Intelligent Predictive Maintenance for Fire Detection System with Improved Holt - Winters method

Samuel Aduroja,
Master's in Data Science
Department of Computer Science,
2023

Supervisor
Inês Dutra, Professor Auxiliar, Universidade do Porto


U.PORTO
FACULDADE DE CIÊNCIAS
UNIVERSIDADE DO PORTO

Acknowledgements

This work is a result of the project Safe Cities - Inovação para Construir Cidades Seguras, with the reference POCI-01-0247-FEDER-041435, co-funded by the European Regional Development Fund (ERDF), through the Operational Programme for Competitiveness and Internationalization (COMPETE 2020), under the PORTUGAL 2020 Partnership Agreement.

Abstract

Fire detection system maintenance is often critical and needs to be carried on time but not when there is a failure. A simple process of frequent checking and maintenance, also known as Preventive Maintenance (PvM), results in high labor cost, loss of productivity and does not help enterprise in saving cost. Long intervals between checking can be dangerous as it could lead to system unscheduled down occurrence and raise continuous false alarm. Getting too used to false alarms, due to system/sensor defect, can impact the decision and response of occupants in case of a fire emergency.

Predictive Maintenance (PdM) can provide the health status monitoring of the sensors/system for the present time, pre-empt, and forecast possible future false alarms, increase productivity and save cost. Accurate modelling of sensor data characteristics and promptly predicting their future values may improve the efficiency of the fire detection system. The goal of this work is to Improve PdM by using models based on Holt-Winters (HW). With suitable parameters in the model, we can predict future values of these sensor variables promptly. Knowing these values ahead helps with the PdM required for optimal operation of a fire alarm system. We will also develop a web app to show results.

Contents

List of Tables

List of Figures

Acronyms

LSTM Long Short-Term Memory

RNN Recurrent Neural Network

HW Holt-Winters

SARIMA Seasonal Autoregressive Integrated Moving Average

ARIMA Autoregressive Integrated Moving Average

ARMA Autoregressive Moving Average

ACF Autocorrelation Function

I Integrated

AR Autoregressive

MA Moving Average

ADF Augumented Dickey-Fuller

KPSS Kwiatkowski–Phillips–Schmidt–Shin

ACF Autocorrelation Function

PACF Partial Autocorrelation Function

FFT Fast Fourier Transforms

PdM Predictive Maintenance

PvM Preventive Maintenance

CM Conditional Monitoring

RM Remote Alert

FDS Fire Detection Systems

FACP Fire Alarm Control Panel

IoT Internet of Things

RUL Remaining Useful Life

TSP Time Series Prediction

TD target device

V+LSTM Vanilla Long Short-Term Memory

CNN+LSTM Convolutional Neural Networks-Long Short-Term Memory

DNN Deep Neural Network

SAFE Supervised Aggregative Feature Extraction

TCN Temporal Convolutional Neural Network

EPA Environmental Protection Agency

SES Simple Exponential Smoothing

NYC New York City

MAE Mean Absolute Error

MSE Mean Squared Error

RMSE Root Mean Squared Error

AIC Akaike Information Criterion

Chapter 1

Introduction

1.1 Context

In the current times, where digital and physical systems co-exists, system maintenance has become increasingly important to guarantee operational efficiency and enhanced productivity. The days when maintenance was merely a reactive function, where engineers and technicians rush to fix system failures after they broke down, are no longer sustainable. These days, intelligent Predictive Maintenance (PdM) has brought about a paradigm shift in how industries go about the safe upkeep of their machinery.

An essential part of this revolution in system maintenance are sensors, though small and often embedded in equipment or positioned strategically, they are powerful devices that continuously monitor various parameters of the environment and systems, such as temperature, humidity, heat, smoke, pressure, etc. These sensors provide real-time data, offering a window into the operational health of the system. However, the raw data from these sensors, though valuable, is not usually insightful immediately. This is where time series analysis comes in.

Time series analysis is a unique branch of statistics that deals with data points recorded or indexed at successive equally spaced points in time. In the context of PdM, it helps in identifying patterns, anomalies, change-points, and potential points of failure using the historical and current data from sensors. Tracking and visualizing the system performance can enable the early identification of potential problems.

But time series analysis alone might not be enough to predict when a machine will fail. To further enhance the predictive capabilities of this analysis, we need the power of time series modeling. Time series modeling, using algorithms and statistical models, can predict future values based on previously observed values. In PdM, this means that industries can foresee potential breakdown and conduct Preventive Maintenance (PvM), minimizing downtime, significantly saving costs and resources, and ensuring smooth operations.

The combination of sensor data, time series analysis and time series modeling is revolutionizing

the domain of maintenance. Leveraging these in a Fire Detection Systems (FDS), not only ensures the longevity and efficiency of the system, but also paves the way for significant cost savings, risks reduction and operational excellence.

1.2 Problem Description

Maintenance of a fire detection system is critical and must be carried out as and when due, but not when there is a failure. A simple process of frequent checking and maintenance, also known as PvM [1], results in high labor cost, loss of productivity and does not help enterprise save cost. Long intervals between checking can be dangerous as it could lead to system unscheduled down occurrence [2] and raise continuous false alarm. Also, getting too used to false alarms, due to sensor defect, can impact the decision and response of occupants in case of a fire emergency.

Bosch has gathered sensors data of several customers' FDS. PdM can provide the health status monitoring of the sensors for the present time, pre-empt, and forecast possible future false alarms, increase productivity and save cost. Accurate modelling of sensor data characteristics and promptly predicting their future values may improve the efficiency of the fire detection system.

1.3 Objectives

The goal of this work is to improve PdM by using models based on Holt-Winters (HW) using fire alarm sensors data of Bosch customers. Knowing these values ahead helps with the PdM required for optimal operation of a fire alarm system. We will also develop a web app to show results.

Autoregressive Integrated Moving Average (ARIMA) and Long Short-Term Memory (LSTM) have previously been used. However, these two are seen to be computationally expensive. Specifically, ARIMA (from **`autoarima - SARIMA(p,d,q)(P,D,Q,s)`**) takes time to train due to the number of parameters required in the model and the complexity in the iteration to find the best combination of **`p`** - autoregressive order, **`P`** - seasonal autoregressive order, **`d`**, differencing degree, **`D`** - seasonal differencing term, **`q`** - number of lagged forecast errors, **`Q`** - number of lagged forecast errors in the seasonal term and **`s`** - the seasonal period. On the hand, LSTM can only provide a point forecast, cannot easily learn long-term dependencies and given the amount of data available in this work, training might also be computationally expensive.

1.4 Contribution of this work

The main contributions of this dissertation and its proposed methodologies are considered to be the following:

- Methodologies tested and verified on anonymized real customer data
- Real scenerio of usually large amount of data from the sensors are used, unlike many papers which test their methodologies on simple, synthetic dataset
- Fast Fourier Transforms (FFT) was used to improve HW results.
- A webapp showing model performance is provided

1.5 Document organization

This dissertation is organised into five different chapters. Chapter 1, this chapter, introduced the framework establishing the necessity to develop this work, highlighting the goals of the work and its contribution to the field of FDS. The next chapter provides a background on the methodology used in this work and explains the general functioning of the FDS. Chapter 3 focuses on the State-of-the-Art while chapter 4 explains our approach, experimental set up and evaluation metrics. The results and analysis are discussed in chapter 5. The dissertation concludes with chapter 6 and also highlights future works.

Chapter 2

Background

This chapter provides a comprehensive background on the key concepts, definitions and terminologies relevant to time series methods and Fire Detection Systems (FDS). The primary goal of this chapter is to establish a comprehensive theoretical foundation for the subsequent analysis and discussions. The chapter also explains the basic mode of operation of the FDS. This foundation is important to gain insights into how the time series methods is applied and integrated into fire detection systems.

2.1 Definition and Terminology

2.1.1 Time Series

A time series is a sequence of observations taken at specific time interval [3]. For time series data observed at regular time intervals, the data points can be hourly, daily, weekly, monthly, quarterly, annually.

2.1.2 Stationarity

A time series is said to be stationary if its statistical properties, like mean, variance and autocovariance remain constant over time. For a stationary time series, its characteristics do not depend on the time at which the series is observed. In most statistical time series methods, stationarity is always assumed.

2.1.3 Non stationarity

A non stationary series has varying mean, variance and autocovariance.

2.1.4 Seasonality

Seasonality is the repetition in the characteristics of the series with time. It is usually with a constant and known frequency. The seasonal period, s, is the number of data points that makes a season. For example, if a time series data is recorded hourly and the season repeats daily, then the seasonal period is 24 since 24 hours makes a day i.e., there are 24 points or records in the season. If the season repeats weekly, the seasonal period is 24 x 7. Cyclic series is when the data shows rises and falls that are not of a constant frequency. This is different from a seasonal series.

2.1.5 Trend

Trend represents the upward or downward underlying pattern in a time series. In order to objectively and quantitatively determine the strength, whether strong or weak, and direction of a trend, whether upward(or positive), downward(or negative) or none, the Mann-Kendall Test [4] is one of the most popular parametric tests used in times series data on the assumption that this data has serial correlation. A trend can also be additive or multiplicative. When there is a constant increment or decrement over time, regardless of the current level of the series, the trend is additive. Multiplicative trend exists when the trend's influence grows or decrease relative to the level of the series. Series with multiplicative trends usually require a transformation, either Logarithmic or Box-cox transformation.

2.1.6 Noise

Noise is the random variation that is not explained by the trend or seasonality. It can also refer to the residuals or error which can arise from various sources, such as measurement errors, sampling variability, and other unpredictable factors that affect the observations.

2.1.7 Autocorrelation

This is the correlation of a time series with its own past and future values. It provides insights into the regularity and predictability of the data.

$$r(k) = \frac{\sum_{t=k+1}^{n}(x_t - \bar{x})(x_{t-k} - \bar{x})}{\sum_{t=1}^{n}(x_t - \bar{x})^2} \tag{2.1}$$

Where:

- $r(k)$: Autocorrelation at lag k.
- x_t : Value of the time series at time t.
- $\bar{x}$: Mean of the time series.

- n : Number of observations.
- k : Lag.

2.1.8 Autocovariance

Autocovariance measures how much a variable at one time is linearly related to the lag of itself. It is similar to autocorrelation but does not give standardized measure between -1 and 1 like autocorrelation.

$$\gamma(k) = \frac{1}{n} \sum_{t=k+1}^{n} (x_t - \bar{x})(x_{t-k} - \bar{x}) \tag{2.2}$$

Where:

- $\gamma(k)$: Autocovariance at lag k.
- x_t : Value of the time series at time t.
- $\bar{x}$: Mean of the time series.
- n : Number of observations.
- k : Lag.

The relationship between autocovariance and autocorrelation is given by:

$$r(k) = \frac{\gamma(k)}{\gamma(0)} \tag{2.3}$$

Where:

- $r(k)$: Autocorrelation at lag k.
- $\gamma(0)$: Variance of the time series.

2.1.9 Lag

The number of time steps shifted for the purpose of comparison. For example, lag 1 compares a series to itself one time step apart(see Table 2.1).

Day	Temperature (°C)	Temperature (°C) with Lag 1
1	30	NA
2	32	30
3	31	32
4	29	31

Table 2.1: Example of original time series with lag of 1 day.

2.1.10 Differencing

This is an approach to transform a non-stationary time series into a stationary one by computing differences between consecutive data points. Differencing can help to stabilise the mean of the time series.

2.1.10.1 Types of Differencing in Time Series

1. **First Order Differencing:** This involves subtracting the previous observation from the current observation.

$$\Delta y_t = y_t - y_{t-1} \tag{2.4}$$

   Where:

   - Δy_t is the difference at time t.
   - y_t is the observation at time t.
   - y_{t-1} is the observation at time $t-1$.

2. **Second Order Differencing:** If the time series is still non-stationary after first-order differencing, one might apply a second round of differencing.

$$\Delta^2 y_t = (y_t - y_{t-1}) - (y_{t-1} - y_{t-2}) \tag{2.5}$$

3. **Seasonal Differencing:** If a time series has seasonality, you might need to apply seasonal differencing. This involves subtracting the observation from the same season in the previous cycle.

$$\Delta y_t = y_t - y_{t-s} \tag{2.6}$$

   Where s is the number of time steps in one seasonal cycle.

2.1.10.2 It is impportant to note the following on Differencing

1. **Over-differencing:** Applying differencing more times than necessary can lead to the introduction of unnecessary noise into the series.

2. **Stationarity Issues:** If a time series is already stationary, applying differencing can make it non-stationary.

3. **Model Interpretation:** Over-differenced series may also complicate the interpretation of model parameters, as they no longer relate to the original data but to differences between data points.

4. **Evaluation Post Differencing:** It is crucial to always check the properties of the time series after differencing (e.g., through plots, Autocorrelation Function (ACF), Partial Autocorrelation Function (PACF)) to ensure that the desired stationarity is achieved.

Augumented Dickey-Fuller (ADF) and Kwiatkowski–Phillips–Schmidt–Shin (KPSS) tests can be used to test the stationarity of a time series, guiding decisions about the need and order of differencing.

2.2 Time Series Methods

2.2.1 Descriptive Methods

Trend Analysis Used to observe the long-term progression of a series.

Decomposition breaks down a time series into its constituent elements such as trend, seasonal and random components. Often decomposition is done to help improve understanding of the time series, but it can also be used to improve the forecast accuracy [3]. Seasonal Decomposition identifies and separates from a time series these components with respect to seasons.

2.2.2 Time Domain Methods

2.2.2.1 Autoregressive (AR)

This is a model that uses the relationship between an observation and its previous values. An AR model is commonly denoted as AR(p), where p is the order of the model, indicating how many previous points (lags) in the series are used. The general form of the AR model of order p, denoted as AR(p), is:

$$X_t = c + \phi_1 X_{t-1} + \phi_2 X_{t-2} + \cdots + \phi_p X_{t-p} + \varepsilon_t \tag{2.7}$$

Where:

- X_t: Value of the series at time t
- c: Constant term
- $\phi_1, \phi_2, \ldots, \phi_p$: Parameters of the model that need to be estimated

- ε_t: White noise error term at time t

For an AR(1) model, the equation simplifies to:

$$X_t = c + \phi_1 X_{t-1} + \varepsilon_t \tag{2.8}$$

Here, the value at time t is a function of the value at $t-1$ (one lag), plus a constant, and a white noise error term. For the AR model to be stationary, the values of the parameters ϕ have constraints. For example, in an AR(1) model, $|\phi_1| < 1$ for the process to be stationary.

2.2.2.2 Moving Average (MA)

This model uses the relationship between an observation and residual errors from previous forecasts. It uses past white noise error terms to forecast future values. Specifically, the MA model assumes that the output variable is a linear combination of the current and various past white noise error terms.

The general form of the moving average model of order q, denoted as MA(q), is:

$$X_t = \mu + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \cdots + \theta_q \varepsilon_{t-q} \tag{2.9}$$

Where:

- X_t is the value of the series at time t.
- μ is a constant representing the mean of the series.
- ε_t is the white noise error term at time t.
- $\theta_1, \theta_2, \ldots, \theta_q$ are the parameters of the model that need to be estimated.

For an MA(1) model, the equation simplifies to:

$$X_t = \mu + \varepsilon_t + \theta_1 \varepsilon_{t-1} \tag{2.10}$$

2.2.2.3 Autoregressive Moving Average (ARMA)

This method combines AR and MA methods. It is particularly useful for modeling time series data that do not show a trend or seasonal effects.

The general form of the ARMA model is usually denoted as ARMA(p, q), where:

- p is the order of the AR part.
- q is the order of the MA part.

The general formula for the ARMA(p, q) model is:

$$X_t = c + \phi_1 X_{t-1} + \phi_2 X_{t-2} + \cdots + \phi_p X_{t-p} + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \cdots + \theta_q \varepsilon_{t-q} \tag{2.11}$$

Where:

- X_t: Value of the time series at time t.
- c: Constant.
- ϕ_i: Coefficients of the AR part.
- θ_j: Coefficients of the MA part.
- ε_t: White noise error term at time t.

2.2.2.4 Autoregressive Integrated Moving Average (ARIMA)

This is a classical time series algorithm that combines AR and MA methods while also incorporating the Integrated (I) component, by differencing the data to make the series stationary. When the seasonality present in a series is considered in the model, it is called Seasonal Autoregressive Integrated Moving Average (SARIMA). ARIMA calculates the parameters (p, d, q) if the time series is non seasonal or (p, d, q)(P, D, Q, s) for seasonal time series given the seasonal period, s. Where p is the autoregressive term, P is the seasonal autoregressive order, d, differencing degree, D - seasonal differencing term, q is the moving average term, and Q is the seasonal moving average term.

The ARIMA model can be represented as:

$$(1 - \sum_{i=1}^{p} \phi_i L^i)(1 - L)^d X_t = \mu + (1 + \sum_{i=1}^{q} \theta_i L^i)\varepsilon_t \tag{2.12}$$

Where:

- X_t: The time series at time t.
- L: The lag operator.
- ϕ_i: Parameters of the AR part of the model.
- θ_i: Parameters of the MA part of the model.
- ε_t: White noise error term.
- μ: Mean of the time series.

2.2.3 Exponential Smoothing

This method assigns exponentially decreasing weights to past observations, giving higher values or more emphasis to more recent observations.

2.2.3.1 Simple Exponential Smoothing (SES)

This is for forecasting data with no clear trend or seasonality. The forecast for the next period is given by:

$$\hat{y}_{t+1} = \alpha y_t + (1 - \alpha)\hat{y}_t \tag{2.13}$$

Where:

- $\hat{y}_{t+1}$ is the forecast for the next period.
- y_t is the actual value at time t.
- $\hat{y}_t$ is the forecasted value for time t.
- α is the smoothing parameter, $0 \leq \alpha \leq 1$.

2.2.3.2 Holt's Linear Exponential Smoothing

Also called double exponential smoothing. This method extends SES to account for trends in the data. The forecast and trend equations are given by:

$$\hat{y}_{t+1} = l_t + b_t \tag{2.14}$$
$$l_t = \alpha y_t + (1 - \alpha)(l_{t-1} + b_{t-1}) \tag{2.15}$$
$$b_t = \beta(l_t - l_{t-1}) + (1 - \beta)b_{t-1} \tag{2.16}$$

Where:

- l_t is the level component at time t.
- b_t is the trend component at time t.
- β is the trend smoothing parameter, $0 \leq \beta \leq 1$.

2.2.3.3 Holt-Winters' Exponential Smoothing

This method incorporates both trends and seasonality. The forecast, level, trend, and seasonal equations are:

$$\hat{y}_{t+m} = l_t + mb_t + s_{t-m+1} \tag{2.17}$$
$$l_t = \alpha(y_t - s_{t-m}) + (1-\alpha)(l_{t-1} + b_{t-1}) \tag{2.18}$$
$$b_t = \beta(l_t - l_{t-1}) + (1-\beta)b_{t-1} \tag{2.19}$$
$$s_t = \gamma(y_t - l_t) + (1-\gamma)s_{t-m} \tag{2.20}$$

Where:

- m is the seasonal period.
- s_t is the seasonal factor at time t.
- γ is the seasonal smoothing parameter, $0 \leq \gamma \leq 1$.

2.2.4 Frequency Domain Methods

2.2.4.1 Spectral Analysis

This analyzes the series in the frequency domain to identify cyclical behaviours. Spectral analysis is closely related to the Fourier transform, which translates a function from the time domain to the frequency domain. The resulting spectrum reveals the time series' strength or power across different frequencies.

For a time series $x(t)$, its Fourier transform is given by:

$$X(f) = \int_{-\infty}^{\infty} x(t) e^{-j2\pi ft} dt \tag{2.21}$$

Where:

- $X(f)$ is the spectrum of the time series in the frequency domain.
- f is the frequency.
- e is the base of the natural logarithm.
- j is the imaginary unit.

The power spectral density (often just called the spectrum) of the time series is then:

$$S(f) = |X(f)|^2 \tag{2.22}$$

This provides a representation of how the variance (or power) of the time series is distributed across different frequencies.

2.2.4.2 Wavelet Analysis

This method decomposes a series into different frequency bands, capturing both high-frequency and low-frequency components. This technique is especially significant for non-stationary time series, where properties evolve over time. The method does the following:

1. **Local Analysis**: Unlike Fourier transform, which gives a global view of frequency components, wavelet analysis provides a local view of the data in both time and frequency. This allows for capturing transient or abrupt changes in a time series.
2. **Scaling**: Wavelets involve two main functions - the mother wavelet and the scaling function. By dilating or contracting, and translating the mother wavelet, the time series can be analyzed at various frequencies or scales.
3. **Decomposition**: Using wavelets, a time series can be decomposed into a set of wavelet coefficients, representing the data at different scales. The series can then be reconstructed using an inverse wavelet transform.

2.2.5 State Space Methods & Machine Learning

2.2.5.1 Kalman Filtering

This is a recursive method to estimate the future state of a time series. It is particularly effective in situations characterized by noisy and uncertain observations.

The Kalman filter works in two steps:

1. **Prediction**: It makes a prediction of the current state variables, along with their uncertainties.
$$\hat{x}_{k|k-1} = A\hat{x}_{k-1|k-1} + Bu_k \tag{2.23}$$
Where:
   - $\hat{x}_{k|k-1}$ is the predicted state estimate.
   - A is the state transition model.
   - $\hat{x}_{k-1|k-1}$ is the previous state estimate.
   - Bu_k represents the control input.
2. **Update**: Once the next measurement is observed, these predictions are updated using a weighted average, with greater weight given to estimates with higher certainty.
$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k(y_k - H\hat{x}_{k|k-1}) \tag{2.24}$$
Where:

- $\hat{x}_{k|k}$ is the updated state estimate.
- K_k is the Kalman gain.
- y_k is the actual measurement.
- H is the observation model.

2.2.5.2 Long Short-Term Memory (LSTM)

This is a type of recurrent neural network especially suited for sequence prediction problems [5]. Unlike standard feedforward neural networks, LSTMs have feedback connections, allowing them to process sequences of data, making them particularly well-suited for time series analysis and other sequential tasks.

The following are keys characteristics of LSTM

1. **Memory Cells:** LSTMs contain memory cells that can maintain information in memory for long periods of time. This is a distinguishing feature of LSTMs compared to traditional Recurrent Neural Network (RNN)s.
2. **Gating Mechanisms:** LSTMs use three main gates:
   - *Input Gate* - Decides what information to store in the memory cell.
   - *Forget Gate* - Decides what information to discard from the memory cell.
   - *Output Gate* - Decides what information from the memory cell should be output based on the input and the current state of the memory cell.
3. **Avoiding Long-term Dependency Problem:** Traditional RNNs face challenges when trying to remember past information for long periods, commonly referred to as the vanishing gradient problem. LSTMs are designed to combat this issue, making them more effective at capturing long-term dependencies in the data.

2.2.6 Detecting Seasonal period, s

The seasonal period is a very important parameter that must be known before training Holt-Winters (HW) model. The seasonal period value may not necessarily be constant throughout the data set, hence there can be more than one seasonal period in a series. Two popular approaches to automatically capture the seasonal period of a time series data are ACF and Fast Fourier Transforms (FFT) [6]. The ACF automatically captures significant seasonality based on a threshold while the FFT captures seasonality by first transforming the signal from time to frequency domain and from the Power Spectral Density, significant seasonalities are outputted. The FFT approach usually outperforms the ACF approach. We also confirmed this using a wind turbine data[1]. The data is the active power of a turbine collected every 10minutes from January

[1]Link to the data: here

2018 to March 2020. Figure 2.1 shows the raw data, while figure 2.2 shows the hourly aggregated data. Figure 2.3 is the daily aggregated data of the turbine data. Visually, we can observe the presence of two seasons within the whole period from 2018-01-01 to 2020-03-16.

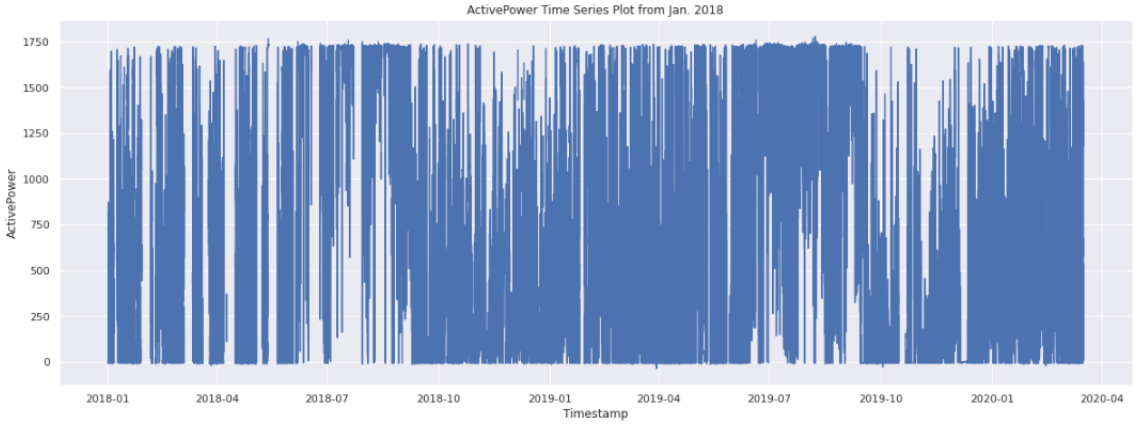


Figure 2.1: Raw turbine data.

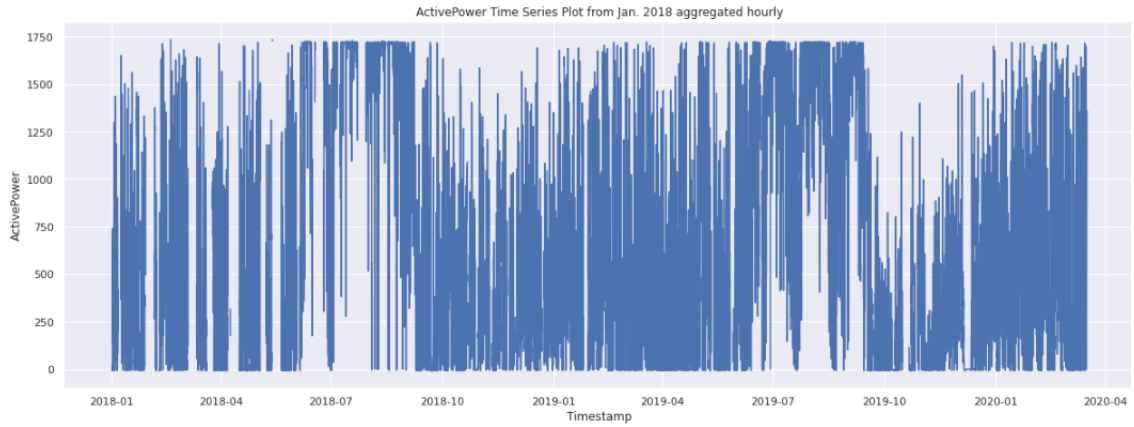


Figure 2.2: Hourly aggregated turbine data.

The ACF and FFT were tested on the daily aggreggated data after filling missing values with moving average method. These tests were implemented with the ACF and FFT seasonality detector module with kats [7]. ACF seasonal period detected two periods: 24 and 170 days. FFT seasonal period detected one periods: 403 days. This value is exactly half the total length of the series, 806 days. This justifies our earlier assertion, by visual inspection, that there exist two seasons. In essence, it means the season repeats after every 403 days.

2.3 Fire Detection System

FDS are designed to alert users of an emergency so that they can take action for their safety. They are found in offices, factories and public buildings. If the alarm is triggered, it will operate to warn people in the building that there may be a fire and to evacuate. Modern day FDS has a Fire Alarm Control Panel (FACP) which is usually the central hub where all the monitoring

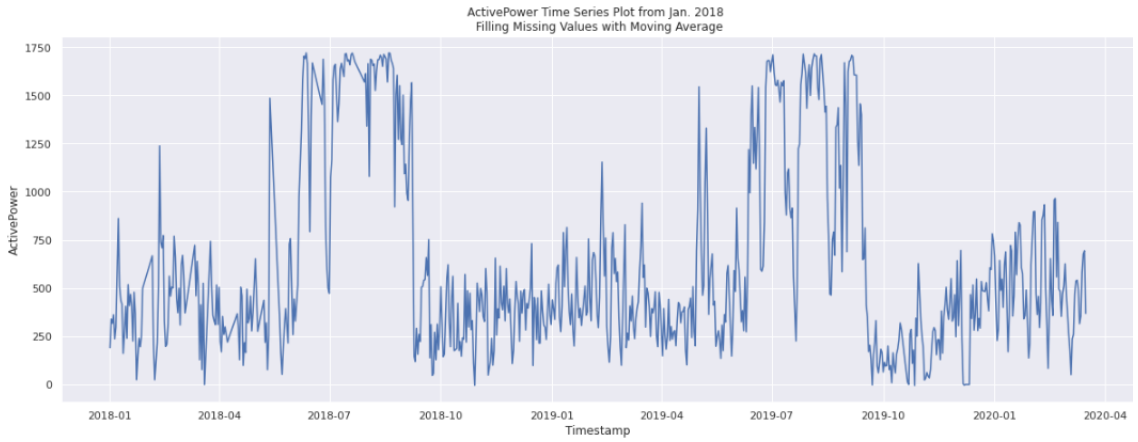


Figure 2.3: Daily aggregated turbine data.

sensors are wired to. The panel provides a status indication to the users and can also be used to simulate an alarm for use in routine fire and evacuation drills, so that everyone knows what action to take in the event of a real fire.

While the FDS work, two forms of data are recorded and stored in the cloud for remote monitoring at the back-end. The Conditional Monitoring (CM) data is mainly numerical and are usually the values of the variable a sensor measures at each specific interval - usually defined during the design or installation of the system. The Remote Alert (RM) data provides the state of the different devices in the system for each customer. For example, in the context of this work, data from Bosch Building Technologies were used through the project titled: Safe Cities - Innovation to Build Safe. The objective of the project is for researchers to collaborate in order to develop several products and services that can be integrated into Bosch systems for the purpose of improving the safety of people and systems they interact with in these cities.

2.4 Conclusion

This chapter has introduced essential definitions and methods related to time series analysis. It has also discussed the basic operation of the FDS. Understanding these foundational concepts is imperative for subsequent chapters, where we delve into the applications and use of the different time series methodologies.

Chapter 3

Time series analysis and modeling of fire sensors

This chapter examines the latest advancements in Predictive Maintenance (PdM), specifically its role in Fire Detection Systems (FDS). Our goal is to highlight the current methodologies and trends in PdM implementation. We start by discussing various sectors where PdM has been adopted. Then, we scrutinize past studies that utilized statistical and machine learning approaches for this objective. We conclude by evaluating the applicability of these techniques to Bosch data.

3.1 Fields of Application of Predictive Maintenance

PdM has gained relevance in many industries due to its main aim of reducing the cost of maintenance and avoiding unexpected downtime. In [2], PdM was used to solve the problem of inaccurate prediction of the Remaining Useful Life (RUL) of a target device (TD) by exponential model. In the work, Time Series Prediction (TSP) algorithm proves more efficient compared to exponential model. The previously used exponential model was built using information criterion to adapt the trend in solving TD fault prediction. [8] also worked on the estimation of RUL based on time series analysis. The experiments were carried out with the publicly available NASA Turbofan Engine Corruption Simulation (C-MAPSS) dataset, which consists of 21 sensor data of the aircraft engine. In the results obtained, traditional ML models - Random Forest and Support Vector Regression outperform Long Short-Term Memory (LSTM), Gated Recurrent Unit and Convolutional Neural Networks-Long Short-Term Memory (CNN+LSTM) algorithms. [9] implemented PdM through anomaly detection by classification. This was done by using two techniques: Deep Neural Network (DNN) with Logistic Regression and and Recurrent LSTM neural network with Autoencoder. The DNN was used to classify various types of failures while the LSTM was used to classify various flaws. In the maritime industry, [10] used PdM to optimise the maintenance schedule for vessel so that they can be available for more time while at the same time preventing down times and minimizing cost. In the work, the focus was to utilize machine

learning and statistical approaches like LSTM, OneClassSVM, Extreme Gradient Boost(XGB) and Permutation Entropy Check for early detection of abnormal operation related to Crosshead Bearing defect. This approach offers extended vessels lifespan and reduced maintenance cost. [11] on the other hand used a functional learning methodology, namely Supervised Aggregative Feature Extraction (SAFE) to exploit time series data for PdM problems and ultimately to solve semiconductor manufacturing maintenance problem. Using only case temperature data, [12] first performed score computation based on operating temperatures in different modes, then carried out feature calculation to generate and identify new features. This was followed by model learning using the identified relevant features. This made PdM applicable in supermarket refrigeration systems for early detection of issues emerging in refrigeration and cold storage systems. [13], with time series intervention analysis, studied the system failure pattern. This made it possible to apply PdM for Management of Multiunit Systems.

In the power sector, PdM has been widely applied at large, medium and small scale. At large scale, in power generation, PdM has been useful to keep the power system frequency constant[1]. This requires that the power generated must equal power demanded and/or consumed at any point in time without the generators necessarily running perpetually. PdM is carried out on some generators that requires it while the rest remain operational with power supply still available. This is done by forecasting the demand [14] thereby lowering operating costs, resulting in significant savings for power companies [15]. This forecast is usually done using both statistical model [16] and machine learning models [9]. In medium and small scale, especially when it comes to residential electricity consumption, the concept of PdM has been applied for energy management and monitoring [17–21].

3.2 Algorithms for Predictive Maintenance in the Fire Industry

In the fire detection industry, the need for early and accurate fire detection is a growing research area [22, 23]. Usually, in the traditional FDS, the interconnected sensors sense the environmental variable(s), temperature for example, generates an alarm to inform occupants of the danger if the temperature exceeds a threshold value, triggers water sprayer or a fire-ceasing foam-producing device to control the fire before firefighters arrive. This is not effective for early and accurate detection. Subsequently, PdM would be inefficient in the traditional FDS.

Due to their feasibility of being integrated with microcontrollers, sensors, using Internet of Things (IoT) systems, can combine with other systems to generate different applications [24]. Integrating sensors into the FDS through IoT system provides an opportunity to describe a phenomenon or variable and its characteristics by using specific sensors designed to collect data for that variable [25]. This robust configuration can allow the FDS to monitor critical variables like temperature and chemical; and internal devices variables like optical, working hours, voltage. In this way, instead of detecting a failure, fault or emergency situation only after a threshold

[1] An imbalance in frequency of the power system can lead to power disruption - unplanned downtime can be very costly

value is reached, it predicts what the threshold value would be in order to plan ahead or detect a possible future defect in the systems or sensors collecting the data. This PdM approach can save cost and prevent loss of productivity provided that the sensors themselves are precisely placed at the desirable sites [26]. Although, these sensors can fail sometimes when their main task of collecting data is impaired by some unpredictable environment conditions. This situation can affect the communication channel and sensors' functionalities, which could in turn cause the loss of information. This problem was solved by [27] by using previous Holt-Winters (HW) and Vanilla Long Short-Term Memory (V+LSTM) models trained with complete data to fill present and future IoT data of temperature and CO_2. This approach provides an effective method for missing value imputation. Also, HW use in [27] provided least error and was implemented fastest. Based on the relevant previous works reviewed as shown in Table 3.1, HW has proven to be fast and computationally efficient.

3.3 Using Holt-Winters for Predictive Maintenance

The problem in this work relates using HW for time series data from FDS for PdM use case. In this dissertation, we have reviewed the literature for PdM where HW models were used. However, there is not so much work on HW's use case for PdM in FDS, as far as the author is aware. We aim to carry out PdM on critical sensors by forecasting their values. Knowing critical sensors future values ahead can help mitigate false alarms common to faulty sensors or components. Occupants getting too used to false alarms, especially those not related fire drill routine, may not respond appropriately and fast enough in the event of a real fire. Also, early identification of faulty sensors and components improves the effectiveness of the FDS. Also, some components of the FDS develop faults that might affect the proper functioning of the whole system. It becomes even dangerous if these defects in the FDS lead to situations where false alarms are given as false alarm may reduce occupants response speed when there is real fire.

[28] has previously worked on the data set using state-of-the-art python libraries to handle the large data for just one system, one customer while comparing Autoregressive Integrated Moving Average (ARIMA), LSTM and HW. In our work, the focus is on the data set of critical sensors not just on the whole system. This is because there are several sensors in a system. Generalizing, by training a single model for a customer's system may be produce a complex model that fails to identify anomaly in some sensors of the same customer's system. The computational speed of such model is also a concern. In the FDS, making the right calls and taking appropriate action on time can save more lifes and properties. In this dissertation, we are going to consider, one at a time, a **`variable`**(say optical, temperature or working hour) of a **`sensor type`** with a specific **`serial number`** given its **`logical address`** and **`physical address`** for a chosen **`system`** of a **`customer`**.

As it was previously noted in subsection 2.2.3.3 of chapter 2, HW requires seasonality and trend. However, accurately detecting the trend and seasonality in a trend can be a daunting task in the presence of local outliers. [29] proposed an online unsupervised deep learning based

algorithm for outlier detection utilizing Temporal Convolutional Neural Network (TCN). This method depends on thresholds individually defined for each data point to better address the presence of seasonality and trend. Our approach, as much as possible, seeks to automate the PdM task for the FDS using HW. [29] approach is not useful in our case as it depends manually defining thresholds. [30], compared used Analysis of Variance (ANOVA) and Autobox ACF to detect seasonality. It was concluded that the accurate inclusion (or exclusion) of seasonal element in a forecast is dependent on the careful analysis of the time series in question to determine the seasonality. Visual inspection has also been used previously for seasonality detection. However, detecting seasonality and trend by visual inspection is not considered a reliable method to determine the time series pattern in case of a huge dataset [31]. [32–34] used Autocorrelation Function (ACF) to detect seasonality and trend and to check whether the time series is stationary or not. [35] used Fast Fourier Transforms (FFT) technique in Box-Jenkins approach to detect seasonality. We will be incorporating this method in HW for PdM in FDS.

3.4 Can we use these methods with our data?

Across all reviewed articles, to the best of our knowledge, there are only a few studies on artificial intelligence or statistical methods for PdM in FDS. Also, the author notices that previous work on HW for PdM use cases use relatively small sample sizes from one month to one year. Additionally, [28], the closest and most recent to our work, only trained one model for a system of a customer. Our approach is promising as it is using HW, which we have confirmed from literature, to be fast and computationally efficient. Our approach considers critical sensors of the FDS tested to have seasonality and trend. We used a novel seasonality detection method FFT previously tested on Box-Jenkins method. Finally, our method has been tested on data set of real users of the FDS.

Table 3.1: Relevant Paper Reviewed

S/N	Year	Method	Result
1 [8]	2022	RF, SVR, MLP , ADABOOST, GBR, LSTM, GRU, CNN, CNN-LSTM, CNN-GRU and TCN	RF ans SVR superior than Deep Learning and remaining Machine Learning models
2 [2]	2019	TSP algorithm for RUL of a TD	TSP algorithm is superior to exponential model
3 [9]	2022	DNN with Logistic Regression to classifiy failures and LSTM to classify various flaws	It is possible to generalize failues and flaws
4 [10]	2019	SVM, Gradient Boosting and different LSTM combination to classify anomally in Marine engine	Models with a combination of approaches work better
5 [11]	2016	Exploits time series data for PdM with SAFE	SAFE outperforms classical feature extraction methods
6 [12]	2018	Score computation, feature calculation and classification TD	The model was limited by the size of data available.
7 [36]	2022	Exponential Smoothing methods	HW has the best performance of all three models
8 [14]	2020	Prophet and HW for Long term forecast	Prophet gave better results compared to HW
9 [15]	2022	Statistical and Machine learning methods	AI and hybrid models are applied more recently
10 [16]	2019	SARIMA, using FFT for seasonality detection	SARIMA model can forecast the hourly electrical load data
11 [17]	2020	LSTM, HW, ELM, HW-ELM hybrid	HW-ELM has the best result
12 [18]	2018	ARIMA and LSTM	ARIMA and LSTM models identify anomalies in the energy delivery system
13 [19]	2022	FFBPNN and HW, with FFBPNN integrated with HW through different weights	Obtains optimum weights by using a local search machine learning technique.
14 [21]	2020	Feature selection with ensemble of RF, AdaBoost, GBDT, XGBoost, SVR and ANN	Relevant features identified and used for prediction

Chapter 4

Holt-Winters fire sensors forecasting

This chapter delves into the application of the Holt-Winters (HW) method for Predictive Maintenance (PdM) in Fire Detection Systems (FDS). We start by discussing the data and pertinent variables essential for the selected HW forecasting model. Following this, we present the specifics of the experiment and the methodologies for model assessment. The chapter wraps up with insights derived from our methodology.

4.1 Data Source

The main dataset used in this dissertation is sourced from Bosch Building Technologies, a prominent figure in the FDS sector. They offer cutting-edge products and solutions to foster urban safety. The dataset from Bosch comprises 52 anonymized customer entries gathered between 2017 and 2020, presented in two distinct types: Conditional Monitoring (CM) and Remote Alert (RM). The former predominantly contains numerical data, capturing details from various sensors regarding the physical state of the building, the condition of system components, and measurements like temperature, heat, chemicals, battery level, smoke concentration, and so forth. In contrast, the RM data details the system's states and any changes therein, indicating events such as fires, tampering, alarm deactivation, pollution, system issues, and more. For this research, only the CM data will be employed. The data structure is as shown in figure 4.1 below.

We explore a second dataset, NYC311[1] dataset[2](from 2010 till 20/10/2021 15h00) to test our methodology. Complete and up-to-date NYC311 dataset is publicly available here. NYC311 is a platform that provides access to the general public resident in New York City (NYC) to report all non emergency. The requests made to this platform goes directly to the responsible agency for action and resolution. This helps the agencies improve service delivery by focusing on their core missions and managing their workload efficiently. NYC311's platform provide the public with quick and easy access to NYC (non-emergency) government services and information while

[1]About NYC311:here
[2]Link to the NYC311 data: here

offering the best customer service. NYC311 also provides insight to improve NYC governance through accurate, consistent measurement and analysis of service delivery. The data used in this report consists of complaints that relates to Environmental Protection Agency only. A summary of columns in the data is as shown in Table 4.1.

Our main aim for using this dataset is to test our methodology. We examined the use of time series models to forecast the number of complaints. When future values are known, we can make plans, in anticipation, especially for days when the number would be overwhelming for customer support team. We can make arrangement for adhoc staff to improvise for such days. This approach can help to save cost and improve planning. This was achieved by first converting the dataset to a time series object.

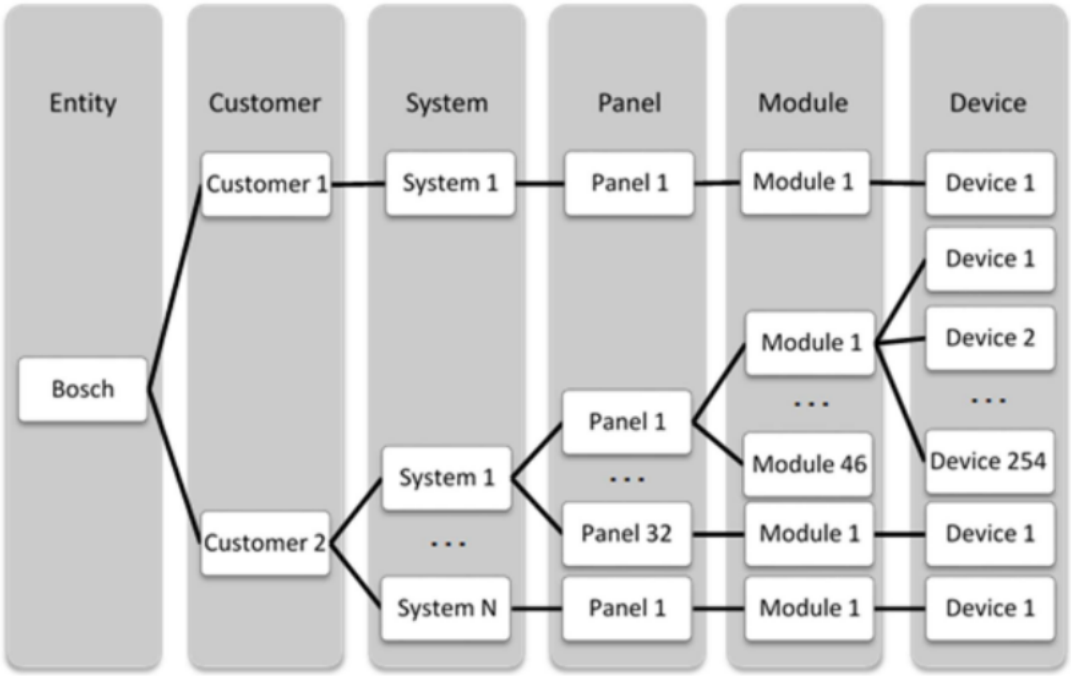


Figure 4.1: Bosch Fire Detection System Structure

4.2 Data Preprocessing

4.2.0.1 Bosch data preprocessing

Data from Bosch is the main dataset for this work and this data was processed using python language. When a customer's sensor—identified by its **`system, physical address, logical address, and serial number`** — detects a variable, for example **`opt1`** variable, its values are captured at specific time intervals throughout the observation period (with a starting point of 2017-01-01 00:00:00 for the majority of the data). This collected data forms the time series dataset utilized for model training. To illustrate, the details for **`customer 14`**, encompassing **`system, physical and logical addresses, serial number, variable,`**

and sensor type are provided below:

- Customer: c02b23084ffcc2666e684875d8abaa13772c930a10bad869dca1a8f352cf08f9
- System: fb1497cd99495140a26ed9aeba7fbd03cc43e724080ed80b14ad8167588a8824
- Physical Address: 13.1.5.0
- Logical Address 1.1.POINT.28.2
- Serial number: 5814293
- Variable: opt1
- Sensor type: FAP - O420

The first five rows of the data of the customer queried with the above credentials is shown in Figure 4.2.

timestamp	physicalAddress	type	wrongAnswerCnt	noAnswerCnt	opt1	workinghourCnt	poll1	serial	opt2	temp1	batV1	batDisC1	batR1	logicalAddress	System	Customer
2017-07-25 09:43:30	13.1.5.0	FAP-O420	NaN	NaN	NaN	14220.0	0.0	5814293	NaN	NaN	NaN	NaN	NaN	1.1.POINT.28.2	fb1497cd99495140a26ed9aeba7fbd03cc43e724080ed80b14ad8167588a8824	c02b23084ffcc2666e684875d8abaa13772c930a10bad869dca1a8f352cf08f9
2017-07-25 09:43:36	13.1.5.0	FAP-O420	0.0	0.0	62.0	NaN	NaN	5814293	NaN	NaN	NaN	NaN	NaN	1.1.POINT.28.2	fb1497cd99495140a26ed9aeba7fbd03cc43e724080ed80b14ad8167588a8824	c02b23084ffcc2666e684875d8abaa13772c930a10bad869dca1a8f352cf08f9
2017-07-25 09:58:39	13.1.5.0	FAP-O420	0.0	0.0	61.0	NaN	NaN	5814293	NaN	NaN	NaN	NaN	NaN	1.1.POINT.28.2	fb1497cd99495140a26ed9aeba7fbd03cc43e724080ed80b14ad8167588a8824	c02b23084ffcc2666e684875d8abaa13772c930a10bad869dca1a8f352cf08f9
2017-07-25 10:16:25	13.1.5.0	FAP-O420	0.0	0.0	61.0	NaN	NaN	5814293	NaN	NaN	NaN	NaN	NaN	1.1.POINT.28.2	fb1497cd99495140a26ed9aeba7fbd03cc43e724080ed80b14ad8167588a8824	c02b23084ffcc2666e684875d8abaa13772c930a10bad869dca1a8f352cf08f9
2017-07-25 10:32:29	13.1.5.0	FAP-O420	0.0	0.0	60.0	NaN	NaN	5814293	NaN	NaN	NaN	NaN	NaN	1.1.POINT.28.2	fb1497cd99495140a26ed9aeba7fbd03cc43e724080ed80b14ad8167588a8824	c02b23084ffcc2666e684875d8abaa13772c930a10bad869dca1a8f352cf08f9

Figure 4.2: First five rows of customer 14 dataframe.

As there are many different variables like poll1, temp1, batv1, opt1 etc, the variable whose values are to be predicted needs to be selected. Below is the first set of rows of the raw and hourly aggregated dataset of customer 14, with variable opt1 and other details provided above.

Raw Opt1 Data

timestamp	opt1
2017-07-25 09:43:30	NaN
2017-07-25 09:43:36	62.0
2017-07-25 09:58:39	61.0
2017-07-25 10:16:25	61.0
2017-07-25 10:32:29	60.0
2017-07-25 10:48:33	59.0
2017-07-25 11:04:38	61.0
2017-07-25 11:19:58	61.0

	opt1
count	95912.000000
mean	59.318167
std	2.430643
min	43.000000
25%	58.000000
50%	59.000000
75%	61.000000
max	73.000000

Aggregated Opt1 Data

timestamp	opt1
2017-07-25 09:00:00	61.500000
2017-07-25 10:00:00	60.000000
2017-07-25 11:00:00	60.500000
2017-07-25 12:00:00	60.500000
2017-07-25 13:00:00	61.333333
2017-07-25 14:00:00	60.250000
2017-07-25 15:00:00	59.750000
2017-07-25 16:00:00	61.250000

	opt1
count	24651.000000
mean	59.320507
std	1.827923
min	53.000000
25%	58.000000
50%	59.250000
75%	60.666667
max	66.333333

Figure 4.3: Opt1 time series data sample.

The raw and aggregated data of the sensors have missing values which were filled up with mean and moving average imputation methods. It is this hourly aggregated data that is then fed into the model for training.

4.2.0.2 NYC311 data preprocessing

The NYC311 data consists of 1,991,218 complaints registered to the Department of Environmental Protection. The data, processed in R language, consists of several complaints registered on NYC311 that are related to Environmental Protection Agency (EPA) from all the five Boroughs: Brooklyn, Bronx, Manhattan, Queens and Staten Island. This dataset initially has 38 columns containing 26 discrete values and 4 with continuous values, which describe environmentally related complaints, a brief description, time data associated with bureaucratic procedures and location features.

Preprocessing commenced by converting date fields to date format, all other string variables to factors, and all remaining fields were considered numeric. Empty or irrelevant values were converted to `NA`, and records with impossible dates removed. After preproceesing, the dataset was reduced to 21 fields, removing fields with no information, no variation, or redundant information. Levels with typos were corrected and merged and empty levels were removed from factors. After this, all records with null values were removed, leaving 1,518,239 records were left to be analysed. Subsets of attributes - Created.Date and Closed.Date - were taken to form a time series object. These time series objects are dataset of opened and closed service requests.

Figure 4.4 shows the volume of calls from 2010 to 2021. The figure shows a consistent increase in complaints from 2014 till 2018. Although there was a slight decrease in 2017, since complaints received climaxed in 2018 it has been followed by a consistent decline yearly from 2019. Many factors could have led to this decline. For example; lack of trust in the ability of NYC311 to guarantee efficient service, poor service delivery, total resolution of a prominent and recurring complaint type in 2018 and so on. The time plot of Figure 4.5 shows the number of complaints received daily across all borough.

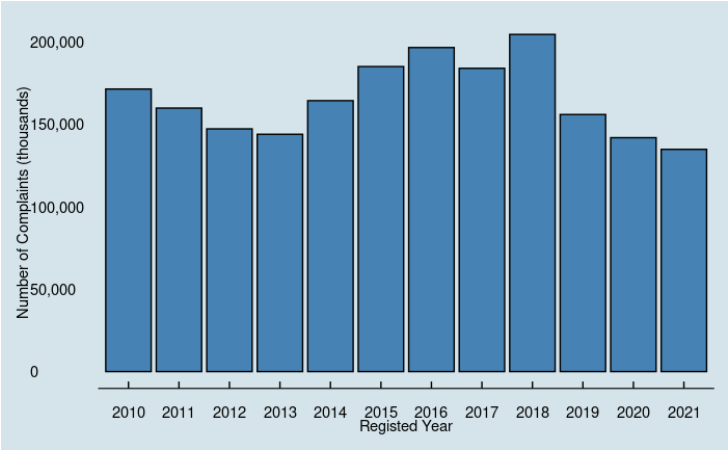


Figure 4.4: Yearly Number of Complaint

Figure 4.6 shows the trend and cycles that repeat regularly over time in the data. Since we want to forecast monthly complaints, the data was aggregated monthly as in Figure 4.7. Here, repeated patterns present in the data is now obvious as against when it was not aggregated. Also, intervention[3] in the series around the third quarter of 2019 is very much clearer in the aggregated figure.

[3]Intervention is a change to a procedure, or law, or policy, etc. that is intended to change the values of the series

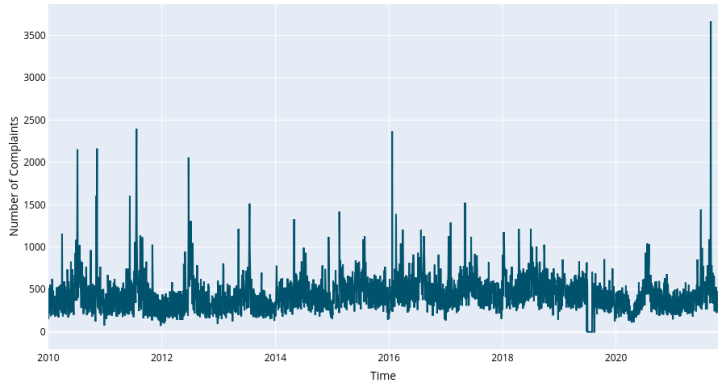


Figure 4.5: Time Plot of NYC311 Number of Complaints

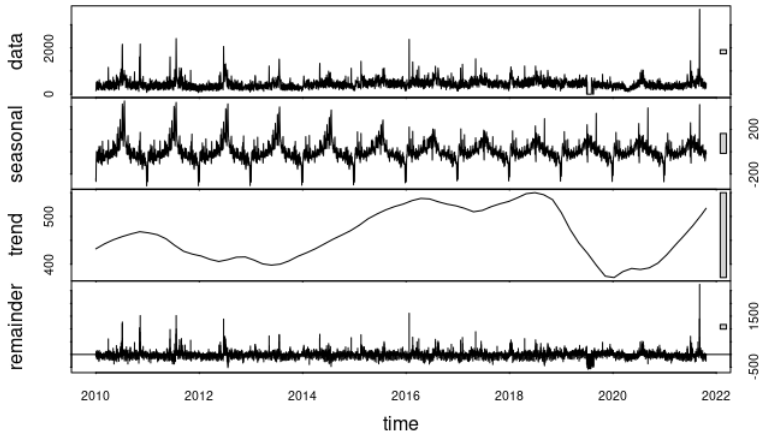


Figure 4.6: Seasonal, Trend and Residual Plot of Time Series

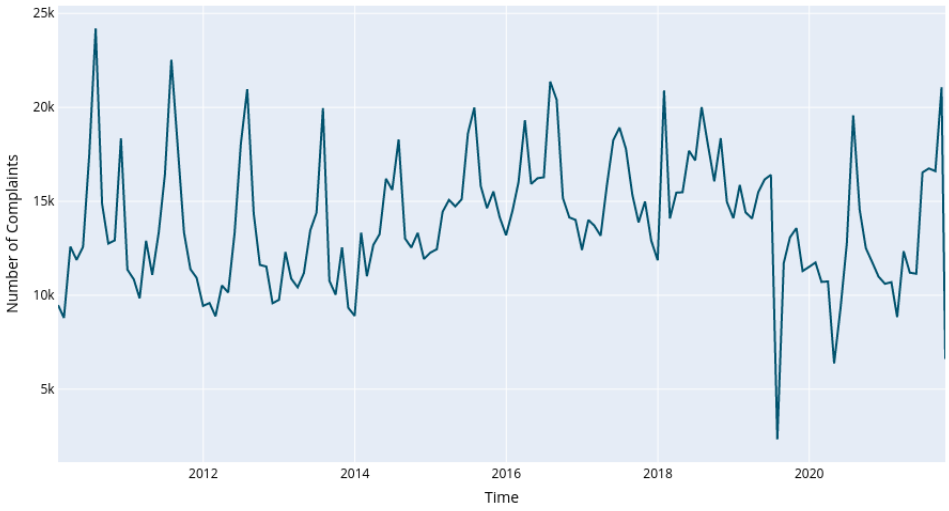


Figure 4.7: Time Plot of monthly aggregated NYC311 Number of Complaints

4.3 Baseline

The work started by developing and training Long Short-Term Memory (LSTM) to predict sensors values for the next three months. The training process was automated such that upon the selection of the following variables, we have a model trained and the result displayed as shown in Figure 4.8 below:

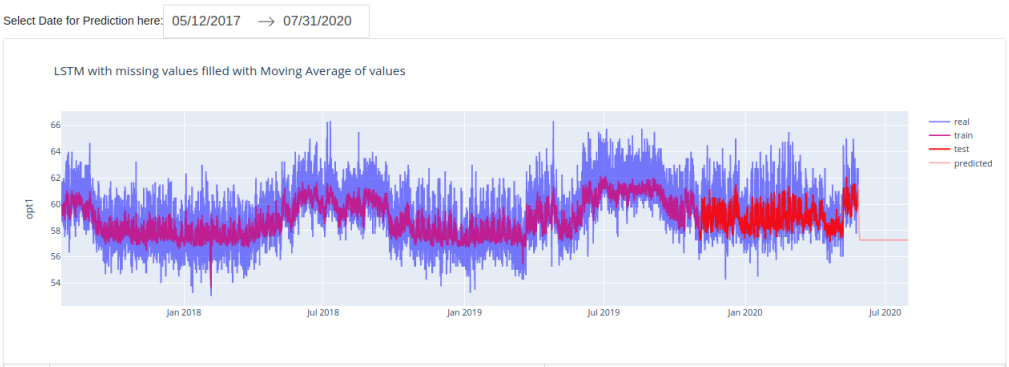
Figure 4.8: View of the Application.

- Customer: 0774f71910cbacf0e630b18dcc8aadf29ed4e3040cde3458dfbbd0341ba0b22e
- System: 098ebbd9dcc581a0064d8319ef14868611ce8aabe034d962ff8bdbca5d1ea2d6
- Physical Address: 4.1.22.0
- Logical Address 1.1.POINT.7.19
- Serial number: 5657605
- Variable: opt1
- Sensor type: FAP - O420

Forecasting the next two months using the model, we have the result shown in Figure 4.9. One of the problems observed with LSTM in this experiment is slow training time.

4.4 Holt - Winters Time Series Modeling

Before applying HW to the dataset, we first need to confirm if whether the series is stationary or non-stationary. Since we aim to automate this process, visual inspection may not be ideal. Also visual inspection may not be applicable in the case of large dataset as is the case here because some details of the data are hidden and may be difficult to spot. We therefore applied two common statistical approaches for the test for stationarity: Augumented Dickey-Fuller (ADF) and Kwiatkowski–Phillips–Schmidt–Shin (KPSS).



Metric	Train set(80%)	Test set(20%)
MAE	0.8887	0.9050
MSE	1.3137	1.4239
RMSE	1.1462	1.1933

Figure 4.9: Prediction with LSTM.

4.4.1 Stationarity test

4.4.1.1 Augumented Dickey-Fuller

The ADF test is a statistical test used to quantitatively test if a series is stationary or not. In this test, the null hypothesis confirms that the series is non - stationary. Figure 4.10 shows the closest example of a non stationary series in our dataset. For example, if the critical values we are considering is a p - value of 0.05 and the test gives a value less than this significant value then we reject the null hypothesis. This confirms that the series is probably strictly stationary. On the other hand, if the p - value is greater that the significant value of 0.05, then we conclude that the series is probably non-stationary.

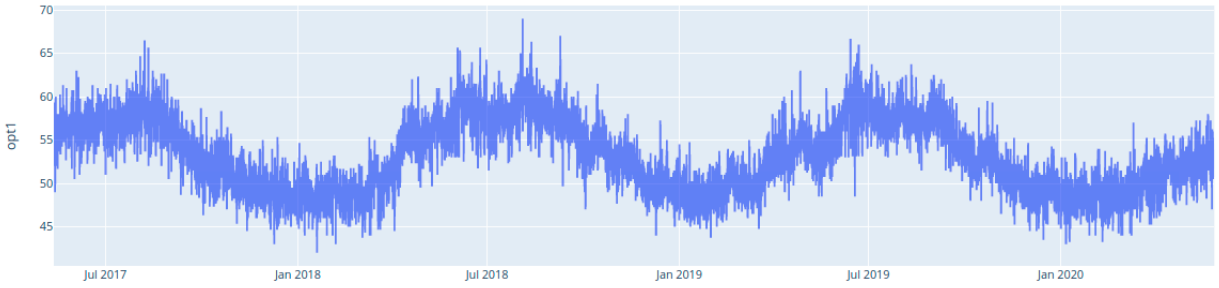


Figure 4.10: An example of a non stationary series in our dataset

4.4.1.2 Kwiatkowski–Phillips–Schmidt–Shin (KPSS)

The null hypothesis of the KPSS test confirms the presence of trend stationarity. It is used such that if the p - value is less than the significant value of 0.05 then we can conclude that the series is non stationary but if the p - value is greater than 0.05 we say the series is trend stationary.

	ADF	KPSS
$p < 0.05$	Series is stationary	Series is not trend stationary
$p > 0.05$	Series is non stationary	Series is trend stationary

Table 4.2: ADF and KPSS results with different p - values.

	ADF	KPSS	Conclusion
case 1	Non stationary($p > 0.05$)	Non stationary($p < 0.05$)	Series is non stationary
case 2	Stationary($p < 0.05$)	Stationary($p > 0.05$)	Series is stationary
case 3	Non stationary($p > 0.05$)	Stationary($p > 0.05$)	Series is trend stationary
case 4	Stationary($p < 0.05$)	Non stationary($p < 0.05$)	Difference stationary

Table 4.3: ADF and KPSS possible results and conclusion.

The KPSS test is usually used in conjunction with the ADF test and the result of these two tests is interpreted as shown in Table 4.2. As the two tests give different results with different p - values, Table 4.3 shows the possible combination of these results and the conclusion. Case 1 confirms the series is non stationary. A series is non stationary if the mean, variance and autocovariance change with time. Case 4 shows that the series is difference stationary. Difference stationary implies that a series needs to be "differenced" before it can be stationary. As was mentioned in chapter 2.1.10, differencing is a method of transforming a time series dataset in other to make it stationary [3]. It is done by by subtracting the previous observation from the current observation.

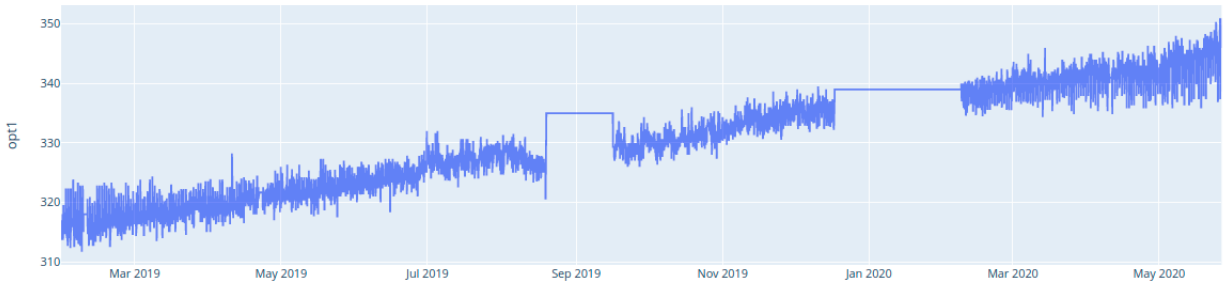


Figure 4.11: Opt1 series upward trend example.

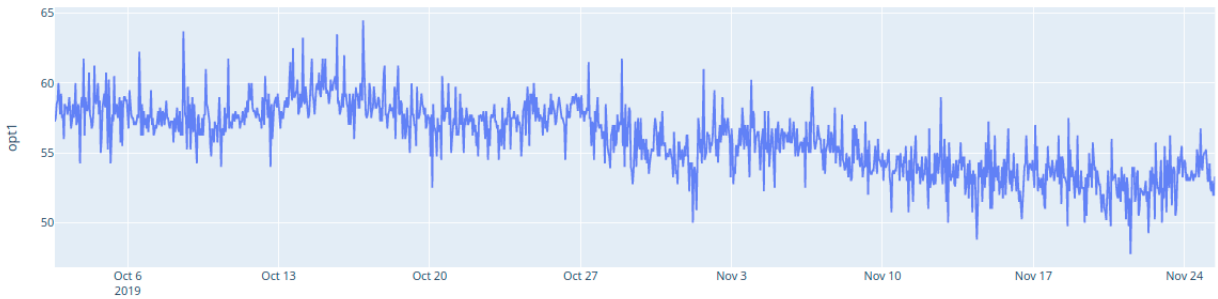


Figure 4.12: Opt1 series downward trend example.

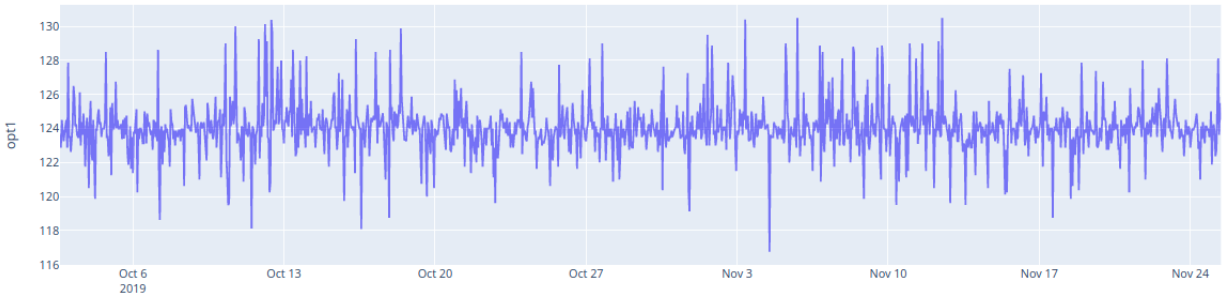


Figure 4.13: Opt1 series without trend example.

A non-stationary series usually possesses seasonality but may or may not necessary have trend. If a series possesses trend in addition to being non stationary(as identified by case 1), HW is expected to perform well. Even without a trend, HW performance is expected to be good(case 4). For cases 2 and 3, especially when there is no seasonality in the series, the performance of HW will be poor.

We also confirmed stationarity on the aggregated dataset (of the NYC311 data shown in Figure 4.7) using ADF and KPSS. We then train Seasonal Autoregressive Integrated Moving Average (SARIMA) models and select lowest Akaike Information Criterion (AIC). We also trained HW. We then compare the results of HW and SARIMA with that of the baseline, LSTM.

4.4.2 Seasonality and Trend test

As previously confirmed, HW can be applied to time series with seasonality and trend [3]. Sections 2.1.4 and 2.1.5 respectively explain seasonality as the characteristic of a time series in which the data experiences regular and predictable changes that occur in a period while the trend is an increasing or decreasing slope in a series. HW automatically computes the trend parameters in the series. Hence, we do not need to perform any computation to detect this. However, in order to objectively and quantitatively determine the strength(strong or weak) and direction(positive, negative or none) of a trend, the Mann-Kendall Test [3] is one of the most popular parametric tests used in times series data on the assumption that this data has serial correlation. Figures 4.11 and 4.12 show a strong positive trend and weak negative trend respectively while Figure 4.13 above shows a zero trend. Other parameters of the HW model automatically calculated are the level and seasonal components. The algorithm also needs to confirm if the series is additive or multiplicative. A series is additive if the magnitude of the seasonal component is constant with time. If it changes with time, then it is multiplicative. We have assumed that the series in this work is additive. For the NYC311 data, a seasonal period of 12(s=12) is confirmed. This is obvious from Figures 4.6 and 4.7. After these, the data is used to train LSTM, SARIMA and HW.

4.4.3 Models Training Time

For Bosch data, there are several python packages for the implementation of SARIMA and HW, `pmdarima` [37] and `statsmodels` [38] were used initially. However, these two take too much time to train. PdM for FDS requires a timely response to critical situations. We therefore had to switch to `rapids` [39]. Figures 4.14 and 4.15 show the training of HW models implemented with `statsmodels` and `rapids`. While in Figure 4.14, HW trained for more than 5 hours, in Figure 4.15, HW trained

for about 4 seconds with improved MAE, MSE and RMSE scores. However, `rapids - CuML` does not output in - sample prediction (this explains why there are no training set prediction in Figure 4.15. `rapids - CuML` works by distributing the training process over the GPU. It should, however, be noted that `rapids` implementation of `autoarima - SARIMA(p,d,q)(P,D,Q,s)` still takes time to train the model. This is after some parameters of the models like `p` - autoregressive order, `d`, differencing degree, `D` - seasonal differencing term and `s` - the seasonal period have been automatically computed. NYC311 data was used to successfully train SARIMA and the model with the lowest Akaike Information Criterion (AIC) selected. Same was done with HW. The selected models are then compared using the evaluation metrics scores.

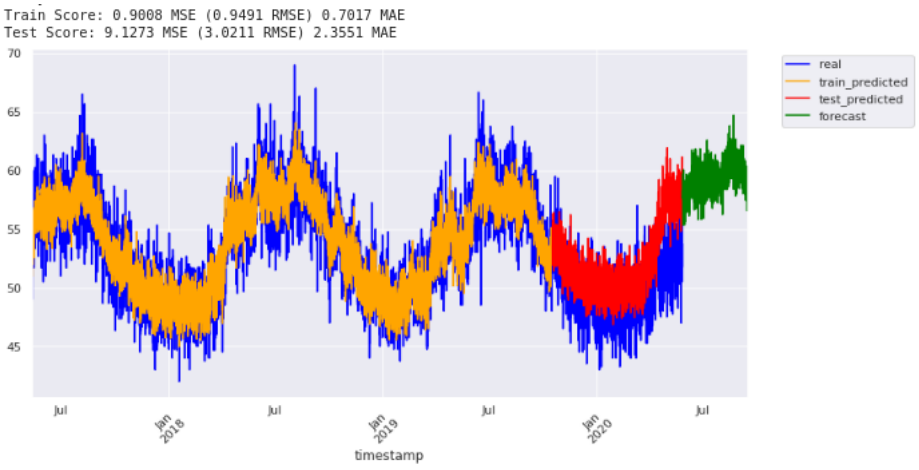


Figure 4.14: HW results with statsmodels

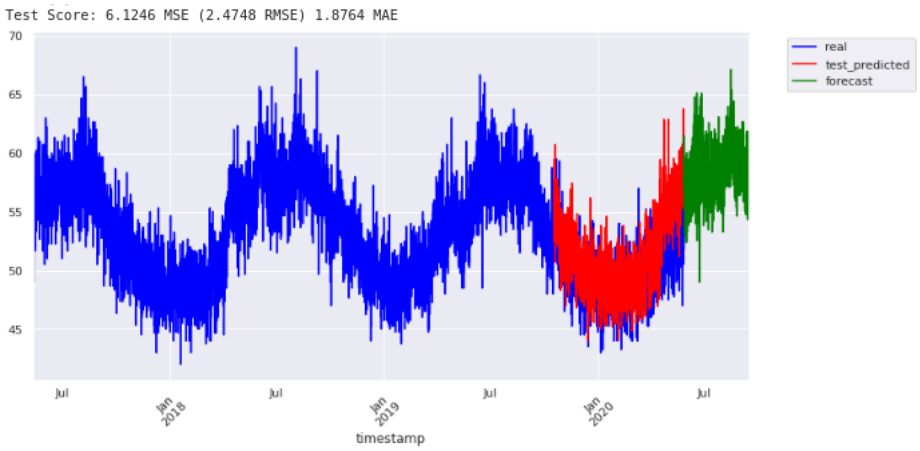


Figure 4.15: HW results with rapids(CUML)

4.4.4 Evaluation Metric

The Mean Absolute Error (MAE), Mean Squared Error (MSE) and Root Mean Squared Error (RMSE) are the evaluation metrics used for the models trained with both data. The models with the lowest values of these metrics are the best. MAE gives a linear penalty to each error. A smaller MAE indicates a better fit of the model to the data.

4.4.4.1 Mean Absolute Error (MAE):

MAE is the average of the absolute differences between the predicted and actual values.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{4.1}$$

where y_i is the actual value, $\hat{y}_i$ is the predicted value, and n is the number of data points.

4.4.4.2 Mean Squared Error

MSE is the average of the squared differences between the predicted and actual values. MSE gives a quadratic penalty to the errors. Larger errors are penalized more than smaller ones. A smaller MSE indicates a better fit but is sensitive to outliers.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{4.2}$$

4.4.4.3 Root Mean Squared Error

This is the square root of the MSE. RMSE can be interpreted in the same units as the dependent variable, making it more intuitive than MSE. Like MSE, it also gives more weight to larger errors.

$$RMSE = \sqrt{MSE} \tag{4.3}$$

or

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{4.4}$$

4.4.4.4 Metrics Comparison:

- MAE is less sensitive to outliers than MSE or RMSE because it doesn't square the errors. As a result, large errors don't get exaggerated.
- MSE and RMSE, by squaring the errors, give more weight to larger errors. This can be useful if large errors are particularly undesirable.
- RMSE, being in the same units as the target variable, can be easier to interpret than MSE.

Table 4.1: Dataset Columns summary

Field Name	Description
Unique Key	Unique identifier
Created Date	Date SR was created
Closed Date	Date SR was closed
Agency	Acronym of responding City Government Agency
Agency Name	Full Agency name
Complaint Type	First level identifying the incident or condition.
Descriptor	Provides further detail on complaint type.
Location Type	Describes the type of location
Incident Zip	Incident location zip code.
Incident Address	House number of incident.
Street Name	Street name of incident address
Cross Street 1	First Cross street incident location
Cross Street 2	Second Cross Street incident location
Intersection Street 1	First intersecting street incident location
Intersection Street 2	Second intersecting street incident location
Address Type	Type of incident location.
City	City of the incident location.
Landmark	If the incident is a Landmark, its name
Facility Type	Type of city facility associated to the SR
Status	Status of SR submitted
Due Date	When responding agency is expected to update.
Resolution Action Updated Date	Date when the SR was last updated .
Community Board	Provided by geovalidation.
Borough	Confirmed by geovalidation.
X Coordinate (State Plane)	Geo-validated, X coord. of the incident location.
Y Coordinate (State Plane)	Geo validated, Y coord. of the incident location.
Park Facility Name	If it is a Parks Dept facility, its Name
Park Borough	If it is a Parks Dept facility, its Borough
Vehicle Type	If it is a taxi facility, its Type.
Taxi Company Borough	If it is a taxi, its Company Borough.
Taxi Pick Up Location	If its a taxi, the taxi pick up location
Bridge Highway Name	If its a Bridge/Highway, the name.
Bridge Highway Direction	If its a Bridge/Highway, the direction.
Road Ramp	If its a Bridge/Highway, says if it is Road/Ramp.
Bridge Highway Segment	Bridge/Highway additional informaion.
Latitude	Geo based Lat of the incident location
Longitude	Geo based Long of the incident location
Location	Combination of the geo based lat and long

Chapter 5

Results and models comparison

In this chapter, we will discuss the outcomes derived from the Holt-Winters (HW) method and other models employed in this study. A comparison will be drawn between the models formulated using Bosch data and those based on NYC311 data. Additionally, we will delve into the web application crafted with Dash for Bosch data and its counterpart designed with the Shiny app for NYC311 data..

5.1 Holt-Winter and LSTM with Bosch Data

Using the dash application, we have trained a model each for Long Short-Term Memory (LSTM) and HW with average and moving average missing value imputation methods. LSTM has shown better results as provided by the evaluation metrics for the test sets in Tables 5.1 and 5.2, however HW has been faster in terms of the training time.

Figure 5.1: LSTM results with average missing values imputation method.

It should, however, be noted that rapids implementation of autoarima - SARIMA(p,d,q)(P,D,Q,s) still takes time to train the model. Seasonal Autoregressive Integrated Moving Average (SARIMA) has however been implemented with the NYC311 data. Also, HW requires the presence of at least two complete cycles to model seasonality(see Figures 5.5, 5.6, 5.7, 5.8).

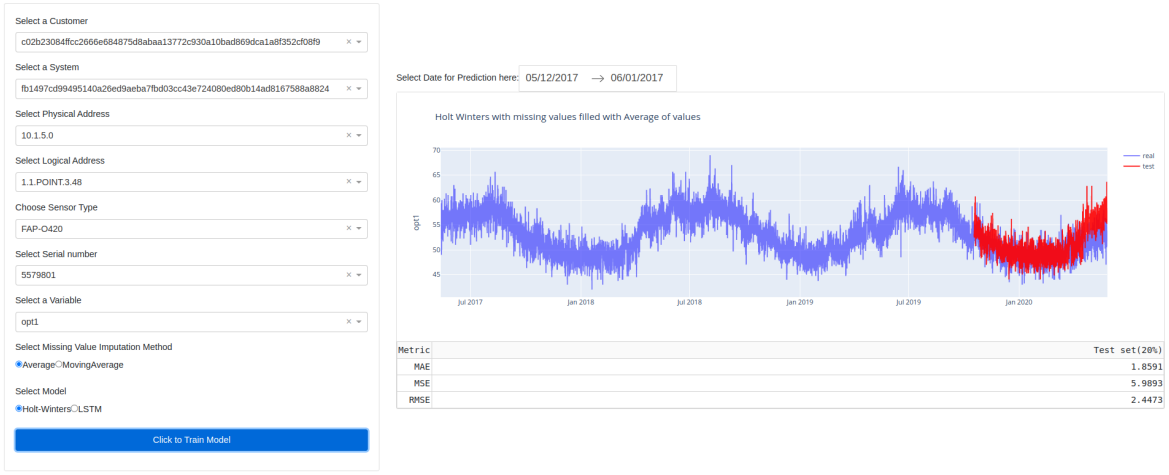


Figure 5.2: HW results with average missing values imputation method.

Figure 5.3: LSTM results with moving average missing values imputation method.

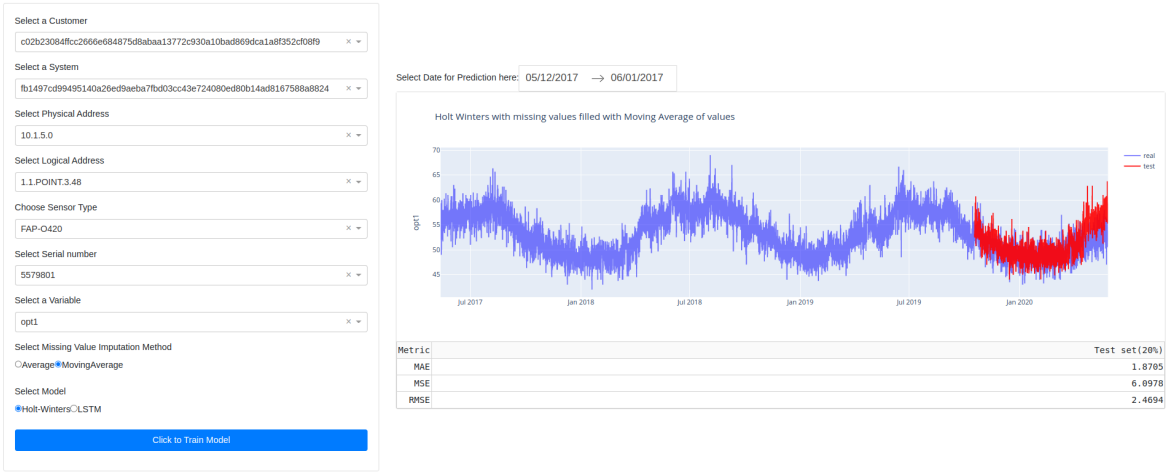


Figure 5.4: HW results with moving average missing values imputation method.

Table 5.1: Bosch data: Models result with Metric using Average

Metric	LSTM	HW
MAE	0.9742	1.8591
MSE	1.7181	5.9893
RMSE	1.3108	2.4473

Table 5.2: Bosch data: Models result with Metric using Moving Average

Metric	LSTM	HW
MAE	0.9728	1.8705
MSE	1.7099	6.0978
RMSE	1.3076	2.4694

5.2 Holt-Winter and SARIMA with NYC311 Data

Table 5.3: SARIMA Model with different parameters

Model	AIC
$M_1 : SARIMA(1,0,1) \times (1,0,1)_{12}$	2432.89
$M_2 : SARIMA(1,0,0) \times (1,0,1)_{12}$	2431.04
$M_3 : SARIMA(1,0,0) \times (1,0,0)_{12}$	2437.40
$M_4 : SARIMA(1,0,0) \times (2,0,0)_{12}$*	2429.85
$M_5 : SARIMA(1,0,0) \times (2,0,1)_{12}$	2431.60
$M_6 : SARIMA(1,0,1) \times (2,0,1)_{12}$	2433.55

From the six SARIMA models based on Table 5.3 and Figures 5.9, 5.10, 5.11, 5.12 , 5.13, 5.14, only models M_2, M_3 and M_4 have parameters that are significant and of these three, the model with the lowest AIC is $M4$.

We compared additive, damped and multiplicative methods and select the model with the lowest AIC. Of the three candidates models in HW method, HW with multiplicative Error, Non Trend and Multiplicative Seasonality has the lowest AIC and it was therefore selected. The AIC values of these models are in Table 5.4. Figures 5.15, 5.16 and 5.17 show the models outputs while Figures 5.18 and 5.19 show the residual plots of SARIMA and HW respectively.

Forecast for the next one year(12-months) were made with the two models: HW and SARIMA as shown in Figures 5.20 and 5.21 respectively. The figures present the train set and test set graph of the observed and fitted data. The forecast is presented over a confidence interval of 80% and 90%.

In the zoomed-in graphics of Figure 5.22, HW test set data is seen to reside 80% and 90% confidence interval till the 9th month(August 2021). The farther into the future forecast is made, the faster the model loses predictability power. Hence forecast for this model was only close to the real values till July 2021. However, for SARIMA using the zoom-in graphics of Figure 5.23, the forecast was inconsistent with the test set data.

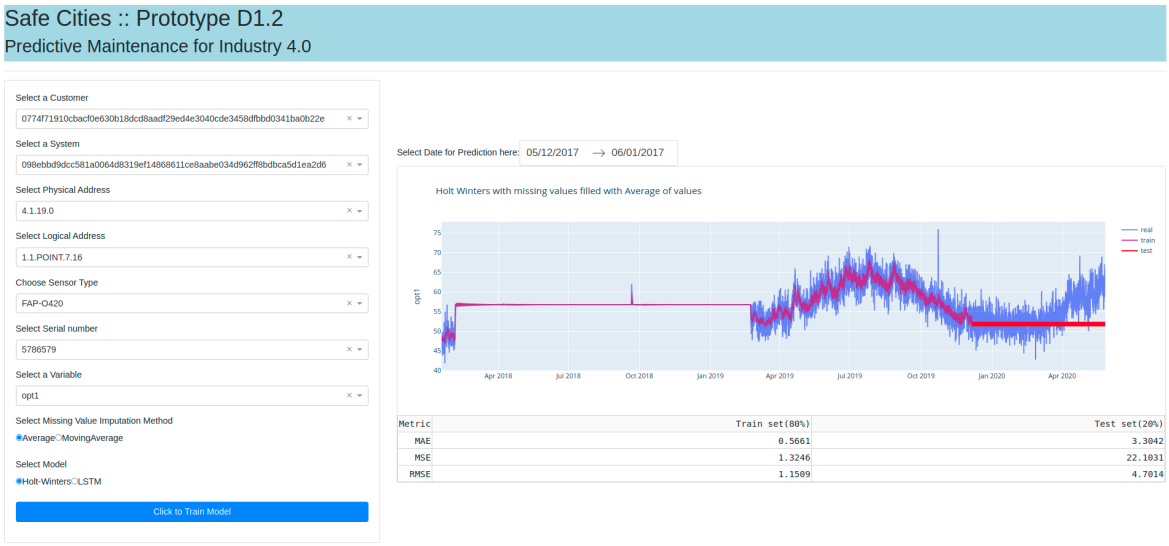


Figure 5.5: Previous results of HW. Fails to model the repeating cycle.

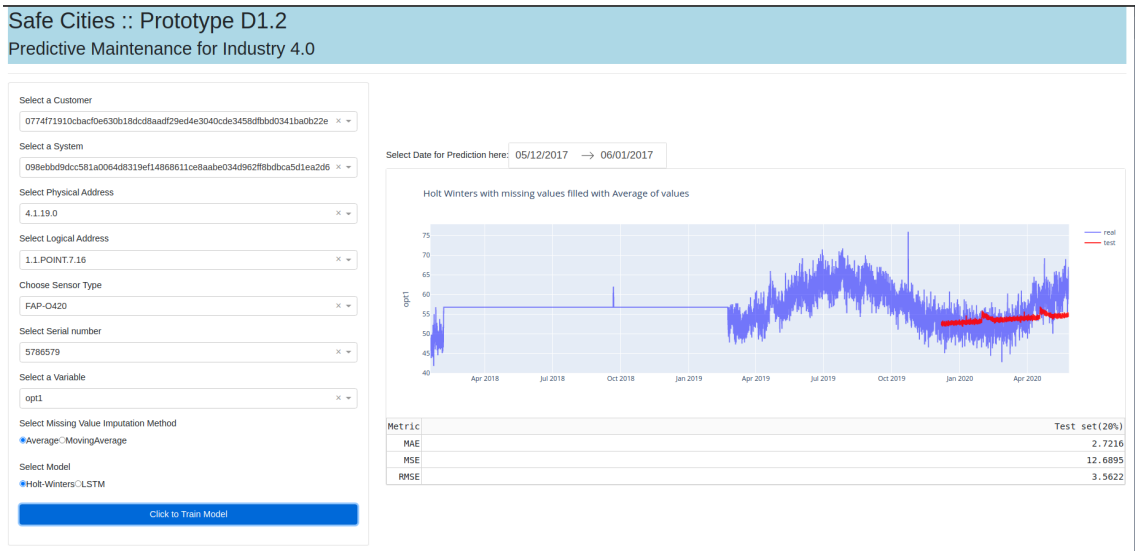


Figure 5.6: Current results of HW. Fails for less than 2 cycles in train set data with missing values filled with mean of values.

From Table 5.5, it is clear that the model that holds the lower values for Mean Absolute Error(MAE), Mean Square Error(MSE) and Root Mean Square Error(RMSE) is HW. From this, we conclude HW gives a better forecast than SARIMA given this particular data set. We have extended the models developed in this work to all the five(5) Boroughs of NYC and also trained LSTM model to forecast monthly complaint(Link to the application: here).

Figure 5.7: Current results of HW filling missing values with the mean. Train set data has less than 2 full cycle.

Figure 5.8: Current results of HW filling missing values with moving average. Train set data has less than 2 full cycle.

Table 5.4: ETS Model with different parameters

Model	**AIC**
$M_1 : ETS(M, N, M)$	2663.807
$M_2 : ETS(A, N, A)$	2698.298
$M_3 : ETS(A, A_d, A)$	2703.889

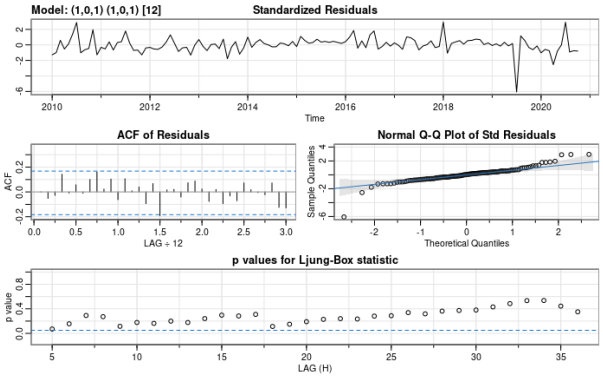


Figure 5.9: $M_1 : SARIMA(1,0,1) \times (1,0,1)_{12}$

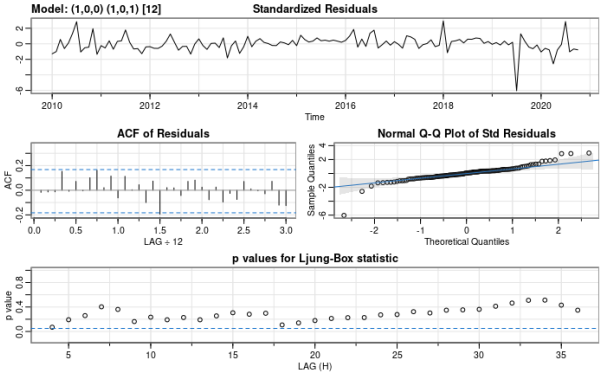


Figure 5.10: $M_2 : SARIMA(1,0,0) \times (1,0,1)_{12}$

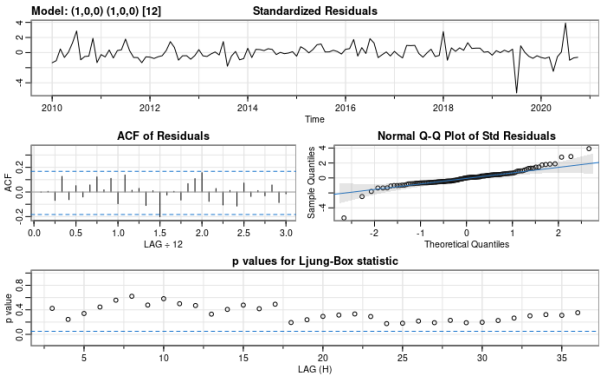


Figure 5.11: $M_3 : SARIMA(1,0,0) \times (1,0,0)_{12}$

Table 5.5: NYC311 data: Models with Metric

Metric	**SARIMA**	**HW**
MAE	29.896	19.023
MSE	15841634.000	11807165.000
RMSE	3980.155	3436.155

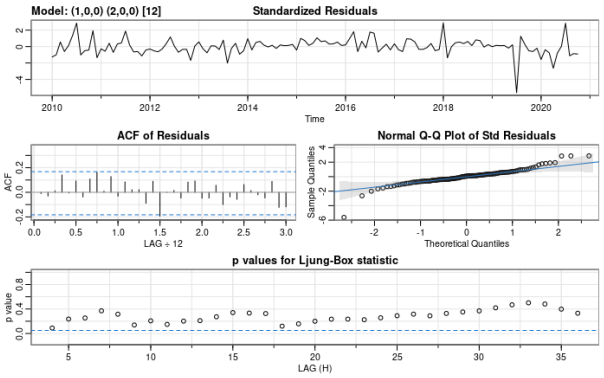


Figure 5.12: $M_4 : SARIMA(1,0,0) \times (2,0,0)_{12}$

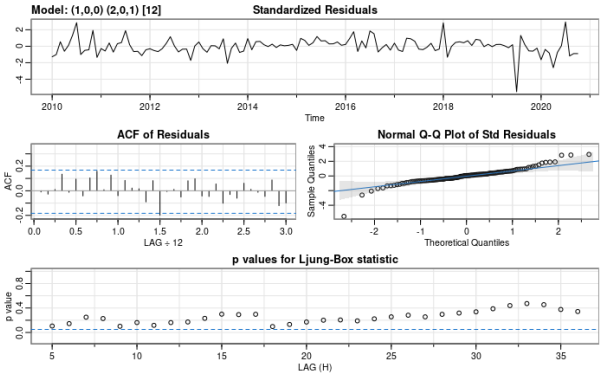


Figure 5.13: $M_5 : SARIMA(1,0,0) \times (2,0,1)_{12}$

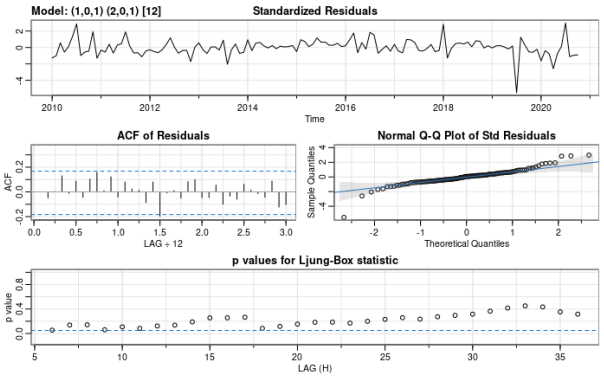


Figure 5.14: $M_6 : SARIMA(1,0,1) \times (2,0,1)_{12}$

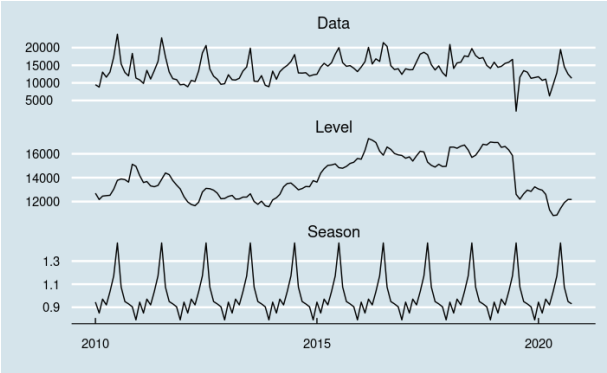


Figure 5.15: $M_1 : ETS(M, N, M)$

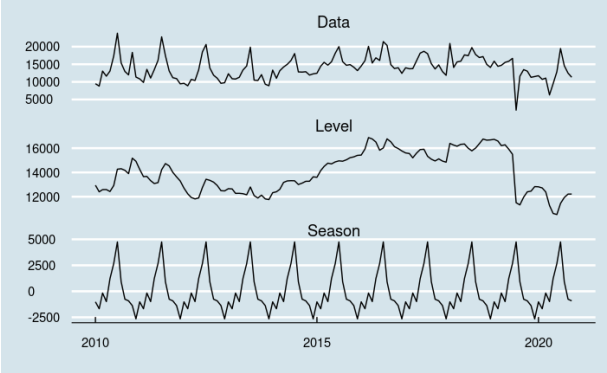


Figure 5.16: $M_2 : ETS(A, N, A)$

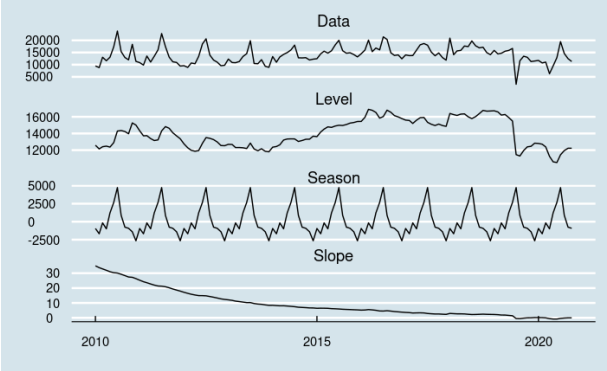


Figure 5.17: $M_3 : ETS(A, A_d, A)$

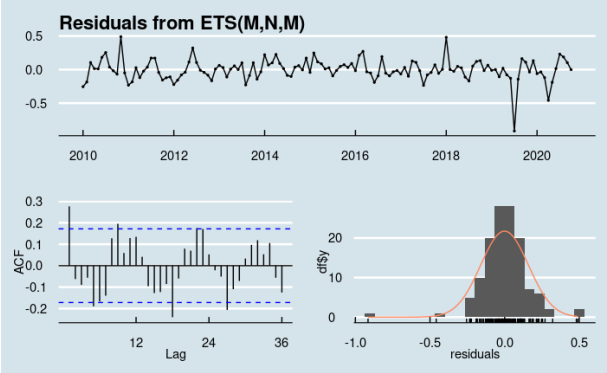


Figure 5.18: Residual Plot of ETS

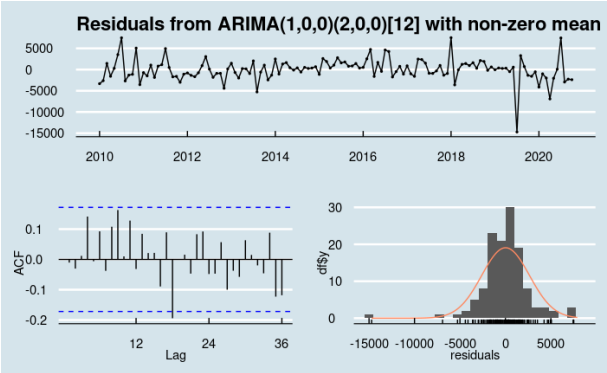


Figure 5.19: Residual Plot of SARIMA

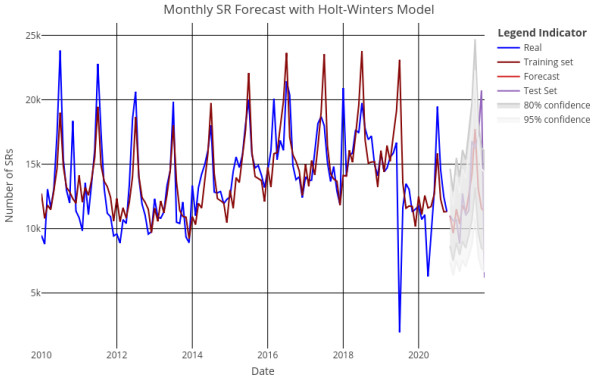


Figure 5.20: Plot of HW showing the train and test set real and forecast values

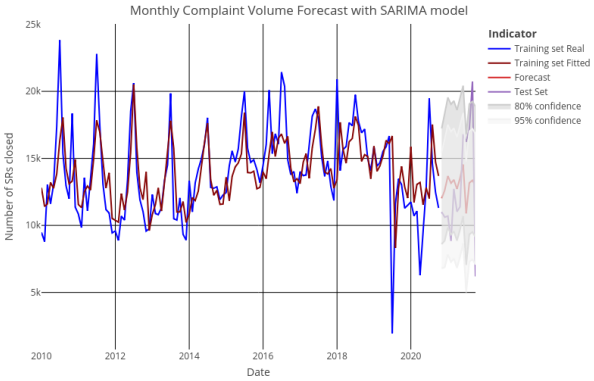


Figure 5.21: Plot of SARIMA showing the train and test set real and forecast values

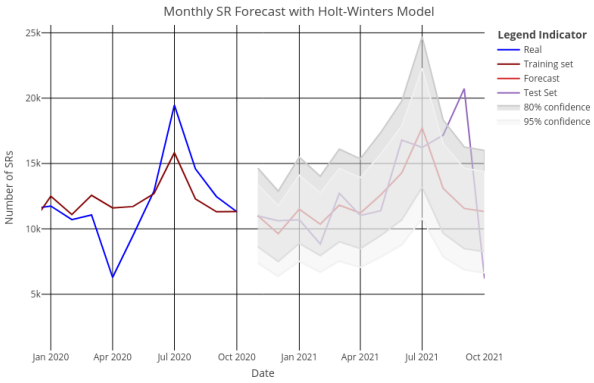


Figure 5.22: Zoomed in Plot of HW showing the train and test set real and forecast values

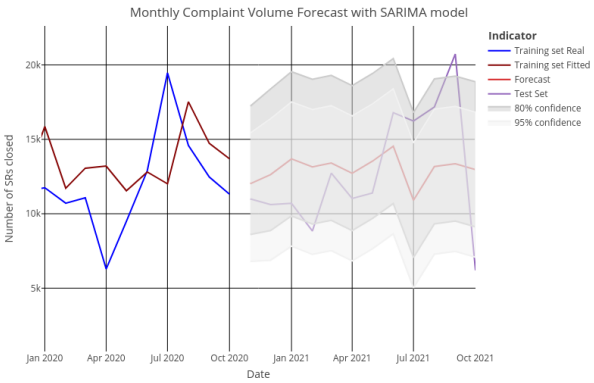


Figure 5.23: Zoomed-in Plot of SARIMA showing the train and test set real and forecast values

Chapter 6

Conclusions and future recommendation

6.1 Summary of the Work

In this work, our primary objective was to enhance Predictive Maintenance (PdM) for Fire Detection Systems (FDS) by leveraging the Holt-Winters (HW) model, given its proficiency in handling time series with notable trend and seasonality attributes. This endeavor has been successfully realized. We integrated the Fast Fourier Transforms (FFT) method for seasonality detection, which substantially improved our ability to identify seasonal patterns in the data. For benchmarking purposes, our methodology and the HW model were juxtaposed against the Long Short-Term Memory (LSTM) and Seasonal Autoregressive Integrated Moving Average (SARIMA) techniques. Due to constraints with Bosch's dataset rendering SARIMA training infeasible, we instead utilized the publicly accessible NYC311 data to train the SARIMA model, subsequently comparing its outcomes with those of the HW model.

Furthermore, to exhibit the practical applications of our research findings, we developed prototype platforms: a Dash prototype with Bosch data in Python, and a Shiny application with NYC311 data in the R programming environment. Both these prototypes elucidate the potential of our methodology in real-world settings, specifically within the FDS sector and in contexts of inventory or resource planning.

6.2 Main Findings

In this study, the HW model has demonstrated superior performance, especially in terms of speed, in forecasting time series data with trends and seasonality, especially when compared with SARIMA and the machine learning approach, LSTM. Furthermore, the FFT method has shown a marked advantage in detecting seasonal periods over the Autocorrelation Function (ACF). Given the dataset utilized in this work, the efficacy of FFT as a tool for seasonality detection, which underpins the success of the HW model, has been conclusively affirmed.

6.3 Other use-case application of our methodology

Our work finds real life usefulness in PdM for FDS in a microservice developed to train HW method to forecast the subsequent 24 hours values of critical sensors. If the next 24 hours prediction confirms some sensors are in dangerous states or the values of the variable a sensor measures approaches a threshold value that may be dangerous for the FDS, PdM is immediately schedule to fix the issues.

Moreover, the methodology adopted in this study extends its usefulness to resource planning and management. By forecasting the volume of issues in the coming days, resource managers are empowered with foresight, ensuring a more proactive and efficient approach to problem resolution.

6.4 Limitation of our Work

The HW package used in this work requires that a series has at least two cycles in the train set data otherwise, the package may not detect the most significant seasonal period. Not detecting the correct seasonal period can affect the result of HW.

Also, there can be more than one season in the series. HW are limited in this regard as they require only one value for the seasonal length. Using one value for the seasonal length for a series that has multiple seasonality will affect the model results.

6.5 Future Works

In light of the primary constraints of the HW model, specifically its ineffectiveness with series exhibiting les than two cycles in the training set and its reliance solely on the most prominent seasonal period, it is valuable to compare HW with other rapid and promising models, such as FFT, to determine which offers superior performance.

Bibliography

[1] Toshio Nakagawa. Sequential imperfect preventive maintenance policies. *IEEE Transactions on Reliability*, 37(3):295–298, 1988.

[2] Chin-Yi Lin, Yu-Ming Hsieh, Fan-Tien Cheng, Hsien-Cheng Huang, and Muhammad Adnan. Time series prediction algorithm for intelligent predictive maintenance. *IEEE Robotics and Automation Letters*, 4(3):2807–2814, 2019.

[3] R.J Hyndman and G. Athanasopoulos. *Forecasting: Principles and Practice*. OTexts, Australia, 2nd edition, 2018.

[4] Md Hussain and Ishtiak Mahmud. pymannkendall: a python package for non parametric mann kendall family of trend tests. *Journal of Open Source Software*, 4(39):1556, 2019.

[5] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[6] Little T Musbah H, Aly H. *A Novel Approach for Seasonality and Trend Detection using Fast Fourier Transform in Box-Jenkins Algorithm*. IEEE Canadian Conference on Electrical and Computer Engineering. Institute of Electrical and Electronics Engineers Inc., 2020.

[7] Xiaodong Jiang, Sudeep Srivastava, Sourav Chatterjee, Yang Yu, Jeffrey Handler, Peiyi Zhang, Rohan Bopardikar, Dawei Li, Yanjun Lin, Uttam Thakore, Michael Brundage, Ginger Holt, Caner Komurlu, Rakshita Nagalla, Zhichao Wang, Hechao Sun, Peng Gao, Wei Cheung, Jun Gao, Qi Wang, Marius Guerard, Morteza Kazemi, Yulin Chen, Chong Zhou, Sean Lee, Nikolay Laptev, Tihamér Levendovszky, Jake Taylor, Huijun Qian, Jian Zhang, Aida Shoydokova, Trisha Singh, Chengjun Zhu, Zeynep Baz, Christoph Bergmeir, Di Yu, Ahmet Koylan, Kun Jiang, Ploy Temiyasathit, and Emre Yurtbay. Kats, 3 2022.

[8] Hilal Tekgöz, Sevinc Ilhan Omurca, and Kadir Yunus Koc. Estimation of remaining useful life based on time series analysis. In *2022 7th International Conference on Computer Science and Engineering (UBMK)*, pages 273–277. IEEE, 2022.

[9] Victor Velasquez and Wilfredo Flores. Machine learning approach for predictive maintenance in hydroelectric power plants. In *2022 IEEE Biennial Congress of Argentina (ARGENCON)*, pages 1–6. IEEE, 2022.

[10] Georgios Makridis, Dimosthenis Kyriazis, and Stathis Plitsos. Predictive maintenance leveraging machine learning for time-series forecasting in the maritime industry. In *2020 IEEE 23rd international conference on intelligent transportation systems (ITSC)*, pages 1–8. IEEE, 2020.

[11] Gian Antonio Susto and Alessandro Beghi. Dealing with time-series data in predictive maintenance problems. In *2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 1–4. IEEE, 2016.

[12] Kedar Kulkarni, Umamaheswari Devi, Amith Sirighee, Jagabondhu Hazra, and Praveen Rao. Predictive maintenance for supermarket refrigeration systems using only case temperature data. In *2018 Annual American Control Conference (ACC)*, pages 4640–4645. IEEE, 2018.

[13] O Geoffrey Okogbaa and Xia Peng. Time series intervention analysis for preventive/predictive maintenance management of multiunit systems. In *SMC'98 Conference Proceedings. 1998 IEEE International Conference on Systems, Man, and Cybernetics (Cat. No. 98CH36218)*, volume 5, pages 4659–4664. IEEE, 1998.

[14] Abdulla I Almazrouee, Abdullah M Almeshal, Abdulrahman S Almutairi, Mohammad R Alenezi, and Saleh N Alhajeri. Long-term forecasting of electrical loads in kuwait using prophet and holt–winters models. *Applied Sciences*, 10(16):5627, 2020.

[15] CK Mandara, Vinay Kumar Jadoun, and Anshul Agarwal. Load forecasting of electric power distribution system using different techniques: A review. In *2022 IEEE Students Conference on Engineering and Systems (SCES)*, pages 01–06. IEEE, 2022.

[16] Hmeda Musbah and Mo El-Hawary. Sarima model forecasting of short-term electrical load data augmented by fast fourier transform seasonality detection. In *2019 IEEE Canadian Conference of Electrical and Computer Engineering (CCECE)*, pages 1–4. IEEE, 2019.

[17] Che Liu, Bo Sun, Chenghui Zhang, and Fan Li. A hybrid prediction model for residential electricity consumption using holt-winters and extreme learning machine. *Applied energy*, 275:115383, 2020.

[18] Keith Hollingsworth, Kathryn Rouse, Jin Cho, Austin Harris, Mina Sartipi, Sevin Sozer, and Bryce Enevoldson. Energy anomaly detection with forecasting and deep learning. In *2018 IEEE international conference on big data (Big Data)*, pages 4921–4925. IEEE, 2018.

[19] Ahmad Abu Sleem, Ayman R Mohammed, Saleh Al Shkoor, and Haitham Saleh. Peak forecasting for electricity loads in jordan using a weighted combination of feed forward back propagation neural network and holt-winter. In *2022 3rd International Conference on Intelligent Engineering and Management (ICIEM)*, pages 226–231. IEEE, 2022.

[20] Majed Saleh Al-Hafid and Ghazi Hussein Al-maamary. Short term electrical load forecasting using holt-winters method. *Al-Rafidain Engineering Journal (AREJ)*, 20(6):15–22, 2012.

[21] Lu Yan and Meng Liu. A simplified prediction model for energy use of air conditioner in residential buildings based on monitoring data from the cloud platform. *Sustainable Cities and Society*, 60:102194, 2020.

[22] Yusun Ahn, Haneul Choi, and Byungseon Sean Kim. Development of early fire detection model for buildings using computer vision-based cctv. *Journal of Building Engineering*, 65:105647, 2023.

[23] Fatma M Talaat and Hanaa ZainEldin. An improved fire detection approach based on yolo-v8 for smart cities. *Neural Computing and Applications*, pages 1–16, 2023.

[24] Swapnil Bagwari, Anita Gehlot, Rajesh Singh, Neeraj Priyadarshi, and Baseem Khan. Low-cost sensor-based and lorawan opportunities for landslide monitoring systems on iot platform: a review. *IEEE Access*, 10:7107–7127, 2021.

[25] Khaled Bashir Shaban, Abdullah Kadri, and Eman Rezk. Urban air pollution monitoring system with forecasting models. *IEEE Sensors Journal*, 16(8):2598–2606, 2016.

[26] Ling-Yu Lv, Cheng-Fei Cao, Yong-Xiang Qu, Guo-Dong Zhang, Li Zhao, Kun Cao, Pingan Song, and Long-Cheng Tang. Smart fire-warning materials and sensors: Design principle, performances, and applications. *Materials Science and Engineering: R: Reports*, 150:100690, 2022.

[27] Paul D Rosero-Montalvo, Pınar Tözün, and Wilmar Hernandez. Time series forecasting to fill missing data in iot sensor data. *IEEE Sensors Letters*, 2023.

[28] Carla Silva, Marvin F da Silva, Arlete Rodrigues, José Silva, Vítor Santos Costa, Alípio Jorge, and Inês Dutra. Predictive maintenance for sensor enhancement in industry 4.0. In *Asian Conference on Intelligent Information and Database Systems*, pages 403–415. Springer, 2021.

[29] Ronghong Mo, Yiyang Pei, Neelakantam Venkatarayalu, Pereira Nathaniel, AB Premkumar, and Sumei Sun. An unsupervised tcn-based outlier detection for time series with seasonality and trend. In *2021 IEEE VTS 17th Asia Pacific Wireless Communications Symposium (APWCS)*, pages 1–5. IEEE, 2021.

[30] AM Davey and BE Flores. Identification of seasonality in time series: A note. *Mathematical and computer modelling*, 18(6):73–81, 1993.

[31] Javier Contreras, Rosario Espinola, Francisco J Nogales, and Antonio J Conejo. Arima models to predict next-day electricity prices. *IEEE transactions on power systems*, 18(3):1014–1020, 2003.

[32] Markéta Arltová and Darina Fedorová. Selection of unit root test on the basis of length of the time series and value of ar (1) parameter. *Statistika: Statistics & Economy Journal*, 96(3), 2016.

[33] DK Dwivedi, GR Sharma, and SS Wandre. Forecasting mean temperature using sarima model for junagadh city of gujarat. *IJASR*, 7(4):183–194, 2017.

[34] Gene S Fisch and Anthony F Porto. Visual inspection of data: does the eyeball fit the trend? In *Human vision, visual processing, and digital display V*, volume 2179, pages 268–276. SPIE, 1994.

[35] Hmeda Musbah, Hamed H Aly, and Timothy A Little. A novel approach for seasonality and trend detection using fast fourier transform in box-jenkins algorithm. In *2020 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)*, pages 1–5. IEEE, 2020.

[36] Roxane Elias Mallouhy, Christophe Guyeux, Chady Abou Jaoude, and Abdallah Makhoul. Forecasting the number of firemen interventions using exponential smoothing methods: a case study. In *International Conference on Advanced Information Networking and Applications*, pages 579–589. Springer, 2022.

[37] Taylor G. Smith et al. pmdarima: Arima estimators for Python, 2017–. [Online; accessed 20-08-2022].

[38] Skipper Seabold and Josef Perktold. Statsmodels: Econometric and statistical modeling with python. In *9th Python in Science Conference*, 2010.

[39] RAPIDS Development Team. *RAPIDS: Collection of Libraries for End to End GPU Data Science*, 2018.