university of groningen

faculty of science and engineering

AI For Predictive Maintenance For Injection Moulding Machines

Author

Zahra Putri Fitrianti

Student number: S2936240

Supervisors

Asst. Prof. dr. Kerstin Bunte

Prof. dr. Michael Biehl

Philips

September 3, 2020

Abstract

Predictive Maintenance is one of the utmost importance in the production process in industry. One of its examples is preventing a potential bad batch during production. Pertaining to the context of the data, a bad batch is typically indicated by triggering an alert or when an anomaly occurs in the data. In this thesis, we investigate the influence of parameter settings of six injection moulding machines to the potential alerts that are predicted by the models. We hypothesised that any changes in the parameter settings could lead to unusual activity on the machines which would potentially lead to alerts in the observed measurements.

Two approaches were considered, namely univariate and multivariate analysis. For univariate analysis, we developed ARIMA models for each of the selected features and we analyse the effect of each individual feature by forecasting future trends. These models were tested during the development phase and their performances were measured by calculating the Mean Absolute Percentage Error (MAPE). The results show that while the models were able to forecast future trends rather well, the inconsistencies in MAPE suggest that an alert is more likely to be caused by a combination of one feature and possibly others.

Subsequently, the results from multivariate analysis imply that the combination of PCA and ARIMA models helps us to better identify the potential cause of an alert. This was done by means of checking if the predicted trends consist of values which are either above or below certain thresholds and then looking at the absolute weights given by the sorted eigenvectors to determine which features have the most influence on the alerts. From the two approaches, one can conclude that the multivariate approach provides a more practical understanding of the cause of the detected or predicted (future) alerts yielded by the models.

Contents

1 Introduction **7**

2 State of the Art **8**

3 Methodology **11**
3.1 Filter method . . . 11
3.2 Time Series . . . 12
3.3 Stationarity . . . 14
3.4 ARIMA model . . . 15
3.4.1 Autoregressive Model . . . 15
3.4.2 Integrated Model . . . 16
3.4.3 Moving Average Model . . . 16
3.5 Principal Component Analysis . . . 17
3.6 Parameters tuning . . . 17
3.6.1 Auto Arima . . . 18
3.7 Evaluation Metrics . . . 18
3.7.1 Mean Absolute Percentage Error . . . 18
3.7.2 Mean Squared Error . . . 19

4 Experiments **20**
4.1 Data . . . 21
4.1.1 Preprocessing . . . 21
4.2 Features . . . 22
4.2.1 Correlation . . . 23
4.2.2 Variance Threshold . . . 24
4.2.3 Common features across machines . . . 24
4.3 Stationary Series . . . 25
4.4 Anomaly detection . . . 27

5 Results **28**
5.1 Univariate . . . 28
5.1.1 Models . . . 28
5.1.2 Predictions . . . 29
5.1.3 Performance . . . 33
5.2 Multivariate . . . 34
5.2.1 Models . . . 34
5.2.2 Predictions . . . 35
5.2.3 Performance . . . 35
5.2.4 Anomaly detection . . . 38
5.2.5 Held-out test data . . . 40

6 Discussion **43**
6.1 Limitations . . . 43
6.2 Generated prediction values . . . 44
6.3 Detected anomalies . . . 44
6.4 Answer to research question . . . 45

7 Conclusion **46**

References **47**

Appendix **48**

A Features **49**

B Univariate Predictions **58**
B.1 Machine by20 58
B.2 Machine by19 61
B.3 Machine 2406 64
B.4 Machine 1622 65
B.5 Machine 1805 67
B.6 Machine 2605 68

C Multivariate Results **70**
C.1 Predictions 70
C.2 Anomaly detection 71
C.3 Forecast on held out test data 73

List of Figures

1 Example time series . . . 12
2 Example of trend in a time series . . . 13
3 Example of cycle in a time series . . . 14
4 Experimental Setting . . . 20
5 Example of irregular values in clamp force parameter for BY machines . . . 22
6 Correlation matrix for machine 2406 . . . 23
7 Correlation matrix for machine by19 . . . 23
8 Boxplot of a feature across BY machines (*left*) and EN machines (*right*). . . . 25
9 Example of non-stationary series . . . 25
10 Example of stationary series . . . 26
11 Predictions made by the model for feature Temperature zone 8 for machine by20 . 29
12 Predictions made by the model for feature Temperature zone 8 for machine by19 . 30
13 Predictions made by the model for feature Injection pressure peak for machine 2406 30
14 Predictions made by the model for feature Cycle time last value for machine 1622 . 31
15 Predictions made by the model for feature Mold protection time actual value for machine 1805 . . . 31
16 Predictions made by the model for feature Heating energy actual value last cycle for machine 2605 . . . 32
17 Overview of the models' performances in terms of Mean Absolute Percentage Error (MAPE). . . . 33
18 Prediction for machine 2406 . . . 35
19 Prediction for machine by20 . . . 36
20 Prediction for machine by19 . . . 36
21 Prediction for machine 1805 . . . 37
22 Mean Squared Error of the predictions for all machines . . . 37
23 Detected anomalies for machine by19 for time series of flow number 1, hold pressure peak 1, injection pressure peak 1, mold protection time actual value, oil temperature, temperature zone 8, and clamp force from top to bottom panel, ordered by the eigenvectors, respectively. . . . 38
24 Detected anomalies for machine by20 for time series of flow number 1, mold protection time actual value, injection pressure peak 1, oil temperature, temperature zone 8, clamp force and hold pressure peak 1, from top to bottom panel, ordered by the eigenvectors, respectively. . . . 39
25 Forecast for machine by19 . . . 41
26 Forecast for machine 2406 . . . 41
27 Forecast for machine 2605 . . . 42
28 Correlation matrix for machine 1622 . . . 51
29 Correlation matrix for machine 1805 . . . 51
30 Correlation matrix for machine 2605 . . . 52
31 Correlation matrix for machine by20 . . . 52
32 Boxplot for Clamp force feature of BY machines . . . 53
33 Boxplot for Flow number feature of BY machines . . . 53
34 Boxplot for Hold pressure peak feature of BY machines . . . 54
35 Boxplot for Injection pressure peak feature of BY machines . . . 54
36 Boxplot for Mold protection time actual value feature of BY machines . . . 55
37 Boxplot for Oil temperature feature of BY machines . . . 55
38 Boxplot for Cycle time last value feature of EN machines . . . 56
39 Boxplot for Heating energy actual value last cycle feature of EN machines . . . 56

40 Boxplot for Mold protection time actual value feature of EN machines 57
41 Predictions made by the model for feature Clamp force for machine by20 58
42 Predictions made by the model for feature Flow number 1 for machine by20 58
43 Predictions made by the model for feature Hold pressure peak 1 for machine by20 59
44 Predictions made by the model for feature Injection pressure peak 1 for machine by20 59
45 Predictions made by the model for feature Mold protection time actual value for machine by20 60
46 Predictions made by the model for feature Oil temperature for machine by20 . . . 60
47 Predictions made by the model for feature Clamp force for machine by19 61
48 Predictions made by the model for feature Flow number 1 for machine by19 61
49 Predictions made by the model for feature Hold pressure peak 1 for machine by19 62
50 Predictions made by the model for feature Injection pressure peak 1 for machine by19 62
51 Predictions made by the model for feature Mold protection time actual value for machine by19 63
52 Predictions made by the model for feature Oil temperature for machine by19 . . . 63
53 Predictions made by the model for feature Heating energy actual value last cycle for machine 2406 64
54 Predictions made by the model for feature Cycle time last value for machine 2406 . 64
55 Predictions made by the model for feature Mold protection time actual value for machine 2406 65
56 Predictions made by the model for feature Heating energy actual value last cycle for machine 1622 65
57 Predictions made by the model for feature Injection pressure peak 1 for machine 1622 66
58 Predictions made by the model for feature Mold protection time actual value for machine 1622 66
59 Predictions made by the model for feature Heating energy actual value last cycle for machine 1805 67
60 Predictions made by the model for feature Cycle time last value for machine 1805 . 67
61 Predictions made by the model for feature Injection pressure peak 1 for machine 1805 68
62 Predictions made by the model for feature Mold protection time actual value for machine 2605 68
63 Predictions made by the model for feature Cycle time last value for machine 2605 . 69
64 Predictions made by the model for feature Injection pressure peak 1 for machine 2605 69
65 Prediction for machine 1622 70
66 Prediction for machine 2605 70
67 Detected anomalies for machine 2406 71
68 Detected anomalies for machine 1622 71
69 Detected anomalies for machine 1805 72
70 Detected anomalies for machine 2605 72
71 Forecast for machine by20 73
72 Forecast for machine 1622 73
73 Forecast for machine 1805 74

1. Introduction

An outlier is a data point that is significantly different than the remaining data points in a data set [1]. In most data mining and statistics literature, an outlier is also often referred to as anomaly. However, an outlier typically refers to abnormalities or even noise in the data whereas an anomaly is a specific kind of outlier as it may indicate an unusual behaviour. Pertaining to the context of this project, this anomaly may represent a failure of a system or a potential bad batch when producing the plastic parts for the electric shaver.

Over the past couple of years, Philips has developed many household products. One of which is electric shavers. Philips Drachten encompasses a large suite of highly automated processes used during the manufacturing of electric shavers. Of these manufacturing processes, injection moulding is of particular importance, as this is used during the fabrication of plastic components for electric shavers. All of these plastic parts are manufactured onsite at Drachten, requiring approximately 80-90 moulding machines. Injection moulding, however, is a competitive market, making it essential for Philips Drachten to continuously improve on quality, production performance, and costs where this process is concerned.

The aim of this project is to model time series predictions in order to detect potential anomalies on the datasets that are produced by the machines. It should pre-warn the operators with an alert on a bad batch (when the machines parameters are off, it can lead to a batch that can not be used). Thus, we will analyse if there are any anomalies in the machine settings in order to alert the workers on a potential bad batch, eventually reducing fall-off rate. At the moment they inspect this manually, but evidently it is very time consuming.

Currently the quality inspection is done manually. But an automated process is much more desirable such that it can give alert in advance in order to prevent producing bad batches. This is to be avoided since a batch could consist of approximately 3000 products in which each has an estimate cost of 0.50 - 1 euro. Of course with this amount, if there are only one or even up to 10 badly produced plastic parts, it does not seem too severe. However, since everything is produced in batches, the cost will be higher if we have to throw away the current batch due to bad quality. Consequently, this kind of cost is more preferred to be reduced while also maintaining the quality of the produced plastic components on the electric shavers.

This thesis aims to answer the following research question: ***To what extent do machines' parameter settings influence the predictions of alerts?***

In this research, two approaches were considered namely univariate and multivariate analysis. This means that in the former, we will analyse one individual machine parameter at a time whereas the latter we take into account a combination of different parameters. In addition, we will be analysing six different injection moulding machines of which four are from the same brand and the other two are from another brand.

The rest of the thesis is structured as follows: in section 2, we will lay out a couple state-of-the-art methods with respect to the two approaches that we considered. In section 3, we discuss the methodology used in this research in terms of feature selection method, univariate and multivariate models and we will also explain the experimental settings. Then, we present our findings and discuss our observations in section 5 and section 6 respectively. Finally, we draw our conclusion in section 7 as well as provide suggestions for future work.

2. State of the Art

Time Series has become a popular research topic within the Data Science field. Nowadays, one can find time series data almost in every field. Some common textbook examples of a time series data include, but not limited to, stock markets in financial field or oil consumption of an aeroplane in automotive. One may also encounter the use of time series data in some other fields such as in modern manufacturing industries and/or information services [21] where the data is taken from the sensors of the IoT devices, for instance. Pertaining to this project, we are mainly dealing with sensors data as well which consists of measurements taken at certain timestamps from six injection moulding machines. With the growth of the popularity of time series, it can be encountered almost in every area of research, even also almost in every branch of industries.

One particular task in time series mining that has also become increasingly popular is outlier or anomaly detection [5]. The extent of its significance often can become trivial in quite a variety of application domains such as credit card fraud detection where one can identify if there are some unusual activities originated from someone's card. Network intrusion detection in cybersecurity whereby if there is a hacker who may be trying to access or send sensitive data to its target, it could produce an abnormal traffic in the computer network [14]. Another example from automotive field is early detection of oil leakage occurrences which turns out to be a challenging problem due to the continuous movement of oil across the machinery equipment parts [15]. In this particular example, the anomaly detection is aiming towards predictive maintenance whereby one uses anomaly detection in order to prevent oil leakage since that could be dangerous. Besides, this research also has the same goal as this thesis whereby we develop models to detect anomalies in order to prevent the injection moulding machines to produce a bad batch of the plastic parts.

There are many algorithms that can be used for anomaly detection, some of which have open source implementation or are implemented in a package or library of certain programming languages. Gaussian Processes, (Deep) Neural Networks, Long Short-Term Memory (LSTM) and ARIMA models, to name a few, are the state-of-the-art of anomaly detection techniques particularly in time series data.

Gaussian Processes The authors of paper [4] propose a novel machine-learning-based method for discovering the inherent structures of anomalies arising in Internet-of-Things (IoT) sensor data. Their ideas consist of modeling and describing anomalies by means of kernel expressions, which are combinations of four well-known kernels. Fitting these kernel expressions to the sensor data allows them to decompose the inherent structure of an anomaly. Moreover, one can also describe its individual behaviour such as linearity and periodicity by natural language. The results show that their method is suitable for modeling differently structured anomalies. Moreover, Gaussian processes turns out to provide a powerful tool for future algorithmic investigations of IoT sensor data. This particular research is relevant for this thesis as it demonstrates the application of Gaussian Processes in anomaly detection problem using sensor data, which incidentally is the same kind of data set that is used in this project.

Neural Network As mentioned previously, another method for anomaly detection is by using a form of Neural Network. In [7], two networks namely Hierarchical Temporal Memory (HTM) and Bayesian Network (BN) were used for a real-time anomaly detection algorithm (RADM). The HTM model was used to evaluate the real-time anomalies of each univariate-sensing time-series. Then, a model of anomalous state detection in multivariate-sensing time-series based on Naive Bayesian is designed to analyze the validity of the above time-series. Lastly, considering the real-time monitoring cases of the system states of terminal nodes in Cloud Platform, the effectiveness of the methodology is demonstrated using a simulated example. Extensive simulation results show that using RADM in multivariate-sensing time-series is able to detect more abnormal, and thus can remarkably improve the performance of real-time anomaly detection. This paper appears to be very much inline with this thesis as well since it considers two approaches, i.e. univariate and multivariate as does this project.

Long Short-Term Memory Long Short-Term Memory (LSTM) is not only becoming popularly used for Natural Language Processing tasks, but it can also be used for anomaly detection technique. In paper [19], a Multivariate Convolution LSTM with Mixtures of Probabilistic Principal Component Analyzers was developed for a data-driven anomaly detection algorithm. The proposed approach uses both neural networks and probabilistic clustering to improve the anomaly detection performance which was then evaluated with a total of 22 million telemetry samples collected for 10 months from Korea Multi-Purpose Satellite 2 (KOMPSAT-2), as well as being compared to other state-of-the-art approaches. The results show that their proposal yields 35.8% better in precision, and 18.2% better in F1 score than the best baseline approach.

PyOD library Aside from the aforemetioned papers, there is also a Python library / package called PyOD [22] which consists of implementations of different anomaly detection algorithms. This library was specifically developed for anomaly or outlier detection algorithms. It contains implementations for some well-known algorithms such as Histogram-based Outlier Detection, Local Outlier Factor (LOF), IsolationForest Outlier Detector (IForest) as well as some state-of-the-art methods like XGBOD: Improving Supervised Outlier Detection with Unsupervised Representation Learning, Variational Auto Encoder (VAE) and beta-VAE for Unsupervised Outlier Detection, Multiple-Objective Generative Adversarial Active Learning (GAAL) to name a few. The reason why we considered this package was because our project mainly deals with unsupervised learning and it provides several implementation of unsupervised anomaly detection algorithms. However, we did not make use of this package since their output is not very inline with what our models try to achieve.

ARIMA model The last method that we considered was using ARIMA models which is a very common method for time series analysis, particularly in terms of univariate analysis. In [20], three models were developed to forecast Hepatitis incidence in Heng County China. These models include an ARIMA model and Generalized Regression Neural Network (GRNN) which were trained using the incidence data from the HengCounty CDC (Center for Disease Control and Prevention), and later a hybrid model which consists of the combination of the two aforementioned models was developed. In order to determine which of these three models perform best, several performance metrics were considered namely mean absolute error (MAE), root mean square error (RMSE), mean absolute percentage error (MAPE) and mean square error (MSE). Based on the results of these four metrics, the paper concludes that the hybrid model outperforms the other baseline models in the validation stage and can be a potential decision-supportive tool for controlling hepatitis in Heng County, China.

Another research that uses ARIMA model to detect anomalous user behaviour on social media [12]. The author hypothesizes that if the gathered user data can be distinguished from white noise, then an ARIMA model can be parameterized in order to identify the underlying structure and forecast data. The results indicate that ARIMA models can identify the anomalous behaviour in the data by means of analysing the underlying patterns, provided that there is enough data available.

Having mentioned the different methods above, there are some advantages and disadvantages that come with each of them. One of the literature demonstrates the robustness of Gaussian process which turns out to provide a powerful tool for for modeling differently structured anomalies. However, there are two limitations of Gaussian processes[8]: 1) the time complexity is $\mathcal{O}(n^3)$ where

n is the number of data points. Given this reason, one can automatically deduce that it will be very time consuming to run the training. 2) Gaussian Processes are examples of non-parametric models which are based on the Bayesian rule. This concept is unfortunately still unfamiliar to industrial engineers. Neural Network-based techniques are also typically computationally expensive, but there are also some advantages for instance Bayesian Network has the ability of dealing with uncertainties by showing the relationship between parameters in the form of probability [7]. Despite these aforementioned advantages, we were not able to implement these methods due to the limited computational resources available during the course of the project.

From the state-of-the-art methods listed above, the only viable option for us was to use ARIMA model. Using this model, making forecasts or predictions can be done easier and it does not need as much resources as the other methods. This is because ARIMA models ignore any additional input variables, or to put it simply, it takes one feature in the model at a time. Moreover, the basis of the analysis can simply be done with respect to the provided historical time series data [18]. Given these reasons, it seems to be an appropriate choice of method for the univariate approach at the very least. Although, one can not draw any causal inferences from the fitted ARIMA model. This means that the model does not necessarily suggest the effect of the parameters of interest has on the detected anomalies. Furthermore, it is important to note that the aforementioned methods are based on multivariate analysis. However, due to the limited computational resources, we had to take another turn and decided to also use ARIMA model in combination with PCA, by making an assumption that the relationship between the eventual selected features are linear.

3. Methodology

Preprocessing step is very essential in any data science projects. This is due to the fact that typically, we are dealing with real data which consists of a lot of measurements and subsequently a lot of features or dimensions. However, it is not very feasible to be dealt with because it also limits further the computational power. Pertaining to this project, the preprocessing step includes cleaning up data and feature selection in order to reduce dimension and only base the analysis on the most informative features. The data preprocessing will be discussed more in details in section 4.

As a part of preprocessing step, feature selection is also of importance because this way we can investigate all features that are present in the data and choose the ones that are more informative or have more influence to the models. Typically, most feature selection methods are wrapper methods which make use of machine learning algorithms to assess the variables and select a subset of those subsequently. However, they usually require having independent and dependent variables and since our data set does not have any independent variables, therefore we opted to do Filter method.

This section is organized as follows: first, we will explain the method that is used for feature selection. Then, we will cover the theory behind time series analysis which includes what it entails, its objectives, its components, and some examples for illustrations. More importantly, we will also introduce the fundamental concepts of the model that we used in the experiment, i.e. ARIMA model. This includes the mathematical definitions, the concept of stationarity series prior to applying the model, parameters of the model as well as the tuning process which yields the best performing model as indicated by the Akaike Information Criteria (AIC). And finally, we will also describe two metrics that were used to evaluate the models' performances.

3.1. Filter method

Filter method aims to choose the best subset of features which are selected based on statistical criteria. This method seems to be the most appropriate considering the size of the six different data sets that we are working with and its further implication on the computational power and running time.

The data sets consist of features which are highly correlated with each other. If feature A is highly correlated with feature B, it means that they are highly dependent on each other which is not really desirable in the context of regression nor classification. Hence, we can exclude features which have high correlations by means of calculating the pairwise correlation using the standard Pearson method [9] which is defined by the following equation

$$\rho(x,y) = \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_i (x_i - \bar{x})^2 (y_i - \bar{y})^2}} \tag{1}$$

Since we do not have independent and dependent variables, therefore we calculated the correlations between features. This way, we can investigate which features are dependent on others and which are not. It is important to note that a subset of features where they have low correlations with each other is much more preferable since highly correlated features most likely tend to convey the same information and hence the interpretation of results will eventually become slightly redundant.

Furthermore, another statistical criterion was also considered which is variance [3] of the remaining features after having removed the highly correlated ones. This is because we want to ensure to filter out the features with low variance since those generally do not have very high predictive capability. In order to achieve this constraint, one can set a threshold as a fixed constant which denotes the minimum variance that will be accepted. This means that any features whose variance is lower than this threshold will not be included in the final resulting subset of features which will be used for the models.

3.2. Time Series

A time series is a sequentially ordered set of values that are observed over a certain time period [18]. In the literature, it is usually denoted by $x_1, x_2, ..., x_t$, where t indicates the time step and x_t denotes the observed value of set x at a specific point in time t. Its objectives are to identify and model the structure of the time series and/or to forecast future values in the time series or to put it simply, predict future trends.

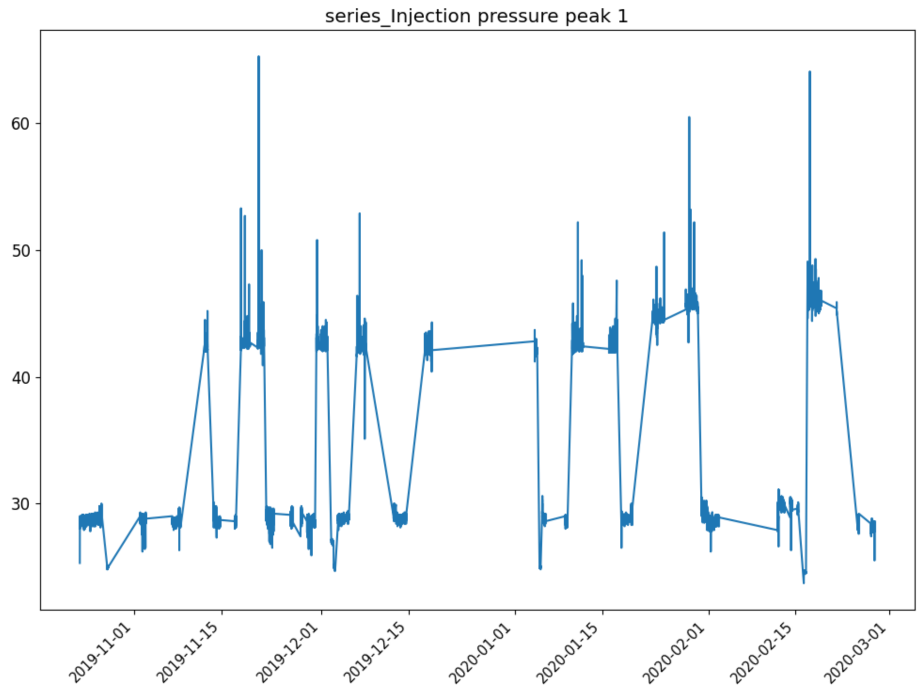


Figure 1: *Example time series*

Figure 1 shows an example of a time series that is taken from the data set. In this particular example, the series consists of observations of injection pressure that are recorded at every shot made by the machine. Looking closely at the plot, it can be seen that each of the observed value in the series is recorded at a specific time stamp.
Typically, a time series consists of the following four components:

- ***Trend*** indicates whether the values in the series exhibit increasing or decreasing trend over time.

  Figure 2 shows an example of a time series which exhibits an increasing trend. Looking closely at the plot, it can be clearly seen that in the observations, the value at time $t + 1$ tends to always be higher than the value at time t. Furthermore, a small insight that we can observe from such a plot is that most likely, this series represents a cumulative sum or total heating energy that is consumed by the machine.

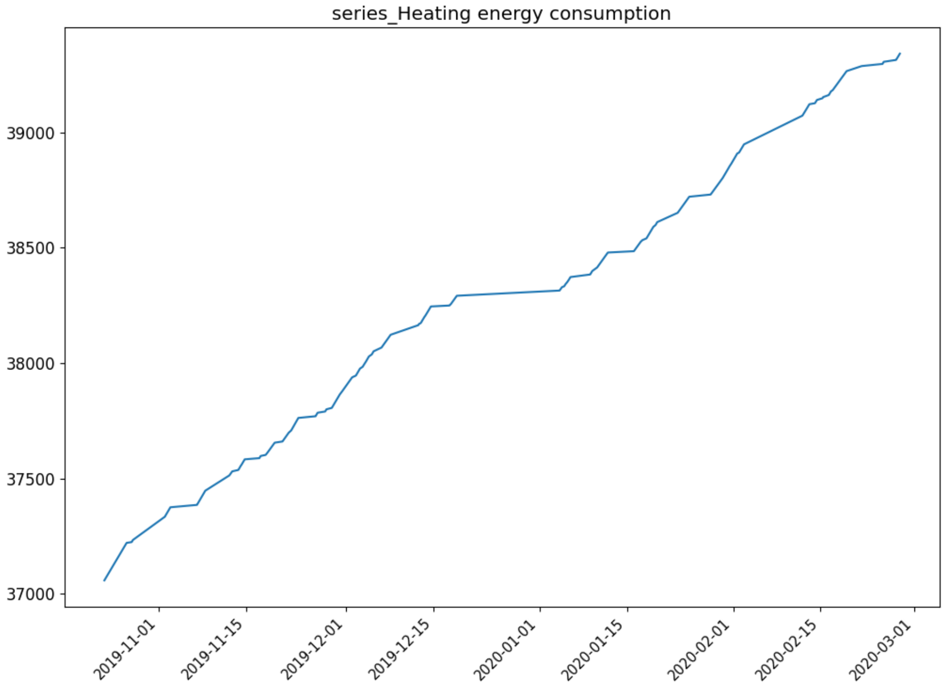


Figure 2: *Example of trend in a time series*

- ***Seasonality*** describes the fixed, periodic fluctuation in the observations over time. Example: monthly retail sales can fluctuate over the year due to the weather and holidays. It is important to note that if we observe seasonality in the data, we should account for this in the model. Usually, if a time series exhibits this behaviour, one can detrend or makes the series stationary by means of differencing it with a fixed order s. In this case, s denotes the seasonal period which takes values 7 for daily data, 52 for weekly data, 12 for monthly data or 365 for annual data [18]. Seasonal Autoregressive Integrated Moving Average model, or SARIMA for brevity, provides a means to do so. Denoted as SARIMA(p,d,q) $\times$ (P,D,Q)s, where

  - p, d, and q are the original ARIMA parameters.
  - s = the seasonal period, e.g. 7 or 52 or 12 etc.
  - P = the order of an AR model across the s periods.
  - D = the order of differencing applied across the s periods.
  - Q = the order of an MA model across the s periods.

- ***Cyclic*** refers to, same as seasonality, a periodic fluctuation except that it does not always happen at the exact fixed period of time.

  Figure 3 depicts an example of a time series which exhibits a cyclic behaviour. Looking closely at the plot, it is clear that series shows an increasing trend which does not always occur at a fixed time period. Furthermore, the parameter shot counter indicates the number of cycle that the machines produce. The counter is first initialized to 0 and is incremented by 1 until it resets back to 0, which indicates that the machine starts a new cycle.

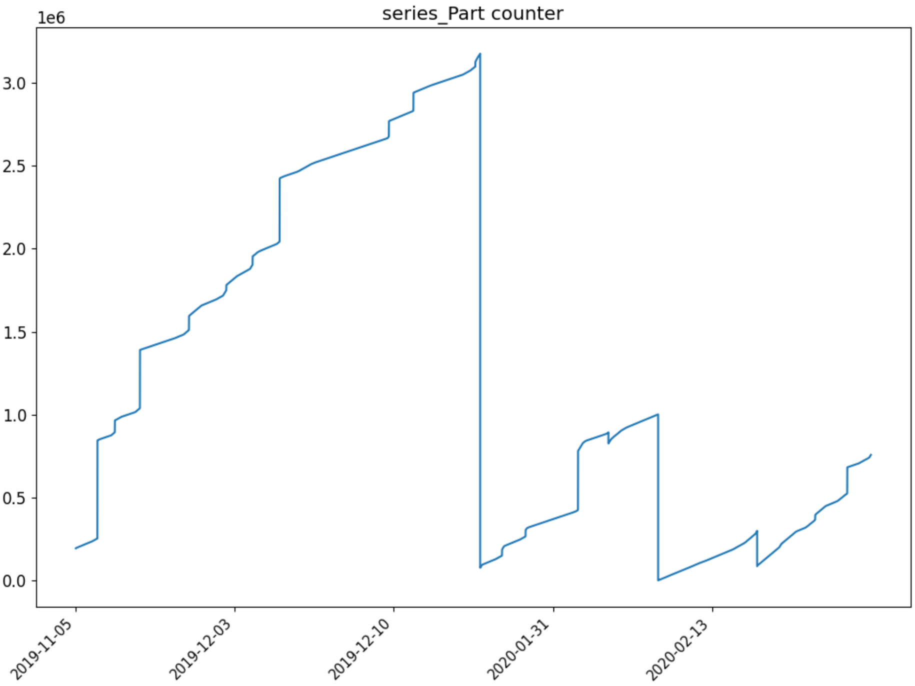


Figure 3: *Example of cycle in a time series*

- ***Random*** components of the series; which refers to the remaining observation after accounting the previous three components.

The Box-Jenkins methodology [18] for time series analysis involves the following steps:

1. Investigate data by means of identifying and accounting for any trends and seasonality in the time series.
2. Examine the remaining time series and determine a suitable model.
3. Estimate the model parameters.
4. Assess the model and return to step 1, if necessary.

3.3. Stationarity

It is important to note that in order to apply an ARIMA model, the series must be stationary. One of its reasons is because, the effect of stationarity allows us to make assumptions that all observations are all independent of each other. Yet, we know that in a time series, all observations are time dependent as each of its value is being recorded at a certain period of time. A series is said to be stationary if it satisfies the following conditions:

- The mean is constant over time.
- The variance is finite and constant over time.
- The covariance of y_t and y_{t+h} depends only on the value of $h = 0, 1, 2, ...$ for all t.

To check for stationarity in a time series, one can go about with two approaches whereby either one can check the series manually, or by using some stationarity tests. Specific to this project, we considered two hypothetical tests namely Augmented Dickey Fuller (ADF) and Kwiatkowski–Phillips–Schmidt–Shin (KPSS) tests. These two tests are based on unit root test which can be used to determine if the time series should first be differenced in order to make it stationary [23].

Augmented Dickey Fuller (ADF) test is a unit root test for stationarity. In this test, the null hypothesis says that there is a unit root and hence, time series is not stationary. On the other hand, the alternative hypothesis suggests that the series is stationary. To determine if the series is stationary or not, one can check if the test statistics is larger than the critical value, for instance at 5% significance level of the test. Or, one can also refer to the p value and see if it is less than 0.05. If either of these conditions are met, then one can conclude that the time series is stationary.

Kwiatkowski–Phillips–Schmidt–Shin (KPSS) test checks if the time series is stationary around a mean or linear trend, or is non-stationary due to a unit root. This test is less popular than the ADF test and has contrary null and alternative hypotheses. In this test, the null hypothesis suggests that time series is stationary whereas the alternative hypothesis suggests that it is non-stationary. Subsequently, one should check if the p value is larger than 0.05 in order to conclude that the time series is stationary.

Aside from the two hypothetical tests that are mentioned above, one can also consider to use some transformations to make a time series stationary. One of the most common method is to apply log transformation to the series, such that we can achieve variance stationary over time. Another method, that is also tied with one of the parameters of an ARIMA model, is to apply differencing such that we achieve mean stationary over time.

3.4. ARIMA model

Autoregressive Integrated Moving Average models [18], or ARIMA for brevity, are a subset of linear regression models that allow us to predict or forecast future values using the target variable's past observations. It is a combination of three sub-models namely Autoregressive model (AR), Integrated model, and Moving Average model (MA). Consequently, an ARIMA model is parameterized by a combination of each sub-model's parameter. To be more specific, if we consider an AR model with order p, with differencing order of d and an MA model with order q, the ARIMA model takes parameters (p, d, q).

Formally, an Autoregressive and Moving Average model of order p and q (ARMA(p, q)) is defined by the following equation

$$y_t = \delta + \phi_1 y_{t-1} + \phi_2 y_{t-2} + ... + \phi_p y_{t-p} + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + ... + \theta_q \tag{2}$$

It is important to note that Equation 2 is composed of two smaller expressions for each of the sub-model which will be explained in the following subsections.

3.4.1. Autoregressive Model

In an Autoregressive model, or AR for brevity, the currently observed value can be expressed as a linear function of its past p values [18] or also referred to as p-lags of values. An AR model of order p is defined by the following equation

$$y_t = \delta + \phi_1 y_{t-1} + \phi_2 y_{t-2} + ... + \phi_p y_{t-p} + \varepsilon_t \tag{3}$$

In Equation 3, δ denotes the mean of the series. If the series is assumed to be stationary with zero mean, therefore $\delta = 0$ [6]. ϕ_j denotes a constant for $j = 1, 2, ..., p$ and $\phi_p \neq 0$. From this equation, one can say that an Autoregressive model of order p regresses the current value of time series y_t by means of linear combination of its past values plus a white noise process, denoted by ε_t. This term is drawn from a normal distribution where $\varepsilon_t \sim N(0, \sigma_\varepsilon^2), \forall t$ and is used to represent random, independent fluctuations which occur in the time series. It is important to note that this particular model has one parameter, p, which denotes the number of past observations that we take into account in order to forecast the current value.

Partial Autocorrelation Function The parameter p of an AR model is directly related to the partial autocorrelation function (PACF). In fact, one can use the PACF plot in order to determine the value of p. Partial Autocorrelation Function calculates the correlation of the remaining series after removing the effect of y_{t+1} to y_{t+h-1} values from the measure. It is expressed by the following equation

$$PACF(h) = \text{corr}(y_t - y_t^*, y_{t+h} - y_{t+h}^*), \text{ for } h \geq 2 \tag{4}$$

$$= \text{corr}(y_t - y_{t+1}), \text{ for } h = 1 \tag{5}$$

From Equation 4, it can be said that the PACF describes the relationship between an observed value at a certain time point t and its lag. Thus, to determine the parameter p of an AR model, one can, again, look at the PACF plot and analyse if there is still any correlation for lag values after lag p. If not, then it is clear from the plot that the series needs an AR model of order p.

3.4.2. Integrated Model

In order to apply ARIMA model, one has to make sure that the time series is stationary. One way to do so is to apply differencing step to the series [18], which is basically what the integrated model does. Referring back to step 1 of the box-jenkins methodology as mentioned in the previous subsection, one has to identify and account for any trends in the time series. If it appears that the series does exhibit a certain trend, for instance it shows an increasing trend over time, one has to remove this trend in order to make the series stationary. One way to do so can be by means of applying differencing of order d. This means that we subtract the value at x_t with its consecutive previous value, i.e. x_{t-1}. Formally, integrated model can be defined by the following formula

$$d_t = y_t - y_{t-1}, \text{ for } t = 2, 3, ..., n \tag{6}$$

In most cases, this model needs at most $d = 1$, which means that we only compute the differences between each successive values in the series. However, some special circumstances may require a higher order for d than 1 and it fully depends on the time series data that is used.

3.4.3. Moving Average Model

A Moving Average Model, or MA for brevity, makes predictions or forecasts the target variable using the model's past errors; which is defined as the deviations between the actual values from the time series and the predicted values generated by the model. More formally, in an MA model, the value of a time series is a linear combination of the current white noise term and the prior q white noise terms [18]. An MA model of order q is defined by the following equation

$$y_t = \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + ... + \theta_q \varepsilon_{t-q} \tag{7}$$

Similar to the Autoregressive model, in Equation 7, θ_j denotes a constant for $j = 1, 2, ..., q$ and $\theta_q \neq 0$ and ε_t represents a white noise process which is drawn from a normal distribution where $\varepsilon_t \sim N(0, \sigma_\varepsilon^2), \forall t$. This model forecasts the current value y_t in the time series by means of forming a linear combination of its past error terms. Subsequently, it also takes one parameter, q, which indicates how many past errors taken into account to predict the current value.

Autocorrelation Function Likewise, the parameter q is directly related to the autocorrelation function (ACF). It measures the correlation of two variables $corr(y_t, y_{t+h})$ and is defined by the following equation

$$ACF(h) = \frac{\text{cov}(y_t, y_{t+h})}{\sqrt{\text{cov}(y_t, y_t) \cdot \text{cov}(y_{t+h}, y_{t+h})}} = \frac{\text{cov}(h)}{\text{cov}(0)} \tag{8}$$

In Equation 8, the value of ACF lies between -1 and 1 and h denotes the "lag", or the difference between the time points t and $t+h$. The closer ACF(h) to 1 the better predictor is y_{t-h} for y_t. To determine the parameter q for an MA model, one has to look at the ACF plot. Subsequently, one can observe a moving average process that shows a strong correlation between y_{t-h} and y_t only up to the lag of q and then a sharp decline to 0. This implies that the time series can be modeled using an MA model of order q.

Summarizing the three sub-models of an ARIMA model, one can conclude that in this model, the current value of the series can be calculated by a linear combination of its past p observations and its q past error terms, as well as incorporating the order of differencing in order to make the series stationary.

3.5. Principal Component Analysis

For the multivariate approach, we assume that a set of different features would make a better predictor rather than only one single feature. In other words, a combination of parameters would allow us to better predict an alert in the future. Hence, in addition to the ARIMA model, we also considered the Principal Component Analysis model, or PCA for brevity. PCA is most commonly used as a dimensionality reduction technique whereby if the data consists of many dimensions, we can project them into a smaller number of dimensions by means of forming a linear combination between them. Its objective is to rigidly rotate the axes of the data into some principal axes whereby the greatest cumulative variance of the data is captured or can be explained by a smaller number of dimensions[2]. The principal axes have the following properties:

1. The principal axes are ordered in such a way that principal axis 1 has the highest variance, axis 2 has the second highest, and so on.

2. The principal axes are uncorrelated; meaning that covariance among each pair of the principal axes is zero.

Before applying PCA, it is important that the data should be centered at the origin, or more commonly known as "mean centering" of the data. This was done by means of subtracting the mean of the data set from each observation in the data; hence making the PCA a row-wise operation. In this project, PCA is applied to the remaining data that we obtained after feature selection. Then, we check how many principal components that we need in order to explain $> 85\%$ of cumulative variance of the data. Having done so, we then fit an ARIMA model to the resulting PCA models and make predictions afterwards.

3.6. Parameters tuning

In steps 3 and 4 of the box-jenkins method, one should estimate the model parameters and assess the model. This means that first we have to set the values for each of the parameters of an ARIMA model, i.e. (p, d, q), and evaluate the fitness or goodness of the model. Provided that the series is stationary, one can estimate the parameter p of an AR model by looking at the Partial Autocorrelation function (PACF) plot. Similarly, the parameter q of an MA model can be determined by looking at the Autocorrelation function (ACF) plot.

3.6.1. Auto Arima

Another way to estimate the parameters of an ARIMA model is to use the auto arima functionality [11]. Developed by Rob J. Hyndman and Yeasmin Khandakar, this function is a part of the updated version of **forecast** package in R. This function aims to choose the optimal orders of a (seasonal) ARIMA model by using unit root tests and the AIC. Meaning, that they use unit root tests to check for stationarity and the Akaike Information Criteria (AIC) to evaluate the fitness of the model. It performs a step-wise algorithm for traversing the model space efficiently since it is much more computationally heavy if one has to take a grid search approach where we fit every potential model and choose the one with the lowest AIC. Furthermore, the user has to specify the range for the parameters. The step-wise algorithm is described as follows

1. Try four possible models to start with using the pre-defined range of parameters. Of these four models, select the one with the smallest AIC, or is also referred to as the "current" model.
2. Consider up to 17 variations on the current model by means of varying the (combinations of) parameters by ± 1. This step is repeated by updating a new model with lower AIC to be the current model. The algorithm stops when it cannot find another model whose AIC is lower or close to the current model's.

This algorithm is guaranteed to converge and return a valid model. Since, at convergence, the model space is finite and at least one of the starting four models will be accepted. However, there are several constraints on the fitted models to avoid problems with convergence or near unit-roots which are outlined below.

- The values of p and q should not exceed 5 and the values of P and Q (in a SARIMA model) should not exceed 2 in each case respectively.
- The algorithm rejects any model which tends to lead to non-invertible or non-causal.
- The algorithm rejects any models which arise any errors in terms of non-linear optimization routine used for estimation. The reason being that if a model is difficult to fit, thus this model is most likely not a good representative for the data.

If the reader is interested to learn more about the implementation, please refer to [11].

3.7. Evaluation Metrics

In the interest of this project, we mainly deal with unsupervised learning and hence evaluation metrics such as accuracy, f1-score, area under the ROC curve might not necessarily be the best choice since they all require having labels in both actual data and the predicted data. Hence, to measure the performance of the models we consider two different metrics for univariate and multivariate approach, namely Mean Absolute Percentage Error and Mean Squared Error respectively.

3.7.1. Mean Absolute Percentage Error

Mean Absolute Percentage Error (MAPE, for brevity) is a forecast evaluation measure that is most widely used in industry practitioners [13]. It is defined by the following equation

$$MAPE = \frac{1}{N} \sum_{t=1}^{N} \left| \frac{A_t - F_t}{A_t} \right| \times 100 \tag{9}$$

where

- A_t denotes the actual values at time point t.

- F_t denotes the forecast values at time point t.
- N denotes the number of data points.

The MAPE has an advantage of being scale-independent which means that the calculated error is independent on the scale of the data and thus it can be used to compare forecasts across different data sets with possibly different scales [10]. Additionally, it is also easy to interpret. On the other hand, it also has quite a couple of disadvantages, one of which is since it highly depends on the division with the actual values (A_t), it tends to yield infinite or undefined values if the actual values consist of 0's and has a skewed distribution when the actual values are close to 0 ($A_t \sim 0$). Given this reason, it is quite impractical to use this measure for the multivariate approach. This is due to the fact that the result after applying PCA to the data brings the scale of the data to somewhere close to 0. Thus, we should consider another evaluation metric which is the mean squared error.

3.7.2. Mean Squared Error

Mean Squared Error (MSE) is a model evaluation metric often used with regression models [17]. It measures the average of the squared deviation between the actual values and the forecast values over all data points in the time series. The deviation measures how much the forecast values vary from the actual values. Mathematically, it is defined as follows

$$MSE = \frac{1}{N}\sum_{t=1}^{N}(A_t - F_t)^2 \tag{10}$$

where

- A_t denotes the actual values at time point t.
- F_t denotes the forecast values at time point t.
- N denotes the number of data points.

4. Experiments

We carried out experiments using the two aforementioned approaches; i.e. univariate and multivariate. Both approaches mainly follow the same pipeline with a small difference for the multivariate approach. In this section, we will explain the details in each step of the experiments, including certain decisions that we made along the way based on the intermediate results that we obtained.

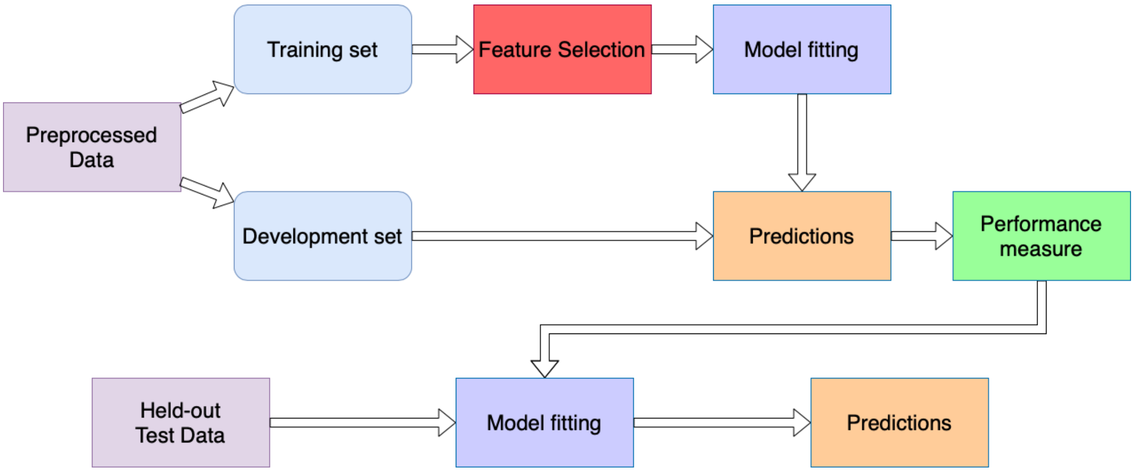


Figure 4: *Experimental Setting*

Figure 4 depicts the pipeline of the experiments. Looking closely at this figure, it can be clearly seen that there are two phases of the experiment involving 1) the (preprocessed) training data and 2) the held-out test data. We used the preprocessed data to develop the models. This data was split into training and development sets where the training set is used for feature selection and train the models. Then, using these pre-trained models, we generate the predictions and evaluate its performance using the development set.

In the second phase, we applied the models to the held-out test data. It is important to note that the significant difference between this data and the training data puts an emphasis on the importance of retraining the models. Subsequently, we might also get different models since the parameters could change with respect to the used data. Furthermore, even though the two approaches follow the same pipeline, there is a major difference in the final steps particularly for the multivariate approach. It involves the generation of thresholds to detect the anomalies in phase 1 and we only applied the second phase in the multivariate approach. The reason is because the results from the univariate analysis gives us more insights to the behaviour of the data between certain time periods as well as the relations between dissimilar features. With respect to the former reason, we found that sometimes the data from today can be very different from the data that was collected yesterday. More so if we compare the data this week to the previous week, and so on. Based on this observation, we decided that it would be more practical to train the models within a shorter period of time. On the other hand, we observed that if there is an anomaly in the data, it could most likely be caused by a combination of features. More on this will be explained in subsection 4.1 and subsection 5.1 respectively.

Aside from the two phases in the experiment, since we are considering six machines, therefore we will have six different sets of data for both training and held-out testing data. The machines are listed in Table 1 as follows

Table 1: *Tables of machines categorized into brands*

BY machines	EN machines
machine by20 machine by19	machine 2406 machine 1622 machine 1805 machine 2605

4.1. Data

The data was obtained from different machines, since the analysis is done independently. We have two sets of data, whereby the first one is used as training while the other one is considered as a held-out test data. The training data consists of 100000 measurements for all machines and 53767 for machine 2605. The reason being that this machine is most likely not used very often and only used to produce special products and therefore it has marginally less measurements. Furthermore, it contains measurements for ± 5 months, i.e. from October 2019 to early February 2020. As depicted in Figure 4, this data is split into training and development sets using 80:20 ratio where training set consists of 80% of the data and the development set consists of the remaining 20%. Coincidentally, the training set is composed of the first 5 months of the data and the development set has measurements from the last two weeks in February. On the other hand, the held out test data only has measurements for two weeks, i.e. roughly from late June or beginning of July towards mid-July of 2020. It is important to note that while in the first phase we are training the models on a larger data set and test the models on a rather smaller data set, it is much more practical if we train the models on a much shorter period of time. This is exactly what is done in the second phase where we re-train the models only on two weeks of data since, based on the results that we obtained, this seems to be an appropriate time frame where the machines do not yield too much inconsistencies in the data.

4.1.1. Preprocessing

1. Remove set parameters and constant series. The data sets not only consist of measurements that are taken at certain time period, but they also contain some set parameters which may refer to certain settings that were done by the operators of the machines. Moreover, there are also some parameters, ones that are not setting parameters, which indicate constant series since their values do not change over time. Generally, these constant series are not necessarily considered to be very informative or useful to the models and further analysis. Therefore, we need to remove them.

2. Preprocess clamp force parameter for BY machines. After having done the univariate analysis on the parameter clamp force for both BY machines, we realized that it yields very poor model as indicated by its performance. Then, we looked back to check the data and noticed that it contains quite a few irregular values which do not necessarily lie within the expected range. To ensure if this is really the case, we consulted this concern with the domain expert and he confirms that it is indeed true. However, since it is still unknown to us what was the cause behind the generation of these irregular values, therefore we treated them as if they were missing values which occur at random and therefore should be removed. Figure 5 shows an example of irregular values that occur in the clamp force parameter. As can be seen, the expected observed value in the by20 machine should be ≥ 96 and any values below 96 are considered to be irregular. Likewise, the data in the by19 machine expects the observed values to be ≥ 60.

timestamp (UTC+02:00) Local - Europe/Amsterdam: CEST	series_Clamp force
11/05/2019 06:55:37.926	103.0317
11/05/2019 06:55:47.177	102.6498
11/05/2019 06:55:56.679	12.96848
11/05/2019 06:56:15.185	103.4136
11/05/2019 06:56:24.475	102.8726
11/05/2019 06:56:33.934	102.9363
11/05/2019 06:56:43.177	36.58223
11/05/2019 06:56:52.428	105.4186
11/05/2019 06:57:01.682	103.0636
11/05/2019 06:57:20.179	102.809
11/05/2019 06:57:38.932	0.2386833
11/05/2019 06:57:48.186	102.6498
11/05/2019 06:57:57.436	-0.07956111
11/05/2019 06:58:06.682	103.2545
11/05/2019 06:58:16.180	9.08589
11/05/2019 06:58:25.432	96.73048
11/05/2019 06:58:34.681	103.9547
11/05/2019 06:58:43.928	37.34602
11/05/2019 06:58:53.189	102.7771

(a) *Example data set for machine by20*

timestamp (UTC+02:00) Local - Europe/Amsterdam: CEST	series_Clamp force
11/03/2019 23:39:30.970	77.97732
11/03/2019 23:39:39.468	78.1351
11/03/2019 23:39:47.965	60.21074
11/03/2019 23:39:56.467	25.5296
11/03/2019 23:40:04.969	4.165524
11/03/2019 23:40:13.467	78.04043
11/03/2019 23:40:21.967	77.18839
11/03/2019 23:40:30.474	77.66174
11/03/2019 23:40:38.973	6.248284
11/03/2019 23:40:47.471	27.01278
11/03/2019 23:40:55.972	10.69782
11/03/2019 23:41:04.472	78.04043
11/03/2019 23:41:12.970	77.72486
11/03/2019 23:41:21.499	3.534385
11/03/2019 23:41:29.971	24.10954
11/03/2019 23:41:38.471	7.857689
11/03/2019 23:41:46.967	77.75642
11/03/2019 23:41:55.482	77.75642
11/03/2019 23:42:03.972	77.69331

(b) *Example data set for machine by19*

Figure 5: *Example of irregular values in clamp force parameter for BY machines*

3. Remove occurrences in the first hour when the machine is turned off and turned back on again. The machines exert a somewhat strange behaviour when they are turned off and turned back on again. This behaviour is shown by a sudden peak in the data, yet, this should not be considered as anomaly. Moreover, this particular step becomes very useful for generating thresholds to detect anomalies, since it requires finding a stable period where there is not a lot of change or inconsistencies.

4.2. Features

Table 2: *Table of initial number of features per machine*

Machines	Number of features
by20	26 features
2406	44 features
1622	37 features
by19	26 features
1805	46 features
2605	43 features

Although the number of measurements in the training data is the same across all machines in general, the number of features do differ. Table 2 shows the number of features that each machine has. See appendix A for a complete list of all feature names. It is important to note that these numbers are obtained after having cleaned up the data. Furthermore, according to the domain expert, these features have a lot of dependencies with each other. A change of one feature could potentially affect the observed measurement of another feature and/or the combination of them could cause what we observe as "peaks" in the data. Hence, we should calculate the correlations between features and remove the highly correlated ones.

4.2.1. Correlation

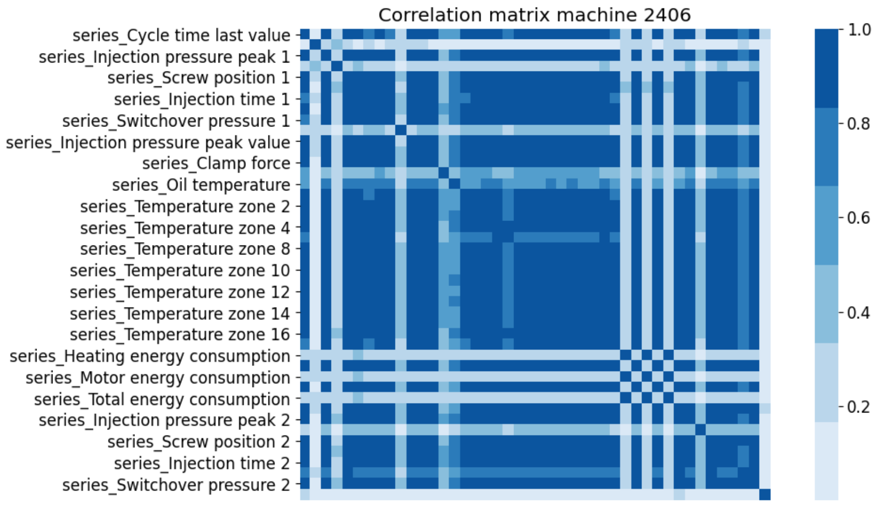


Figure 6: *Correlation matrix for machine 2406*

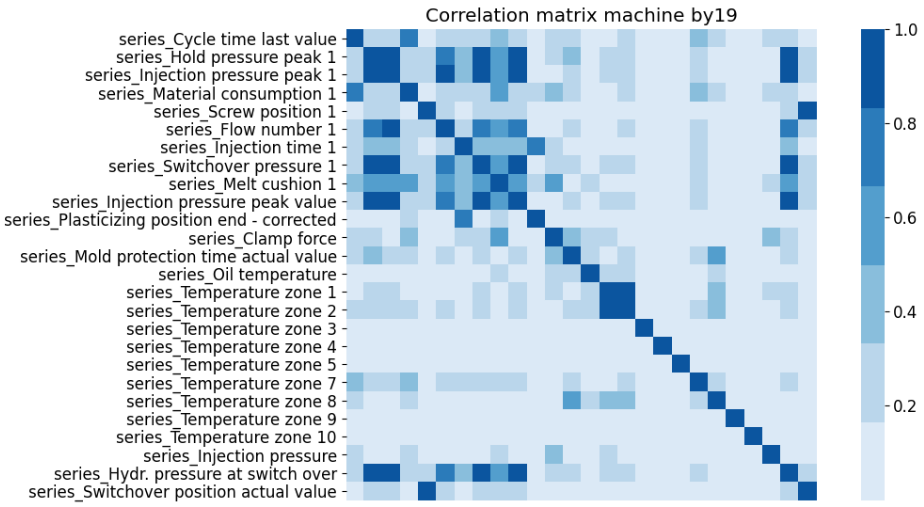


Figure 7: *Correlation matrix for machine by19*

As explained in subsection 3.1, correlation measures statistical dependencies between two variables. In this case, between different pairs of features. Figure 6 and Figure 7 depict correlation matrices of features for machine 2406 and machine by19 respectively. In these matrices, it can be seen that any features which have correlation values between 0.9 and 1.0 are considered to have high correlations, whereas anything less than that is considered low correlations. In these two matrices, it can be clearly seen that machine 2406 have more highly correlated features compared to machine by19. And as a result of feature selection, we might expect to have less number of selected features. Looking closely at Figure 6, there are 44 features and as can be seen from the matrix, we see a lot of features which are highly correlated with the others. For example, the temperature zones parameters seem to have high correlations with each other and this is due to the fact that they control the temperature of the heating of the barell. Or, if we look closely at Figure 7, we can see that Hold pressure and Injection pressure are also strongly correlated. This could be due to the fact that during the injection moulding process, the machine has to do the injection at a certain pressure.

4.2.2. Variance Threshold

After removing the highly correlated features, we apply another method that is called Variance Threshold [16]. As the name suggests, the objective is to exclude features with very low variance since generally they do not necessarily have a strong predictive capability. For the purpose of this project, the variance threshold for all EN machines is set to be 0.0001, since we have already ruled out features with zero variance by means of removing constant series. On the other hand, the threshold for BY machines is set to be 1.0, because it appears that features whose variance is less than 1.0 do not really help us to reduce the number of features and their results were not that informative in the end. Ultimately, the point of using the variance threshold is to ensure that our final subset consists of features with non-zero variance.

4.2.3. Common features across machines

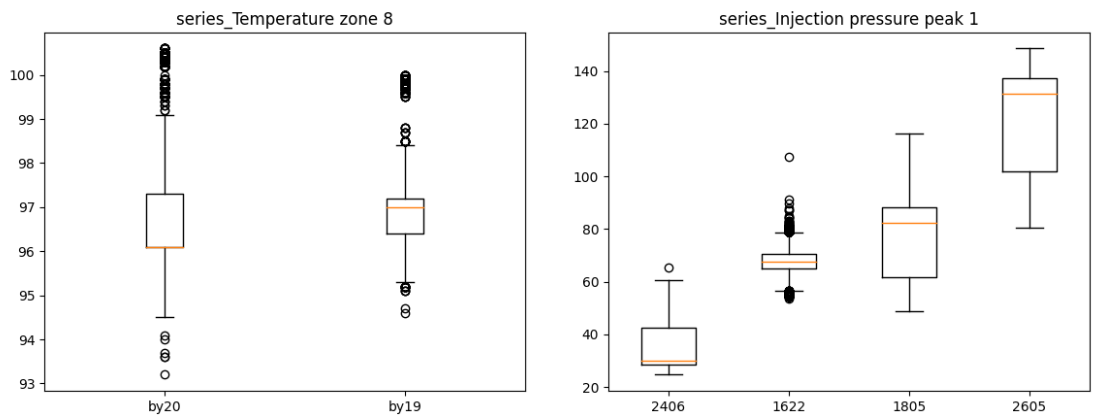
The last step of the feature selection phase is to find common features across machines, provided that they belong to the same brand. This means that we will eventually have two final sets of features, i.e. for EN machines and BY machines. Below we show the final list of features that are considered for the models and the analysis.

Table 3: *Results from feature selection*

BY machines	EN machines
Hold pressure peak 1 Injection pressure peak 1 Flow number 1 Clamp force Mold protection time actual value Oil temperature Temperature zone 8	Cycle time last value Injection pressure peak 1 Mold protection time actual value Heating energy actual value last cycle

Looking closely at Table 3, it can be clearly seen that we have selected 7 features for both BY machines and 4 for EN machines. Having done this step, one may ask if these features are indeed comparable with each other. To do so, we perform a small t-test to investigate whether the same feature from different machines are comparable.

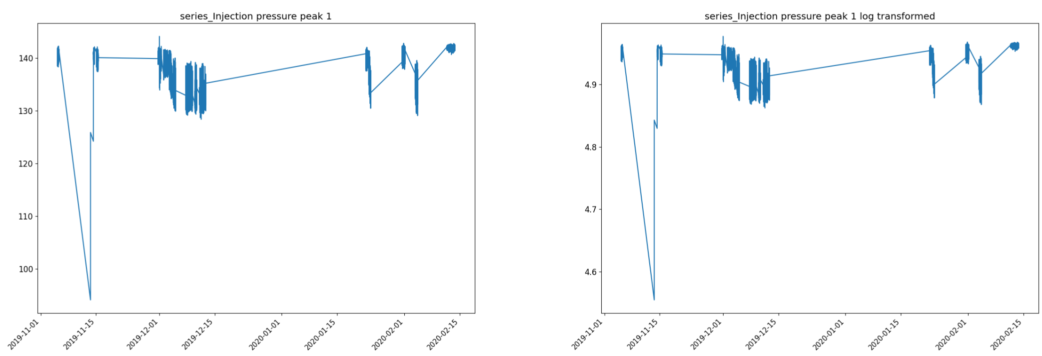
Figure 8a shows the boxplots of parameter Temperature zone 8 with respect to both BY machines. As can be seen from the plot, they seem to be rather comparable. Even though their means are slightly different but their distributions would still be more or less similar to each other. Furthermore, Figure 8b shows the boxplots of parameter Injection pressure peak with respect to different EN machines. From a hindsight, we can see that it depicts a contrast observation than what we can derive from Figure 8a. By contrast, Figure 8b shows that this parameter is not really comparable across all machines. This is due to the fact that these machines produce different products and therefore will not be comparable.



(a) *Boxplot for Temperature zone 8* **(b)** *Boxplot for Injection pressure peak 1*

Figure 8: *Boxplot of a feature across BY machines (left) and EN machines (right).*

4.3. Stationary Series



(a) *Original series* **(b)** *Log transformed*

Figure 9: *Example of non-stationary series*

Figure 9 and Figure 10 show an example of the process of making the series Injection pressure peak stationary whereby in Figure 9a we can see the original time series, in Figure 9b we can see the series after log transformation and lastly in Figure 10 we can see the resulting series after being differenced. To check for stationarity, we applied the ADF and KPSS tests to each of these series, then we perform the typical hypothetical testing in order to draw a conclusion. We can either check if the test statistics value is higher or lower than the critical value at 5% significance level of the test, or see if the p-value is higher or lower than 0.05. Confirming whether the series is stationary or not is based on this and the initial null and alternative hypotheses.

Table 4 shows the results applying the ADF test and KPSS test to check for series' stationarity. Looking closely at the result, the test outputs all values for the test statistics, p-value and the critical values at 1%, 5% and 10% levels. First, let us check whether the original series is stationary or not. If we compare the values for t-statistics and critical value at 5%, it is clear that the former is smaller than the latter. Moreover, the p-value is also larger than 0.05. Provided that the null hypothesis for the ADF test is that there is a unit root and hence, time series is not stationary, we

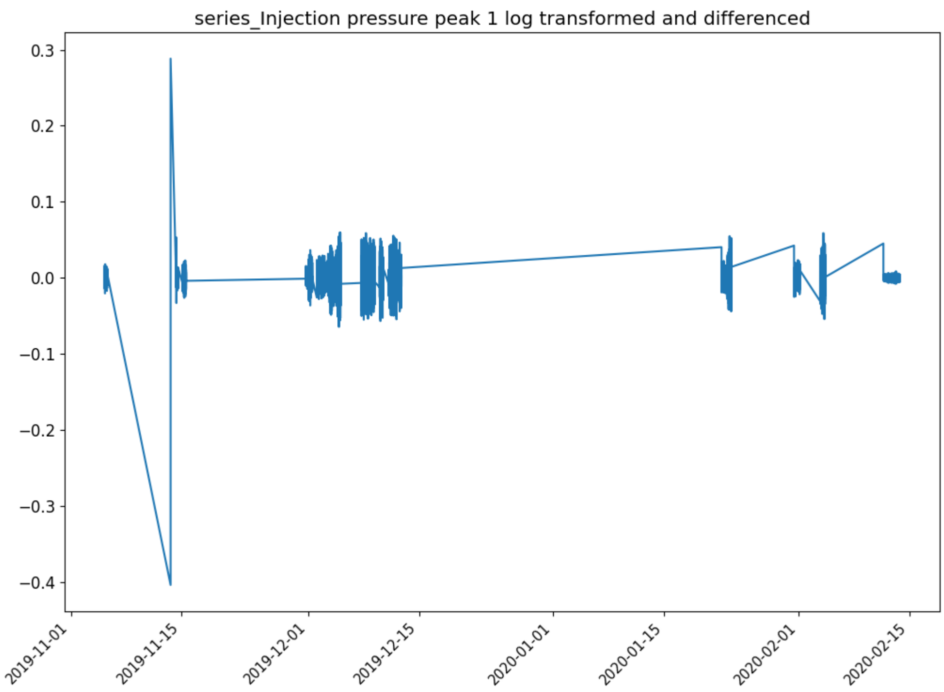


Figure 10: *Example of stationary series*

Table 4: *Stationarity tests*

Series	ADF test	KPSS test
Original series	t-statistic: -2.2002264791706927 p-value: 0.20616326824737136 Critical Values: 1%: -3.4304845661434378 5%: -2.8615994754339664 10%: -2.5668016568130527	t-statistic: 6.6727403385824235 p-value: 0.01 Critical Values: 1%: 0.739 5%: 0.463 10%: 0.347
Log transformed series	t-statistic: -2.2260394446748633 p-value: 0.19693072818850427 Critical Values: 1%: -3.4304845661434378 5%: -2.8615994754339664 10%: -2.5668016568130527	t-statistic: 6.6350740320854324 p-value: 0.01 Critical Values: 1%: 0.739 5%: 0.463 10%: 0.347
Log transformed and differenced series	t-statistic: -42.83364131968597 p-value: 0.0 Critical Values: 1%: -3.4304845661434378 5%: -2.8615994754339664 10%: -2.5668016568130527	t-statistic: 0.08072046457688913 p-value: 0.1 Critical Values: 1%: 0.739 5%: 0.463 10%: 0.347

can conclude that based on this test, the series is not stationary. Now, we will see if applying log transformation could render the series stationary. As it turns out, we encounter a similar situation where the t-statistic is smaller than the critical value at 5% and the p-value is still larger than 0.05. Hence, we can again deduce that the series is still not stationary. Finally, as a last resort, we apply differencing to the log transformed data and perform the test again. Now, we obtained a completely different result where the t-statistic value is now much lower than the critical value at 5%. And, the p-value is smaller than 0.05. Hence, from this test, we can conclude that the differenced series is stationary.

Now, we take a look at the results given by the KPSS test. Looking closely at the results for the original and the log-transformed series, we can see that both of the p-values are lower than 0.05. This means that we reject the null hypothesis and accept the alternative hypothesis. In other words, we can conclude that the series is not stationary. On the other hand, the resulting p-value for the log transformed and differenced series is larger than 0.05, which means that the series is stationary. From the two tests, we can conclude that in order to render the series stationary we have to 1) apply log transformation to the data and 2) apply first order differencing.

4.4. Anomaly detection

In order to detect anomalies in the data, we took an approach whereby we generate a threshold which defines a lower and upper bounds of the acceptable range in the data and is only applicable to one particular data set. This means that we aim to have one threshold per machine. To generate such threshold, there are a couple of steps that need to be taken

1. Find a stable period.
   We set a window size which denotes the number of records that we take into account. Then, we use a sliding window to find a part of the time series whereby within the selected period of time, there is not much variations. In another word, stable.

2. Make predictions on the selected period.

3. Calculate the threshold and count the number of alerts.
   The threshold was defined as $\pm 3.5 \cdot \text{std}$, or the standard deviation, of the selected period. We tried different values for it but we chose 3.5 to be the suboptimal solution since typically $3 \cdot \text{std}$ is used as a rule of thumb in a general industry problem but the results obtained from using that as a threshold is there are a lot of points which are detected as alerts but we can know for sure that they are not by looking at the resulting plots. Aside from that, we also considered a higher threshold but this could lead to having a lot of false negatives. Long story short, the threshold was found by means of trial and error. Having calculated the threshold, we can then apply it to the predictions to count how many values exceed it.

As a result, there will be two final thresholds generated from two different window sizes and as a final one, we chose the value which leads to the minimum number of detected alerts. Having done so, we can then apply it to the predictions on the development set and count the number of values which are either above/below the thresholds, i.e. anomalies.

5. Results

This section will be divided into three parts; the first part will present the steps taken to make the time series stationary using the ADF and KPSS tests as presented in subsection 3.3. Then, on the second and third parts we present all the results obtained from the univariate and multivariate models, respectively.

5.1. Univariate

We developed 30 ARIMA models for each of the selected features for all six machines. Their parameters were automatically determined by applying auto arima [11]. As for starting point, the initial parameters of an ARIMA model were set to be $(1, 0, 1)$, and the auto arima will adjust these values until it finds the most optimal parameters. More specifically, this function returns the best performing model; one which has the lowest AIC value, with the total training time of 2 hours, 9 minutes and 28.44 seconds. This model was found from the steps that are explained in subsubsection 3.6.1, whereby for each intermediate model, or current model, the function calculates the corresponding AIC value and chooses the model which has the lowest AIC. After that, we make predictions based on the development set and evaluate the models' performances by means of calculating the Mean Absolute Percentage Error (MAPE, for brevity).

5.1.1. Models

Table 5: *ARIMA models for BY machines*

Features	by20	by19
series_Hold pressure peak 1	ARIMA(0, 1, 3)	ARIMA(2, 0, 3)
series_Injection pressure peak 1	ARIMA(0, 1, 3)	ARIMA(3, 0, 3)
series_Flow number 1	ARIMA(1, 1, 1)	ARIMA(3, 0, 3)
series_Clamp force	ARIMA(3, 0, 3)	ARIMA(3, 0, 3)
series_Mold protection time actual value	ARIMA(3, 0, 3)	ARIMA(3, 1, 1)
series_Oil temperature	ARIMA(1, 0, 2)	ARIMA(1, 0, 0)
series_Temperature zone 8	ARIMA(0, 1, 3)	ARIMA(2, 1, 2)

Table 6: *ARIMA models for EN machines*

Features	2406	1622	1805	2605
series_Cycle time last value	ARIMA(3, 0, 3)	ARIMA(2, 0, 3)	ARIMA(2, 0, 2)	ARIMA(2, 0, 0)
series_Injection pressure peak 1	ARIMA(2, 0, 1)	ARIMA(3, 0, 3)	ARIMA(3, 0, 3)	ARIMA(3, 0, 1)
series_Mold protection time actual value	ARIMA(1, 0, 1)	ARIMA(3, 0, 3)	ARIMA(2, 0, 2)	ARIMA(3, 0, 3)
series_Heating energy actual value last cycle	ARIMA(3, 0, 3)	ARIMA(3, 0, 3)	ARIMA(1, 0, 3)	ARIMA(3, 0, 0)

Table 5 and Table 6 show the list of models for all machines. Looking closely at Table 5, it can be seen that in most cases, applying logarithm to the series is enough to make the series stationary. However, there were also some features which require differencing to achieve stationarity for instance hold pressure peak, injection pressure peak, flow number in machine by 20, mold protection time actual value in machine by 19, and temperature zone 8 in both machines. On the other hand, if we refer to Table 6, we can clearly see that all of the models do not need a differencing order. In other words, we resort to using ARMA(p, q) models for all the EN machines.

Furthermore, we can also clearly see from the tables that there are two particular models which are purely AR(p) models, i.e. Oil temperature in machine by19 and heating energy actual value last cycle in machine 2605. This is shown by the fact that they have $d = 0$ and $q = 0$. Hence, in simple terms, these particular series can be modeled by an Autoregressive model of order $p = 1$ and $p = 3$, respectively.

5.1.2. Predictions

The following plots show some results from predictions made by a few of the models using 95% confidence interval. For convenience, we will show one example for each machine and the remaining plots will be shown in Appendix B.

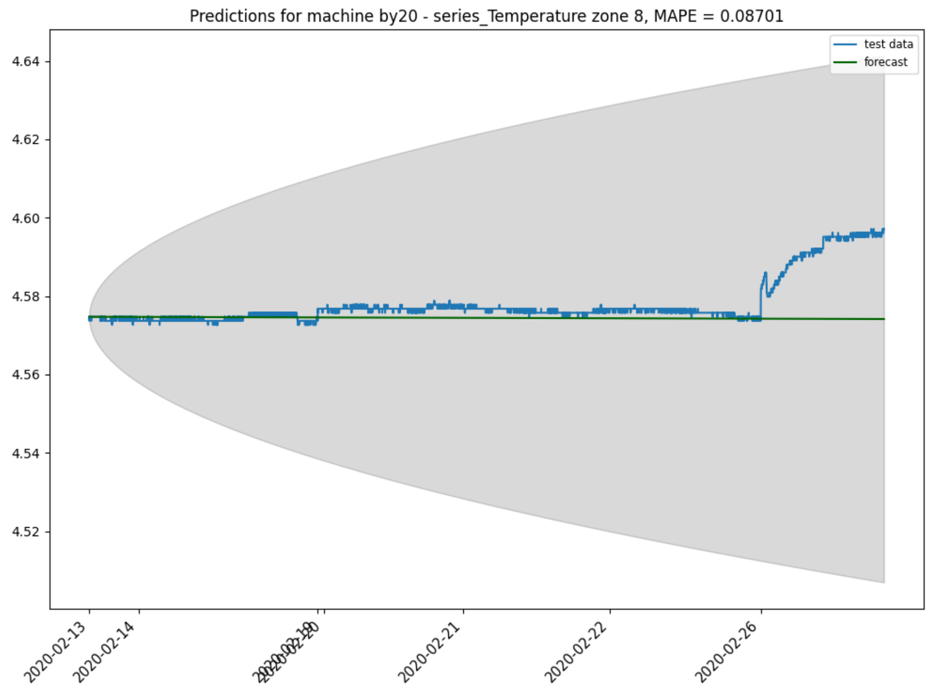


Figure 11: *Predictions made by the model for feature Temperature zone 8 for machine by20*

Figure 11 and Figure 12 show the predictions made for Temperature zone 8 feature in machine by20 and by19 respectively. Coincidentally, they both have the lowest MAPE compared to the other features in the respective machines. Looking closely at Figure 11, we can clearly see that it makes sense for this model to have a low MAPE since the prediction values do not deviate too much from the actual values. In fact, they follow the actual values rather closely but then remain on the predictions' scale while the actual values go up. The MAPE for this model is 0.087% which implies that the model correctly predicts $\sim 99.913\%$ of the actual values and thus one can conclude that this model generates pretty good predictions. On the other hand, we observe something slightly different in Figure 12. In this plot, it can be clearly seen that the predictions seem to have an increasing trend while the actual values do not seem to exhibit such behaviour. This increasing trend in predictions can be inferred to as the model trying to capture the trend of the series and the fact that, the data changes range to a completely different scale on certain time periods. For instance, we can see within the period of 27/01/2020 - 28/01/2020 the values lie within the range of $\pm 4.56 - 4.58$ and then in a couple hours in 28/01/2020 where the range of

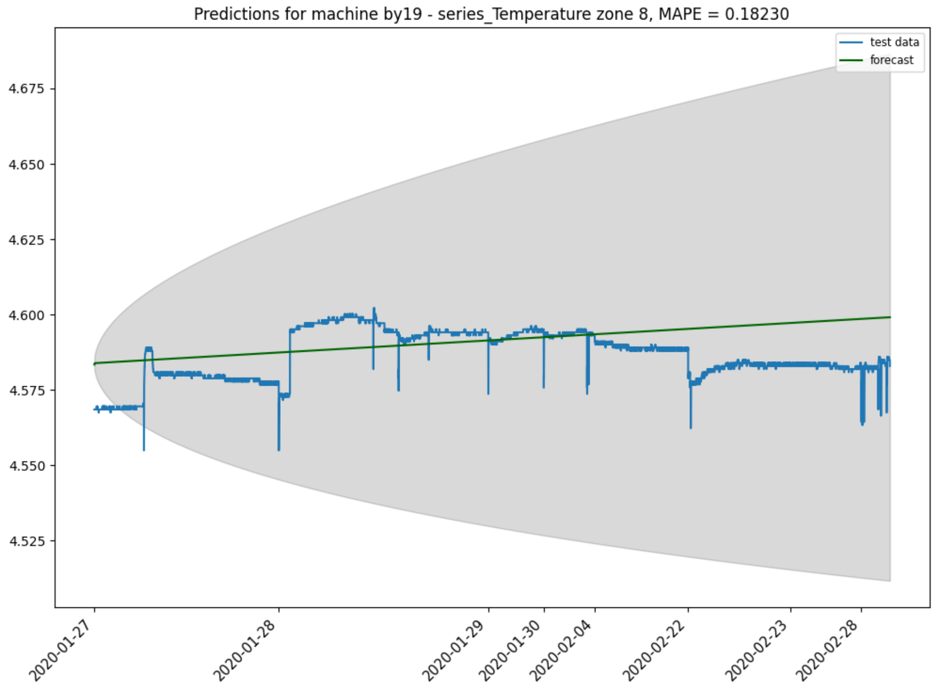


Figure 12: *Predictions made by the model for feature Temperature zone 8 for machine by19*

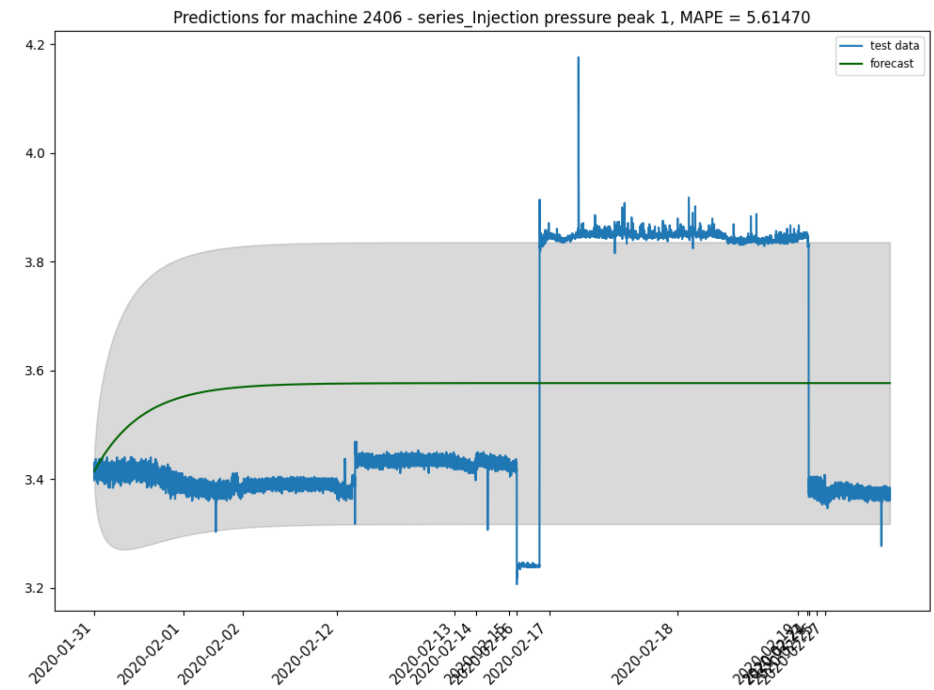


Figure 13: *Predictions made by the model for feature Injection pressure peak for machine 2406*

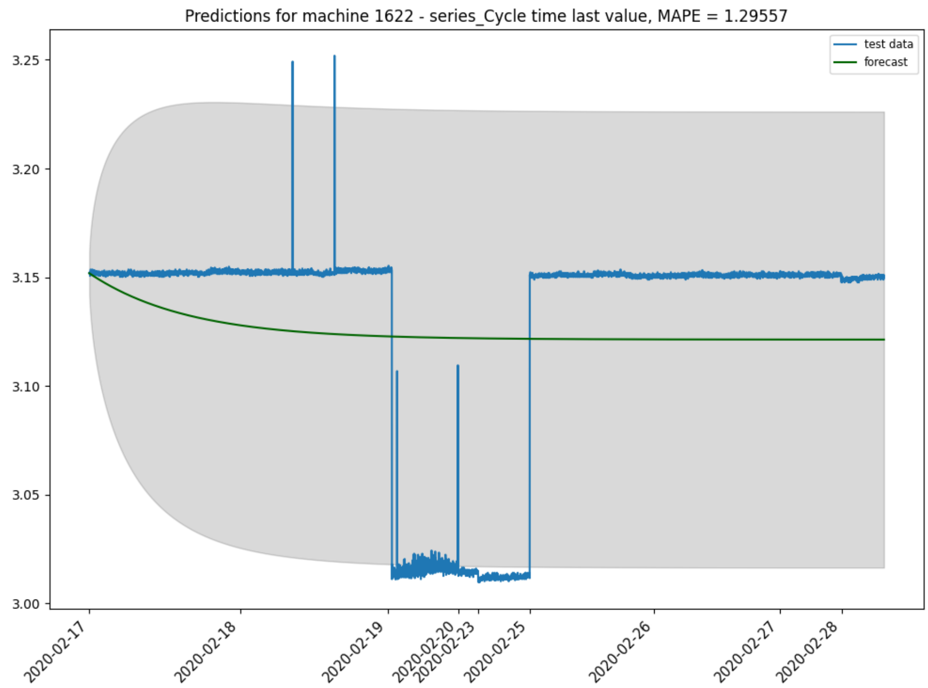


Figure 14: *Predictions made by the model for feature Cycle time last value for machine 1622*

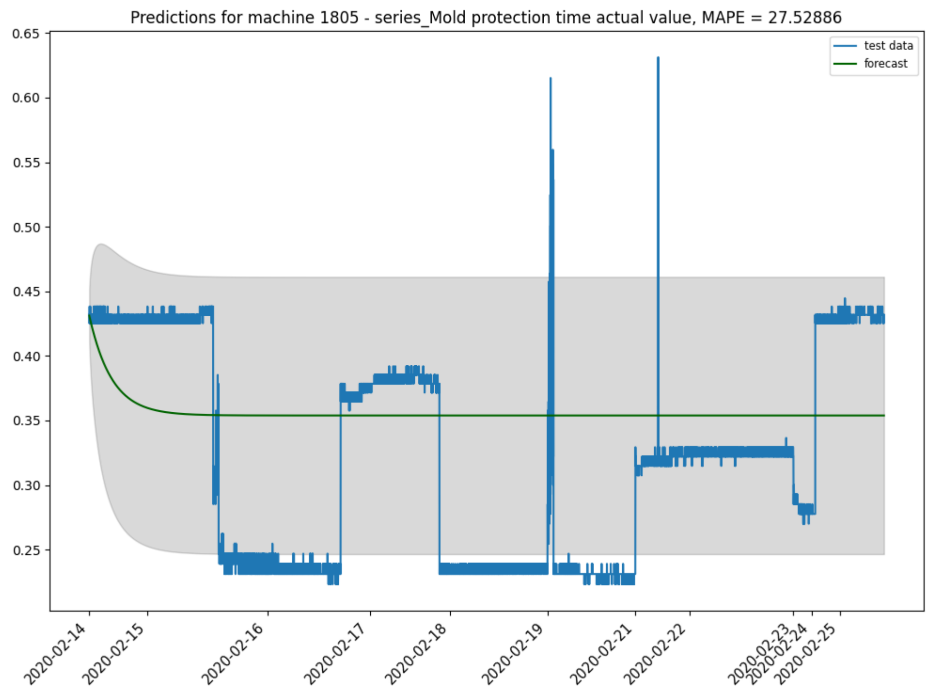


Figure 15: *Predictions made by the model for feature Mold protection time actual value for machine 1805*

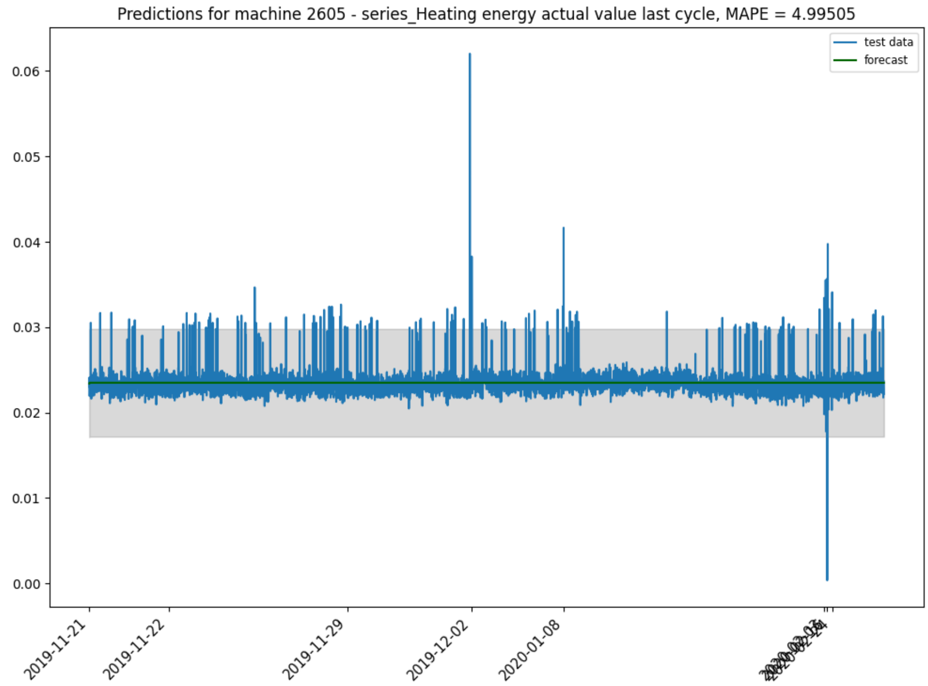


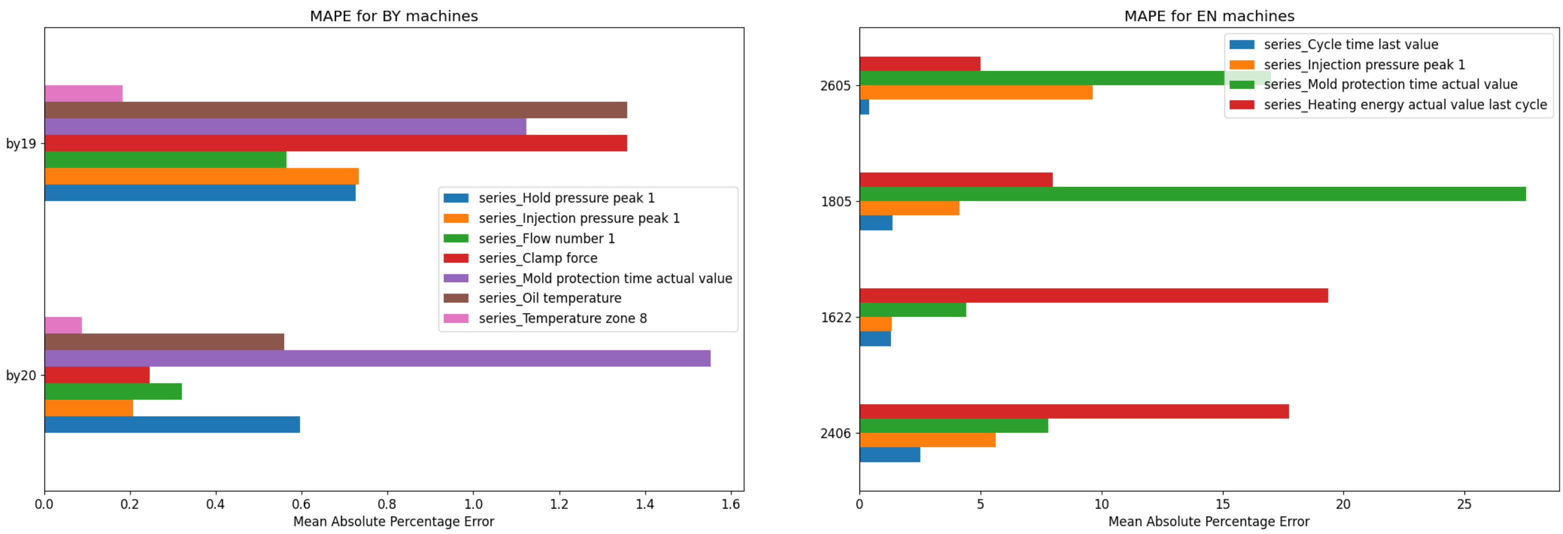
Figure 16: *Predictions made by the model for feature Heating energy actual value last cycle for machine 2605*

values seem to change to be around ± 4.59 or close to ~ 4.6. Albeit the fact that this model also has the lowest MAPE of 0.1823%, the predictions only allow us to deduce that within the next two weeks, the values will keep increasing but they are not able to capture the behaviour of the data.

As for the other four machines, we can observe a similar behaviour from the predictions. Figure 13 shows the predictions made for injection pressure peak for machine 2406. If we look closely at this plot, we can see that there is a huge jump in the actual values which occur roughly at 17/02/2020. However, the prediction values only have a slight increase in the values and they become a kind of plateau. Similar to Figure 12, this increasing trend in the beginning could indicate the model's attempt to capture the huge change in the values' range. Likewise, in Figure 14 we can make the same conclusion except that this time the predictions have an opposing behaviour as before. Instead of having a slight increase there is now a slight decrease. Nevertheless, their MAPEs can still be considered low, i.e. 5.6% and 1.295% respectively. One can argue that although the predictions do not seem to be good, their values still do not deviate much from the actual values. Particularly at a time point where the huge jump or sudden change in range occurs, it can be argued that the predictions are not that far from the actual values considering the fact that these predictions are typically very generic and seem to be yielding values around the mean of the series. Having said that, we can verify this by referring to Figure 15. Looking closely at the plot, it can be clearly seen that in the actual values, not only can we observe them having a change in scale, but it also generates some peaks as can be seen at time point 19/02/2020 and between 21/02/2020 - 22/02/2020. From a hindsight, one can argue that these peaks could have a high probability of being an anomaly. However, this deduction might be inconclusive if we also take into account the prediction values. Furthermore, the MAPE for this model is $\sim 27.53\%$ and it is arguably the highest MAPE compared to all the other models. One obvious reason for this is due to the fact that the model is not able to capture the huge change in ranges within the actual values and the predictions consist of values that are about the mean of the data and most of the times they are a bit off. Although, one can argue that the data exhibits some cyclic behaviour which can be shown by the different patterns and scales in different time intervals. In other words, it can be argued that every time the data changes scale, or makes a huge jump, this can be considered to be a cycle. Moreover, it is also obvious from the plots that the different machines do not have the same cycle time, even if they are from the same brand.

5.1.3. Performance

After generating the predictions as shown by the plots in the previous section, we now need to evaluate the models' performance by means of calculating the Mean Absolute Percentage Error (MAPE).



(a) *MAPE for BY machines.* **(b)** *MAPE for EN machines.*

Figure 17: *Overview of the models' performances in terms of Mean Absolute Percentage Error (MAPE).*

Figure 17a and Figure 17b show the MAPEs of each model for all machines. Looking closely at Figure 17a, it can be clearly seen that on average the MAPEs are quite low. Even the model for mold protection time actual value for machine by20 which has the highest MAPE of $\sim 1.55\%$ amongst the two BY machines, it can still be considered a good model since it correctly predicts $\sim 98.45\%$ of the actual values. By contrast, the model for mold protection time actual value for machine 1805 has the highest MAPE compared to all other five machines, due to the reason mentioned before in which the predictions are somewhat off with respect to the scale of the actual values. In this case, the model correctly predicts $\sim 72.47\%$ of the actual values which is not so great. Yet, the other models can still be considered good on average. Moreover, it can also be seen that the MAPEs for machines 2406 and 1622 seem to be more or less the same or comparable, in the sense that the lowest MAPE being $\sim 2.53\%$ and $\sim 1.3\%$ respectively, are given by the model for cycle time last value. Whereas the highest MAPE being $\sim 17.75789\%$ and $\sim 19.37126\%$ respectively are given by the model for heating energy actual value last cycle. Likewise, machines 1805 and 2605 do also seem to be comparable to each other but not to the other two machines.

Actually, the next step would have been to generate threshold for each of these models in order to detect the anomalies. Yet, if we do so, this would mean that there would be different thresholds to detect, let us say, one anomaly in the data. This however, is not a very practical approach for this particular project. Additionally, the inconsistencies of MAPE also leads us to conclude that if a certain anomaly occurs in the data, it can not be pinpointed to only one specific feature as a cause. Hence, we have to go about tackling this problem in a multivariate way.

5.2. Multivariate

In this approach, we first applied PCA to the data and check how many components we need to explain as much cumulative variance as possible. The reason being, based on the results we obtained from univariate analysis, we now assume that an anomaly that occurs in the data could be caused by a combination of features. This also implies that we may be able to predict when the data would generate some sudden peaks by means of using a multivariate model. Hence, we hypothesize that these features can be expressed in a linear combination of each other whereby each has its own weight given by the absolute value of the eigenvectors.

5.2.1. Models

Table 7: *PCA components and cumulative explained variance*

Machines	Number of PCA components	Cumulative explained variance
by20	1	0.956
2406	1	0.979
1622	1	0.959
by19	1	0.880
1805	1	0.975
2605	1	0.995

Table 8: *ARIMA models for all machines*

Machines	Models
by20	ARIMA(3, 1, 1)
2406	ARIMA(3, 1, 3)
1622	ARIMA(1, 1, 3)
by19	ARIMA(3, 1, 1)
1805	ARIMA(2, 1, 3)
2605	ARIMA(3, 1, 2)

Table 7 shows the results after applying PCA to the training set. As can be clearly seen, we only need one component to explain $> 85\%$ of the cumulative variance of the data. Since PCA yields, essentially, a model with one component, or in other words, another univariate model, therefore we can fit another ARIMA model to this PCA model. To put it simply, we can repeat the steps taken for univariate approach to the resulting PCA models.

Table 8 shows the result of applying auto arima using KPSS test for stationary check, to the PCA models. Looking closely at the table, it can be clearly seen that auto arima yields a set of models that are much different than what we obtained for univariate models as shown in Table 5 and Table 6. After training all models, we now can use it to make predictions based on the development set.

5.2.2. Predictions

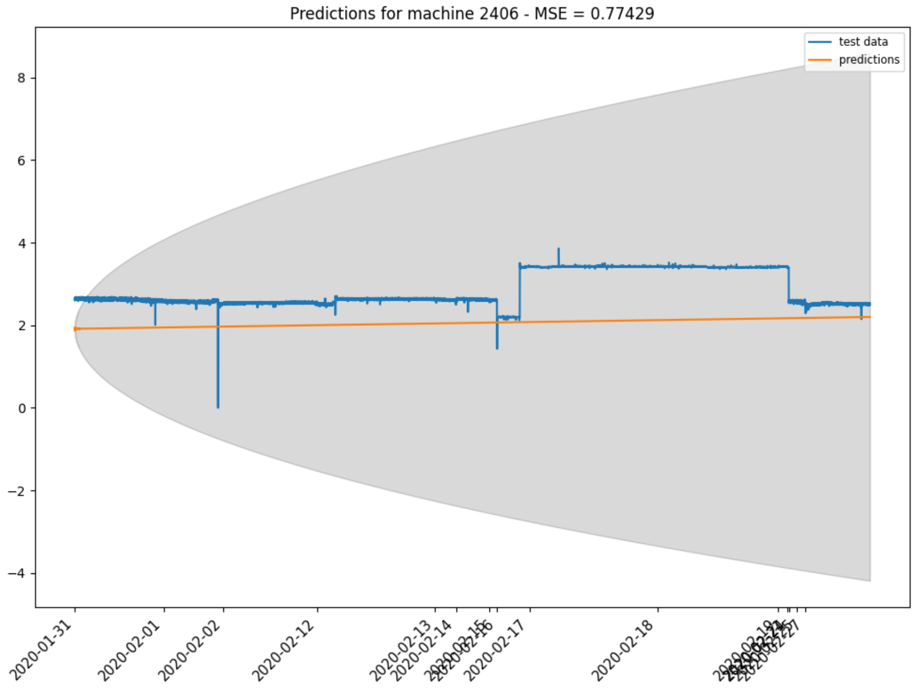


Figure 18: *Prediction for machine 2406*

Figure 18 and Figure 19 show the results of predictions yielded by the models for machines 2406 and by20 respectively. Looking closely at Figure 18, we can see a slight increasing trend in the predictions as shown by the orange line. On the other hand, in Figure 19, the predictions seem to be slightly decreasing. Again, the same observation can be made as in the univariate approach in which this slight increase or decrease in the predictions might be an indication of the sudden change of value range in the actual values. Nonetheless, the fact that these two models have quite low mean squared error (MSE) of 0.774 and 0.161 respectively, indicates that the models are able to forecast future values that are still within the range of the actual data. Yet still, the prediction values seem to still show little to no variance which do not seem to be able to fully capture the nature of the original data.

5.2.3. Performance

For the multivariate approach, the performance of the models are measured by means of calculating the mean squared error (MSE) instead of MAPE. This is because when we applied PCA to the data and log transformation to render the data stationary, the range of values change to somewhere rather close to 0 and therefore MAPE would not be a suitable measure for this case.

Figure 22 shows the summary of the calculated MSEs for all models. Looking closely at the plot, one can clearly see that the model for machine 1805 yields the highest MSE of 2.73 whereas the model for machine by19 yields the lowest MSE of 0.05071. As can be seen from Figure 20, the actual values do not contain any sudden jumps or change of range and the model generates a set of values which are completely within the bounds of the data, albeit with small variations. In addition, we can see in Figure 21 that the data contains many jumps and the model was not

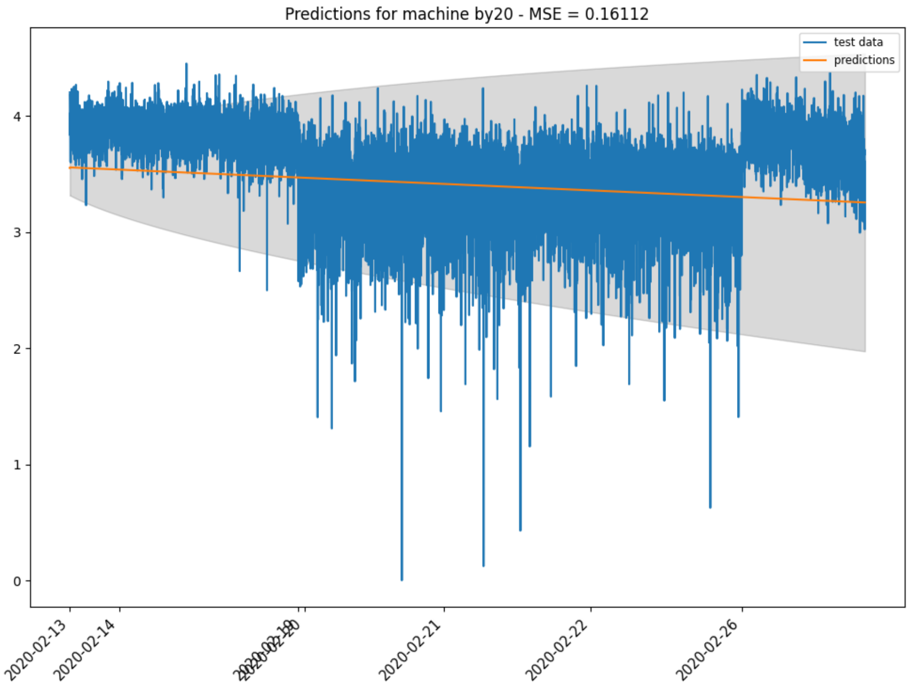


Figure 19: *Prediction for machine by20*

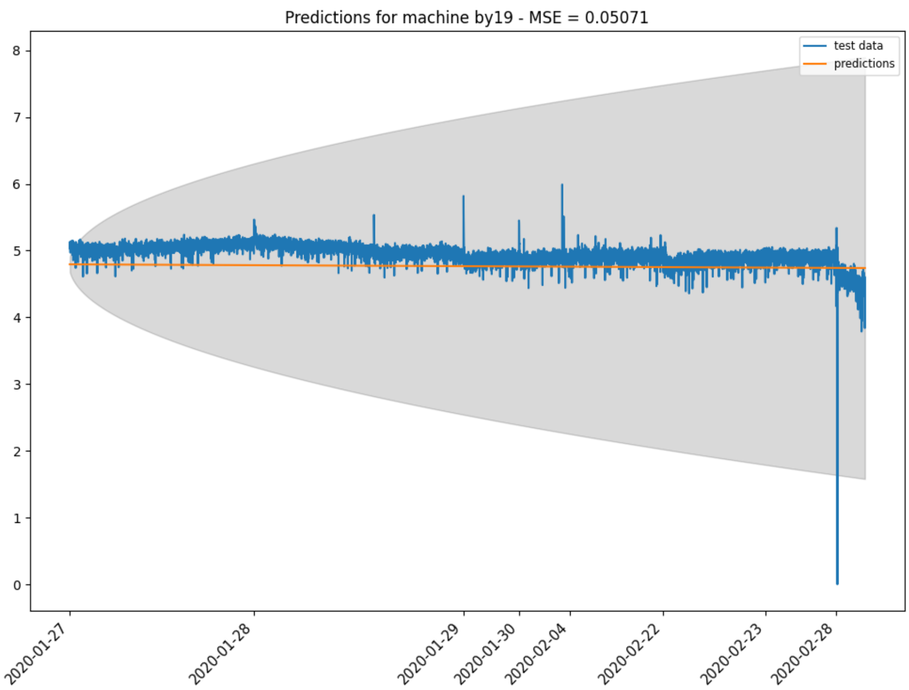


Figure 20: *Prediction for machine by19*

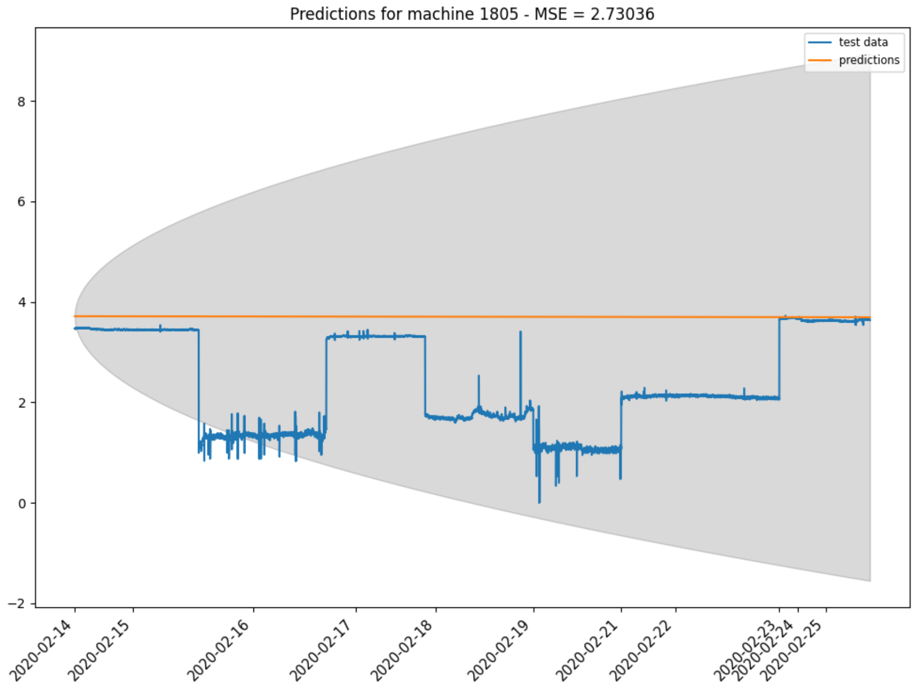


Figure 21: *Prediction for machine 1805*

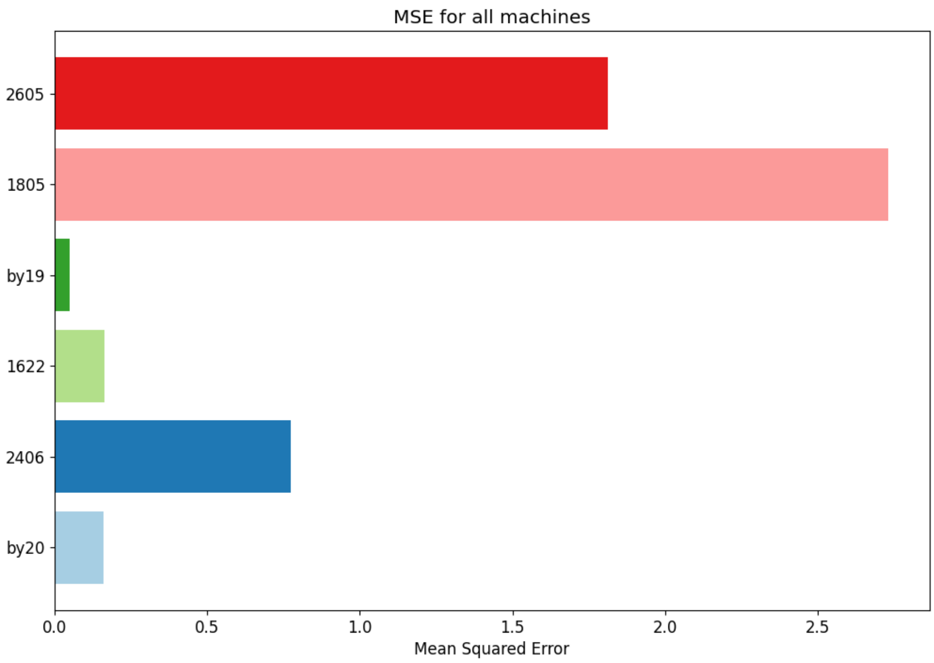


Figure 22: *Mean Squared Error of the predictions for all machines*

able to capture such behaviour. As a result, it only generates values that are closer to the first "cycle" and subsequently yields a rather high MSE. Going back to the MSEs, since on average the models have relatively low MSEs, therefore we consider the models to be good and can be used to generate the thresholds for the next step which is the anomaly detection.

5.2.4. Anomaly detection

As explained in section 4, we only apply this step to the multivariate approach. Hence, after having made the predictions, we can then use the models as shown in Table 8 to generate the threshold which would indicate whether a data point is an anomaly or not. This is decided by means of checking if a certain point in the series is either above or below the threshold as mentioned in subsection 4.4. Subsequently, to detect potential anomalies that are present in the data, we checked if there are values in the series which are either below or above the threshold. If such values exist, we labelled it as anomalies.

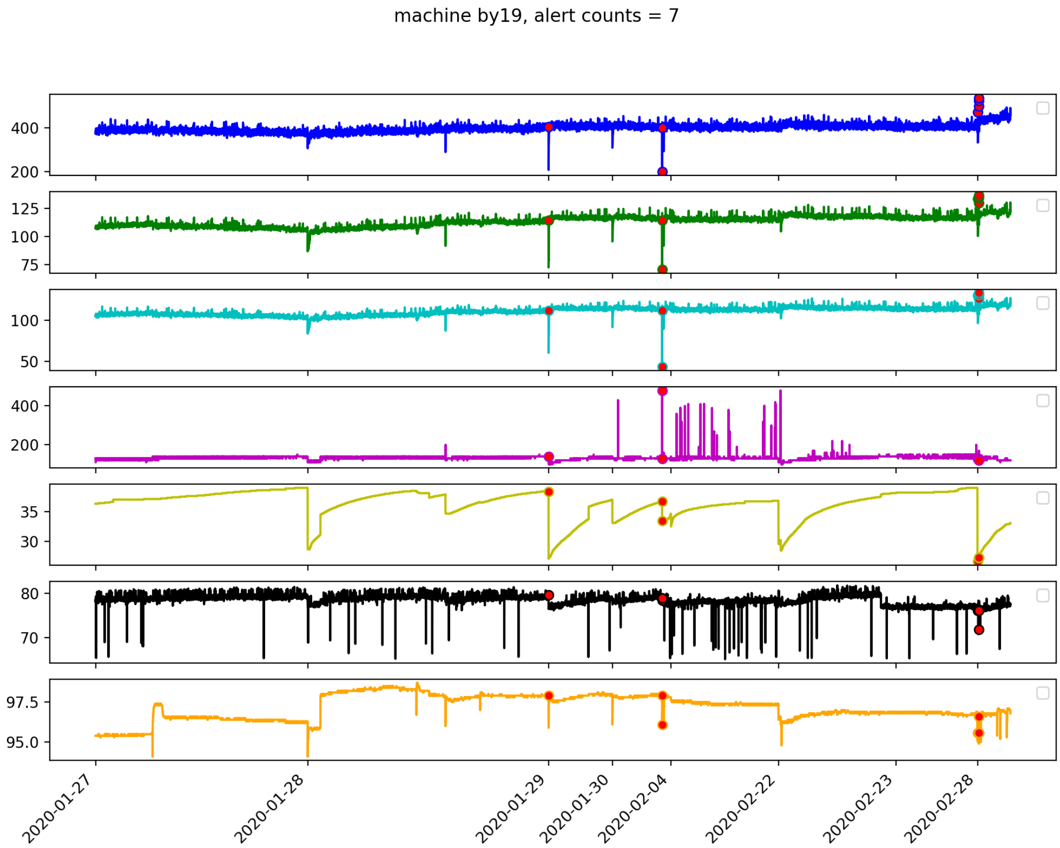


Figure 23: *Detected anomalies for machine by19 for time series of flow number 1, hold pressure peak 1, injection pressure peak 1, mold protection time actual value, oil temperature, temperature zone 8, and clamp force from top to bottom panel, ordered by the eigenvectors, respectively.*

Figure 23 and Figure 24 show the results of anomalies which occur in the data as detected by the respective models. In this plot, we can see all the selected features sorted from highest to lowest with respect to the eigenvectors. This means that Flow number has the highest weight or more influence to causing the anomaly, whereas Clamp force has the lowest weight. Additionally, it can also be seen from the labelled peak which indicates anomalies in the plot, that at around time stamp 04/02/2020, we can observe the data to have a sudden peak particularly in parameters flow

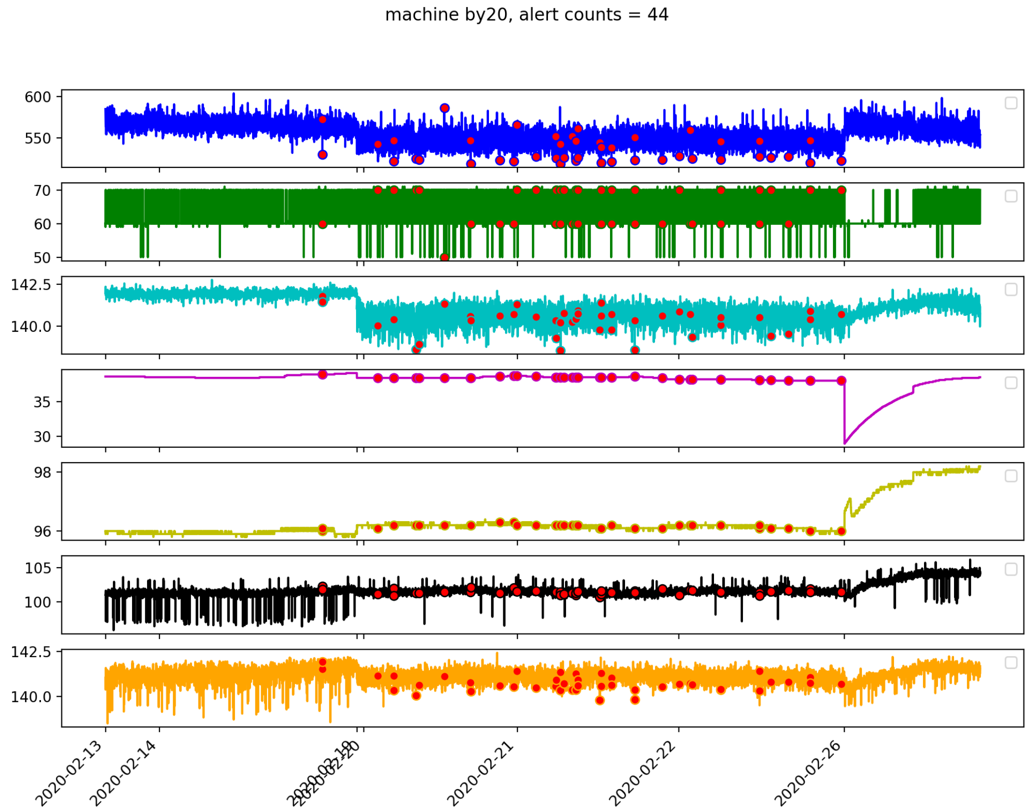


Figure 24: *Detected anomalies for machine by20 for time series of flow number 1, mold protection time actual value, injection pressure peak 1, oil temperature, temperature zone 8, clamp force and hold pressure peak 1, from top to bottom panel, ordered by the eigenvectors, respectively.*

number 1, hold pressure peak 1, injection pressure peak 1 and mold protection actual value. This implies that such alert is most likely to be caused by a change in the set parameter(s) which affect the aforementioned parameters. Hence, one can argue that, only taking the series into account, the anomaly at that time point can be caused by several parameters of which four of them have more influence.

Moving on to Figure 24, it can be clearly seen that the model for machine by20 seems to detect more alerts compared to the previous case. However, the cause of an alert might not be too obvious this time. If we look closely at the plot, even though we might be able to infer something from the flow number parameter, our initial assumption that an anomaly might be caused by a combination of parameters is not necessarily valid in this case since the other parameters do not really allow us to draw any conclusion. To make it more concrete, if we take one of the points of the detected anomaly, we can say that it might be a sudden peak in the flow number parameter but when we look at the Oil temperature, we can not make the same observation. Because, all the detected anomalies in the oil temperature parameter seem to lie on a rather stable series and are less likely to be considered as such. On the same note, it could also be argued that the threshold value could be causing this results. In Figure 19, we notice how the data slightly change scale in the middle. Since the threshold was generated by taking into account the stable period in a given window size, therefore it might be the case that it does not generalize well to the entire series. More specifically, it might be able to detect anomalies rather well with respect to the beginning and/or end of the series as they have arguably same scale, but with respect to the middle period. Furthermore, it is also important to note that even though this two machines come from the same brand and, according to the domain expert, produce the same products and hence should be comparable, they still exhibit somewhat dissimilar behaviour. One way to observe it by means of considering the different order of parameters shown by these two plots. As mentioned previously, in Figure 23 flow number has the highest weight whereas clamp force has the lowest. On the other hand, even though flow number also has the highest weight in Figure 24 but in this case hold pressure peak has the lowest weight. Same statement can also be applied to the rest of the machines whose plots are shown in appendix subsection C.2.

5.2.5. Held-out test data

Having completed phase one of the experiment, now we test if our approach would lead to the same results as what we obtained previously. This means to train new models on held out test data and predict or forecast completely new values in the future.

Table 9: *PCA components and cumulative explained variance on held-out test data*

Machines	Number of PCA components	Cumulative explained variance
by20	1	0.798
2406	1	0.800
1622	1	0.652
by19	1	0.988
1805	1	0.982
2605	1	0.833

Table 10: *ARIMA models for all machines on held-out test data*

Machines	Models
by20	ARIMA(2, 0, 2)
2406	ARIMA(2, 0, 2)
1622	ARIMA(3, 0, 3)
by19	ARIMA(3, 0, 1)
1805	ARIMA(3, 1, 3)
2605	ARIMA(2, 0, 2)

Table 9 shows the result of applying PCA with one component to the held out test data. As can be seen from the table, the explained cumulative variance is not as high as what we observe in the training set shown in Table 7. In a way, this is to be expected since the data is obviously not the same as the training data, nor are they likely to be comparable possibly. Similar to what is done previously, we then fit ARIMA models to these resulting PCA models and the models are shown in Table 10. Comparing these results to Table 8, it is clear that the models are different. One vital observation that we can take from here is that the behaviour of the machines change quite a lot. Not only can we see that from the sudden change in values, or what is presumed to be a cycle, but we can also notice how in the training data, we need ARIMA models for all machines whereas in the held out test data, we only need one ARIMA model for machine 1805 and the rest are ARMA models. Nevertheless, now using the trained models, we can forecast a set of future values and try to see of the models are able to give an indication if an alert will occur. By the same token, since the number of measurements taken per day is never the same, therefore we made a rough estimate of 5000 values (500 for machine 2605 since the data is very small) which is assumed to be equivalent to predicting one day ahead.

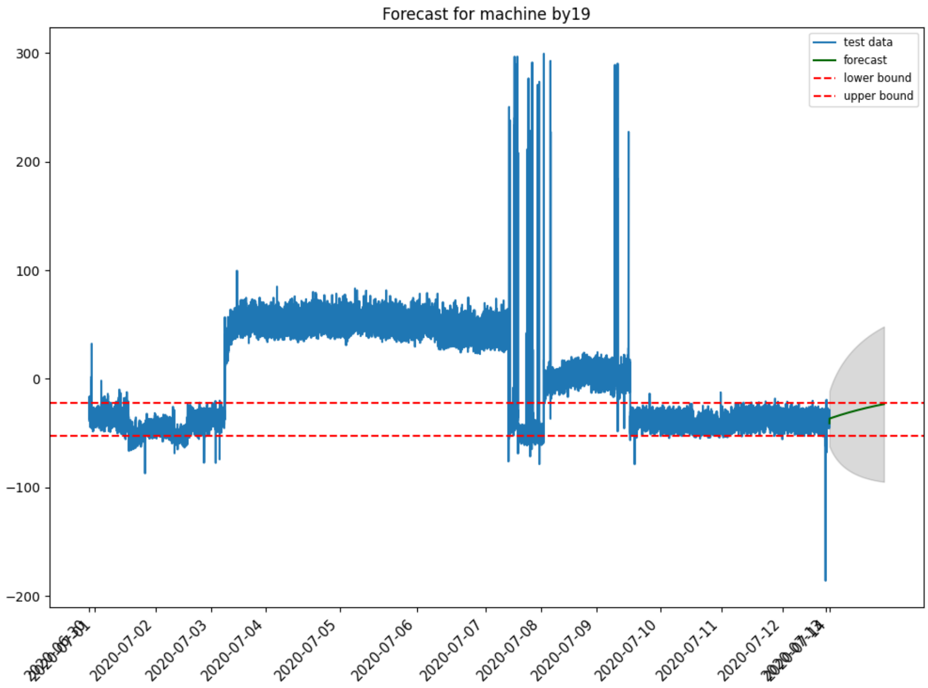


Figure 25: *Forecast for machine by19*

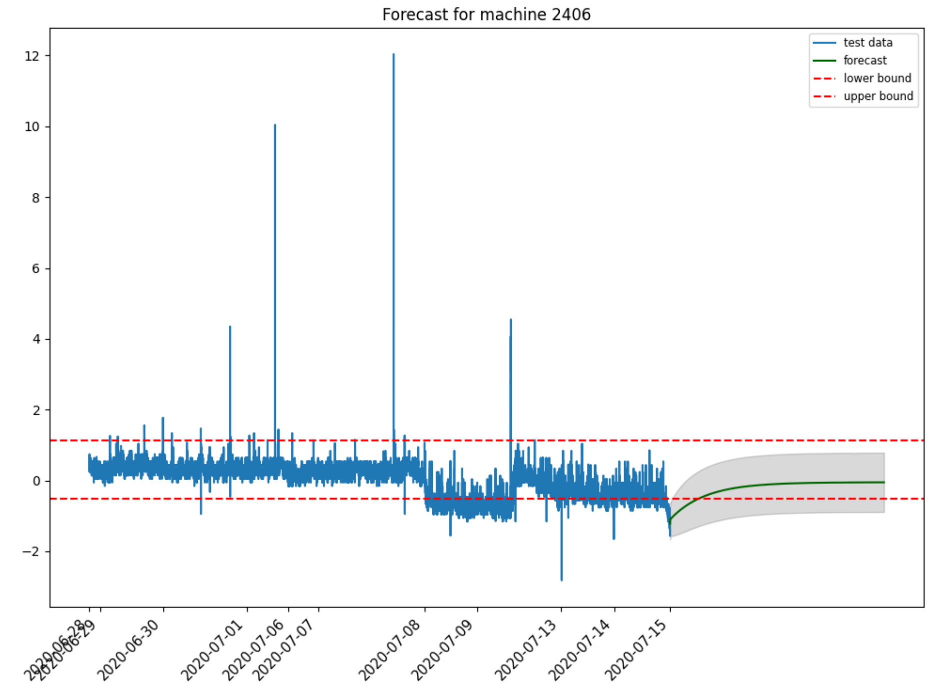


Figure 26: *Forecast for machine 2406*

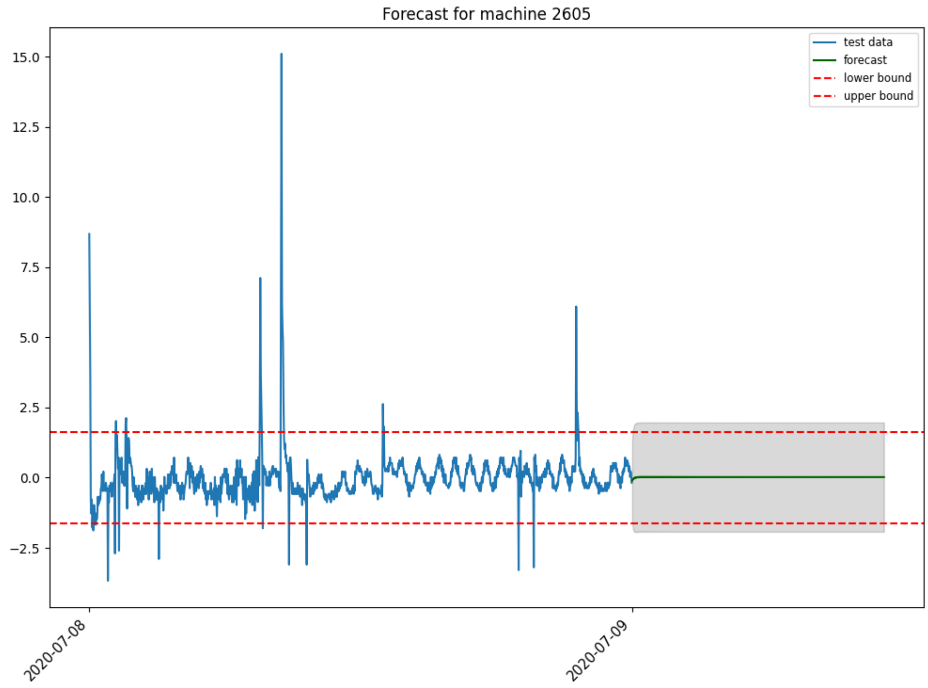


Figure 27: *Forecast for machine 2605*

Figure 25 shows the forecast made by the model for machine by19. Note that in the plot, we can also see two dotted red lines which denote the threshold as lower and upper bounds. Looking closely at the plot, we can see that the predictions show an increasing trend. And at the 5000th value, it hits the upper bound. From here, it can be argued that if we set a larger number of records, i.e. 5500 or 6000, we would expect some of the latter prediction values to exceed the upper bound and hence it would give an alert that there will be anomalies. However, this model can already be used as an indication to take such action. We can already give a warning at the 5000th value and say to check for the selected parameters for machine by19. Likewise, Figure 26 also shows the same behaviour but in the opposite order. Meaning that based on the predictions, the model first recognizes that there will be anomalies since a couple of values lie below the lower bound but then the values will go back to the acceptable, or expected, range. However, we still can not really say that for sure because it could also be the case that the values below the threshold may indicate the current cycle and when they go up, the machine starts a new cycle. Conversely, Figure 70 show neither of the aforementioned observations. Looking closely at the prediction line, we can see that it looks fairly stable and even arguably constant. This could either imply that the model predicts there will not be any anomalies for the next day, or the model is not able to capture the behaviour of the data to predict it. Even though the data looks fairly stationary, there are still some sudden peaks that occur and we could expect the model to at least reflect this behaviour in the predictions.

6. Discussion

This thesis aims to answer the following research question:

To what extent do machines' parameter settings influence the predictions of alerts?

Based on the results that we obtained from the experiment that we carried out, the discussion can be done in terms of a couple point of interests that will be detailed in the following subsections.

6.1. Limitations

Unfortunately, we had no way of verifying these results since, one of the limitation of this project was that, there is no ground truth. Thus, we could not actually count the number of false positives or false negatives with respect to the ground truth. Although this might be considered a big limitation in developing the models, it actually makes sense that there is no ground truth. This is because the machines' behaviour are always changing, sometimes perhaps even unexpectedly. One example to prove this is by looking at the models generated using the training set and held out test data for machines by19 and by20. The auto arima two sets of different models which would be indicative of the fact that it is applied on different sets of data. Another insight to this is that these machines have just recently been moved to a different factory so now there are new operators which are working on them. Different circumstances could affect the way these machines behave and this was one of the reasons why the data was different and the resulting models were also very different. Hence, it makes sense that there is no ground truth to compare our models with because there never was any particular good machine behaviour that occurs at certain time period. On top of that, this is also the reason why first we had to find a stable period in the data while generating the threshold such that we can treat it, in a way, as ground truth. Another impact of the changing behaviour of the machines is also reflected in the inconsistencies of models' performances, especially in terms of the MAPEs.

Some other limitations include lack of background knowledge about the data and the nature of the machines. For instance, if the machines have certain settings, why do they behave in a certain way, and so on. And, in order to verify our results, we could consult our findings with the domain expert or even the operators who work directly with the machines. However, due to the current circumstances caused by the Coronavirus, there was no possibility of doing so since the government enforces the "work-from-home" situation and hence we missed the opportunity of discussing our results more thoroughly.

On a separate note, there are also certain limitations with regards to the methodology chosen for this project. Even though ARIMA model is very commonly used mainly for univariate approach, it has an advantage of doing the analysis only based on the historical time series data with respect to the chosen input variable [18]. The forecasting process or generating the predictions using the pre-trained models can be done simpler since ARIMA models generally ignore any additional input variables. On this note, we can conclude that perhaps using ARIMA models fitted on PCA models might not be a good approach to go about tackling this problem in a multivariate way. Conversely, its disadvantage is that the fitted models do not necessarily allow us to infer any causal inferences from the data. Specifically for this project, this means that even if the models were able to predict a potential alert, we can not derive from the model what can be done to prevent it from happening. And this will require a further dig deep into the project and could become very complex. Having said that, we can deduce based on the results that we obtained, they might not be applicable in practice. Probably, it can be used for some kind of offline analysis where we consider a batch of products and analyze from the data if it could be considered a good or bad batch. It will be useful because then this will eliminate the time allocated to perform manual quality checking, however it is not very useful for predictive maintenance.

6.2. Generated prediction values

We presented the results from univariate approach in subsection 5.1. Based on the results, we discover that the models were able to make predictions in which their values do not deviate a lot from the actual values. This was indicated by the fact that, on average, the models have relatively low MAPEs and hence can be considered as good. Similarly, the multivariate models also demonstrate the same observations regarding the prediction values and their corresponding performance measure, i.e. MSEs. Albeit the fact that models which have low MAPEs or MSEs are often considered to be good representative of the data, these measures do not necessarily guarantee the models' ability to predict future values or capture the behaviour that the data exhibits, i.e. cyclic. This was shown by the predictions that always seem to be constant or have little to no variations. In addition, we also observed that there are inconsistencies in the MAPEs which could be an indication that prediction of alerts based on only one feature or parameter is not necessarily a very practical approach that resembles what usually happens in a production process. Hence, by assuming that we can predict an alert by considering a combination of dissimilar parameters and assume that these parameters can be represented by a linear combination as done in PCA, we obtained a more insightful set of results which confirms our proposed assumption for the multivariate approach.

Furthermore, the results obtained from training new models using held-out test data show that they might be useful in practice. Taking one example shown in Figure 25, the predictions seem to be able to indicate that there could be a potential alert in the future since its last value just reaches the upper bound of the threshold. However, as of its current state, there is no way of finding out which parameters could cause the alert since we have no actual "x" values to compare to as these are completely new sets of values. Although it might be able to be used in practice, the models still have limitations in which the prediction values are indicative of potential alerts due to its little variations which do not seem to represent the data, as well as lack of background knowledge for this particular matter.

One solution to overcome the little variations in the prediction values is we can consider making predictions with respect to a smaller time period in order to account for this cyclic behaviour. This is because the results will most likely be more reliable if we consider more recent observations, which is what we attempted to do in the second phase of the experiment.

6.3. Detected anomalies

Furthermore, taking into account the results of anomalies that occurred in the data which were detected by the models, the two example plots that were shown depict contrasting insights whereby one has very low amount of alerts whereas the other one has quite a lot of anomalies within the span of two weeks. Based on a feedback from the domain expert, we should not look at the number of alerts that the models yield. In Figure 23, it can be argued that the model could potentially have a lot of false negatives which are the data points that should be alerts but were not detected as such. By contrast, in Figure 24, the model might have generated a lot of false positives which are the data points that are not supposed to be alerts but detected as such. We can look closely at the plot and observe that in one of the parameters, some labelled data points appear to lie on a rather stable series which may imply that they are quite unlikely to be considered as alerts. By the same token, the chosen threshold value also appears to highly affect the detected anomalies. Since it was generated by considering a rather stable period within a fixed window size, therefore it might only work rather well if the data has uniform scale throughout, which is unfortunately not the case in our data sets. One possible solution to avoid having a lot of false negatives and false positives is by decreasing or increasing the threshold value respectively. However, this has to be chosen carefully as well as discussed closely with the domain experts.

6.4. Answer to research question

Based on the two approaches, multivariate seems to provide a more practical solution to the problem at hand. We can see from the resulting detected anomalies in the development set that, indeed if a data point turns out to be an anomaly in the data, where typically this is displayed by a sudden peak, its cause could be from not just one specific parameter. Hence, to answer the research question, the extent that machines' parameters have on the influence of the predictions of alerts can be observed by considering a combination of parameters and not only one specific parameter. One can argue that a change in one or multiple different parameters could affect others and hence could result in an unusual behaviour shown by the machines, which confirms our initial assumption for the multivariate approach. Furthermore, this can also be verified by the nature of the data that we observed during feature selection process in which there are many correlations between features in the data. This phenomenon implies that there are many dependencies between features for example there are several temperature parameters which are required to heat up the barrell so as to ensure that it is heated up with a specific temperature. Or also another example was that there were at least two pressure parameters (could be many more), in which their combination is required to do the injection at the right pressure so as to not crush or potentially defect the products.

7. Conclusion

This thesis aims to develop models for time series predictions, specifically for anomaly detection. We considered two approaches namely univariate and multivariate, where in the former we attempted to predict the anomalies by using each of the selected features while the latter involves using a combination of features to do so. We carried out an experiment which was divided into two phases. The first phase concerns developing ARIMA models using the (preprocessed) training data. This data was split into training and development sets where we used the former for feature selection and model fitting while the latter was used to generate predictions and evaluate the models' performances by means of calculating MAPE, for univariate, and MSE, for multivariate. The second phase concerns using held-out test data to train new models and generate predictions using those. The reason being this data contains measurements for two weeks and were considered to be more recent as opposed to training the models using 5 months worth of data.

Our results show that the univariate models seem to have good performances in the predictions as denoted by their low MAPEs. However, it is not a very practical approach which resembles what happens in the actual production environment. Thus, we moved on to the multivariate approach where we observed that a combination of features seems to have more influence and provide more insights to the predictions of alerts. Although again, the models could ordinarily be considered to be good as denoted by their rather low MSEs, the resulting predictions generated by the models did not appear to be very conclusive of detecting any potential alerts. This was shown by the prediction values that seemed to have little to no variations and did not capture the behaviour of the data.

There are many limitations of this project which ranges from the choice of methodology, particularly for the multivariate approach. The obtained results also did not seem to be quite useful in practice and these also needed to be consulted with the domain expert and/or the operators of the machine since we do not have any ground truth which we can use to compare our models with. Another limitation was also in terms of the limited computational resources which could not allow us to implement a more state-of-the-art method. Due to these reasons, we propose a couple of things for future work which are, but not limited to, consider other methodology for multivariate approach which is able to account for the somewhat cyclic behaviour in the data. And, among others, since the results produced by the models that were trained using held-out test data seems to be closer to what can be used in practice, therefore one can try to find a means of mapping the forecasted values back to its original individual features' scale. Because we might expect the operators to not be too familiar with the concept of PCA or multivariate and they only want to know which parameters they should tweak in order not to produce any alerts.

References

[1] Charu C. Aggarwal. "An Introduction to Outlier Analysis". In: *Outlier Analysis*. New York, NY: Springer New York, 2013, pp. 1–40. ISBN: 978-1-4614-6396-2. DOI: 10.1007/978-1-4614-6396-2_1. URL: https://doi.org/10.1007/978-1-4614-6396-2_1.

[2] Charu C. Aggarwal. "Data Preparation". In: *Data Mining: The Textbook*. Cham: Springer International Publishing, 2015, pp. 27–62. ISBN: 978-3-319-14142-8. DOI: 10.1007/978-3-319-14142-8_2. URL: https://doi.org/10.1007/978-3-319-14142-8_2.

[3] Venkatesh Bachu and J. Anuradha. "A Review of Feature Selection and Its Methods". In: *Cybernetics and Information Technologies* 19 (Mar. 2019), p. 3. DOI: 10.2478/cait-2019-0001.

[4] Christian Beecks, Fabian Berns, Alexander Grass, and Kjeld Willy Schmidt. "Gaussian Processes for Anomaly Description in Production Environments". In: *Proceedings of the Workshops of the EDBT/ICDT 2019 Joint Conference EDBT/ICDT 2019 Lisbon Portugal March 26 2019* 2322, 2019 (2019). URL: http://ceur-ws.org/Vol-2322/dsi4-4.pdf.

[5] Ane Blázquez-García, Angel Conde, Usue Mori, and Jose A. Lozano. *A review on outlier/anomaly detection in time series data*. 2020. arXiv: 2002.04236 [cs.LG].

[6] Jonathan D. Cryer and Kung-sik Chan. "Time series analysis with applications in R". In: 2nd ed. Springer, 2011. DOI: 10.1007/978-0-387-75959-3.

[7] Nan Ding, Huanbo Gao, Hongyu Bu, Haoxuan Ma, and Huaiwei Si. "Multivariate-Time-Series-Driven Real-time Anomaly Detection Based on Bayesian Network". In: *Sensors* 18.10 (Oct. 2018), p. 3367. ISSN: 1424-8220. DOI: 10.3390/s18103367. URL: http://dx.doi.org/10.3390/s18103367.

[8] Jingjing Fei and Shiliang Sun. "Online Anomaly Detection with Sparse Gaussian Processes". In: *CoRR* abs/1905.05761 (2019). arXiv: 1905.05761. URL: http://arxiv.org/abs/1905.05761.

[9] Isabelle Guyon and André Elisseeff. "An Introduction to Variable and Feature Selection". In: *J. Mach. Learn. Res.* 3.null (Mar. 2003), pp. 1157–1182. ISSN: 1532-4435.

[10] Rob J. Hyndman and Anne B. Koehler. "Another look at measures of forecast accuracy". In: *International Journal of Forecasting* 22.4 (2006), pp. 679–688. ISSN: 0169-2070. DOI: https://doi.org/10.1016/j.ijforecast.2006.03.001. URL: http://www.sciencedirect.com/science/article/pii/S0169207006000239.

[11] Rob Hyndman and Yeasmin Khandakar. "Automatic Time Series Forecasting: The forecast Package for R". In: *Journal of Statistical Software, Articles* 27.3 (2008), pp. 1–22. ISSN: 1548-7660. DOI: 10.18637/jss.v027.i03. URL: https://www.jstatsoft.org/v027/i03.

[12] Tim Isbister. *Anomaly detection on social media using ARIMA models*. 2015.

[13] Sungil Kim and Heeyoung Kim. "A new metric of absolute percentage error for intermittent demand forecasts". In: *International Journal of Forecasting* 32 (July 2016), pp. 669–679. DOI: 10.1016/j.ijforecast.2015.12.003.

[14] Vipin Kumar. "Parallel and Distributed Computing for Cybersecurity". In: *Distributed Systems Online, IEEE* 6 (Feb. 2005). DOI: 10.1109/MDSO.2005.53.

[15] Athanasios Naskos, Anastasios Gounaris, Ifigeneia Metaxa, and Daniel Köchling. "Detecting Anomalous Behavior Towards Predictive Maintenance". In: *Advanced Information Systems Engineering Workshops*. Ed. by Henderik A. Proper and Janis Stirna. Cham: Springer International Publishing, 2019, pp. 73–82. ISBN: 978-3-030-20948-3.

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

[17] "Mean Squared Error". In: *Encyclopedia of Machine Learning*. Ed. by Claude Sammut and Geoffrey I. Webb. Boston, MA: Springer US, 2010, pp. 653–653. ISBN: 978-0-387-30164-8. DOI: 10.1007/978-0-387-30164-8_528. URL: https://doi.org/10.1007/978-0-387-30164-8_528.

[18] EMC Services. "Advanced Analytical Theory and Methods: Time Series Analysis". In: Aug. 2015, pp. 233–254. DOI: 10.1002/9781119183686.ch8.

[19] Shahroz Tariq, Sangyup Lee, Youjin Shin, Myeong Shin Lee, Okchul Jung, Daewon Chung, and Simon S. Woo. "Detecting Anomalies in Space Using Multivariate Convolutional LSTM with Mixtures of Probabilistic PCA". In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. KDD '19. Anchorage, AK, USA: Association for Computing Machinery, 2019, pp. 2123–2133. ISBN: 9781450362016. DOI: 10.1145/3292500.3330776. URL: https://doi.org/10.1145/3292500.3330776.

[20] Wudi Wei, Junjun Jiang, Hao Liang, Lian Gao, Bingyu Liang, Jiegang Huang, Ning Zang, Yanyan Liao, Jun Yu, Jingzhen Lai, Fengxiang Qin, Jinming Su, Li Ye, and Hui Chen. "Application of a Combined Model with Autoregressive Integrated Moving Average (ARIMA) and Generalized Regression Neural Network (GRNN) in Forecasting Hepatitis Incidence in Heng County, China". In: *PLOS ONE* 11 (June 2016), e0156768. DOI: 10.1371/journal.pone.0156768.

[21] Chuxu Zhang, Dongjin Song, Yuncong Chen, Xinyang Feng, Cristian Lumezanu, Wei Cheng, Jingchao Ni, Bo Zong, Haifeng Chen, and Nitesh V. Chawla. "A Deep Neural Network for Unsupervised Anomaly Detection and Diagnosis in Multivariate Time Series Data". In: *CoRR* abs/1811.08055 (2018). arXiv: 1811.08055. URL: http://arxiv.org/abs/1811.08055.

[22] Yue Zhao, Zain Nasrullah, and Zheng Li. "PyOD: A Python Toolbox for Scalable Outlier Detection". In: *Journal of Machine Learning Research* 20.96 (2019), pp. 1–7. URL: http://jmlr.org/papers/v20/19-011.html.

[23] Eric Zivot and Jiahui Wang. *Modeling Financial Time Series with S-PLUS®*. Berlin, Heidelberg: Springer-Verlag, 2006. ISBN: 0387279652.

A. Features

Table 11: *Features names for BY machines*

by20 and by19	
Cycle time last value	Cycle time last value
Hold pressure peak 1	Hold pressure peak 1
Injection pressure peak 1	Injection pressure peak 1
Material consumption 1	Material consumption 1
Screw position 1	Screw position 1
Flow number 1	Flow number 1
Injection time 1	Injection time 1
Switchover pressure 1	Switchover pressure 1
Melt cushion 1	Melt cushion 1
Injection pressure peak value	Injection pressure peak value
Plasticizing position end - corrected	Plasticizing position end - corrected
Clamp force	Clamp force
Mold protection time actual value	Mold protection time actual value
Oil temperature	Oil temperature
Temperature zone 1	Temperature zone 1
Temperature zone 2	Temperature zone 2
Temperature zone 3	Temperature zone 3
Temperature zone 4	Temperature zone 4
Temperature zone 5	Temperature zone 5
Temperature zone 7	Temperature zone 7
Temperature zone 8	Temperature zone 8
Temperature zone 9	Temperature zone 9
Temperature zone 10	Temperature zone 10
Injection pressure	Injection pressure
Hydr. pressure at switch over	Hydr. pressure at switch over
Switchover position actual value	Switchover position actual value

Table 12: *Features names for EN machines*

2406	1622	1805	2605
Cycle time last value Hold pressure peak 1 Injection pressure peak 1 Back pressure peak 1 Screw position 1 Flow number 1 Injection time 1 Plasticizing time 1 Switchover pressure 1 Melt cushion 1 Injection pressure peak value Plasticizing position end - corrected Clamp force Mold protection time actual value Oil temperature Temperature zone 1 Temperature zone 2 Temperature zone 3 Temperature zone 4 Temperature zone 7 Temperature zone 8 Temperature zone 9 Temperature zone 10 Temperature zone 11 Temperature zone 12 Temperature zone 13 Temperature zone 14 Temperature zone 15 Temperature zone 16 Heating energy actual value last cycle Heating energy consumption Motor energy actual value last cycle Motor energy consumption Total energy actual value last cycle Total energy consumption Hold pressure peak 2 Injection pressure peak 2 Back pressure peak 2 Screw position 2 Flow number 2 Injection time 2 Plasticizing time 2 Switchover pressure 2 Melt cushion 2	Cycle time last value Hold pressure peak 1 Injection pressure peak 1 Back pressure peak 1 Screw position 1 Flow number 1 Injection time 1 Plasticizing time 1 Switchover pressure 1 Melt cushion 1 Injection pressure peak value Plasticizing position end - corrected Clamp force Mold protection time actual value Oil temperature Temperature zone 1 Temperature zone 2 Temperature zone 3 Temperature zone 4 Temperature zone 5 Temperature zone 6 Temperature zone 7 Temperature zone 8 Temperature zone 9 Temperature zone 10 Temperature zone 11 Temperature zone 12 Temperature zone 13 Temperature zone 14 Temperature zone 15 Temperature zone 16 Heating energy actual value last cycle Heating energy consumption Motor energy actual value last cycle Motor energy consumption Total energy actual value last cycle Total energy consumption	Cycle time last value Hold pressure peak 1 Injection pressure peak 1 Back pressure peak 1 Screw position 1 Flow number 1 Injection time 1 Plasticizing time 1 Switchover pressure 1 Melt cushion 1 Injection pressure peak value Plasticizing position end - corrected Clamp force Mold protection time actual value Oil temperature Temperature zone 1 Temperature zone 2 Temperature zone 3 Temperature zone 4 Temperature zone 5 Temperature zone 7 Temperature zone 8 Temperature zone 9 Temperature zone 10 Temperature zone 11 Temperature zone 12 Temperature zone 13 Temperature zone 14 Temperature zone 15 Set temperature zone 1 Set temperature zone 2 Set temperature zone 3 Set temperature zone 4 Set temperature zone 5 Set temperature zone 8 Set temperature zone 9 Set temperature zone 10 Set temperature zone 11 Set temperature zone 12 Set temperature zone 13 Heating energy actual value last cycle Heating energy consumption Motor energy actual value last cycle Motor energy consumption Total energy actual value last cycle Total energy consumption	Cycle time last value Hold pressure peak 1 Injection pressure peak 1 Back pressure peak 1 Screw position 1 Flow number 1 Injection time 1 Plasticizing time 1 Switchover pressure 1 Injection pressure peak value Plasticizing position end - corrected Clamp force Mold protection time actual value Temperature zone 1 Temperature zone 2 Temperature zone 3 Temperature zone 4 Temperature zone 5 Temperature zone 7 Temperature zone 8 Temperature zone 9 Temperature zone 10 Temperature zone 11 Temperature zone 12 Temperature zone 13 Temperature zone 14 Temperature zone 15 Temperature zone 16 Heating energy actual value last cycle Heating energy consumption Motor energy actual value last cycle Motor energy consumption Total energy actual value last cycle Total energy consumption Hold pressure peak 2 Injection pressure peak 2 Back pressure peak 2 Screw position 2 Flow number 2 Injection time 2 Plasticizing time 2 Switchover pressure 2 Melt cushion 2

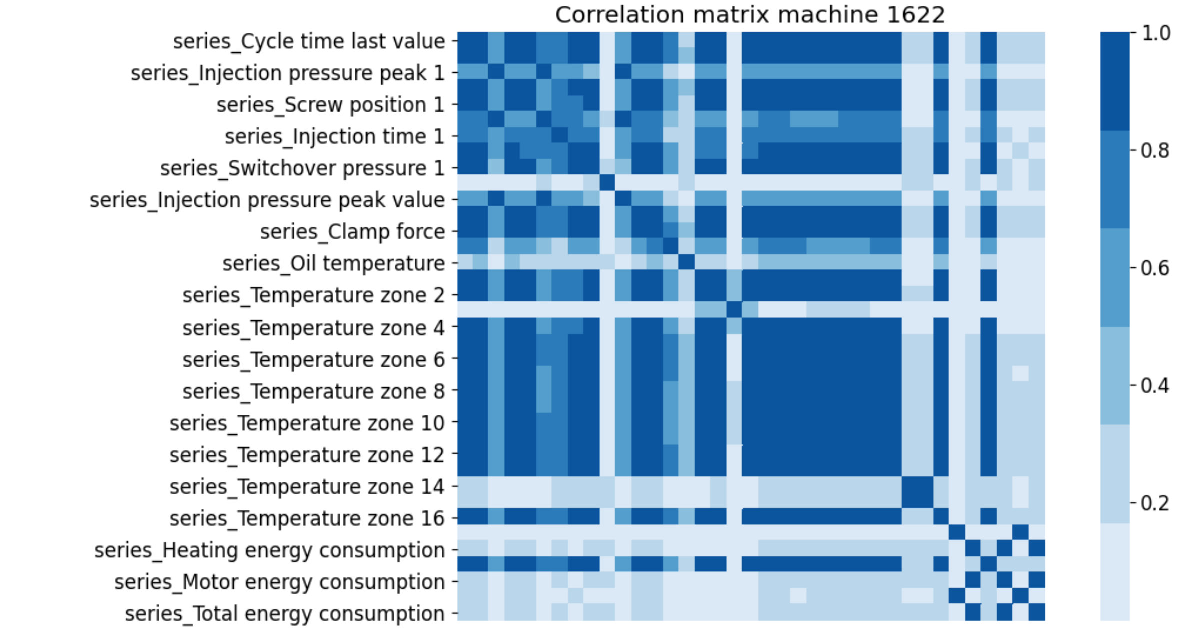


Figure 28: *Correlation matrix for machine 1622*

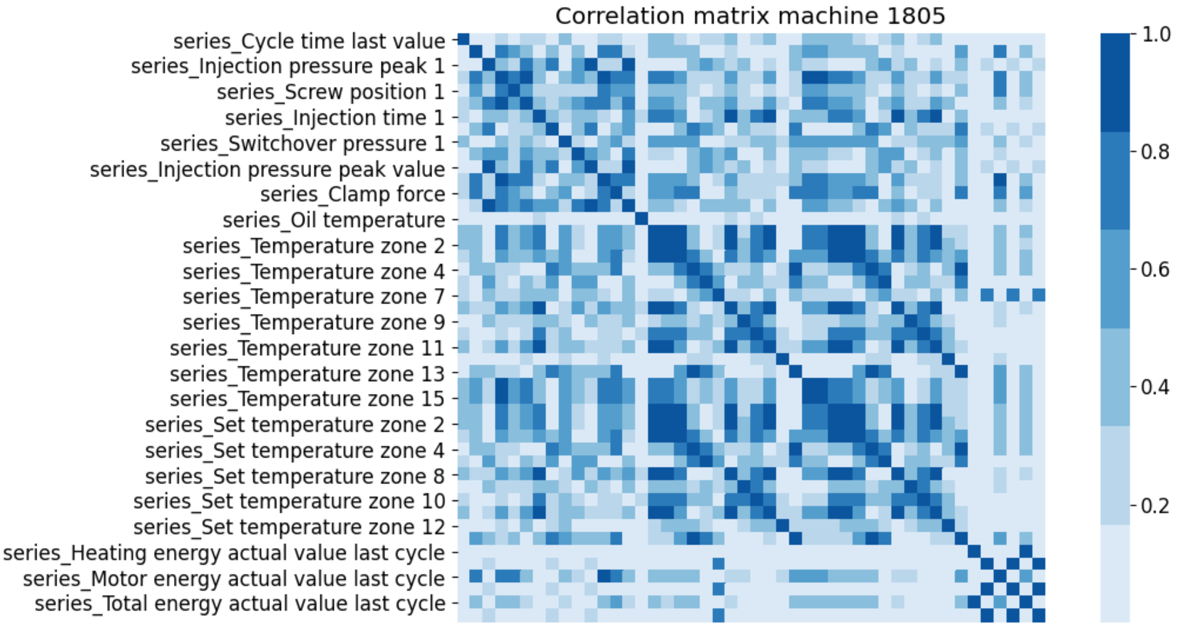


Figure 29: *Correlation matrix for machine 1805*

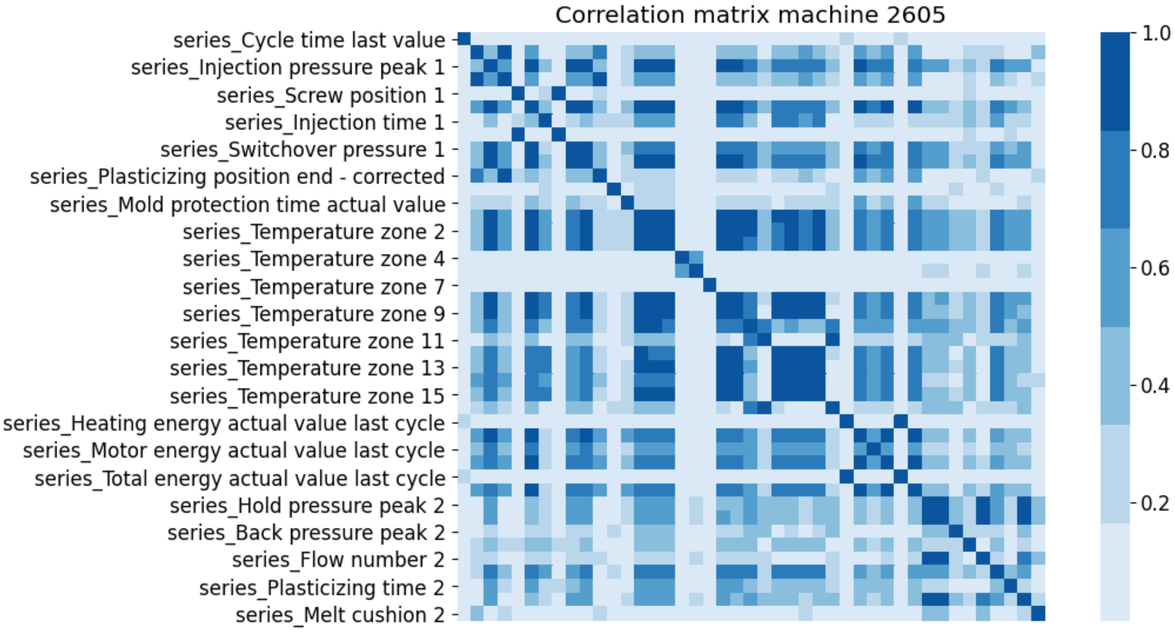


Figure 30: *Correlation matrix for machine 2605*

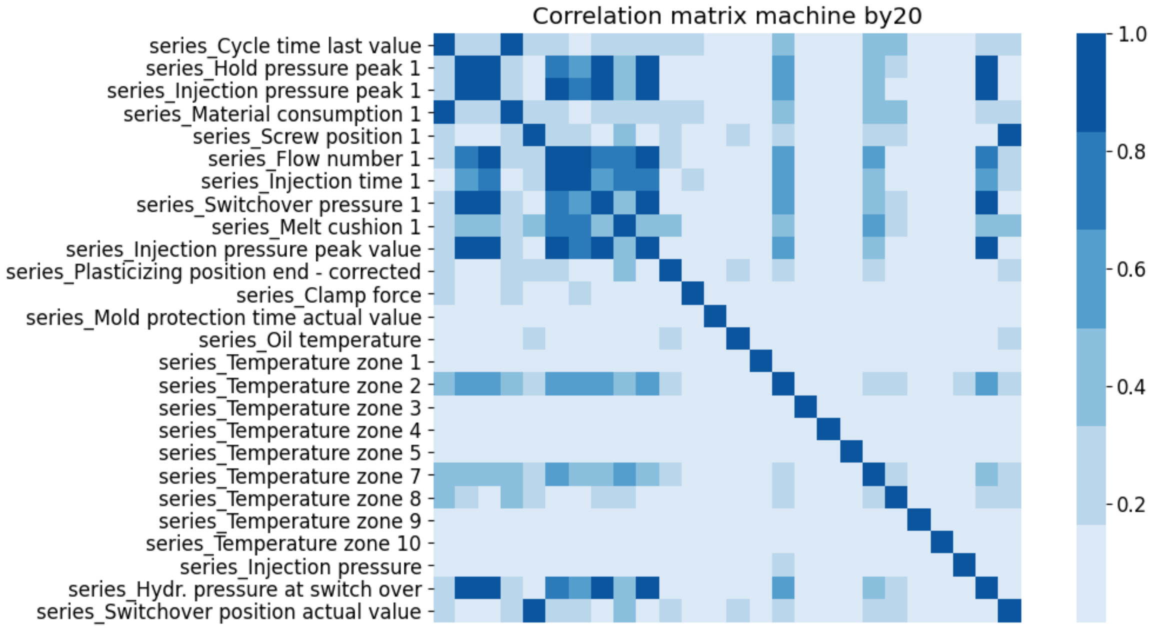


Figure 31: *Correlation matrix for machine by20*

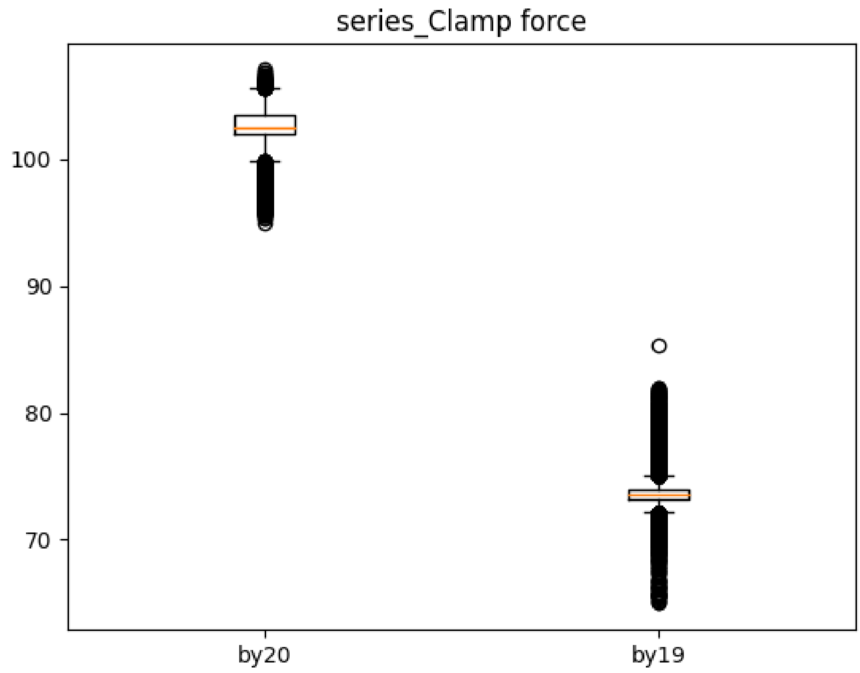


Figure 32: *Boxplot for Clamp force feature of BY machines*

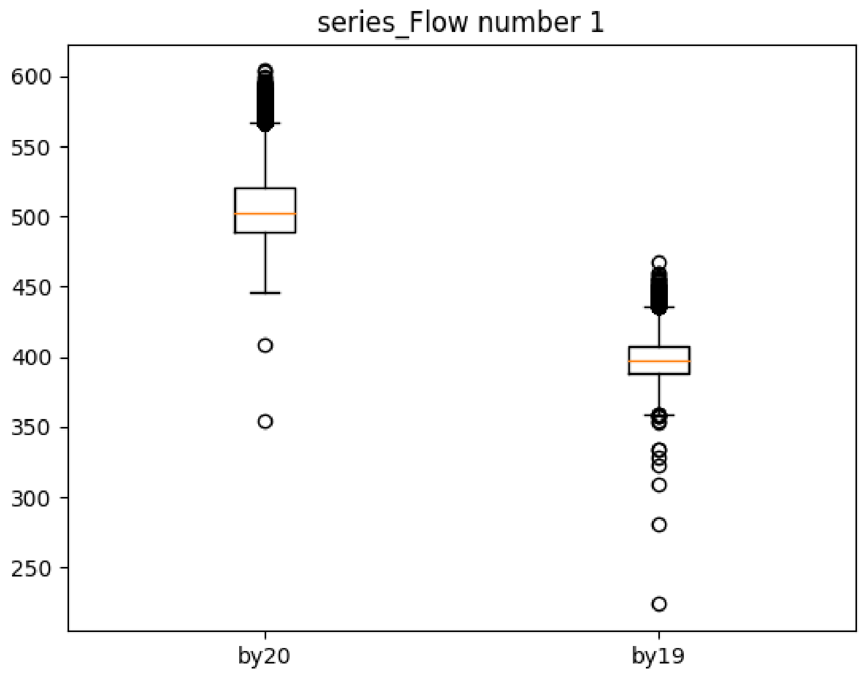


Figure 33: *Boxplot for Flow number feature of BY machines*

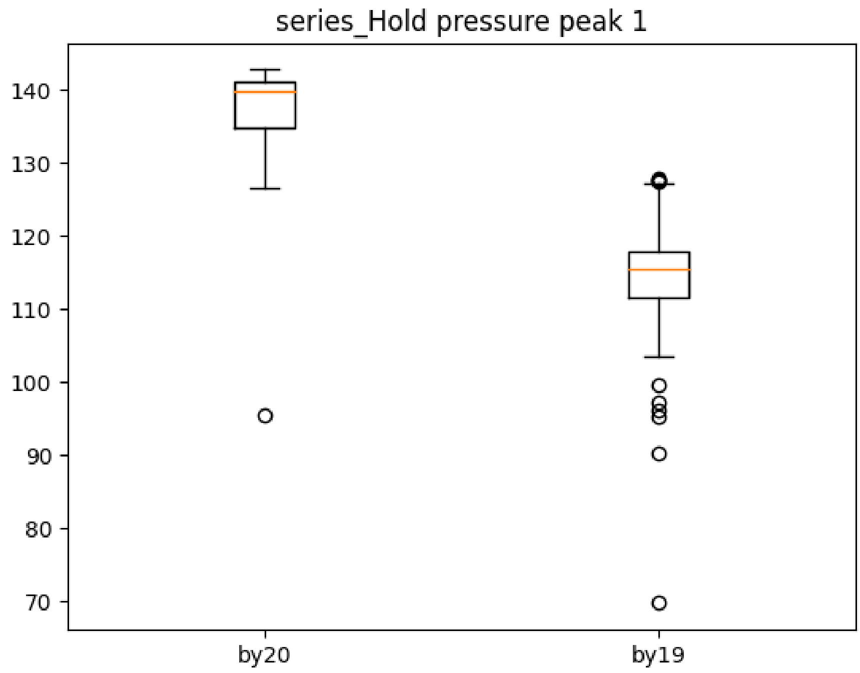


Figure 34: *Boxplot for Hold pressure peak feature of BY machines*

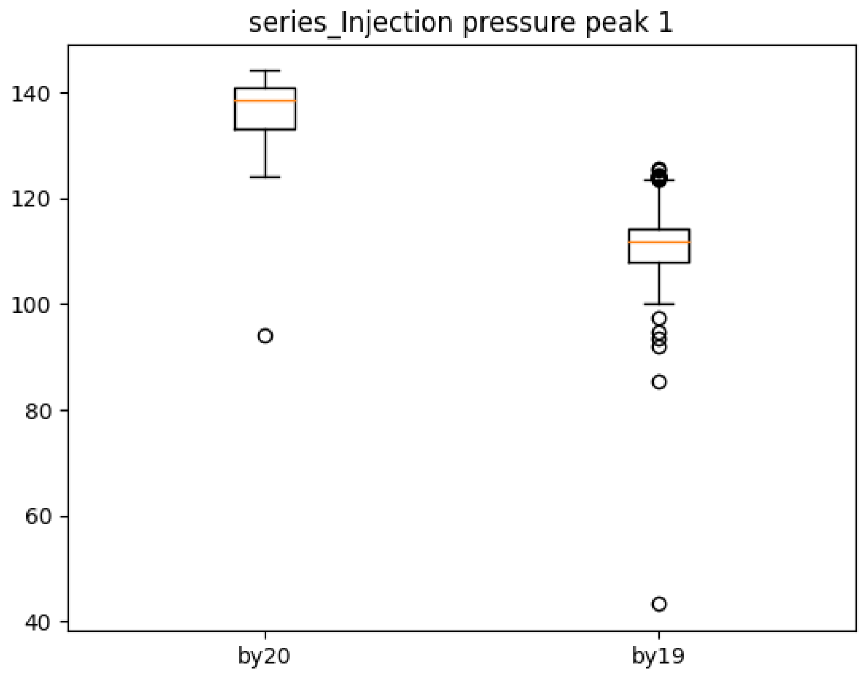


Figure 35: *Boxplot for Injection pressure peak feature of BY machines*

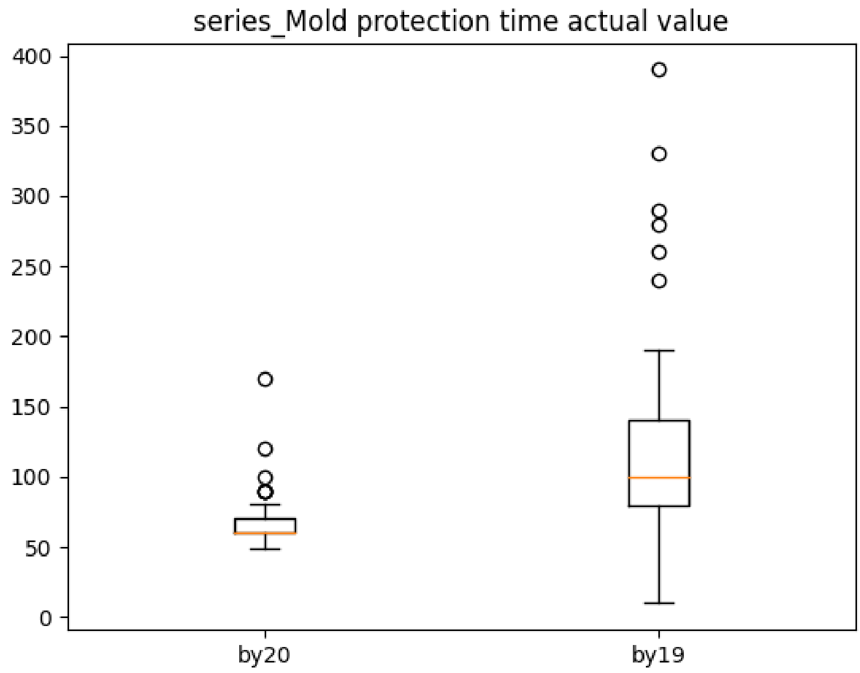


Figure 36: *Boxplot for Mold protection time actual value feature of BY machines*

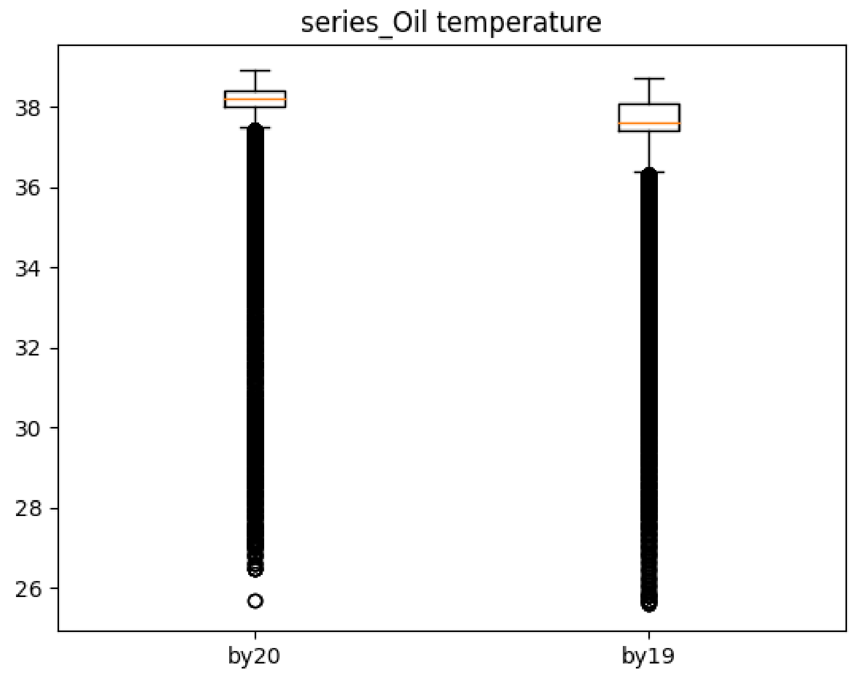


Figure 37: *Boxplot for Oil temperature feature of BY machines*

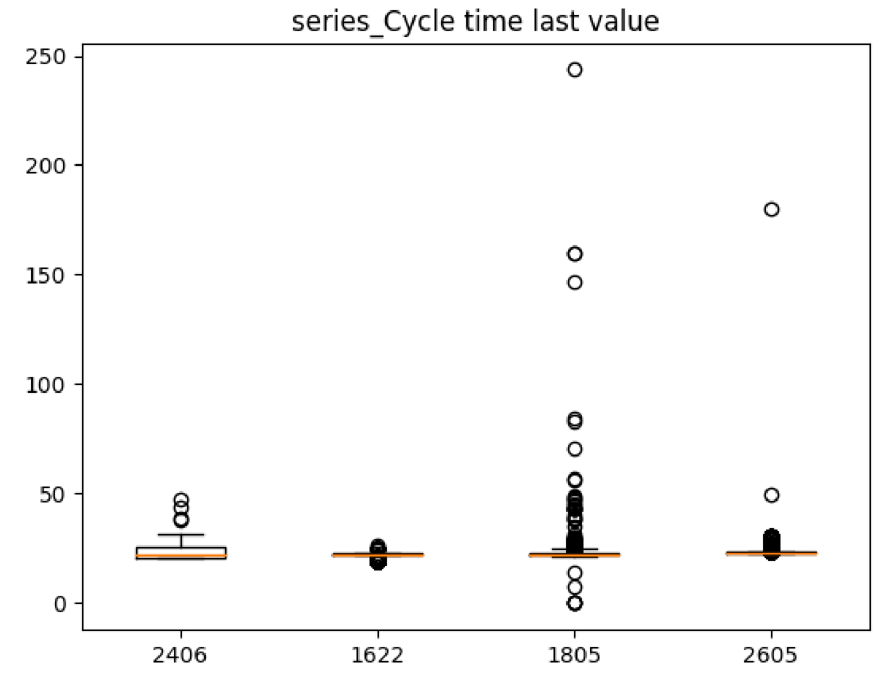


Figure 38: *Boxplot for Cycle time last value feature of EN machines*

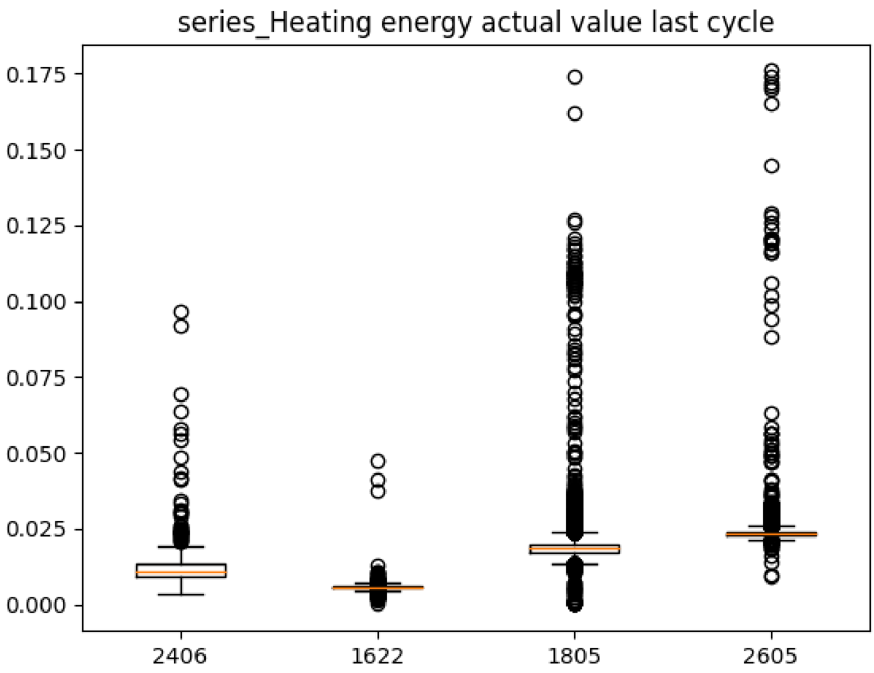


Figure 39: *Boxplot for Heating energy actual value last cycle feature of EN machines*

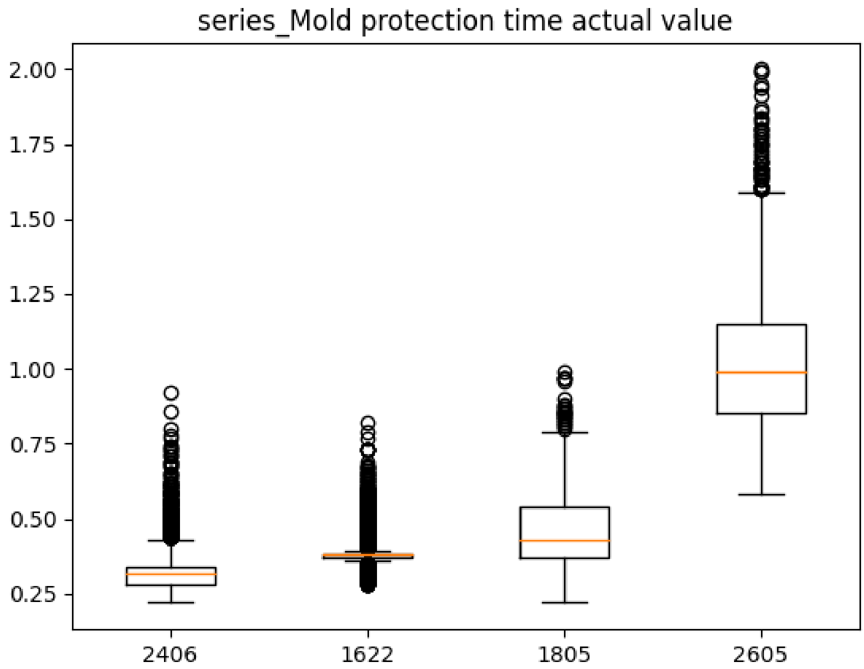


Figure 40: *Boxplot for Mold protection time actual value feature of EN machines*

B. Univariate Predictions

B.1. Machine by20

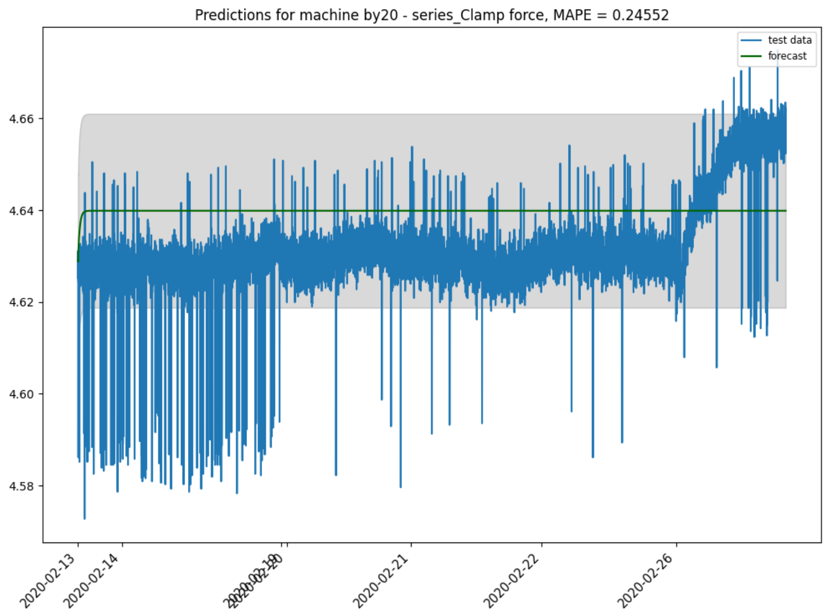


Figure 41: *Predictions made by the model for feature Clamp force for machine by20*

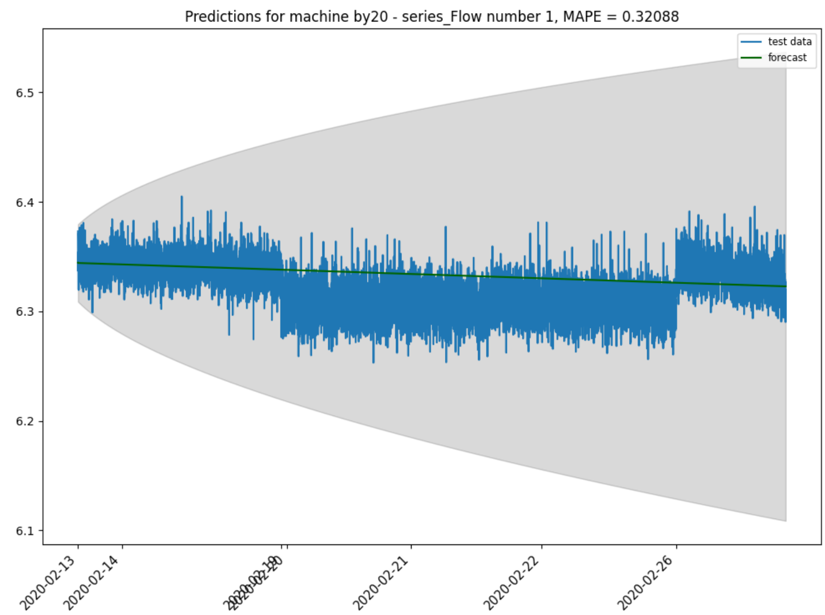


Figure 42: *Predictions made by the model for feature Flow number 1 for machine by20*

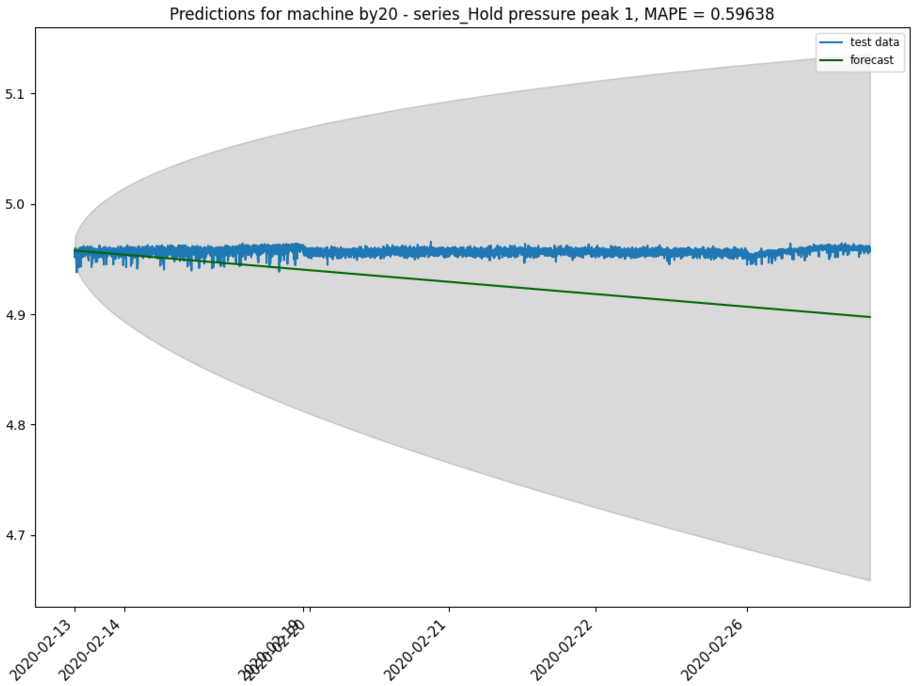


Figure 43: *Predictions made by the model for feature Hold pressure peak 1 for machine by20*

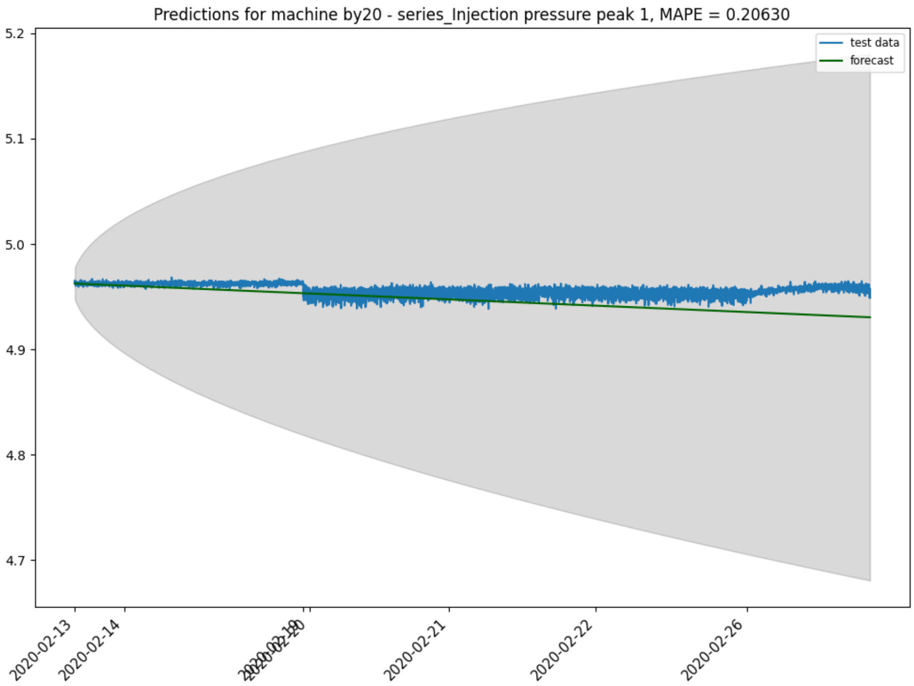


Figure 44: *Predictions made by the model for feature Injection pressure peak 1 for machine by20*

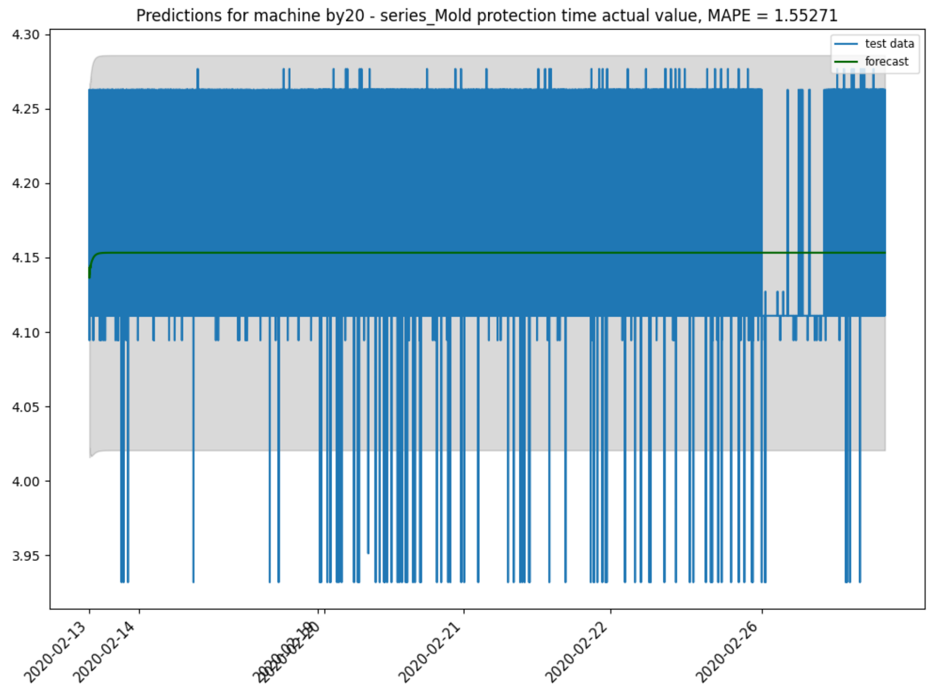


Figure 45: *Predictions made by the model for feature Mold protection time actual value for machine by20*

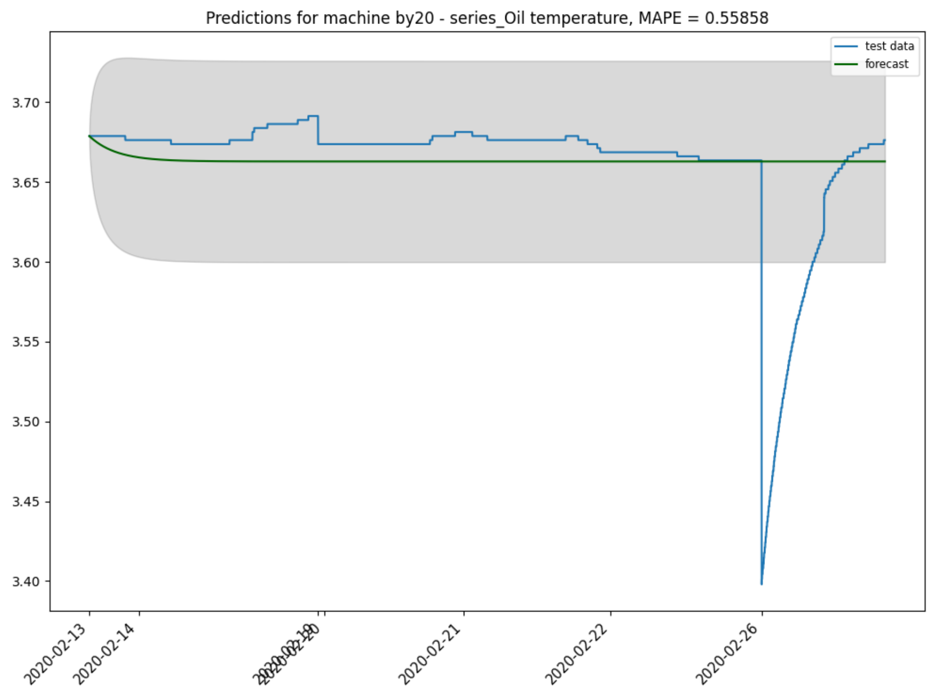


Figure 46: *Predictions made by the model for feature Oil temperature for machine by20*

B.2. Machine by19

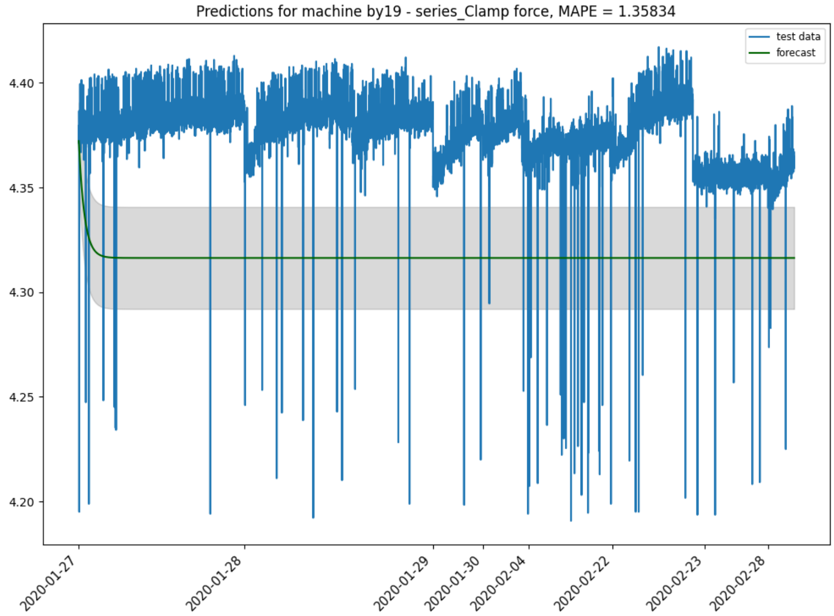


Figure 47: *Predictions made by the model for feature Clamp force for machine by19*

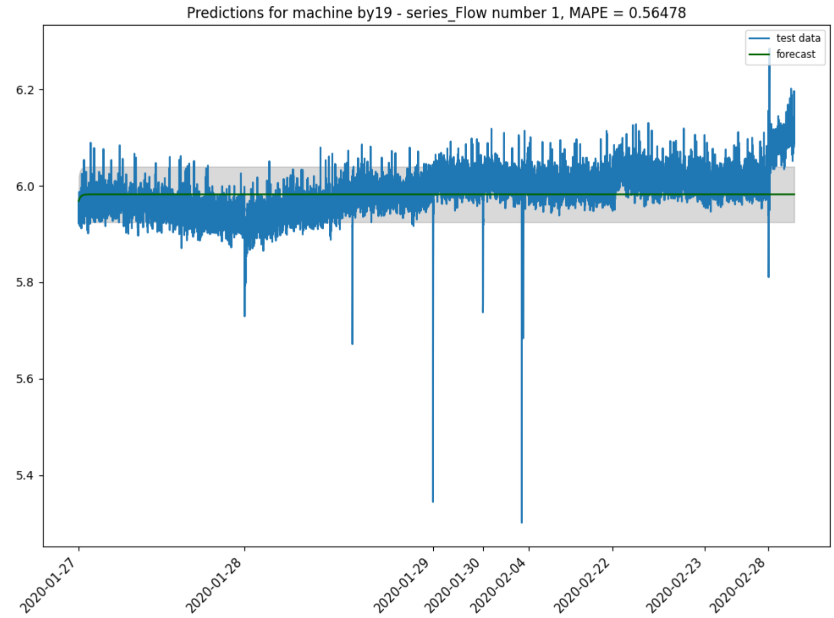


Figure 48: *Predictions made by the model for feature Flow number 1 for machine by19*

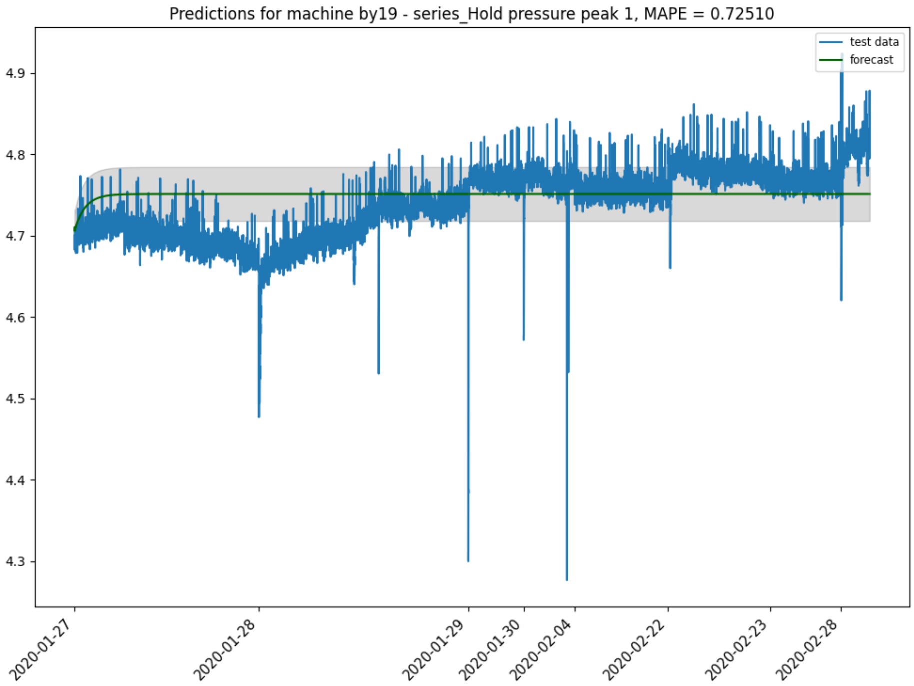


Figure 49: *Predictions made by the model for feature Hold pressure peak 1 for machine by19*

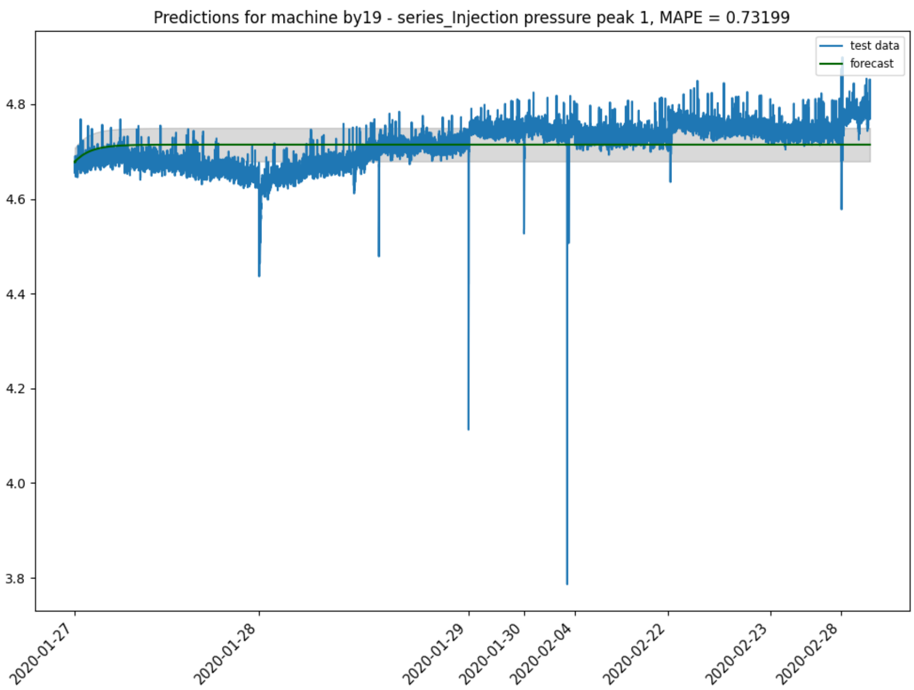


Figure 50: *Predictions made by the model for feature Injection pressure peak 1 for machine by19*

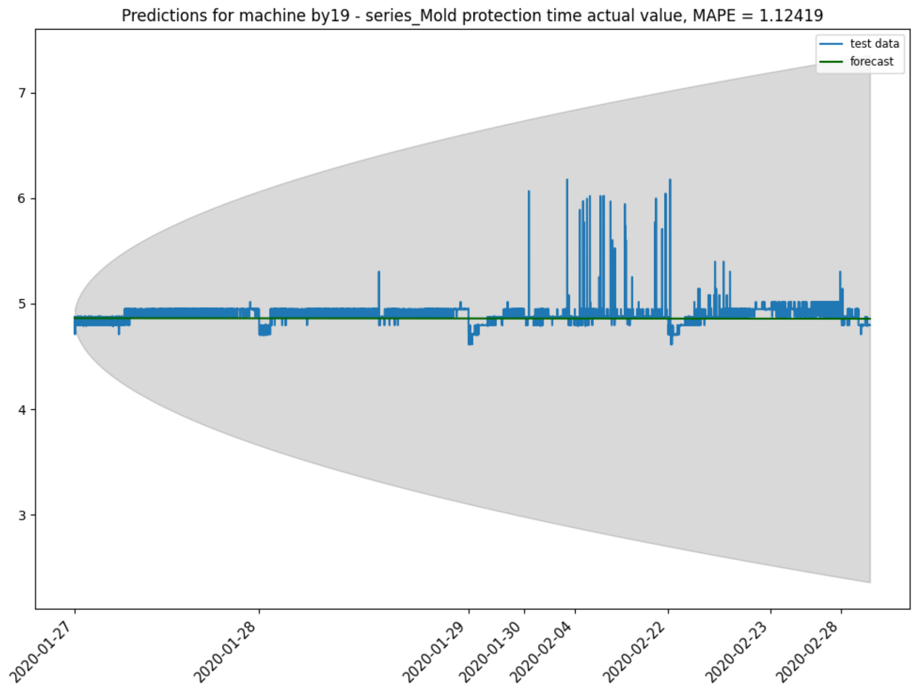


Figure 51: *Predictions made by the model for feature Mold protection time actual value for machine by19*

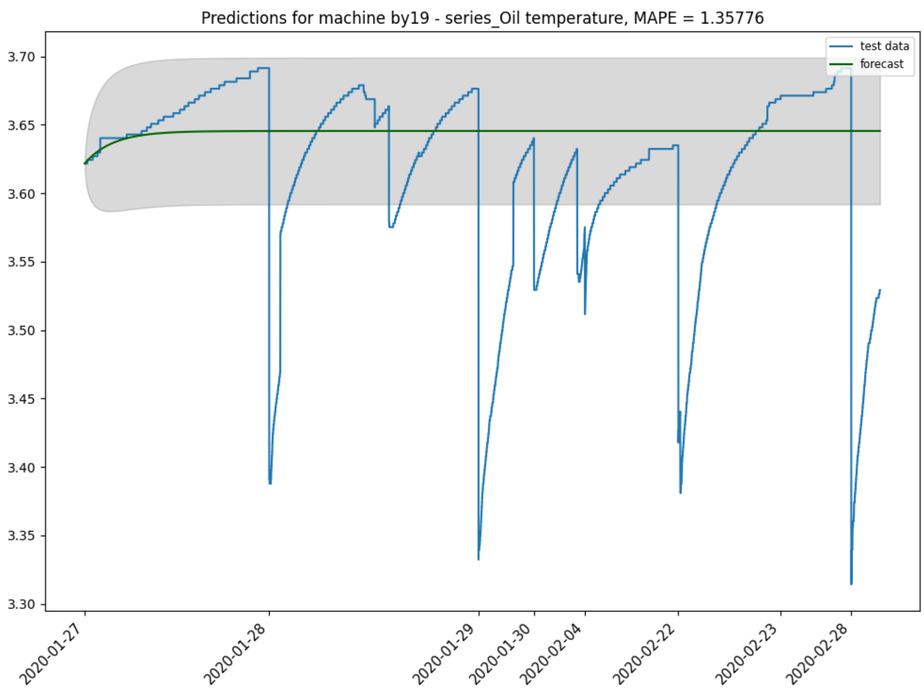


Figure 52: *Predictions made by the model for feature Oil temperature for machine by19*

B.3. Machine 2406

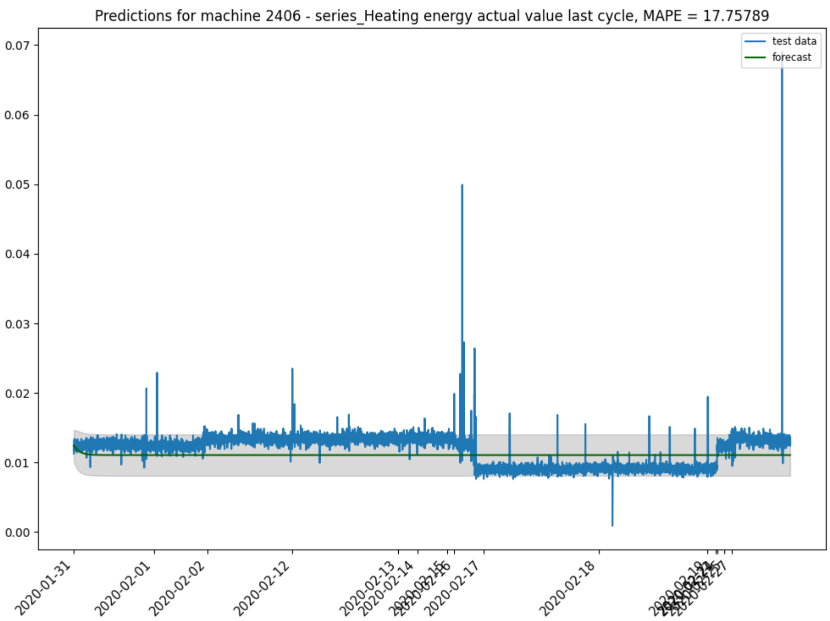


Figure 53: *Predictions made by the model for feature Heating energy actual value last cycle for machine 2406*

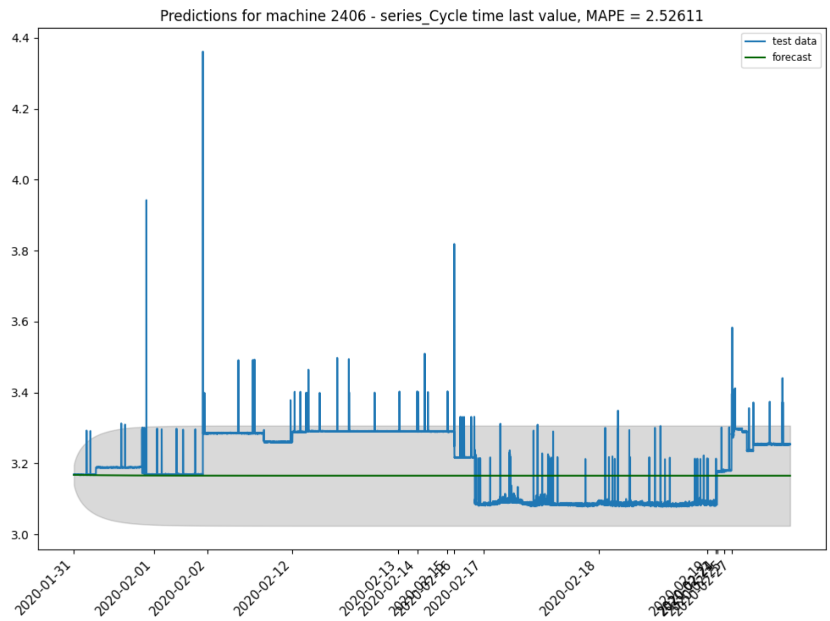


Figure 54: *Predictions made by the model for feature Cycle time last value for machine 2406*

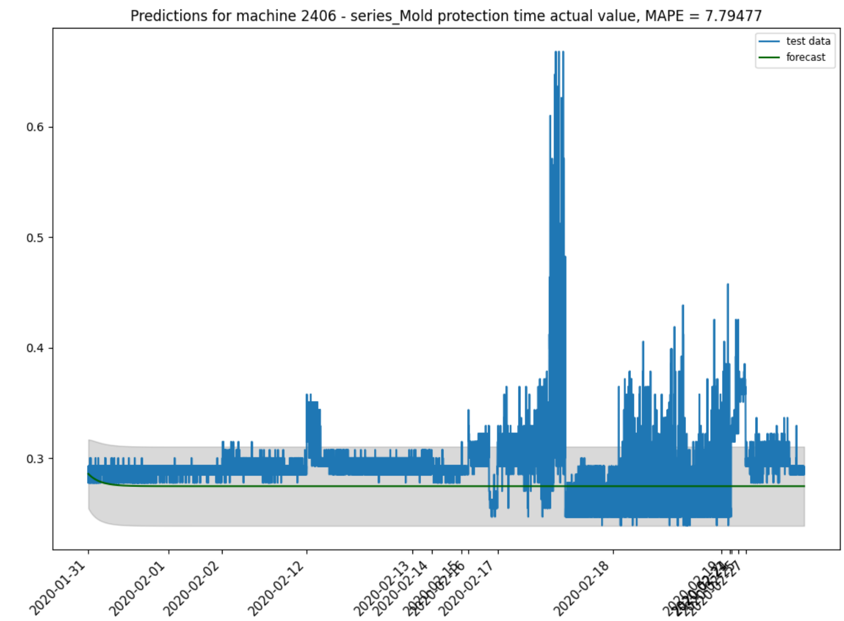


Figure 55: *Predictions made by the model for feature Mold protection time actual value for machine 2406*

B.4. Machine 1622

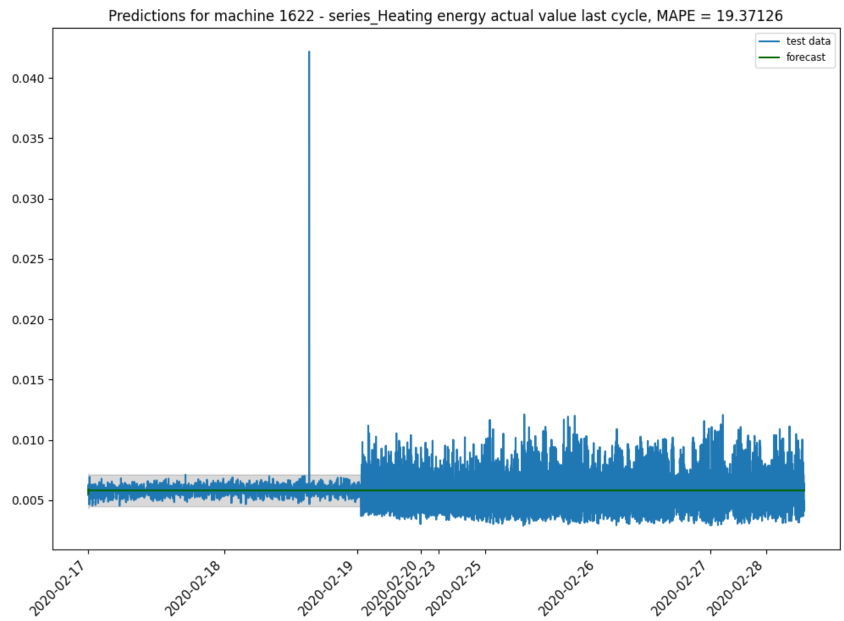


Figure 56: *Predictions made by the model for feature Heating energy actual value last cycle for machine 1622*

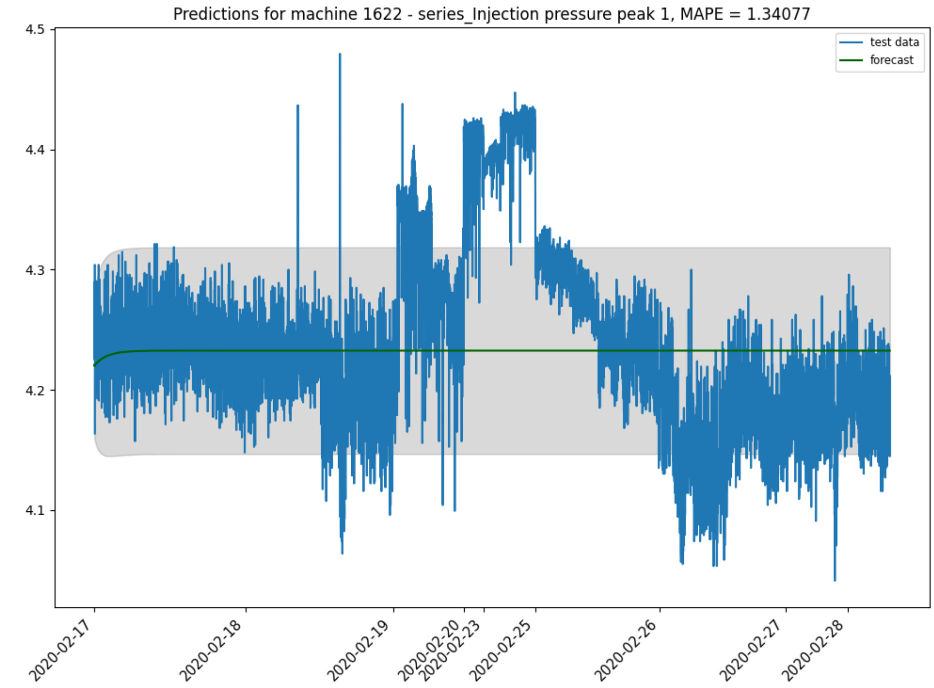


Figure 57: *Predictions made by the model for feature Injection pressure peak 1 for machine 1622*

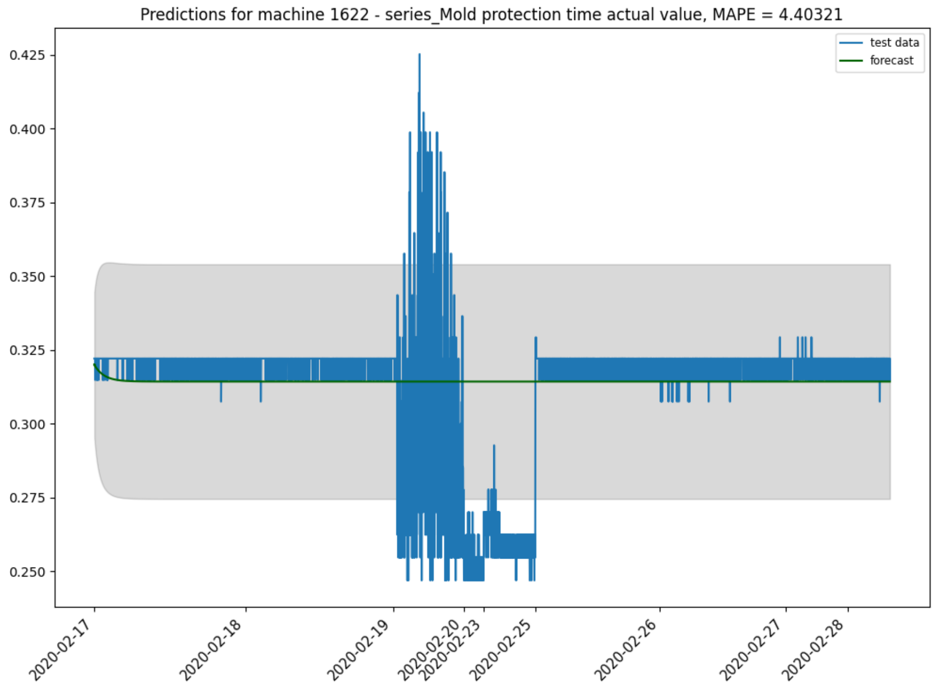


Figure 58: *Predictions made by the model for feature Mold protection time actual value for machine 1622*

B.5. Machine 1805

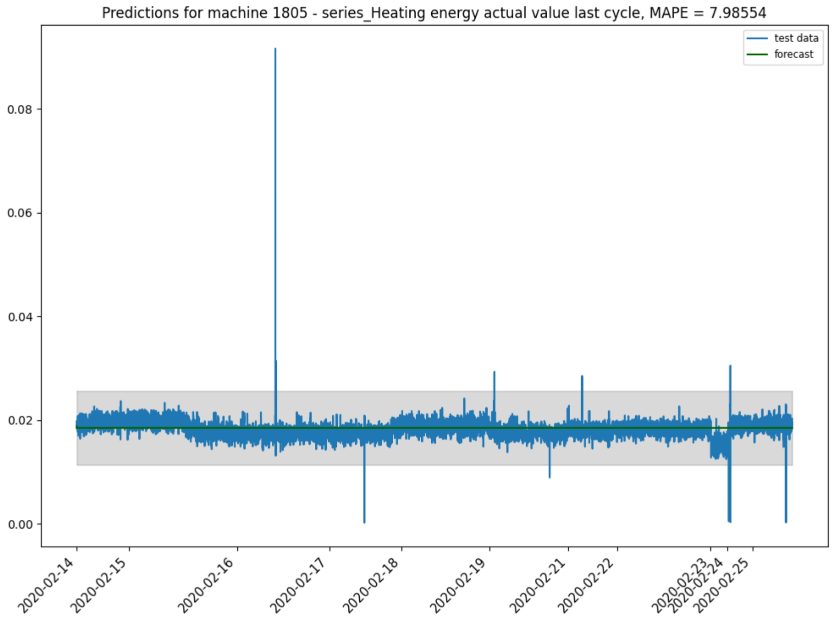


Figure 59: *Predictions made by the model for feature Heating energy actual value last cycle for machine 1805*

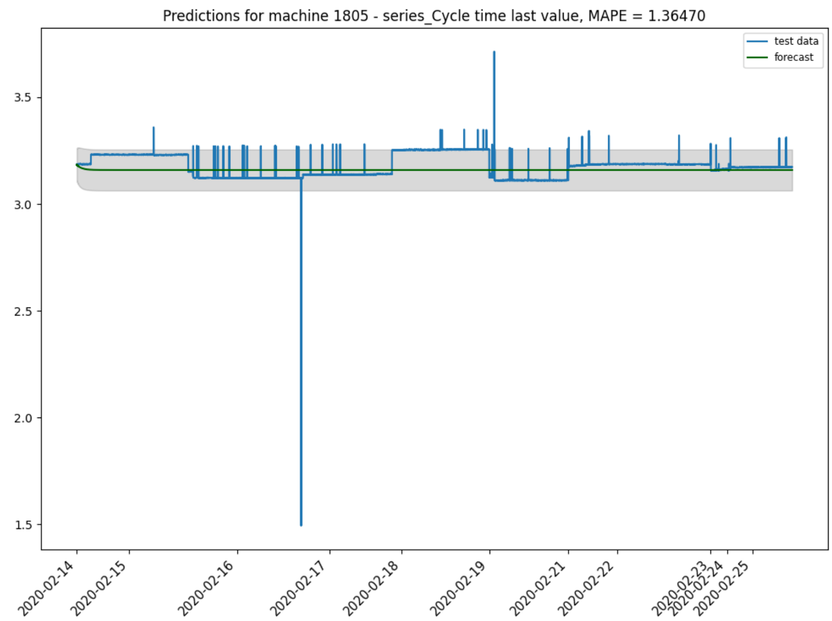


Figure 60: *Predictions made by the model for feature Cycle time last value for machine 1805*

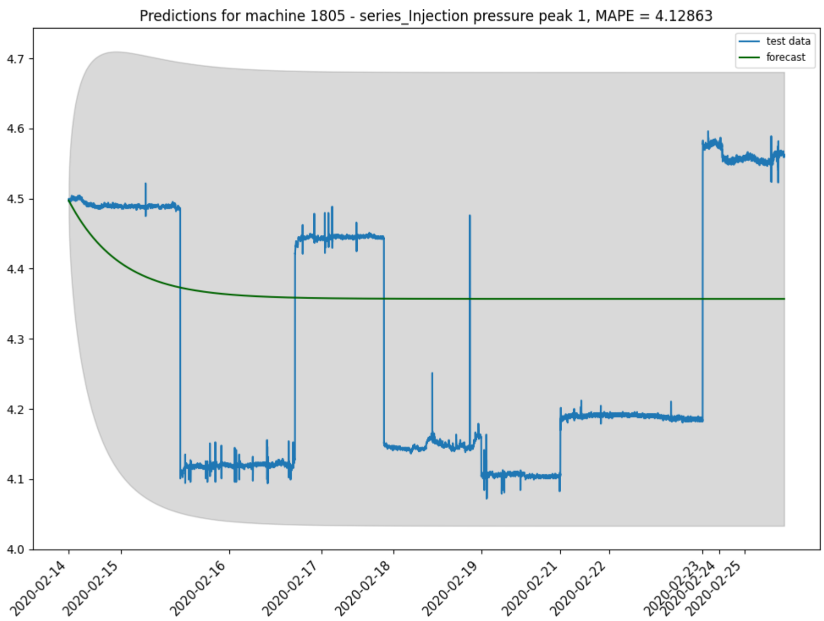


Figure 61: *Predictions made by the model for feature Injection pressure peak 1 for machine 1805*

B.6. Machine 2605

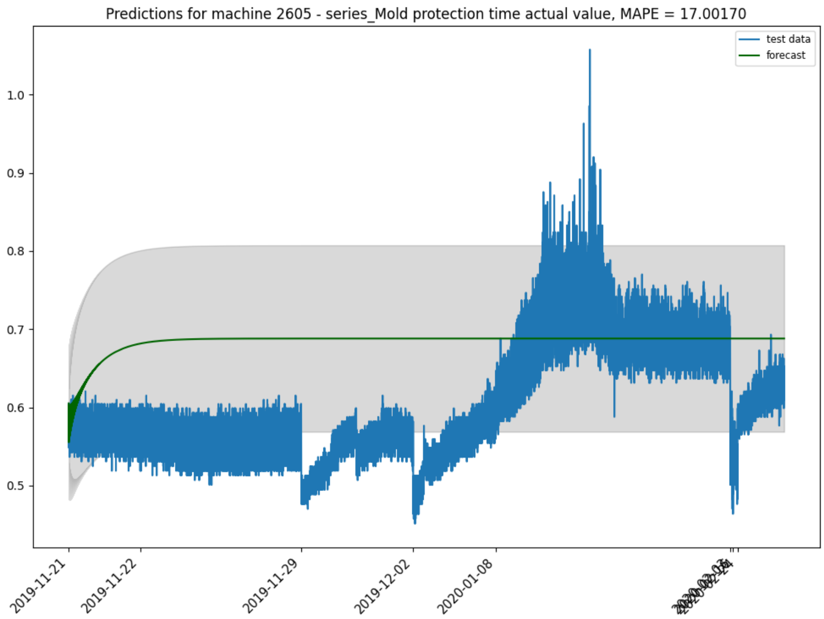


Figure 62: *Predictions made by the model for feature Mold protection time actual value for machine 2605*

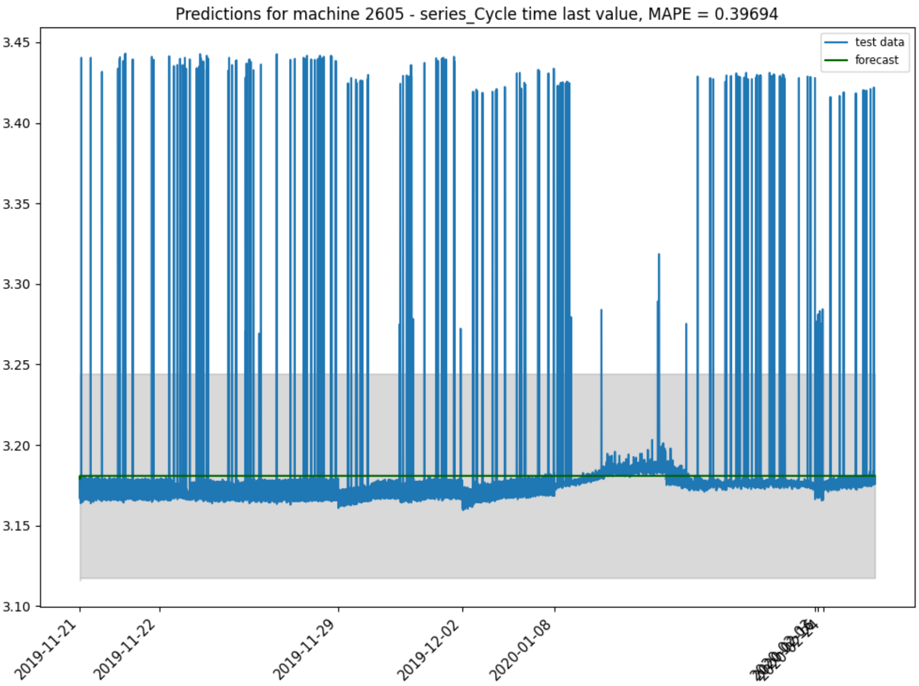


Figure 63: *Predictions made by the model for feature Cycle time last value for machine 2605*

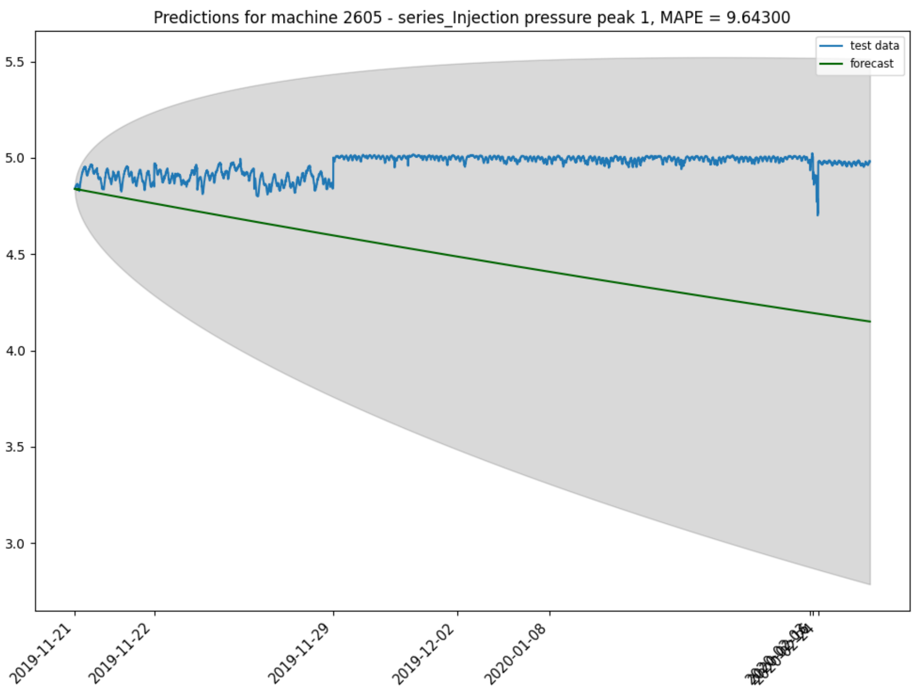


Figure 64: *Predictions made by the model for feature Injection pressure peak 1 for machine 2605*

C. Multivariate Results

C.1. Predictions

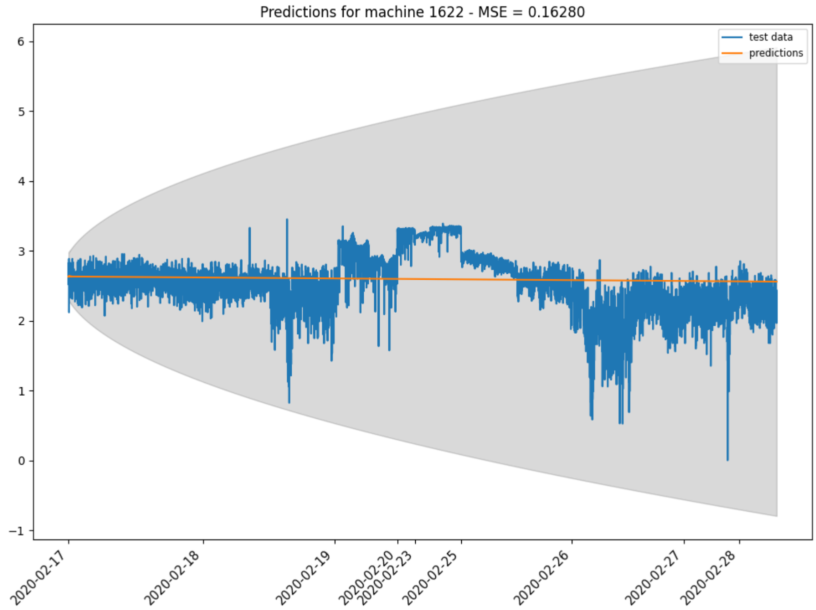


Figure 65: *Prediction for machine 1622*

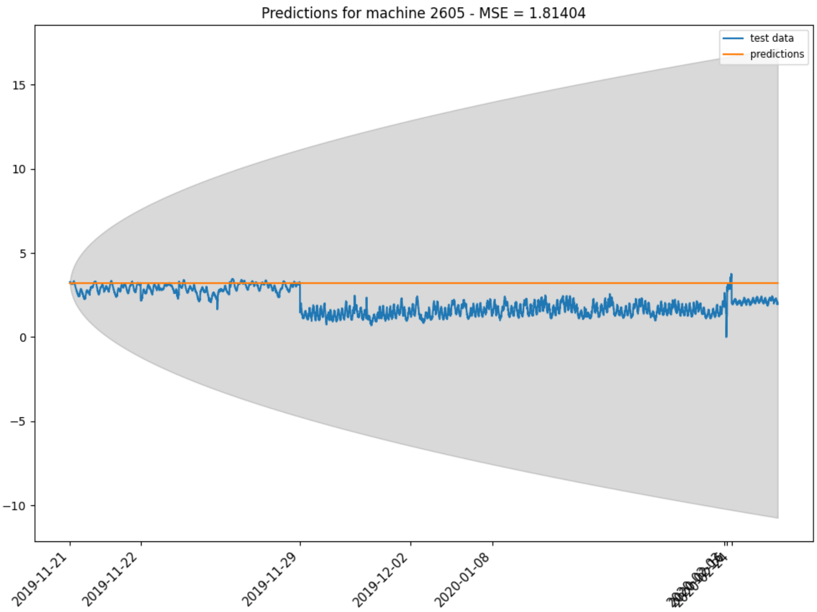


Figure 66: *Prediction for machine 2605*

C.2. Anomaly detection

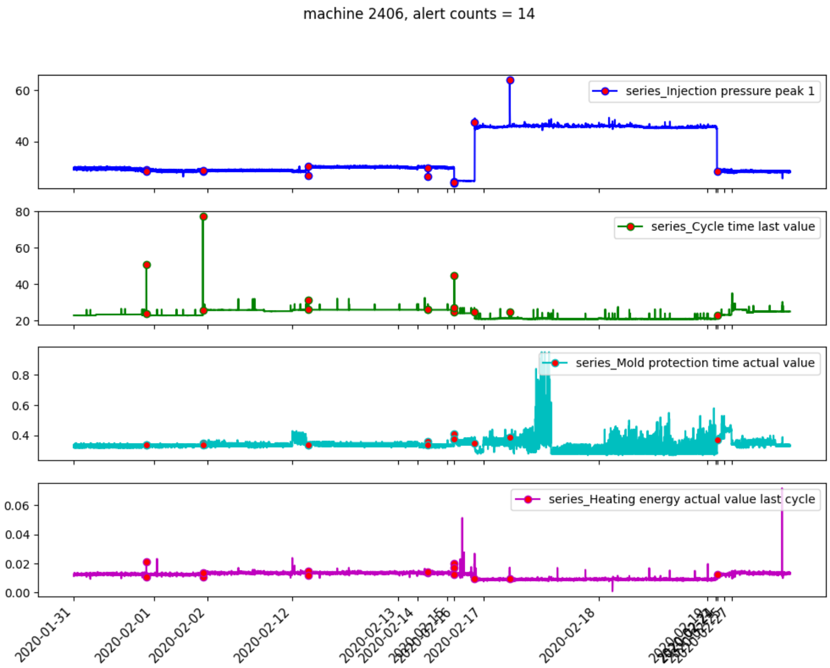


Figure 67: *Detected anomalies for machine 2406*

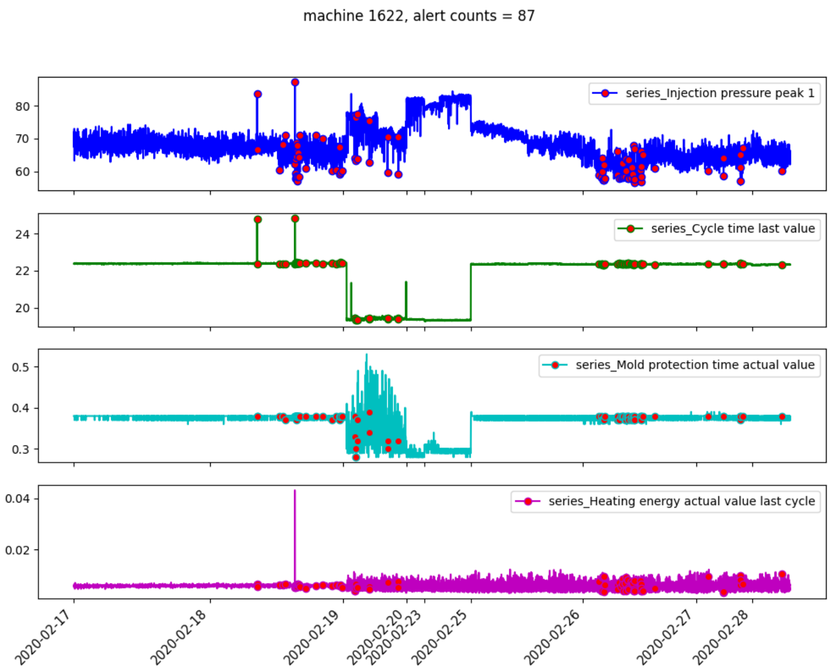


Figure 68: *Detected anomalies for machine 1622*

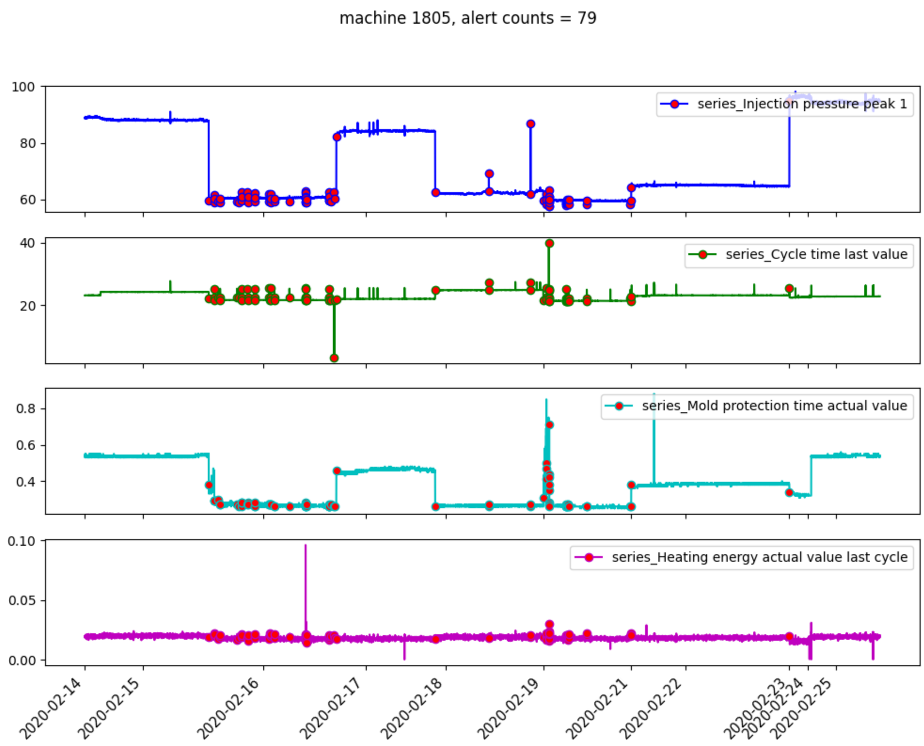


Figure 69: *Detected anomalies for machine 1805*

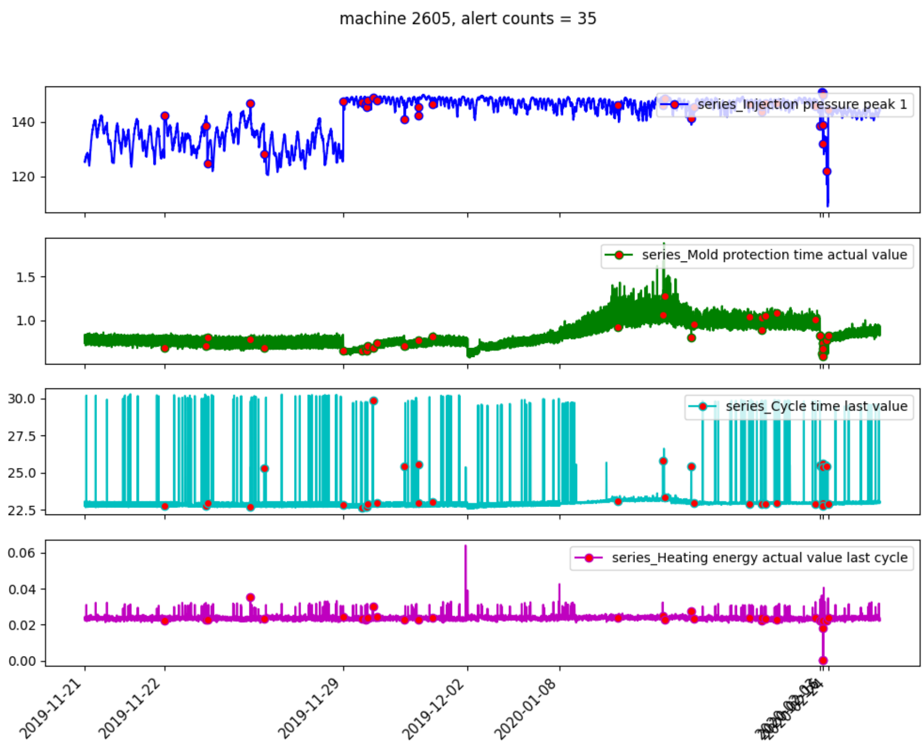


Figure 70: *Detected anomalies for machine 2605*

C.3. Forecast on held out test data

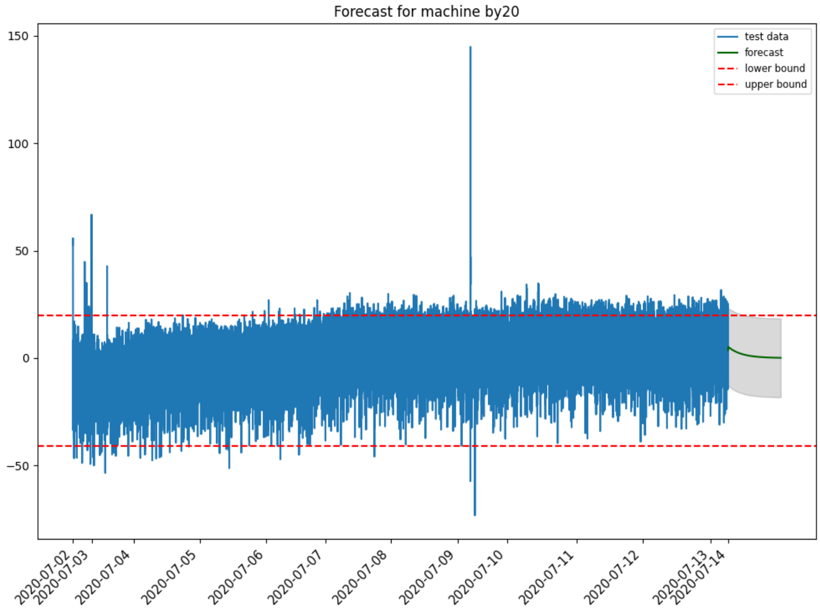


Figure 71: *Forecast for machine by20*

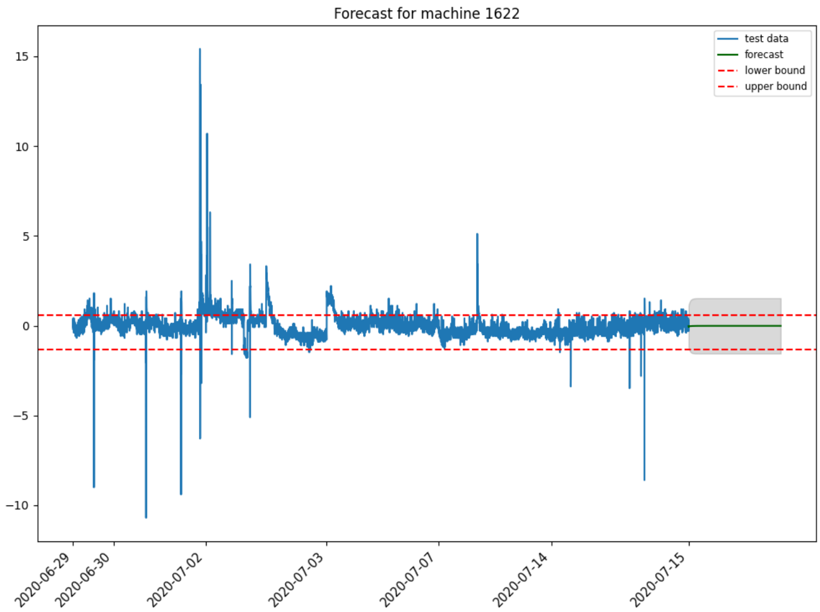


Figure 72: *Forecast for machine 1622*

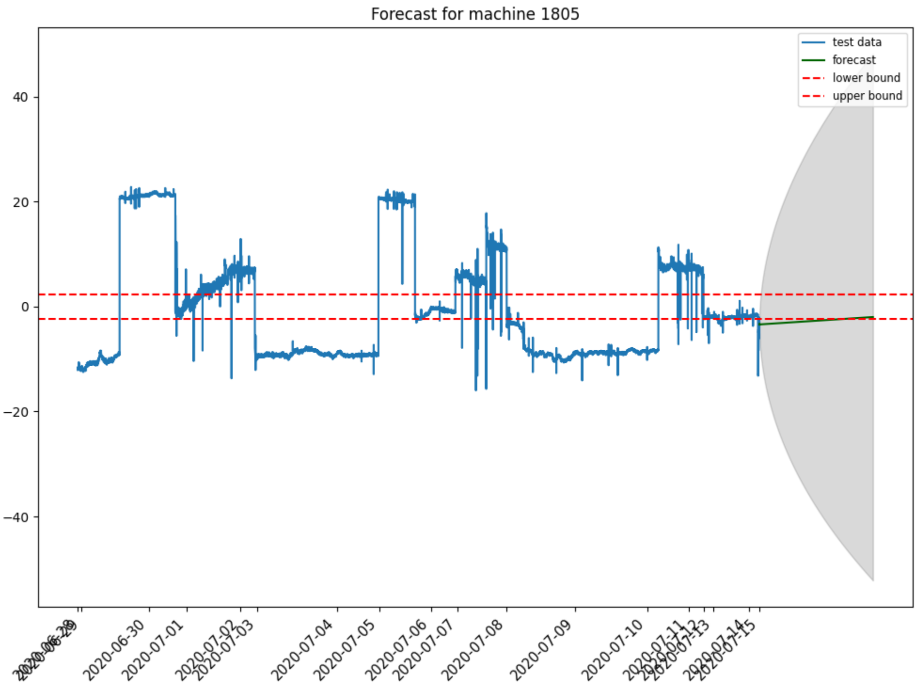


Figure 73: *Forecast for machine 1805*