Aalborg University

Computer Science - Semester 1

Group it705e19

Weather Sensor Anomaly Detection

Alef Pir
David Jahanshiri
Andrei Ciobanu
Richard Tarnoczi
Marius Untaru

Manfred Jaeger December 19th 2019

This page intentionally left blank

AALBORG UNIVERSITET
STUDENTERRAPPORT

CS
Aalborg University
http://www.aau.dk

Abstract:

This report is about performing anomaly detection on a set of data collected by CityProbe weather sensors around the city of Aalborg, Denmark.
Anomaly detection was to be performed by making use of time series prediction, specifically an seasonal ARIMA prediction model.
This report documents the design, implementation and evaluation stages of the ARIMA model and the anomaly detection model it is to be used for.
It is concluded that the model fulfills the requirements set forth in the problem statement, but that there is potential for improvement, e.g. improving the seasonal ARIMA prediction model to use the multivariate features of the sensor data set.

Title:
Weather Sensor Anomaly Detection

Project Period:
01-09-2019 - 19-12-2019

Project Group:
it705e19

Participants:
David Jahanshiri
Alef Pir
Andrei Ciobanu
Richard Tarnoczi
Marius Untaru

Supervisor:
Manfred Jaeger

Copies:
1

Page Numbers:
47

Date of Completion:
19-12-2019

This page intentionally left blank

Preface

This report is written by group it705e19 of the department of computer science at Aalborg University, during the period of 1st September 2019 to the 19th December 2019.

The "Vancouver Numeric" method is used for citations. They are of the following form: *...according to Jens Jensen at Aalborg University [1].* If this was a real citation, the source for this citation would be shown as [1] in the section of this report named "Bibliography".

Citations may have the following information: author, title, publishing method, year, online status and last visited.

David Jahanshiri
djahan14@student.aau.dk

Alef Shenol Pir
apir12@student.aau.dk

Ciobanu Andrei Cristinel
acioba19@student.aau.dk

Richard Tarnoczi
rtarno19@student.aau.dk

Marius Constantin Untaru
muntar19@student.aau.dk

Contents

Part I

Introduction

Chapter 1

Introduction

This project takes part of the curriculum for Master's Degree in Computer Science program, and investigates a few areas which must be solved with tools, knowledge and skills gained throughout the semester. The goal is to plan and implement a model for anomaly detection.

1.1 Problem Statement

In Aalborg, there is a network of multi-functional weather sensors called CityProbes. They are located in each neighborhood throughout the city and they provide data points from all of the specific locations. They provide pollution data, weather data, such as temperature, humidity, CO levels, and so on.

These weather sensors sometimes have hardware, installation or software faults that lead to producing unreliable readings. Both catastrophic failures, but also smaller 5-10% sensor reading shifts over time. There can also be environmental factors that can lead to temporary unreliable readings.

It could therefore be useful to detect anomalies in the data provided by the sensors. This could show when a sensor is failing, so that it can be fixed/replaced quickly, limiting the amount of missing or erroneous data.

It can also be useful in order to be able to show users of the sensor data when they can trust a sensor and when they can not.

From this there was derived an initial problem statement:

- ***How can anomalous sensor data be identified?***

Part II

General Theory

Chapter 2

General Theory

The purpose of the following chapter is to give a brief introduction to the reader about the most important theories that are going to be used throughout the report. The chapter includes theory about anomalies, machine learning and time series.

2.1 Anomaly Theory

An anomaly is a data point, which value is not following the pattern of the other outliers. anomaly is often meant for data point, that has a legit value, but it is farther from the mean or the median in a distribution.

Anomaly on the other hand is used for those data points, that are illegitimate. In this case the value gets generated by a different process than most of the other data points. In the case of this project it would mean that a sensor gave a faulty reading.

There are several types of anomalies[1].

- **Point**
- **Contextual**
- **Collective**

A data point is called a **point anomaly** if it has a value that is far from the entirety of the rest of the data set. For example, a reading of 100 degrees Celsius would be anomalous regardless of context when considering weather sensors.

A data point is a **contextual anomaly**if it has a value that differs significantly from the other data points in the same context. The same anomaly might not considered an anomaly in a different context. For example 20 degrees might be an in-lier in the summer period, but in the winter period it is a definited anomaly.

A **collection of data points** can be seen as anomalous if they, when looked at together, are significantly different from the rest of the data set, but aren't anomalous in a contextual or point sense. For example, 25.55 degrees might be a normal temperature reading on a summer day, but it would be abnormal to have that exact reading many times in a row, i.e. at 13:00, 14.00, 15.00 and so on.

2.2 Machine Learning

The following chapter is going to give a brief introduction to the idea behind Machine Learning (ML) and some of its most crucial components and concepts.

"Learning is the ability of an agent to improve its behaviour based on experience" [2].

The goal of machine learning is to to create an agent, that can get better at doing a certain task, by learning from inspecting data that is given to it. This can be also referred as a learning problem.

A learning problem has numerous components. By taking a closer look at the most important of these, one is going to be able to get a better understanding about Machine Learning.

Such a learning problem has a so called task. A task is basically the procedure that is being performed by the agent and what it is supposed to improve at the end of the learning procedure.

Data is another crucial component. This is what is provided to the agent as a source for learning. The experience gained from this procedure will be used by the agent in the future in order to make assumptions on data, that has not been seen before by the given agent.

When the provided data is not perfect, then it can be assumed that there is noise. Noise can be meant both for missing data and errors. Anomalies can be also considered as noise.

In machine learning one can talk about different kinds of learning. The most common ones are supervised learning and unsupervised learning.

Supervised learning is applied, sets of examples and sets of features are being dealt with and the goal is to predict a specific outcome. It is being used, when values for data are being predicted, that has not been seen by the agent before. The knowledge for making the predictions is gained by learning a set of example data. There are multiple ways of doing supervised learning and the two most common ones are classification and regression.

In classification the model is being trained with a data set, where the models are labelled - differences between labelled and unlabelled data will be discussed shortly. During the learning the agent learns what are the differences between the values of the different labels. When it gets applied to a set of data that has not been seen before, then the agent is going to be able to label them based on its previous experiences.

Regression is applied, when the data is not labelled - meaning that the data is unlabelled - as it is a normal value or an anomalous value, but instead it is ordered and continuous. It is a linear approximation of a relationship between two or more variables.

It is important to clarify what is actually the difference between labelled and unlabelled data. Labels are indicating a certain attribute of a data point.

Unsupervised learning on the other hand is being dealt with, when it is needed to extract structure from the data. It is useful when the aim is to train the agent to carry out different operations in the data set.

Training the agent is an important part of the machine learning procedure. In this phase the agent is being prepared to deal with unseen data in the future by giving a model to it. Besides a model training data is given to the agent as well. The training data gets split into two further parts: training set and testing set.

The training set is used to learn and gain experience by the agent, meanwhile the gained it applies the gained experience on the test data, so the developers can get a feedback, how well the model did.

2.3 Time Series Analysis

A time series is a number of observations recorded one after the other at frequent and regular time intervals while avoiding any random and non-consecutive measurements[3]. These observations would have the benefit of improving the overall identification of a phenomenon that could be reflected by the data.

The time series analysis consist of four major components:

- **Trend** - throughout the time series analysis, whenever there is a constant uprise or downfall observed within the data, then that could be interpreted as a trend and depending on the frequency of the observations, this information could be useful for various statistics.
- **Seasonality** - this refers to the observations of the data that show how a specific pattern occurs in between regular intervals which is also what makes it different from the other component in the sense that it tends to repeat itself overtime.
- **Cyclic Variations** - compared to seasonality, the cyclic variation in a time series has an oscillatory movement over a period that is longer that one year. One cycle is made up of four phases such us prosperity, recession, depression and recovery, in that exact order.
- **Irregular Variations** - another component of the time series analysis also includes the random fluctuations that are often times unpredictable because of various natural causes or even because of man-made reasons. [4]

Often times, there are cases where a time-series does not necessarily have all the above mentioned components. There could be the case where there is no trend, but there could be seasonality and that really depends on the type of data upon which the observations are made.

Another essential aspect regarding time series is decomposition procedure, which describe the trend and seasonal factors. The main reason of making use of decomposition is to effectively estimate the seasonal effects that can be used to remove the seasonal effect from a value, resulting in more clearly trends visibility. There are two main decomposition models that can be made use of:

- **Additive model** - based on the values of the components, another further part of the analysis would be the composition of those values, where by adding them in between each other, the time series value could be calculated at a given time and this would only be applied when the components are not proportionate to each other. The additive model is useful when the seasonal variation is relatively constant over time. This includes this project time series, leading our group to choose it as decomposition model later for removing seasonality. Here is the mathematical representation of the model:

$$Y(t) = T(t) + S(t) + C(t) + I(t)$$

- **Multiplicative model** - The values of the components could also be multiplied altogether in the case where if the values were added, but they would not be independent from one another, then the multiplication would help with getting a better composition. Mathematically, the model is represented as follows:

$$Y(t) = T(t) * S(t) * C(t) * I(t)$$

Where Y(t) represents the value of the observation at time **t** and T(t), S(t), C(t), I(t) respectively represent the Trend, Seasonality, Cyclic Variations and the Irregular Variations, all at time **t**. An extensive decomposition might include long-run cycles, holiday effects but for the purpose of this project only the trend and seasonal decomposition will be taken in consideration.

In the figures below a visual example of how both models look like are illustrated, where the seasonal variance difference from one model to another. The first model visualization refers to the Additive procedure where a more constant variance is present. On the other hand, a more emphasised variation is present in the second figure. This reflects in the mathematical equation of each model as well.

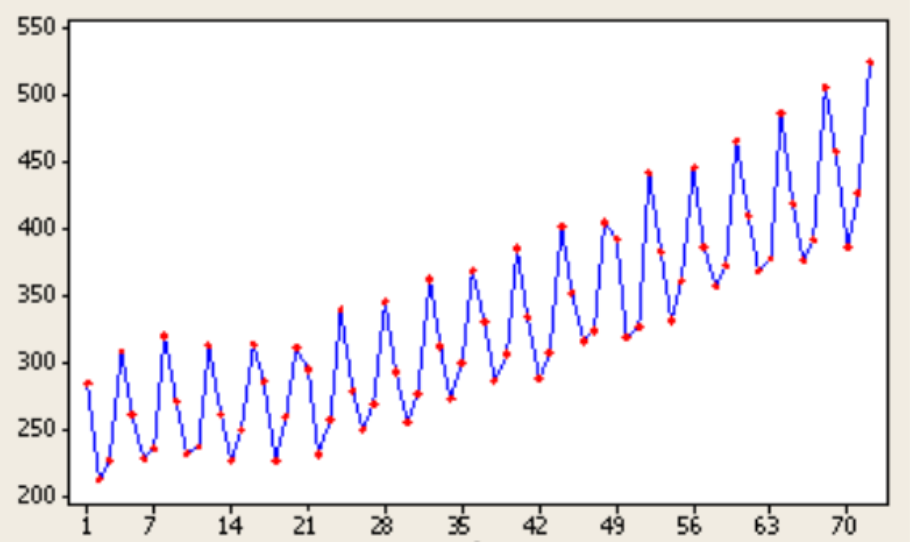

Figure 2.1: Decomposition Additive Model[3]

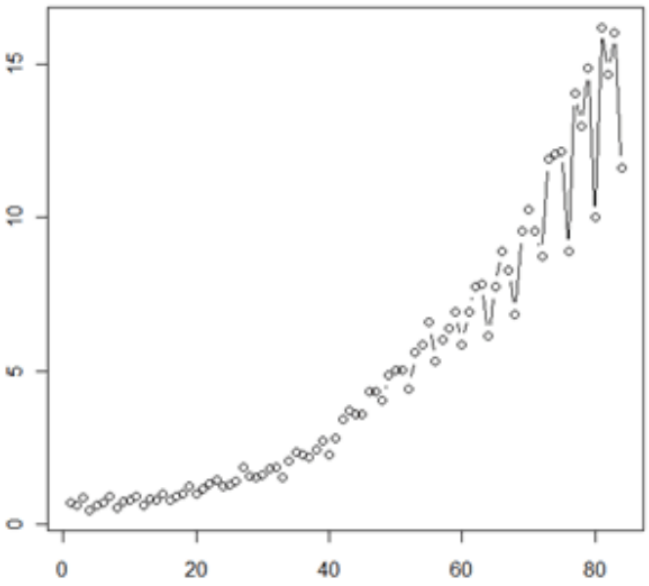

Figure 2.2: Decomposition Multiplicative Model[3]

Part III

Data

Chapter 3

Data

In order to start researching the problem area it is necessary to analyze the given data. In this chapter the data set is presented, describing the contents and data types. The scope is also discussed, i.e. which features and sensors from the data set will be used when performing anomaly detection later in the report. Finally the preprocessing performed on the data is discussed.

3.1 Data Description

In this section the data set used for this report will be described.

The data set consists of data collected from the multi-functional weather sensors called CityProbes, that are installed around the city of Aalborg in North Jutland, Denmark. The CityProbe sensors have a variety of sensors that collect various types of weather data, such as temperature, noise, pollution and more. The sensors are made by a Danish company called MONTEM A/S[5] and are installed on lamp posts around the city.

Cityprobe data was accessed using the website `http://cityprobe.ciss.dk/`. The website is a collaboration between Aalborg University and Center for Embedded Software Systems (CISS)[6]. The website allows you to visualize the data collected by the sensors, as well as export the data to csv file format as has been done for this project.

The latest data set accessed from the above website is from 8. November 2019 and contains 484541 individual observations from the period 6. September 2018 until 8. November 2019. The data set contains all of the observations from each installed sensor. The sensors were all installed at different times, leading to different amounts of observations being available from each. Each sensor took observations at varying intervals ranging from just 10 minutes to several hours depending on factors such as sensor battery level. Each measurement includes a timestamp and the data set is, therefore, considered a *time series*.

The data set can also be seen as having multiple timeseries, in that you can separate observations by which sensor they were taken from.

The data set is unlabelled, meaning that there is just one set of data that includes both anomalous observations and normal observations. This means that the anomaly detection methods used later in the report must be unsupervised.

Each observation includes the features described in Table 3.1:

Feature Name	Description	Data Type
core_id	Unknown, most likely related to device firmware	int
serial	Serial of the sensor it was measured on	string
fw_version	Firmware version running on the sensor	int
loc_id	Each location has a unique id	int
lat	Latitude coordinates for sensor location	float
lon	Longitude coordinates for sensor location	float
desc	Text description of location	string
record_date	Date and time of measurement	string
battery_level	Battery level of sensor in percent (0-100%)	float
rain_median	Not used, no data provided	
rain_avg	Avg rain fall (0-4096)	float
rain_max	Max rain fall (0-4096)	int
rain_min	Min rain fall (0-4096)	int
noise_median	Not used, no data provided	
noise_avg	Average DBA value measured	float
noise_max	Maximum DBA value measured	int
noise_min	Minimum DBA value measured	int
luminosity	Luminosity measured in Lux (0-88.000)	int
temperature	Temperature measured in Celsius	float
humidity	Relative Humidity (0-100%)	float
pm10	PM10 (m/m^3)	int
pm25	PM2.5 (m/m^3)	int
no2	NO2 concentration (0-4096)	int
co	CO concentration (0-4096)	int
pressure	Atmospheric Pressure in hPa	float

Table 3.1: Description of data set features

Rain and noise data is measured by taking 100 samples in a 5 second period of time, this is done once for every measurement. From these samples, the avg, max and min are calculated. The rain data, NO2 concentration, and CO concentration data are analog readings that should have a value of between 0 and 4096, there are no measurement units for these features. These are, therefore, relative observations and can only be compared between these CityProbe sensors and not with external sources.

3.2 Data Sorting

3.2.1 Redundant Data Removal

The data set includes some empty features and some extra data that isn't necessary for our purposes.

The features "rain_median" and "noise_median" have been removed from the data set, as they are completely empty.

The feature fw_version was also removed as all measurements have the same value of "7". Core_id feature was removed as it does not contain relevant info and appears to have the same value on all measurements for a particular sensor.

It was also deemed unnecessary to have multiple features containing identifying data for the sensor. Desc (description) feature was removed, as there is a feature called "loc_id" containing a unique id that corresponds to the description. Latitude and longitude were removed too, as they correspond to a loc_id as well and were just duplicated data.

The Serial feature was removed, as there were never multiple serials at one loc_id in the data. Although it is possible it might be worth it to keep this data in the future, in case a new sensor with a new serial number is installed at the location of an existing sensor.

With these modifications the data set columns look as in Figure 3.2:

Feature Name
loc_id
record_date
battery_level
—
rain_avg
rain_max
rain_min
noise_avg
noise_max
noise_min
luminosity
temperature
humidity
pm10
pm25
no2
co
pressure

Table 3.2: Data set columns after cleaning

3.3 Data Scope

The data set has many different features and it is likely that each one would benefit from a different approach with regards to anomaly detection. It would therefore be beneficial to limit the scope of the anomaly detection to just one of these features.

There are also many different sensors, each one likely with a different anomaly profile. It is, therefore, necessary to choose just one of these sensors to focus on.

In this section it will be determined which feature and sensor this report will focus on.

3.3.1 Choice of Specific Sensors

It is important to choose the sensor or sensors that will form the basis for the analysis. The data set contains observations from a total of twenty-four unique sensors. Sensors can be identified based on the "loc_id" feature contained in each observation. Sensors will in the future be referred to as "sensor", followed by their unique location id("loc_id"). Each sensor has a different location and there are many other variables, that mean they each have a separate anomaly profile. It is for this reason not advisable to interpret the data set as one single time series, but rather as individual time series for each sensor.

Because of this it was chosen to limit the scope, by selecting only one of the sensors.

It is important to have one year of data in order to be able to model the yearly seasonality of the time series properly. Two-thirds of the twenty-four sensors did not start operating until between late March 2019 and early June 2019, heavily reducing the amount of data as well as limiting the capability to model full yearly seasonality.

It is also important to have a a little more than one year, as that allows there to be one year of training data and then some test data that can be used to compare against. The Pareto Principle[7] states that 80% of effects come from 20% of the causes and this is a common split in the world of data science, 80% training data and 20% test data.

There are only 2 sensors that can fulfill this requirement, sensor 16 and sensor 17. Sensor 16 started sending data on 6. September 2018 and sensor 17 started sending on 5. October 2018. These two sensors are also the two sensors with most individual observations. The sensors that have the next earliest initial measurement times did not start measuring until the 14. December 2018, meaning there would be very little test data, if one year of training data is needed.

As the scope is limited to one sensor, and data from sensor 16 is the most complete of the two sensors 16 and 17. Sensor 16 is chosen and only data from sensor 16 will be used in further analysis and eventually for evaluation of the anomaly detection methods.

3.3.2 Choice of Feature

In order to limit the scope, only one feature will be chosen from the data set. First one checks for features that have missing data. I.e. there is an observation but

there is a NaN value for the specific feature. The data set has a total of 41933 observations for location 16.

Feature	Missing Values
luminosity	0
rain_max	0
rain_avg	7627
rain_min	0
temperature	2
humidity	7
noise_avg	7627
noise_max	0
noise_min	0
pm10	0
pm25	0
no2	0
co	0
pressure	2

Table 3.3: Location 16 - Missing Values

It was decided that for the project the chosen feature is going to be temperature. There are multiple reasons for this decision. First of all, this was the data set feature, that most people are familiar with. The data set's temperature feature was almost complete, meaning that it was missing only 2 data points. Last but not least multiple 3rd party sources are available for historical temperature data, which could be also used to validate the data, that is gained from the sensors.

3.3.3 Conclusion

It was decided that it was best to interpret each location as a separate time series and that it was best to limit the scope to deal with just one sensor. Sensor 16 was chosen, as it has the longest period of data. The temperature feature was chosen as the only feature due to its familiarity and the quality of the data.

record_date	temperature

Table 3.4: Data set features after Feature Removal

3.4 Data Preprocessing

Although machine-learning based anomaly detection is to be used later, it can be still be worthwhile to do some simple preprocessing beforehand.

Removing observations that are outside hard boundaries is one method that can be done, essentially a simple filter. For example, the interval that temperatures will fall in between is known. The highest temperature ever measured in Denmark is 36,4 degrees Celsius and the lowest measured is -31,2 degrees Celsius[8]. Expanding

the interval by about five on each side, should allow some extra room in case of inaccuracies. Therefore, the temperature should be equal to or between -36 degrees and 41 degrees and observations outside this interval will be removed.

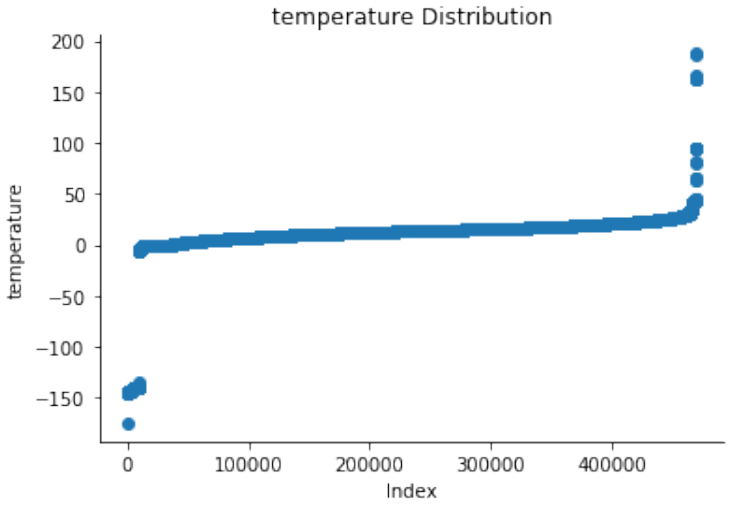

Figure 3.1: Temperature Distribution Before Filtering

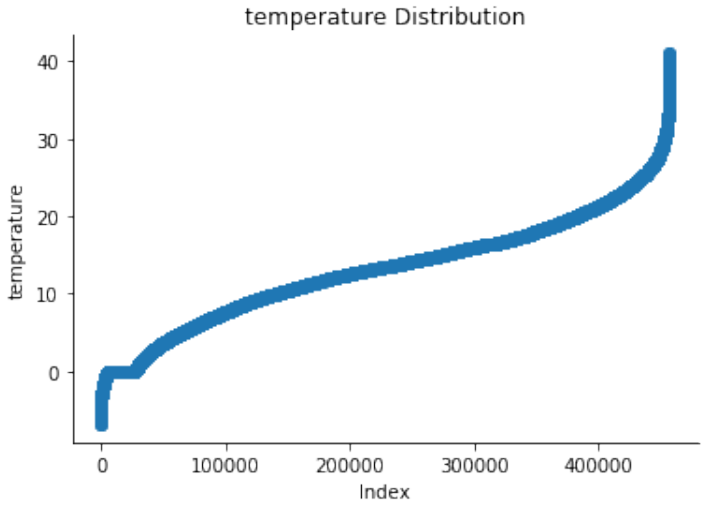

Figure 3.2: Temperature Distribution After Filtering

Figure 3.2 shows the distribution of temperature observations, temperature on the y-axis and x-axis is the index number of the observation given after sorting observations by temperature value in ascending order.

3.4.1 Data Resampling

Observations in the data set have a variable frequency. However, varying frequency cannot provide the result. Due to this, it is necessary to resample the data set in manner of having a more consistent frequency. This means that the frequency will change and there will be one value for each hour in the resampled data set. There are two types of resampling that could be considered:

- **Upsampling** refers to the increasing the frequency of the samples, resulting in more data points. New rows are created with NaN values, then a missing values interpolation takes place to fill up the data set with new values.
- **Downsampling** refers to decreasing the frequency of the samples, resulting in less data points. It is being applied to the current data set for consistency reasons. A common way to do this, is to group the data set as desired (in the case of the project case hourly) and create new value from each group of all the data existing for an hour time span. By applying the **mean()** function, can result one single value of every group which will then represent the hourly data set resample. With the data frequency adjusted, the data preprocessing can now continue with dividing it into training respectively testing data sets.[9]

3.4.2 Data Split

The data set is split into training and train data. To do so, the Pareto Principle was followed. The used data from September 6. of 2018 until September 6. of 2019 for training and the data for testing was collected in 2019, from the 6. of September until the 8. of November. The training data set will be part of the model configuration, while the test data will be part of the model validation and evaluation.

3.5 Conclusion

The data set contains 484541 individual observations from the period 6. September 2018 until 8. November 2019. It contains a varying number of observations from 24 unique sensors and can be considered a time series, due to being indexed in time order.

Based on the amount of data available and the scope of the project, it was decided that only data from sensor number 16 which is at location called Computer Science was to be used going forward. Temperature was also chosen as the only feature that would be used going forward as it is one of the most representative one and serves best to the purpose of this project. A first major factor in achieving the goal of the project was to allocate the adequate time and resources to analyse and preprocess the data set, since it was part of the requirements before proceeding with the next chapter.

Part IV

Approach

Chapter 4

Approach

This chapter introduces the general idea of how anomaly detection will be performed using time series forecasting. It also introduces the ARIMA model of time series forecasting and describes the theory in further detail.

4.1 Related Work

In order to decide on methods of anomaly detection for the data set, research was done into related work. Specifically research papers regarding anomaly detection performed on time series.

One of the examined papers was a paper called Learning States and Rules for Time Series Anomaly Detection by Stan Salvador, Philip Chan, and John Brodie. In this paper the authors demonstrate a way to perform time series anomaly detection. The authors conclude, that it is possible to use time series forecasting in order to detect anomalies, furthermore they confirm that their model was able to detect anomalies.

Another research paper called Comprehensive Real-time Anomaly Detection System by Sooyeon Lee, and Huy Kang Kim was also relevant. It proposes conventional time-series approaches such as the previously mentioned time series forecasting. The paper takes in consideration the main points that presents interest for this project, such as *time series data, short-term forecasting, anomaly detection.* The document performs a detailed analysis regarding the methodology applied for each model and an experimental evaluation. It concludes that ARIMA is well-suited for time series forecasting in the context of anomaly detection.

Based on these two papers it seems that ARIMA-forecasting-based anomaly detection is a good fit for the data set as it is oriented towards time series data. They seem to indicate that a short-term forecast is the most suited when performing anomaly detection.

4.2 Anomaly Detection General Approach

Based on the earlier in chapter, General Theory, section Time Series Analysis, it is possible to do anomaly detection with the help of time series forecasting.

The idea is to train an initial prediction model based on the training data. That would be used to forecast values forward in time, an out-of-sample forecast. That same model would be fed new data constantly as it comes in and prior to forecasting future values. How far into the future to forecast is to be decided, but it would be no more than one day into the future. This is because theoretically, accuracy is reduced the farther the predicted observation is from the latest actual data add to the model.

The predicted values would then be compared to the actual values. An interval would be set, i.e. if the actual value is within 5 degrees of the predicted, it would be considered not anomalous. This is depicted in Figure 4.1, where the red line shows the predicted observations and the blue line shows the actual observations. The grey shaded area represents the upper and lower bounds based on the predicted observations. Several actual observations are outside of these bounds and would be classified as anomalous (red circles).

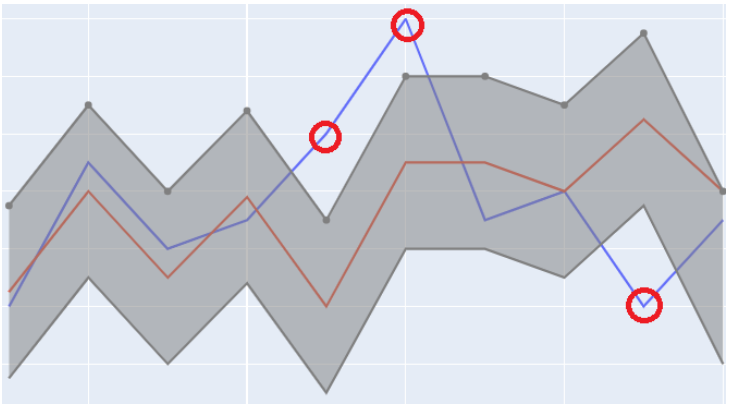

Figure 4.1: Graphical Depiction of Anomaly Detection Method

4.3 Fundamental Concepts

4.3.1 ARIMA model

Arima Model Basics and Parameters

The forecasting model chosen to go with, is called ARIMA. It stands for Auto-Regressive Moving Average and incorporates various types of methods. The basic methods that fall under ARIMA family, are Auto-Regressive(AR), the Integrated Part(I), and Moving Average(MA). To fully understand the model, a brief introduction into its fundamental methods (AR, I, and MA) is required.

- **AR** - It is an autoregressive model which predicts future behaviour based on past behaviour. It works the best when there is some correlation between values in a time series and the values that precede and succeed them. To represent it mathematically, the following formula must be considered:

$$y_t = \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \epsilon_t$$

  where:
  y_t = outcome variable **y** at some point **t** in time
  $\phi_1, \cdots, \phi_p$ = are the model coefficients/parameters
  $y_{t-1}, \cdots, y_{t-p}$ = are the past observations
  ϵ_t = white noise(e.g. randomness)

  The model above is described as AR(p) model, an autoregressive model or the order p.

- **I** - The integrated part of the model represents the amount of differencing that needs to be done in order to achieve stationary. The mathematical formula behind the integrated part of the model can be written for first, second difference or when no differencing is required:
  - Formula for the first difference(d=1):

$$y_t = Y_t - Y_{t-1}$$

  - Formula for the second difference(d=2):

$$y_t = Y_t - 2Y_{t-1} + Y_{t-2}$$

  - Formula when no seasonal differences are needed(d=0):

$$y_t = Y_t$$

  Note that for a proper understanding of the above formulas, consider Y as the original series, and **y** the differenced/stationaried series.

- **MA** - The moving average is a technique used to observe the trends of the data set. It is the mean of any subsets of values. The moving average takes the first subset and calculate the average of the values then it "shifts forward" once more data come in. It plays an essential role in forecasting trends, at any

period of time. Usually *moving average* model is used for forecasting future values, but due to the problem statement addressed, where the main focus is to detect anomalous data, rather than forecasting future values, a *moving average smoothing* technique is more suitable. Smoothing the moving average differs from the initial moving average by removing the noise and express better the long term average. This helps in revealing more clearly the trend, seasonal and cyclic component of past values. The mathematical formula for the moving average of order q, MA(q) can be written as:

$$y_t = \mu + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \cdots + \theta_q \varepsilon_{t-q}$$

where:

y_t = weighted moving average **y** at some point **t** in time
$\theta_1, \cdots, \theta_q$ = are the model coefficients/parameters
$\varepsilon_{t-1}, \cdots, \varepsilon_{t-p}$ = are the noise error terms [10]

Both AR and MA components are linear combinations of present and past values of random variables, being considered stochastic processes. Stochastic means that the values come from a random probability distribution, and can be analyzed statistically, but cannot always be predicted precisely. The difference between the two is the fact that first model (**Autoregressive model**) is a combination of the observable values

$$y_{t-1}, \cdots, y_{t-p}$$

while the second one (**Moving Average model**) is a combination of the non-observable values

$$\varepsilon_{t-1}, \cdots, \varepsilon_{t-p}$$

ARIMA model can be summarized by three parameters:

- **p - the number of autoregressive terms**. It is the AR part, and it allows to incorporate the effect of past values into the model. Its description can be exemplified by considering that if it has been warm the past days, its most likely that will be warm tomorrow.
- **d - the number of nonseasonal differences**. There is a fundamental requirement to successfully implement ARIMA model, and that is to ensure data is stationarity. If this assumption fails when the stationarity test is run, then a nonseasonal difference must be applied. This would be similar to consider that will be the same temperature tomorrow if the difference in temperature in the last days has been very small.
- **q - the number of moving-average terms**. The last parameter of the model refers to the number of lagged forecast errors that can occur in the prediction equation/MA part. By doing this, can then be set the error of the model as a linear combination of the error values observed in the past.

The ARIMA model can be extended with different parameters and have diverse methods joining; e.g. *seasonality* is not initially a capability of the ARIMA model, but an extension can be added, which will result in forecasting seasonal data for various periods. By doing so, the model will then be called SARIMA, for distinguishing reasons.

This particular model proposes a seasonal differencing of a specific order, for the purpose of removing non-stationarity from the series. As an example, the difference between a current observation and the observation from the previous year respectively, is calculated as follows:

$$z_t = y_t - y_{t-s}$$

where:

s = the number of the periods in each season.(e.g. it could be 12 for a monthly series).
z_t = the seasonally differenced series itself

The model also introduces 3 extra parameters, often referred to as hyper-parameters, which represent the seasonal terms. The first one is specified as P for the autoregressive part of ARIMA (AR), continuing with D for the differencing part (I) and eventually Q for the moving average part (MA). The general term of the model can be expressed as:

$$SARIMA(p,d,q)(P,D,Q)[s]$$

Another capability ARIMA offers is forecasting based only on the past values of the forecast variable. The model assumes that the future values of the variable depend on its past values. It will then be called ARIMAX or dynamic regression model or vector ARIMA, and it will also include other independent predictors/variables.

ARIMA has no explicit seasonal indices, can be tough to interpret its coefficients and explain the functionality behind. Over-fitting with data or coefficients mis-identification are a constant danger as well, but it represents a strong underlined mathematical theory which comes in handy to predict intervals, with a large documentation, being one of the most popular machine learning models available.

The problem statement allows to stick with the basic methods that form ARIMA, therefore none of the above extensions will be part of this project.

As already established, ARIMA model requires data which have the characteristic of being stationary. Before moving on, an comprehensive understanding of the stationarity should take place.[10]

Stationarity

Stationarity is the property of a data set not to present signs of trend or variance in the values. In order to be able to come up with reliable forecasts, the time series observations should be as stationary as possible while avoiding to remain non-stationary by going through a transformation process. What that implies is that the statistical properties of the series such as the variance of the values, the mean of the values as well as the autocorrelation between the current values and the previous values, should be as constant as possible over time.

There are two main stationarity approaches for establishing if a data set is stationary or not, and each approach will consist on a series of methods to transform the existing time series into a stationary one:

Visualisation testing

The observations are based on the current data set, the Gaussian distribution and the Autocorrelation plot. Often the data set can reveal obvious trends or patterns which can conclude into stationary or non-stationary data, therefore it is considered a reliable approach in that sense.

Data points distribution can also represent a decisional factor when comes to stationarity. If the data near the mean of all values are less frequent in occurrence than data far from the mean, there is most likely is that the timeseries is non-stationary. This can be observed using Gaussian/Normal distribution plot.

Autocorrelation plot is an alternative to the previous two, since its decay tend to zero value can help expressing the stationarity characteristic. The slower the decay is, the more likely is to deal with non stationary data and vice versa.

Statistical testing

Statistical stationarity approaches come as complement to the visualization testing. There are diverse methods which can help with it, such as Augmented Dickey Fuller(ADF) test or Kwiatkowski-Phillips-Schmidt-Shin(KPSS) test. Both tests are based on two hypothesis, and one can be accepted while the other rejected.

Hypothesis Testing

The concept of Hypothesis Testing which need to be understood when considered dealing with stationarity. The summary of it consists in estimating the P value, which refers to the probability value that a specific hypothesis can be true. There are a series of steps that needs to be followed in Hypothesis Testing:

· **Specify the Null Hypothesis** (H_0)

The null hypothesis is defined as a statement with no effect, or difference between two or more factors, and usually the interest is to disprove it.

· **Specify the Alternative Hypothesis** (H_1)

The alternative hypothesis is the statement that has an actually effect/difference, and presents interest in proving it. The alternative hypothesis can be either *one-sided* providing one direction of the normal distribution only or two-sided, where both sides of the normal distribution are being considered.

· **Choose level of significance** (α)

The level of significance indicates what is the chance of an alternative hypothesis is getting accepted, when H_0 is true. In some cases $\alpha$is given. However in some other cases it is required to be chosen. It is common to to choose 0.05 as a value for α, since it is the most is the most common level of significance. This would mean that there is 5% that the alternative hypothesis is going to be accepted in case H_0 is true.

· **Calculate test statistic and corresponding P-value**

In order to evaluate a hypothesis, test statistics can be applied. The role of p-value is to describe the probability of getting a sample statistic. This value is calculated with the help of test statistics, which has many different implementations.

· **Conclusion with test statistic**

If the p-value is smaller or equal to α, then H_0 can be rejected. This also means that H_1 can be accepted, however, it does not mean that H_1 true, but it is acceptable. In case the test statistics is not smaller or equal to α, H_0 cannot be rejected.[11]

Parameter estimations

ARIMA takes in multiple parameters, which are the earlier mentioned p, q and d. These values need to be non-negative integers. In order to be able to make time series forecasts using ARIMA, values for these are needed to be found. The estimation of the parameters has many different approaches and this is considered as an important step, since it can have a great impact on the results.

In this subsection, the methods to find the best values for the parameters and were considered by team team, are going to be introduced. The processes, that are going to be considered are: autocorrelation function (ACF) of the data for , partial correlation graph of the data, Akaike's Information Criterion (AIC) and last but not least the Schwartz Bayesian Information Criterion (BIC) are going to be used.

The first one that is going to be looked at is the autocorrelation function. AFC creates an autocorrelation graph - it is also known as autocorrelation plot. It is important, to clarify, what autocorrelation really is. Autocorrelation in the reality is the correlation calculated for time series observations. It is using the values of earlier observations - the so called lags - of the same time series. When the ACF is getting used a plot is being drawn, where the x-axis is the lag values, whilst the y-axis is the value of correlation between the values of -1 and 1. These values can be calculated with the help of the formula below. In case the value, 0 it means, that there is no correlation. On the autocorrelation graph a confidence interval can be seen as well. Values outside of this correlation interval have a high likelihood of correlating. The usage of ACF is sufficient in case the time series was generated by an autoregression process.

$$p(h) = \frac{\gamma(h)}{\gamma(0)}$$

where:

$\gamma(h)$ = the covariance at lag h
$\gamma(0)$ = variance

Partial autocorrelation is the summary of an observation in a time series' relationship with the earlier observations with the removal of intervening observations. The correlation of an observation and an observation in the past has direct correlation as well as indirect correlation. Partial autocorrelation's goal is to remove these indirect correlations. In order to get a better understanding about partial

autocorrelation, it can help to take a look at the equation below, where PACF is applied to AR(p).

$$X_t - \phi_1 X_{t-1} - ... - \phi_p X_{t-p} = Z_t$$

where:

X_t = the value of lag at position t
ϕ = the φ coefficient, used to measure association between two binary variables
p = the number of autoregressive terms

The main purpose of Akaike's Information Criterion (AIC) to find the best model, and this way helping with the search for the efficient parameters. When comparing two models the better one is the one with the lower AIC score. AIC quantifies the goodness of fit and the simplicity of the model. The likelihood of a model can be increased by adding more parameters to it. There a corrected version of AIC for ARIMA, namely AICC. Akaike's Information Criterion can be applied in many different cases in order to find the best model.

Schwartz Bayesian Information Criterion (BIC) has a similar purpose as AIC, it helps with finding the better model. The greatest difference of BIC compared to AIC is that fact that the earlier gives penalty for the usage of more parameters, with other words it helps avoiding overfitting.

There is no need to choose either one or the other approach in order to find good parameters, but these different approaches can also be combined together.[12]

Part V

Application of Methods

Chapter 5

Application of Methods

5.1 Implementation of ARIMA Forecasting

In this section the theory of Box-Jenkins approach will be main point with emphasis on the principles of creating the ARIMA model that will be used later for anomaly detection.

5.1.1 Box-Jenkins method

The Box-Jenkins methodology was first introduced in 1970 by statisticians George Box and Gwilym Jenkins.[13] It is a widely used method which refers to a set of procedures in identifying a proper ARIMA model for training and fitting time series data, and then forecasting and evaluating the results of the fitted model. The model follows an iterative three-stage process of *model identification* and *model selection*, *parameter estimation* and *model checking*. It is important to mention that Box-Jenkins is a pattern which follows a concrete set of steps where the procedures involved in accomplishing the final goal may differ based on the project's specifications. This chapter will describe the application of the Box-Jenkins method in relation to CityProbe time series. Figure 5.1 shows the approach of the method for this project.

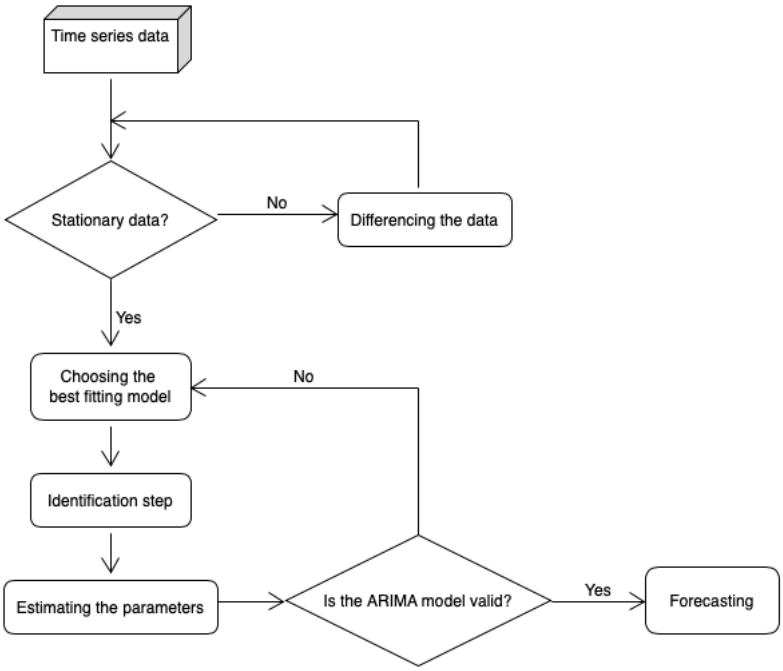

Figure 5.1: Box-Jenkins Approach

The first part of the method involves procedures of identifying if the time series is stationary and detecting if there is significant seasonality. In order to move on towards parameter estimation, the data must be made stationary, usually through differencing and transformation techniques. As shown on Figure 5.1, differencing can be reapplied as many times as needed to yield stationarity.

Analysis of Stationarity and Seasonality

As mentioned previously and as observed on Figure 5.1, first comes making sure the time series is stationary. Two methods of identifying stationarity will be applied as follows:

1. **Looking at visual plots**
2. **Statistical tests**

Visual Observations

As mentioned earlier in Chapter 4, section stationary, time series must be flat looking, without an obvious trend and no seasonality for it to have constants statistical values over different time periods. Firstly, simple visual observations can be done on the original data set. Figure 5.2 shows noticeable change in levels such as upwards going trend from January 2019 towards July 2019 and vice versa from a warmer season towards colder seasons. This indicates inconsistency in the mean value over time.

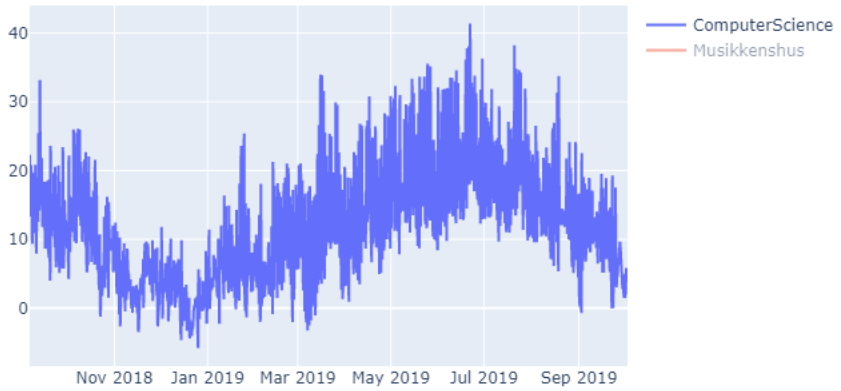

Figure 5.2: Original time series

Another requirement, as previously mentioned, is having a constant variance over time. Looking at Figure 5.2 is hard to tell if there is inconsistency in the variance. Plotting a histogram of the Gaussian distribution for temperature can help identify the statistical properties furthermore. Observing figure Figure 5.3, indicates that there is inconsistency in the variance over different periods of time due to the plot having a right skewed distribution with a longer right tail.

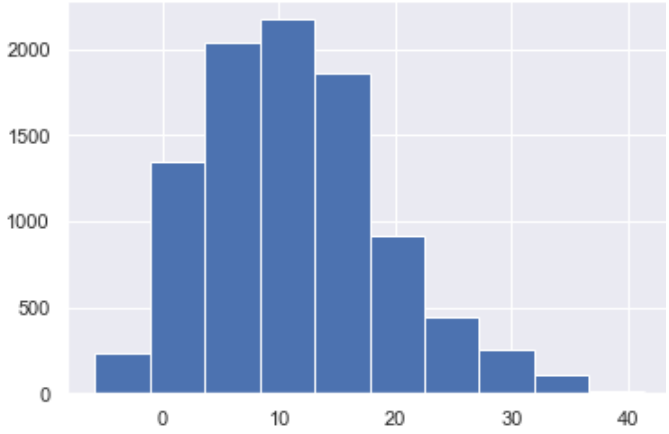

Figure 5.3: Gaussian distribution of temperature values

This means there is a further spread from the average value. Yielding symmetric looking distribution where the left side is a mirror of the right side hints that the

time series is more likely to be stationary. Another useful technique to identify stationarity of time series is by looking at the Autocorrelation function plot. In stationary time series, the ACF plot would drop to zero in few lags, while in non-stationary time series it would maintain positive autocorrelations at a high number of lags and slowly decay to zero. Figure 5.4 illustrates an ACF plot with a very slow decaying, linear pattern of over 50 lags which is typical for a non-stationary series.

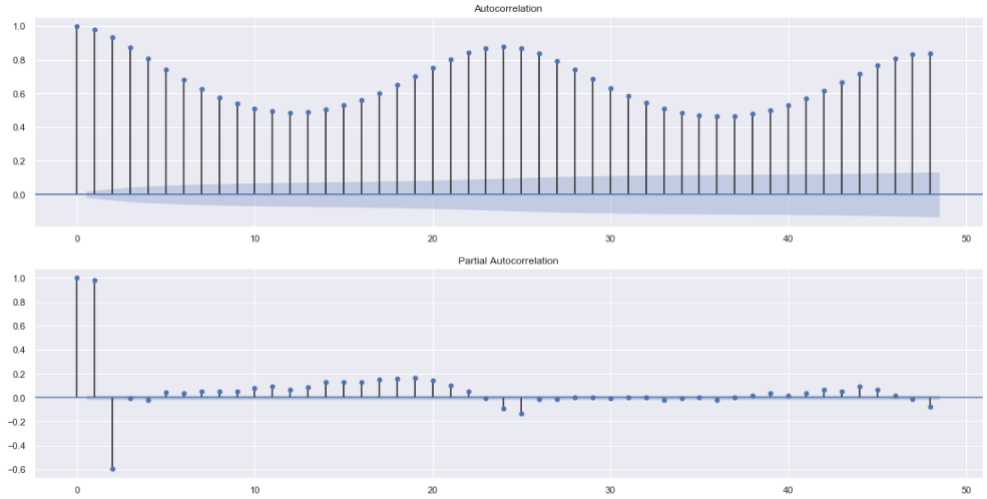

Figure 5.4: ACF and PACF plots of the original time series

Statistical tests

Two statistical tests are carried out to confirm the stationary properties based on the observations from the previous chapters. Augmented Dickey-Fuller(ADF) and Kwiatkowski-Phillips-Schmidt-Shin (KPSS) tests are widely used methods and can be both implemented using **statsmodels** python library.

ADF tests are ran whenever the series is afflicted with higher order autocorrelations, which needs to be removed from the residuals of the regressions. Under these conditions some sort of lagged differencing would be required to transforms the residuals into white noise disturbances.[14]
Running ADF on the original time series produces the following results:

```
Results of Dickey-Fuller Test:
Test Statistic                  -3.743619
p-value                          0.003540
#Lags Used                      24.000000
Number of Observations Used    9366.000000
Critical Value (1%)              -3.431048
Critical Value (5%)              -2.861849
Critical Value (10%)             -2.566934
dtype: float64
```

Figure 5.5: Augumented Dickey-Fuller test results on original time series

H_0: There is a unit root for the series.
H_A: There is no unit root for the series.

In the ADF test results, the probability value p at which the H_0 can be true, is less then any of the values at the significance level of 0.05. All the critical values at different significance levels are considerable greater than the test statistics value which concludes to reject the null hypothesis and accept the alternative hypothesis. The results indicate the time series has a unit root, which implies a systematic pattern in its stochastic trend, through it can be a random walk and cannot be predicted. The formal testing proceeds with KPSS test to further determine stationarity. Below are the results from the KPSS test:

```
KPSS Statistic: 8.509924593314565
p-value: 0.01
num lags: 38
Critial Values:
   10% : 0.347
   5% : 0.463
   2.5% : 0.574
   1% : 0.739
```

Figure 5.6: ACF plot of the original time series

H_0: The series is stationary.
H_A: The series is not stationary

It can be observed that the probability value at which the H_0 can be true, is less then the values at the significance level of 0.05. All the critical values at different significance levels are considerable smaller than the test statistics value. This concludes that the null hypothesis must be rejected in the favor of the alternative hypothesis.[15]

To summarize the results from the visual observations and statistical tests, various conclusions can be withdrawn. There are hints of inconsistency in the statistical properties of the data which implies the integrated part of the time series cannot be 0 due to the fact that only stationary series are said to be I(0). This leads to a conclusion that a lagged differencing to the time series is required to be applied in order to stabilize the trend of the time series, which will be discussed in the section. [16]

To remove the unit root and ensure a proper stationarity a data set transformation is required. The implementation implies to apply a first order difference algorithm, as explained earlier.(ARIMA model, formula for the first difference).

With the visualization and statistical stationarity tests run, the next requirement before moving to estimate the prediction model parameters, is to check the seasonality presence. As mentioned before, to conclude that data is stationarity, both trend and seasonality factors must be considered, therefore the data analysis process continues.

First step is to observe the current status of the data seasonality factor existence.

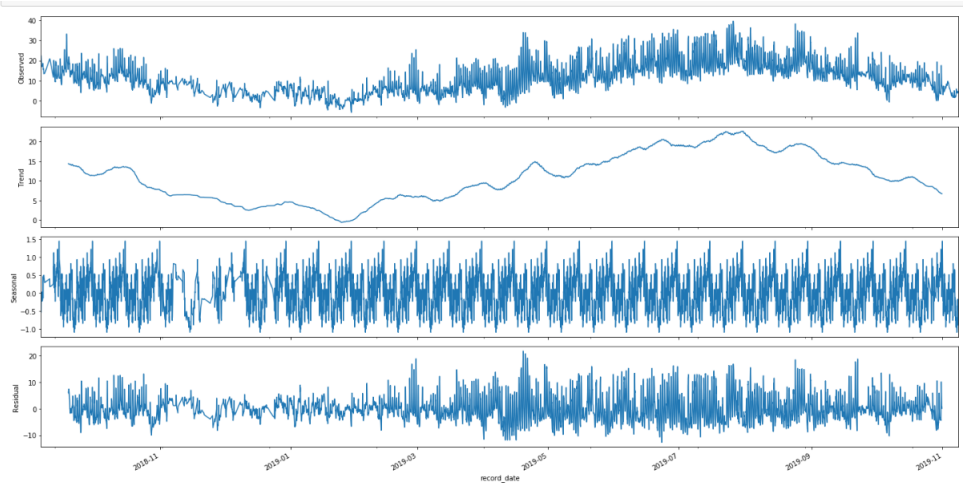

Figure 5.7: Current seasonality

The Figure 5.7 illustrates four different aspects of the data seasonality:

- First subplot shows the original time series between -35 degrees and 40 degrees across twelve months period starting from November 2018. The min and max interval was set in the chapter Data, section Data Preprocessing, therefore will follow along the whole process.
- Second subplot displays the presence of a trend in the data set. Following the knowledge regarding stationarity, the presence of a trend in the data relates to a potential non-stationarity. Since the period is set to exactly twelve months, a drop of the values is expected during the cold season, followed by a rise of the values in the warm season. This leads to ups and downs during the entire period, which results in a trend presence. The logic behind relates to generating a line of best fit for the associated observations, taking into account that the shape can change over time. For example, if the trend line were to be constant across the whole year, most likely a straight line would be shown regardless of a few large temperatures deviation across winter and summer. The trend will however be affected if the temperature deviation will occur too often.
- The third subplot illustrates the seasonal component reflected on the seasonal variance. This appears the most when a time series is affected by seasonal factors.
- The last component refers to the error or residual margin. A small residual is always the aim, since it represent what's left from the time series after trend and seasonal have been removed from the original data point. [10]

The observations of the above plots directs to the next move, which is seasonality decomposition. **Statsmodels** python library provides a decent approach to apply seasonal decomposition using moving averages, for both *additive* and *multiplicative*

models. Its a simplistic, straight forward procedure which assumes that the seasonal component is constant from year to year. The decomposition required for the purpose of this project, is for the additive model and consists in a series of steps where the trend-cycle(T) component is computed using the seasonal period(m) and the moving average(MA). Since the m in this specific case its an odd number, T is calculated by differencing the MA from m.

The actual method responsible with the seasonal decomposition is called **seasonal_decompose()** and it takes as parameters, the *time series, the model name* (in this case, additive) and *the frequency* which in this case is 365 as the number of days in one year time period.

The method responsible with transforming the data to a stationary time series including the seasonal component is called **diff()**, and it takes as parameters, the *time series*, the default number of differences, and the seasonal differences to perform and 24 for the seasonal differencing. [17] After applying non-seasonal and seasonal differencing, the ARIMA model assumes the initial following order *ARIMA(p, 1, q)(P, 1, Q)24*. Then reapplication of the previously done observations must be done on the differentiated time series.

Figure 5.8 shows a Gaussian distribution of the time series after first order differencing and seasonal differencing is applied. Compared to Figure 5.3, the distribution of values is more symmetrical, thus indicating stronger claim on yeilding stationarity.

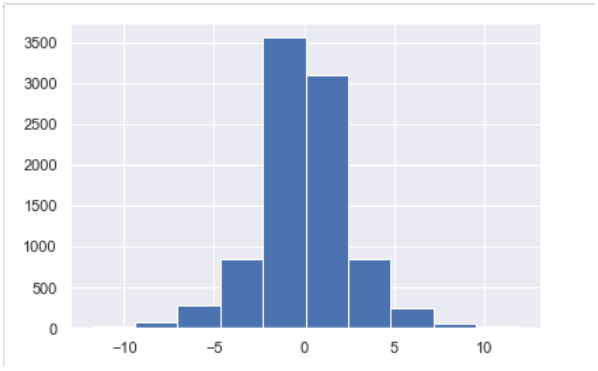

Figure 5.8: Gaussian distribution of differenced time series

Furthermore, looking at newly plotted, differenced correlograms at ref , significant change is observed compared to the original ones. By looking at these, it is possible to not only identify if the data set is stationarized but also whether it differenced properly and not overdifferenced. The ACF plots now shows fairly rapid decay towards zero, as well as introducing negative correlation which differencing tends to apply. It is a common mistake to overdifference. Overdifferencing can be observed when the lag - 1 spike (second spike) is negative beyond -0.5, which is not the case as displayed, therefore an overall differencing of two(normal and seasonal) suggests to be the right option. These observations can be followed by looking at the results extracted from the formal statistical tests described in the following section. ACF and PACF correlograms are also used when identifying the order of the rest of the parameters in ARIMA

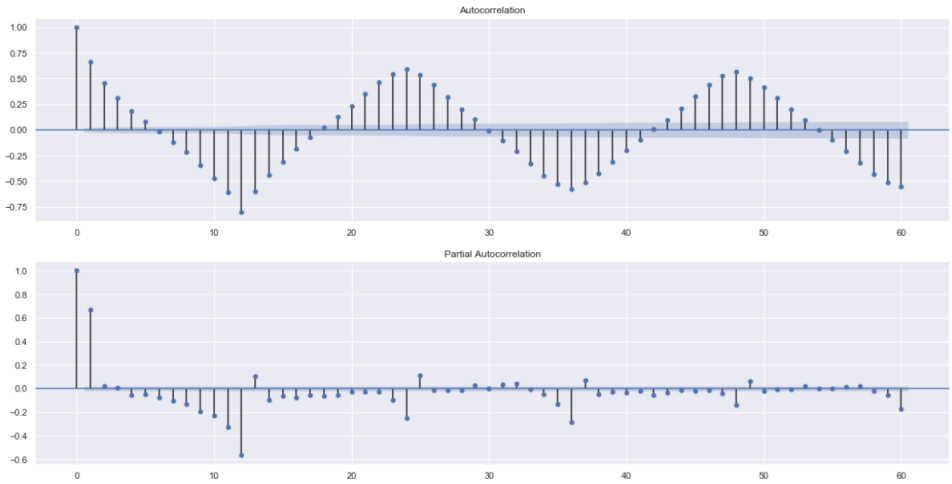

Figure 5.9: ACF and PACF after differencing

The ADF and KPSS tests results after transforming the data are now analysed. The expected outcome is fully stationary data set ready to deliver the parameters for the ARIMA model.

```
Results of Dickey-Fuller Test:
Test Statistic                -1.530128e+01
p-value                        4.309312e-28
#Lags Used                     3.800000e+01
Number of Observations Used    9.214000e+03
Critical Value (1%)           -3.431060e+00
Critical Value (5%)           -2.861854e+00
Critical Value (10%)          -2.566937e+00
```

Figure 5.10: ADF test results of differenced time series

The figure above reveals the expected outcome regarding the ADF stationarity test. The new values of the probability on which the null hypothesis is true remained smaller than 0.05, as well as the critical values still greater than then test statistics value. This concludes that ADF test results remained unchanged.

```
KPSS Statistic: 0.23492362787161225
p-value: 0.1
num lags: 38
Critial Values:
   10% : 0.347
   5% : 0.463
   2.5% : 0.574
   1% : 0.739
Result: The series is stationary
```

Figure 5.11: KPSS test results of differenced time series

The observations of the KPSS test results after applying differencing to the data set, changed. The[18]value is now as expected, as well as the critical values at the

significant levels. The process can now continue with observations of the seasonal component and the ARIMA model implementation.

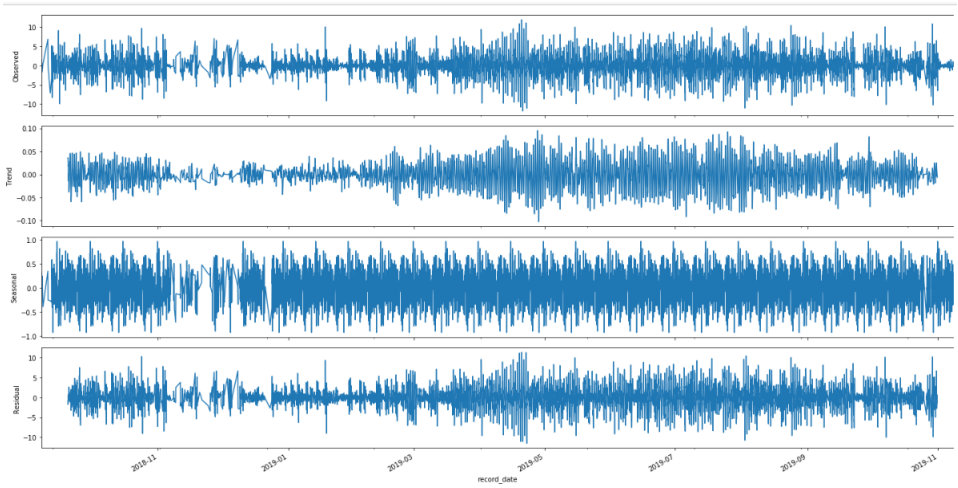

Figure 5.12: Seasonality removed plots

The Figure 5.12 illustrates how the four plots were affected when seasonality aspect was removed. Notice there is no trend present in any of the plots, the seasonal component is now the average of all the detrended values in the data. The seasonal component is obtained by stringing the values together(for example monthly values), then replicated the sequence. To be mentioned that this approach is rather simplistic, and comes with a few drawbacks, such as trend-cycle unavailable for the last few observations or that it assumes that the seasonal component repeats from one time period to another. All these were taken in consideration, of course, but since the time series consists of weather data, which does repeat yearly, and has an implicit seasonal trend in the warm versus cold period, it ensures the reliability required to proceed. [10]

Model Identification

The next step in the ARIMA modeling is identified by Box and Jenkins as model identification. Having determined the d parameter for the model, it is now time to determine the p and q parameters. If p = 0, then the process would be pure moving average process and if q = 0 then it would be a pure autoregressive process. As mentioned earlier, the autocorrelation functions ACF will be used together with the partial autocorrelation function PACF to identify the order of the autogressive and moving average parts of the model. To make this easier it is necessary to plot ACF and PACF graphical examples displaying lagged versions of the series. According to the theory described in [19] the process of estimation is rather theoretical and experimental and does not always turn out to be the most accurate. The general characteristics of the theoretical ACF and PACF for the AR and MA processes will be discussed ,in order to identify the terms. The p value is equal to the number of lags before the last spike in the PACF reaches zero. Looking at the PACF of the

differenced series shows a sharp cut off to zero at the third spike, therefore an AR of order two is considered to be optimal. The q value, on the other hand, is determined by the number of lags in the ACF before the last spike hits zero. Following this principle and observing the ACF of the differenced series, it is cutting off to zero in six lags. To conclude, the observations in this section suggest an ARIMA of order AR(3) ,I(1), MA(6).

Adding Data to Trained Model

An important aspect when comes to clarifying the logic behind the ARIMA model implementation is how future observations/input will be handled once the model has been built.

With the parameters set, the created model should not be modified, therefore no model refitting or retraining should take place. As "***All models are wrong**, but some are useful"*[20], of course tweaking the parameters and creating a new model should happen periodically, but not every time new data is incorporated as is the intention with this model and as will be explained in the next section. Creating or modifying the model would require a human process like the one that has been completed.

From the technical point of view of the ARIMA model implementation, "appending" new data to a built model is done using the method **append()**, which is provided by the **statsmodels** library. The method allows one to use new data when making predictions, without refitting or re-training the mode, therefore, keeping the model and its' coefficients intact. New observations can, therefore, be used to make new predictions, without modifying the model. Incorporating new predictions as time passes is important especially due to the MovingAverage portion of ARIMA.[18]

5.2 Implementation of Anomaly Detection

With the help of the ARIMA forecasting model described in chapter Application of Methods, this section will create a model that can be used to identify anomalies in our data set. The overall points of anomaly detection using time series forecasting follows the cycle:

- Create initial model based on training data and fit it to training data.
- Forecast forward in time.
- Compare predicted observations with actual at same point in time. If outside of confidence interval, it is anomalous.
- Add new data to existing model (only if deemed non-anomalous).
- Forecast forward in time.

Within this very general plan there are some specifics that need to be determined.

Forecast Window

One of the main ideas related to the addressed problem statement is to forecast only a very short period into the future, as accuracy naturally decreases the farther one forecasts. Therefore, it makes sense to forecast as short a period into the future as possible. In this case it is one hour into the future, as the frequency of the data set is set to hourly and consequently in forecasting this frequency must be matched. As mentioned in chapter ARIMA Basics and Parameters, ARIMA performs best on short-term, therefore it is safe to stick with a small forecast window.

Incorporating New Data

Another key point is how often to incorporate new data. Incorporating data as often as possible should theoretically produce the most accurate predictions, i.e. predicting for February 16 at 16:00 based on data up to and including February 16 at 15:00. A downside of this approach is that if the anomaly is of the kind that would more slowly reveal itself, i.e. the faulty actual observations staying barely inside the confidence interval over several readings, it would be difficult to detect. The rolling average part of the ARIMA model would smooth out these types of anomalies.

Confidence Interval

It is also important to decide the interval created around the predicted observation, that the actual observation needs to fall within in order to be non-anomalous. When forecasting observations, it is common to calculate a confidence interval for the predicted value. The ARIMA implementation used in this report automatically calculates confidence level upper and lower bounds for each predicted step and allows custom confidence levels.

This is the interval that will be used for determining which observations are anomalies. The most common confidence levels are 95% and 99%, meaning that there should be a 95% or 99% chance that a random sample of actual observations will have a combined mean within the confidence level interval. The higher the confidence level, the larger the interval will be. In this case it is decided choose a 95% confidence level. This is because it is often used and it provides high confidence, but not too high and letting too many anomalies slip by due to giving a very broad interval.

In Figure 5.13 the blue line contains the actual observations and the red point is the predicted value one hour ahead. The model incorporates data up until and including September 20 at 00:00, one hour before the prediction and the current actual observation that is being checked. The grey shaded area indicates the confidence interval for the prediction. In this example case the predicted value is very close to the actual and well within the confidence level interval.

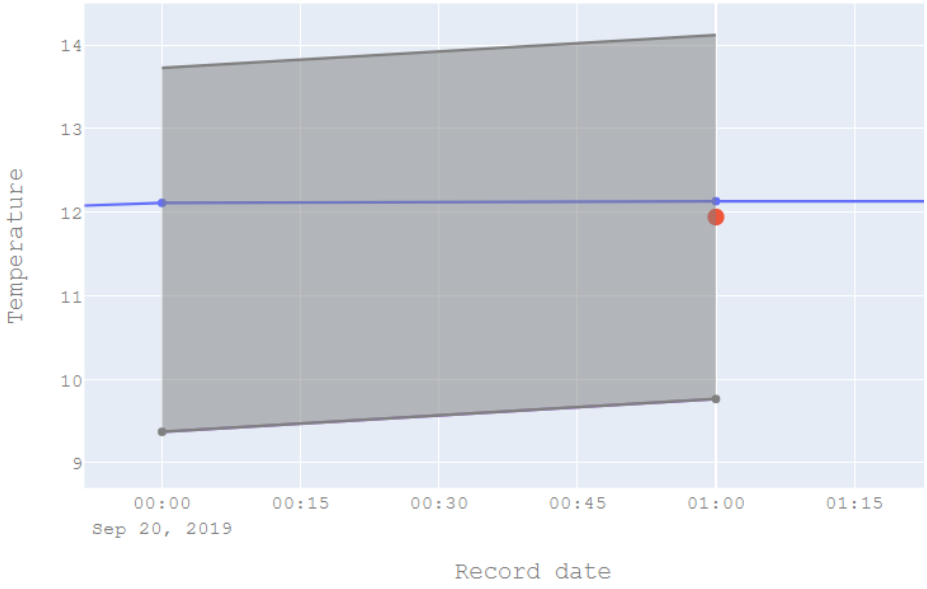

Figure 5.13: Example of Confidence Level

Discussion

Based on the details decided above, if the time is 14.00 and an observation for 14.00 just came in, the model would incorporate data up to and including 13.00. A prediction would then be made for 14.00. If the actual observation falls within the confidence interval of the predicted observation, it is not anomalous.

This would easily identify large spikes, so called point anomalies, as they would differ significantly from the prediction. Contextual anomalies would like-wise also be identified, due to the nature of using a prediction model based on the moving average. Temperature observations not fitting the context, i.e. far too warm or cold for the season would be outside the confidence interval for the predicted value.

Detecting collective-type anomalies would require additional logic. Logic would have to be created to look at the last x number of observations together to see if, for example, they had the exact same value. This is not really related to the model, but could be a separate component to be done at a later date.

5.3 Evaluation

5.3.1 Evaluation of the Arima Model

The third and final step from the Box-Jenkins method is evaluation of the model. A model's performance may be assessed through running a series of evaluation metrics. These metrics are used to compare the results of the different models for choosing the most adequate model. In the case of this project the goal is to find out the most accurate ARIMA order.

In order to access the relative performance of ARIMA models created on the same training data, but using different parameters, it is necessary to compare them by the use of AIC and BIC.

Models were created using the same and similar order parameters to those theorized earlier in the section "Implementation of ARIMA Forecasting". The **d** parameter was kept the same, as that relates to differencing of the data, which is the same in all cases.

Order(p,q,d)	AIC	BIC
(3, 1, 6)	24459.5917	24633.1567
(4, 1, 6)	25431.8587	25480.1859
(3, 1, 5)	26859.3922	26913.4844
(4, 1, 5)	28153.5811	28504.9590
(2, 1, 4)	26994.2921	27487.8501

Table 5.1: AIC and BIC Scores of Models

As can be seen in Figure 5.1, the order chosen earlier (3,1,6), produces the best model, measured by AIC and BIC, compared to these other similar orders.

Two primary metrics were considered to evaluate the prediction accuracy of the model:

Mean Absolute Error, (MAE) represents the average of all absolute errors. An Absolute Error or Absolute Accuracy Error is defined as the amount of error in a given measurement. It is the sum of absolute differences between the actual and predicted values. To compute MAE, the following formula must be taken in consideration

$$\mathrm{MAE} = \frac{1}{N}\sum_{i=1}^{N} |y_i - \hat{y}_i|$$

Figure 5.14: Formula for Mean Absolute Error [21]

The root mean squared error, (RMSE) error measures the accuracy in the results of predicted values and the actual values by depicting their squared differences. Compared to MAE, RMSE produces results with a higher weight to larger errors.

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

Figure 5.15: Formula for Root Mean Squared Error[21]

Both MAE and RMSE are negatively-oriented metrics, meaning the lower the results, the better the model is.[21]

The table on Figure 5.2 shows Mean Absolute Error and Root Squared Mean Error scores by inputing forecasted values of one hour ahead. Looking at the table, an ARIMA order of (3,1,6) produces lowest errors. It is noticeable that the MAE results are equal to the ones of RMSE for each sensor, respectively. This indicates that the errors from the evaluation have the same error magnitude. The cause of this phenomenon is presumed to be due to the fact of using smaller sample of values for forecasting.

Order(p,q,d)	MAE	RMSE
(3, 1, 6)	0.16978	0.16978
(4, 1, 6)	0.18375	0.18375
(3, 1, 5)	0.23129	0.23129
(4, 1, 5)	0.20104	0.20104
(2, 1, 4)	0.19663	0.19663

Table 5.2: MAE and RMSE Scores of Models - 1 hour forecast

Figure 5.3 shows similar results in terms of best fitted order for the forecasting model, just as 5.2. The successive test is performed on a largest test set by forecasting into the next six hours. The magnitude of the observed errors is increased, as well as RMSE scores now tend to be higher than those of MAE.

Order(p,q,d)	MAE	RMSE
(3, 1, 6)	0.459543	0.72627
(4, 1, 6)	0.475656	0.77616
(3, 1, 5)	0.506774	0.84419
(4, 1, 5)	0.480104	0.80194
(2, 1, 4)	0.453711	0.76453

Table 5.3: MAE and RMSE Scores of Models - 6 hour forecast

5.3.2 Conclusion

Based on the evaluation Root Mean Squared Root and Mean Absolute Error metrics run, it can be concluded that the ARIMA model with the order (3,1,6) that is in the current testing, the best suited to provide the best predictions.

5.3.3 Evaluation of Anomaly Detection

In the previous section, an ARIMA model was created to predict future temperature observations. In this section the anomaly detection model that has been created with the help of that ARIMA model is demonstrated and evaluated.

Demonstration of Anomaly Detection

On Figure 5.16 is an example of anomaly detection that was run on the training data. Anomaly detection was run on the twenty-four hour period September 6, 2019 until September 7, 2019. In this sample no anomalies were detected, all observed values were within the confidence interval.

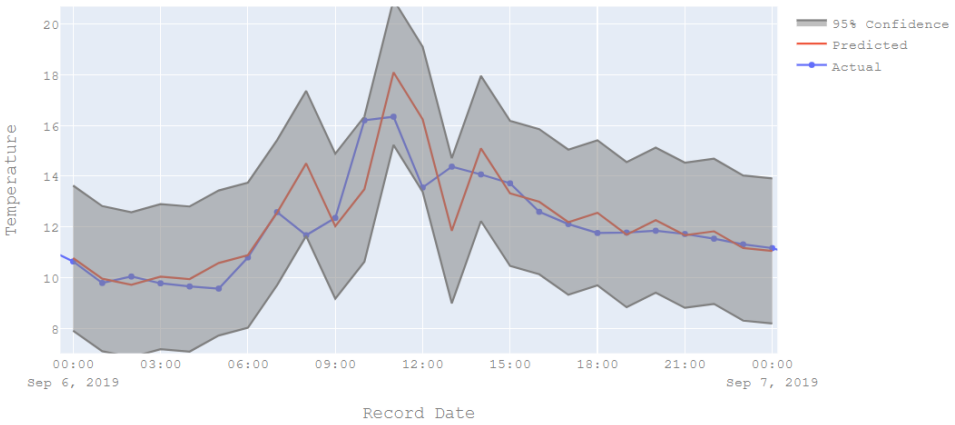

Figure 5.16: Example of Anomaly Detection

In order to evaluate the detection of anomalies with the model the test data is injected with anomalies.

The test series is injected with two point and contextual anomalies. One day of real-time anomaly detection was then simulated.

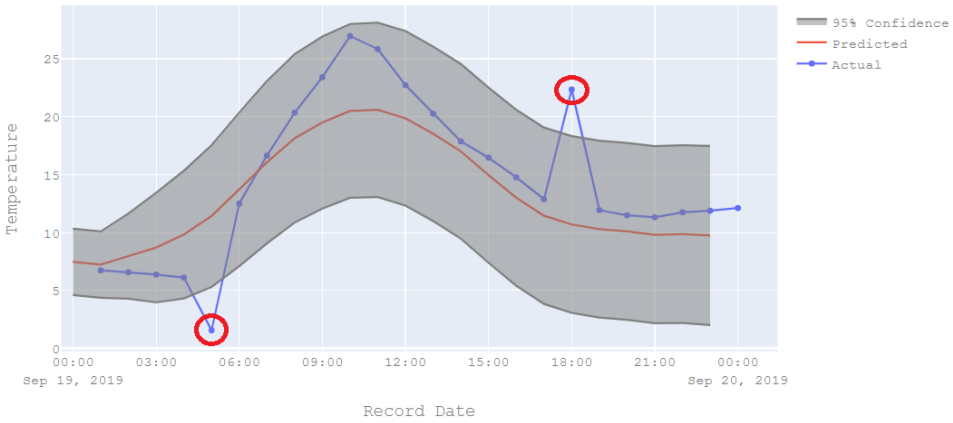

Figure 5.17: Example of Anomaly Detection

In Figure 5.17 anomaly detection was run on the twenty-four hour period September 19, 2019 until September 20, 2019. Anomalies that are both considered as contextual and point anomalies, were inserted into the period, which had been tested to be non-amomalous beforehand. The inserted anomalous observations are both uncharacteristic for their respective times of day, relative to the season, and are far from the rest of the data points.

The two anomalous observations that were inserted are circled in red on the figure. They are outside of the confidence interval and are considered anomalous. Because they were considered anomalous they were not included as new data for the next prediction at each step.

5.3.4 Evaluation Conclusion

Once a time series forecasting model has been produced, the method of detecting anomalies is quite simple. How good the anomaly detection is depends on how accurate the prediction model is. By working from theory and testing different ARIMA model parameters and comparing their performance metrics, it is believed that an accurate forecasting model has been produced. Although more testing should be done to compare test data and predicted data.

In order to properly access the performance, the produced anomaly detection model should in the future be compared to other models, both using time series forecasting, but also using other methods.

The combined anomaly detection model could also perhaps be tested programatically and on a larger scale, rather than individual small tests.

5.4 Tools and Libraries

Throughout the development of the project, various tools and libraries were put together in order to come up with the appropriate model that would create valid forecasts while using the data set available for this project.

5.4.1 Jupyter

The main development environment used for implementing the solution of the project is Jupyter, which is a so called web-based "notebook", in essence referring to the web application itself. There are many kernels compatible with Jupyter for a lot of programming languages, which handle different types of requests such as executing the code, completing the code and inspecting it as well. [22]

5.4.2 Python

By default, Jupyter comes pre-installed with the IPython kernel. This allows the use of the Python programming language which is the only one that has been chosen for this particular project. A good reason behind using it is the advantage of having many open source packages that allow for executing data analyses on various data sets. Another great advantage is the simple syntax that helps with improving the speed of the implementation. [23]

5.4.3 Anaconda

Python offers a great versatility in terms of the use cases that it could be used for and that is why for the purpose of utilizing it towards data analysis and machine learning development, there exists one distribution dedicated to managing the data science packages. It's called Anaconda, and it is also free and open-source. [24]

5.4.4 Pandas

A very useful library that was written for Python, is Pandas. Dataframes are used to form the data for machine learning models where multiple data structures and operations are included for importing, manipulating and cleaning the data. Having the ability to handle missing data tends to be very useful, as well as being able to merge and join different datas ets while also indexing them. [25]

5.4.5 NumPy

For the purpose of implementing mathematical functions, NumPy is the library that is meant to be used through Python. It has support for handling large multi-dimensional arrays because it was designed with the idea of having a higher speed for processing mathematical functions and operators. [26]

5.4.6 Matplotlib

As part of the visualization aspect of the analysed data, the library that is dedicated to displaying different types of plots and graphs, is called Matplotlib and it's a free and open-source library created for Python. [27]

5.4.7 Plotly

Another library allocated to visualizing the data is Plotly. This one provides interactive graphs that can be used with many kinds of statistics. [28]

5.4.8 Statsmodels

When it comes to how a data set could be explored in order to extract statistical models, Statsmodels is the Python package that focuses on that. A great ability that it has is that it can integrate with Pandas for handling the data and that it was also designed on top of NumPy and Matplotlib. [29]

Part VI

Conclusion

Chapter 6

Conclusion

6.1 Future Work

This section discusses possible scenarios of tackling the problem of detecting anomalous data from a different angle by e.g. proposing new ideas involving new methods or building up on the results achieved thus far. As part of the future work that could be done to further extend the project, a good approach would involve the expansion of the model to be able to handle multivariate forecasts, since this project is focused on detecting anomalies driven by a univariate time series forecast. The main difference would consist of avoiding to forecast the future values based only on the previous values from a given time, where basically only one variable varies over time, making it a one-dimensional value. The multivariate time series forecasting is using multiple variables that are varying over time, so the value that is predicted is made up from multiple dimensions. Other dimensions that could be taken into consideration can be the data from the sensors positioned in different locations. In this way, the results of the forecasting model should be able to reflect any type of anomalies within different features.
A potentially useful implementation of the multivariate approach would be the extension of the ARIMA forecasting model itself. There is an Autoregressive Integrated Moving Average with Explanatory Variable model named ARIMAX for short, where the explanatory variable would have the purpose of increasing the performance of the model and potentially leading towards a better anomaly detection. Some candidates for the variable could be the features that are related to each other in terms of climate conditions, but this would also imply that more research can be done in regards to the scientific proof of the influence of a feature over another one[30].

6.2 Conclusion

The main goal of the project was to identify anomalous data given a data set consisting of observations from twenty-four unique sensors placed all around Aalborg, Denmark.

Firstly, basic knowledge about the anomaly topic was acquired. Such as basic knowledge in the concept of Machine Learning, since there are various Machine Learning techniques that could have been applied in order to correctly address the project problem statement.

Next phase consisted of describing the data set and making decisions regarding what approaches must be used to use the most relevant and representative features to continue with. It was chosen to use just data from one sensor, sensor 16, and to work with just one feature, temperature. Some preprocessing was also done on the given data set, such as resampling and filtering out some anomalous data.

Another important aspect which had to be taken in consideration was regarding the most suitable method for anomaly detection. After looking at research on the subject, it was chosen to use an ARIMA forecasting model to perform anomaly detection. The idea is to compare predicted data points with actual data points.

Preceeding the implementation of the ARIMA forecasting model were a series of steps (e.g data visualization, transformation, parameter choices) that needed to be accessed. With all that in place, the ARIMA model was then trained, parameters validated and the forecasting performance was evaluated.

Finally the implementation of the anomaly detection itself was discussed. Some examples were demonstrated and it was concluded that the model is able to detect multiple types of anomalies. Anomaly detection performance would depend on the accuracy of the forecasts from the ARIMA model, that is theoretically sound.

Part VII

Bibliography

Bibliography

[1] VARUN CHANDOLA, ARINDAM BANERJEE and VIPIN KUMAR. Anomaly Detection : A Survey. http://cucis.ece.northwestern.edu/projects/DMS/publications/AnomalyDetection.pdf. *17/12-2019*.

[2] Alan K. Mackworth David L. Poole. *Artifical Intelligence Foundations of Computational Agents*, chapter 7. Cambridge University Press, 2017.

[3] Various Authors. Applied Time Series Analysis. https://newonlinecourses.science.psu.edu/stat510/lesson/5/5.1.

[4] Ratnadip Adhikari,R. K. Agrawal . An Introductory Study on Time Series Modeling and Forecasting. https://arxiv.org/ftp/arxiv/papers/1302/1302.6613.pdf.

[5] Montem A/S. Montem A/S - Homepage. https://www.montem.io/. *07/11-2019*.

[6] CISS. CISS — Center for Embedded Software Systems. https://www.ciss.dk/. *07/11-2019*.

[7] Various Authors. Pareto principle. https://en.wikipedia.org/wiki/Pareto_principle. *17/12-2019*.

[8] DMI. Temperaturen i Danmark. https://www.dmi.dk/klima/temaforside-klimaet-frem-til-i-dag/temperaturen-i-danmark/. *18/11-2019*.

[9] Moniz, Nuno and Branco, Paula and Torgo, Luís. Resampling strategies for imbalanced time series forecasting. *International Journal of Data Science and Analytics*, 10-05-2017.

[10] Athanasopoulos G.) (Hyndman, R.J. Forecasting: principles and practice, 2nd edition, OTexts: Melbourne, Australia. OTexts.com/fpp2. https://otexts.com/fpp2/MA.html. *2018*.

[11] NEDARC. Hypothesis Testing. https://www.nedarc.org/statisticalHelp/advancedStatisticalTopics/hypothesisTesting.html?fbclid=IwAR0tCLQud4He6jr_zNHJeTfhPpQguNBW7bFbeKCAGp0tCFq8BeYG8HNBzbY.

[12] Monnie McGee Robert Alan Yaffee. *Introduction to Time Series and Forecasting, Second Edition*. Springer, 2002.

[13] Robert Alan Yaffee, Monnie McGee. *An Introduction to Time Series Analysis and Forecasting: With Applications*. 2000.

[14] Richard A. Davis Peter J. Brockwell. *An Introduction to Time Series Analysis and Forecasting: With Applications*. Academic Press, 2000.

[15] Various Authors. Unit Root Tests. `https://faculty.washington.edu/ezivot/econ584/notes/unitroot.pdf`.

[16] G. S. Maddala. *Unit roots, cointegration, and structural change*. Cambridge University Press, 1998.

[17] Seabold, Skipper and Perktold, Josef. statsmodels: Econometric and statistical modeling with python, 2010.

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

[19] Alan Pankratz. *Forecasting with Univariate Box - Jenkins Models: Concepts and Cases*. 1983.

[20] GEORGE E. P. BOX . Short-Term Forecasting with ARIMA Time Series Models . `https://pdfs.semanticscholar.org/d628/ce0979f118fca4f7e7fc65e704885fd2378e.pdf`.

[21] Nick Pentreath. *Machine Learning with Spark*. 2015.

[22] Various authors . Jupyter. `https://jupyter.org`.

[23] Various authors . Python. `https://www.python.org/about/`.

[24] Various authors . Anaconda. `https://www.anaconda.com`.

[25] Various authors . Pandas. `https://pandas.pydata.org`.

[26] Various authors . NumPy. `https://numpy.org`.

[27] Various authors . Matplotlib. `https://matplotlib.org`.

[28] Various authors . plotly. `https://plot.ly`.

[29] Various authors . StatsModels. `http://www.statsmodels.org/stable/index.html`.

[30] Chaleampong Kongcharoen and Tapanee Kruangpradit . Autoregressive Integrated Moving Average with Explanatory Variable (ARIMAX) Model for Thailand Export. `https://forecasters.org/wp-content/uploads/gravity_forms/7-2a51b93047891f1ec3608bdbd77ca58d/2013/07/Kongcharoen_Chaleampong_ISF2013.pdf`, 23-06-2013.