Time Series Analysis in the Presence of Noise and Missing Data. Focus on: Water Quality System

Master Thesis

Fitore Muharemi

Frankfurt University of Applied Sciences

October 2018

ABSTRACT

Every day time series data are generated in large volumes and in different areas. However analysis and forecasting of time series still pose challenges due to noise and missing values. This thesis presents time series analysis using a water quality data set. Our aim is to present different techniques for time series analysis and anomaly detection for this specific data set. The contribution here is to solve water quality anomaly detection which can be part of an EDS for the public water company in Germany Thüringer Fernwasserversorgung. The difficulty of this problem is that it consists of highly imbalanced data set where events, the minority class, must be correctly detected based on water quality time series data set. We try to resolve this problem using machine learning algorithms. We test three different inference models including Adaptive Boosting, Random Forest, and Support Vector Machines(SVM). In this study, a comparison is made between changes in preprocessing, and classifiers, using the training and testing data. Finally, the paper identifies some weaknesses and discusses future research direction.

Keywords Water Quality · Anomaly Detection · Time Series · Supervised learning · AdaBoost · SVM · Random Forest

Doina Logofãtu, Frankfurt University of Applied Sciences, Supervisor
Christina Andersson, Frankfurt University of Applied Sciences, Assistant

Contents

List of Figures

List of Tables

1

INTRODUCTION

Efforts to understand and forecast the behaviors of different real life processes has led to a large amount of digital information in the world today. A lot of time-stamped data are being processed and measured daily. Only Google manages over 20 petabytes of user-generated data every day, and also there are billions of photos and videos uploaded each day on Facebook, Instagram, and YouTube. A billion of data are also registered by intelligent sensors which monitor different occurrences during the time, cell phone GSM-GPS-WIFI signals, financial records, medical devices. All these examples justify the ubiquity of time-stamped data, named also by Zicari [40]. The large amount of data that are being collected usually are in form of time-stamped data. The time-stamped data are known as time series. So time series is a collection of observations recorded sequentially on a regular or irregular interval. Time series analysis are becoming very important in the analysis of almost every domain. Weather forecasting is related

to time(season) and time series data are used to forecast next days weather. Energy consumption, air temperature measurements, traffic congestion, all are described and analyzed by time series data. A well known problem in our lives today is the urban water system quality. Today the awareness of the drinking water quality is vulnerable by man-made, any threat, earthquakes or terrorist attacks. Before the occurrence of the earthquake, groundwater chemistry has been found to change in numerous instances as a promising precursor of the earthquake. The change of the water chemistry by the earthquake is a good indicator of earthquake predictions, but in the other hand, these changes can be very harmful to human health as some of the changes have a toxic nature. Several earthquakes have demonstrated the disastrous effect that they have on urban water systems. The earthquake in Kobe city, Japan 1995, demonstrated the disaster the earthquake can cause to water systems [34]. In 2015, Nepal got the attention of the community due to its devastating earthquake of 7.8 magnitude. However, the interest in the scientific aspect seems very unsatisfactory compared to the interest in the socio-economic and cultural aspect of it. Due to highly unstable geology and frequent earthquakes, the region of Nepal can tell a lot for the per-earthquakes and post-earthquakes. So far, scientists have been able to find changes in several water quality parameters associated with the earthquake. The changes have been observed both before and after the occurrence of the earthquakes. Different researches have used different parameters to check the chemical changes in water such as pH[26], radon count rates, electrical conductivity, ion concentration[36]. We're not sure of whether or these anomalies are subtle and insignificant or significant and injurious. Earthquake can be attributed to numerous types of diseases that include infectious and parasitic diseases, Neoplasms, Disease of the blood and blood-forming organs Endocrine, nutritional and metabolic diseases, Mental and behavioral disorders, Diseases of the nervous system, Diseases of the respiratory system, Diseases of the skin and subcutaneous tissue, Diseases of the genitourinary system. Of these diseases, several are caused by chemical changes in groundwater such as changes in ion concentration and increase in turbidity[32]. Because of all these risks in public health the anomaly detection of water systems has a high importance. Another factor that can threaten human life by urban water is the terrorist attack. Terrorism is a major threat to water security, and recent attention has turned to the potential that these attacks have for disrupting urban water supplies. The chances that terrorists will strike

at water systems is real, as there is a long history of such attacks. Analysis and historical evidence suggest that massive causalities from the attacking water system are difficult to produce, although there may be some significant exceptions. There is a long history of using water as a political or military target or tool. The water systems are attractive targets because there is no substitute of water. The chances of terrorist attacks in water systems are often poorly understood by water companies. As a result, a control system with the attention to water quality is very important. Recent water security research efforts have focused on the advancement of methods for mitigation of the contamination threats to drinking water systems. A promising approach is a contamination warning system (CWS). This system is deployed and operates with online sensors, it has a rapid communication and can support early indications of water contamination. The online monitoring component of CWS is composed of multiple sensor stations which collect data continuously and transmit them to a central database, known as Supervisory Control and Data Acquisition (SCADA) database. In this infrastructure, we can use two types of different sensors: surrogate or direct. Direct sensors are used to detect a specific contamination on the water chemical values, while the surrogate sensors observe different parameters of the water quality; pH; chlorine; electrical conductivity; temperature, and detect the presence of any contaminants. It has been reported that water quality parameters tend to vary due to normal changes significantly. To distinguish between periods of normal and anomalous water quality variability from measures made with surrogate sensors it is needed an event detection system (EDS). This problem is discussed also by Clark et al. [10]. An example of CWS: Philadelphia Water Department (PWD) developed a warning system(CWS) for their drinking water system under a Water Security (WS) initiative grant from the U.S. Environmental Protection Agency (EPA). The CWS Dashboard serves as a centralized CWS event data management, incident investigation, and issue resolution platform for the CWS project. The CWS Dashboard provides automatic event detection, the opportunity for preliminary cross-component data analysis and map display for quick response and coordination among a large group of users, including PWD, Philadelphia Department of Public Health (PDPH), and other agencies[30]. Typically, an EDS reads in SCADA system (e.g., water quality signals and operations data), performs an analysis in near real time, and then returns the calculated probability of a water quality event occurring at the current time step. A water quality event is defined as a

time period over which water with anomalous characteristics is detected. The working definition of ”anomalous” can be set by the user by selecting configuration parameters that govern the sensitivity and operation of the EDS. The values of these configuration parameters will vary from one utility to the next and can even vary across monitoring stations within a single utility. Investment in automated EDS allows online monitoring and change detection in water quality data, also historical data for recurring patterns and trends. Event detection from time series is a research topic in many different fields, including weather forecast, earthquake prediction, system fault detection, anomaly signal detection, and data mining. A number of experiments on water quality parameters conducted in the laboratory have concluded that water quality parameters values change rapidly and significantly in the presence of every contamination [8]. In our daily life, we need an EDS to automatically distinguish changes due to the presence of contaminants or any changes due to normal variability. EDS can work on two different approaches, online event detection or offline event detection. The online EDS uses any tool like artificial neural network, deep neural network, SVM, it predicts a future water quality value based on recently observed values. The predicted and the observed values are compared, and the residual between the prediction and the observation is classified and a then it determines if the water quality at that time step is anomalous or not [4]. Offline approaches work by using a pre-recorded data set, and the goal here is to detect change points. The change point is the time where an abrupt change signal occurs.

This thesis aims to present different techniques for time series analysis for a specific data set. The contribution here is to solve water quality anomaly detection which can be part of an EDS for the public water company in Germany Thüringer Fernwasserversorgung[1]. We build and analyze different machine learning models for the purpose of gaining minimal false positives and negatives.

[1]The Thüringer Fernwasserversorgung, located at the heart of Germany, is a public water company with its headquarters in Erfurt. Thüringer Fernwasserversorgung operates more than 60 dams and reservoirs, 2 central water treatment plants and 550 km of bulk water transport network. With about 200 employees Thüringer Fernwasserversorgung transfers more than 50 million cubic meters of raw water and drinking water to its clients, local and municipal water supply companies, thus ensuring a reliable supply of highest quality drinking water to more than 1 million people.

1.1 Goal of the Thesis

The goal of this thesis is to analyze different methods for a specific time series data set, as different methods can perform differently for different data sets. The principal objective of this work is not to find generally a good model for all kinds of data, but to concentrate on the water-quality data set collected by Thüringer Fernwasserversorgung public water company, and used also in the GECCO IC 2018 competition [31]. We want to try out different models on this data set and find an optimal one.
During the thesis we try to answer the following questions:

- What are time series and what do they differ from other data
- What are necessary steps on working with time series?
- How can machine learning help anomaly detection in time series?
- How to improve forecasting accuracy?
- How to test time series models?

In this work for statistical analysis, Python was used, which is a free software environment for statistical computing and graphics.

1.2 Chapter Overview

The thesis contents are broadly grouped into 8 chapters. Chapter 1, provides an introduction and background on urban water systems and event detection systems. Chapter 2 describes the related work in this area. The next chapter, 3, provides a technical overview on time series data sets and their use. What was done before for the same problem? What were the challenges? Also describes some real-life situations were accurate forecasting can make a big difference. Chapter 4 represents the description of the data we use on this thesis. Chapter 5, details preprocessing choices for time series data and what we have used. Classification models along with features and classifier selection are described in chapter 6. Chapter 7 contains dataset setup and performance results. Section 8 concludes the discussion and outline the future work.

2

RELATED WORK

In this chapter, we describe work that is closely related to ours and an overview of other works which are related to the big picture of water quality and monitoring systems.

2.1 Anomaly detection in water quality

Water quality has been subject to increased attention in the past decade. Thus, numerous studies attempt to tackle different aspects of anomaly detection on drinking water quality with different approaches. A comprehensive work on this topic has been initiated by U.S. EPA resulting in the CANARY tool [16]. CANARY is a software which is designed for online water quality event detection. The software can directly reads data from sensors and considers historical data to detect events. The best of the CANARY software is that it provides both real-time, and off-line analysis

tools to aid in the development of the algorithms, and such it allows algorithm developers to focus on the algorithms performance, rather than on how to read in data and drive the algorithms. Event detection is performed in two parallel phases: the first phase is called state estimation and second phase is residual computation and classification. In first phase, history is combined with new data to generate the estimated sensor values that will be compared with actual measured data. In the second phase, the differences between the estimated values and the new measured values are computed and the highest difference is checked using a threshold. If the highest value exceeds the threshold, it is declared as an outlier. The number of outliers in the recent past are then combined by a binomial distribution to compute the probability of an event in the current time step.

Byer and Carlson [8] are among the first to create and test an online monitoring of drinking water distribution systems. They added four crediblethreat drinking water contaminants (aldicarb, sodium arsenate, sodium cyanide, and sodium fluoroacetate) to a tap water and analyzed at different concentrations to determine their detectability in a distribution system. Benchtop analysis and online monitoring equipment were used to measure pH, chlorine residual, turbidity, and total organic carbon values before and after the introduction of these contaminants. Results indicate that all four contaminants can be detected at relatively low concentrations. Three of the four contaminants were detected below a concentration that would cause significant health effects[8].

Zhang et al. [38] propose a novel anomaly detection algorithm for water quality data using dual time-moving windows, which can identify anomaly data from historical patterns in real-time. The algorithm is based on statistical models, autoregressive linear combination model. They have tested the algorithm using 3-month water quality data of PH from a real water quality monitoring station in a river system. Experimental results show that their algorithms can significantly decrease the rate of false positive and has better anomaly detection performance than AD and ADAM algorithms.

Most of the works of water quality anomaly detection focus on using statistical algorithms. Our work consists of building machine learning classifiers with the aim of outperforming statistical approaches, so decreasing number of false alarms.

3

TIME SERIES THEORY

Today's the collection of a large amount of data is present in both the public and the private sectors. Data is what we need in order to perform better analysis and make a better decision. The collected data that have been observed at different points in time are known as time series data.

In this chapter, we present the theoretical background of time series and discuss the current state of time series research.

3.1 Basics of Time Series

A time series is a sequence of data observed at a successive equally spaced points in time. In time series is important the order of samples, as there is correlation by the samples of adjacent points in time [33]. Time series data are present in almost all fields. In economics,

observing the daily stock market. In medicine, blood pressure measurements traced over time In hydrology, monitoring the quality of drinking water. Weather forecasting, signal processing, pattern recognition, mathematical finance, weather forecasting, earthquake prediction, etc. Two approaches to time series analysis exist, the time domain approach and the frequency domain approach. The time domain approach views the investigation of lagged relationships as most important (e.g., how does what happened today affect what will happen tomorrow), whereas the frequency domain approach views the investigation of cycles as most important (e.g., what is the economic cycle through periods of expansion and recession). Time series can be represented as a set of observations X_t, each one being recorded at a specific time t and it is written as: $X_1, X_2, X_3, ..., X_t$ or X_t, where $t = 1, 2, 3, ...t$. Most often, the observations are made at regular time intervals. Time series analysis accounts for the fact that data points taken over time may have an internal structure, such as autocorrelation, trend or seasonal variation. Time series analysis is the analysis of a series of data points over time, allowing one to answer the question such as what is the causal effect on a variable Y of a change in variables $X_1, X_2, ...X_t$ over time. the most important distinction between time series data and cross-section data is that the ordering of observations matters in time series. When investigating a time series, it is generally useful to start with graphical representations to detect the properties of the series which can be seen by simply looking at the plot of a time series.

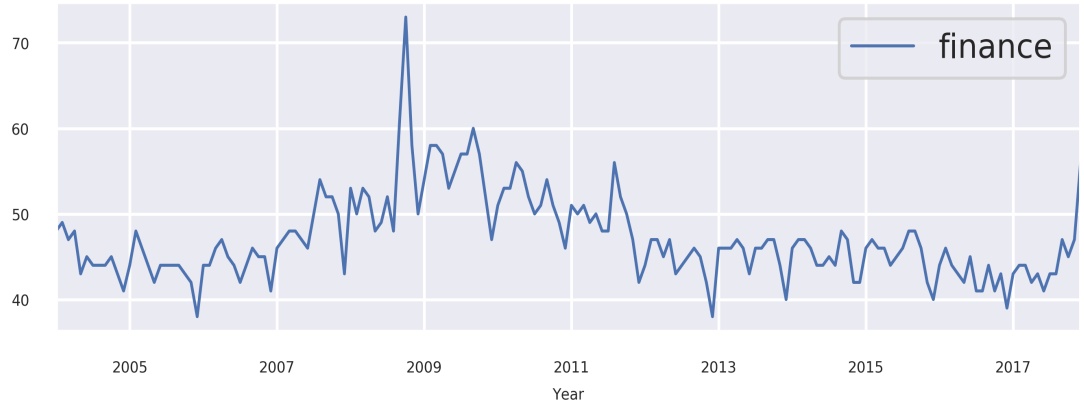


Figure 3.1: Time series finance change during the years in a company

Figure 3.1 shows the graph of the time series data from the finance sector. In the long run, we

do not see any trend, but there are some seasons on short-run movements which take place within one year.

Warren M.Persons(1919) distinguished four different components on time series:

- a long-run development, the **trend**,
- a **cyclical** component with periods of more than one year
- a component that contains the ups and downs within a year, the **seasonal** cycle, and
- a component that contains all movements which neither belong to the trend nor to the business cycle, nor to the seasonal component, the **residual**.

According to Diggle [12], one simple method of describing a series is that of classical decomposition. Time series can be decomposed into four elements: Trend, Seasonal effects, Cycles, Residuals. The idea of time series decomposition in to make easier the analysis and forecasting. It is much easier to forecast the individual regular patterns produced through decomposition of time series than the actual series. The general mathematical representation of the decomposition approach is:

$$Y_t = f(T_t, S_t, E_t) \tag{3.1}$$

where : Y_t is the time series values at period t; T_t is a deterministic trend-cycle or general movement component; S_t is a deterministic seasonal component; E_t is the irregular(residual) component. In the common approach it is assumed that the equation has an additive form:

$$Y_t = T_t + S_t + E_t \tag{3.2}$$

Trend, seasonal and irregular components are simply added together on the observed time series. Alternatively, the multiplicative decomposition has the form:

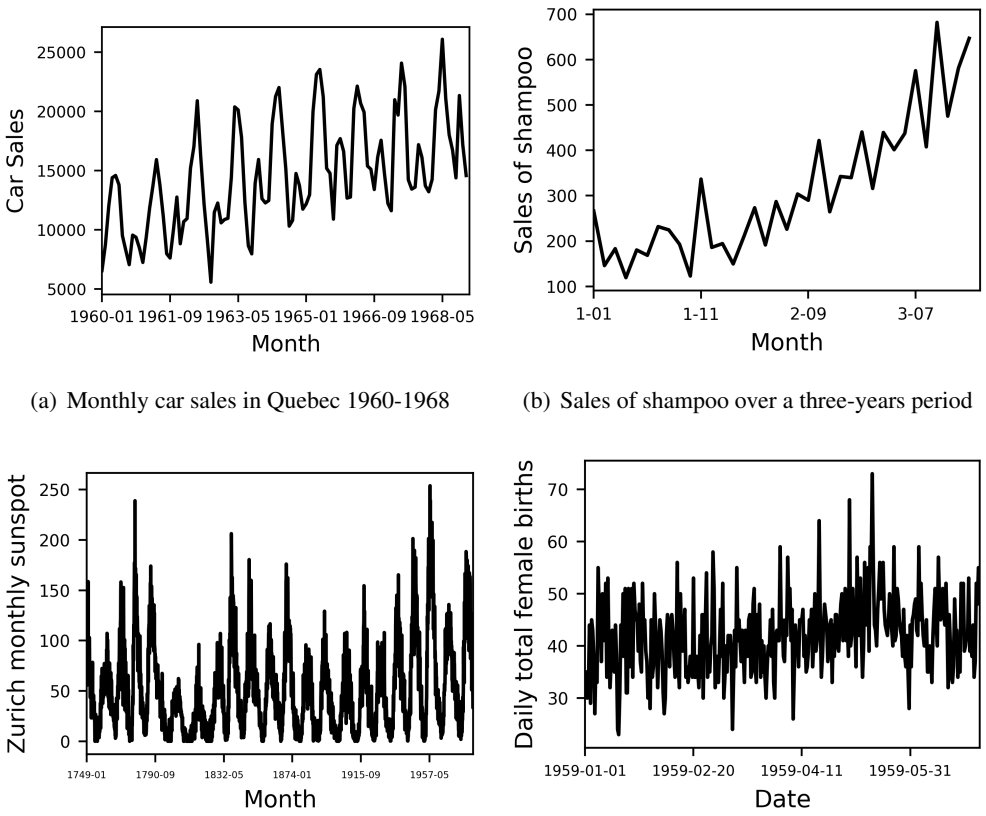
$$Y_t = T_t \times S_t \times E_t \tag{3.3}$$

Time series data have a natural temporal ordering. They are generally written in a predefined order and they differ from typical machine learning applications where each data point is an

independent example of the concept to be learned. Time series data are usually manipulated as one single object i.e. as a collection of data. This is because the order in which the data occurs in time series is very important because, unlike other data types, the ordering often represents the dependencies between the collected data. Consequently, manipulation of time series data puts more emphasis on aggregation operations on collections of data rather than on an individual data item [23] In time series analysis, the past behavior of a variable is analyzed in order to predict its future behavior. By observing the different past states of a variable, the future states can be predicted through forecasting

3.2 Univariate Time Series

Univariate time series refers to a series that consists of single observations recorded sequentially over equal time increments. When we model a univariate time series, we are modeling time series changes that represent changes in a single variable over time. Sometimes it is enough to only have one collected variable to predict an occurrence, but there is some phenomenon which is very hard to predict, like earthquakes, tornadoes, weather. In hard to predict a phenomenon, we usually collect more than one variable, analyze their relationship and the impact of each of them on the occurrence of that event. Examples of univariate time series are made visible in the figure 3.2. Figure 3.2(a) describes the number of car sales in Quebec, Canada between 1960 and 1968. We can notice that this data set contains the season and the tread pattern. Figure 3.2(b) describes the monthly number of sales of shampoo over a 3 year period. The units are a sales count and there are 36 observations. They appear to fluctuate erratically about a slowly changing level. There is a clear trend in the plot but no identifiable seasonality. The original data set is credited to Makridakis et al. [25]. A monthly count of the number of observed sunspots for just over 230 years (1749-1983) is provided by Andrews and Herzberg [3], and this univariate time series is described in the figure 3.2(c). It is clear that this data set has no trend but it looks like there is a long-run season pattern. The figure 3.2(d) shows one year of daily female birth. The data set for the number of daily female births in California in 1959 is credited to Newton (1988). In this data set, there is no trend and no seasonal pattern.



(a) Monthly car sales in Quebec 1960-1968

(b) Sales of shampoo over a three-years period

(c) Zurich monthly sunspot numbers 1749-1983

(d) Daily total female births in California, 1959

Figure 3.2: Examples of univariate time series from real life data

3.3 Multivariate Time Series

Much of the theory of univariate time series extends in a natural way to the multivariate case; however, new problems arise to multivariate data analysis. Many time series in practice are best considered as components of some vector-values time series. A time series dataset is called multivariate where a sequence of measurements from the multiple variables are collected. Over the past decade, multivariate time series have received a significant interest in many different areas of work, anomaly detection of telecommunication signals up to drinking water quality.

The first question we might ask for the multivariate time series data is whether all the variables are needed to construct a good prediction model. If any of the variables can be exactly predicted from the others, so there is an dependence, then this variable is not needed to be used on modeling.

It is important to identify the relations among different variables. It saves time on not measuring x-variables in the future, instead, we can estimate it from the relation of the measured ones. Another application of dependent variables is in control engineering, where one might want to minimize the number of control variables (Cao et al. [9]). Figure 3.3 shows the plot of air quality dataset that reports on the weather and the level of pollution each hour for five years at the US embassy in Beijing, China. The data includes the date-time, the pollution called PM2.5 concentration, and the weather information including dew point, temperature, pressure, wind direction, wind speed and the cumulative number of hours of snow and rain.

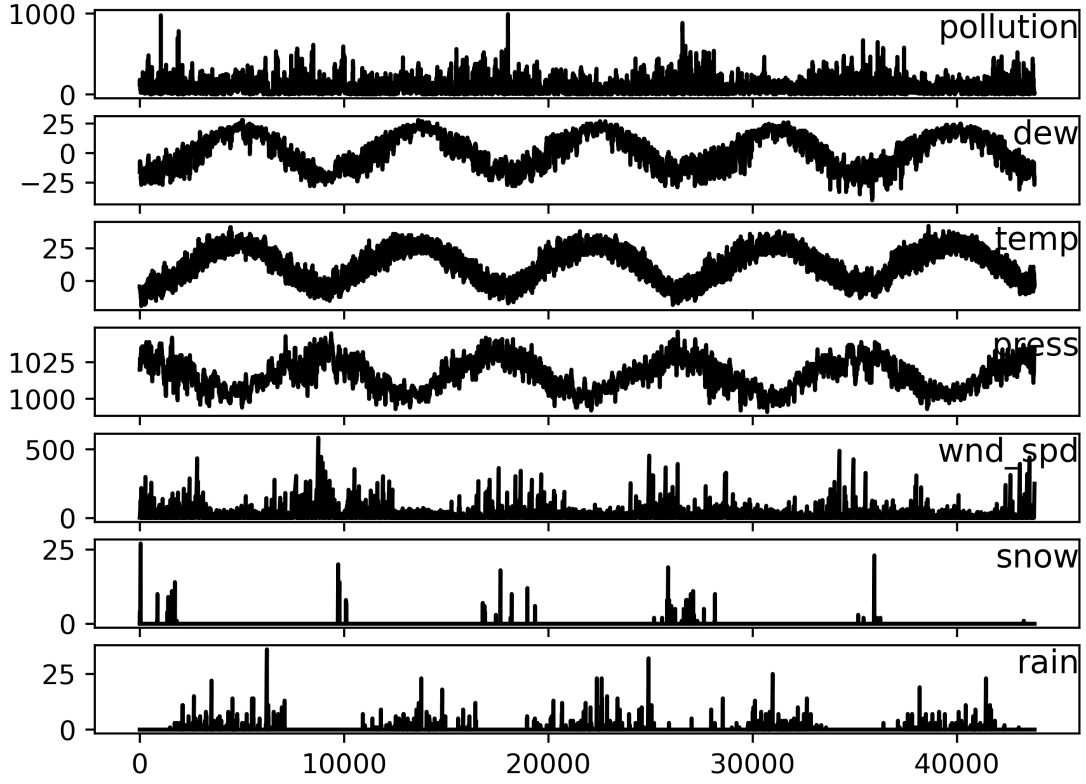


Figure 3.3: Air quality dataset- a multivariate time series from real life data

3.4 Stationarity and Non-Stationarity of Time Series

"Experience with real-world data, however, soon convinces one that both stationarity and Gaussianity are fairy tales invented for the amusement of undergraduates."(Thomson 1994)

This quotation bears us in mind that in time series analysis the basic building block is the purely random process. A purely random process is the process $\epsilon_{t=-\infty}^{\infty}$, where each element ϵ_t is independent of every other elements, ϵ_s for $s \neq t$ and each element has an identical distribution [28].

By definition it is immediate that the mean and the variance of the above purely random process are

$$E(\epsilon_t) = \mu \quad and \quad var(\epsilon_t) = \sigma^2 \tag{3.4}$$

In time series the key aspect is how observations are related to each other in time. The autocovariance formalizes the concept between elements which measured the degree of second-order variation between two elements at two different times. The autocovariance between X_t and X_s for some process X_t is defined as:

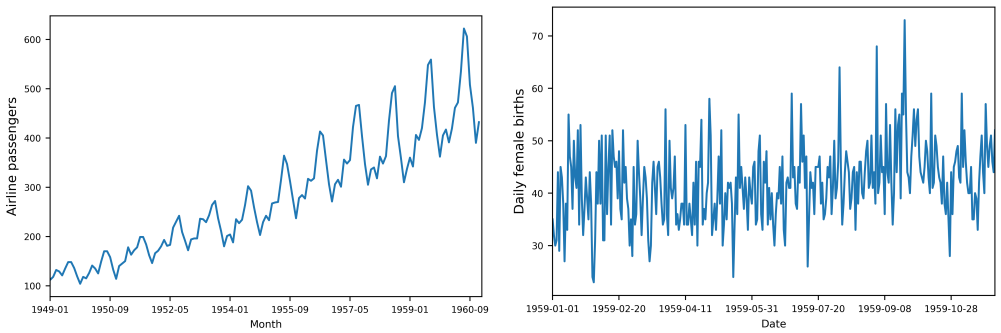
$$cov(X_T, X_s) = E[X_t - E(X_t)X_s - E(X_s] \tag{3.5}$$

A stationary process is one whose statistical properties do not change over time. To define it more formally; a stationary process is one where given $t_1, t_2, ..., t_n$ the joint statistical distribution of $X_{t_1}, X_{t_2}, ..., X_{t_n}$ for all n. From this definition, we understand that all moments(expectations, variances) of the process, anywhere are the same. On the daily life processes, we say that a stationary time series is one whose properties do not depend on the time at which the series is observed. Thus, time series with trends, or with seasonality, are not stationary the trend and seasonality will affect the value of the time series at different times. On the other hand, a white noise series is stationary, it does not matter when you observe it, it should look much the same at any point in time [22].

We must distinguish that some time series with cyclic behavior but with no trend or seasonality are stationary. This is because the cycles are not of a fixed length, so before we observe the series

we cannot be sure where the peaks and troughs of the cycles will be. In general, a stationary time series will have no predictable patterns in the long-term. Time plots will show the series to be roughly horizontal (although some cyclic behavior is possible), with constant variance.

In realistic processes often the time series are non-stationary. So time series have means, variances, and covariances that change over time. Non-stationary behaviors can be trends, cycles, seasons or combinations of the three. Non-stationary data, as a rule, are hardly predictable and normally difficult to be modeled or forecasted. The results obtained by using non-stationary time series may be spurious in that they may indicate a relationship between two variables where one does not exist. In order to receive consistent, reliable results, the non-stationary data needs to be transformed into stationary data. In contrast to the non-stationary process that has a variable variance and a mean that does not remain near, or returns to a long-run mean over time, the stationary process reverts around a constant long-term mean and has a constant variance independent of time. Figure 3.4(a) gives the plot of a stationary time series from real life, while



(a) Nonstationary time series- Airline passengers (b) Stationary time series- daily female birth

Figure 3.4: Examples of stationary and non-stationary time series from real life data

figure 3.4(b) shows the plot of a non-stationary process.

3.4.1 Check for stationarity

There are many methods to check whether a time series is stationary or not. By only looking at the plots we can review the data visually if there is an obvious trend or seasonality. Another method is by summary statistics, where we check whether the data of different seasons have

obvious differences. Also, there are many test statistics to determine the stationarity.

Augmented Dickey-Fuller Test(ADF) tests the null hypothesis that a unit root is present in a time series data. Normally, this test statistic is used under the assumption that the time series stems from some autoregressive process of order *p*. Hence, the ADF test the following regression model:

$$\Delta y_t = \alpha + \delta y_{t-1} + \sum_{i=1}^{p} \Delta\beta_p y_{t-p} + \epsilon_t \tag{3.6}$$

where Δ is the difference operator, α is a constant, and δ is the autoregressive lag coefficient. The ADF tests the hypothesis:

$$H_0 : \gamma = 0 \quad \text{, the time series is non-stationary, against}$$
$$H_1 : \gamma \leq 0 \quad \text{, the time series is stationary}$$

If we assume an AR(2) model for our time series we would regress $\Delta y_t = \alpha + \beta\mu + \gamma y_{t_1} + \Delta\delta y_{t-1}$ and then perform the ADF test. After that a comparison is done between this value and ADF test statistic.

$$DF = \frac{\hat{\delta}}{\sqrt{\text{Var}[\hat{\delta}]}} \tag{3.7}$$

The conclusion derives from: if the test statistic is smaller than the critical value, the hypothesis is rejected. Otherwise, if the null is rejected this would imply that there is no information gained from the inclusion of y_{t-1} into the prediction of variable y_t. And it means that the series is stationary, and no further transformations of the data are required.

4

DATA DESCRIPTION

Historical data available from different monitoring stations within the same distribution network are available for this analysis. For the monitoring of the water quality, the Thüringer Fernwasserversorgun performs measurements at significant points throughout the whole water distribution system, in particular at the outflow of the waterworks and the in- and outflow of the water towers. For this purpose, a part of the water is bypassed through a sensor system where the most important water quality indicators are measured. The data we use in this work have been measured at different stations near the outflow of a waterworks and have been provided for the GECCO Industrial Challenge [31]. In this chapter, the data set will be described.

4.1 Description of the water quality data set

The data set we use in this work is time series, and it is composed of 139566 registered samples with six water quality indicators, three operational data attributes and the label which indicates if there is an event or not. The data are registered on a minute basis and are used to identify the parameter settings in the event detection algorithms. These time series start at 03/AUG/2016 followed with other 139565 values and end at 08/NOV/2016.

Table I provides the water quality indicators measured for this data set. The chlorine dioxide, the pH value, the redox potential, the electric conductivity and the turbidity of the water provide an indication for any changes on the water (event), while the flow rate and the temperature are considered as operational data, changes in these values may indicate changes in the related quality values but are not considered as events themselves.

Name	Description
Time	Time of measurement, given in following format: yyyy-mm-dd HH:MM:SS
Tp	The temperature of the water, given in C.
Cl	Amount of chlorine dioxide in the water, given in mg/L (MS1)
pH	PH value of the water
Redox	Redox potential, given in mV
Leit	Electric conductivity of the water, given in μS/cm
Trueb	Turbidity of the water, given in NT
Cl_2	Amount of chlorine dioxide in the water, given in mg/L (MS2)
Fm	Flow rate at water line 1, given in m^3/h
Fm_2	Flow rate at water line 2, given in m^3/h
EVENT	remarkable change, given in boolean.

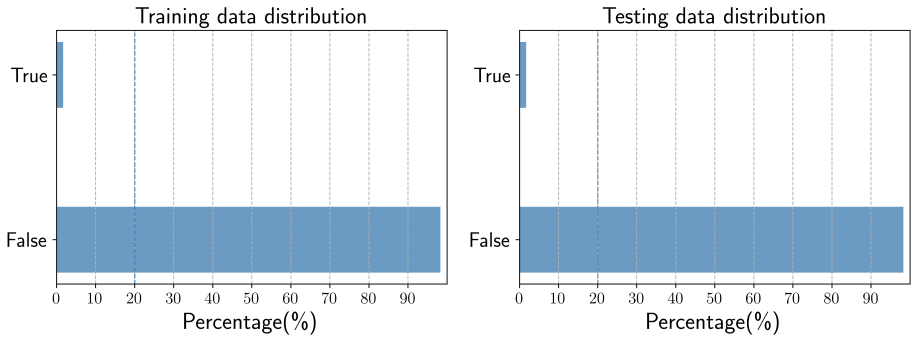
Table I: Description of the given time series data

Real drinking-water time series are provided for training, testing, and assessing event detection methods. Same as for training, the testing set has three months collecting data. The period of the collection is different for the training and testing, from August-November and November-February respectively.

4.2 Distribution of Data

Looking at real, uncleaned data one of the first things we notice is that it's a lot noisier and imbalanced. The data we use on this work belong to rare events data, binary dependent variables with dozens to thousands of times fewer ones (events, cases of fraudulent use, or epidemiological infections) than zeros ("nonevents") [19]. According to King and Zeng [19], these variables have proven difficult to explain and predict, a problem that seems to have at least two sources.

Research on imbalanced classes often considers imbalanced to mean a minority class of 10% to 20%, but in reality, datasets can get far more imbalanced than this. For example, factory production defect rates typically run about 0.1%; about 2% of credit card accounts are defrauded per year; medical screening for a condition is usually performed on a large population of people without the condition, to detect a small minority with it (e.g., HIV prevalence in the USA is around 0.4%)[1]. The training and testing data we use are highly unbalanced, with less than 2% of true events, and in such a problem maximizing the accuracy is meaningless if we assume that the rare class examples are much more important to classify, which is true for the problem we try to solve. Figure 4.1 shows the distribution of training (4.1(a)) and testing (4.1(b)) data based on event detected in drinking water quality.



(a) Training dataset (b) Testing dataset

Figure 4.1: Distribution of the collected data based on event

[1] https://www.svds.com/learning-imbalanced-classes/

4.3 Features description

Temperature has a direct influence on water quality. According to Kumar et al. [21], temperature impacts both the chemical and biological characteristics of surface water. It affects the dissolved oxygen level in the water, photosynthesis of aquatic plants, metabolic rates of aquatic organisms, and the sensitivity of these organisms to pollution, parasites, and disease. Temperature changes by even a few degrees could indicate a source of unnatural warming of the water or thermal pollution. Thermal pollution caused by human activities is one factor that can affect water temperature.

Chlorine dioxide is a powerful oxidizing agent and disinfectant. It is used to destroy microorganisms which can produce disease such as cholera, typhoid and so on. Chlorine is very reactive and will only remain in treated waters of high quality. It is important for public water supply and it gives a high indication to determine contaminated water. Low levels are effective for disinfection in normal circumstances but a higher chlorine input may be needed to achieve a given degree of protection. According to [1], if the water is polluted by phenols or by trace organic compounds released from decaying algal growths, chlorination can give rise to very severe taste and odor problems, rendering the water unfit to drink. It should also be noted about chlorination that where a water contains even small amounts of organic (humic) coloring matter, the reaction between it and the added chlorine will give rise to undesirable chlorinated by-products [e.g. trichloroethane; q.v.] which are also subject to restriction.

pH is the negative logarithm of the hydrogen ion concentration of a solution. Its measurement gives us the indication of whether the liquid is alkaline or acid. pH can be affected by chemicals in the water and it is an important indicator of water that is changing chemically. Rocks, soil, coral, and forms of organic debris will affect the pH level as the fluid washes over them and releases various minerals into the environment. These minerals, such as calcium and sulfide minerals, transform into organic acids and alkalines that can cause the pH level to change. For the same reason, dust and other small airborne contaminants can change the pH level. The pH scale ranges from 0 (very acid) to 14 (very alkaline). The most frequent range in surface water systems is 6.5-8.0. It is important to mention that pH can affect others water quality parameters, chlorine, ammonia, metal solubility [2].

Redox- Reductivity potential measures the ability of lake to cleanse itself from the contaminants and dead plants and animals. This parameter of water quality is measured often as it provides scientists with additional information about the water quality and degree of pollution, if present. When Redox value is high, there is a lot of oxygen present in water and means that bacteria that decompose dead tissue and contaminants can work more efficiently. While low Redox, dissolved oxygen is low and obviously is not healthy. In normal drinking water, the Redox value should be between 300-500 mV [17].

Electrical Conductivity express the ability of water to conduct an electric current. Its property is of a little interest to water analyst but it is also a good indicator of the range into which the hardness and alkalinity values are likely to fall [1]. Electrical conductivity is measured as the free ions in the water conduct electricity, so the water electrical conductivity depends on the concentration of ions. Salinity and total dissolved solids (TDS) are used to calculate the EC of water, which helps to indicate the water's purity. There exists an interrelationship between conductivity and temperature, the higher the temperature the higher the mobility of ions, hence the conductivity.

Turbidity is a measure of the degree to which the water loses its transparency due to the presence of suspended particulates [29]. It is caused by particles suspended or dissolved in water that scatter light making the water appear cloudy or murky. The existence of turbidity in water will affect its acceptability to consumers and it will also affect markedly its utility in certain industries. Turbidity is considered a good measure of the quality of water [29].

Flow is important of its impact on water quality and on the living organisms and habitats in the stream. It is affected by weather, increasing during rainstorms and decreasing during dry periods. It also changes during different seasons of the year, decreasing during the summer months when evaporation rates are high and shoreline vegetation is actively growing and removing water from the ground. August and September are usually the months of lowest flow for most streams and rivers in most of the country [27].

5

DATA PREPROCESSING

Incompleteness and noise are an unavoidable problem when dealing with real-world data sets. Data may be lost due to various reasons like malfunctioning of equipment's, erroneous from human and so on. The presence of stochastic or deterministic trends in time series data can be also a major obstacle for producing good predictions. Adjusting the data before the classification is the first and necessary step. Objectives on this phase are data imputation, data cleansing, and data transformation. This chapter describes the steps required for the drinking water quality data preparation.

5.1 Missing Data

The data set we use in this thesis are noisy and contains missing values, so the preprocessing step is required. The training data set contains around 1% missing values. For non-time series data sets, this amount of missing values can be ignorable, but it is not the case with our water quality data set which is a time series data set. Handling missingness by deleting cases containing any missing values is a simple form but it has serious drawbacks in terms of elimination of useful information in the data. The second approach on handling missingness is by filling the missing values based on other information in the data set. In this experiment data imputation is necessary. There is a vast of methods of filling in the missing values. The method used can be statistical or a machine learning method. Some of statistical methods are mean imputation, median, mode, interpolation, multiple imputation [13]. Machine learning algorithms often are used such as k- nearest neighbors(KNN), decision trees(DT), random forest(RF), multi-layer perceptron(MLP)[18].

KNN is an algorithm that is useful for matching a point with its closest k neighbors in a multi-dimensional space. KNN imputation is a technique based on KNN algorithm designed to find k nearest neighbors for a missing instance of data from all cases complete in a given dataset. Then, it fills in missing values of example with the most frequent one occurring in the neighbors if the target feature (or attribute) is categorical, referred to as majority rule, or with the mean of the neighbors if the target feature is numerical, referred to mean rule. The assumption behind using KNN for missing values is that a point value can be approximated by the values of the points that are closest to it, based on other variables.

Multiple Imputation There are two basic imputation strategies: single imputation which provides single estimation for each missing data and multiple imputations such as multiple imputations (MI) (Little and Rubin [24]) and EM algorithm (Dempster et al. [11]). Some popular single imputation methods include hot deck imputation and mean imputation. Hot-deck imputation replaces missing values with responses from other records that satisfy certain matching conditions. Mean imputation estimates missing values by the mean value of appropriately selected ”similar” samples. A limitation of single imputation strategy is that it tends to reduce the variability

of characterizations of the imputed dataset artificially. With single-imputation techniques, missing values in a variable are imputed by an estimated value that results from matching some specific conditions. Single imputation cannot provide valid standard errors and confidence intervals since it ignores the implicit uncertainty. The alternatives of single imputation are to fill in the missing values with multiple imputations or iterative imputation such as MI or EM algorithm. MI algorithm assumption requires that the data (explicitly missing data) should be of type MCAR (missing completely at random) to generate a general-purpose imputation. In multiple imputation strategies, several different imputed datasets are, and a set of statistical results can be computed from it.

Random Forest missing data algorithms are new and attractive approaches for dealing with missing data because of the desirable properties of being able to handle mixed types of missing data, they are adaptive to interactions and non-linearity, and they have the potential to scale to big data settings. Random forests replace missing values only in the training set, and it begins by doing a rough and inaccurate filling in of the missing values. After this filling, it does a forest run and computes proximities. If $x(a, b)$ is a missing continuous value, estimate its fill as an average over the non-missing values of the $a - th$ variables weighted by the proximities between the $b - th$ case and the non-missing value case. If it is a missing categorical variable, replace it by the most frequent non-missing value where the frequency is weighted by proximity. After that iteration, construct a forest again using these newly filled in values, find new fills and iterate again. Normally 5-6 iterations are considered enough [7].

Mean imputation is one of the easiest ways to impute. It replaces each missing value with the mean of the observed values for that variable, which is known as unconditional mean imputation. In our data the three mentioned algorithms above, make no sense as the missing values on the drinking water quality data set occurred on all variables registered at a point of time. As seen in the image 5.1 there are 1044 samples with missing values. Because of this pattern of missingness, mean imputation has more sense. On the training set, all samples with missing values have a false event, and the imputation is done by using the mean of the day of samples with a false event.

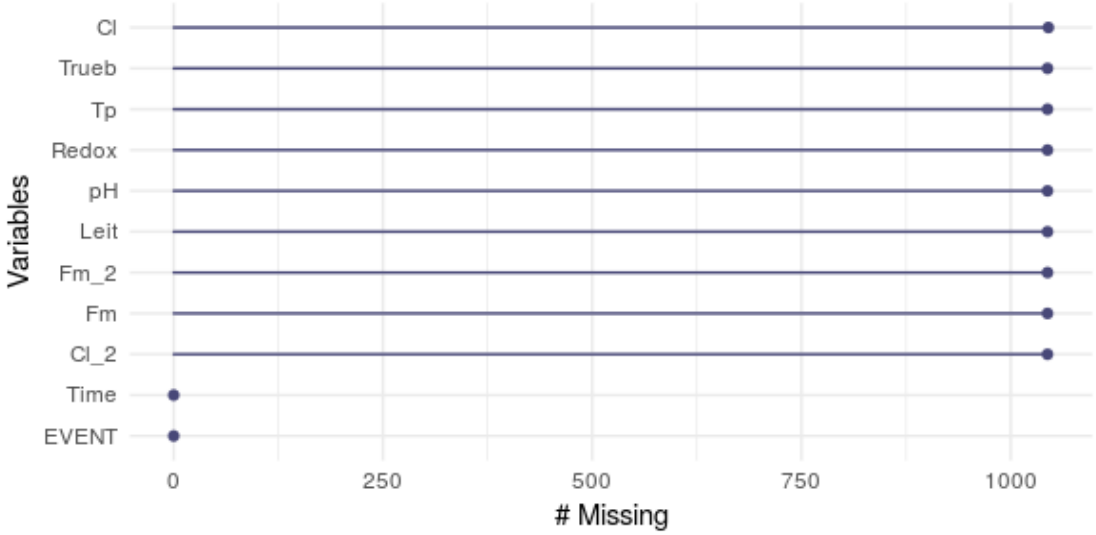


Figure 5.1: Drinking Water Quality missingness pattern

5.2 Stationarity of features

Why stationarity is important in time series data? Suppose we have the data $X_1, ..., X_n$. The most basic assumption is that X_i are independent. The independence is a nice property, since using it we can derive a lot of useful results. Also, machine learning models perform well under stationarity assumption.

To check whether the data is stationary or not, as we are using a multivariate time series then we have to do it separately for each feature of the data set, independently. Below we try to describe the analysis done for each feature and the conclusion gained for its stationarity. A general idea about the data has been generated for each variable of the data set, figure 5.4, 5.5. From the plots we can interpret that only *temperature* shows a long-run trend, as it is increasing till November, and then we see the behavior of decreasing, but not more training data were available. From the plots of other variables, it is obvious that there is no trend on other features. Because of the minutely based data, from these plots is not possible to detect any seasonality. Figure 5.2 maximizes the plot of each feature to a three-day interval, for the reason of seeing if there is any repeating pattern over the day(day-night seasonality). Again from the plots, we can gait that only the temperature shows to have a day-night seasonal pattern. It achieves its peak on the mid-day and it decreases during the night. Also, it is clear that there is no day-night

seasonality for other variables(Section 3.4).

It is not enough to decide upon stationarity only by looking at the plots, that's why we have applied two other different tests. First we tested each variable using ADF-test (described in chapter 3), secondly we used summary statistics. If the data is stationary the summary statistics should be consistent over time. The mean should be consistent with a consistent variance indicating a Gaussian distribution.

ADF- test results in the table II show that only temperature is non-stationary, p-value is greater than 0.05, and test statistic is greater than 5%critical values, hence the hypothesis can not be rejected. For all other variables we reject the null hypothesis, hence we derive that these variables are stationary.

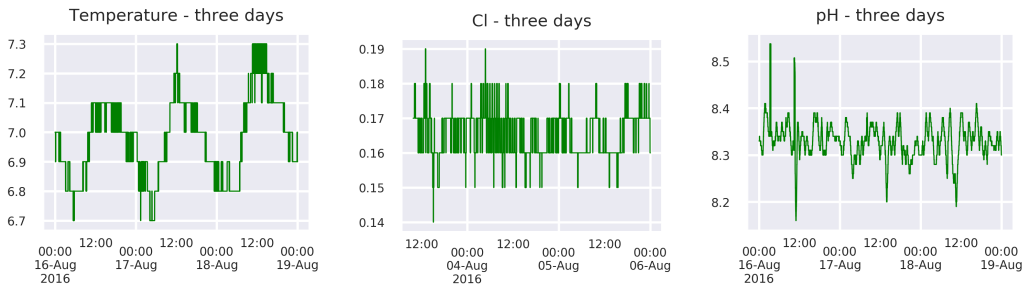
	Tp	Cl	pH	Redox	Leit	Cl2	Fm	Fm2
Test statistic	-1.369	-34.342	-29.63	-25.28	-17.77	-24.12	-10.38	-15.5
p-value	0.596	0.00	0.00	0.331	0.00	0.00	-0.209	0.221
Critical Value(5%)	-2.861	-2.861	-2.861	-2.861	-2.861	-2.861	-2.861	-2.861

Table II: The results of the ADF test for each variable. It is compared with only 5% critical value

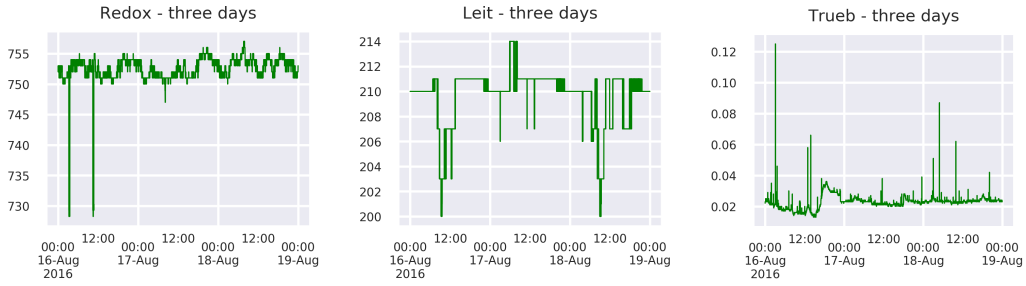
To check the summary statistics for all features of our data set we split the time series into two partitions and compare the mean and variance of each group. For the temperature, the mean and variance of each group difference are statistically significant, so this is again a strong indication that temperature is non-stationary.

5.2.0.1 Detrending temperature

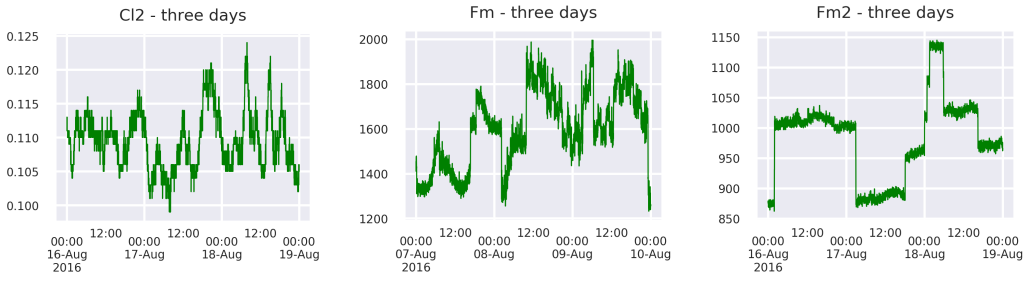
Most statistical forecasting methods are based on the assumption that the time series is stationary. Another reason for trying to stationarize a time series is to be able to obtain meaningful sample statistics such as means, variances, and correlations with other variables. When we deal with a non-stationary variable the most common way to deal with trends is to first remove them. However, sometimes even detrending is not sufficient to make the series stationary, so we can remove the trend from our data using first-order differencing. If first-order differencing does not work, then other methods have to be applied.



(a) Three days temperature time series (b) Three days chlorine dioxide time series (c) Three days pH-value time series



(d) Three days reductivity potential time series (e) Three days electrical conductivity time series (f) Three days turbidity time series

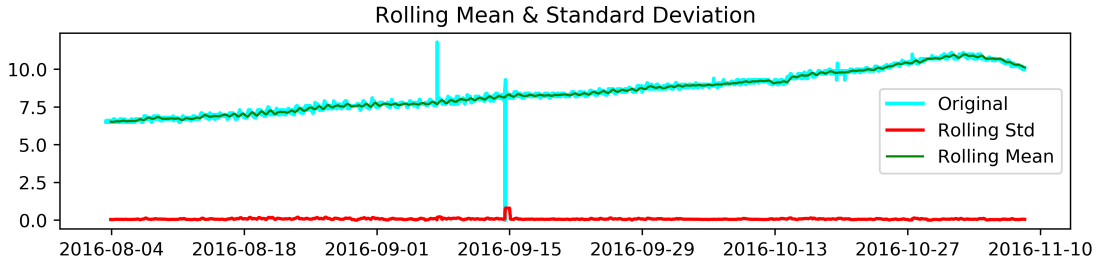


(g) Three days chlorine dioxide(2) time series (h) Three days flow rate time series (i) Three days flow rate(2) time series

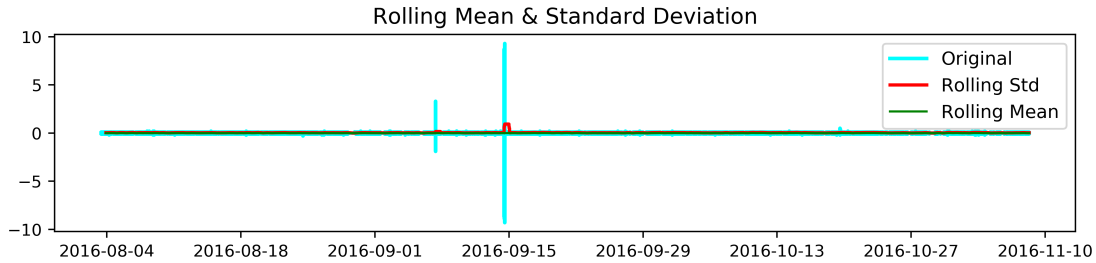
Figure 5.2: Looking for day-night seasonality using three days time series plot

We took the first order difference of the observation at a particular instant with that at the previous instant for the differencing. This appears to have reduced trend considerably. Figure 5.3 verifies it by these plots: figure 5.3(a) shows that the variation in standard deviation is small, mean is clearly increasing with time and this is not a stationary series, while the figure 5.3(b) interprets the differenced time series with a removed trend. Sometimes it can happen that first order differencing does not work on removing trend, in that case, we have to apply second or more order differencing. For the temperature feature in our data set the first order differencing was enough.

After temperature detrending with first order differencing we again applied the Dickey-Fuller test statistic to check for stationarity. The ADF test statistic for the differenced temperature is less than the 1% critical value, thus the differenced time series is stationary with 99% confidence.



(a) Rolling mean and standard deviation for temperature



(b) Rolling mean and standard deviation after first order differencing the temperature

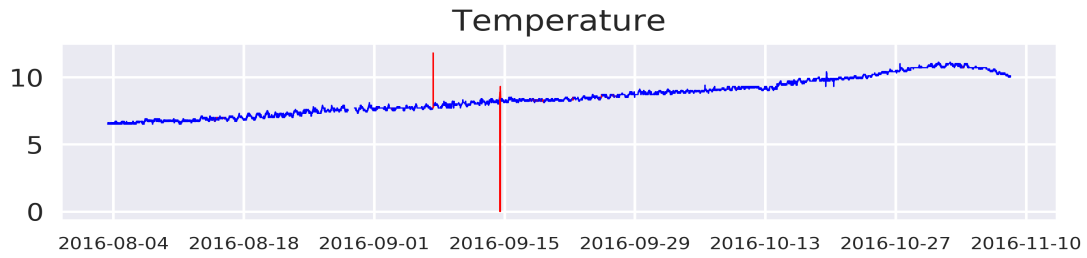
Figure 5.3: Rolling mean and standard deviation plots to check stationarity after first order differencing of the temperature

5.3 Moving average filtering

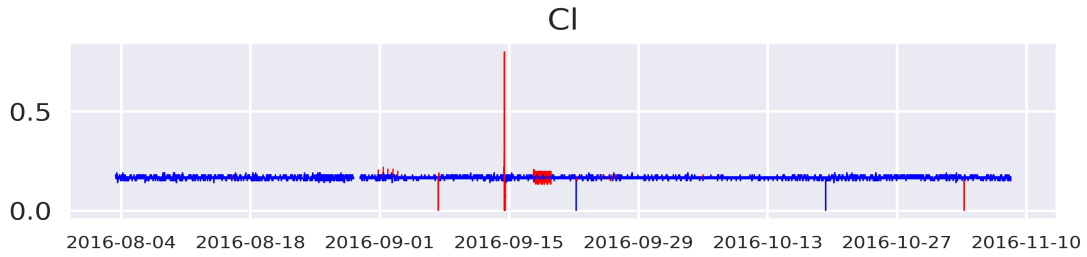
Moving average filtering is a simple data filtering technique by calculating the average value of every point in the selected window. The calculation of the new filtered data is done using the equation 5.1, where $((y_k)$ is the data at point k (before smoothing), n is a number of points in each side (left side or right side), and $-n \leq i \leq n$. The window size equals to $2n + 1$ [37].

$$(y_k)_s = \frac{\sum_{i=-n}^{i=n} y_{k+i}}{2n+1} \tag{5.1}$$

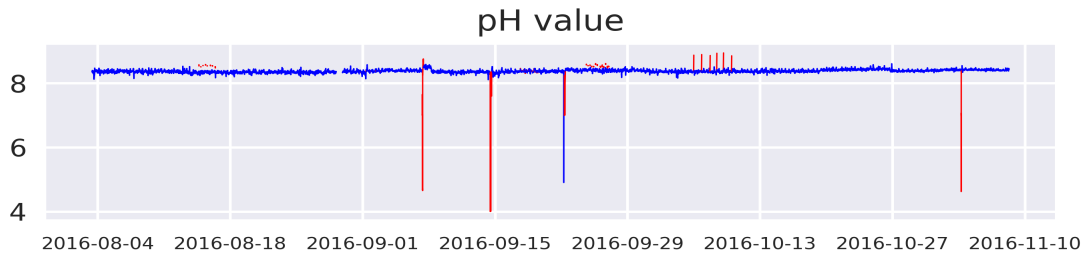
It is often useful to either low-pass filter (smooth) time series in order to reveal low-frequency features and trends, or to high-pass filter (detrend) time series in order to isolate high frequency transients (e.g. storms). A moving average is commonly used with time series data to smooth out short-term fluctuations and highlight longer-term trends or cycles. It helps us to remove the effect of the ”noise” from the data set. The larger the moving window, the smoother and less random the graph will be, but at the expense of accuracy. The purpose we are using moving average consists of filtering out noisy samples that could be just an anomaly and not a real event. There are two ways how rolling means can be applied in our data set, using heterogeneous windows - one that allows inclusion of different samples consisting of events and non-events samples and homogeneous windows - where it is restricted that windows contain only samples of one type: an event or no-event instances. We have gone with the first approach, since we use 1-step moving average in the case of an event happening on the last sample added to the window, it is trivial that any form of contamination couldn’t be detected immediately by the sensors because it would need some time for the contamination to spread and in this way by allowing heterogeneous windows we try to include this gradual change in values till the moment that event was finally reported.



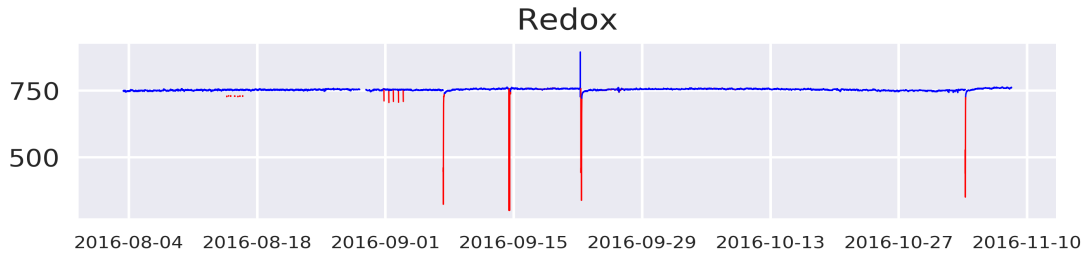
(a) Temperature of the water given in C



(b) Amount of chlorine dioxide in the water

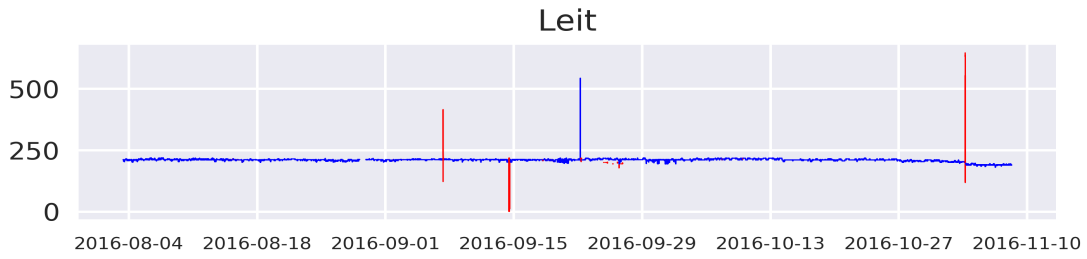


(c) pH-value of the water

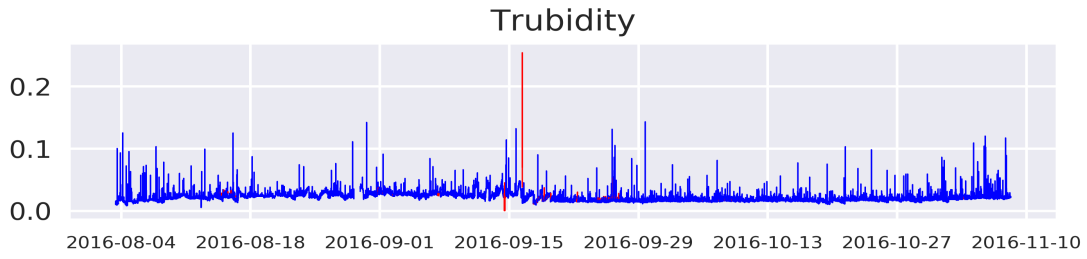


(d) Redox potential

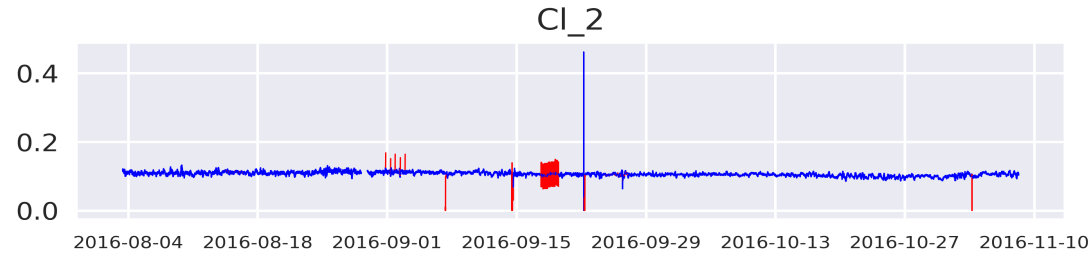
Figure 5.4: The plot shows an extract of the given time series data with original events marked with red (1).



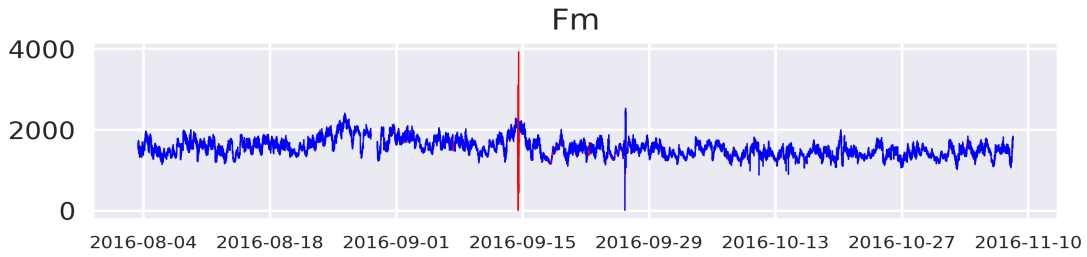
(a) Amount of chlorine dioxide in the water 2



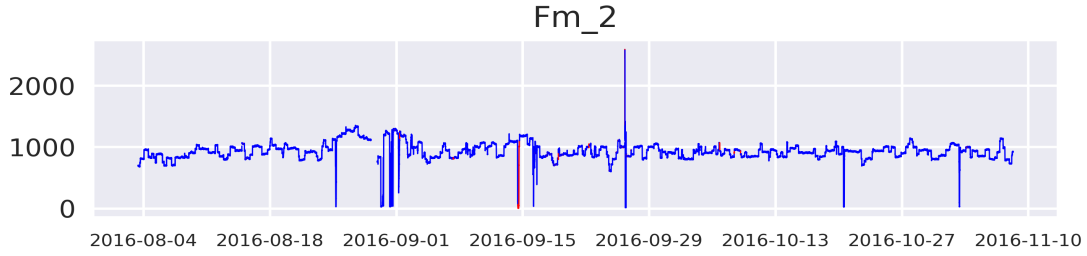
(b) Flow rate at water line 1



(c) Flow rate at water line 2



(d) Amount of chlorine dioxide in the water 2



(e) Amount of chlorine dioxide in the water 2

Figure 5.5: The plot shows an extract of the given time series data with original events marked with red (2).

6

ANOMALY DETECTION IN DRINKING WATER QUALITY

This section describes in detail the framework used for drinking water quality anomaly detection. First, we explain the process of feature selection. Secondly, we discuss classifiers used in our system.

6.1 Feature Selection

Increasing the number of features to modeling does not necessarily increase classifier accuracy since features may be redundant or not indicative of class (event/not event). Thus, feature selection is necessary to identify the essential features and eliminate redundant features. Feature selection methods can be categorized as filter methods, wrapper methods, or embedded methods

[39]. Filter methods use general characteristics of the data to evaluate features without involving a classifier in the process. A wrapper method is based on using accuracy from a specific classifier to select features. Embedded methods incorporate feature selection as an internal mechanism of classifier's training process.

Thus, both wrapper and embedded methods produce results that are specific to the classifier used for the task. Therefore, features weights, or feature subset selection, may only be useful to researchers using that particular classifier. Feature subsets with similar classifier performance to the full feature set should reduce computational burden, thus facilitating real-time implementations.

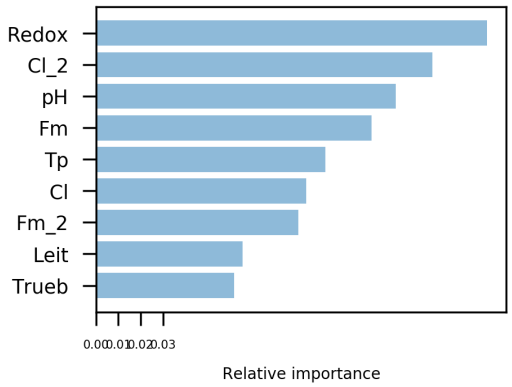


Figure 6.1: Feature importance using random forest

For the feature selection task, we employ embedded methods which consist of using classifiers feature importance internal mechanism to measure information gain (the predictive power) of each feature and then select those with highest predictive power. Both AdaBoost with DT, and Random Forest classifiers provide straightforward methods for feature selection: mean decrease impurity and mean decrease accuracy. The mean decrease impurity or sometimes called gini importance is defined as the total decrease in node impurity which is weighted by the probability of reaching that node which is approximated by the proportion of samples reaching that node averaged over all trees of the ensemble. Individual decision trees intrinsically perform feature selection by selecting appropriate split points. This information can be used to measure the importance of each feature; the basic idea is that the more often a feature is used in the split points of a tree, the more important that feature is. Features used at the top of the tree contribute to the final prediction decision of a larger fraction of the input samples. The expected fraction of the samples they contribute to can thus be used as an estimate of the relative importance of the features. This notion of importance can be extended to decision tree ensembles, in our case AdaBoost and Random Forest, by simply averaging the feature importance of each tree. In Figure 6.1 we have presented features with the highest

importance using ada boost with DT classifier. The effect of feature selection will be described in the next sections.

6.2 Classification models

Adaptive Boosting

Boosting is a general ensemble method that creates a strong classifier from a number of weak classifiers. The most popular boosting algorithm Adaptive Boosting (AdaBoost) which was introduced by Freund and Schapire[14] as a solution to many of the difficulties of earlier boosting algorithms, extended the idea of boosting as it was the first algorithm that could adapt to the weak learners. AdaBoost takes the training data, initially giving equal weights to all samples:

$$w_i^1 = \frac{1}{n}, i = 1, ..., n$$

Let W_e be the sum of the weights of misclassified samples we have

$$W_e = \sum_{G_t(x_i) \neq y_i} w_i^{(t)}$$

Over a series of rounds $t = 1, ..., T$ we always select the classifier G_t which minimizes the sum W_e, in other words, the classifier with the lowest weighted classification error.
Then, we set the weight α_t as a function of W_e and we use it to update the weights of the samples for the next round such that to assign a higher priority to samples that are misclassified:

$$w_i^{(t+1)} = \begin{cases} w_i^{(t)} e^{\alpha_t} & \text{if } G_t(x_i) \neq y_i \\ w_i^{(t)} e^{-\alpha_t} & \text{if } G_t(x_i) = y_i \end{cases}$$

The final equation for classification(binary cl.) can be represented as

$$G(x) = sign(\sum_{t=1}^{T} \alpha_t G_t(x))$$

In the process, features that are selected by the weak learners provides us with the mechanism of *features importance*; an automatic to identify the most relevant features for the final classifier design. As a weak learners we use decision trees(DT) with depth two and three, and we determine the number of boosting rounds T such that we have a good balance between the classifier accuracy and classifier complexity such that further increasing the value of T resulted only in marginal gains in accuracy. Decision Trees are reasonably fast to train and also fast to classify, which is good property since we have to build and run many of them before we can output the decision.

Random Forest

Bagging or *bootstrap aggregation* is a technique for reducing the variance of an estimated prediction function. The essential idea in bagging is to average many noisy but approximately unbiased models, and hence reduce the variance. Trees are ideal candidates for bagging since if grown sufficiently deep they have relatively low bias and can capture complex interaction structures in the data. They benefit greatly from the averaging since they are notoriously noisy. Bagging seems to work especially well for high-variance, low-bias procedures, such as trees. For regression, we simply fit the same regression tree many times to bootstrap-sampled versions of the training data and average the result. For classification, a committee of trees each cast a vote for the predicted class.

Random forests is a supervised learning algorithm introduced by (Breiman, 2001) is a substantial modification of bagging that builds a large collection of de-correlated trees, and then averages them. On many problems, the performance of random forests is very similar to boosting, and they are simpler to train and tune. As a consequence, random forests are popular and are implemented in a variety of packages. When used for classification, a random forest obtains a class vote from each tree and then classifies using majority vote. When used for regression, the predictions from each tree at a target point x are simply averaged. Decision trees tend to overfit very easily when trees are grown very deep which is often needed to increase accuracy and result in model learning highly irregular patterns, having a low bias, but very high variance. Random forests are a way of averaging multiple deep decision trees, trained in different parts of the same

training set, with the goal of reducing the variance. As it can easily be inferred from its name, it creates a 'forest' which is an ensemble of Decision Trees, in simple words: it builds multiple decision trees and merges them together to get a more accurate and stable prediction. The idea in random forests algorithm is to improve the variance reduction of bagging by reducing the correlation between the trees, without increasing the variance too much. This is achieved in the tree-growing process through a random selection of the input variables.

Instead of searching for the best feature while splitting a node, it searches for the best feature among a random subset of features. This process creates a wide diversity, which generally results in a better model. For the reasons that we mentioned above, Random Forests have been claimed that "cannot overfit" the data, which as described [15] does not hold as RF can overfit as result of having too many trees in the forest and to some extent by over-growing the trees. But in reality, RF classifiers are less sensitive to variance and the effect of overfitting is very seldom seen with RF classification [15].

Another great quality of the random forest algorithm is that it is very easy to measure the relative importance of each feature on the prediction. Sklearn provides a great tool for this, that measures a features importance by looking at how much the tree nodes, which use that feature, reduce impurity across all trees in the forest. It computes this score automatically for each feature after training and scales the results so that the sum of all importance is equal to 1. Through looking at the feature importance, you can decide which features you may want to drop because they don't contribute enough or nothing to the prediction process. This is very important to us because we want to compute as less as possible features without compromising in performance and it also helps to prevent overfitting because the more features you have, the more likely your model will suffer from overfitting and vice-versa.

Random Forests have a mechanism that estimates the test set error internally during the run such there is no need for cross-validation or a separate test set to get an unbiased estimate of the test set error.

Random Forests offer an excellent way to see how the data looks like through *proximity plot*. They tend to have a star shape, one arm per class, which is more pronounced the better the classification performance. If one tries to get this result by any of the present clustering algorithms, one is

faced with the job of constructing a distance measure between pairs of points and is very hard to do even in relatively small data sets.

Support Vector Machines

The SVM algorithm is a supervised learning algorithm introduced by Boser et al. [6]. SVMs are among the best "off-the-shelf" supervised algorithm. SVM is a linear two-class classifier. The idea of the linear classifier is to find a hyperplane that can classify data points appropriately. There are many hyperplanes that might classify the data but we are looking for the optimal separating hyperplane between the two classes by maximizing the margin between the classes closest points (see Figure 1) the points lying on the boundaries are called *support vectors*, and the middle of the margin is our optimal separating hyperplane - known as the *maximum-margin hyperplane* and the linear classifier it defines is known as a *maximum margin classifier*. Intuitively, a good separation is achieved by the hyperplane that has the largest distance to the nearest training data point of any class (so-called functional margin), since in general the larger the margin the lower the generalization error of the classifier[15]. SV classifiers are based on the class of hyperplanes given with

$$f(x) = w^T x + b$$

where vector $w \in \mathbb{R}^N$ is known as the weight vector and $b \in \mathbb{R}$ is called the bias.
Set of points x for which $f(x) = 0$, constitutes a hyperplane which divides the space into two regions such that to separate data points in two classes. The distance between the support vectors and the hyperplane is the margin; which can be shown that is equal to $\frac{2}{\|w\|}$.
The aim is to maximise the geometric margin $1/\|w\|$, which is equivalent to minimising $\left\|w^2\right\|$.
Finally, we can construct this as an optimisation problem:

$$\min \left\|w^2\right\| \text{ } \textit{subject to } y_i(w^T x_i + b) \geq 1 \quad i = 1, ..., n$$

Omitting the details of the calculations, there are two crucial properties of the algorithm that we need to emphasize: both the quadratic programming problem and the final decision function depend only on dot products between patterns which lets us to generalize to the nonlinear case,

and that support vectors carry all relevant information about the classification problem. SVM has a technique called the *kernel trick*. These are functions $\Phi : R^N \rightarrow F$ which takes low dimensional input space and transform it to a higher dimensional space (called the *feature space F*) i.e. it converts not separable problem to separable problem, these functions are called *kernels*. As we noted, performing the linear algorithm in F only requires the evaluation of dot products:

$$k(x, y) := (\Phi(x) \cdot \Phi(y))$$

. For instance, some famous kernels are shown below:

$$\text{Polynomial kernels} \quad k(x, y) = (x^T \cdot y + c)^d$$

$$\text{Radial basis function(RBF) kernels} \quad k(x, y) = exp(-\gamma \left\|x - y\right\|^2), \gamma > 0$$

$$\text{Sigmoid kernels(gain } \kappa\text{, offset } \Theta) \quad k(x, y) = tanh(\kappa(x \cdot y) + \Theta)$$

With the help of kernels, the concept of linear classifiers can be extended to non-linear problems. Kernels are used because direct computation of non-linear features is very expensive in the case of a huge quantity of data. As SVM is a binary class classifier, the one-against-one technique was employed and the correct class was determined using a voting mechanism.

The SVM algorithm constructs models that are complex enough: it contains a large class of neural nets, radial-basis-function (RBF) nets, and polynomial classifiers as special cases. Yet it is simple enough to be analyzed mathematically because it can be shown to correspond to a linear method in a high-dimensional feature space nonlinearly related to input space. Moreover, even though we can think of it as a linear algorithm in a high-dimensional space, in practice, it does not involve any computations in that high-dimensional space. By the use of kernels, all necessary computations are performed directly in input space.

7

EVALUATION

In this section, we discuss performance comparison between the classifiers mentioned in section 6.2. We present the event detection results when we employ the moving average filtering algorithm and compare with the event detection results on raw data. We also discuss the effect of the temperature-stationary in the classification performance.

7.1 Performance Evaluation

Many real-life situations create highly imbalanced data, so nowadays imbalanced data learning is one of the challenging problems in data mining. The first intuition when checking the performance of a classifier is to look at the accuracy of that model as the number of correct predictions from

all predictions made(Eq. 7.3). A predictive model may have high accuracy, but be useless. In our data set, we have only 1726 events out of 139566 samples, which is nearly 1.2% of the data. A trivial classifier that classifies all examples as non-events will achieve an accuracy of 98.8%, though its error rate for the minority class is 100%. The cost of diagnosing a true event as false can threat peoples life. When the performance on the minority class is as important or more important than overall accuracy, other performance measures must be used.

Table III: Confusion Matrix

	Predicted Positive	Predicted Negative
Actual Positive	TP	FN
Actual Negative	FP	TN

For imbalanced data sets when the minority class is an important class, performance metrics mentioned by Bekkar et al. [5] are often used. They are based on a confusion matrix (see Table III) that reports the number of true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). These are then used to define metrics that evaluate the performance of a learner on the minority class, such as recall, precision, and F_β -measure. The precision of a class (Eq. 7.2) is the number of TPs divided by the total number of examples predicted as positive. A precision score of 1.0 means that every example predicted as a positive example is a positive example, though there may be some positive examples that were labeled as negative. The recall of a class (Eq. 7.1) is the number of TPs divided by the number of examples that are actually positive. A recall score of 1.0 means that every positive example is labeled correctly, though some negative examples may have also been labeled as positive [20]. There is always a trade-off between precision and recall, but for data sets where the cost of false negatives is high, a high recall value is preferable. The F_1-measure (Eq. 7.4) is the weighted harmonic mean of precision and recall and merges recall and precision into a single value. The best F_1 score is 1 and the worst is 0 [5].

$$Recall = \frac{TP}{TP + FN} \tag{7.1}$$

$$Precision = \frac{TP}{TP + FP} \tag{7.2}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{7.3}$$

$$F_\beta = 2 \times \frac{Recall \times Precision}{\beta^2 Precision + Recall} \tag{7.4}$$

In our case, both, false positive and false negative have the same importance, so as a measure we consider F_1 score.

7.2 Detection results

Traditional k-fold cross-validation methods of model evaluation have limitations for time series data, as time series data is characterized by the correlation between observations that are near in time. To validate the classifiers, we use time series cross-validator, as it is very important to evaluate our model for time series data on ”future” observations least like those that are used to train the model. Time series cross-validator is a variation of k-fold which returns first k fold as train set and the (k+1)th fold as the test set. Differently from standard cross-validation methods, here the successive training sets are super-sets of those that come before them, it also adds all surplus data to the first training partition which is always used to train the model. In our training data set we have 4 months of data(the first and last months are not complete) and so we have split the data to four partitions. Additionally, we use a separate unseen testing data set to further validate the obtained results and which is a good estimation of how well the model will perform for the new and unseen cases in the future.

The procedure we followed is:

1. Perform time series cross validation which gives us a relatively good estimation error
2. Rebuild the model with the full dataset (the one that was split into folds in the previous step)
3. Use a separate test set to try the final model obtaining a similar(almost surely a higher) error than the one obtained by CV.

Table IV shows f1, precision and recall values after temperature detrending (Subsection 5.2.0.1. The best performance was gained by random forest, while SVM and DNN showed poor performance. From all the results we have seen that in conventional SVM classifier, a highly imbalanced distribution of data usually brings about poor classification accuracy for unseen samples from the minority class, because the classifier may be toward the majority class. SVMs

Classifier	F1 score	Precision	Recall
AdaB	0.40	0.40	0.40
RF	0.48	0.34	0.76
LSVM	0.29	0.3	0.73

Table IV: The effect of temperature stationarizing on models performance

Classifier	F1 score	Precision	Recall
AdaB	0.63	0.53	0.61
RF	0.45	0.29	1
LSVM	0.32	0.21	0.44

Table V: Model performance after moving average filtering

tend to learn how to majority class without considering the minority class, and this good accuracy performance can be identified as meaningless when we are concerned about minority class.

We have also applied a moving average filter to our time series as e reason to smooth the data since this often can lead to better prediction as it reduces noise. We have used a window of size 10 (minutes) because we assumed that when any contaminant enter the water it needs some time to spread and then to be detected from the sensors. Table V presents the performance after applying moving average filter, and here the ADA boosting outperformed the random forest. The problem with smoothing can be that is often less beneficial than we might think and that is why we decided to test how it affects the algorithm performance in our classifiers.

To address the problem of highly imbalanced data we used one well-known approach to deal with it, balancing the original data set, oversampling and undersampling. For undersampling and oversampling, we produce a random subsample of the data set The results are summarized in Table VI. From this table, we find that only Random Forest algorithm performance has been improved after undersampling. Also, the precision of each algorithm has been improved, which means that True Positive predictions have been increased, but especially for the SVM the recall and f1 are very small. From our experiment, we can conclude that balancing class prevalence before training a classifier does not across-the-board improve classifier performance. In fact, in

	Classifier	f1	Precision	Recall
Raw Data	AdaB	0.50	0.35	0.87
	RF	0.45	0.32	0.77
	LSVM	0.33	0.20	1.0
Oversampling	AdaB	0.47	0.33	0.78
	RF	0.37	0.23	0.94
	LSVM	0.26	0.98	0.21
Undersampling	AdaB	0.39	0.40	0.38
	RF	0.53	0.42	0.71
	LSVM	0.05	0.60	0.02

Table VI: The effect of oversampling, undersampling and raw data on classifiers detection on the testing set

drinking water quality data set it is contraindicated for AsaB and SVM models. It may be that the algorithms we choose are not the best choice for very unbalanced classes, but this problem can also be addressed to the small number of data we have, approximately 97 days of training data. Temperature and season change during the year can have a huge impact on accurate prediction of events.

Just for the curiosity, we have tested the effect of feature selection based on feature importance where we select the best 5 variables for predicting water quality data set (Section 6.1). The results are presented on table VII. When we compare with the performance of the raw data (Table VI) we can notice that random forest classifier and SVM classifier have almost the same f1 score, while for the ADA boosting classifier it is worst. As the number of features selected does not differ much from all features (5 out of 9) it does not affect much the computation time and we suggest that it is not necessary to remove them.

Table VII: Model performance using only feature importance

Classifier	F1 score	Precision	Recall
AdaB	0.36	0.38	0.35
RF	0.44	0.31	0.78
LSVM	0.30	0.20	0.63

8

CONCLUSION AND FUTURE WORK

In this master thesis, we have analyzed and discussed time series data, with the focus on water quality real-life data set. We have provided different techniques which can be used when working with time series and have explained the results we got after applying them to our data set. We have been focused on drinking water quality data set as it is a real-world problem and contributing to solving its issues can be beneficial for drinking water companies. The interesting part of the drinking water quality data set was that it was part of GECCO Industrial Challenge, and its solution was linked directly with the industry (water company- Thüringer Fernwasserversorgung). During this work, we wanted to explore more the difficulty of this data set, by using time series techniques. As we have discussed in the related work (Chapter 2) many works are focused on using statistical models for anomaly detection on water quality, so we wanted to test machine

learning models. The choice of classification algorithms was restricted by time, normally these systems expect that it should not exceed a maximum of 30 seconds per prediction. Because of the proven performance on many time series data sets, and because of the computational cost we used and compared (a) Adaptive Boosting (AdaBoost); (b) Random Forests (RF); (c) Support Vector Machines (SVM). Since this data set is considered noisy and has missing values we have tried to address this problem by using filtering methods and imputation techniques. The drinking water quality data set is characterized by being highly imbalanced. Imbalanced data sets often have detrimental effects on the performance of conventional classifiers. To solve this problem, we used the strategy of modifying the data distribution by re-sampling minority and majority classes to increase the generalization ability. Effects of each preprocessing step are shown on tables and comparison between them is done.

We considered different evaluation methods: the f1 score, precision, and recall of the classifiers. We have discussed why these metrics had to be used in highly imbalanced data and why accuracy is worthless as a measurement (Section 7.1). Finally, we give a conclusion that not much work has been done to solve highly imbalanced data, for this reason, a large amount of data are needed when we want to achieve good prediction.

As future work we can apply the work of Tian et al. [35] for class imbalance. [35] proposed a novel resampling approach and build an SVM ensemble for tackling problems associated with imbalanced data sets. Their method mainly improves the performance in two ways. First, the minority examples are trained using an OCSVM model, the SMOTE technique is applied to the support vector examples in order not to add too many outliers into the data set. Second, after removing the clusters farthest or nearest to the minority class, the novel MSK clustering algorithm decomposes the majority class into clusters for improving the generalization ability of the SVM ensemble [35]. Since anomaly detection in water quality is considered from many researchers difficult to be predicted, we encourage water companies to collect more data. In the future, we hope to cope with a larger real-world data set and be able to predict it correctly, by really minimizing the false predictions.

Bibliography

[1] Environmental Protection Agency. Time series; a biostatistical introducti parameters of water quality interpretation and standards. Technical report, 2001.

[2] SL Amarasiri. Caring for water. In *Sri Lanka Nature Forum (SLNF)*, volume 546, 2008.

[3] David F Andrews and Agnes M Herzberg. *Data: a collection of problems from many fields for the student and research worker*. Springer Science & Business Media, 2012.

[4] Elizabeth A Basha, Sai Ravela, and Daniela Rus. Model-based monitoring for early warning flood detection. In *Proceedings of the 6th ACM conference on Embedded network sensor systems*, pages 295–308. ACM, 2008.

[5] Mohamed Bekkar, Hassiba Kheliouane Djemaa, and Taklit Akrouf Alitouche. Evaluation measures for models assessment over imbalanced datasets. *Iournal Of Information Engineering and Applications*, 3(10), 2013.

[6] Bernhard E Boser, Isabelle M Guyon, and Vladimir N Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152. ACM, 1992.

[7] Leo Breiman and Adele Cutler. Random forests-classification description. *Department of Statistics, Berkeley*, 2, 2007.

[8] David Byer and Kenneth H Carlson. Real-time detection of intentional chemical con-

tamination in the distribution system. *Journal-American Water Works Association*, 97(7), 2005.

[9] Liangyue Cao, Alistair Mees, and Kevin Judd. Dynamics from multivariate time series. *Physica D: Nonlinear Phenomena*, 121(1-2):75–88, 1998.

[10] Robert M Clark, Simon Hakim, and Avi Ostfeld. *Handbook of Water and Wastewater Systems Protection*. Springer, 2011.

[11] Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society. Series B (methodological)*, pages 1–38, 1977.

[12] Peter J Diggle. Time series; a biostatistical introduction. Technical report, 1990.

[13] Gift Dumedah and Paulin Coulibaly. Evaluation of statistical methods for infilling missing values in high-resolution soil moisture data. *Journal of Hydrology*, 400(1-2):95–102, 2011.

[14] Yoav Freund and Robert E Schapire. A desicion-theoretic generalization of on-line learning and an application to boosting. In *European conference on computational learning theory*, pages 23–37. Springer, 1995.

[15] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics New York, 2001.

[16] David Hart, Sean A McKenna, Katherine Klise, Victoria Cruz, and Mark Wilson. Canary: a water quality event detection algorithm development tool. In *World Environmental and Water Resources Congress 2007: Restoring Our Natural Habitat*, pages 1–9, 2007.

[17] Alexander J Horne and Charles R Goldman. Nitrogen fixation in clear lake, california. i. seasonal variation and the role of heterocysts1. *Limnology and Oceanography*, 17(5): 678–692, 1972.

[18] José M Jerez, Ignacio Molina, Pedro J García-Laencina, Emilio Alba, Nuria Ribelles, Miguel Martín, and Leonardo Franco. Missing data imputation using statistical and machine

learning methods in a real breast cancer problem. *Artificial intelligence in medicine*, 50(2): 105–115, 2010.

[19] Gary King and Langche Zeng. Logistic regression in rare events data. *Political analysis*, 9 (2):137–163, 2001.

[20] Suzan Köknar-Tezel and Longin Jan Latecki. Improving svm classification on imbalanced time series data sets with ghost points. *Knowledge and information systems*, 28(1):1–23, 2011.

[21] Ashok Kumar, BS Bisht, VD Joshi, AK Singh, and Amitabh Talwar. Physical, chemical and bacteriological study of water from rivers of uttarakhand. *Journal of Human Ecology*, 32(3):169–173, 2010.

[22] Denis Kwiatkowski, Peter CB Phillips, Peter Schmidt, and Yongcheol Shin. Testing the null hypothesis of stationarity against the alternative of a unit root: How sure are we that economic time series have a unit root? *Journal of econometrics*, 54(1-3):159–178, 1992.

[23] Jae Young Lee, Ramez Elmasri, and Jongho Won. An integrated temporal data model incorporating time series concept. *Data & knowledge engineering*, 24(3):257–276, 1998.

[24] Roderick JA Little and Donald B Rubin. Bayes and multiple imputation. *Statistical analysis with missing data*, pages 200–220, 2002.

[25] Spyros Makridakis, Steven C Wheelwright, and Rob J Hyndman. Forecasting methods and applications, john wiley& sons. *Inc, New York*, 1998.

[26] M Malakootian and J Nouri. Chemical variations of ground water affected by the earthquake in bam region malakootian, m. *International Journal of Environmental Research*, 4(3): 443–454, 2010.

[27] Tom Murdoch, Martha Cheo, and Kate O'Laughlin. *The streamkeeper's field guide: watershed inventory and stream monitoring methods*. 1996.

[28] Guy P Nason. Stationary and non-stationary time series. *Statistics in Volcanology. Special Publications of IAVCEI*, 1:000–000, 2006.

[29] Ahmad Fairuz Omar and Mohd Zubir Mat Jafri. *Optical system in measurement of water turbidity: Design and Analytical Approach (Penerbit USM)*. Penerbit Usm, 2015.

[30] Philadelphia Water Department. Contamination warning system dashboard development, 2013.

[31] Chandrasekaran Rebolledo Friese Rehbach, Moritz and Bartz-Beielstein. Gecco 2018 industrial challenge: Monitoring of drinking-water quality. Technical report, 2018.

[32] Alexander V Shitov. Health of people living in a seismically active region. *Man and the Geosphere, Nova Science Publishers, Inc., New York*, pages 185–214, 2010.

[33] Robert H Shumway and David S Stoffer. Time series analysis and its applications. *Studies In Informatics And Control*, 9(4):375–376, 2000.

[34] Paul Somerville. Kobe earthquake: An urban disaster. *Eos, Transactions American Geophysical Union*, 76(6):49–51, 1995.

[35] Jiang Tian, Hong Gu, and Wenqi Liu. Imbalanced classification using support vector machine ensemble. *Neural Computing and Applications*, 20(2):203–209, 2011.

[36] Urumu Tsunogai and Hiroshi Wakita. Precursory chemical changes in ground water: Kobe earthquake, japan. *Science*, 269(5220):61–63, 1995.

[37] Wiphada Wettayaprasit, Nasith Laosen, and Salinla Chevakidagarn. Data filtering technique for neural networks forecasting. In *Proceedings of the 7th WSEAS International Conference on Simulation, Modelling and Optimization*, pages 225–230. World Scientific and Engineering Academy and Society (WSEAS), 2007.

[38] Jin Zhang, Xiaohui Zhu, Yong Yue, and Prudence WH Wong. A real-time anomaly detection algorithm/or water quality data using dual time-moving windows. In *Innovative Computing*

Technology (INTECH), 2017 Seventh International Conference on, pages 36–41. IEEE, 2017.

[39] Mi Zhang and Alexander A Sawchuk. A feature selection-based framework for human activity recognition using wearable multimodal sensors. In *Proceedings of the 6th International Conference on Body Area Networks*, pages 92–98. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2011.

[40] Roberto V Zicari. Big data: Challenges and opportunities. *Big data computing*, 564, 2014.

Statement of authorship

I confirm that this Master's thesis is my own work and I have documented all sources and material used. This thesis was not previously presented to another examination board and has not been published.

Frankfurt Am Main, October 11, 2018 . . . . . .