



KANDIDAT

**4**

PRØVE

# IE502414 1 Masteroppgave i Simulering og Visualisering

---


Emnekode	IE502414
Vurderingsform	Oppgave
Starttid	01.06.2018 12:00
Sluttid	29.06.2018 12:00
Sensurfrist	29.09.2018 02:00
PDF opprettet	05.11.2018 09:35
Opprettet av	Anne Lise Grande

---




## 1 Ny oppgave

Please upload your master thesis here (PDF file).

We refer to the e-mail dated 30.05.18 from Anne Lise Grande regarding: 1. The Template for the Front Page, 2. Mandatory Statement and 3. Publication Agreement. These three files must be included in your PDF file together with the thesis.



Din fil ble lastet opp og lagret i besvarelsen din.

 Last ned Fjern Erstatt

Filnavn:	Thesis.pdf
Filtype:	application/pdf
Filstørrelse:	13.6 MB
Opplastingstidspunkt:	28.06.2018 08:35
<b>Status:</b>	<b>Lagret</b>

Besvart.



Norwegian University of  
Science and Technology

# Master's degree thesis

Applying hyperparameter optimization and other  
model adaptation methods to tune existing models for  
microbial pathogens in drinking water supplies

Author(s): Niklas Noem

Number of pages including this page: 82

Aalesund, 29.06.2018

## Mandatory statement

Each student is responsible for complying with rules and regulations that relate to examinations and to academic work in general. The purpose of the mandatory statement is to make students aware of their responsibility and the consequences of cheating. **Failure to complete the statement does not excuse students from their responsibility.**

Please complete the mandatory statement by placing a mark <u>in each box</u> for statements 1-6 below.		
1.	I/we hereby declare that my/our paper/assignment is my/our own work, and that I/we have not used other sources or received other help than is mentioned in the paper/assignment.	<input checked="" type="checkbox"/>
2.	<p>I/we hereby declare that this paper</p> <ol style="list-style-type: none"> <li>1. Has not been used in any other exam at another department/university/university college</li> <li>2. Is not referring to the work of others without acknowledgement</li> <li>3. Is not referring to my/our previous work without acknowledgement</li> <li>4. Has acknowledged all sources of literature in the text and in the list of references</li> <li>5. Is not a copy, duplicate or transcript of other work</li> </ol>	<p>Mark each box:</p> <ol style="list-style-type: none"> <li>1. <input checked="" type="checkbox"/></li> <li>2. <input checked="" type="checkbox"/></li> <li>3. <input checked="" type="checkbox"/></li> <li>4. <input checked="" type="checkbox"/></li> <li>5. <input checked="" type="checkbox"/></li> </ol>
3.	I am/we are aware that any breach of the above will be considered as cheating, and may result in annulment of the examination and exclusion from all universities and university colleges in Norway for up to one year, according to the <a href="#">Act relating to Norwegian Universities and University Colleges, section 4-7 and 4-8</a> and Examination regulations at NTNU.	<input checked="" type="checkbox"/>
4.	I am/we are aware that all papers/assignments may be checked for plagiarism by a software assisted plagiarism check.	<input checked="" type="checkbox"/>
5.	I am/we are aware that The Norwegian University of Science and Technology (NTNU) will handle all cases of suspected cheating according to prevailing guidelines.	<input checked="" type="checkbox"/>
6.	I/we are aware of the University's rules and regulations for using sources.	<input checked="" type="checkbox"/>

# Publication agreement

ECTS credits: 30

Supervisor: Assoc. Prof. Hao Wang and Prof. Razak Seidu

## Agreement on electronic publication of master thesis

Author(s) have copyright to the thesis, including the exclusive right to publish the document (The Copyright Act §2).  
All theses fulfilling the requirements will be registered and published in Brage with the approval of the author(s).

I/we hereby give NTNU the right to, free of charge, make the thesis available for electronic publication: yes no

Is there an [agreement of confidentiality](#)? yes no  
(A supplementary confidentiality agreement must be filled in)

Date: 29.06.2018

## **Acknowledgements**

I would like to thank my supervisors Prof. Razak Seidu and Assoc.Prof Hao Wang for the support they have provided during this thesis. I would also like to thank Hadi Mohammed and Di Wu, whose work this thesis is built upon, for the support they have provided. Finally, I'd like to thank the Water and Wastewater Departments of Ålesund and Oslo Kommune for providing the data.

## **Foreword**

The availability of drinking water in the right quantity and quality is critical for the socio-economic transformation of every country. Nevertheless, in many parts of the world, systems for ensuring the provision of drinking water in the right quality and quantity are significantly challenged. In Norway, most water supply systems depend on surface water sources for the provision of drinking water. These surface water sources are often susceptible to microbial contamination. Consequently, all kinds of microbial organisms of significant public health concern such as viruses, bacteria, parasites and protozoa can be found in the raw water sources used for drinking water supply.

Machine learning models have been employed in attempts to predict the concentration of microbial organisms to varying degrees of success. In this thesis some of these algorithms will undergo further enhancement by performing hyperparameter optimization combined with various methods for data pre-processing. Ideally, the water treatment plants should be able to rely on these models to enhance reaction times when deploying countermeasures for microbial contamination. Not only could this serve to further protect public health, but also reduce the cost of operating such water treatment plants.

## **Abstract**

The collection and analysis of data on the concentration of pathogenic organisms in raw water sources is critical for the optimization of disinfection processes in water treatment plants. Nevertheless, there are no robust real-time sensors for determining microbial concentrations in raw water sources, and many water treatment plants still rely on very laborious, time consuming and costly traditional laboratory methods. In the traditional laboratory methods, the concentration of faecal indicator bacteria in a raw water that is to be treated may not be known until 18-24 hours after the water has been distributed to the population, while that of virus and parasites may take several days to determine. Consequently, waterborne disease outbreaks associated with water supply systems often occur before remedial actions are taken because the concentrations of microbial pathogens in the raw water sources is not known beforehand.

To achieve the ultimate goal of protecting public health, early detection of microbial organisms in raw water is necessary for the development of proactive risk management strategies. Besides enhancing microbial detection tools, mathematical models can be employed to reliably predict the concentration of microbial organisms in raw water. For this purpose, several mathematical models have been developed including Machine Learning Algorithms to predict the occurrence/concentration of pathogenic organisms in raw water sources. These models use easily analysable physio-chemical parameters (e.g. temperature, pH, turbidity, electrical conductivity, colour, etc) in the raw water to predict the occurrence/concentration of microbial pathogens. Machine learning models that have been successfully applied in predicting the concentration of microbial organisms in raw water sources include support vector machines, random forests, extreme learning machines and adaptive neuro-fuzzy inference systems. However, these models are often built to predict the concentration of microbial pathogens in a single



water source and are therefore often poor in predicting the concentrations of pathogens when applied to other water sources.

The overall aim of this work is to apply hyperparameter optimization of machine learning models combined with various methods of data pre-processing to improve the adaptability and effectiveness of models developed for a plant in predicting the concentrations of microbial organisms in other plants.

Data used in this work were obtained from Brusdalsvatnet in Ålesund and Maridalsdalsvannet in Oslo. Brusdalsvatnet is the main drinking water source for Ålesund Kommune and surrounding communities, while Maridalsvannet is the main drinking water source for parts of Oslo Kommune. Using a library for the programming language Python called hyperopt-sklearn, hyperparameter optimized models were trained and the best configuration was selected. Hyperopt-sklearn, is based on scikit-learn, a package for Python containing a wide variety of tools for machine learning applications, and hyperopt, a tool for hyperparameter optimization in Python. In each experiment, the optimizer evaluates 100 different configurations of hyperparameters and pre-processors. The learners that have been used are limited to those available in the library. Of the machine learning algorithms that have been used previously to predict the concentration of microbial organisms in water, support vector machine and random forest are the only ones available in the tool. However, to see if any other of the available learners could potentially be successful in predicting microbial organisms, the optimizer was allowed to choose the learner itself as well. Optimized random forest, support vector machine and k nearest neighbour along with default random forest and support vector machine were the machine learning algorithms trained to predict the concentration of coliform bacteria and *E. Coli* in the data from Maridalsvannet. A visual inspection of the predictions made by these algorithms was done by plotting them against the observed values. The plots showed that that the optimized model

clearly performs better than the default ones. Particularly since these values have a tendency to rise in several peaks over values that otherwise are close to zero, and the optimized models were better at recognizing these peaks and their magnitudes. However, the machine learning package comes with scoring metrics which tell a different story. These scoring metrics would in many cases score the default algorithms better than the optimized ones. Since the optimizer uses one such scoring method by default in its internal model selection mechanism, it is reasonable to assume that even better results might be achieved if the scoring methods could recognize that predicting the peaks are much more important than accurately predicting small values around zero. Moreover, there are some challenges caused by the different water treatment plants not having the same standard procedures for accounting for different parameters in their raw water sources.

The study has shown that optimized models can significantly improve the models' ability to predict the concentration of microbial organisms in raw water sources. It is recommended that these procedures be used in further developing models for water treatment plants. Ideally, to improve the usability of these models the water treatment plants should work on a more standardized procedure as well. Furthermore, developing a new scoring method tailored for this problem in particular might further improve optimization of these models.

# Table of contents

1	Introduction .....	1
1.1	Problem definition .....	1
1.1.1	Background and traditional models.....	1
1.1.2	Addressing the challenges with the traditional models.....	2
1.1.3	New challenges .....	2
1.2	Motivation.....	3
1.3	Scope.....	3
1.4	Objectives.....	3
1.4.1	Research questions .....	4
2	Literature Review .....	5
2.1	Machine Learning.....	5
2.1.1	Types of learning.....	5
2.1.2	Popular approaches to ML.....	6
2.1.3	Hyperparameters .....	7
2.2	Other intelligent systems .....	7
2.2.1	Decision Tree.....	7
2.2.2	Rule-based expert system.....	8
2.2.3	Fuzzy expert system .....	8
2.3	ML methods used for predicting pathogens.....	13
2.3.1	ANFIS .....	13
2.3.2	Random forest.....	14
2.3.3	Support Vector Machine .....	14
2.3.4	Extreme Learning Machine .....	15
2.4	Hyperparameter optimization .....	16
2.4.1	Random search.....	16
2.4.2	Bayesian optimization .....	16
2.5	Microbial pathogens in water supply systems.....	17

2.5.1	Indicator organisms.....	17
2.5.2	Pathogens of concern for Norwegian drinking water.....	18
3	Methodology.....	20
3.1	Set-up.....	20
3.2	Pre-study.....	20
3.3	Experiments.....	20
3.3.1	Principal Component Analysis.....	22
3.3.2	Standard Scaler.....	22
3.3.3	Min-Max Scaler.....	23
3.4	Model comparisons.....	23
3.4.1	Mean Absolute Error.....	23
3.4.2	Mean Squared Error.....	23
3.4.3	Median Absolute Error.....	24
3.4.4	Explained Variance.....	24
3.4.5	R <sup>2</sup> score.....	24
4	Results.....	25
4.1	The Data.....	25
4.1.1	Maridalsvannet.....	25
4.1.2	Brusdalsvatnet.....	26
4.2	Web-based optimization tool.....	28
4.3	Hyperparameter optimization results.....	28
4.3.1	Predicting Coliform Bacteria.....	28
4.3.2	Coliform Bacteria: Comparison of scoring metrics.....	28
4.3.3	Coliform Bacteria: Visual comparison.....	33
4.3.4	Predicting E. coli.....	37
4.3.5	E. coli: Comparison of scoring metrics.....	37
4.3.6	E. coli: Visual Comparison.....	41
4.3.7	Training models on the data without temperature.....	46

5	Discussion.....	50
5.1	The Data .....	50
5.1.1	Differences in raw-water catchments.....	50
5.2	Hyperparameter optimization results.....	51
5.2.1	The scoring metrics .....	51
5.2.2	Stochasticity in training the models.....	52
5.2.3	Training models on the data from Brusdalsvatnet .....	53
5.2.4	The choice of learners .....	53
6	Concluding Remarks.....	54
7	References.....	55
	Appendix A: Optimization tool.....	1

## Figures

Figure 1.1 Measuring stations and treatment plant.....	2
Figure 2.1 A simple mathematical model for a neuron.....	6
Figure 2.2 Partitions (left) and decision tree structure (right) (Loh, 2011).....	7
Figure 2.3 The basic structure of Mamdani-style fuzzy inference (Negnevitsky, 2011, p. 108).....	10
Figure 2.4 The basic structure of Sugeno-style fuzzy inference (Negnevitsky, 2011, p. 114).....	12
Figure 2.5 ANFIS Architecture (Suparta & Alhasa, 2016, p. 12).....	13
Figure 2.6 Hyperplane with margin separating two classes (Negnevitsky, 2011, p. 171).....	15
Figure 2.7 Modified space to achieve linear separability (Negnevitsky, 2011, p. 176).....	15
Figure 2.8 Number of incidents from outbreaks in Norway 1992-2014 (Andersen, 2016, p. 9).....	18
Figure 3.1 Methodology flow chart .....	21
Figure 3.2 Experiments flow chart .....	21
Figure 4.1 Coliform Bacteria in Maridalsvannet .....	26
Figure 4.2 E. Coli in Maridalsvannet .....	26
Figure 4.3 Box plot of the values from Maridalsvannet .....	26
Figure 4.4 Coliform Bacteria in Brusdalsvatnet.....	27
Figure 4.5 Box plots of the values from Brusdalsvatnet.....	27
Figure 4.6 Comparing models using Mean Absolute Error .....	29
Figure 4.7 Comparing models using Mean Squared Error.....	30
Figure 4.8 Comparing models using Median Absolute Error .....	31
Figure 4.9 Comparing models using Explained Variance .....	32
Figure 4.10 Comparing models using R <sup>2</sup> score.....	32
Figure 4.11 Optimized Random Forest with no pre-processing .....	33
Figure 4.12 Optimized SVM with no pre-processing .....	34
Figure 4.13 Optimized SVM pre-processed with PCA .....	35
Figure 4.14 Optimized KNN with no pre-processing .....	35

Figure 4.15 Default Random Forest .....	36
Figure 4.16 Default SVM .....	37
Figure 4.17 Mean Absolute Error comparison predicting E. coli .....	38
Figure 4.18 Mean Squared Error comparison predicting E. coli .....	39
Figure 4.19 Median Absolute Error comparison predicting E. coli .....	39
Figure 4.20 Explained Variance comparison predicting E. coli.....	40
Figure 4.21 R <sup>2</sup> comparison predicting E. Coli .....	41
Figure 4.22 Optimized Random Forest with no pre-processing .....	42
Figure 4.23 Optimized Random Forest pre-processed with Min-Max Scaler.....	43
Figure 4.24 Optimized SVM with no pre-processing .....	43
Figure 4.25 Optimized SVM pre-processed with PCA .....	44
Figure 4.26 Optimized KNN with no pre-processing .....	45
Figure 4.27 Optimized KNN pre-processed with Min-Max Scaler.....	45
Figure 4.28 Optimized Random Forest on data without temperature.	47
Figure 4.29 Predictions on the data from Brisdalsvatnet using model trained on the data from Maridalsvannet .....	48
Figure 4.30 Optimized SVM on the data from Maridalsvannet without temperature .....	48
Figure 4.31 Optimized KNN on the data from Maridalsvannet without temperature .....	49
Figure A.1: Projects home page.....	1
Figure A.2: Adding a new project.....	2
Figure A.3: Project start page .....	2
Figure A.4: Project data configuration .....	3
Figure A.5: Two data sets for one project .....	4
Figure A.6: Creating models .....	5
Figure A.7: Model details .....	6
Figure A.8: Crosschecking model with other datasets.....	7

## **Abbreviations**

ML – Machine Learning

ANN – Artificial Neural Network

GA – Genetic Algorithm

SMBO – Sequential Model-Based Optimization

ANFIS – Adaptive Neuro-Fuzzy Inference System

SVM – Support Vector Machine

ELM – Extreme Learning Machine

WHO – World Health Organization

NIPH – Norwegian Institute of Public Health

CART – Classification and Regression Trees

COG – Centre of gravity

WA – Weighted average

SLFN – Single-hidden layer feedforward neural network

PCA – Principal Component Analysis

CSV – Comma Separated Values

MAE – Mean Absolute Error

MSE – Mean Squared Error

MedAE – Median Absolute Error

EV – Explained Variance



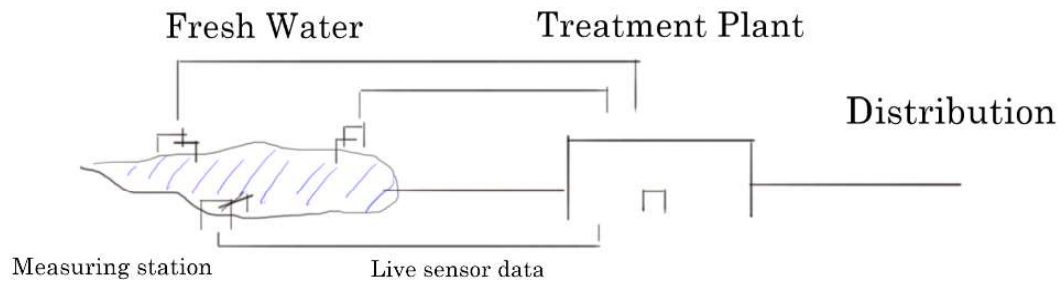
# 1 Introduction

## 1.1 *Problem definition*

### 1.1.1 Background and traditional models

The availability of drinking water in the right quantity and quality is critical for the socio-economic transformation of every country. Nevertheless, in many parts of the world, systems for ensuring the provision of drinking water in the right quality and quantity are significantly challenged. In Norway, where most water supply systems depend on surface water sources for the provision of drinking water to the population. These surface water sources are often susceptible to significant contamination. Consequently, pathogens of significant public health concern such as viruses, bacteria, parasites and protozoa origins can be found in raw water sources used for drinking water supply. To protect public health, the water treatment plants must deploy proactive and reactive risk management strategies. For the proactive risk management strategies to work effectively, early detection of microbial organisms in raw water is necessary. Besides improving the tools used for microbial detection, statistical models can be employed to reliably predict the concentration of microbial organisms in raw water. The water treatment plants can use live measurements of physical and chemical parameters to predict this concentration of microbial organisms. Measurements of microbial data is typically non-negative integers, possibly skewed with a large number of zeroes. In general, this zero-inflatedness is being accounted for by approximation to normal distribution in multiple linear regression models (Eregno, et al., 2014; Herring, et al., 2015). However, studies have shown that such transformations do not entirely eliminate certain violations of traditional regression models such as heteroscedasticity and could

produce biased parameter estimates (Coxe, et al., 2009; Beaujean & Morgan, 2016).



*Figure 1.1 Measuring stations and treatment plant*

### **1.1.2 Addressing the challenges with the traditional models**

To address the challenges associated with traditional linear models, a suite of AI, random forest models and zero-inflated models have been developed to predict the occurrence of microbial organisms in Norwegian water supply systems (Mohammed, et al., 2018; Mohammed, et al., 2017; Wu, et al., 2012). These models have been shown to have very good predictive capabilities for indicator organisms in raw water sources given a set of physical and chemical parameters. The models were based on data from the Oset Water Treatment Plant, which has advanced systems for the collection and collation of water quality data. However, the applicability of the models to other water supply systems in Norway has been encumbered by the availability and quality of predictive data.

### **1.1.3 New challenges**

The main issue is that the type of measurements, the frequency for which these measurements are recorded, and how the measurements are recorded vary greatly across different treatment plants. Given this issue the models that are designed for, and trained on data from, one water treatment plant cannot necessarily be used for the data from other plants.

## ***1.2 Motivation***

Being able to predict microbial pathogens in drinking water supplies could significantly improve response time in water treatment plants and reduce cost of operation by increasing the use of “cheap” data provided by live sensors instead of expensive lab tests. To increase the capacity and adaptability of the existing models could potentially increase the accuracy and usability of these models for a wider variety of treatment plants.

## ***1.3 Scope***

The scope of this project is a study of the existing models and the available data, identification of possible model adaptation methods and the application of at least one of these model adaptation methods and study of the model’s performance when applied to other data sources. The project will not include an infrastructure for use of these model adaptation methods but is limited to identifying what methods can be used and whether these methods could improve the existing models.

## ***1.4 Objectives***

The objective of this thesis is to understand the existing data and the existing models, to understand parameters and hyperparameters and perform data pre-processing. To evaluate the existing data and existing models and identify possible hyperparameter optimization methods. And to assess (semi)-automatic model adaptation and fine-tuning methods for the applicability of the developed models for water supply systems in Norway.

#### **1.4.1 Research questions**

- Could pre-processing the data before training the models improve the results?
- What methods can be used for fine-tuning of the existing models?
- Could hyperparameter optimization improve the models' ability to predict concentration of microbial organisms?

## **2 Literature Review**

The literature search leading to this literature review was mostly conducted via Google's search engine for academic texts *Google Scholar* and *Oria*. Keywords used to find these texts include but are not limited to: "machine learning", "hyperparameter", "hyperparameter optimization", "model selection", "imputation" and "pathogens in water supplies". Some of the books that have been studied are books that has been used as learning material in the courses during the Simulation and Visualization MSc program at NTNU Ålesund.

### ***2.1 Machine Learning***

Machine Learning is about making computers modify or adapt their actions so that these actions get more accurate, where accuracy is measured by how well the chosen action reflect the correct answer (Marsland, 2015). In machine learning the computer is learning from data, which is used in situations where we don't have an analytic solution, but we do have data that we can use to construct an empirical solution (Abu-Mostafa, et al., 2012).

#### **2.1.1 Types of learning**

The main types of learning are: supervised learning, reinforcement learning and unsupervised learning (Russel & Norvig, 2010). In supervised learning the machine learning model is provided by a labelled dataset where each set of inputs are marked with the correct answer. In reinforcement learning the ML model is not provided with an exact answer, but rather an indication of whether or not its action was good. Finally, in unsupervised learning a ML model is tasked with making sense of the inputs without any feedback on what the correct answer is. Evolutionary learning is also seen as a form of learning, a process inspired by evolutionary biology (Marsland, 2015).

### 2.1.2 Popular approaches to ML

The most popular approaches to machine learning are artificial neural networks and genetic algorithms (Negnevitsky, 2011).

An **Artificial Neural Network** (ANN) seeks to imitate the way our brains work based on a hypothesis from neuroscience that mental activity primarily consists of electrochemical activity in networks of brain cells called neurons (Russel & Norvig, 2010). A simple mathematical model devised by McCulloch and Pitts (McCulloch & Pitts, 1943) is shown in Figure 2.1.

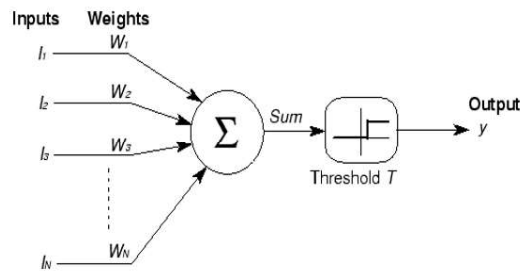


Figure 2.1 A simple mathematical model for a neuron

In short, the neuron activates when a linear combination of its inputs  $I_1, I_2, \dots, I_n$  exceeds some threshold  $T$ . A neural network is just a collection of units connected together where the properties of the network are determined by its topology and the properties of the neurons (Russel & Norvig, 2010).

A **Genetic Algorithm** (GA) is a computational approximation to how evolution performs search, which is by producing modifications of the parent genomes in their offspring and thus producing new individuals with different fitness (Marsland, 2015). While there is no universally accepted definition, most of the methods called “GAs” have at least the following components: population of chromosomes, selection according to fitness, crossover to produce new offspring, and random mutation of new offspring (Mitchell, 1998).

### 2.1.3 Hyperparameters

A common trait across all ML methods is that they are parameterized by a set of hyperparameters, denoted by  $\lambda$ , which must be set appropriately by the user to maximize the usefulness of the approach (Claesen & De Moor, 2015). Examples of such hyperparameters in ANNs are number of layers and number of neurons in each layer, learning rate and threshold function.

## 2.2 Other intelligent systems

### 2.2.1 Decision Tree

A decision tree is a prediction algorithm using recursive binary partitioning of the data space, fitting a simple prediction model within each partition, to create a predictive model (Loh, 2011). There is a wide variety of decision tree algorithms available, one of these is CART – Classification and Regression Trees (Breiman, et al., 1984). As a result of the binary partitioning, the algorithm can be represented graphically as a decision tree. An example is given in Figure 2.2 with the partitioned dataset shown on the left and the decision tree structure shown on the right.

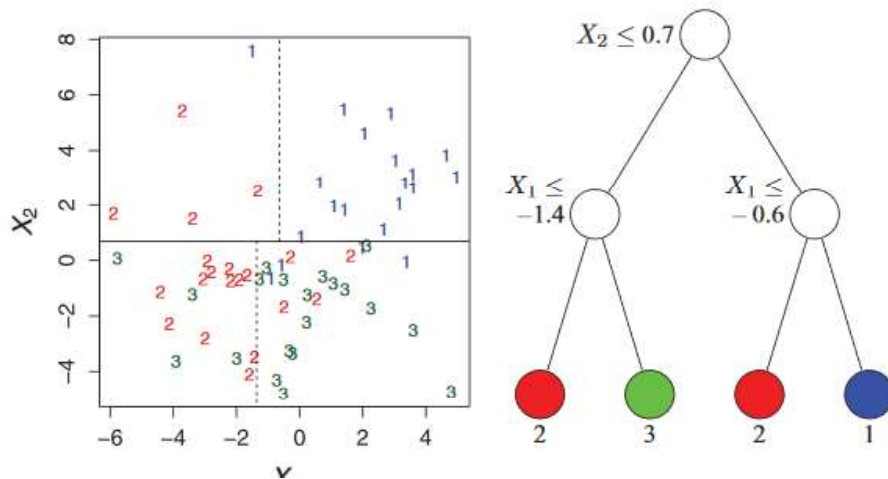


Figure 2.2 Partitions (left) and decision tree structure (right) (Loh, 2011).

### **2.2.2 Rule-based expert system**

A rule-based expert system is a knowledge-based system where knowledge is represented by simple IF-THEN rules. These rules are determined by a domain expert, which means that the system is only as good as a human expert (Negnevitsky, 2011). A general rule-based expert system consists of a knowledge base where the rules are stored, a database containing the known facts of a given problem, and an inference engine that can infer an answer based on the rules and the facts in its database (Buchanan & Duda, 1982). In addition to these components, an expert system typically has some kind of explanation facility that can explain how it arrived at the conclusion it did.

### **2.2.3 Fuzzy expert system**

Fuzzy expert systems are systems based on the same rules of logic as the rule-based expert system, but instead of using crisp values they operate on fuzzy logic. Fuzzy logic was first introduced by the Polish philosopher and logician Lukasiewicz (Lukasiewicz, 1930) whose work led to an inexact reasoning technique often referred to as possibility theory. Another philosopher, Max Black, argued that continuum implied degrees and not just true or false (Black, 1937). As an example, Black said to imagine a line of countless chairs where at one end there was a Chippendale. Next to it is a very similar chair, in fact so similar that they are indistinguishable by the bare eye. Further along the line are less and less chair like items until it finally ends with a log. Black asked at what point on the line is the item in question no longer a chair? Professor Lotfi A. Zadeh later rediscovered this fuzziness and published his famous paper *Fuzzy Sets* (Zadeh, 1965). Zadeh's work extended Lukasiewicz's possibility theory into a formal system of mathematical logic and introduced a new concept for applying natural language terms (Negnevitsky, 2011). As opposed to Boolean logic, which has only two values, fuzzy logic is multi-valued and has degrees of membership.



Fuzzy inference, according to Negnevitsky (2011), can be defined as a process of mapping from a given input to an output, using the theory of fuzzy sets. There are two fuzzy inference techniques: Mamdani and Sugeno.

The Mamdani method is the most commonly used fuzzy inference technique, applied by London University Professor Ebrahim Mamdani (Mamdani & Assilian, 1975). The Mamdani-style fuzzy inference method consists of four steps: fuzzification of the inputs, rule evaluation, aggregation of the rule outputs, and defuzzification. During the fuzzification step the inputs, which are crisp values, are mapped to membership degrees for their respective fuzzy sets. In the rule evaluation step the fuzzified inputs are applied to the antecedents of the fuzzy rules. After all the relevant rules has been applied the results are aggregated into a single fuzzy set in the third step. Finally, in the fourth step the output from the rule aggregation step is defuzzified to a single number. While there are several methods of performing defuzzification (Cox, 1998), the most popular one is probably the centroid technique (Negnevitsky, 2011). This technique finds a point where a vertical line would split the set into two equal masses, a centre of gravity (COG).

$$COG = \frac{\sum_{x=a}^b \mu_A(x)x}{\sum_{x=a}^b \mu_A(x)}$$

The basic structure of a Mamdani-type fuzzy inference method is shown in Figure 2.3.

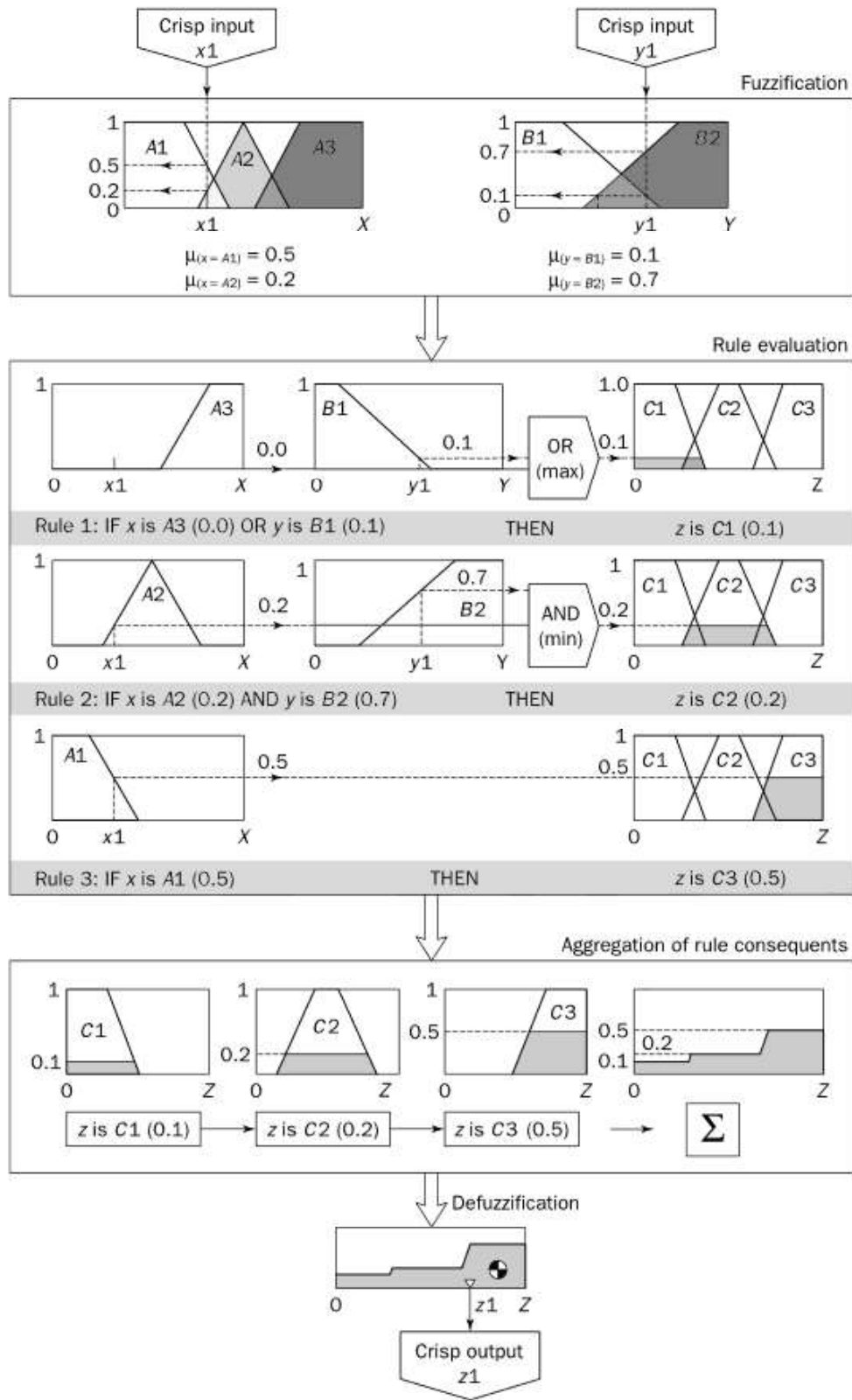


Figure 2.3 The basic structure of Mamdani-style fuzzy inference (Negnevitsky, 2011, p. 108).

The Sugeno method of fuzzy inference offers a less precise but more computationally efficient method, introduced by Michio Sugeno (Sugeno, 1985). Sugeno's method uses a spike, or singleton, as the membership function of the rule consequent. Sugeno-type fuzzy inference is very similar to the Mamdani-type, only the rule consequent is different. The output is calculated by finding a weighted average (WA) of the singletons (Negnevitsky, 2011, p. 113):

$$WA = \frac{\mu(k1) * k1 + \mu(k2) * k2 + \mu(k3) * k3}{\mu(k1) + \mu(k2) + \mu(k3)}$$

The basic structure of a Sugeno-type fuzzy inference method is shown in Figure 2.4.

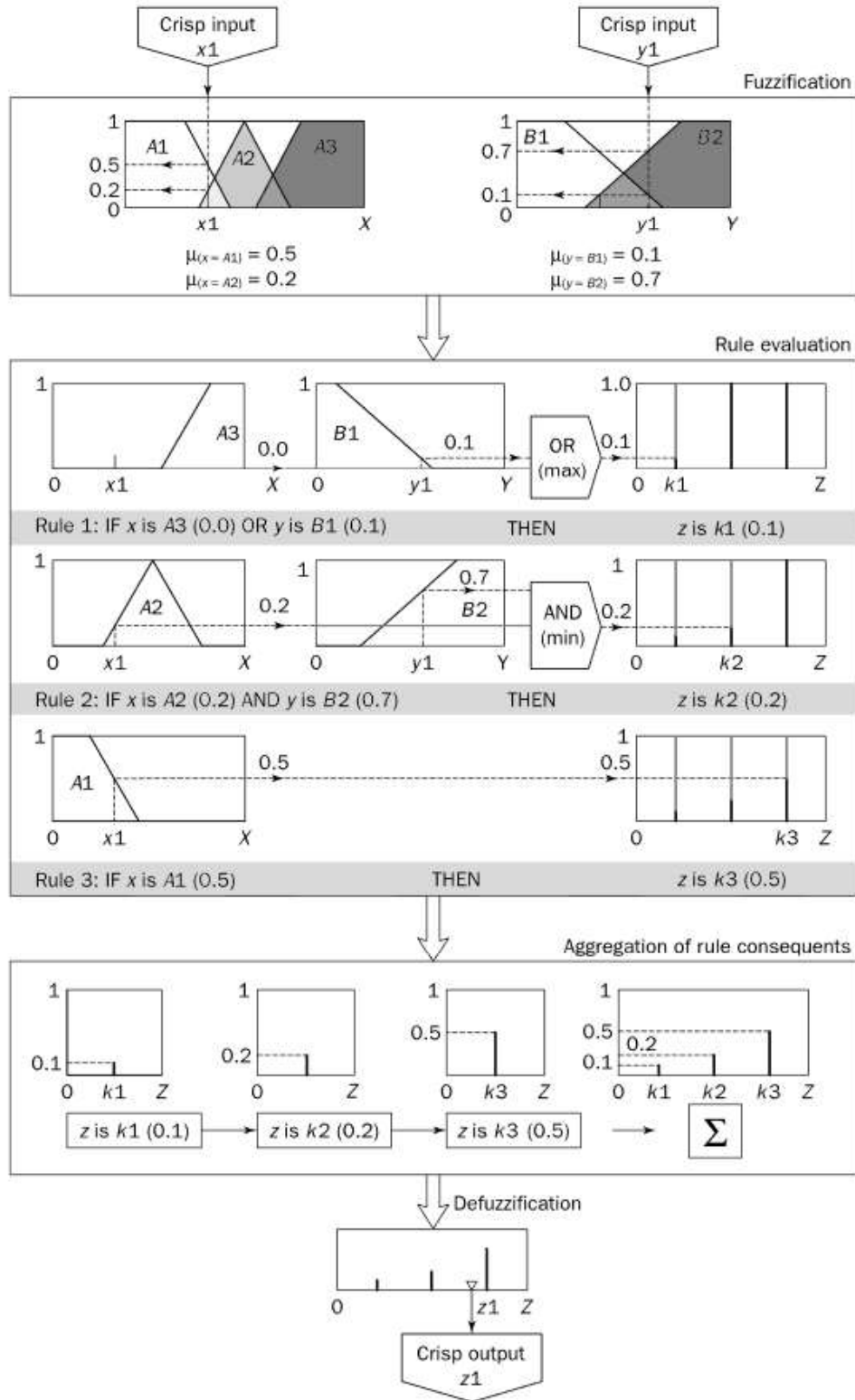


Figure 2.4 The basic structure of Sugeno-style fuzzy inference (Negnevitsky, 2011, p. 114).

## 2.3 ML methods used for predicting pathogens

Currently the main machine learning models used for predicting pathogens in drinking water supplies are: Adaptive Neuro-Fuzzy Inference System (ANFIS), Random forest, Support Vector Machine (SVM), and Extreme Learning Machine (ELM).

### 2.3.1 ANFIS

ANFIS is a hybrid intelligent system integrating artificial neural network and fuzzy logic principles. ANFIS was proposed by Roger Jang (1993) and is a neural network functionally equivalent to the Sugeno fuzzy inference model (Ch. 2.2.3). The ANFIS architecture, ref. Figure 2.5, usually has five or six layers depending on whether the inputs are counted as a layer. A short description of each layer follows (Jang, 1993; Negnevitsky, 2011; Suparta & Alhasa, 2016):

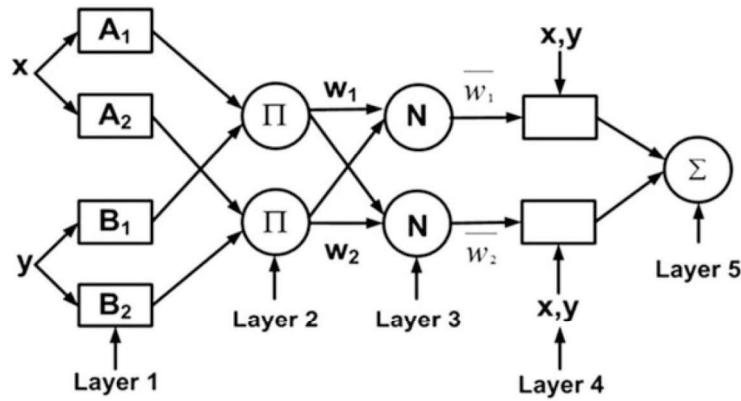


Figure 2.5 ANFIS Architecture (Suparta & Alhasa, 2016, p. 12)

The first layer, often called the fuzzification layer, has square nodes representing membership functions. In Jang's model, these neurons have a bell activation function. The output from this layer is the membership degrees of each input.

The second layer, often called the rule layer, consists of circular nodes labelled  $\pi$ . These neurons multiply their input and outputs the product. Each of these nodes represent the firing strength of a rule.

The third layer, often called the normalisation layer, has circular nodes labelled  $N$ . Each neuron in this layer receives the output from all the neurons in the previous layer and calculates a normalised firing strength of a given rule.

The fourth layer is often called the defuzzification layer. In this layer the neurons receive the normalised output from the previous layer, as well as the initial inputs, and calculates a weighted consequent value of a given rule.

The fifth and final layer is a single summation neuron. This neuron calculates the sum of all the outputs from the defuzzification layer and produces the overall output.

### **2.3.2 Random forest**

Random forest is the idea of using several decision trees (2.2.1) with variation for either classification or regression (Breiman, 2001). It works by creating a number of trees trained on slightly different data. To achieve this, bootstrap samples are taken from the dataset for each tree (Marsland, 2015). In addition, the number of features a tree can choose from is reduced.

### **2.3.3 Support Vector Machine**

SVM is a very popular algorithm in modern machine learning introduced by Vapnik in 1995 (Vapnik, 1995). The SVM attempts to separate classes in a dataset with a hyperplane. Optimal separation is given by the margin, labelled  $M$ , which is the largest possible region around the separating hyperplane with no data points entering the region (Negnevitsky, 2011).

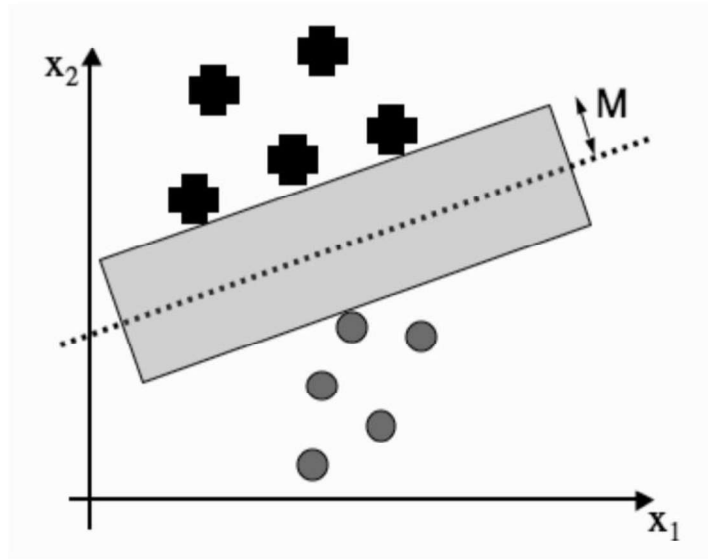


Figure 2.6 Hyperplane with margin separating two classes (Negnevitsky, 2011, p. 171).

The classifier that has the largest margin is called the maximum margin linear classifier, and the data points closest to the classification line is called support vectors. To achieve linear separability, it is often necessary to transform the data. The kernel function provides a solution by adding an additional dimension (Noble, 2006).

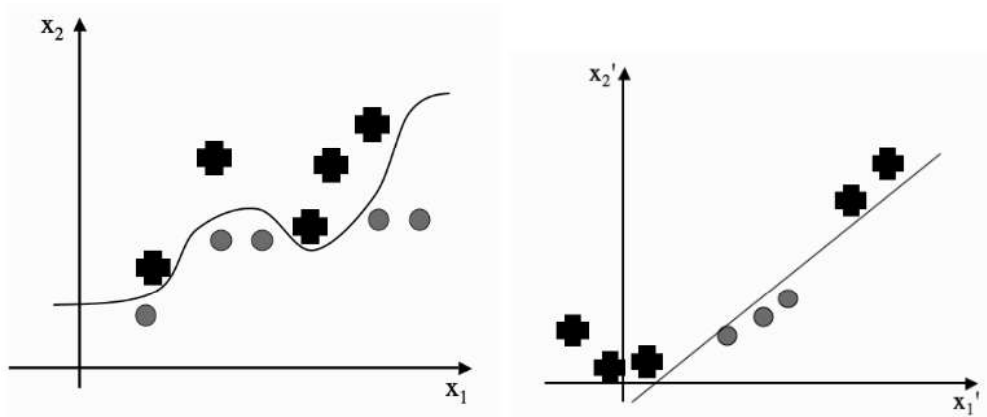


Figure 2.7 Modified space to achieve linear separability (Negnevitsky, 2011, p. 176).

### 2.3.4 Extreme Learning Machine

ELM is an algorithm for single-hidden layer feedforward neural networks (SLNFs) which randomly chooses the input weights and analytically determines the output weights of SLNFs (Huang, et al., 2005).

## ***2.4 Hyperparameter optimization***

Hyperparameter optimization, also called hyperparameter tuning or hyperparameter search, is the problem of finding the optimal set of hyperparameters for a given machine learning algorithm. According to Claesen and De Moor (Claesen & De Moor, 2015) this has commonly been performed by manually, either via rules-of-thumb (Hinton, 2012; Hsu, et al., 2003) or by testing sets of hyperparameters on a predefined grid (Pedregosa, et al., 2011). However, these methods are lacking in terms of reproducibility and proves to be quite impractical when the number of hyperparameters increase (Claesen, et al., 2014). For these reasons, automated hyperparameter search has gained increasing amounts of attention in machine learning over the years.

### **2.4.1 Random search**

Claesen and De Moor (Claesen & De Moor, 2015) lists several optimization methods that have been applied for hyperparameter tuning: particle swarm optimization (Lin, et al., 2008), genetic algorithms (Tsai, et al., 2006), coupled simulated annealing (Souza, et al., 2010) and racing algorithms. (Birattari, et al., 2010).

### **2.4.2 Bayesian optimization**

Bayesian optimization, or sequential model-based optimization (SMBO), is a general term for techniques that involve Bayesian methods for optimizing a function (Groch, et al., 1981; Bergstra, et al., 2013). These algorithms are typically used for applications where it is expensive to evaluate the function because these methods minimize the number of evaluations (Bergstra, et al., 2013).



## ***2.5 Microbial pathogens in water supply systems***

The World Health Organization (WHO) is constantly working on a framework for assessing and improving the quality of drinking water (World Health Organization, 2011). In the WHO publication *Guidelines for Drinking-water quality* (World Health Organization, 2011) they consider the drinking-water quality with respect to microbial- and chemical water quality. The relevant part for this thesis is the microbial water quality.

### **2.5.1 Indicator organisms**

Traditionally, detection of pathogens in water bodies has been done with the use of indicator organisms (Berg, 1978). Smith (1895) showed that a type of bacteria, known then as *Bacillus coli communis*, could be ‘a valuable indication or symptom of pollution’, and was the first bacteriologist to promote what later became known as the coliform group of bacteria as indicators of faecal pollution (Horan, 2003). Faecal indicator organisms remain at the forefront of water and wastewater microbiology (Horan, 2003). The ideal indicator organism would be suitable for all categories of water, is present in wastewaters and polluted waters whenever pathogens are present, is present in greater numbers than pathogens, have similar survival characteristics as pathogens, is unable to multiply in waters, is non-pathogenic, and is reliably detectable in low numbers at low cost (Bonde, 1962; World Health Organization, 1993; Grabow, 1996; Godfree, et al., 1997). Naturally, such an ideal organism does not exist, but there are some who come close (Horan, 2003). World Health Organization (2011) lists several indicator organisms along with their indicator value and source, and occurrence. Among these are *total coliform bacteria*, *Escherichia coli* (*E. coli*) and *thermotolerant coliform bacteria*, and *intestinal enterococci*. These organisms are easily detectable, at low cost and within a short time span. For this reason, water treatment plants both in Norway and globally analyse for these organisms as indicators for pathogens. While

these organisms have proven useful in indicating bacterial pathogens in water, difficulties still remain in detecting non-bacterial pathogens like viruses and protozoa (Horan, 2003).

### 2.5.2 Pathogens of concern for Norwegian drinking water

According to a report from the Norwegian Institute of Public Health (Andersen, 2016) the most common outbreak of pathogens in food and water are norovirus, *Campylobacter* and *Cryptosporidium*. In 2016 one outbreak and several incidents were confirmed to be waterborne. Between 2012 and 2016 there were 8 outbreaks confirmed to be waterborne. Another report from the Norwegian Institute of Public Health (Herrador, et al., 2016) shows 4 outbreaks since 1992 with over 1000 incidents. The most common sources of infection were norovirus and *Campylobacter*, and there was one large outbreak of *Giardia* in Bergen in 2004.

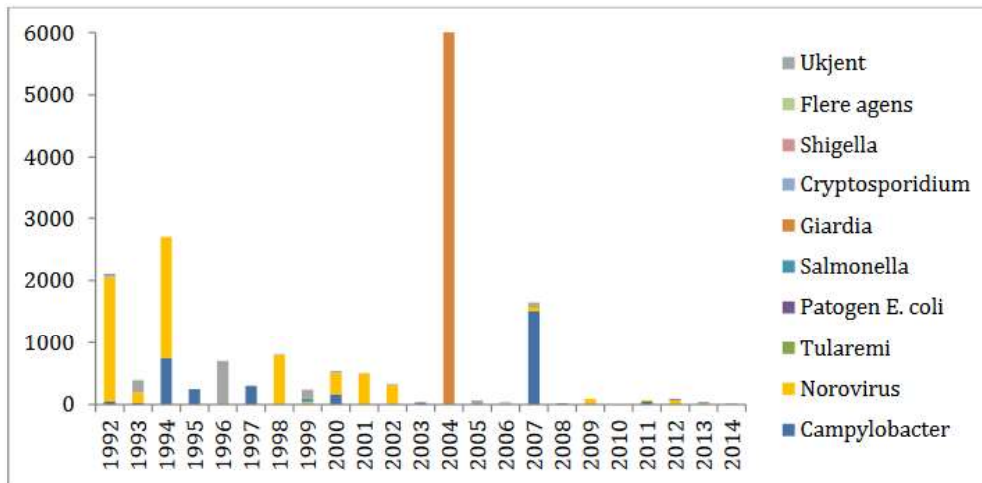


Figure 2.8 Number of incidents from outbreaks in Norway 1992-2014 (Andersen, 2016, p. 9)

*Campylobacter* is a genus of bacteria, *Cryptosporidium* and *Giardia* are protozoa, and norovirus is a virus. Most of these pathogens remain hard to detect and prevent due to the aforementioned problem of lack of easily detectable indicator organisms. Testing for these pathogens are typically expensive and takes longer time because they need more extensive lab tests. Likewise, the availability of data on these

organisms are severely lacking since water treatment plants typically does not gather such expensive data.

### **3 Methodology**

This chapter presents the approach used during this thesis. Several models have already been made by others for the purpose of predicting indicator organisms in drinking water. As previously mentioned, the idea of this thesis is to further tune these models in an attempt to make them better suitable for their purpose. This was done by experiments with hyperparameter optimization.

#### ***3.1 Set-up***

The experiments were done in Python using Hpopt-Sklearn (Komer, et al., 2014) which is built on the Scikit-Learn (Pedregosa, et al., 2011) and Hyperopt (Bergstra, et al., 2013) libraries. The data is stored in CSV files and is loaded in python using pandas (McKinney, 2011). Trained models are saved as “.sav”-files using joblib which is included in the scikit-learn library. Joblib can also load the models for later use.

#### ***3.2 Pre-study***

A short pre-study was conducted on the data and on the existing models before the experiments started. The study of the data was mostly conducted working in a Jupyter Notebook (Kluyver, et al., 2016) using the pandas library. The study included analysing the frequency of measurements, looking for empty or NaN values, plotting the observed values and visually inspecting them, performing a correlation analysis of the data, and determining the integrity of the data by looking at the values.

#### ***3.3 Experiments***

Using the set-up explained in section 3.1 a set of experiments were systematically run. Each experiment begun with the selection of a learner and a pre-processor from the list of available ones. An overview of the overall methodology is shown in Figure 3.1.

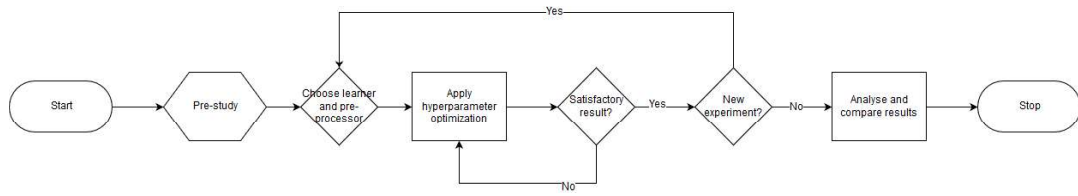


Figure 3.1 Methodology flow chart

A more specific chart of the experiments themselves are shown in Figure 3.2. Random Forest and Support Vector Machine were specifically chosen as learners because they have been used in previous attempts (Mohammed, et al., 2018). Other learners were chosen by the optimizer.

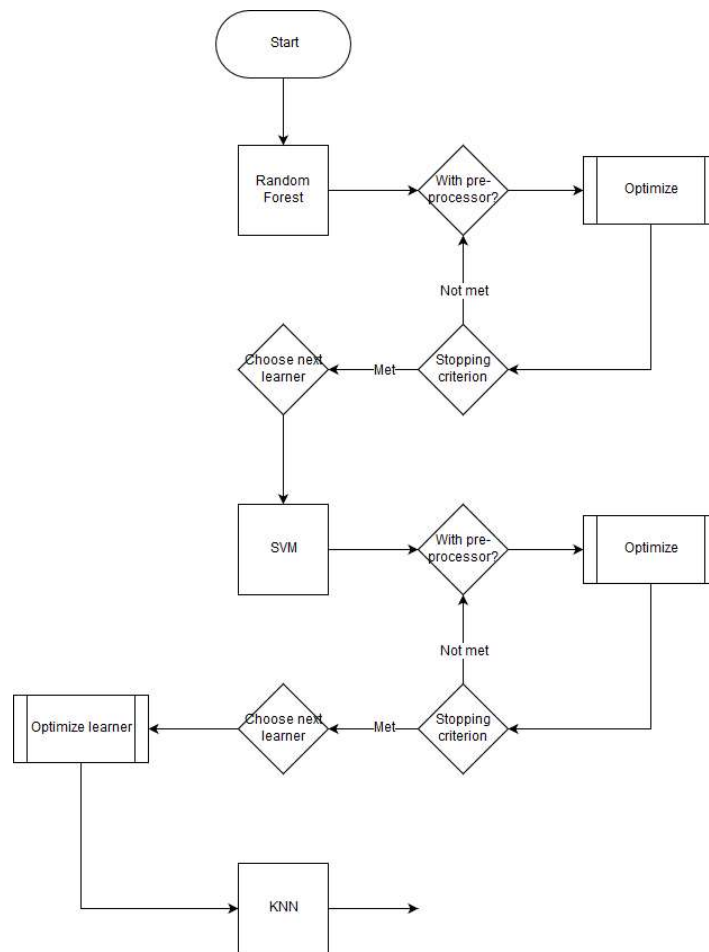


Figure 3.2 Experiments flow chart

The optimizer was configured to test 100 different configurations for each experiment. Each configuration was trained on the data until it met its internal stopping criterion. When choosing a pre-processor, the optimizer could choose from all available pre-processors the first few

experiments. Later, to reduce the search space, a specific pre-processor that has shown to give good results was chosen specifically and optimized together with the learner.

Each configuration was optimized at least 3 times. If, by visual inspection, a learner pre-processor combination did not return any usable result, it could be run several more times to ensure that it is not able to produce any good results.

Pre-processors that has been used is explained in the following sections. There were more pre-processing methods available in the library, like TfidfVectorizer and OneHotEncoder, but these are the ones that were actually used.

### 3.3.1 Principal Component Analysis

PCA is a method derived from statistics. Essentially it is linear dimensionality reduction using Singular Value Decomposition of the data to project it to a lower dimensional space (Scikit-Learn, 2017).

### 3.3.2 Standard Scaler

Standard scaler is a pre-processing method that transforms the data such that the distribution has a mean of 0 and a standard deviation of 1. If the mean  $\mu$  is given as

$$\mu = \frac{\sum_{i=1}^n x_i}{n}$$

And the standard deviation  $\sigma$  is

$$\sigma = \sqrt{\frac{\sum_{i=1}^n (x_i - \mu)^2}{n}}$$

Then standardization  $z$  is given as:

$$z = \frac{x - \mu}{\sigma}$$

### 3.3.3 Min-Max Scaler

Min-Max scaling transforms features by scaling each of them to a given range. The scaling is given by

$$X_{scaled} = X_{std} * (max - min) + min$$

If min and max is the feature range and  $X_{std}$  is

$$X_{std} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

## 3.4 Model comparisons

By the end of the experiments all the best models from each configuration were compared to each other. Models are compared by two methods: using the built-in scoring metrics in the scikit-learn library, and visually comparing their accuracy by plotting the predicted values and comparing it to a plot of the observed values.

Following is a description of the built-in metrics (Scikit-Learn, 2017):

### 3.4.1 Mean Absolute Error

The scoring metric used in the first comparison is the Mean Absolute Error. MAE is a measure of the average vertical distance between the predictions and the observed values. If  $\hat{y}_i$  is the predicted value of the  $i$ -th sample, and  $y_i$  is the corresponding observed value, then the MAE estimated over  $n$  samples is defined as

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

### 3.4.2 Mean Squared Error

Mean Squared Error, the second scoring metric used, is quite similar to MAE. The only difference is that the error is squared.

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

Squaring the error means that greater deviations is punished more than smaller deviations from the observed values.

### 3.4.3 Median Absolute Error

The Median Absolute Error is the median of all absolute differences between predicted and observed values.

$$MedAE = median(\sum_{i=1}^n |y_i - \hat{y}_i|)$$

MedAE is interesting because it is robust to outliers.

### 3.4.4 Explained Variance

Explained Variance is calculated by:

$$EV = 1 - \frac{Var\{y - \hat{y}\}}{Var\{y\}}$$

The best possible score is 1.0 where lower values are worse.

### 3.4.5 R<sup>2</sup> score

The R<sup>2</sup> score, also called the coefficient of determination, provides a measurement of how well future samples are likely to be predicted by the model.

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

Like EV the best possible score is 1.0 and can be negative because models can be arbitrarily worse.



## 4 Results

In this chapter all the results are presented. Firstly, the results from the study of the data, including correlation analyses and distribution of the measurements, is presented. Then the results from the hyperparameter optimization experiments are presented along with a comparison of the various models' performance scores.

### 4.1 The Data

The data used for this thesis is from Maridalsvannet in Oslo and Brusdalsvatnet in Aalesund.

#### 4.1.1 Maridalsvannet

The data from Maridalsvannet has 6 values and 4 labels: pH, temperature, conductivity, turbidity, colour, alkalinity, coliform bacteria, *E. coli*, *intestinal enterococci* and *Clostridium perfringens*. The data is indexed by the date it was measured. The frequency of measurement varies from 4 days to 21 days at most.

**Error! Reference source not found.** Table 4.1 is the result from a correlation analysis of the data from Maridalsvannet.

	pH	Temp	Cond	Turb	Color	Alk	Coliform	Ecoli	Int	CIPerf
pH	1.000000	-0.242692	-0.302789	0.131410	0.254255	0.284041	-0.195577	0.060660	0.083013	0.089076
Temp	-0.242692	1.000000	-0.265657	0.014205	0.189781	0.053020	0.288676	0.180323	0.207606	0.098679
Cond	-0.302789	-0.265657	1.000000	0.006289	-0.724396	-0.395128	-0.054450	-0.097312	-0.243094	-0.235224
Turb	0.131410	0.014205	0.006289	1.000000	-0.187929	-0.255866	0.033801	0.042158	0.123178	0.029312
Color	0.254255	0.189781	-0.724396	-0.187929	1.000000	0.672443	0.031702	0.036532	0.051683	0.187595
Alk	0.284041	0.053020	-0.395128	-0.255866	0.672443	1.000000	0.010181	-0.023871	-0.005203	0.112252
Coliform	-0.195577	0.288676	-0.054450	0.033801	0.031702	0.010181	1.000000	0.299571	0.079823	0.087543
Ecoli	0.060660	0.180323	-0.097312	0.042158	0.036532	-0.023871	0.299571	1.000000	0.246538	0.151833
Int	0.083013	0.207606	-0.243094	0.123178	0.051683	-0.005203	0.079823	0.246538	1.000000	0.214801
CIPerf	0.089076	0.098679	-0.235224	0.029312	0.187595	0.112252	0.087543	0.151833	0.214801	1.000000

Table 4.1 Correlation analysis on the data from Maridalsvannet

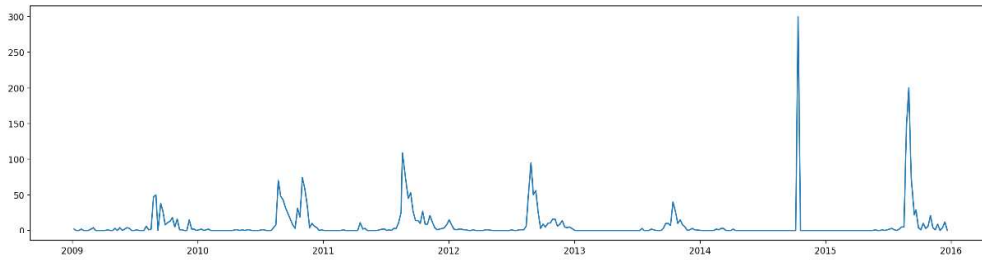


Figure 4.1 Coliform Bacteria in Maridalsvannet

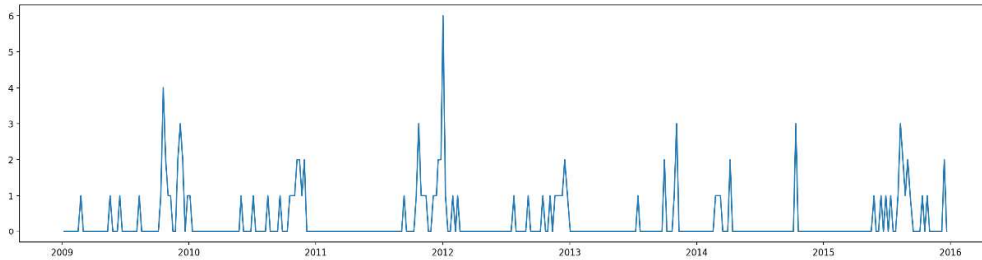


Figure 4.2 E. Coli in Maridalsvannet

Figure 4.3 is a box plot of the values in the data from Maridalsvannet. Save for alkalinity, all the variables have non-zero medians (green lines) and a distribution to some degree.

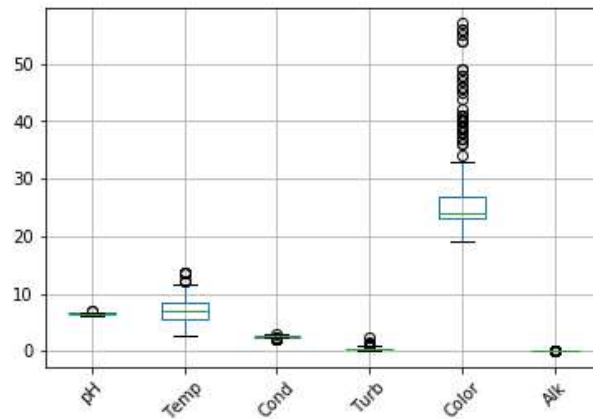


Figure 4.3 Box plot of the values from Maridalsvannet

#### 4.1.2 Brusdalsvatnet

The data from Brusdalsvatnet has 5 values and 4 labels: colour, turbidity, pH, conductivity, alkalinity, coliform bacteria, thermotolerant coliform bacteria, E. coli and intestinal enterococci. The data is indexed by the date it was measured. The frequency of measurement varies from 1 day to 21 days at most. Unlike the data for Maridalsvannet, the data

for Brusdalsvatnet does not contain measurements of temperature. Table 4.2 is the result from a correlation analysis of the data from Brusdalsvatnet.

	Color	Turbidity	Conductivity	Alkalinity	Coliform bacteria	Termotol. Coliform bacteria	E. coli	Int. enterococci
Color	1.000000	0.395519	0.608176	NaN	-0.044545	-0.042751	0.040547	0.477463
Turbidity	0.395519	1.000000	0.290282	NaN	-0.040568	-0.034207	-0.013932	0.244682
Conductivity	0.608176	0.290282	1.000000	NaN	-0.026144	-0.039662	-0.026024	0.654542
Alkalinity	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Coliform bacteria	-0.044545	-0.040568	-0.026144	NaN	1.000000	0.358883	-0.034106	0.076619
Termotol. Coliform bacteria	-0.042751	-0.034207	-0.039662	NaN	0.358883	1.000000	-0.069763	0.012267
E. coli	0.040547	-0.013932	-0.026024	NaN	-0.034106	-0.069763	1.000000	-0.023879
Int. enterococci	0.477463	0.244682	0.654542	NaN	0.076619	0.012267	-0.023879	1.000000

Table 4.2 Correlation analysis on the data from Brusdalsvatnet

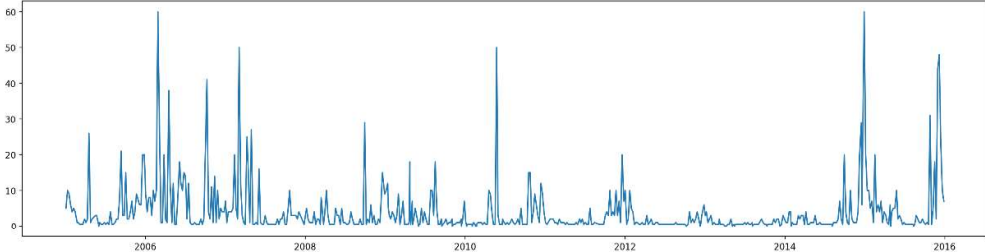


Figure 4.4 Coliform Bacteria in Brusdalsvatnet

Figure 4.5 is a box plot of the values from the Brusdalsvatnet data. For all values but the colour, the typical value is 0 (the green lines) and makes up such large portions of the data that all non-zero values are considered to be outliers (circles).

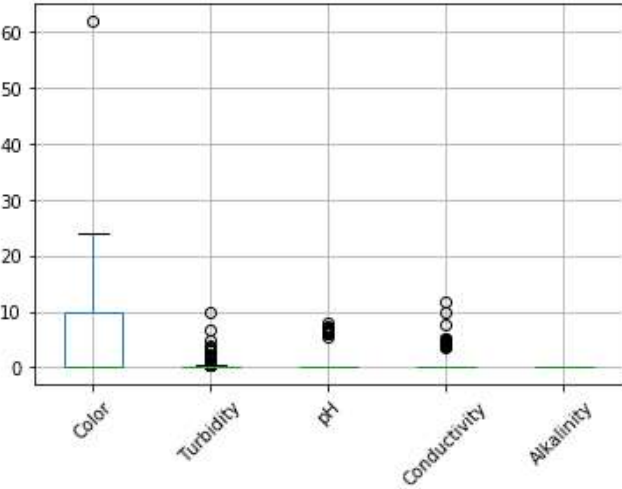


Figure 4.5 Box plots of the values from Brusdalsvatnet

## ***4.2 Web-based optimization tool***

A web-based tool for creating hyperparameter-optimized models was developed as a utility for this thesis. The tool allows uploading a file containing the measurements from water treatment plants and the training of hyperparameter-optimized models. These models are stored by the tool and can be reviewed to see the parameters used or tested on other datasets. For more information see Appendix A: Optimization tool.

## ***4.3 Hyperparameter optimization results***

The optimized and default models are compared to each other using built in scoring metrics in the Scikit-Learn library and plotting the scores as box plots.

### **4.3.1 Predicting Coliform Bacteria**

For the first set of experiments the level of coliform bacteria is predicted using pH, conductivity, temperature, colour, turbidity and alkalinity as inputs.

### **4.3.2 Coliform Bacteria: Comparison of scoring metrics**

Figure 4.6 shows a box plot of the negative mean absolute error score for optimized k-nearest neighbours, random forest and support vector machine as well as default random forest and support vector machine. According to the MAE scoring, the optimized KNN is clearly the best algorithm with the least median error and a small distribution. The default SVM comes second with a median error of around -12 compared to the KNN's -9, however, it has a wider distribution. Next comes the optimized Random Forest, the default Random Forest, and finally the optimized SVM all of which have greater median errors and wider distributions.

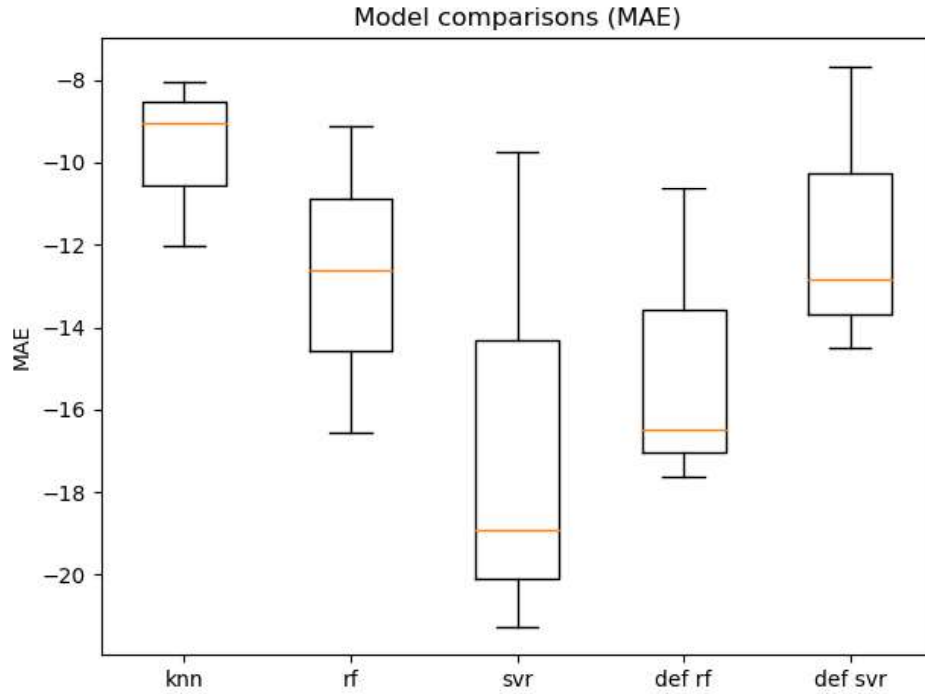


Figure 4.6 Comparing models using Mean Absolute Error

Figure 4.7 shows a box plot of the mean squared error scores for the same models. This figure mostly reflects the same results as Figure 4.6, but the greater errors are emphasized. This have improved the default random forest score relative to the others, but the order remains roughly the same as in Figure 4.6.

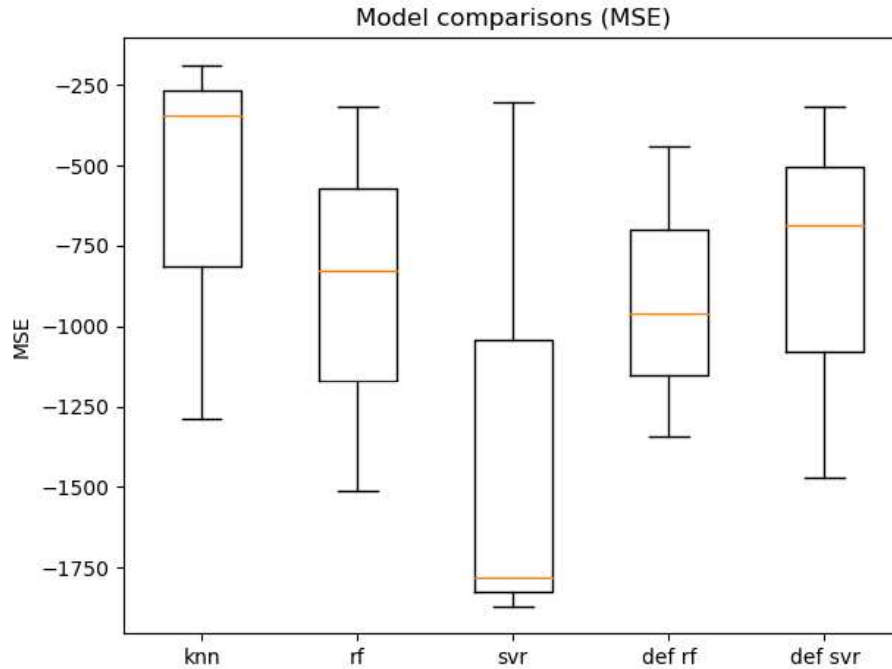


Figure 4.7 Comparing models using Mean Squared Error

Figure 4.8 shows a comparison of the models using median absolute error (MedAE). MedAE gives a slightly different ranking of the models from the previous two. Here the default random forest is the best. The default SVM and optimized random forest algorithm are similar, the default SVM has a better median but the random forest has a tighter distribution and a better lower quartile. The optimized knn is rated fourth in this list although it is better on its best score, its distribution and median are worse than that of the optimized random forest. The optimized SVM is poorest, as it scores worst among these five models although it has a relatively narrow distribution.

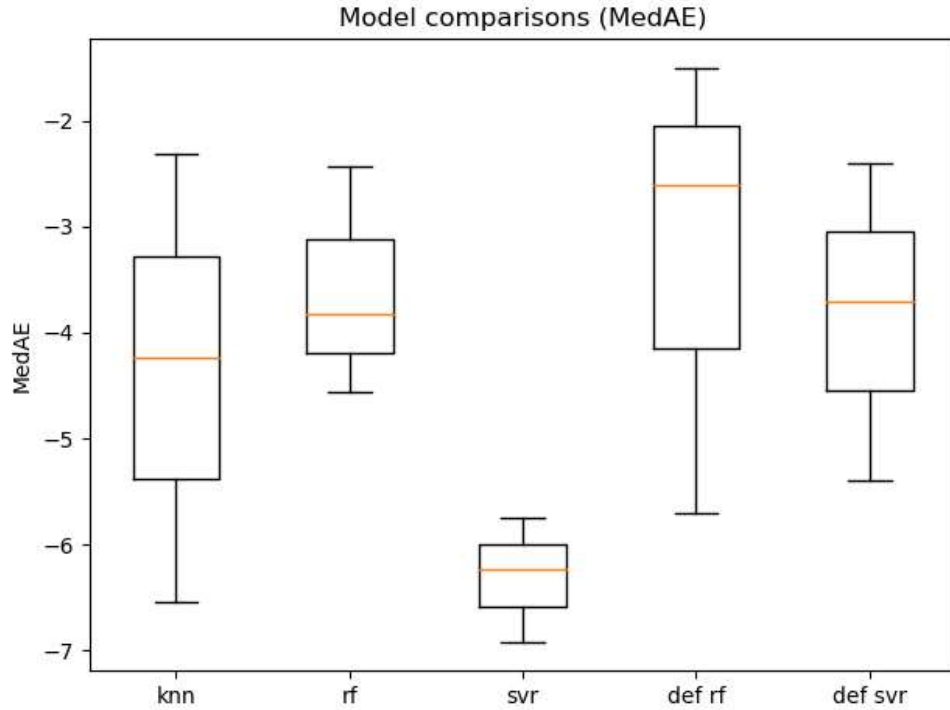


Figure 4.8 Comparing models using Median Absolute Error

Figure 4.9 and Figure 4.10 respectively compares the models using explained variance and  $R^2$  score which give quite similar results. From these results KNN is best model, followed by the default SVM, default Random Forest, optimized Random Forest, and finally optimized SVM.

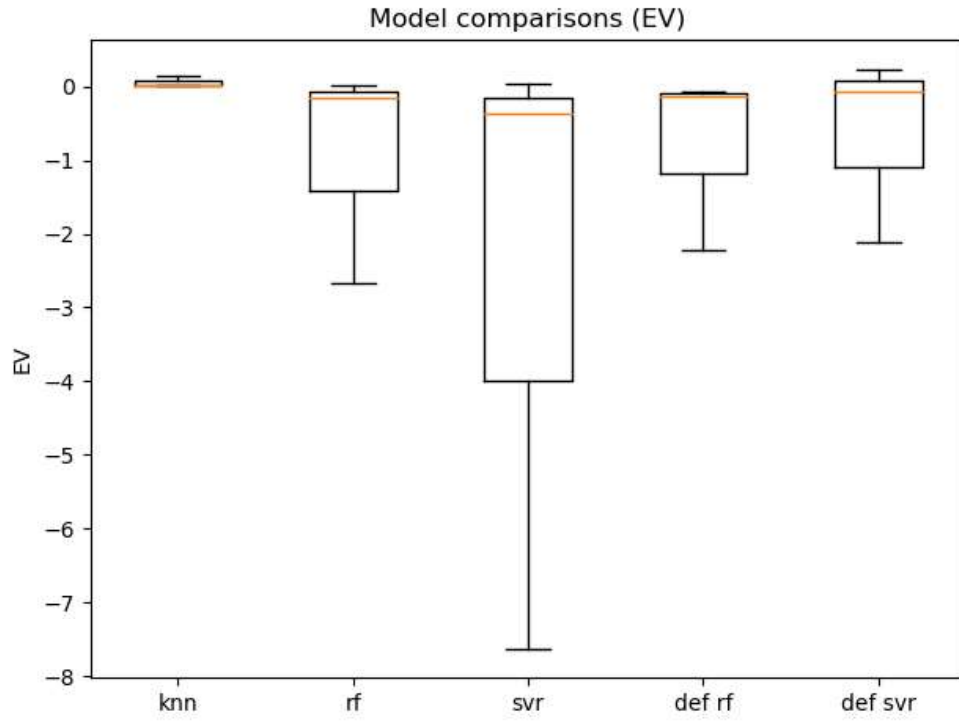


Figure 4.9 Comparing models using Explained Variance

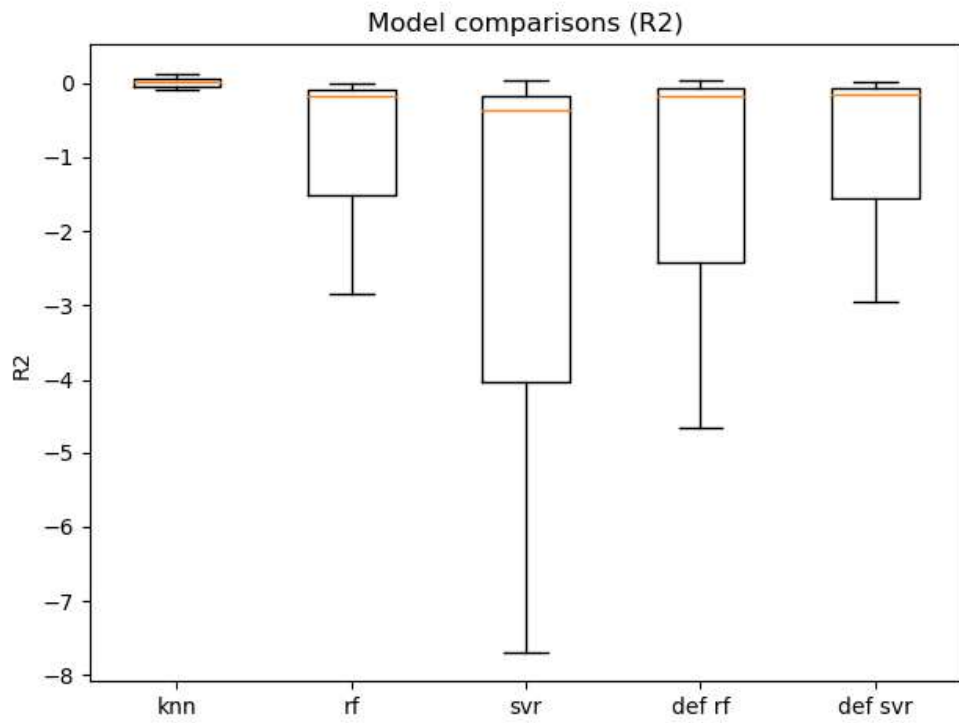


Figure 4.10 Comparing models using  $R^2$  score



According to these results it seems like the default Random Forest and SVM algorithms give the best results. However, visual comparison between observed values and the values that are predicted by the models show different results.

### 4.3.3 Coliform Bacteria: Visual comparison

In addition to model comparisons using scoring methods, the models' accuracy is compared visually by making a plot of the observed values and placing it above a plot of the values that were predicted by the trained model. Finally, a plot of the observed values overlaid by the predicted value is located at the bottom of each figure.

Figure 4.11 shows a comparison between the observed and predicted values from an optimized Random Forest model with no pre-processing on the data:

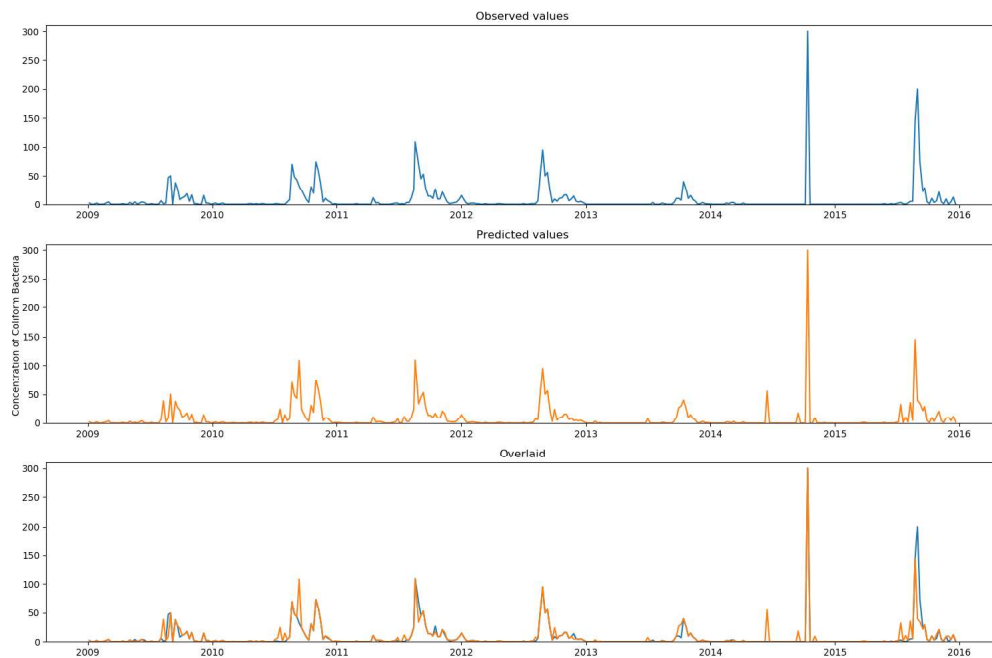


Figure 4.11 Optimized Random Forest with no pre-processing

The optimized Random Forest captures most of the spikes quite nicely and has only a few false spikes with the size of the smallest observed

peaks. In general, this trained model seems to understand the data quite well and is able to predict the spikes with few mistakes.

Figure 4.12 shows the result from an optimized SVM with no pre-processing on the data:

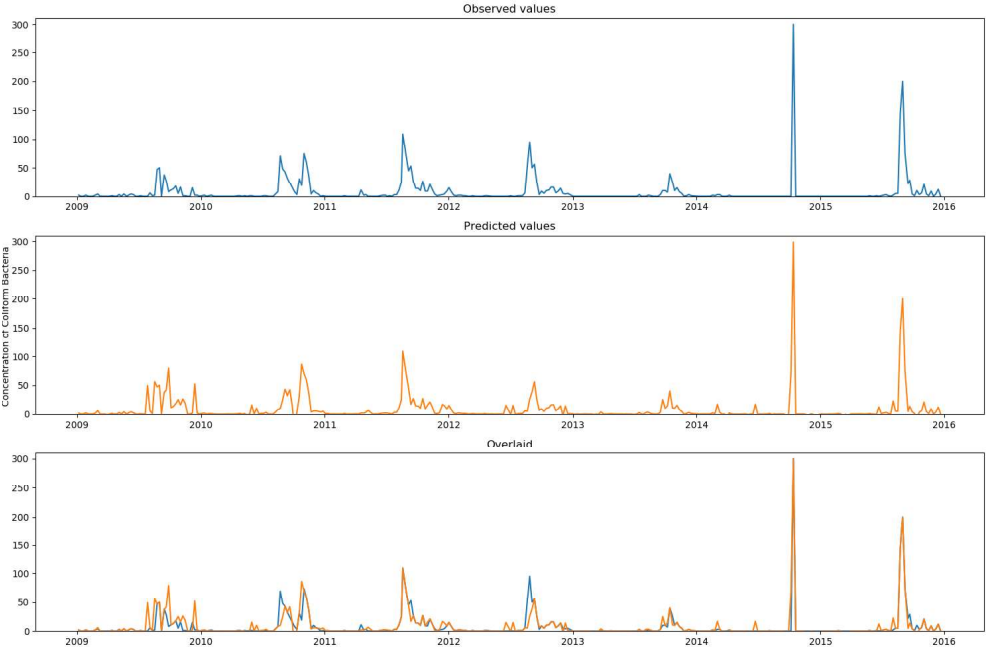


Figure 4.12 Optimized SVM with no pre-processing

Like the optimized RF, this model is able to predict most of the spikes, however, it performs slightly worse around the two first spikes. The model does predict some false positives, but they are mostly quite small.

Figure 4.13 shows the result from an optimized SVM trained on data that has been pre-processed using Principal Component Analysis. Like the SVM and RF without pre-processing, the optimized SVM also predict the spikes fairly well, but this one has a lot more noise in its predictions.

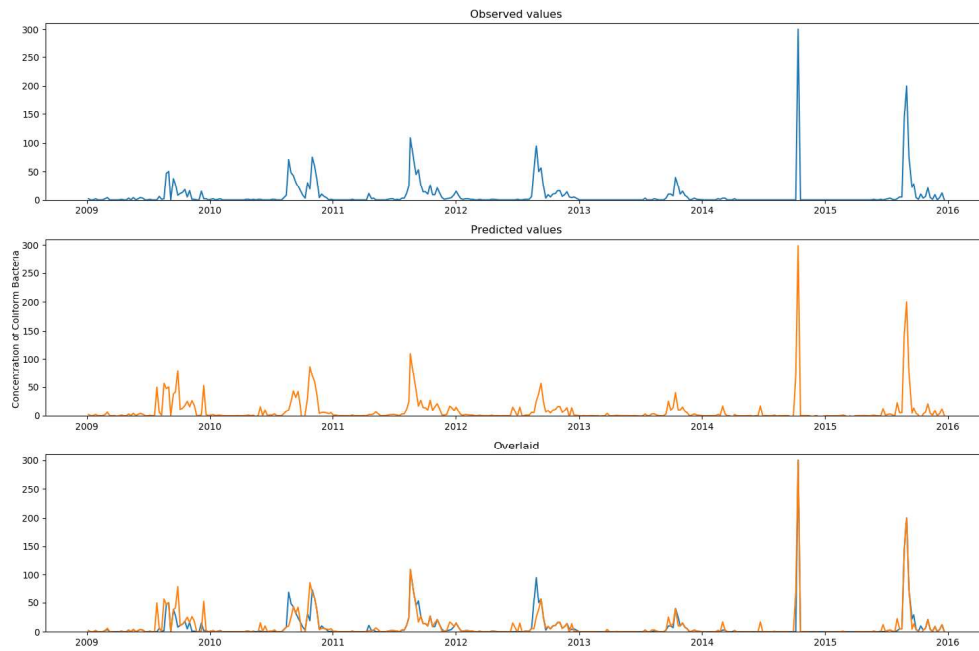


Figure 4.13 Optimized SVM pre-processed with PCA

Figure 4.14 shows the result from an optimized K-Nearest Neighbours trained on the data with no pre-processing:

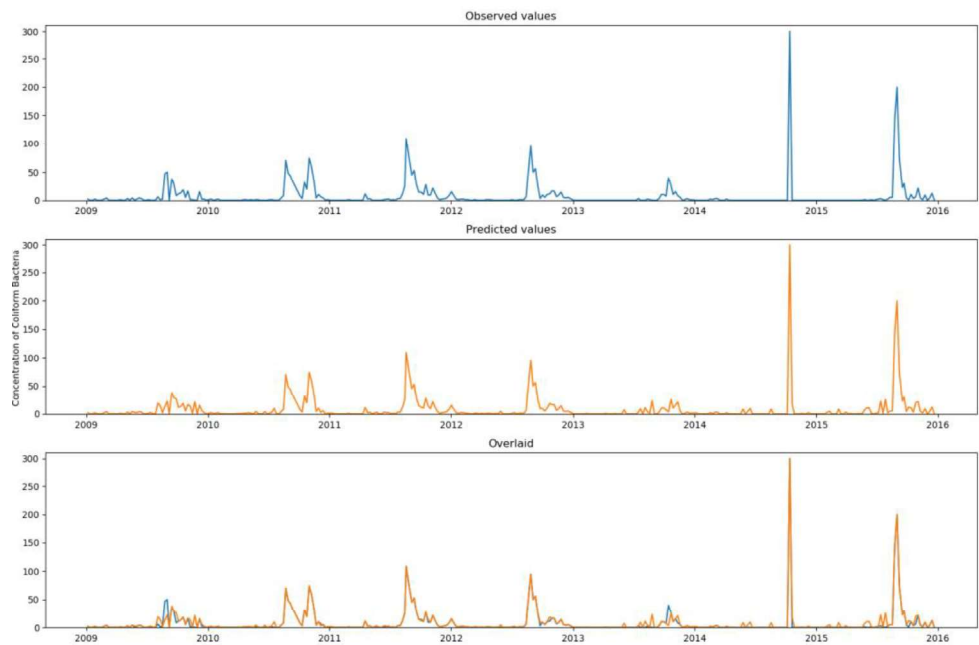


Figure 4.14 Optimized KNN with no pre-processing

Like the others so far, the optimized KNN is able to predict the spikes quite well. From this visual comparison it looks to be the best so far given that it has a relatively low level of noise and no large false positives.

Figure 4.15 shows the best result that was achieved using a Random Forest with non-optimized hyperparameters:

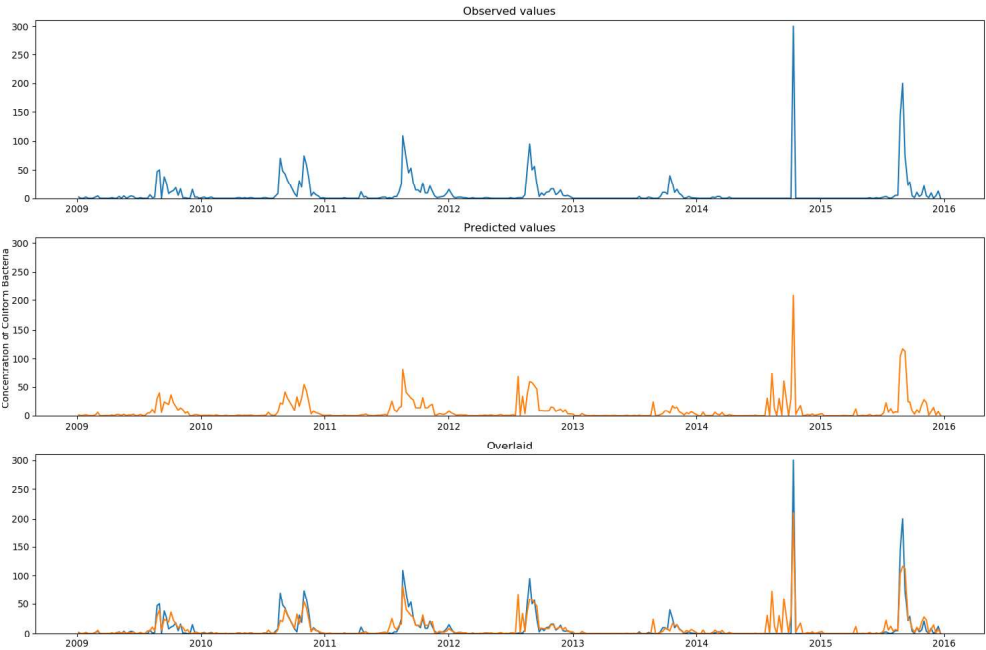


Figure 4.15 Default Random Forest

While the result is not the worst that has been seen during these experiments it is having a harder time correctly predicting the magnitude of the spikes and predicts several spikes that is not present in the observed data.

Figure 4.16 shows the best result that was achieved using a Support Vector Machine with default hyperparameters and no pre-processing on the data.

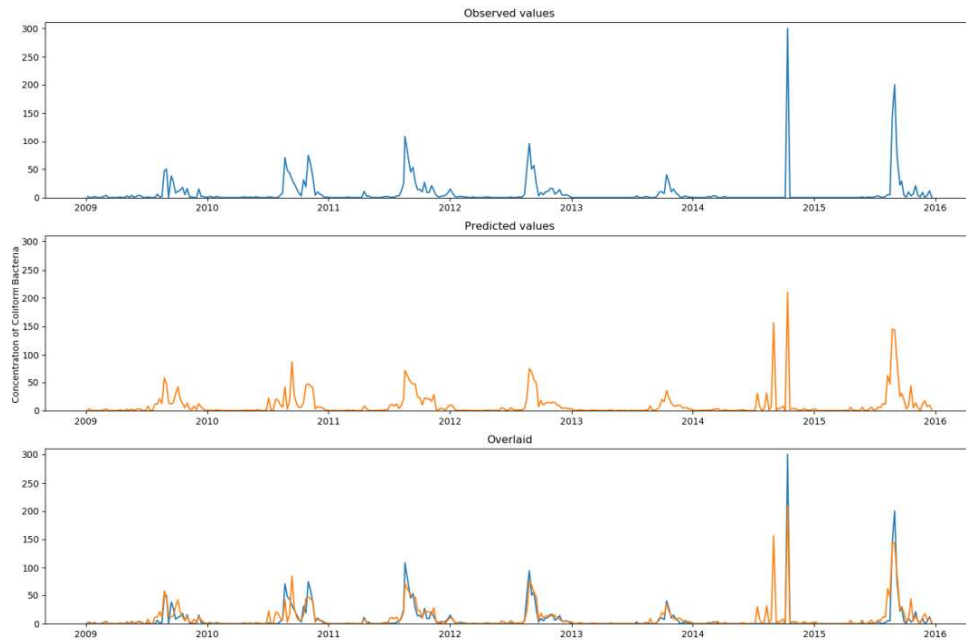


Figure 4.16 Default SVM

Much like the default RF, the default SVM is mostly able to recognize the spikes but can't quite predict their magnitude. The default SVM also predicts some spikes where there are none observed

#### 4.3.4 Predicting *E. coli*

For the second set of experiments the level of *E. coli* is predicted using the same inputs as with Coliform bacteria. The first thing to note is that the range of values on the measurements of *E. coli* is much smaller than that of Coliform Bacteria. The measurements range from 0 to 6. As can be seen later in section 4.3.6 there are not many clear peaks in the *E. coli* measurements in contrast to the Coliform bacteria measurements.

#### 4.3.5 *E. coli*: Comparison of scoring metrics

Like the models predicting Coliform Bacteria, these models are compared to each other using the built-in scoring metrics in the Scikit-learn library. Figure 4.17 is a box plot showing the distribution of the Mean Absolute Error scoring for the models predicting *E. coli*.

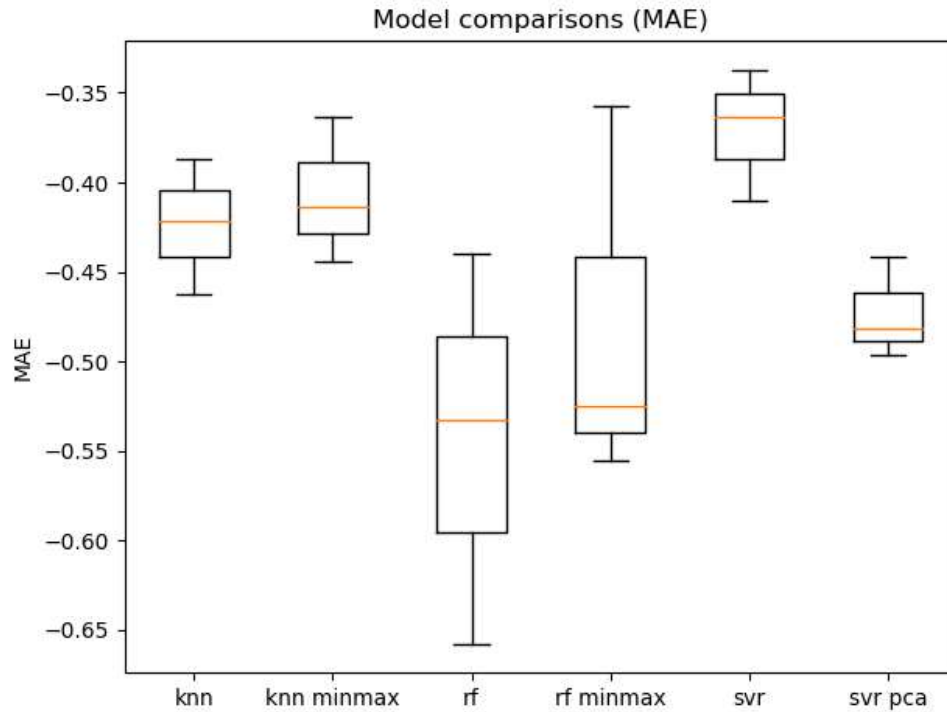


Figure 4.17 Mean Absolute Error comparison predicting *E. coli*

The Mean Absolute Error ranks the algorithms: SVM, KNN + Min-Max Scaler, KNN, SVM + PCA, and both RFs last.

Looking at Figure 4.18, which is a plot of the Mean Squared Error scoring for the same models, the results are a bit different. Interpreting this result, the KNN sores best followed by SVM + PCA, which is arguably better than KNN + Min-Max although it's median is worse, SVM, RF + Min-Max, and finally RF.

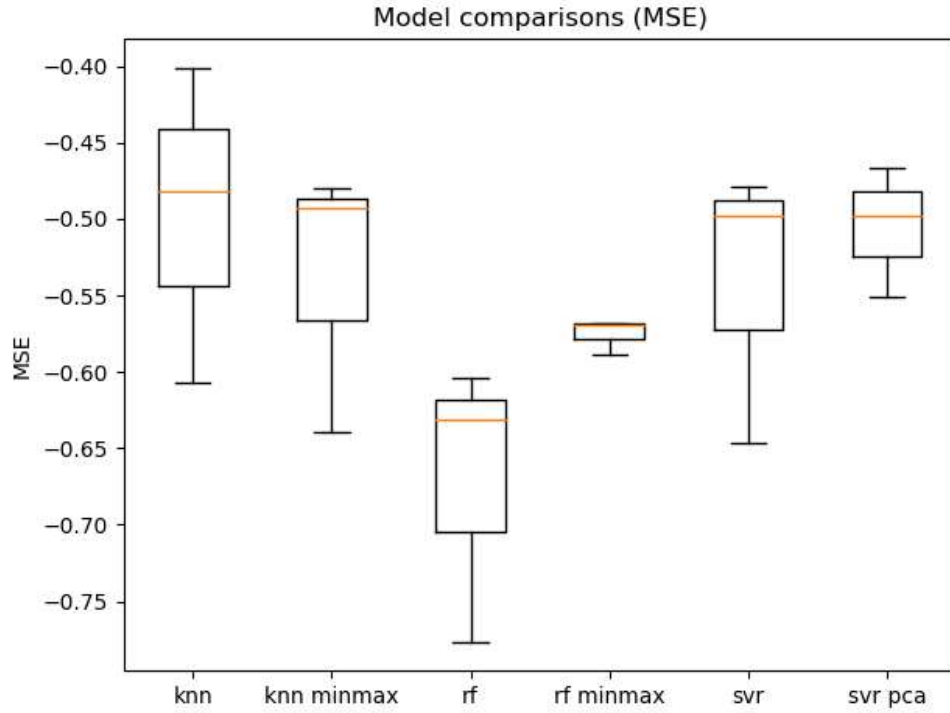


Figure 4.18 Mean Squared Error comparison predicting *E. coli*

Figure 4.19 is a box plot of the Median Absolute Error scoring:

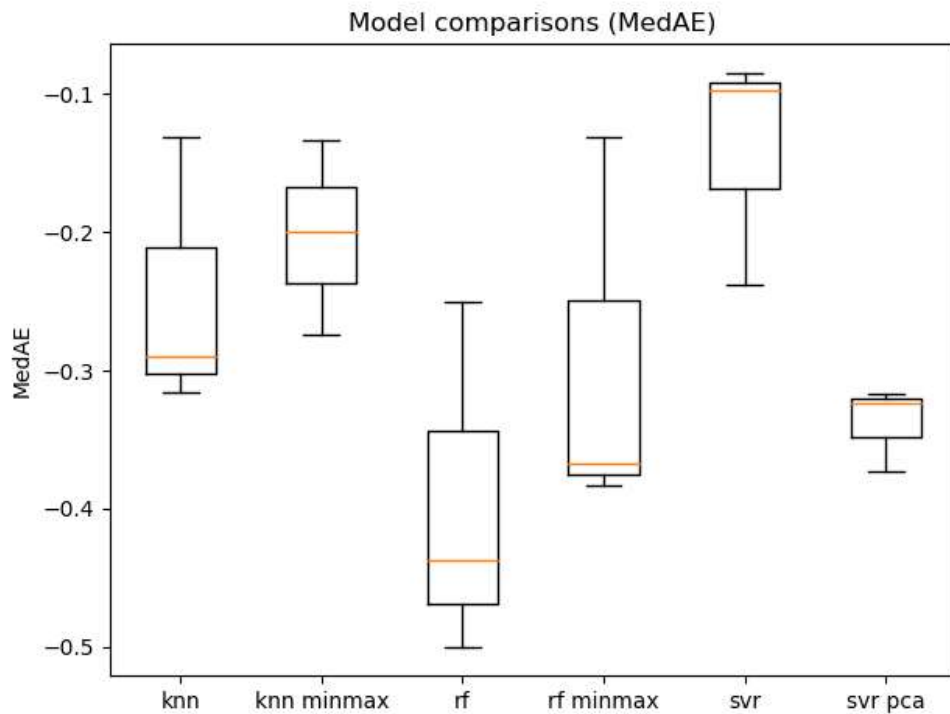


Figure 4.19 Median Absolute Error comparison predicting *E. coli*

The Median Absolute Error ranks the models differently. Here the SVM is clearly the best followed by KNN + Min-Max, KNN, RF + Min-Max, SVM + PCA and finally RF.

Figure 4.20 is a box plot of the Explained Variance for the models:

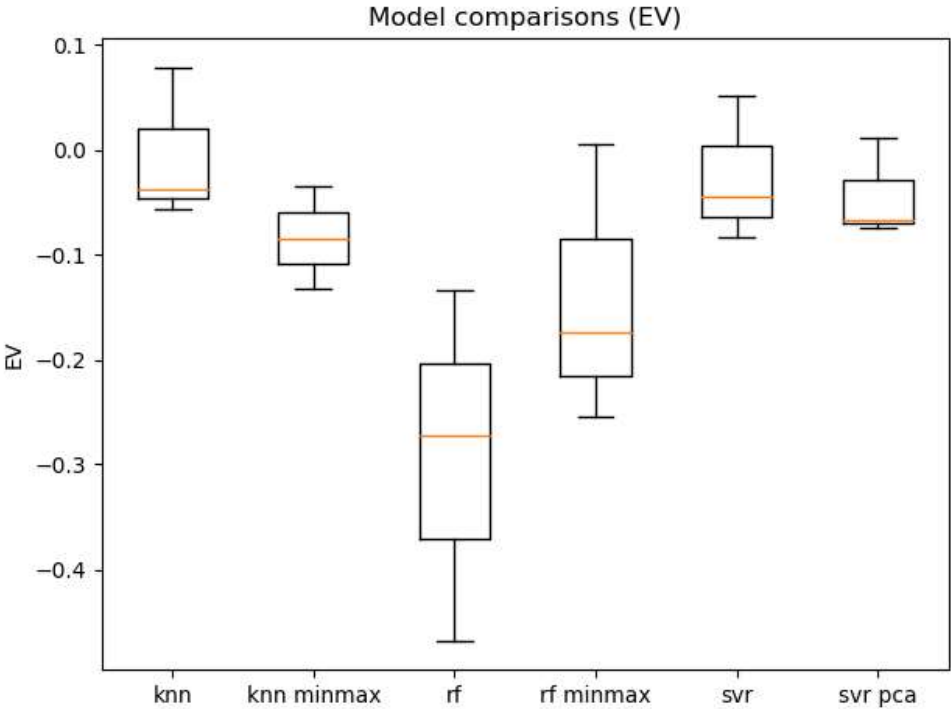


Figure 4.20 Explained Variance comparison predicting *E. coli*

According to this metric the models are ranked KNN, SVM, SVM + PCA, KNN + Min-Max, and the RFs last.

Figure 4.21 is a box plot of the R<sup>2</sup> scores for the models. The scores are similar relative to each other compared to that of the EV, but some minor differences changed the order of which the models are ranked. Here the KNN comes first followed by SVM + PCA, SVM, KNN + Min-Max, RF + Min-Max and RF.



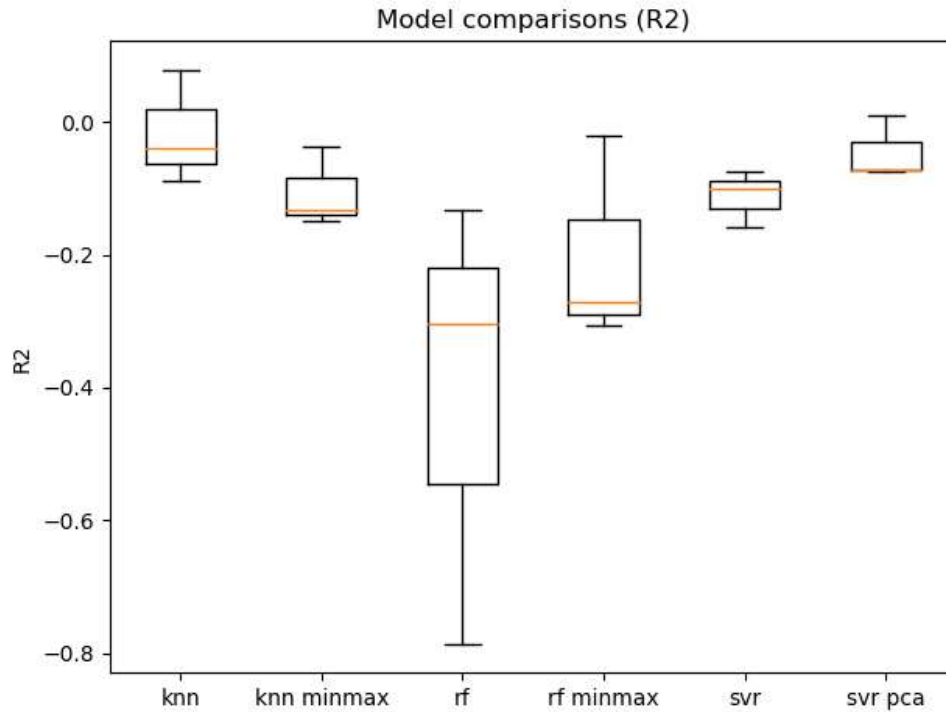


Figure 4.21  $R^2$  comparison predicting *E. Coli*

#### 4.3.6 E. coli: Visual Comparison

The performance of the models trained to predict *E. coli* is measured visually as well, as with the ones for Coliform Bacteria. Again, there are three plots per figure showing the observed and predicted values separately and then overlaid on each other.

The first figure, Figure 4.22, shows the result from the best Random Forest that was found:

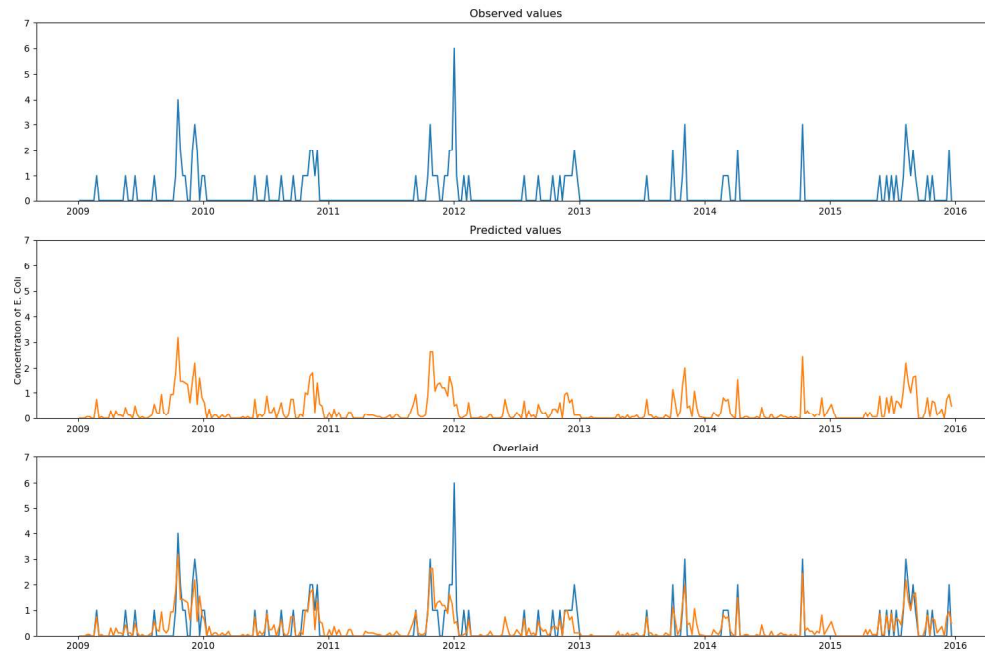


Figure 4.22 Optimized Random Forest with no pre-processing

The optimized Random Forest is mostly able to predict the E. coli data. It does not recognize the magnitude of the largest peak, and it has some false positives, particularly around late 2014, but in general it does a decent job.

Searching for the optimal Random Forest with pre-processing returned Min-Max Scaler as the best alternative. Figure 4.23 shows one such result. Pre-processing the data with Min-Max Scaler does not seem to improve the result, however, initially no good solutions were found using Random Forest with no pre-processing and it is therefore included in the results.

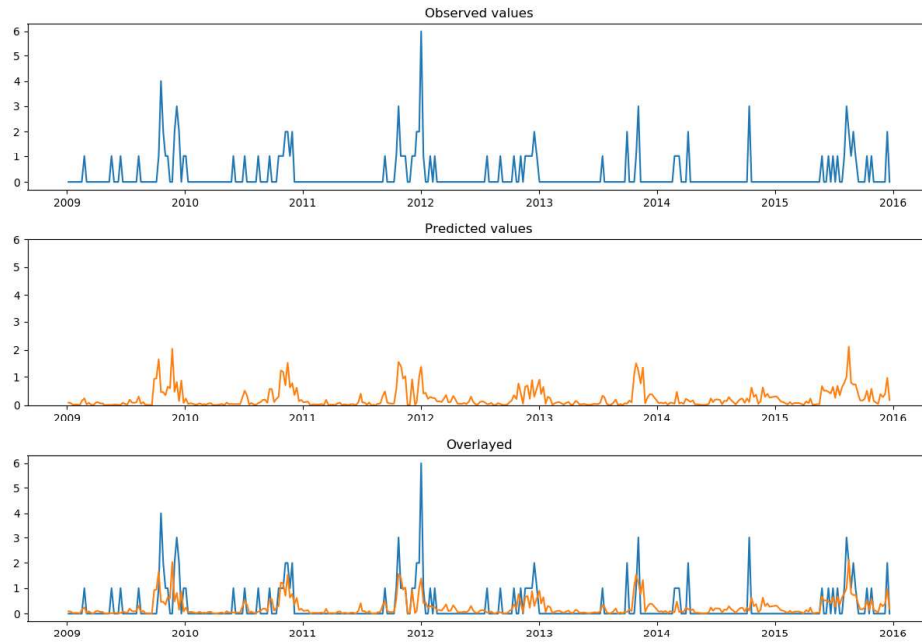


Figure 4.23 Optimized Random Forest pre-processed with Min-Max Scaler

Figure 4.24 is the result from an attempt at training an SVM on the E. coli data:

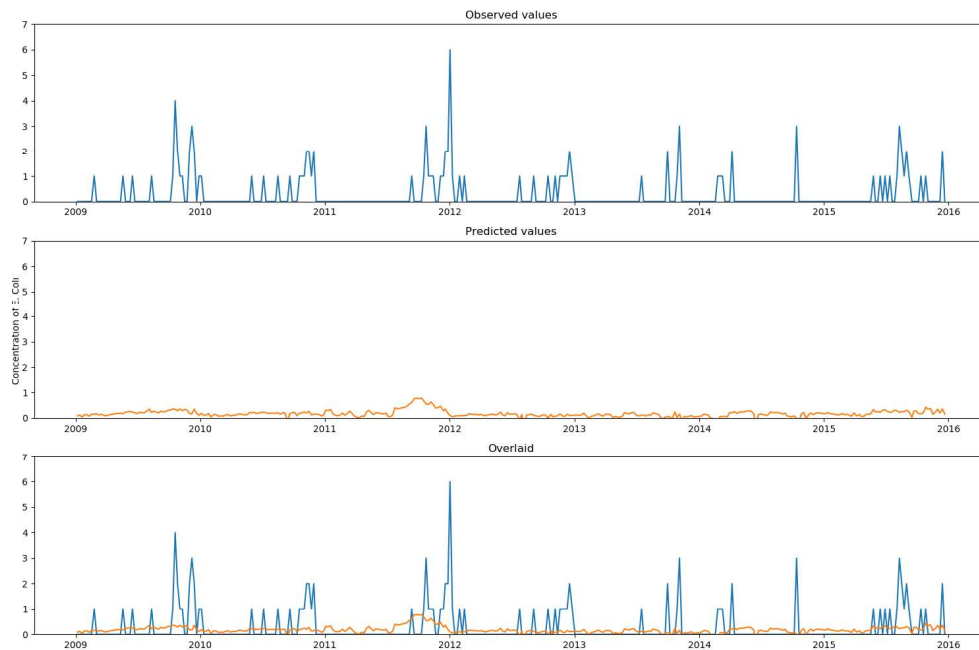


Figure 4.24 Optimized SVM with no pre-processing

An SVM with no pre-processing does not seem to be able to make much sense of the data. Several attempts to train SVMs without pre-processing were made, all yielding similar results. Adding all available pre-processors to the search space for the SVM yielded the result seen in Figure 4.25.

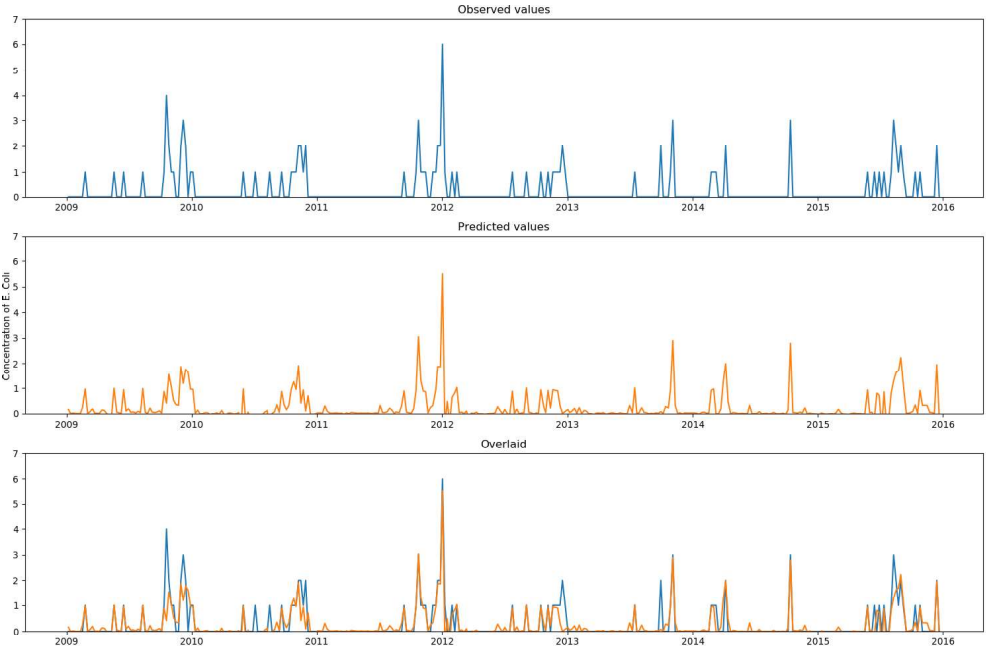


Figure 4.25 Optimized SVM pre-processed with PCA

An optimized SVM trained on data pre-processed with PCA appears to be far better suited for understanding the *E. coli* data. The model is able to correctly predict much of the data but fails in some cases where it's not able to predict the magnitude or predicts negative values.

Figure 4.26 is the result from a trained KNN with optimized parameters and no pre-processing. Like the SVM, the KNN appears to be unable to make sense of the data.

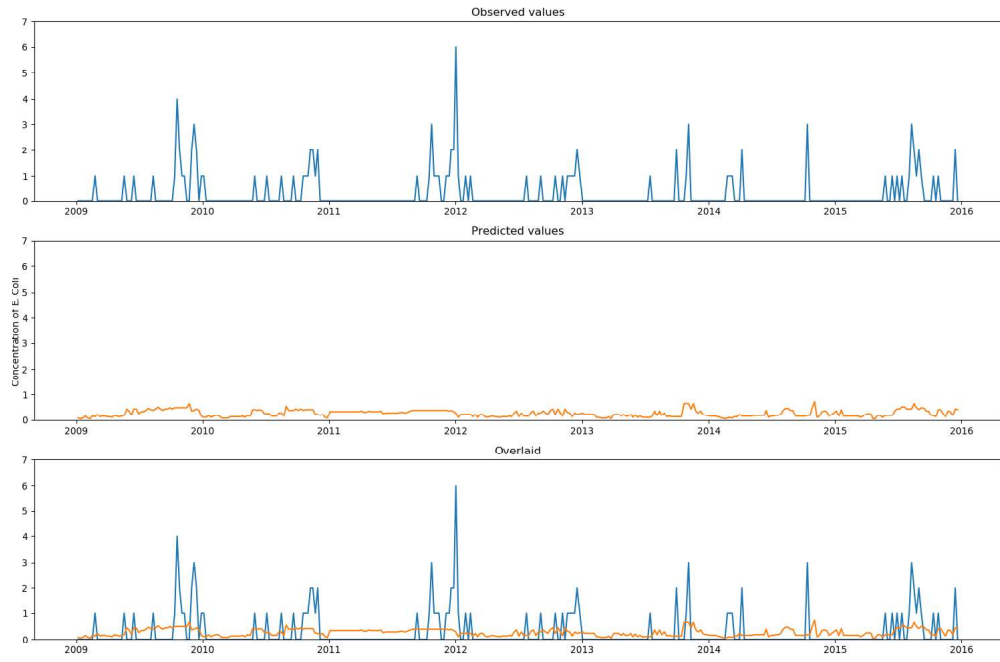


Figure 4.26 Optimized KNN with no pre-processing

Adding the available pre-processors to the search space for an optimal KNN yielded the result seen in Figure 4.27. When the data is pre-processed using Min-Max Scaler it is better able to predict the data. However, it is still not particularly good.

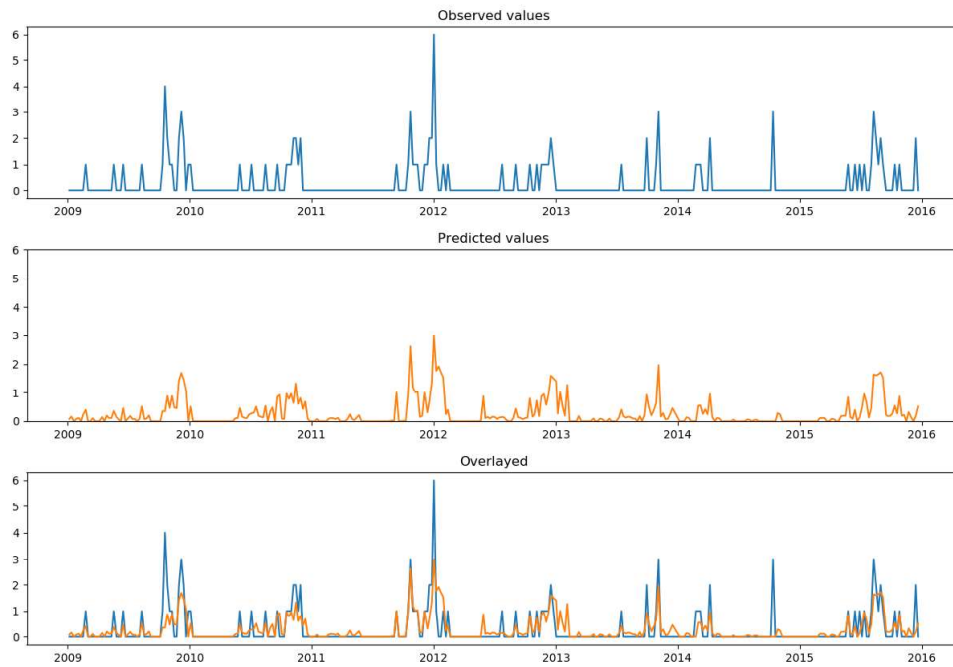


Figure 4.27 Optimized KNN pre-processed with Min-Max Scaler

#### 4.3.7 Training models on the data without temperature

All the results that have been presented so far are from the data for Maridalsvannet. As can be noted in the description of the data in section 4.1, the data from Brusdalsvatnet does not have measurements of temperature. For this reason, some attempts were made to train models on the data from Maridalsvannet, excluding the temperature measurements, for the sake of testing whether these models could accurately predict the levels from Brusdalsvatnet. Training models on the data without temperature proved to be a lot harder, which makes sense since one could expect that temperature has great impact on bacterial growth. This is reflected in the correlation analysis in **Error! Reference source not found.** where Coliform bacteria has a strong correlation to temperature compared to the other parameters, as mentioned in Section 4.1.

Figure 4.28 is the result from the best model that was achieved on the data from Maridalsvannet without temperature. Several other learners like SVM (Figure 4.30) and KNN (Figure 4.31) were tested, none of which produced any usable result.

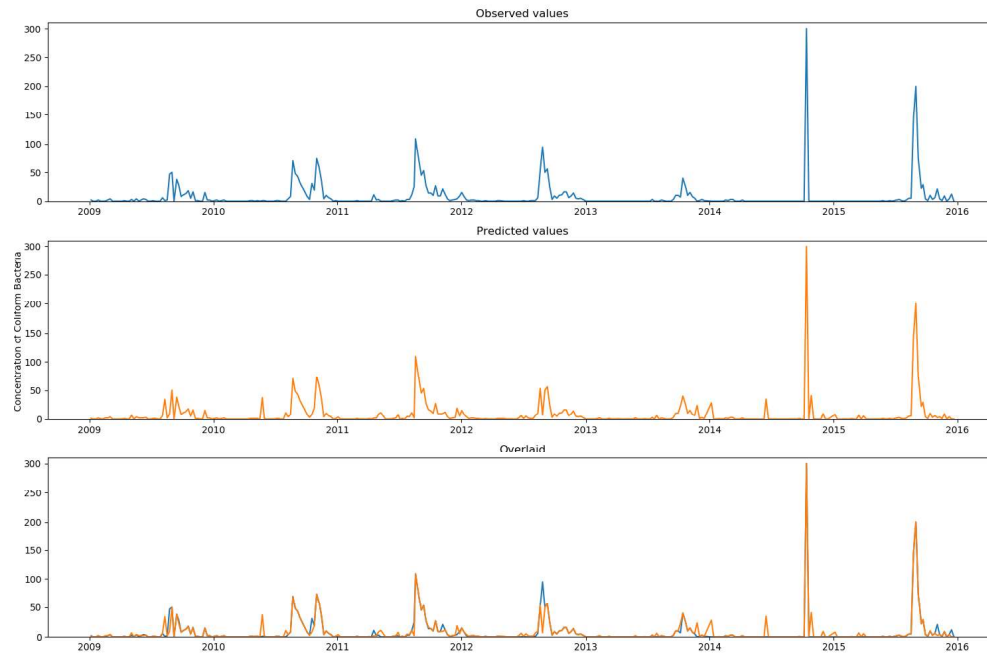


Figure 4.28 Optimized Random Forest on data without temperature

This optimized RF is able to predict the levels of Coliform Bacteria in the data from Maridalsvannet as well as any of the other models presented so far even without using temperature as input. Since this model does not require temperature as input it can be tested on the data from Brusdalsvatnet to see if it is able to generalise enough to work on data from other treatment plants.

Figure 4.29 shows the predictions the model made on the data from Brusdalsvatnet. The model does not seem to be able to make any sensible prediction on the data at all.

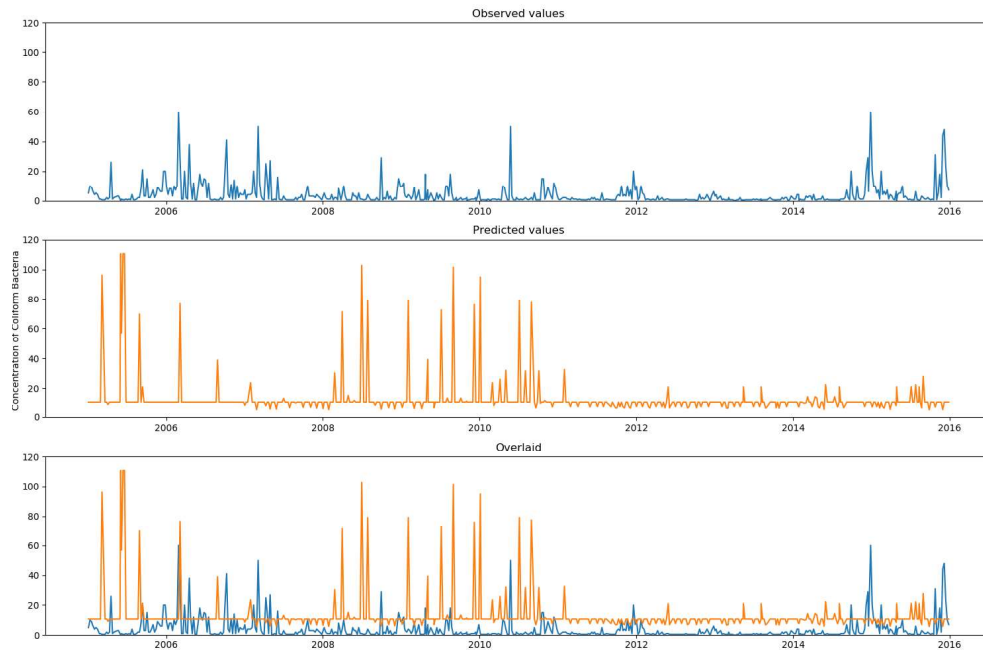


Figure 4.29 Predictions on the data from Brusdalsvatnet using model trained on the data from Maridalsvannet

Figure 4.30 shows the best SVM that was achieved on the data from Maridalsvannet without temperature.

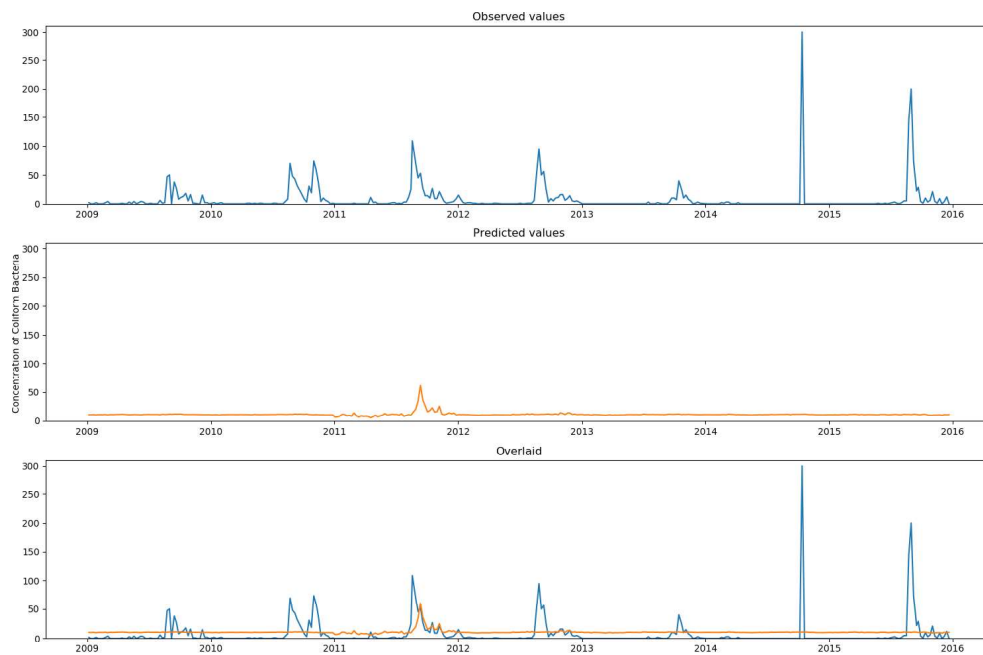


Figure 4.30 Optimized SVM on the data from Maridalsvannet without temperature



Similarly, Figure 4.31 shows the best result that was achieved on the data from Maridalsvannet without temperature using KNN as learner.

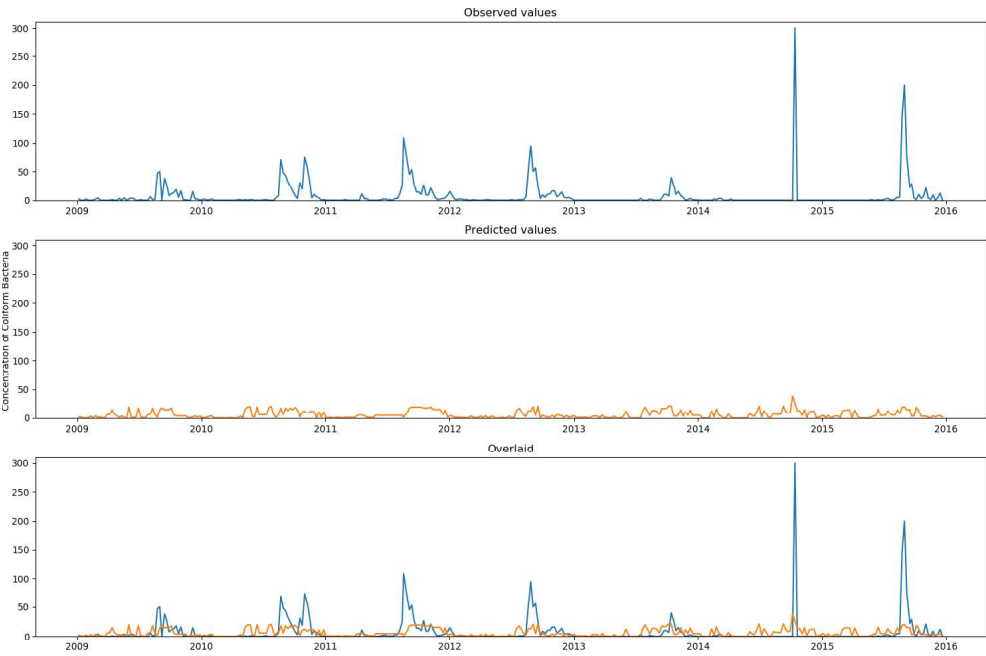


Figure 4.31 Optimized KNN on the data from Maridalsvannet without temperature

## **5 Discussion**

### ***5.1 The Data***

There are several issues with the data that limits the effectiveness of the models. Firstly, the treatment plants do not account for the same parameters in their raw water sources, as this generally depend on the size of the plant. For instance, the plant supplying the data from Brusdalsvatnet do have some records of the temperature. However, it is not kept together with the rest of the data and so the only temperature data that was acquired for this thesis was the temperature measured each day in 2017. Likewise, there was data from a water treatment plant in Bergen and elsewhere, but neither of those included measurements of temperature. Moreover, they didn't include any record of the water's alkalinity either, which is included for both Brusdalsvatnet and Maridalsvannet. Temperature is a very easy variable to measure, and it stands to reason that temperature would have an impact on bacterial growth.

Secondly, it would appear that the different treatment plants do not have the same standard procedure for accounting for different parameters in their raw water sources. According to a domain expert, the plants use different methods of measuring levels of bacteria, and so a number from one of them would not mean the same on the other. In addition, some treatment plants have larger budgets and may have employed water engineers or other people with special training while others are run by staff with limited training. Obviously, this greatly impedes a model's ability to generalise and understand the data in such a manner that it is able to make accurate predictions for any plant.

#### **5.1.1 Differences in raw-water catchments**

The way the data is measured might not be the only element making general predictions hard, but the source could be an issue as well. Although the two main water sources used in this are lakes, they have

different catchment characteristics that may contribute to different microbial contamination dynamics in the two lakes. Also, Brusdalsvatnet is in Ålesund Kommune on the north-western coast of Norway, while Maridalsvannet is in Oslo Kommune to the south-east of Norway. These locations have different climates, which to some degree may affect the dynamics of microbial pathogens in both lakes.

## ***5.2 Hyperparameter optimization results***

Hyperparameter-optimized models have shown promising results in improving the models' ability to learn from the data. In all cases presented in Chapter 4.3, the optimized models performed better than the default ones.

### **5.2.1 The scoring metrics**

Each section in the previous chapter starts by comparing the models using box plots of scores generated by various scoring algorithms. These algorithms score a model's performance by calculating "how wrong" the prediction was compared to the observation in some way or another. Usually this is a fair indicator of the model's performance since it means that the better score is, the more accurate the model is all over. However, for this particular purpose we are not interested in whether the model is very accurate in predicting the lower levels of the indicator organisms. The most important cases are the peaks, naturally, since the treatment plants would have to deploy countermeasures for them. One of the problems mentioned in the introduction for this thesis is that the data is often zero-inflated. While this is not a problem for the machine learning algorithms themselves, it remains a problem for the scoring metrics in this case. For simplicity, let's say that there are 200 points of measurements of which there are 10 peaks and the rest is zero or close to zero. A model that predicts only zeroes would be right in 95% of the cases, which in general is quite good. On the other hand, a model that tries to capture the peaks might be slightly off even around all the zeroes and thus receive a worse score overall than the model predicting only

zeroes even though we can clearly see it's better. Mean Squared Error is the only scoring algorithm that might be fairer for this purpose since larger errors are punished more than small errors. Still, looking at the scoring in Figure 4.7 and visual comparisons in Figure 4.11 through Figure 4.16 we can clearly see that the optimized models are better than the default ones as opposed to what the scores tell us.

By default, the hyperopt-sklearn library uses this loss function:

$$loss = 1.0 - r2\_score(y_{target}, y_{prediction})$$

Given that the scoring methods does not seem to favour models that properly predicts the peaks, it stands to reason that a tailored loss function specifically made for this purpose could further improve the optimized models.

### 5.2.2 Stochasticity in training the models

All the results presented in the previous chapters were, of course, the best result from each experiment. Each combination of learning algorithm, data pre-processor, and label was optimized at least 3 times. The hyperopt-sklearn optimizer takes a parameter “max\_evals” which defines how many different configurations the fit function will try. The results varied greatly from experiment to experiment. For instance, training models on the data from Maridalsvannet without temperature yielded no usable result until near the end of the experiments when it was decided to try once more. There are a couple possible reasons for the highly variable results. Firstly, the data was randomly split for each experiment where 20% was kept for testing and the rest used for training. Since there are very few peaks in the data it might be that too few was included in the training data for the model to be able to recognize them properly.

Secondly, during all the experiments the “max\_evals” parameter was set to 100. In other words, the optimizer evaluated a maximum of 100

configurations for each learner. While this have indeed lead to better configurations than the default ones in many cases, there might be better configurations to be found yet. Furthermore, depending on how the optimizer is progressing, limiting it to a relatively small number of configuration might be another source for the variable results.

### **5.2.3 Training models on the data from Brusdalsvatnet**

Several attempts have been made to make models using the data from Brusdalsvatnet, none of which have produced any usable results. As mentioned in the description of the data in section 4.1.2, the data from Brusdalsvatnet does not contain measurements of the water's temperature. Presumably the lack of temperature measurements has a great impact on the models' ability to predict the level of indicator organisms. Looking at the correlation analysis of the data from Maridalsvannet, the temperature does indeed have a strong correlation with the level of most indicator organisms. This is, however, likely a small part of the problem. Looking at the box plot of the values in the data from Brusdalsvatnet (Figure 4.5, p.27), most of them is mainly zeroes.

### **5.2.4 The choice of learners**

As mentioned in section 3.3, where the experiment methodology is explained, Random Forest and Support Vector Machine were specifically chosen as learners because they were used in earlier attempts. However, one might have noticed that existing models also includes other learners like Artificial Neural Networks and Adaptive Neuro-Fuzzy Inference System, but do not include K Nearest Neighbours. The main reason is simple, the experiments were limited to those learners offered by the hyperopt-sklearn library. Since the goal has been not only to improve the existing models but also to explore other models with potential, the optimizer was allowed to choose the best learner. For some cases it turned out to be K Nearest Neighbours.

## 6 Concluding Remarks

The study shows that optimized models instead of default models can significantly improve the prediction of microbial organisms' concentration in raw water sources. In some cases, pre-processing the data before training improves the results, like when training models to recognize *E. Coli* in the data from Maridalsvannet. In other words, whether pre-processing will improve the results depends on the choice of learner and the data. It is recommended that these procedures be used in further developing models for water treatment plants. Ideally, the water treatment plants should use the same standardized procedures for accounting for the parameters in the raw water sources as that would greatly improve the reusability of the models. Furthermore, developing a new scoring method tailored for this problem in particular might further improve optimization of these models.

## 7 References

- Abu-Mostafa, Y. S., Magdon-Ismail, M. & Lin, H.-T., 2012. *Learning From Data*. s.l.:AML.
- Andersen, E., 2016. *Vannrapport 127: Vannforsyning og helse. Veiledning i drikkevannshygiene.*, s.l.: Norwegian Institute of Public Health.
- Ashbolt, N. J., Grabow, W. O. K. & Snozzi, M., 2001. Indicators of microbial water quality. In: *Water Quality: Guidelines, Standards and Health*. London: IWA Publishing, pp. 289-316.
- Beaujean, A. A. & Morgan, G. B., 2016. Tutorial on Using Regression Models with Count Outcomes using R. *Practical Assessments, Research & Evaluation*, 21(2).
- Berg, G., 1978. The Indicator System. In: *Indicators of Viruses in Water and Food*. s.l.:Ann Arbor Science Publishers.
- Berg, G., Dahling, D. R., Brown, G. A. & Berman, D., 1978. Validity of Fecal Coliforms, Total Coliforms, and Fecal Streptococci as Indicators of Viruses in Chlorinated Primary Sewage Effluents. *Applied and Environmental Microbiology*, December, Issue 36:6, pp. 880-884.
- Bergstra, J., Bardenet, R., Bengio, Y. & Kégl, B., 2011. Algorithms for Hyper-Parameter Optimization. *Advances in Neural Information Processing*.
- Bergstra, J. & Bengio, Y., 2012. Random Search for Hyper-Parameter Optimization. *Journal of Machine Learning Research*, Issue 13, pp. 281-305.
- Bergstra, J., Yamins, D. & Cox, D. D., 2013. *Hyperopt: A Python Library for Optimizing the Hyperparameters of Machine Learning Algorithms*. Austin, Texas, s.n., pp. 13-20.

- Birattari, M., Yuan, Z., Balaprakash, P. & Stützle, T., 2010. F-race and Iterated F-race: An Overview. In: *Experimental Methods for the Analysis of Optimization Algorithms*. s.l.:Springer, pp. 311-336.
- Black, M., 1937. Vagueness. An Exercise in Logical Analysis. *Philosophy of Science*, 4(4), pp. 427-455.
- Bonde, G. J., 1962. Bacterial Indicators of Water Pollution. *International Review of Hydrobiology*, 49(1), p. 181.
- Breiman, L., 2001. Random Forests. *Machine Learning*, October, Issue 45:1, pp. 5-32.
- Breiman, L., Friedman, J., Stone, C. J. & Olshen, R. A., 1984. *Classification and Regression Trees*. s.l.:CRC Press.
- Buchanan, B. G. & Duda, R. O., 1982. *Principles of Rule-Based Expert Systems*, Stanford: Stanford University.
- Claesen, M. & De Moor, B., 2015. *Hyperparameter Search in Machine Learning*. Agadir, Morocco, s.n.
- Claesen, M., Simm, J., Popovic, D. & De Moor, B., 2014. Easy Hyperparameter Search Using Optunity. *Computer Research Repository*, Volume abs/1412.1114.
- Claesen, M., Smet, F. D. & Suykens, J. A. K., 2014. EnsembleSVM: A Library for Ensemble Learning Using Support Vector Machines. *Journal of Machine Learning Research*, Volume 15, pp. 141-145.
- Cox, E., 1998. *The Fuzzy Systems Handbook: A Practitioner's Guide to Building, Using, and Maintaining Fuzzy Systems*. 2 ed. Oxford: Elsevier Science & Technology.
- Coxe, S., West, S. G. & Aiken, L. S., 2009. The analysis of count data: A gentle introduction to Poisson regression and its alternatives. *Journal of personality assessment*, 91(2), pp. 121-136.



- Ding, S. et al., 2015. Extreme learning machine: algorithm, theory and applications. *Artificial Intelligence Review*, 44(1), pp. 103-115.
- Eggensperger, K. et al., 2013. *Towards an Empirical Foundation for Assessing Bayesian Optimization of Hyperparameters*. Long Beach, s.n.
- Eregno, F. E., Nielsen, V., Seidu, R. & Heistad, A., 2014. Evaluating the trend and extreme values of faecal indicator organisms in a raw water source: a potential approach for watershed management and optimizing water treatment practice. *Environmental Processes*, 1(3), pp. 287-309.
- Godfree, A. F., Kay, D. & Wyer, M. D., 1997. Faecal streptococci as indicators of faecal contamination in water. *Journal of Applied Microbiology*, October, 83(S1), pp. 110-119.
- Grabow, W. O. K., 1996. Waterborne diseases: Update on water quality assessment and control. *Water S. A.*, 22(2), pp. 193-202.
- Groch, A., Vidigal, L. M. & Director, S. W., 1981. *A Bayesian based method for global optimization*, s.l.: Carnegie Mellon University.
- Herrador, B. G. et al., 2016. *Utbrudd av smittsomme sykdommer i norge. Årsrapport. Delrapport 3 av smittsomme sykdommer i Norge 2016*, s.l.: Norwegian Institute of Public Health.
- Herring, I. M., Böer, S. I., Brennholt, N. & Manz, W., 2015. Development of multiple linear regression models as predictive tools for fecal indicator organisms in a stretch of the lower Lahn River, Germany. *Water Research*, Volume 85, pp. 148-157.
- Hinton, G. E., 2012. A Practical Guide to Training Restricted Boltzmann Machines. In: *Neural Networks: Tricks of the Trade*. Berlin: Springer, pp. 599-619.
- Horan, N. J., 2003. 7: Faecal indicator organisms. In: *Handbook of Water and Wastewater Microbiology*. s.l.:Academic Press, pp. 105-112.

- Hsu, C.-W., Chang, C.-C. & Lin, C.-J., 2003. *A Practical Guide to Support Vector Classification*, Taipei, Taiwan: National Taiwan University.
- Huang, G.-B., Zhu, Q.-Y. & Siew, C.-K., 2005. Extreme Learning Machine: A New Learning Scheme of Feedforward Neural Networks. *Neurocomputing*, 70(1-3), pp. 489-581.
- Jang, J.-S. R., 1993. ANFIS: Adaptive-Network-Based Fuzzy Inference System. *IEEE Transactions on Systems, Man, And Cybernetics*, 23(3), pp. 665-685.
- Kluyver, T. et al., 2016. Jupyter Notebooks - a publishing format for reproducible computational workflows. I: *Positioning and Power in Academic Publishing: Players, Agents and Agendas*. s.l.:s.n., pp. 87-90.
- Komer, B., Bergstra, J. & Eliasmith, C., 2014. Hyperopt-Sklearn: Automatic Hyperparameter Configuration for Scikit-Learn. *Proceedings of the 13th Python in Science Conference*, 13(1), pp. 34-40.
- Lin, S.-W., Ying, K.-C., Chen, S.-C. & Lee, Z.-J., 2008. Particle swarm optimization for parameter determination and feature selection of support vector machines. *Expert Systems with Applications*, 35(4), pp. 1817-1824.
- Loh, W.-Y., 2011. Classification and regression trees. *Data Mining and Knowledge Discovery*, 1(1), pp. 14-23.
- Lukasiewicz, J., 1930. Philosophical Remarks on many-valued systems of propositional logic. *Selected Works*.
- Mamdani, E. H. & Assilian, S., 1975. An experiment in linguistic synthesis with fuzzy logic controller. *International Journal of Man-Machine Studies*, 7(1), pp. 1-13.
- Marsland, S., 2015. *Machine Learning. An Algorithmic Perspective*. s.l.:CRC Press.

McCulloch, W. S. & Pitts, W. H., 1943. A Logical Calculus of the Ideas Immanent in Nervous Activity. *Bulletin of Mathematical Biophysics*, Issue 5, pp. 115-133.

McKinney, W., 2011. pandas: a Foundational Python Library for Data Analysis and Statistics. *Proceedings of the 13th Python in Science Conference*.

Mitchell, M., 1998. *An Introduction to Genetic Algorithms*. s.l.:The MIT Press.

Mohammed, H., Hameed, I. A. & Seidu, R., 2017. Comparison of Adaptive Neuro-Fuzzy Inference System (ANFIS) and Gaussian Process for Machine Learning (GPML) Algorithms for the Prediction of Norovirus Concentration in Drinking Water Supply. In: *Transactions on Large-Scale Data- and Knowledge-Centered Systems XXXV. Lecture Notes in Computer Science*. Berlin: Springer, pp. 74-95.

Mohammed, H., Hameed, I. A. & Seidu, R., 2017. Random Forest Tree for Predicting Fecal Indicator Organisms in Drinking Water Supply. *International Conference on Behavioral, Economic, Socio-cultural Computing (BESOC)*, pp. 1-6.

Mohammed, H., Hameed, I. A. & Seidu, R., 2018. Comparative predictive modelling of the occurrence of faecal indicator bacteria in a drinking water source in Norway. *Science of the Total Environment*, Volume 628-629, pp. 1178-1190.

Negnevitsky, M., 2011. *Artificial Intelligence. A Guide to Intelligent Systems*. s.l.:Pearson Education Limited.

Noble, W. S., 2006. What is a support vector machine?. *Nature Biotechnology*, Volume 24, pp. 1565-1567.

Pandey, P. K. et al., 2014. Contamination of water resources by pathogenic bacteria. *AMB Express*, June, Issue 4:51.

Pedregosa, F. et al., 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, Volume 12, pp. 2825-2830.

Russel, S. J. & Norvig, P., 2010. *Artificial Intelligence. A Modern Approach*. s.l.:Pearson Education Limited.

Schafer, J. L., 1999. Multiple Imputation: A Primer. *Statistical Methods in Medical Research*, February, 8(1), pp. 3-15.

Scikit-Learn, 2017. *Scikit-Learn Model evaluation*. [Online] Available at: [http://scikit-learn.org/stable/modules/model\\_evaluation.htm](http://scikit-learn.org/stable/modules/model_evaluation.htm) [Accessed 21 June 2018].

Scikit-Learn, 2017. *Scikit-Learn: PCA*. [Online] Available at: <http://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html> [Accessed 21 June 2018].

Smith, T., 1895. Notes on *Bacillus coli communæ* and related forms, together with some suggestions concerning the bacteriological examination of drinking water. *American Journal of Medical Science*, Issue 110, pp. 283-302.

Souza, S. X.-d., Suykens, J. A., Vandewalle, J. & Bollé, D., 2010. Coupled simulated annealing. *IEEE Transactions on Cybernetics*, 40(2), pp. 320-335.

Steinberg, D., 2009. 10 CART: Classification and Regression Trees. In: *The Top Ten Algorithms in Data Mining*. s.l.:CRC Press, pp. 179-202.

Sugeno, M., 1985. An introductory survey of fuzzy control. *Information Sciences*, 36(1-2), pp. 59-83.

Suparta, W. & Alhasa, K. M., 2016. Adaptive Neuro-Fuzzy Inference System. In: *Modeling of Tropospheric Delays Using ANFIS*. s.l.:Springer, pp. 5-18.

Tsai, J.-T., Chou, J.-H. & Liu, T.-K., 2006. Tuning the Structure and Parameters of a Neural Network by Using Hybrid Taguchi-Genetic Algorithm. *IEEE Transactions on Neural Network*, 17(1), pp. 69-80.

Vapnik, V. N., 1995. *Statistical Learning Theory*. Berlin: Springer.

World Health Organization, 1993. *Guidelines for Drinking-water Quality 2nd ed..* s.l.:s.n.

World Health Organization, 2011. *Guidelines for Drinking-water Quality 4th ed..* s.l.:s.n.

Wu, D., Mohammed, H., Wang, H. & Seidu, R., 2012. *Data-Driven Quality Control for a Water Supply System*, Ålesund: Norwegian University of Science and Technology.

Zadeh, L. A., 1965. Fuzzy Sets. *Information and Control*, Issue 8:3, pp. 358-353.

## Appendix A: Optimization tool

A web-based optimization tool was created as a utility for use during the thesis, and so that others might explore hyperparameter optimized models without the need for programming.

On the “Projects” home page a list of current projects is shown on the left.

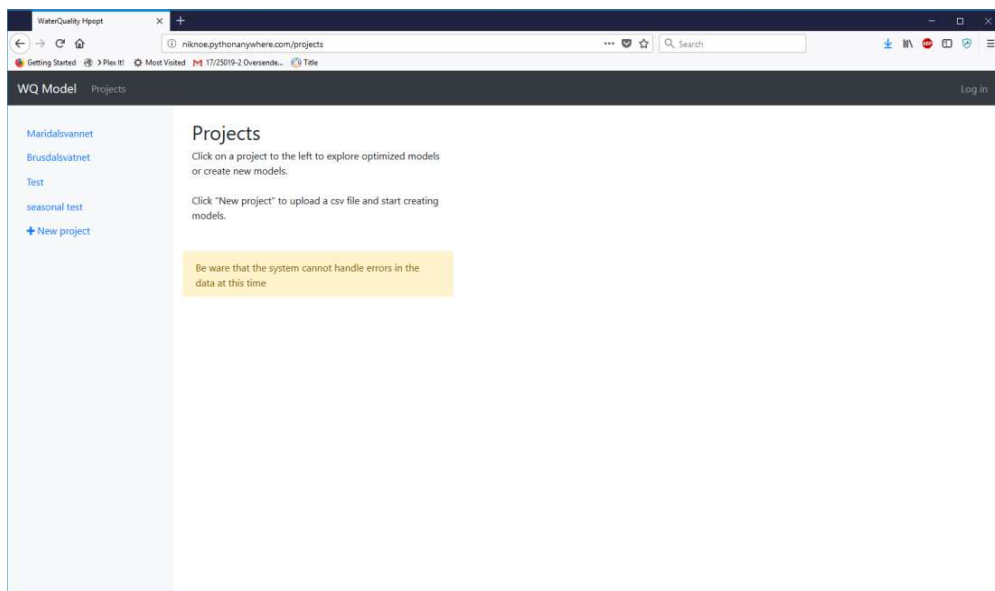


Figure A.1: Projects home page

Adding a new project is simple. Click “+ New Project”, fill in the form with a title and a CSV file containing the data for the project.

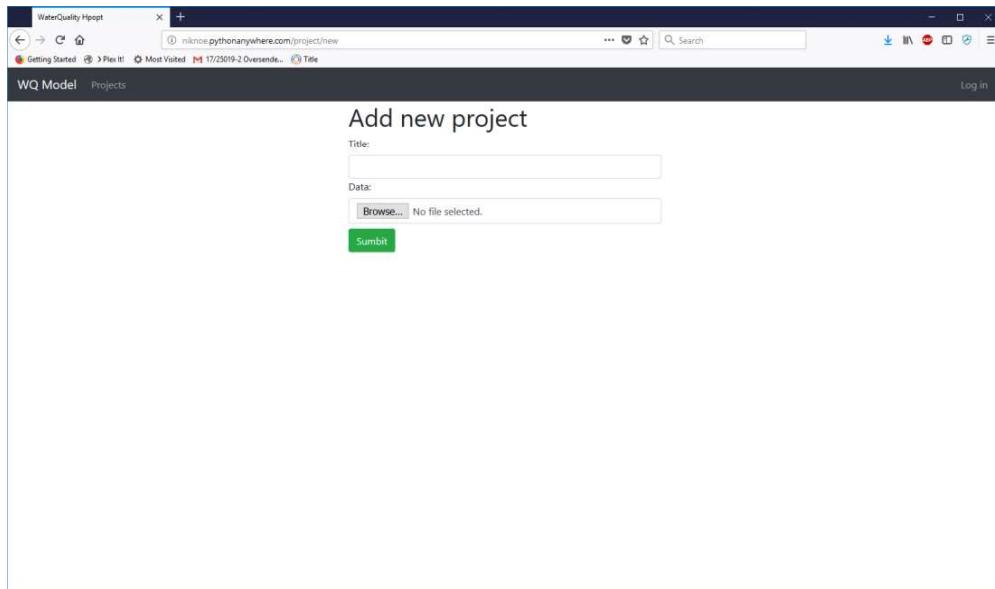


Figure A.2: Adding a new project

Clicking the name of a project on the left-hand side on the Projects page will show some recent models created for this project, the total model count and a link to the data used.

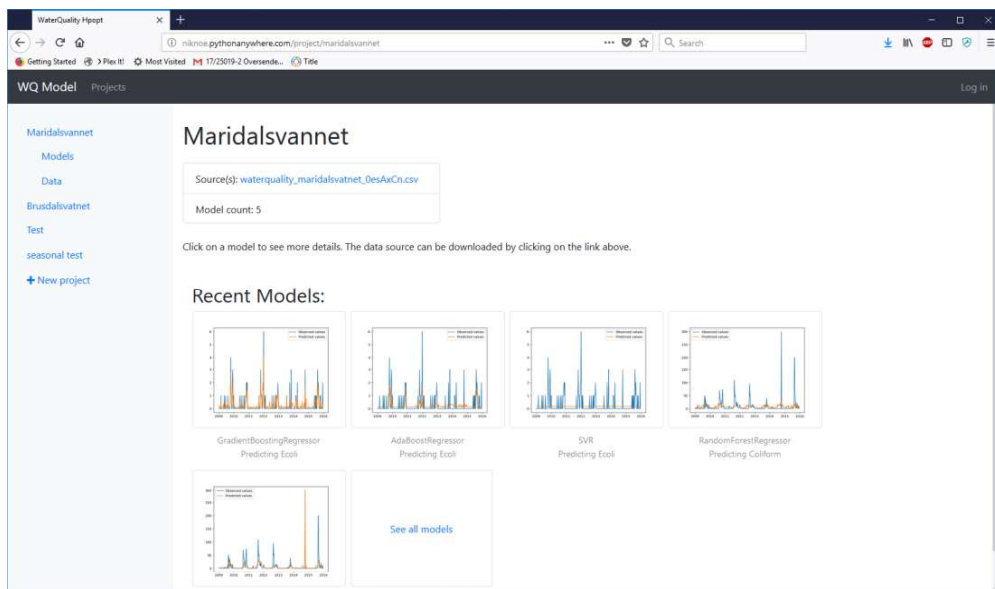


Figure A.3: Project start page

The options “Models” and “Data” have now appeared under the selected project. If this is a newly created project the data file might not be properly interpreted by the system. For instance, if the data is not separated by commas but by some other character, this would need to be specified on the “Data” page.

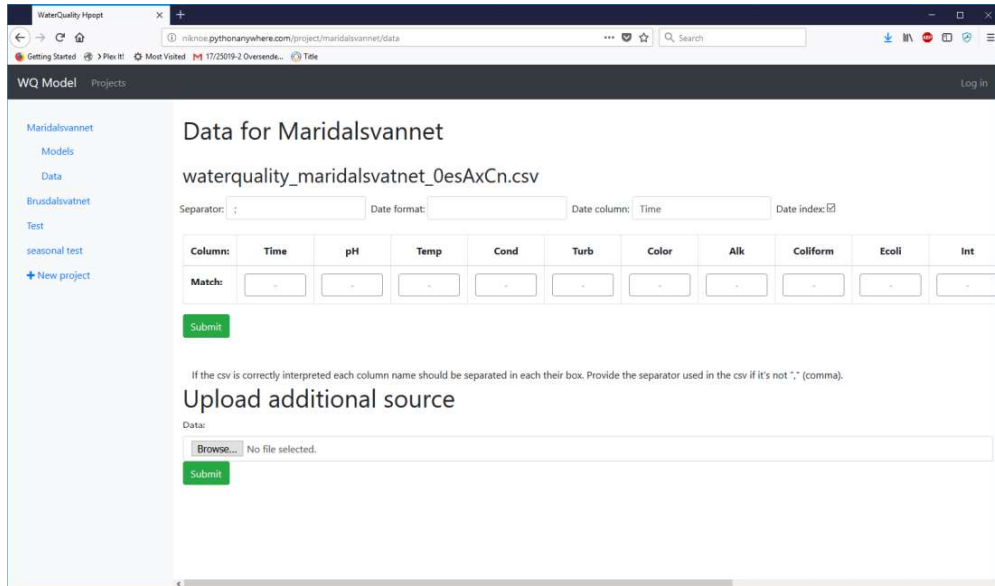


Figure A.4: Project data configuration

For each dataset that is uploaded to a project provide the separator if it is anything other than a comma, the date format if it looks like the date is parsed incorrectly, the title of the date column, and whether the data should be indexed by the date. The date format is given by the C standard format code. The tool will use the first line in the data as headers if all the columns are strings and automatically detect where the data start. Ideally, the data should be comma separated and the first line should be the headers for each column. The file must be a CSV file. No further preparation is needed.

If the tool has been able to correctly parse the data, then each column in the data should be separated in the table below the data set options. This



table has two rows: “Column” and “Match”. The second row is for matching the columns when there are multiple data sets available. Clicking on these boxes will give each column a number. The columns are matched by clicking the respective columns in a second data set in the order they appear in the first data set. Columns that the two data sets do not have in common will be left out. This option is used for combining the data from two files in order to train optimized models on the combined data.

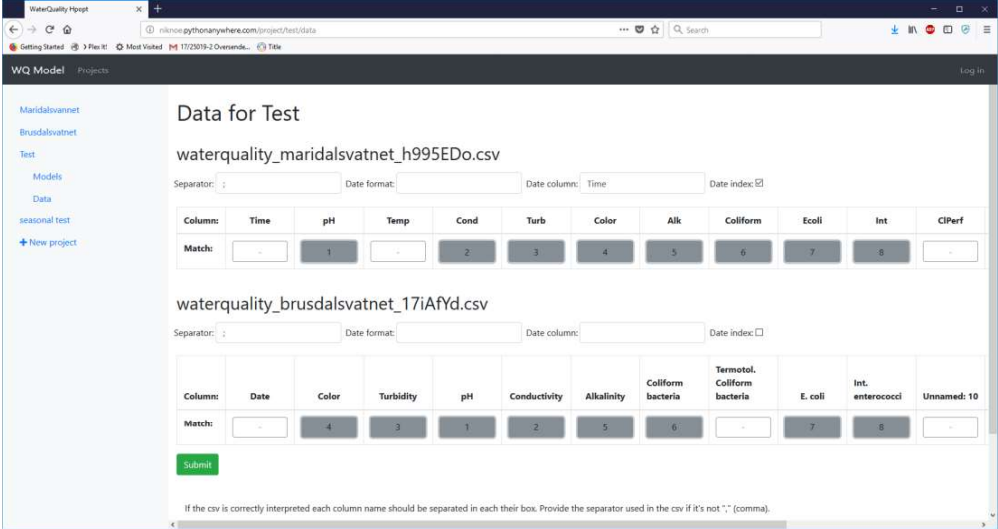


Figure A.5: Two data sets for one project

Clicking the other option below the selected project: “Models”, will allow the user to select inputs and output from the data and train optimized models. A specific learning algorithm can be selected, or “Any” to perform a general search over all the learning algorithms. The same goes for pre-processors with the addition of “None” where no pre-processor will be used.

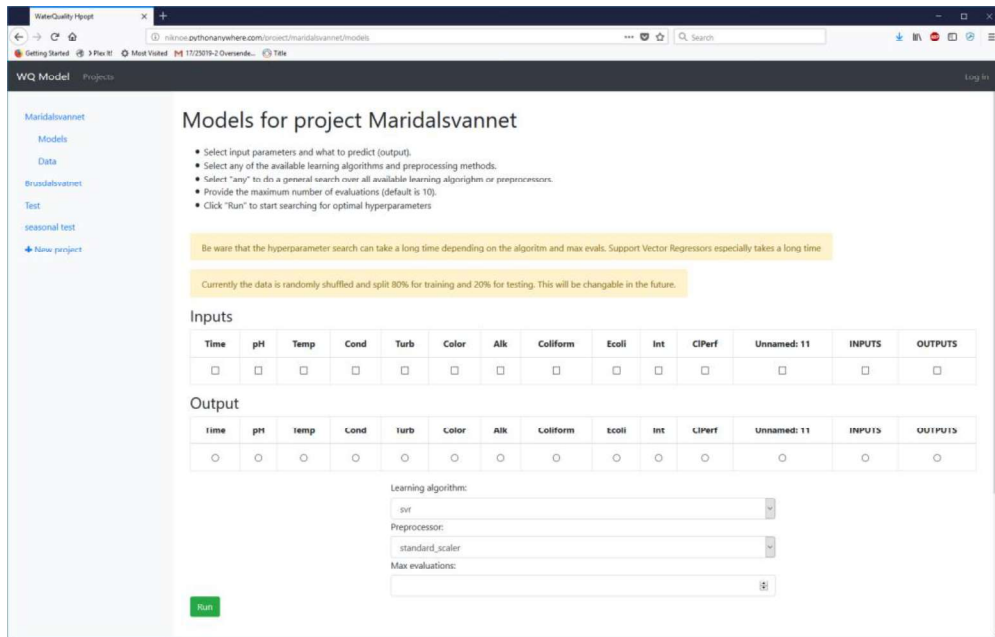


Figure A.6: Creating models

Further down on the models page a list of all models trained for this project is shown. Clicking any of these will show the details of that model including the type of learner, its parameters, pre-processing used, and some scoring metrics. A link to the model is also provided which can be used to download it and load the model into python for later use.

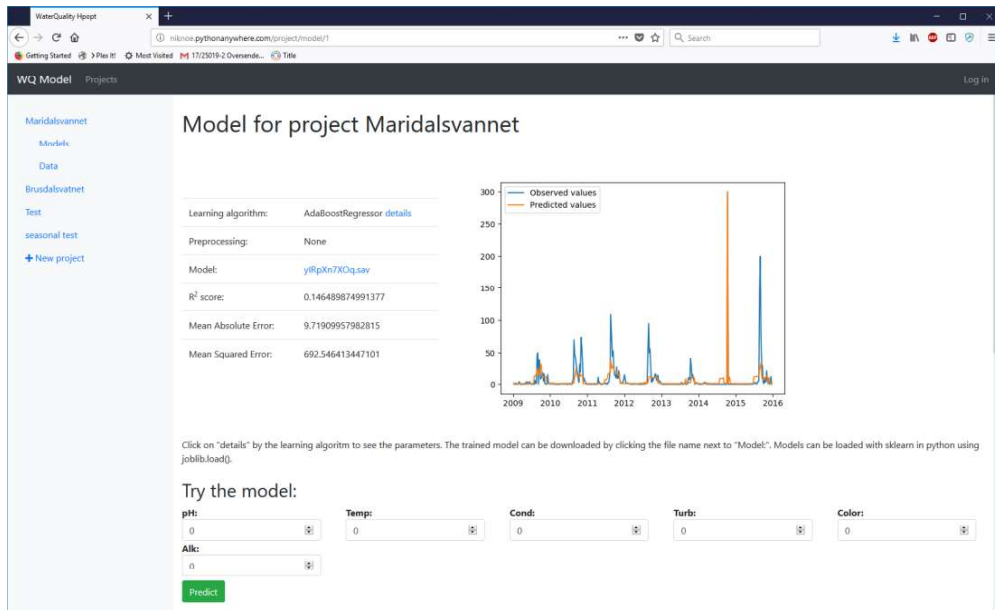


Figure A.7: Model details

Below the model details is a form that can be used to see what the model would predict given a specific set of measurements. Scrolling further down, an option to crosscheck the model with an entire dataset will appear. A dialog window appears with a dropdown for which to select the project to get the data to use on this model. Upon selecting a project several boxes will appear representing the columns in the data for the selected project next to the columns that the model requires. Clicking the boxes in the order presented to the left will allow the model to use the correct columns for its predictions. If there is a column required by the model that does not exist in the data from the other project, the crosscheck cannot be done.

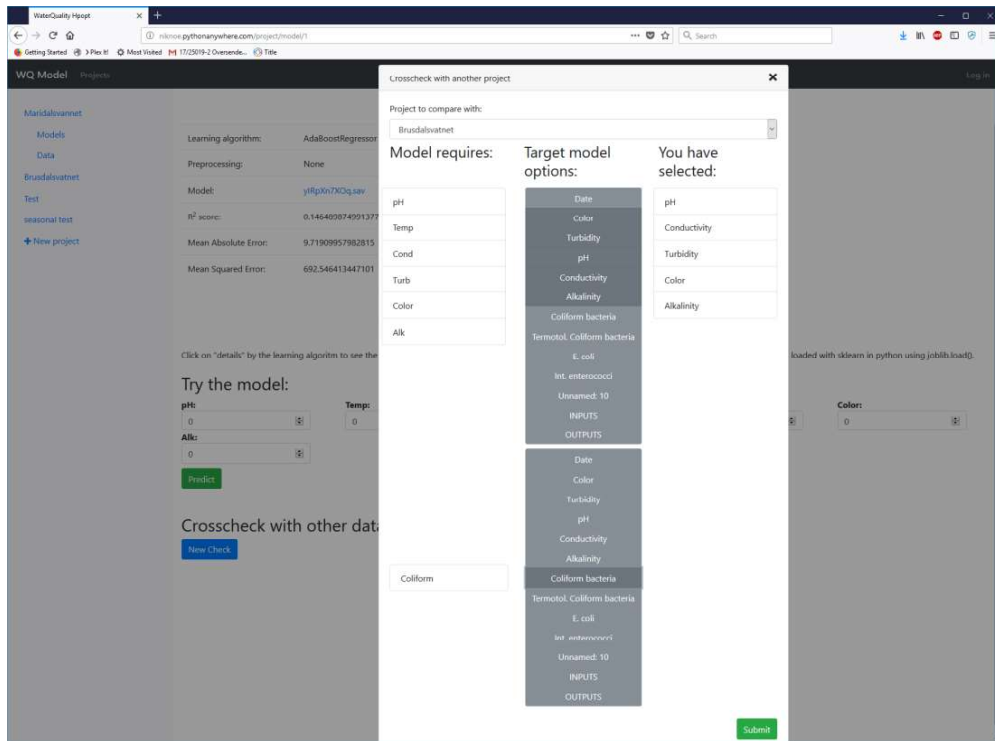


Figure A.8: Crosschecking model with other datasets

If all the columns are matched, then a crosscheck can be done. Upon completion a new page will appear showing a plot of the original “observed vs predicted” values and a plot showing the same type of plot only from the prediction made on the other data set.