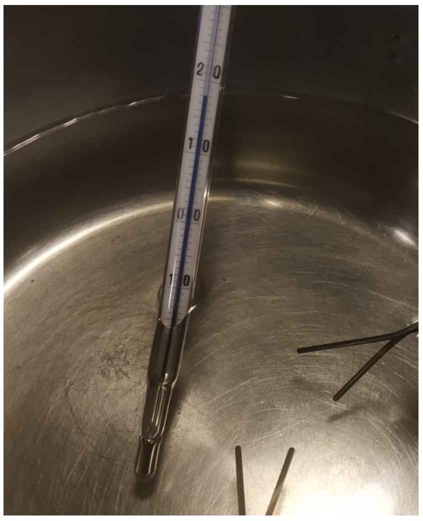
Appendix B

Temperature Measurement Tests

The measurement from internal thermocouples have been validated with a standard laboratory thermometer. The temperature measured by the thermometer and our thermocouples were found to be in agreement. The setup was tested in an ice bath, in water left at room temperature and in boiling water. The images of the thermometer reading are shown in figures B.1 and B.2 and the corresponding graphs of thermocouple measurement are shown in figure B.3.
In all the three cases, the difference in measurements is less than 1 degree Celsius. The thermometer reading in the ice bath, room temperature water and boiling water is between 0-1 degrees, 17-18 degrees and 98-99 degrees respectively.



(a)



(b)

Figure B.1: Thermometer readings -a) In ice bath b) In room temperature water

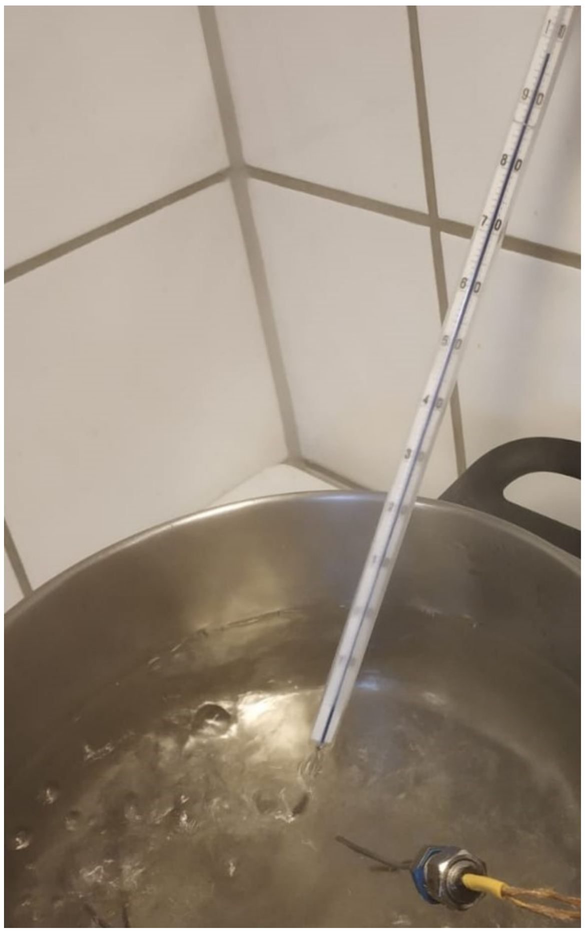
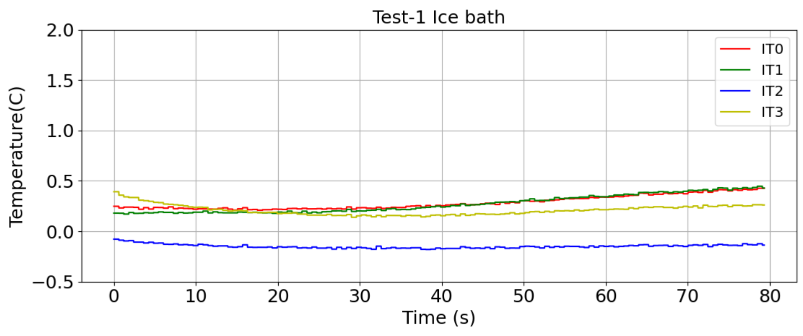
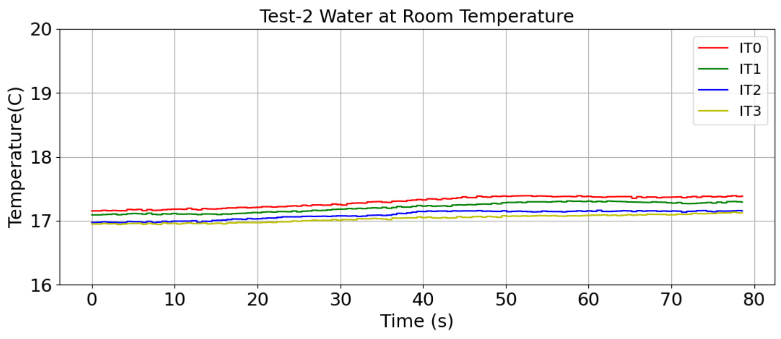


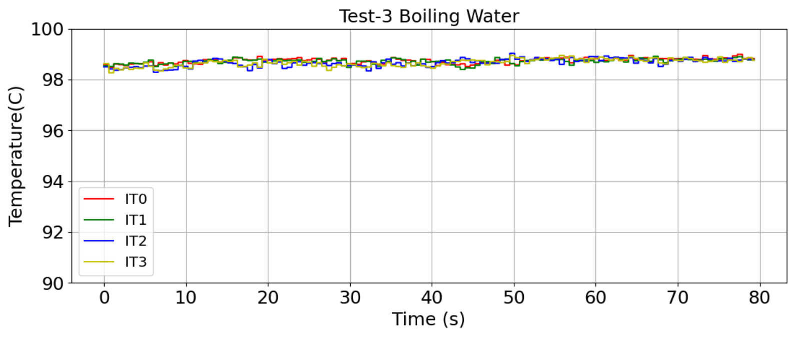
Figure B.2: Thermometer reading in boiling water



(a)



(b)



(c)

Figure B.3: Measurement of Thermocouples(a) In Ice bath (b) In room temperature water (c) In boiling water