Arindam Saha

Predictive Fault Detection in Elevators from Change Points analysis

Metropolia University of Applied Sciences

Master Of Engineering

Information Technology

Master Thesis

05.09.2022

Abstract

Author(s):	Arindam Saha
Title:	Metropolia Univeristy of Applied Sciences
Number of Pages:	60 pages + 2 appendices
Date:	10 September 2022
Degree:	Master of Engineering
Degree Programme:	Information Technology
Instructor(s):	Mari Zakrzewski (Data Analyst Expert) Sami Sainio (Principal Lecturer)

Time series analysis can be a crucial and valuable form of data source to track changes over time. It has a variety of applications such as gaining business insights, weather forecasts, stock prices predictions, and application performance. It can also be used as a tool for anticipating potential device malfunctions. This incentive is the motivation for this thesis with the goal of identifying probable issues in the elevator machines in advance. The technicians can be alerted and remedy the problem before it materializes, leading to overall improved performance.

This thesis aims to predict possible breakpoints or maintenance needs for elevators based on change points data generated from internet of things (IoT) devices connected to the elevators.

This work analyzes the statistics in the time series change points data, such as high peaks, and correlate them with previous historical maintenance needs of the elevators, which are recorded as service orders. The data is visualized to determine if there may be a possible breakpoint in the future.

Change point data collected over 1 year for 10 elevators was used as a test set. It was found that the peak values of change points correlate with maintenance service needs reported within a certain time frame after the peak. There were various sets of parameters which were tested against the model. The results have been measured with metrices such as accuracy, precision, f1Score, recall, specificity. The metric values differed with different sets of parameters.

Based on this work it can be concluded that change points data is a valuable source of information when used in predicting breakdown of an elevator. The data can be further refined to have more accurate results and other algorithms can also be investigated for the same. The current study serves as a proof of concept to determine whether such data can be used for further research on a larger set of elevators.

Keywords: Fault Detection, Data visualization, Change points, Time Series Analysis

Contents

1 Introduction

Internet of Things (IoT) has enabled connectivity of devices and enabled collection of data to measure key performance indicators (KPIs) of the IoT systems. This has been done by equipping the devices with sensors, communication interfaces that allow them to gather and transmit data over a network. This has led to the creation of an interconnected network of devices that can collect, process, and share data, enabling new applications and services in areas such as smart homes, industrial automation, health care, and transportation. Additionally, advances in cloud computing, big data analytics, and machine learning have enhanced the aggregation, analysis, and utilization of the vast amounts of data generated by IoT devices. The data collected over time from these devices can have high volumes. Time series data is formed by collecting information of these events over time and storing them for later use.

Time series data can be crucial in identifying various attributes of the system.

Out of various kinds of data which are generated by the elevator, one such kind is the Fstats statistics data which is calculated from the constant movement phase of the elevator. Using this Fstats data, change points time series data is generated. Although change points data has uses in various fields, this research concentrates on using the change points data to find possible anomalies of the elevator in a time window. The goal is to predict the possible breakdown of the system prior to the event.

The literature part of this work concentrates on explaining the various theoretical literature that are required for the understanding of the thesis. It describes the procedures of generating the change points data, which is the main information needed for this thesis. It also defines the fields in other required data such as service order and equipment.

The implementation has been done in python programming language. It is an interpreted object-oriented programming language and very popular in fields of data science, data mining and analytics. It has been highly rated in a comparative study of tools used in data analysis by Anmol Bansal[1]. Python has a large and active community of developers and users, which has resulted in a wealth of high-

quality libraries and tools for data analysis and machine learning, such as NumPy, Pandas, Matplotlib, scikit-learn, TensorFlow, and PyTorch. These libraries make it easier to perform complex data analysis and machine learning tasks in a concise and efficient manner. Additionally, Python has a simple, readable syntax that makes it well-suited for prototyping and experimenting with different models. This has contributed to its popularity and widespread adoption in the data science and machine learning communities.

The metrics of the results for the model are measured with metrics accuracy, F1-score, and specificity. The data is also plotted for visualization. The results are presented as result sets with different parameters.

2 Theoretical Background

2.1 Performance metrics of a classification model

In a data analytical model, it is usually focussed on how accurate the predictive value can be. When dealing with categorical data, several models can be built to predict values based on the given data. Agresti[2] defines binary categorical data as the kind of data that can be classified into one of two categories or classes. For example, gender (male or female), success/failure, or yes/no responses are examples of binary categorical data. These types of data can be analyzed using statistical methods such as logistic regression or chi-square tests. A mechanism is needed that helps in estimating correct classification and misclassification. The confusion matrix serves this purpose. Kai Ming Ting[3] summarizes a confusion matrix as the classification performance of a classifier with respect to some test data, and the matrix being two dimensional, indexed in one dimension by the true class of an object and the other by the class that the classifier defines.

The table shows an example of a three-class classification confusion matrix, the classes being A, B and C.

Table 1: Three class confusion matrix

		Assigned Class		
		A	B	C
Actual Class	A	10	2	1
	B	0	6	1
	C	0	3	8

For binary classification data, confusion matrix is utilized with two classes, positive and negative. In this scenario, the matrix will be 2 * 2, with four cells. These cells are designated as true positives (TP), false positives (FP), true negatives (TN), false negatives. (FN). This is shown below in table 2.

Table 2: Confusion matrix for binary classification data

		Assigned Class	
		Positive	Negative
Actual Class	Positive	TP	FN
	Negative	FP	TN

A True Positive (TP) is defined as the positive examples that are correctly classified by a classification model[4].
A False Positive (FP) is an example of negative class that has been incorrectly classified as positive[5].
A True Negative (TN) is a negative example that are correctly classified by a classification model[6].
A False Negative (FN) is an example of positive class that has been incorrectly classified as negative[7].

There are several performance metrics that can be calculated from TP, FP, TN, and FN. Some of them are listed below.

Precision or Positive Predictive Value refers to the proportion of positive categorizations that was correct. It represents how often the model correctly identifies a positive class.

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

For example, in a binary classification problem where the positive class is "elevator breakdown" and the negative class is "no elevator breakdown", precision would measure the proportion of elevator breakdown instances that were predicted correctly by the model, even if it also produced some false positives.
Precision ranges from 0 to 1, where a score of 1 represents perfect precision (i.e. the model made only correct positive predictions) and a score of 0 represents the worst possible precision (i.e. the model made only incorrect positive predictions).
A high precision value indicates that the model is making fewer false positive predictions, however, it may also produce a higher number of false negatives,

resulting in lower recall. So, it is important to consider the trade-off between precision and other metrics such as recall and F1-score to evaluate the performance of a model.

Recall, also known as Sensitivity, refers to the proportion of real positive outcomes that are correctly predicted as positive by the classification model.

$$sensitivity = \frac{True\ Positives}{True\ Positives + \ False\ Negatives}$$

For example, in a binary classification problem where the positive class is "elevator breakdown" and the negative class is "no elevator breakdown", recall would measure how well the model is able to identify all the instances of elevator breakdown, even if it also produces some false positive predictions.
Recall ranges from 0 to 1, where a score of 1 represents perfect recall (i.e. the model identified all the positive examples) and a score of 0 represents the worst possible recall (i.e. the model did not identify any positive examples).
A high recall value indicates that the model is able to correctly identify most of the positive examples in the dataset, however, it may also produce a higher number of false positives. So, it is important to consider the trade-off between recall and other metrics such as precision and F1-score to evaluate the performance of a model.
The **Specificity** (True Negative Rate) refers to the proportion of actual negative cases which are predicted as negative by the classification model. It is a measure of how well a classification model is able to identify all the negative examples in a dataset.

$$specificity\ = \frac{True\ Negatives}{True\ Negatives + \ False\ Positives}$$

For example, in a binary classification problem where the positive class is "elevator breakdown" and the negative class is "no elevator breakdown", specificity would measure how well the model is able to identify all the instances of "no elevator breakdown" and not the breakdown, even if it also produces some false negatives.

Specificity ranges from 0 to 1, where a score of 1 represents perfect specificity (i.e. the model identified all the negative examples) and a score of 0 represents the worst possible specificity (i.e. the model did not identify any negative examples).

A high specificity value indicates that the model is able to correctly identify most of the negative examples in the dataset, however, it may also produce a higher number of false negatives. So, it is important to consider the trade-off between specificity and other metrics such as sensitivity and F1-score to evaluate the performance of a model.

Accuracy is the proportion of correct predictions to the total number of predictions. It measures how often the model makes correct predictions. The formula for accuracy is:

$$Accuracy = \frac{True\ Positives + True\ Negatives}{Total\ Samples}$$

It's worth noting that, these metrics are useful when you have a balanced dataset, where the proportion of positive and negative samples is roughly the same. If the dataset is imbalanced, these metrics can be misleading and other metrics such as AUC-ROC, F1-Score, G-mean can be used.

F1-score is a metric that combines precision and recall providing a single, comprehensive measure of a model's performance. It is commonly used in cases where the data is imbalanced and precision and recall alone may not provide an accurate representation of the model's performance. The F1-score is the harmonic mean of precision and recall. The formula for F1-score is:

$$F1score = \frac{2 * (Precision * Recall)}{Precision + Recall}$$

The F1-score ranges from 0 to 1, where a score of 1 represents perfect precision and recall, and a score of 0 represents the worst possible performance. The F1-

score is particularly useful in situations where the positive class is rare or when you want to find a balance between precision and recall.

To fully examine the effectiveness of a classification model, these parameters are used. We would use this metrics for calculating how the classification model works.

2.2 Change points

As the name suggest, the change point locates the position where there was an abrupt change in the characteristics of a system, in a times series segment. As described by Shohreh D.[8] et al, the change points identify the time associated changes in trends and properties of a system, which can be used to study the underlying behavior of the same. He further describes that it is an analytical method to identify the times associated with abrupt transitions of a series, which can be used to extract the meaning from non-annotated data. Change points from different kinds of systems can provide critical understanding of the underlying behaviour of the system.

Change points can detect abrupt changes which represents transitions which occur between states, and hence it is found useful in different application areas such as medical condition monitoring, climate change detection, speech and image analysis, and human activity analysis[9].

Change points in time series data can be useful in a variety of ways such as:

- Anomaly detection: Change points can be used to identify abnormal behavior or outliers in the data, which can be useful for monitoring systems and identifying potential issues.

- Trend analysis: Change points can be used to identify when a trend in the data is beginning or ending, which can be useful for understanding the underlying dynamics of the system being studied.

- Seasonality analysis: Change points can be used to identify when seasonal patterns are beginning or ending, which can be useful for understanding the underlying dynamics of the system being studied.

- Cycle analysis: Change points can be used to identify when cycles in the data are beginning or ending, which can be useful for understanding the underlying dynamics of the system being studied.

- Forecasting: Change points can be used to identify when the underlying dynamics of a system are changing, which can be useful for improving the accuracy of forecasting models.

- Control: Change points can be used to identify when a process or system is deviating from normal behavior, which can be useful for controlling the process or system.

- Decision Making: Change points can be used to identify when a process or system is deviating from normal behavior, which can be useful for making decisions about how to respond to changes in the system.

There are several types on which change point are analyzed. Some common types are as follows[10]:

- *Change in Mean*: This can be detected on a time series signal divided into different constant segments with varying mean values. This has applications in process and quality control. The image below shows an example of change in mean over time.

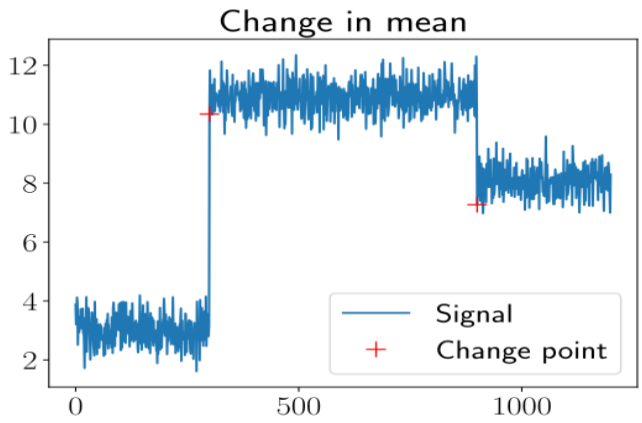

Figure 1: Change in mean of a signal

- *Change in Variance* - In a change of variance, the time series is divided into constant segments with varying variance, keeping the mean constant. As shown below, it can be interpreted as a noise in a signal.

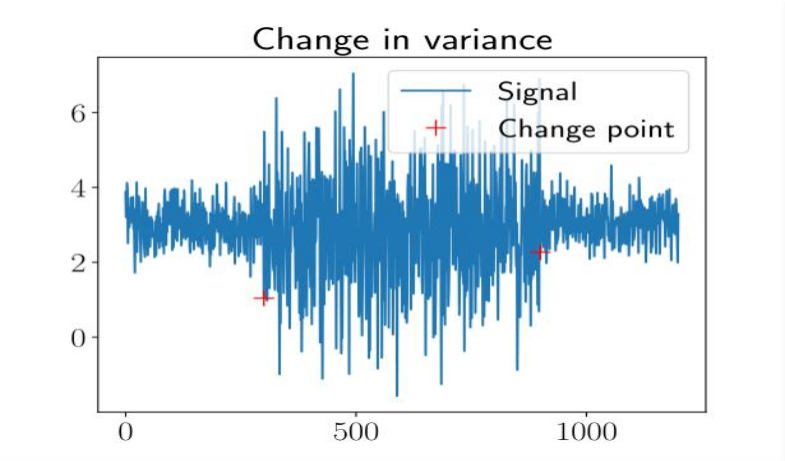

Figure 2: Change in variance of a signal

- *Change in periodicity* - This type of change occurs when there is a sudden change of the frequency in the signal. This kind of change can be done in frequency domain, for example Fourier Transform or wavelet transform.

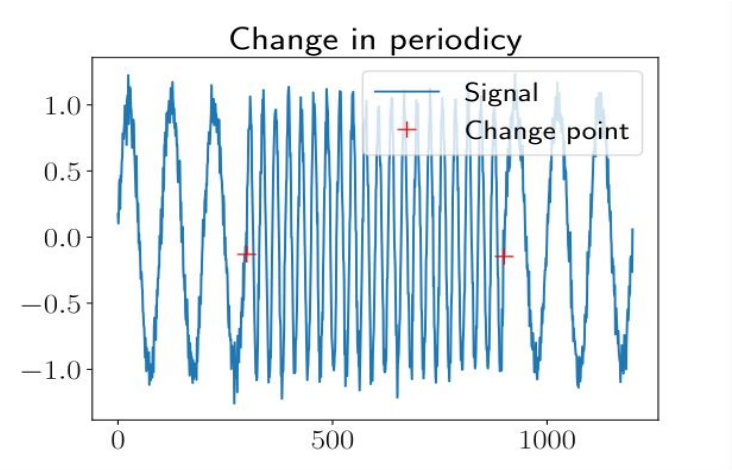

Figure 3: Change in periodicity of a signal

- *Change in correlation* - Change in correlation refers to a shift in the relationship between two or more variables. For example, in a time series, there may be a change in the correlation between different variables at a certain point, indicating a change in the underlying dynamics of the system. This type of change point can be analyzed using statistical methods such as the Cumulative Sum Control Chart (CUSUM) test or the change point analysis method. These methods can be used to detect a change in correlation between multiple time series, or between a time series and a categorical variable. In some cases, a change in correlation may indicate a causal relationship between the variables, and further investigation may be required to understand the underlying mechanisms.

The other various types of changes can be change in trends, skewness, structure etc. There are various algorithms that have been developed for detecting and producing change points data. A comparison and overview of some of the popular algorithms have been done by Aminikhanghahi and Cook[9].

Some of the common algorithms for detecting change points data are:

- CUSUM (Cumulative Sum) - This is a simple algorithm that compares the current data point with a reference value or threshold, and updates the

cumulative sum. When the cumulative sum exceeds a certain threshold, a change point is detected.

- Bayesian change point detection - This algorithm uses Bayesian statistics to model the data and detect changes in the distribution.
- Hidden Markov Models (HMMs) - These are probabilistic models that are used to detect changes in the underlying state of a system.
- Online Change Point Detection - These are the algorithms that can detect changes in the data as they arrive in real-time, rather than waiting for all the data to be collected.
- Change Point Detection with Neural Networks - These algorithms use neural networks to detect change points in data.
- Nonparametric Change Point Detection - These are methods that do not require the assumption of a specific distribution for the data and they are widely used in practice.
- Binary Segmentation - This algorithm segments the time series into two regions, one before and one after the change point using a recursive binary search strategy.
- Pruned Exact Linear Time (PELT) - This algorithm is a linear time complexity method for detecting change points in large datasets.
- Wild Binary Segmentation (WBS) - This is a non-parametric method for change point detection that is robust to noise and outliers in the data.

The F-stats Chow test, also known as the Chow test for structural change, is a statistical test used to detect structural changes or breaks in linear regression models. The test compares the fits of two nested models, one with a breakpoint and one without, to determine if the addition of a breakpoint improves the model fit. The test statistic is the ratio of the explained variation between the two models, and is distributed as an F-distribution. The test compares the F-statistic value to a critical value from the F-distribution table to determine if there is evidence of a breakpoint. The test assumes that the errors of the models are normally distributed and independent. It is commonly used to identify change points in time series data, such as changes in the mean, variance or correlation of the data.

As pointed out by Truong et al[11] (2018) the choice of algorithm depends on the application domain and characteristics of the data. It reviews the algorithms which are characterized by three elements: a cost function, a search method, and a constraint on a number of changes. It also concluded that the approaches are included in a python scientific library called rupture[12].

Change point detection is a common problem in industrial applications, such as monitoring the performance of manufacturing processes, detecting changes in the behavior of machines, and identifying faults in the system.
There are several methods that have been proposed for change point detection in time series data from machines. These methods can be broadly categorized into two groups: parametric and non-parametric methods.
Parametric methods assume that the underlying data distribution is known and that the change point corresponds to a change in the parameters of the distribution. Examples of parametric methods include likelihood ratio test and Bayesian change point detection.
Non-parametric methods, on the other hand, do not make any assumptions about the underlying data distribution and are based on the change in the distribution of the data. Examples of non-parametric methods include the Cumulative Sum (CUSUM) algorithm and the Exponentially Weighted Moving Average (EWMA) method.
Recently, machine learning methods have been used in change point detection in time series data. For example, deep learning techniques like Recurrent Neural Network (RNN), Long Short-Term Memory (LSTM), and Autoencoder have been applied to detect anomalies in time series data[13].

There have also been research studies that focus on detecting change points in time series data in order to predict breakdowns or failures in machines. These studies typically use change point detection methods to identify changes in the behavior of machines that may indicate an impending breakdown or failure.

One such study by Chen et al[14] presents a method for detecting change points in time series data from industrial machines in order to predict failures. The

method uses a change point detection algorithm to identify changes in the behavior of the machine, and then uses a machine learning algorithm to predict the likelihood of a failure based on the identified change points.

Another study is Change Point Detection in Industrial Time Series Data for Predictive Maintenance by X. Liu[15] in 2019 which proposed a deep learning-based change point detection method that uses an autoencoder to detect changes in the behavior of machines, which can be used for predictive maintenance.

2.3 Pandas Dataframe

Pandas[16] is an open-source Python library that provides powerful data manipulation and analysis capabilities. One of the most important features of Pandas is the DataFrame data structure, which is essentially a two-dimensional table with labeled axes (rows and columns). The DataFrame is a highly flexible and efficient tool for working with time series data.

Time series data is a sequence of data points that are indexed by time. Some examples of time series data include stock prices, weather data, and sensor readings. Time series data is commonly analyzed to identify patterns, trends, and anomalies, and to make forecasts.

Pandas makes it easy to work with time series data by providing a variety of tools for manipulating and analyzing time series data.

Here are some of the ways that Pandas can be used to work with time series data:

- Time-based indexing: Pandas provides a set of tools for indexing data based on time, including the DatetimeIndex and PeriodIndex classes. These tools make it easy to slice and dice time series data based on time intervals.
- Time-based aggregation: Pandas provides a variety of tools for aggregating time series data, including the resample and rolling functions.

These tools make it easy to compute rolling statistics, moving averages, and other time-based aggregates.

- Data cleaning and transformation: Pandas provides a variety of tools for cleaning and transforming time series data, including tools for handling missing data, smoothing noisy data, and detrending data.
- Visualization: Pandas provides a variety of tools for visualizing time series data, including the plot method and the time series plot function. These tools make it easy to create informative visualizations of time series data.

2.4 Fast Fourier Transform

As Muller quotes in Fundamentals of music processing[17], a Fourier transform breaks up a signal into its corresponding frequency components. For each frequency ω, the Fourier transform produces a coefficient d_ω and a phase φ_ω which tells us to which extent the given signal matches the sine wave oscillations in that frequency.

A property of Fourier transform is that the original signal can be reconstructed from the coefficient d_ω and phase φ_ω. The original signal and the Fourier transform of the signal contains the same information, but the original signal is represented in time domain, while the Fourier transform of the signal is represented in frequency domain.

The Discrete Fourier Transform (DFT) is a mathematical technique for transforming a time-domain signal into a frequency-domain representation. The DFT is used to analyze the frequency content of signals, such as audio and image data, and is a key component of many signal processing algorithms.

The DFT takes a discrete-time signal, represented by a sequence of samples, and produces a corresponding sequence of complex numbers that represent the amplitude and phase of the signal's frequency components. The DFT can be computed using the fast Fourier transform (FFT) algorithm, which is an efficient

method for computing the DFT. The equation of DFT can be represented as below:

$$X_k = \sum_{n=0}^{N-1} \mathrm{x(n)} e^{-i2\pi kn/N}$$

The formula represents DFT for a sequence of N complex numbers $\{x_n : x_0, x_1 \ldots x_{N-1}$ into another set of complex numbers $\{X_k\} : X_0, X_1 \ldots X_{N-1}$. The time complexity of computing DFT is of the order $O(N^2)$.

Fast Fourier transform (FFT) produces the same results as of DFT but provides fast execution for the computation. The time complexity of FFT is of the order O(NlogN).

Fast Fourier Transform (FFT) is an algorithm that can be used to analyze time series data for change points, which are points in the data where there is a abrupt change in the underlying pattern.

FFT works by decomposing a time series into its constituent frequency components, represented as complex numbers. By analyzing the magnitudes and phases of these frequency components, FFT can be used to identify patterns in the data that repeat at specific intervals, such as trends, seasonality, and cycles.

When there is a change point, the spectral density of the time series will change, and this can be detected by analyzing FFT of the data. For example, if there is a sudden change in the trend of the data, this will be reflected as a change in the magnitudes of the frequency components at certain frequencies.

FFT can be useful in detecting change points in time series data as it can identify patterns that are not visible in the raw data, such as periodicity and seasonality.

2.5 Peak Detection

Peaks are sharp spikes in data, that are caused by abrupt changes in the observed (or recorded) signal. It has many uses such as sudden increase in price or demand or sudden bursts of data requests for an API can mean that something is unusual compared to usual set of things.

Peak detection has been used in different industries and use cases. For example, Zhang [18] et al (2017) proposed a peak detection method for glucose signals using a combination of continuous wavelet transform and support vector machine. The method was tested on a database of 100 glucose signals and showed improved accuracy and robustness compared to traditional peak detection methods. This study demonstrates the usefulness of peak detection in the medical industry, specifically for analyzing glucose signals in diabetes patients.

Another peak detection usage was presented by Wang[19] et al (2015) for capillary electrophoresis data using an improved particle swarm optimization algorithm. The proposed method was tested on real-world capillary electrophoresis data and showed improved accuracy and efficiency compared to traditional peak detection methods. This research highlights the importance of peak detection in the field of analytical chemistry and its applications in various industries such as pharmaceuticals, biotechnology, and food analysis.

When the peaks occur in a small set of data in each range of time, it can be easily spotted manually, and then respond accordingly. Manual peak detection can be done by human eyes when dealing with simple, well-defined peaks or when the data is small and easy to visualize. For example, manual peak detection can be done in cases where the peaks are distinct, well-separated, and the data is limited to a few data points. In these cases, the analyst can easily spot changes in the peaks by visually inspecting the data. However, this approach is not feasible when dealing with complex, large, or noisy data. There is a need for automatic peak detection in time series data because manual detection can be time-consuming, subjective, and prone to error, especially when dealing with large

amounts of data or data that is difficult to visualize. Automated peak detection algorithms can quickly and objectively identify peak patterns in the data, reducing the risk of human error and improving the efficiency of the analysis process. For example, if there are data points generated each minute from memory utilization of a server, there needs to be mechanism if there is a sudden increase of the same so that an alarm can be raised to take necessary actions, like increasing the memory size.

There are several robust and tested peak detection algorithms. Some of the simple peak detection algorithms has been discussed by Girish Palshikar[20].

There are several approaches to peak detection in time series data, including:

- Moving average - Smooth the time series data using a moving average filter and identify peaks as local maxima.
- First-order differentiation - Compute the first-order derivative of the time series data and identify peaks as zero-crossings of the derivative.
- Second-order differentiation - Compute the second-order derivative of the time series data and identify peaks as local maxima of the derivative.
- Wavelet transformation - Use wavelet transformation to decompose the time series data into different frequency components and identify peaks in the high-frequency components.
- Machine learning-based methods - Use machine learning algorithms such as Random Forest, Support Vector Machines, and neural networks to detect peaks in time series data.
- Hybrid approaches - Combine multiple methods to improve the detection of peaks. For example, using moving average and wavelet together.

The best approach depends on the specific characteristics of the time series data and the desired level of precision.

There are also some general issues that comes with peak detection. Some of them include:

- Noise: Noise in the data can make it difficult to accurately identify peaks, as small fluctuations in the data can be mistaken for true peaks.

- Overlapping peaks: In some cases, multiple peaks may occur close together in time, making it difficult to distinguish between them.

- Multiple scales: Peaks can occur at different scales in the data, such as local peaks and global peaks, and different methods may be better suited to detecting peaks at different scales.

- Non-uniform sampling: Irregular or non-uniform sampling of the data can make it difficult to accurately identify peaks, as the data may not be evenly spaced in time.

- Complex shapes: Some peaks may have complex shapes that are difficult to accurately detect, such as asymmetric peaks or peaks with multiple humps.

- Parameters sensitivity: Some peak detection methods are sensitive to the choice of parameters, such as the window size for moving average or the threshold for peak detection, making it difficult to achieve optimal performance.

Most of these problems can be mitigated by using appropriate preprocessing before detecting the peaks, using the right algorithm according to the use case, and passing correct parameters to the algorithm.

3 Current State Analysis

3.1 Predictive Maintenance

Predictive maintenance is a proactive maintenance strategy that uses data and analytics to identify when equipment is likely to fail and then performing maintenance before that failure occurs. It is a departure from traditional reactive maintenance, where maintenance is only performed after a failure has occurred. Predictive maintenance has the potential to improve equipment reliability, increase safety, and reduce downtime and maintenance costs.

In this context, Hashemian[21] provides an overview of the current research and development of predictive maintenance technologies, particularly those related to condition monitoring techniques.

The paper discusses three categories of condition monitoring techniques: passive techniques that use signals from existing process sensors to verify the performance of sensors and identify problems in the process, passive techniques that use signals from test sensors to measure parameters and trend vibration amplitude to identify degradation or failure, and active techniques that involve injecting a test signal into the equipment to measure its response and diagnose its performance.

The passive techniques using existing process sensors involve monitoring process data such as resistance temperature detectors (RTDs), thermocouples, or pressure transmitters. The data is analyzed to identify trends or changes that could indicate a problem or degradation of the equipment. For example, if the temperature of a machine is consistently higher than expected, it could indicate that there is a problem with the cooling system, and maintenance can be performed to prevent a failure. The benefit of using existing sensors is that they are often already in place, so the costs of implementing this type of monitoring are relatively low.

The second category of passive techniques involves the use of test sensors that are installed on plant equipment, such as rotating machinery. These sensors measure parameters such as vibration amplitude, which is then trended to identify the onset of degradation or failure. This category also includes the use of wireless sensors to provide additional data points, allowing plants to measure multiple parameters to cover not only vibration amplitude but also ambient temperature, pressure, humidity, etc. The benefit of this type of monitoring is that it can detect degradation earlier than monitoring existing sensors since the sensors are installed specifically to measure the relevant parameters.

The third category of techniques is active and involves injecting a test signal into the equipment to measure its response and diagnose its performance. For example, the response time of temperature sensors (RTDs and thermocouples) can be measured by applying a step current signal to the sensor and analyzing the sensor response. Cable anomalies can be located by a similar procedure referred to as time-domain reflectometry (TDR). This test involves a signal that is sent through the cable to the end device. Its reflection is then recorded and compared to a baseline to identify impedance changes along the cable and thereby identify and locate anomalies. This type of monitoring requires more equipment and is more expensive than the previous categories, but it provides more detailed information about the equipment's performance.

The paper concludes by emphasizing the importance of deploying predictive and online strategies that assume any failure can occur at any time. The onset of equipment failure may manifest itself in data generated by the methods used to monitor the equipment, providing clues as to whether the equipment should be repaired, replaced, or left to continue in operation. Integrating the predictive maintenance techniques described in this paper with the latest sensor technologies will enable plants to avoid unnecessary equipment replacement, save costs, and improve process safety, availability, and efficiency. Ongoing research promises to deliver technologies that may be applied remotely, passively, and online in industrial processes to predict failures before they occur.

In this research, change point analysis has been used to detect changes in sensor data collected by passive techniques, such as those using signals from existing process sensors, it improves the diagnostic capabilities of these techniques and enable earlier detection of equipment problems.

3.2 Anomaly prediction from sensor data

Analyzing sensor data for anomaly detection is a common method of predicting irregularities. Giannoni[22] et al (2008) points out that the increasing use of the Internet of Things (IoT) has led to a massive growth in the amount of time series data generated by IoT devices. However, this data is often noisy, incomplete, and subject to anomalies, which can make it difficult to analyze and extract meaningful insights from. Anomaly detection is therefore an important task in IoT applications, as it can help to identify abnormal patterns of behavior or events that may indicate problems or opportunities for improvement. To address this problem, the authors propose a novel approach for anomaly detection in IoT time series data that combines different machine learning techniques. Specifically, they propose using autoencoders and one-class support vector machines (SVMs) to improve the detection of anomalies in these types of datasets. In the pre-processing stage, the authors use a sliding window approach23 to split the time series data into fixed-size windows. They then use a feature extraction method to transform each window into a lower-dimensional feature space, which captures the essential characteristics of the time series data.

In the detection stage, the authors use an autoencoder to learn a compressed representation of the pre-processed time series data. They then use the reconstruction error of the autoencoder as a measure of anomaly score, with higher reconstruction errors indicating a higher likelihood of an anomaly. The authors also use a one-class SVM to learn a decision boundary that separates the normal data points from the anomalous data points.

The authors evaluate their approach on a new dataset that they propose for evaluating anomaly detection methods in IoT, which they call the "IoT-TS-Benchmark". The authors compare the performance of their hybrid approach to

several other baseline methods and show that their hybrid approach outperforms the baseline methods in terms of accuracy and efficiency.

3.3 Change points analysis for anomaly detection

There are some approaches that have been used to predict breakdowns of machines using change points data as well. These approaches range from traditional statistical methods, such as change point detection algorithms, to more advanced machine learning techniques such as neural networks and support vector machines.

One approach is the use of univariate change point detection methods[24], which focus on detecting changes in a single time series. Univariate change point detection is a statistical technique for identifying points in time where a change has occurred in a single time series data. The goal of univariate change point detection is to identify these points of change as accurately as possible, which can then be used for further analysis and decision-making. Chen[25] et al (2020) sprovide a comprehensive overview of various methods for detecting change points in univariate data. They classify the different methods into several categories, including classical methods, Bayesian methods, and frequentist methods, and provide a detailed explanation of each. The authors also discuss the challenges associated with univariate change point detection, such as high-dimensional data and non-stationarity, and the methods used to overcome these challenges. This thesis uses univariate method of detecting change points.

Another approach is the use of multivariate change point detection methods, which focus on detecting changes in multiple time series simultaneously. Li and Wang[26] discusses this technique which is used to identify points in time where a change has occurred in multiple time series data. The goal of multivariate change point detection is to accurately identify these points of change, which can then be used for further analysis and decision-making. These methods can be more effective than univariate methods because they can consider the relationships between different time series, and can also be used to detect more complex changes in the data.

There have been several research done in the analyzing change points data for anomaly detection.

Guoliang Lu[27] et al. (2016) highlights the need for automatic machine monitoring in industries and proposes a new framework for change-point detection in machinery monitoring. The proposed framework uses an automatic regression (AR) model to measure anomalies in a given time series and then employs a statistical test using martingale for detecting potential changes in the series. The framework has three key properties: it does not require any prior knowledge on the given data, it is self-conducted, and it is efficient.

The authors have experimentally tested the proposed framework using testing data captured in real scenarios, and the results demonstrated the effectiveness and realizability of the framework for change-point detection in machine monitoring. The proposed framework can be directly applicable in many real-world applications, and the authors plan to optimize its computational efficiency and apply it in workshops for practical usage. The proposed framework offers a promising approach for automatic machine monitoring and has the potential to improve the efficiency and effectiveness of fault detection in rotating machinery components.

3.4 Anomaly detection in change points data from IoT sensors

One of the sources from where change points can be generated is data from IoT sensors. An approach to detecting anomalies in IoT time series data has been presented by Apostol et al[28]. The proposed method combines change point analysis and statistical modeling to identify significant shifts in the data generating process and subsequently detect anomalies.

The authors explain that traditional anomaly detection methods can be inadequate for IoT time series data because they often assume that the data is stationary and normally distributed[29]. However, IoT data is often non-stationary and can exhibit various types of distributions, making it challenging to detect anomalies accurately. The proposed method overcomes these challenges by

identifying change points in the time series data, which represent significant shifts in the data generating process. Then, it uses a statistical model to compare the observed data to the expected data after the change points and identify anomalies.

The paper provides a detailed explanation of the proposed method, including the mathematical models used for change point analysis and anomaly detection. The authors also describe the implementation of the method and provide experimental results on several real-world IoT datasets to demonstrate its effectiveness. The results show that the proposed method outperforms several other state-of-the-art anomaly detection methods in terms of accuracy, precision, and recall.

The paper makes a significant contribution to the field of anomaly detection in IoT time series data by presenting a novel approach that overcomes the challenges associated with traditional methods. The proposed method can be useful in a wide range of applications, including environmental monitoring, healthcare, and smart cities, where IoT time series data is commonly collected.

3.5 Summary

This research proposes similar kind of work for predicting anomalies from change points data which are generated from IoT sensor data. The analysis for prediction does not involve any complex machine learning algorithms in this case, but a simple and straightforward approach of finding the local peaks in the time series change points data to predict a possible breakdown.

4 Data Specifications

4.1 Change Points Data Description

The change points data is the primary data set for this thesis work, which have been generated as prior work. The KPI data from the elevator rides are generated by an analytics engine in edge locations in the devices, which are then sent to cloud environment such as MQTT (Message Queuing Telemetry Transport) for further processing.

There are different kinds of sensor data that are generated from an elevator. The data is fetched by the analytics and pipelines and different KPIs are generated. One particular pipeline calculates the frequency spectrum of the acceleration signal of the elevator from the constant velocity phase of the elevator ride, from the movement sensor data. This is called the “constant movement spectrum”.

The data for this pipeline is acceleration signal in direction z (vertical direction). The pipeline selects the longest consistent range of values in the constant velocity movement phase and calculates Fast Fourier Transform on the signal and forms the frequency spectrum. It then wraps the transformed data into a format which is suitable for data transfer and publishes to MQTT.

Each amplitude ai where i ∈ [1, N], in produced spectrum is calculate as follows:

$$a_i = \frac{\sqrt{\mathrm{Re(Xi)}^2 + \mathrm{Im(Xi)}^2}}{\mathrm{N}}$$

where $\mathrm{X}i \in \mathrm{X}$ and X is the Discrete Fourier Transform of the input signal. N is the number of amplitudes in a single spectrum.

The second important pipeline which is relevant to the data of this thesis is the frequency domain change points analysis pipeline. This pipeline produces a change point statistic over multiple elevator rides. The statistic calculates the Fstats statistic over each frequency band of Fourier transform X*i* taken from the

constant-speed phase of each elevator ride, which is generated by the first pipeline. So, the input data to this change point pipeline is the Fourier transform data from multiple elevator rides collected to local database of the device, that is the constant movement spectrum. To produce a change point time series, Fstats is applied to each amplitude column $X_{1,n}$ to $X_{n,N}$ in a matrix and then take the row wise sum.

The change points data are generated in edge locations in an elevator device. The data is then sent to cloud platform through MQTT messages, where it is stored in a database. Although an elevator equipment can have more than one device, it is assumed here that there is one to one mapping between an elevator equipment id and its corresponding device id. The data used in this thesis have been queried from the database and saved in csv files where it has been read by a python program.

The following figure shows an example of change points data:

deviceid	k_timestamp	start_times	change_points
357042064016999	2020-07-03T22:01:37.337Z	[2020-06-28T09:55:10+00:00, 2020-06-28T10:14:31+00:00, 20	[0.0, 4260.272, 6423.361, 8867.662, 11726.055, 12599.37, 14428.924, 16058.857, 152
357042064016999	2020-07-21T20:01:33.657Z	[2020-07-14T20:14:47+00:00, 2020-07-14T20:15:25+00:00, 20	[0.0, 3109.28, 4314.739, 3067.229, 8880.733, 7303.038, 6752.637, 9751.182, 10487.45
357042064016999	2020-07-30T21:01:33.771Z	[2020-07-23T07:50:42+00:00, 2020-07-23T08:01:47+00:00, 20	[0.0, 8740.418, 10417.255, 15260.904, 15670.196, 21088.037, 23485.811, 29371.975, 2

Figure 4: Example of Change Points data

The description of the columns are as follows:

- *deviceId*: The id of the device connected to the elevator.
- *k_timestamp*: The timestamp when the change points pipeline was run.
- *start_times*: The timestamp value for which the change point is calculated.
- *change_points*: The value of the change point.

4.2 Service Order Data

When the data from an IoT device integrated to an elevator/escalator indicates that there is a fault, a service order is created by an analytical workflow. The service orders are dispatched to the technicians and are worked upon on the maintenance sites. The necessary information is then updated by the technicians accordingly in the service order. The historical service orders are used in this thesis work for correlating with the change points. The service order data contains information about when and for which equipment the service order was raised and what were the actions taken.

The following figure shows an example of service order data from a csv file.

equipment_number	datetime	affected_part	action_taken	status
9876	2020-01-19 08:32:58	1130	656	Finished
9065	2020-01-20 10:04:35	2130	587	Finished
3653	2020-01-23 10:02:33	3450	984	Finished

Figure 5: Example of Service Order data

The description of the columns are as follows:

- *equipment_number*: The id of the elevator equipment.
- *datetime*: The timestamp in which service order was received.
- *affected_part*: The code of the elevator part that was affected, for which the service order was issued. Example: Doors, Shaft.
- *action_taken*: The code for the action that was taken for the service order. Example: Technical Failure, Vandalism.
- *status*: The current status of the service order.

4.3 Equipment Data

The equipment data consists of a mapping between *equipment_number* described in section 3.3 for which service order data is generated, and *deviceId* described in section 3.2 for which change_points are calculated. An equipment may have multiple devices attached to it.

The following figure shows an example of equipment data mapping:

equipment_number	device_id
42657999	357042064199999
40061111	357042011111748

Figure 6: Example of Equipment data

The description of the column are as follows:

- *equipment_number*: The id of the elevator equipment
- *device_id*: The id of the device attached to the equipment. This is the same id as in change points data.

5 Implementation

The implementation for the plotting the change points time series with service order events and calculating the accuracy of the model, has been done in python programming language.

The following figure shows the high-level implementation flow.

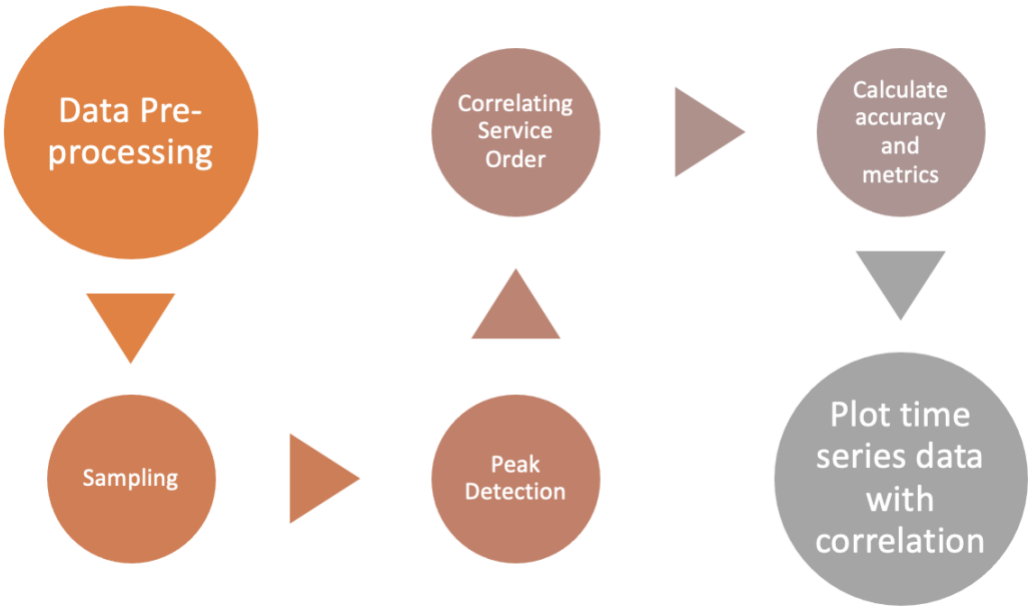

Figure 7: High level implementation flowchart

5.1 Data Pre-processing

The change points data is present in csv files for each month, along with service order data. The first step of processing is to aggregate the change point data and service order data into a single *pandas dataframes* respectively to make further data processing easier. This is done by using the *path.join* command under *os* library to join all the files and then read the csv files using data frame.

Since the service order data is mapped with equipment number, and change points data is mapped with deviceId, the equipment number for the device id in the change points data can be easily looked up in the csv file containing direct mapping of equipment number vs device id. The equipment data file is also converted to pandas dataframe for lookup.

As shown in section 3.2, the change points data table has columns *start_times* and *change_points*. Each change point value corresponds to a timestamp in the *start_times*.

Since the data is aggregated for all months, the two columns in change points data (*start_times* and *change_points)* in the dataframe contains data for all months. Each *start_times* row contains timestamps at various points in a day, and each row in change_points contain its corresponding change point value. There are 1000 timestamps in one day and therefore 1000 change point values for each timestamp. There can be cases where a row can contain timestamps from a day which is same as previous row.

This research tries to find the correlation between maintenance needs of an elevator equipment and the change point values. The information for the maintenance needs is present in service order data. Although there are more devices (and hence equipments) present in the change points data, only those equipments are considered which is present in service order data, so that the correlation with the change points and maintenance needs can be found.

For each equipment present in the service order dataframe, a new pandas dataframe is created from the aggregated dataframe consisting of change point values for all equipments for all months. This new data frame consists of two columns, *start_times* and *change_points*. If there are any duplicate timestamps but different values, a random one is taken.

The following table shows an example of rows of a dataframe named *data_df* for an equipment consisting of *start_times* timestamps and *change_points* values.

Table 3: Example of dataframe consisting of timestamp and change point values.

index	start_times	change_points
0	2020-05-15 07:04:52+00:00	6359.341
1	2020-05-15 07:07:14+00:00	5346.730
2	2020-05-15 08:08:41+00:00	2957.364
….2804	2022-01-12 18:00:00	4661.829

In a time-series data analysis, it is a good practice to set the timestamp as the index[30]. There are various advantages of having timestamp of the series as the index since several direct and faster operations can be performed. For example, all the rows can be efficiently fetched for a particular year or a date range by passing the year as an argument to the dataframe. As shown in the example table 3, the index for the dataframe is the default row number, and the *start_times* column has timestamps as string. To make the *start_times* as datetime index, the datatype of the values in that column is changed from string to datetime, and then *set_index* functionality is used to set the datetime index of the dataframe. The index can now also be sorted easily.

Now from the example data above, if rows for a particular date range are needed to be fetched, the following python lines can be used.

```
data_df = data_df.set_index(keys='start_times')
data_df = data_df.sort_values('start_times', axis='index')
print(data_df['2020-05-15 07:00:00' : '2020-05-15 07:10:00'])
```

The output rows from the example above produces a sliced part of dataframe as shown below.

Table 4: Slice of dataframe for a particular date range based on datetime index

start_times (index)	change_points
2020-05-15 07:04:52+00:00	6359.341
2020-05-15 07:07:14+00:00	5346.730

5.2 Sampling

In a time-series data, values or data points are collected in order of time. Usually, the sequence of data points is taken equally spaced points in time. In this case, the timestamps present in the start_times column are not equally spaced. The samples are varied over a period of time.

To make the data equally spaced in the time series, sampling is used. Sampling can change the frequency of time series data points. A resample function is provided in the pandas dataframe package which provides a convenient method for resampling time series data. The frequency of the sampling can be controlled by passing suitable parameters to the function.

Downsampling is the process in which data which is recorded at a high frequency or sampling rate is compressed into smaller sampling rate. Thus, the number of data points decreases. For example, if there are recordings taken each day, but only monthly observations are needed in the time series model, downsampling has to be used.

Upsampling or interpolation is the opposite of downsampling. Upsampling increases the frequency of time series to contain more data points. For example, if it is needed to increase data points from minutes to seconds, upsampling needs to be used. This process of identifying or inventing new unknown data points that lie between known data points is knows as interpolation. There are different methods used to approximate the unknown or missing data points. The dataframe

resample interpolation method provided by python has some alternative methods to interpolate the data. Some of them are described as following:

- Linear Method: This interpolation method assumes the missing values as equally spaced from previous data points and the data points after the missing values. This is the default method.

- Polynomial method: This interpolation method uses the numerical value of the index and fills the missing values with the lowest possible degree that passes through the data points. The polynomial order needs to be passed in the arguments. A polynomial of degree 1 is same as linear.

- Padding: Interpolation through padding fills the missing values with existing values in the dataset. It can be backward filling by filling with last known value, or forward filling which is the opposite.

In this thesis project linear method has been used to interpolate the change point values. Change point values are resampled for each day. The following lines shows how resampling, and interpolation is done using python.

```
data_df_resampled = data_df.resample('D').max()

data_df_resampled['change_points'] =
data_df_resampled['change_points'].interpolate(method='linear')
```

The dataframe *data_df* is downsampled as needed, to generate values for every day. It is resampled using ‘D’ parameter denoting one day with the max function, which means it will take the highest change point value that has occurred in that day among all the values. But there can be cases where there is missing data for some days. The max function will produce NaNs for those timestamps. The NaNs are then interpolated with linear method. The following table gives a sliced example of the dataframe for the first four rows of how the data samples look after resampling.

Table 5: Example of dataframe after resampling

start_times (index)	change_points
2020-05-15 00:00:00+00:00	123471.398
2020-05-16 00:00:00+00:00	120496.516
2020-05-17 00:00:00+00:00	63394.484
2020-05-18 00:00:00+00:00…	677459.375

5.3 Detecting peak values in the time series data

The peak detection algorithm used in this paper is *find_peak*s function from scipy signal python package, and this function is used to identify the local maxima in the resampled data. These local maxima are considered as peaks in the change point data.

As described in the official documentation[31], in the context of this function, a peak or local maximum is defined as any sample whose two direct neighbors have a smaller amplitude. For flat peaks (more than one sample of equal amplitude wide) the index of the middle sample is returned (rounded down in case the number of samples is even). For noisy signals the peak locations can be off because the noise might change the position of local maxima.

The scipy.signal.find_peaks function uses the *peak_prominences* algorithm to find peaks in a signal. This algorithm is based on the concept of peak prominence, which is the vertical distance between a peak and its highest contour line. The function uses a combination of parabolic interpolation and a modified version of the peak threshold algorithm from the peakdetect package to calculate the prominence of each peak and return the indices of the peaks that meet a user-specified threshold.

This function makes peak detection very convenient. It uses a peak detection algorithm that is based on the first derivative of the data, and it can handle noisy

data and multiple peaks. Additionally, the function allows for the setting of different parameters such as height, threshold, and distance, which can be used to filter out less prominent peaks. There are different ways to adjust the height and threshold parameters based on a given change points data. In this research mean and standard deviation methods have been used to set the *height* and *threshold* parameters. The advantage of using the mean and standard deviation to set the parameters is that it allows you to set thresholds for the *height* and *threshold* parameters that are based on the distribution of the change point data. The height parameter is set to the mean of the change point data + 2 * standard deviation of the change point data. This will ensure that the peaks returned are at least 2 standard deviations higher than the mean of the data and are therefore more likely to be prominent peaks. Similarly, threshold parameter is set to mean - 2 * standard deviation of the change point data. This approach helps in filtering out less prominent peaks by removing the noise from the data, which can be useful when analyzing the change point data.

The following line of code calculates the peaks in the timeseries dataframe data_df_resampled.

```
mean = data_df_resampled['change_points'].mean()
std = data_df_resampled['change_points'].std()
peaks, properties = find_peaks(data_df_resampled['change_points'],    height=mean + 2*std,
threshold = mean - 2*std, distance=3)
```

There are two outputs for this function: *peaks,* and *properties*. The *peaks* output output is an array of integers, where each integer represents the index of a local maximum found in the data. This can be used to identify the location of the peaks in the data. The *properties* output is a dictionary with several keys and values. Each key corresponds to a property calculated for each peak, such as the *peak_heights, prominences, left_bases, right_bases* and others. These properties provide more information about the shape and significance of the peaks and can be used to filter or classify the peaks.

When the indexes are found for the peaks, the existing dataframe is enriched with this information. A new column *isPeak* is added which denotes if the change point value is a peak or not. The new column is initialized as False for all rows, and then the indexes for which peaks occurred are marked as True. An example of the dataframe is given in the below table.

Table 6: A dataframe example of how the information of the peaks is incorporated

start_times (index)	**change_points**	**isPeak**
2020-05-15 00:00:00+00:00	123471.398	False
2020-05-16 00:00:00+00:00	120496.516	False
2020-05-17 00:00:00+00:00	63394.484	False
2020-05-18 00:00:00+00:00…	677459.375	True

5.4 Flagging service order data

Since the thesis correlates the change point peaks to events in which service orders or maintenance needs were generated, it is important to add this information to the dataframe. As explained in section 4.3 the *datetime* column in the service order data provides the timestamp at which the service order was generated for that equipment. In the code, all the service order rows for that equipment are stored in a dataframe. This dataframe is iterated and the received date is rounded off to its nearest minute if it is not already. The resampled dataframe data_df_resampled contains datetime indexes for each minute between the first and last timestamps of original change point data. That means if the receives date falls inside the range of the first and last recorded timestamp, then the datetime index would also be present. So, the received date is matched with the index datetime and it is marked whether service order was received for that row or not. As done for indicating the peaks, same strategy is applied for

service order received indication. A new column *so_received* is created and initialized with False values. And during the iteration, as explained earlier, the matching row is marked as True. An example of the dataframe after adding the service order information is given below.

Table 7: Example of dataframe with service order data

start_times (index)	**change_points**	**isPeak**	**so_received**
2020-03-19 09:15:00	9458.012	False	True
2020-03-19 10:05:00	10908.800	True	False
2020-03-19 11:00:00	2516.378	False	False
2021-01-12 18:00:00	4661.829	False	True

5.4.1 Filtering of service order data

Section 4.4 explains how service order data is inserted into the dataframe. But all service order data is not considered, and some needs to be filtered out based on affected part code. As defined in section 3.3, the affected part code of a service order gives information about which part or component of the elevator was affected for which the service order was issued. The pipeline that generates change point values takes into consideration the constant speed phase of the elevator in vertical directions. There are a lot of elevator components that are not affected by this elevator movement; hence the change points cannot be correlated with service orders which were generated for such kind of components. For example, control box or switch components are not affected by the vertical movement and hence the service orders cannot be correlated. The program filters out such kind of service orders from the calculation and plotting.

5.5 Calculating TP, TN, FP, FN

The paper mentions True Positives (TP), True Negatives (TN), False Positives (FP) and False Negatives (FN) in section 2.1. These are the metrics which have been used in this research to define how accurate the correlation or prediction is for the peak of the signal to the service order of the elevator. The metrics are defined as the following in this use case:

True Positive (TP): When a change point peak occurs and a service order has been received within a defined period of lead time after the peak occurred, then the event is concluded as a true positive. A true positive indicates that there was a correlation between change point peak and the service order. Example: If the defined threshold period is 15 days, and a peak has occurred on 27th January 2022, and a service order was received on 1St February 2022, then the sample would be true positive.

True Negative (TN): When there is no peak in the change point and there is no service order received within a defined period of lead time, then the sample would be considered a true negative since it correctly predicted that there will not be any service orders. Example: If the defined threshold period is 15 days, there is no peak on 1st March 2022, but there is also no service order received till March 16th, the event would be True Negative.

False Positive (FP): When a change point peak occurs but there is no occurrence of service order within a defined period of data after the peak occurred, then the sample would be concluded as false positive. A false positive indicates that although the peak occurred, there is no correlation with the service order occurrence for that peak. For example: Taking the define threshold lead time as 15 days, if there is a peak on 1st March 2022, but there is no service order received till March 16th 2022, the event would be False Positive.

False Negative (FN): When there is a service order received for an elevator but there are no preceding peaks within 15 days of the service order timestamp, then the sample can be marked as False Negative. False negative indicates that the maintenance need was not identified by the model. For example, there is a

service order received on March 16th 2022, but there are no peaks from 1st March 2022 to March 16th 2022, then the sample would be False Negative.

The confusion matrix in this case would be as shown in table below.

Table 8: Confusion matrix for predicting if an elevator needs service

	Actual:No Service Order	**Actual: Service Order**
Predicted:No Service Order	True Negative	False Negative
Predicted: Service Order	False Positive	True Positive

As mentioned in section 2.1, from these parameters precision and specificity can be calculated.

For example, if there are 4 TP, 10 FP, 100 TN, 68 FN events occurred for an elevator equipment as evaluated by the model, the confusion matrix should look like this:

Table 9: Confusion matrix for example samples

	Actual:No Service Order	**Actual: Service Order**
Predicted:No Service Order	TN (100)	FN (68)
Predicted: Service Order	FP (10)	TP (4)

The evaluation matrix would be:

Precision = 4 / (4 + 10) = 0.29

In an ideal scenario of all true positives, precision will be 1.

Specificity = 100 / (100 + 10) = 0.9

Recall = 4 / (4 + 68) = 0.05

5.6 Visualizing data by plotting data points

Apart from calculating the metrics of how change points peaks affect the health of the elevator, it is also important to plot a time series graph of the same. Visualizing plots in time series data is important because it allows us to quickly identify patterns and trends in the data that may not be immediately apparent from looking at the raw data. This can help us to understand the underlying dynamics of the system we are studying, and can also help us to identify potential outliers or anomalies in the data. Additionally, visualizing the data can also help us to gain a better understanding of the relationships between different variables in the data, and can also help us to identify any potential relationships or correlations that may not be immediately apparent from looking at the raw data.

Visualizing time series data through plots allows for an easier and more intuitive understanding of the data. It allows for quick identification of patterns, trends, and anomalies in the data. For example, a plot of time series data can clearly show the presence of a change point, such as a sudden increase or decrease in the data, which may not be as obvious when looking at the raw data in a table format. Additionally, plots can also help with identifying seasonality or periodicity in the data. Furthermore, visualizing the data can also help with identifying any outliers or missing data points.

The y-axis displays the change points while the x-axis represents the time series. The time series timestamps are taken from the column *start_times*, which is also the index of the dataframe. The graph also illustrates the peaks of the change points and service orders. The graph plots the service orders on the change point values that occurred at that timestamp.

6 Results

6.1 Plotting results

The following figure shows a time series plot of change points vs start times along with the peaks and marks the timestamps in x-axis where the service order was received. The title also gives information about the identification numbers of the equipment, and the metrics discussed previously used to calculate the accuracy.

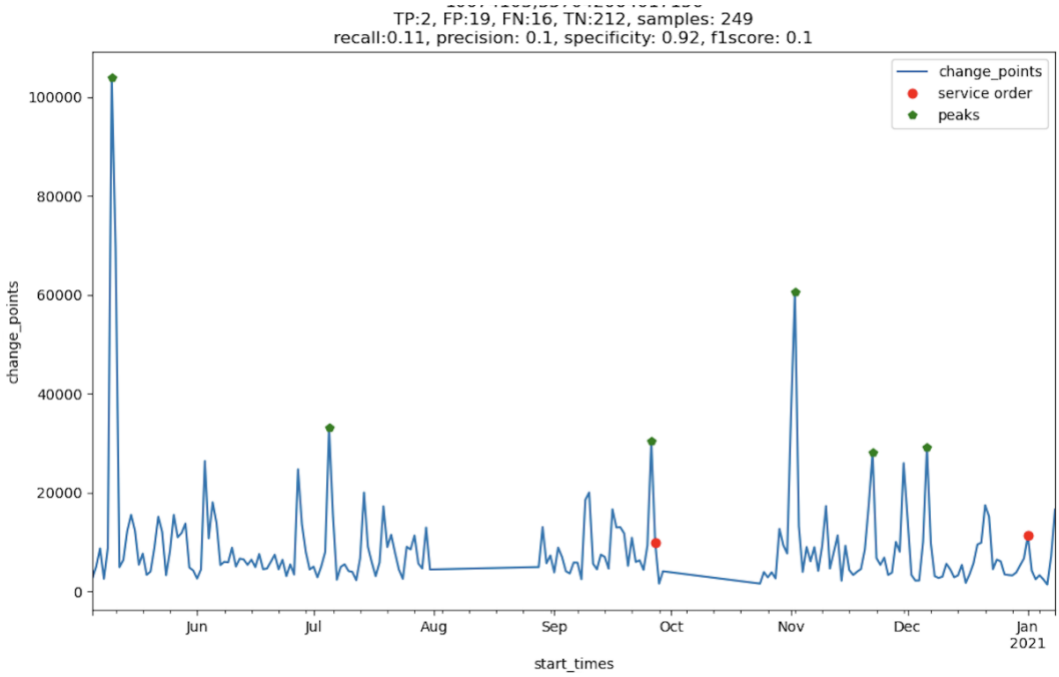

Figure 9: Time series change points plot of an elevator equipment

From the figure it can be observed that for the first two peaks which have been identified are false alarms, there was no service need generated for it. Shortly after third peak there is a true positive since there was one service order generated (just before October).

The above result was generated for a threshold lead time of 15 days. That means that the program considered a peak to be true positive if there was a service order generated within 15 days after the change point peak.

Let's look at another example with better recall, specificity and F1 score.

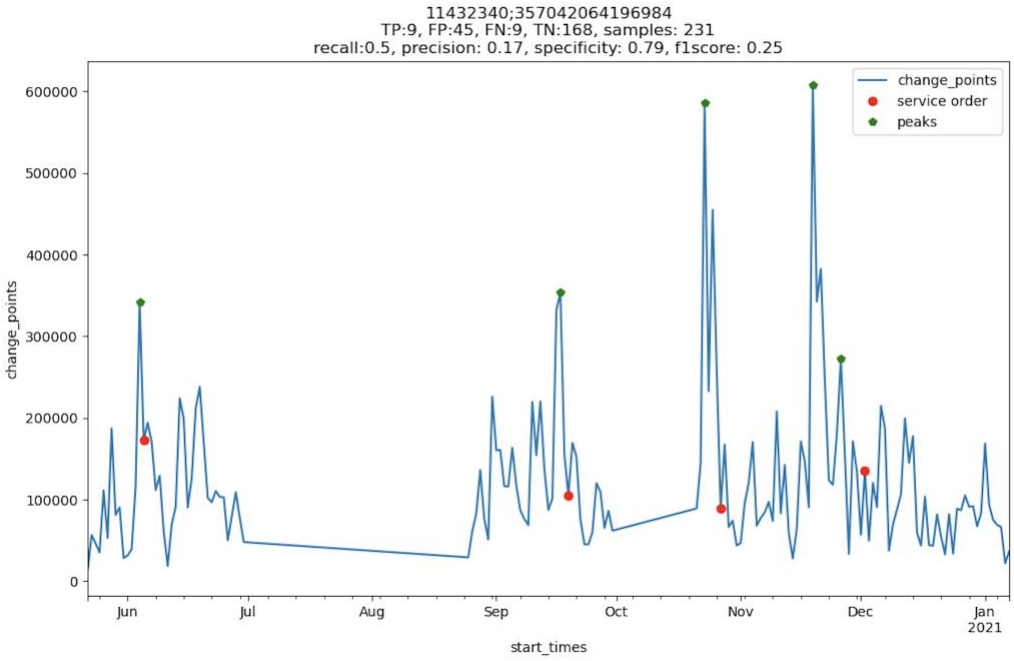

Figure 10: Time series change points plot of an elevator equipment with better scores

6.2 Evaluation for different parameters

For this research there were 10 elevator equipments chosen to test against this model. The parameters for this model were changed and data was collected for each run with the changed value of the parameters. The parameters/factors that were modified are as follows:

- Threshold lead time: This is the amount of time in days after a peak within which a service order is expected to be received to classify the peak as a true positive.

- Distance between peaks: This is the minimum horizontal distance in days between two neighbouring peaks. For example, if a peak occurs on 1st January and the distance between peaks is 3 days, then all maxima from January 1 (after the first peak) and January 4 would not be eligible to be considered as a peak. It is a parameter given in the *spicy.find_peaks* function.

- Peak factor - Peak factor is a factor which influences the minimum height and minimum threshold of a maxima to be eligible as peak. The parameters *height* (minimum height of peaks) and *threshold* (minimum vertical distance to its neighbouring samples) are passed to the function *spicy.find_peaks*. Peak factor is the multiplier with the standard deviation of the change points. As discussed in section 4.3, the *height* is set as *mean + 2 * standard deviation* and threshold is set as *mean - 2 * standard deviation.* Here 2 is the peak factor, and the value is modified to see how it influences the performance of the model.

The following are results when the parameter is following:

Threshold lead time: *15 days*

Distance between peaks: *5 days*

Peak factor: *2*

Table 10: Metrics results for first set of parameters

Equipment	Recall	Precision	F1Score	Specificity	Accuracy
E1	1.00	0.17	0.29	0.92	0.92
E2	0.50	0.17	0.25	0.79	0.77
E3	0.25	0.17	0.2	0.96	0.93
E4	0.36	0.14	0.2	0.89	0.87
E5	1	0.11	0.2	0.93	0.93
E6	0.4	0.12	0.18	0.94	0.93
E7	0.5	0.1	0.17	0.92	0.91
E8	0.25	0.17	0.2	0.96	0.93

Equipment	Recall	Precision	F1Score	Specificity	Accuracy
E9	0.43	0.14	0.21	0.92	0.91
E10	1	0.11	0.2	0.89	0.89
Average	**0.57**	**0.14**	**0.21**	**0.91**	**0.90**

For the second run, the parameters are modified as follows:

Threshold lead time: *7 days*

Distance between peaks: *3 days*

Peak factor: *2*

Table 11: Metrics results for second set of parameters

Equipment	Recall	Precision	F1Score	Specificity	Accuracy
E1	1	0.27	0.43	0.95	0.95
E2	0.47	0.33	0.39	0.93	0.89
E3	0.25	0.29	0.27	0.98	0.95
E4	0.29	0.36	0.32	0.97	0.93
E5	0.4	0.25	0.31	0.97	0.96
E6	0.4	0.25	0.31	0.97	0.96
E7	0.5	0.15	0.23	0.95	0.95
E8	0.25	0.29	0.27	0.98	0.95
E9	0.43	0.21	0.28	0.95	0.94

Equipment	Recall	Precision	F1Score	Specificity	Accuracy
E10	1	0.16	0.28	0.93	0.93
Average	**0.50**	**0.26**	**0.31**	**0.96**	**0.94**

From the above two tables it can be sen that the Precision and F1Score has increased by 85%, 10% respectively but Recall has decreased slightly.

For the third run, the parameters are modified as follows:

Threshold lead time: *7 days*

Distance between peaks: *3 days*

Peak factor: *2.5*

Table 12: Metrics results for third set of parameters

Equipment	Recall	Precision	F1Score	Specificity	Accuracy
E1	1	0.29	0.45	0.96	0.96
E2	0.33	0.29	0.31	0.93	0.88
E3	0.25	0.5	0.33	0.99	0.97
E4	0.13	0.33	0.19	0.98	0.93
E5	0.4	0.25	0.31	0.97	0.96
E6	0.4	0.29	0.34	0.98	0.97
E7	0.5	0.22	0.31	0.97	0.96
E8	0.25	0.5	0.33	0.99	0.97
E9	0.43	0.21	0.28	0.95	0.94

Equipment	Recall	Precision	F1Score	Specificity	Accuracy
E10	1	0.19	0.32	0.94	0.94
Average	**0.47**	**0.31**	**0.32**	**0.97**	**0.95**

The precision has slightly increased in the third set of parameters and the recall has slightly decreased in the third set of parameters, the other metrices remaining almost same.

The following figures shows the plots for equipment E4, for the three sets of parameters.

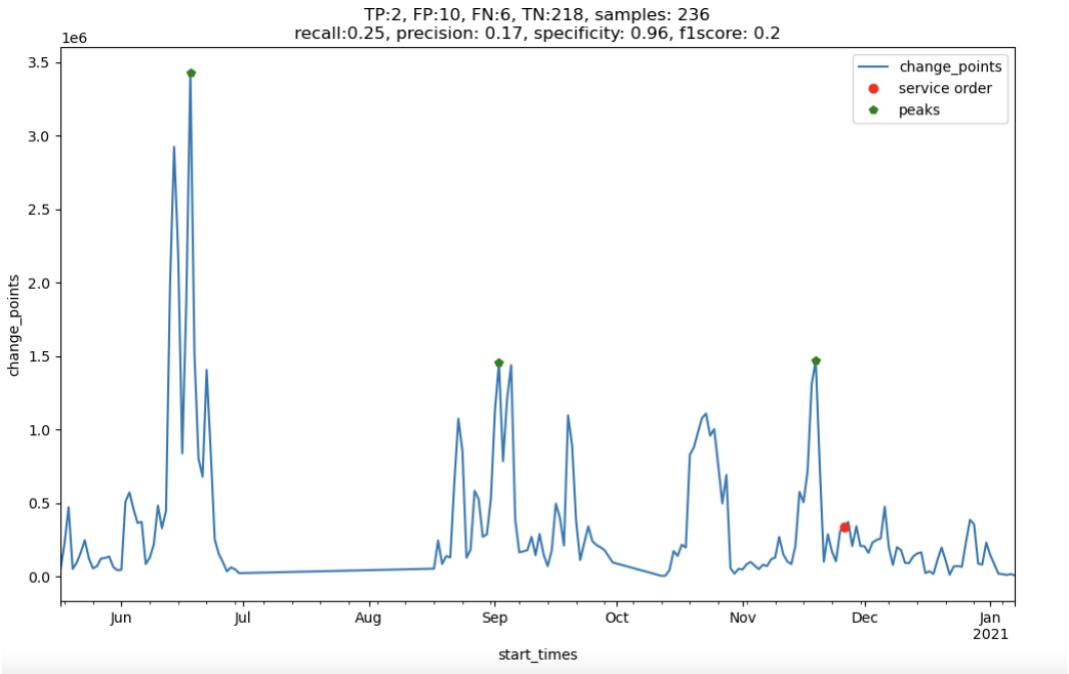

Figure 11: Change points plot for equipment E3 for first set of parameters

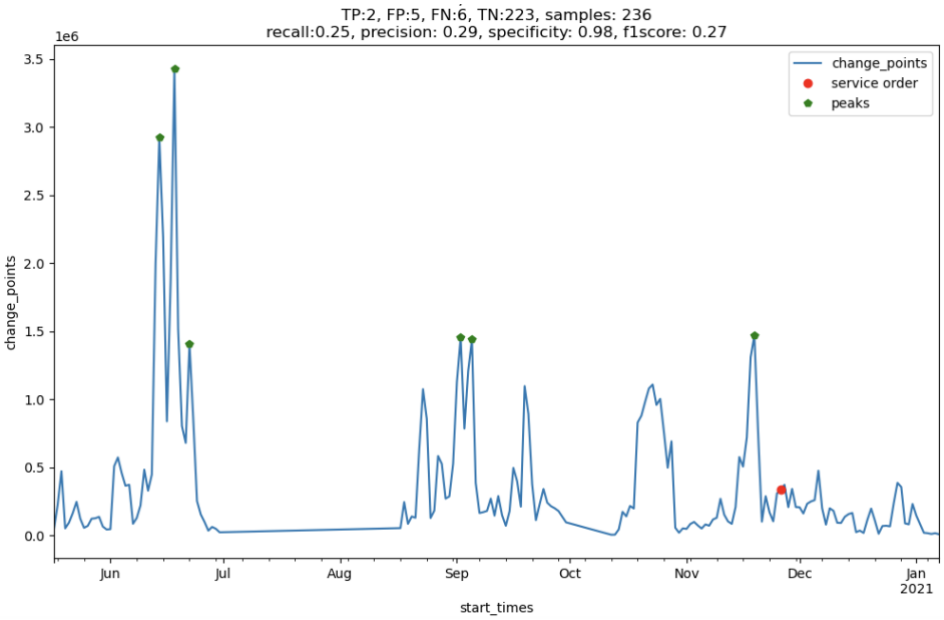

Figure 12: Change points plot for equipment E3 for second set of parameters

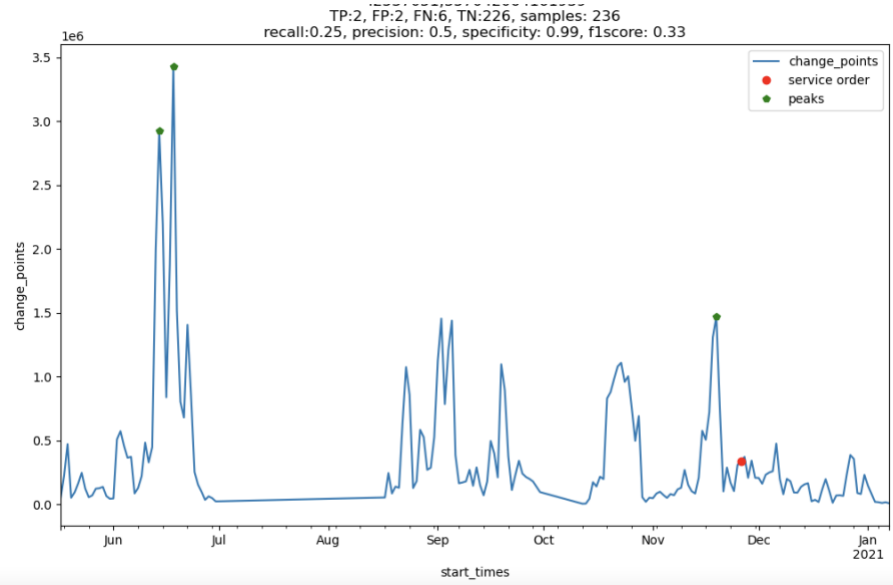

Figure 13: Change points plot for equipment E3 for third set of parameters

It can be clearly seen by comparing the figures from first, second and third set that although the number of true positives remains the same, false positives decreases with the subsequent parameters, and the true negative increases.

7 Conclusion

7.1 Summary

The thesis begins with a comprehensive review of the relevant literature, highlighting the previous work done in utilizing change points data for predictive purposes. It details the commonly used metrics to evaluate the performance of predictive models and provides in-depth explanations of change points, fast Fourier analysis, and peak detection - all of which play a crucial role in the development of the proposed model.

In addition, the data specifications are carefully defined and illustrated with practical examples of equipment data, service order data, and change points data, to facilitate a better understanding. The methodology for implementing the model is thoroughly discussed, including the steps taken for data pre-processing, to ensure a transparent and comprehensive understanding of the research process.

From the result data sets provided in section 5 for three sets of parameters, it is inferred that there is an increase of Accuracy and F1 Score for the third set of parameters, when the peak factor is increased and the minimum distance between two maxima is decreased for a maxima to be considered as a peak. The peak factor increase would lead to a smaller number of peaks since the threshold is now higher. The decrease in two neighbouring maxima distance increased chances of having more frequent peaks.

In general, it is observed that the specificity and accuracy is very high while the F1 Score and precision is comparatively lower. This indicates that the model is good at identifying negative cases (high specificity). It means that the cases when the change points peaks do not occur there is no service need have been more accurately determined. This is also because the data set is such that it is not balanced. There are very few instances when there is a service need which is affected by change points in general, and the peaks in change points are also

few. So, there are fewer true positives, which lowers the values of F1 Score and Precision. There are more negative cases in actual in the dataset than positive cases.

Since F1 Score is a measure of balance between precision and recall, if precision is low then it means that the model is classifying more false positives and that is why it is not able to balance recall, which is medium in this case.

7.2 Limitations of data set

There are several limitations of the data sets that affect the quality of results. Some of them are:

- Data Availability: The data set is incomplete for some periods of time due to various reasons like sensor malfunctioning. The interpolation functions fill the data with values on assumption which can be very different from actual values.

- Data consistency: The data may be inconsistent or noisy if the conditions under which the data was collected from sensor vary significantly. For example, changes in temperature, humidity, or power supply can affect the readings of the sensors.

- Integrity of service order data: This thesis assumes that for all the service orders that were received for specific components of the elevator, the change points should be affected. But it might be that the service order was received for a reason which does not affect the generation of change points. The affected part code which is marked in the data for which the service order was received might also be incorrectly categorized. There are also scheduled maintenances of the elevators which have not been considered. Often a lot of issues are taken care in the regular maintenance visits. There can also be cases where the change points peak if there are temporary vibrations, for

example, trampling inside the elevator, but there are no service order generated for it.

7.3 Future Improvements

There are several improvements that can be done for this model to predict the breakdown of the elevator.

The dataset which has been used is for one year. But with more data, the results are expected to be more accurate. Also, as discussed in section 6.2, the quality of data also affects the results. Better quality of data can be achieved by having a pipeline which feeds continuous change points data to the model in defined schedules, like daily or weekly. The regular maintenance of the elevators factor can also be considered.

This research has defined and discussed the methods to predict anomalies on a test set of ten elevators. The results need to be generated for a greater number of elevators with similar methodologies as describe in the thesis for a conclusive result, whether the peaks in change points affects the breakdown.

This research uses a simple correlation-based model which uses the peak of the change points an indication for the probable breakdown. There are more complex ways of achieving this using different algorithms which might give more accurate results.

For example, Pruned Exact Linear Time (PELT) algorithm is one such strategy for achieving better results. PELT is a well-known algorithm used in change point analysis. The main idea behind PELT is to prune the search space by using a penalty function to determine which change points to include in the final results. PELT provides an exact solution in linear time, making it an efficient and effective method for change point analysis.

To make predictions of elevator breakdowns more accurate, PELT can be used to identify significant changes in the time series data representing the change

points of the elevators. This can be done by calculating the optimal segmentation of the data, which separates the data into contiguous segments where each segment has a homogeneous mean. The change points correspond to the breakpoints between these segments. Once the change points have been identified, they can be used to model the underlying time series data and make predictions about future breakdowns. This can be done by fitting a statistical model to each segment and using it to forecast future values.

References

1 Anmol Bansal, Satyajee Srivastava.Tools Used In Data Analysis: A Comparative Study. International Journal of Recent Research Aspects, 2018 March. Vol 5, 15-18.

2 Agresti, A. Categorical Data Analysis. John Wiley & Sons, Inc. Hoboken, NJ, USA; 2002.

3 Kai Ming Ting. Confusion Matrix, In: Sammut, C., Webb, G.I. (eds) Encyclopedia of Machine Learning. Springer 2011. Springer, Boston, MA. Available from: https://doi.org/10.1007/978-0-387-30164-8_157

4 Kai Ming Ting. True Positive, In: Sammut, C., Webb, G.I. (eds) Encyclopedia of Machine Learning. Springer 2011. Springer, Boston, MA. Available from: https://doi.org/10.1007/978-0-387-30164-8_855

5 Kai Ming Ting. False Positive, In: Sammut, C., Webb, G.I. (eds) Encyclopedia of Machine Learning. Springer 2011. Springer, Boston, MA. Available from: https://doi.org/10.1007/978-0-387-30164-8_300

6 Kai Ming Ting. True Negative, In: Sammut, C., Webb, G.I. (eds) Encyclopedia of Machine Learning. Springer 2011. Springer, Boston, MA. Available from: https://doi.org/10.1007/978-0-387-30164-8_853

7 Kai Ming Ting. False Negative, In: Sammut, C., Webb, G.I. (eds) Encyclopedia of Machine Learning. Springer 2011. Springer, Boston, MA. Available from: https://doi.org/10.1007/978-0-387-30164-8_299

8 Shohreh Deldari, Daniel V. Smith et al. Time Series Change Point Detection with Self-Supervised Contrasting Predictive Coding. Melbourne 2011. Available from: https://arxiv.org/pdf/2011.14097.pdf

9 Samaneh Aminikhanghahi, Diane J. Cook. A Survey of Methods for Time Series Change Point Detection. Springer, May 2017. Available from: https://link.springer.com/article/10.1007/s10115-016-0987-z DOI: 10.1007/s10115-016-0987-z.

10 Christof Schorth, Julien Siebert, Janek Grob. Time Traveling with Data Science: Focusing on Change Point Detection in Time Series Analysis. Fraunhofer IESE, 2021 Sep. Available from: https://www.iese.fraunhofer.de/blog/change-point-detection/

11 Truong C., Oudre L., & Vayatis, N. 2018 A review of change point detection methods. March 2020. Available from: https://arxiv.org/abs/1801.00718.

12 Truong C, Oudre L, Vayatis N. Ruptures: Change point detection in Python. January 2018. DOI: https://doi.org/10.48550/arXiv.1801.00826

13 Do JS, Kareem AB, Hur J-W. LSTM-Autoencoder for Vibration Anomaly Detection in Vertical Carousel Storage and Retrieval System (VCSRS). Sensors. 2023;23(2):1009. doi: 10.3390/s23021009.

14 Chen W, Li X, Dai Y, et al. Anomaly detection and failure prediction in industrial machines using change point analysis. IEEE Transactions on Reliability. 2015;64(1):51-62

15 Liu X, Chen X, Li X, et al. A deep learning framework for change point detection in industrial time series data. IEEE Transactions on Industrial Informatics. 2019;15(9):5231-5239.

16 McKinney W, others. Data structures for statistical computing in python. In: Proceedings of the 9th Python in Science Conference. 2010. p. 51–6.

17 Meinard Müller. The Fourier Transform in a Nutshell. Springer, August 2015. Available from:

https://www.researchgate.net/publication/290440858_The_Fourier_Transform_in_a_Nutshell DOI: doi:10.1007/978-3-319-21944-8

18 Zhang, Y., & Li, Z. (2017). Peak detection in glucose signals using continuous wavelet transform and support vector machine. Journal of medical systems, 41(5), 129.

19 Wang, Z., Liu, Y., & Wang, X. (2015). A novel approach for peak detection in capillary electrophoresis based on the improved particle swarm optimization algorithm. Talanta, 136, 42-48.

20 Girish Palshikar. Simple Algorithms for Peak Detection in Time Series. Jaunary 2019. Available from: https://www.researchgate.net/publication/228853276_Simple_Algorithms_for_Peak_Detection_in_Time-Series

21 H. M. Hashemian. State-of-the-Art Predictive Maintenance Techniques. IEEE Transactions on Instrumentation and Measurement, vol. 60, no. 1, pp. 226-236, Jan. 2011, doi: 10.1109/TIM.2010.2047662.

22 Giannoni F, Mancini M, Marinelli F. Anomaly detection models for IoT time series data. November 2018. DOI: https://doi.org/10.48550/arXiv.1812.00890

23 Yu Y, Zhu Y, Li S, Wang D. Time Series Outlier Detection based on Sliding Window Prediction. College of Computer and Information. Hohai University, 2014.

24 Wang D, Yu Y, Rinaldo A. Univariate mean change point detection: Penalization, CUSUM and optimality. Electronic Journal of Statistics. 2020. DOI: https://doi.org/10.1214/20-EJS1710

25 Chen J, Zhang J. Univariate change point detection methods: A review. Statistical Methods and Applications. 2020;29(3):567-600.

26 Li X, Wang J. Multivariate change point detection: A review. Statistical Methods and Applications. 2021 Apr 1;30(2):337-374.

27 Guoliang Lu, Yiqi Zhou, Changhou Lu, Xueyong Li. A novel framework of change-point detection for machine monitoring. Mechanical Systems and Signal Processing. 2017;83:533-548. ISSN 0888-3270. doi: 10.1016/j.ymssp.2016.06.030. Available from: https://www.sciencedirect.com/science/article/pii/S0888327016302126

28 Apostol, E.-S.; Truică, C.-O.; Pop, F.; Esposito, C. Change Point Enhanced Anomaly Detection for IoT Time Series Data. *Water* **2021**, *13*, 1633. https://doi.org/10.3390/w13121633

29 M. Fahim and A. Sillitti, "Anomaly Detection, Analysis and Prediction Techniques in IoT Environment: A Systematic Literature Review," in *IEEE Access*, vol. 7, pp. 81664-81681, 2019, doi: 10.1109/ACCESS.2019.2921912.

30 McKinney, Wes. "Data Structures for Statistical Computing in Python". Proceedings of the 9th Python in Science Conference. vol. 445, 2010 pp. 51-56. Available from: https://pandas.pydata.org/docs/user_guide/timeseries.html

31 The Scipy community. spicy.signal.find_peaks [Internet]. October 2022. Available from: https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.find_peaks.html

Appendices

List of Figures

List of Tables