APPLYING AI-BASED ANOMALY DETECTION TECHNIQUES TO IDENTIFY WASTE DISCHARGERS AT THE NORTH SEA

by

Shpat Cheliku (ICA-6549128)

M.S., Computing Science, Utrecht University, 2020

Submitted to Utrecht University's Graduate School of Natural Sciences
in partial fulfillment of
the requirements for the
Master's degree in Computing Science

Graduate Program in Computing Science
Utrecht University
2020

APPLYING AI-BASED ANOMALY DETECTION TECHNIQUES TO IDENTIFY WASTE DISCHARGERS AT THE NORTH SEA

APPROVED BY:

Dr. H. (Heysem) Kaya (Thesis Supervisor)	....
Dr. ir. R.W. (Ronald) Poppe (Thesis Co-supervisor)	....
(External supervisor) M. Sc. Paul Merkx	....

DATE OF APPROVAL: 12.09.2020

ABSTRACT

APPLYING AI-BASED ANOMALY DETECTION TECHNIQUES TO IDENTIFY WASTE DISCHARGERS AT THE NORTH SEA

While there is a considerable number of machine learning and anomaly detection studies as far as the maritime domain is concerned, the particular topic of ship waste and residue discharge remains an interesting but understudied problem. In this study, we explore the potential of machine learning, more specifically anomaly detection, in order to detect ship waste discharge at the North Sea, where several techniques ranging from supervised to unsupervised are investigated. One of the main problems we face in this study is the small amount of labeled data of the anomalous class in our disposition, leading to an extreme class imbalance in favor of the normal class. In order to tackle this problem, different class imbalance handling techniques are experimented with, one of them being the utilization of ensemble learning techniques. Our main contribution involves the development of a time series supervised learning pipeline that is used to discriminate between normal and anomalous behavior effectively (with performance significantly higher than chance level) and efficiently (with minimal cost and preferably real-time). The final performance results show that our model achieves a Macro Recall of 80%, and does so at $t = 18$ hours. Given that in perfect circumstances it would only be possible to detect our targeted behavior around or after 12 hours, $t = 18$ hours is considered relatively quick. Another, albeit smaller, aspect of our research is the development of an unsupervised learning model that aims to find previously undetected anomalous instances residing in our AIS dataset. The results of this approach give positive early indicators that the model is able to detect our targeted behavior, however, further refinements and a higher amount of anomalous data are necessary to make a case for the deployment of this model.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

a	Slope of the line (Equation 5.7)
c	Number of classes (Equations 5.14, 5.15 and 5.17)
d	Degree of curvature (Equation 5.8)
k	Number of normal trajectories used in the training set (Section 6.1)
N	Size of window taking into account previous predictions (Section 6.3)
o	Ordered list (Equation 5.4)
P_i	Precision of class i (Equation 5.14)
R_i	Recall of class i (Equation 5.15)
t	Time point (Section 6.1 and 6.3)
$\overline{x}$	Mean (Equation 5.1)
κ	Size of the sliding/expanding window (Subsection 5.2.1)
$\sum$	Summation

LIST OF ACRONYMS/ABBREVIATIONS

AI	Artificial Intelligence
AIS	Automatic Identification System
argmax	Argument of the maxima (Equation 5.9)
argmin	Argument of the minima (Equation 5.10)
CNN	Convolutional Neural Network
DBSCAN	Density-based spatial clusering of applications with noise
ETA	Estimated time of arrival
EM	Expectation Maximization
EU	European Union
GBM	Gradient Boosting Machines
GMM	Gaussian Mixture Model
KDE	Kernel Density Estimation
IDLab	Innovation and Data Lab
ILT	Inspectie Leefomgeving en Transport (Human Environment and Transport Inspectorate)
IMO	International Maritime Organization
MARPOL	The International Convention for the Prevention of Pollution from Ships
NLS	Noxious Liquid Substances
One-Class SVM	One-Class Support Vector Machines
PCA	Principal Component Analysis
RBF	Radial Basis Function
SVM	Support Vector Machine
VHF	Very high frequency
WEKA	Description Waikato Environment for Knowledge Analysis
XGBoost	Extreme Gradient Boosting

1. INTRODUCTION

1.1. The North Sea

The North Sea is a marginal sea of the Atlantic Ocean located between The Netherlands, Belgium, Great Britain, Denmark, Norway, Germany and France, and covers an area of 570.000 square kilometers [7]. The North Sea is known for its importance for marine transport where its shipping lanes are among the busiest in the world [8]. A number of major ports are located in the coasts of the North Sea, the largest in The Netherlands being: the port of Rotterdam, which is the busiest port in Europe and the fourth busiest port in the world by tonnage as of 2013, and the port of Amsterdam, which is the 4th busiest port in Europe by metric tonnes of cargo [9].

1.2. International Convention for the Prevention of Pollution from Ships (MARPOL)

The International Convention for the Prevention of Pollution from Ships (MARPOL), adopted at the International Maritime Organization (IMO), is the main international convention covering prevention of pollution of the marine environment by ships from operational or accidental causes [10]. This convention includes regulations aimed at preventing and minimizing pollution from ships, both accidental pollution and those from routine operations, and currently includes six technical Annexes, from which Annex II is the most important one as far as our research is concerned. Annex II contains regulations for the control of pollution by Noxious Liquid Substances (NLS) in bulk and sets out a pollution categorization system for the NLS substances that contains four categories [11]. The four categories are [11]:

(i) Category X: Noxious Liquid Substances which, if discharged into the sea from tank cleaning or deballasting operations, are deemed to present a major hazard to either marine resources or human health and, therefore, justify the prohibition of the discharge into the marine environment.

(ii) Category Y: Noxious Liquid Substances which, if discharged into the sea from tank cleaning or deballasting operations, are deemed to present a hazard to either marine resources or human health or cause harm to amenities or other legitimate uses of the sea and therefore justify a limitation on the quality and quantity of the discharge into the marine environment.

(iii) Category Z: Noxious Liquid Substances which, if discharged into the sea from tank cleaning or deballasting operations, are deemed to present a minor hazard to either marine resources or human health and therefore justify less stringent restrictions on the quality and quantity of the discharge into the marine environment.

(iv) Other substances: substances which have been evaluated and found to fall outside Category X, Y or Z because they are considered to present no harm to marine resources, human health, amenities or other legitimate uses of the sea when discharged into the sea from tank cleaning of deballasting operations. The discharge of bilge or ballast water or other residues or mixtures containing these substances are not subject to any requirements of MARPOL Annex II.

1.2.1. Zeezwaaien

Zeezwaaien is a Dutch word describing the discharge of washing water with wax, paraffin or other chemical cargo residues from seagoing ships [12]. It is mainly motivated by financial reasons. This act is permitted, however, it is only permitted when in compliance with the following requirements:

- The ship is proceeding en route at a speed of at least 7 knots in the case of self-propelled ships or at least 4 knots in the case of ships which are not self-propelled [13].
- The discharge is made below the waterline through the underwater discharge outlet not exceeding the maximum rate for which the underwater discharge outlet is designed [13].

- The discharge is made at a distance of no less than 12 nautical miles from the nearest land in a depth of water of no less than 25 meters [13].

In The Netherlands, the government body that ensures that the conditions for discharging chemicals outside the 12 nautical mile zone are met by ships calling at Dutch ports is the Port State Control of the Human Environment and Transport Inspectorate [12]. Despite the rules, regulations, and the governing body enforcing those rules, there are still acts of illegal waste discharge, where for example in 2015, 105 illegal discharges were identified by the Dutch Coast Guard [12].

1.3. Automatic Identification System (AIS)

The richest source of information and data on sea movement, making machine learning applications possible, comes from vessel AIS data. The Automatic Identification System (AIS) is an automated, autonomous tracking system which is extensively used in the maritime world for the exchange of navigational information between AIS-equipped terminals [14], and since December 2004, the International Maritime Organisation (IMO) has made it mandatory for all passengers' vessels, as well as commercial vessels over 299 gross tonnage that travel internationally, to carry AIS Transponders and be able to transmit and receive AIS messages [14].

Firstly, AIS was developed in the 90s for use as a short range identification and tracking system used on ships and other marine traffic, but in recent times its usage and benefits have been multi fold. It has been used as an instrument for collision avoidance, fishing fleet monitoring, coordination of search and rescue operations, but also to improve maritime security by identifying and monitoring suspicious activity patterns [15]. Each AIS system consists of one VHF transmitter, several VHF receivers, and standard marine electronic communication links to shipboard display and sensor systems [16]. In addition to that, AIS transponders can acquire position, timing, speed and course information using a built-in GPS receiver. The AIS systems are able to transmit three basic types of information: dynamic information, static information and voyage-specific information. Along with the dynamic information specified previously such as location, speed, course or rate of turn, AIS systems also transmit static information such as the ship's unique identity number (IMO number), the type of ship, the type of cargo that it carries, but also voyage-specific information such as its specified destination, the estimated time of arrival, the ship's draught, etc.

Some main factors that affect the frequency of the transmitted AIS messages are the type of the AIS transponder, which can be Class A or Class B, but also the moving status of the subject vessel. In terms of transponder classes, we are only interested in the Class A transponder type because they are designed to be installed in large ships and vessels with over 300 gross tonnage [16], while Class B transponders have been developed to provide navigation benefits to very small sea vessels. An AIS system is able to send updated dynamic messages every 2 to 10 seconds while the vessel is underway with a certain speed, and send updated messages every 3 minutes if a vessel is anchored or moored [17]. The remaining data, which are mostly static and voyage-specific information are sent every 6 minutes and this information is usually required to be updated manually by the vessel's operator. The range or coverage of the AIS systems is similar to the range of the VHF radios, which is between 15-20 nautical miles around the vessel, however, in specific cases with certain antenna elevation levels, perfect weather conditions and no other obstacles, the range could be extended up to 40-60 nautical miles [18].

As discussed in the previous paragraphs, AIS provides a lot of benefits in terms of traffic monitoring and vessel assistance and it has made vessel owners and maritime authorities reliant on it to ensure maritime traffic safety. However, the AIS system is not without its limitations. As mentioned previously, the AIS system coverage range is similar to other VHF applications which depends upon the height of the antenna and it has a range of 15-20 nautical miles. In some cases this range is enough, but in some cases it may be not, depending on the location of the vessel at a point in time. Another limitation is that the accuracy of the data received is only as accurate as what the owner of the vessel manually inserts into the static and voyage-specific messages of AIS system, and it is not always the case that these are correct. To make sure that correct AIS data is sent to other vessels and shore authorities, vessel operators are reminded to enter current voyage related data such as draught, type of hazardous cargo, destination and ETA properly at the beginning of each voyage and whenever changes occur [18]. Another thing that can occur is that the AIS systems can be faulty, may be repeatedly switched on and off (maybe even intentionally), leading to inaccurate location messages and resulting in a ship's passage that was captured neither entirely nor correctly. A very important limitation, or let's say misuse of the system, can be the so-called AIS spoofing, or the intentional reporting of incorrect information. Given that parts of the AIS message need to be manually entered by the operators of the vessel, the trustworthiness of the AIS reports depends on the willingness of the crew of the ship to report their true data, and this might not be always the case for different reasons [19].

A typical volume of worldwide maritime data represents an estimated 18 million AIS positions per day [20], coupled with the great amount of available historical AIS tracking data, up to billions of records, collected by a multitude of sensors and systems, make the effective detection of illegal activities in the maritime domain using Machine Learning applications feasible and promising, but very challenging nevertheless [21].

1.4. The Human Environment and Transport Inspectorate (ILT)

The Human Environment and Transport Inspectorate (ILT) is the supervising authority of the Ministry of Infrastructure and Water Management of The Netherlands. One of the matters the ILT is interested in is the so-called "zeezwaaien" behavior, discussed in Subsection 1.2.1, which is the act of ships discharging their tank washing waters and residues at sea. This kind of behavior is regulated by The International Convention for the Prevention of Pollution from Ships (MARPOL) to prevent pollution by ships discharging harmful liquids in the sea. The ILT is interested in analyzing this kind of behavior because it could also be performed illegally, while also in the cases where it is performed legally, it is still damaging to the environment. From a political or regulation point of view, it is very informative for the ILT to have quantitative and qualitative information about the frequency and characteristics of this behavior. The ILT has a considerable amount of moving vessels' data (AIS data) in their disposition gathered by The Netherlands Coastguard which makes a data-driven approach viable.

This leads to seeking answers to questions such as: Can detecting and predicting waste discharge behavior be automated using AI techniques given the available data?

1.5. Research problem

1.5.1. Scope of the research

The scope of the research is limited to the North Sea, more specifically the ports of Rotterdam and Amsterdam. The main reason for narrowing down the scope of the research is because the jurisdiction of the governing body, The Human Environment and Transport Inspectorate (ILT), is restricted to The Netherlands and the maritime boundaries of the North Sea governed by The Netherlands.

Different types of vessels are characterized by different types of behavior. After discussions with domain inspectors we came to the conclusion that we are not interested in all vessel types because most of them are not associated with the specified waste discharge behavior. The small number of labeled cases that we have, and discussions with domain inspectors as stated before, instructed us to focus on tankers, more specifically chemical tankers and NLS (Noxious Liquid Substances) tankers, the vessel types more likely to be involved in waste discharge behavior.

The Human Environment and Transport Inspectorate (ILT), based on previous confirmed instances of ship waste and residue discharge at the North Sea, is interested in performing a data-driven research in order to detect and predict the previously discussed waste discharge behavior. The data-driven approach is made possible given the availability of the AIS dataset, collected by The Netherlands Coastguard, consisting per-vessel information such as the position of the ship, the actual speed at particular positions and the orientation it is facing, among other features. This dataset and the dynamic and static features it contains make it potentially possible for machine learning techniques to learn vessel behavior, and potentially be able to differentiate between normal vessel behavior and vessels undertaking waste discharge behavior. Another useful dataset available for our research is the Portcall dataset, collected and provided by the European Maritime Safety Agency. The Portcall dataset provides information that can be used to find the ports visited by specific vessels, or to find all vessels that have visited specific ports. The availability of the AIS and Portcall datasets gives us the necessary information to extract all arriving and departing trips involving our ports of interest.

The IDLab data scientists consulting with domain inspectors have already identified 20 cases from the AIS dataset where potential waste discharge behavior might have been undertaken, however, it is important to note that none of the cases were inspected and confirmed. An indicator of waste discharge behavior is the so-called looping behavior where vessels depart a port, in our case the ports of Rotterdam or Amsterdam, and return to the same port within 48 hours without visiting another port in the meantime. These potential waste discharge cases are not the norm. It is usually the case that a ship simply exhibits normal behavior, and this can be seen from the fact that at the moment there are only 20 potential cases of waste discharge out of thousands of vessel trips. This means that this problem can be defined as an anomaly detection problem where the anomalous instances are the trips exhibiting waste discharge behavior.

1.5.2. Problem description and research questions

There are two important aspects that define our research direction. First, the AIS data in disposition for our research portray the ship trajectories in a time series representation. Second, we only have a very small number of labeled anomalous cases and, in comparison, a large number of normal cases. This means that we are dealing with an extreme class imbalance problem.

Taking into account the general description of our problem in the previous paragraph, it was decided to tackle the problem of identifying ship waste and residue discharge at the North Sea using two different techniques: supervised and unsupervised. Initially, the plan is to experiment with time series supervised learning techniques, where the goal is to develop a model that achieves acceptable performance as far as discriminating between normal and anomalous cases is concerned.

The second phase will consist of experimenting with unsupervised learning techniques. The goal is to find out if the machine learning techniques of our choice are able to discover new potentially anomalous instances on top of the ones we already have.

This leads us to the main research question:

Given the availability of the AIS dataset and the potential of supervised and unsupervised techniques to solve complex problems: Is it possible to develop machine learning models that detect ship waste and residue discharge at the North Sea effectively (with performance significantly higher than chance level) and efficiently (with minimal cost and preferably real-time)?

After defining the main research question we can get a bit more into specifics and define some sub-questions that can potentially help us answer the main research question. The trajectories are made up of individual AIS data points that collectively represent each individual trip in a time series data representation. This means that in order to represent each trip with as much information as possible, and not by a single feature vector, we need to perform time series feature engineering in order to summarize the trajectory up to a specific point in time. Time series feature engineering could be performed by considering different time resolutions, such as: using the records of the last 10 minutes of the trajectory, using the records of the last 60 minutes of the trajectory, or by implementing a full trajectory time resolution by expanding the number of records taken into consideration until the very end of the trajectory. This leads to the following sub-research questions:

- What is the effect of summarizing the low-level descriptors that come with the AIS data (e.g. latitude, longitude, speed, orientation), over past records?
  - Based on the performance scores of the individual time resolutions (Macro Recall and Macro F1), which time resolutions are the most useful for building accurate models?
  - Do the performance scores indicate a marked improvement when using all the engineered features containing historical trajectory information compared to only using the already existing low-level descriptors?

As mentioned previously, looping behavior seems to be a good indicator of our targeted behavior. However, at the point when this knowledge is available, so having access to the Portcall arrival and departure events, the potential illegal action (waste discharge) would have already been committed. This leads to the following research sub-question:

- Utilizing the AIS trajectory data, can we train models using previous sequential data points up to a specific point to detect our targeted behavior before occurring, or as early as possible after occurring?
  - Looking at the Macro Recall and Macro F1 performance scores of specific time points, how quickly is our model able to effectively discriminate between normal and anomalous trajectories?
  - Looking at the Macro Recall and Macro F1 performance scores of specific time points, which time interval of the trajectories is the most informative in order to discriminate between normal and anomalous trajectories?

We aim to fit tanker-specific normality and abnormality Gaussian Mixture Models. The purpose of this unsupervised approach is to better capture the nuances of the normal and anomalous trips instances, each on their own model. Then, using anomaly detection scores such as the log-likelihood in this case, we aim to find new cases that deviate from the normal behavior, leading us to potential new waste discharge cases undetected until now. This leads us to the following research sub-questions:

- Based on the log-likelihood scores of the individual trips, are the tanker-specific fitted models able to detect new anomalous cases residing in the dataset?
  - Are we able to find credible indicators that the anomalous behavior, was in fact, waste and residue discharge?

Eventually, as a part of this research or as an extension of it, after visualizing the Top N ranked anomalous cases according to our models, the domain inspectors will give their opinion whether the trajectories are indicative of waste discharge behavior.

And finally, a general research question involving the amount and the quality of the data that we have. The AIS data that was used in this research was gathered by The Netherlands Coast Guard and consists of AIS messages covering only a two-month period between April 2017 and May 2017. Along with that, as mentioned previously, at the beginning of this research project, we only have 20 labeled cases, which may not be enough for supervised learning. This leads to the following research sub-question:

- Is the amount and the quality of the data available for this study sufficient for waste discharge anomaly detection?

1.6. Outline of the document

The rest of the document will be organized as follows: In Chapter 2, we perform an extensive literature review mostly focused on anomaly detection in the maritime domain. In Chapter 3, we give background info on the algorithms and techniques that we are going to use for the most technical parts of our research. In Chapter 4, we depict the initial data exploratory phase, the data preprocessing part, and the maritime traffic analysis. In Chapter 5, we discuss and explain our most important methodological choices concerning both the supervised and unsupervised learning approaches under consideration for our research. In Chapter 6, we describe the experimental setup of both the supervised and unsupervised approaches, and also present and discuss the final results of our research. And finally, in Chapter 7, we summarize the conclusions of this research, explain what our new findings mean for the ILT and discuss the answers that we obtained as far as our research questions are concerned.

2. LITERATURE REVIEW

2.1. Anomaly detection

This section briefly discusses what anomaly detection is in general, and what kind of anomaly detection approaches are there. The first definition on what anomaly detection signifies was probably given by Grubbs in 1969 [22] where he said that an outlying observation, or an outlier, is one that appears to deviate markedly from other members of the sample in which it occurs. At the onset of development of outlier detection techniques, pattern recognition algorithms were not as developed as in recent times, meaning that they were quite sensitive to outliers in the data. This lead to using anomaly detection techniques to actually clean the data instead of focusing on the anomalous data points itself [23]. A turning point can be identified at the beginning of this century when researchers started to get interested in analyzing the anomalies itself instead of simply categorizing them as noise and removing them altogether. The main reason for this shift was that it was becoming very noticeable that these anomalous cases are often associated with particularly interesting events or suspicious activities, and it was worth investigating further instead of performing the so-called data cleaning procedure on them.

In contrast to the well-known classification setup, where training data is used to train a classifier and test data is used to measure performance afterwards, there are multiple setups possible when talking about anomaly detection. A specific formulation of the anomaly detection problem is determined by several factors such as: the nature of the input data, the availability (or unavailability) of labels, as well as the constraints and requirements induced by the application [24]. This leads to the three following categorizations of anomaly detection techniques: supervised anomaly detection, semi-supervised anomaly detection and unsupervised anomaly detection. The properties and differences between each of these techniques will be discussed in the following paragraphs.

(i) Supervised anomaly detection is characterized by the scenario where the training and testing datasets contain fully labeled data. This also means that the available data also contain labeled instances for normal as well as anomaly classes. Looking at the explanation given in the previous sentences, the supervised anomaly detection problem may simply be considered as a more difficult variation of the usual classification problem. However, the big difference is that in most cases the classes are extremely imbalanced, which makes sense because outliers (anomaly class) are defined as rare instances and it is natural that the distribution between the normal and anomaly class will be very skewed [25]. What we can deduce from this is that not all classification algorithms are suited for the supervised anomaly detection task, however, algorithms such as Support Vector Machines or Artificial Neural Networks are known to achieve good results.

(ii) Semi-supervised anomaly detection is a technique that operates in a semi-supervised mode, where one may also have access to some labeled normal or anomalous samples in addition to the unlabeled data [26]. Such samples could be hand labeled by a domain expert for example. The main idea of semi-supervised anomaly detection is that a model of the normal class is learned first, and afterwards the detected cases deviating from the model can be labeled as anomalous [23]. Well known semi-supervised anomaly detection algorithms that are known to achieve good performance are One-Class SVMs and Autoencoders.

(iii) Unsupervised anomaly detection is the most flexible and widely used anomaly detection technique because initially it does not require any labels. These techniques detect outliers solely based on intrinsic properties of the data instances [27]. First, the unsupervised anomaly detection algorithm learn what normal is, and then applies a statistical test to determine if a specific data point is an anomaly. A system based on this kind of anomaly detection technique is able to detect any type of anomaly, including ones which have never been seen before. The main challenge in using unsupervised machine learning methods for detecting anomalies is deciding what is normal for the application being monitored, a challenge that is indeed not straightforward.

2.2. Machine Learning and anomaly detection in the maritime domain

Lately, just like in almost every other domain where Computing Science is applicable, with the emergence of Artificial Intelligence and Machine Learning there also has been an increased interest in data-driven approaches for anomaly detection in the maritime domain. There has been research carried out in areas such as knowledge discovery, more specifically mapping maritime routes, mapping fishing activities, knowledge based trajectory prediction or event based knowledge discovery. Furthermore, research also has been carried out in the anomaly detection area, more specifically model-based analysis has been performed to detect low-likelihood behavior, detect spoofing, detect AIS off switching, among other implementations [21]. This section will discuss some of the most relevant topics and approaches considered in this particular domain, going from general to more specific, as far as our research thesis is concerned.

2.2.1. General ML implementations in the maritime domain

One of the most successful machine learning implementations in the maritime domain has been the research performed by the European Union Science Hub, where they managed to introduce a first map at European scale of EU fishing activities over one year [28]. AIS offers the ability to track ships with better precision at a much wider scale, and in Europe, starting from May 2004, all EU fishing vessels the length of which exceeds 15 meters are required to be fitted with AIS systems [29] leading to a significant impact in understanding the spatial distribution of fishing activities. In this particular study only the fishing activities of trawlers are analyzed. Trawlers represent the majority of the fishing vessels above 15 meters length in Europe. First, position and speed data cleaning was performed and the data were downsampled to an interval between 5 minutes between consecutive observations. The downsampling reduced the number of AIS messages from 150 million to 60 million, a great reduction if it doesn't lead to too much loss of information. After the data cleaning, to identify fishing behavior the authors made an assumption that the speed of vessels when performing fishing activities is relatively low compared to the steaming speed of vessels, so they created speed profiles, excluding zero-speed points indicating that the ship is at a port. Other features such as vessel size, fishing gear, etc, were also used to classify individual points as “fishing” or “not fishing”, meaning that the fishing behavior identification had to be performed for each individual vessel. This was done by using a Gaussian Mixture Model that isolated the two distributions of the two main activities and obtained the necessary parameters that were used to built fishing speed confidence intervals for each individual vessel. The points that were classified as “fishing” then were aggregated into small spatial cells and density maps were created from them, in which high intensity fishing areas are clearly distinguishable.

Another exploration of the machine learning potential in the maritime domain was performed in [30] where a framework was developed to perform automated rescue detection based on vessel trajectory information in the Mediterranean Sea. Initially the collection and the combination of data on rescues in the Central Mediterranean was performed, where sources were categorized as “ready-made” and “custom-made”. By “ready-made” sources the authors of course mean AIS data, which made it possible to monitor the behavior of search and rescue vessels at the Mediterranean Sea, but also identify the involvement of merchant ships in these search and rescue operations. Other sources of “ready-made” data were broadcast warnings, Twitter, or photos and videos when available. All these multiple data sources were fused together to define the so-called quantified rescue framework, and this allowed to build coherent narratives of specific situations seen from different points of view. One of the difficulties they encountered was fusing the different sources of data for a specific situation with the correct AIS trajectory data. They found out that timestamps were the most helpful feature that would help fuse the data by associating the descriptive sources with the AIS coordinates that are the closest in time. This is important to mention because we encountered a similar problem when trying to merge the datasets available for our research thesis, and the solution was very similar. Finally, they developed a rescue behavior model based on the dynamic information of the AIS signals such as: latitude, longitude, speed over ground, course over ground, and also time information such as day of week, hour of day and month of year. The two final categories for the individual

AIS points were “rescue” and “non-rescue” based on the shapes of the trajectories and corresponding information about the vessels speed over ground. When it comes to the classification step, standard binary algorithms such as SVM, AdaBoost and Logistic Regression were used, and an accuracy of 96% was achieved which seems extremely good at first sight. However, after further analysis it was found out that the performance of the classifiers was very location dependent, and after removing the latitude and longitude features, the accuracy dropped to 69%. In order to overcome this limitation a clustering approach was performed and a density-based clustering algorithm was used. The density-based algorithm, CB-SMoT [31], was initially created for stop point detection, however, in this case it was adapted to break trajectories into segments with different motion profiles. Even though this approach showed good results, it still contains a few issues: first, clustering points prior to classification reduces the dataset substantially, and second, even after reincorporating position features, it still doesn’t perform as good as the initial point-wise approach.

2.2.2. Anomaly detection in the maritime domain

2.2.2.1. Types of anomalies in the maritime domain. The previously reviewed papers are some more general examples on what machine learning implementations are able to achieve in the maritime domain. Now we are going to focus on some examples more closely related to anomaly detection in the maritime domain, analyze their approaches, and find out the state-of-the-art methods in this area. But first, it is important to define what types of anomalies there exist in the maritime domain and how can those types be fundamentally characterized. According to [32], there exist two types of maritime anomalies: kinematic anomalies and static anomalies. Kinematic anomalies are defined as anomalies manifested during the motion of the ship, an example being unusual maneuvering, while static anomalies are anomalies manifested by analyzing the properties of the ship, such as ships having unusual cargo. In [32], there is also a mention of some sub-types of anomalies such as: maneuvering, location and interaction anomalies. Maneuvering anomalies have to do with events such as an unexpected change of direction, unusual stopping, unusual speeds considering ship type and area, or inconsistent changes given that we already have some information on what the expected movement patterns would be. Location anomalies are mostly related to vessels not confining their travel route to normal historical travel routes. However, location anomalies could also include vessels positioned in restricted areas, exclusion zones, or in zones where the usual maritime activities are inconsistent with the ship type.

In [33] there are also discussions about the types of anomalies in the maritime domain, albeit they are defined a bit differently. Global anomalies are the types of anomalies involving vessels that change their specified destination. It is possible to detect global anomalies because each vessel is required to declare their specified destination in the AIS system, and if the specified destination does not correspond with the real destination, a system flag is raised indicating anomalous behavior. Local anomalies on the other hand are detected when a vessel deviates from the traffic routes that are considered normal or expected. When these deviations are first detected it might not imply anything yet, because context has to be taken into account, meaning that it could have been simply a deviation because of bad meteorological conditions. However, the bigger the deviation distance from the normal traffic pattern routes, the higher the probability of that particular instance being an anomaly. In order to estimate when a local anomaly has occurred, first local spatial statistics have to be calculated from the maritime traffic network. Then, local anomalies can be detected in real-time, by calculating the distance between the real location of the vessel and the declared maritime route. When the distance is above some threshold value calculated a priori, the instance is flagged as anomalous. On the other hand, global anomalies can only be fully verified when a vessel has arrived at a destination, and then it can be checked if it is actually the specified destination on the AIS system or some other destination.

2.2.2.2. Technical aspects of maritime anomaly detection. After the previous discussions about what kind of anomalies we are expected to encounter, in the next few paragraphs we will get into a bit more detail when it comes to the technical aspects of anomaly detection in the maritime domain. First, it is important to mention that the methods for anomaly detection can be categorized in two classes: point-based anomaly detection techniques, and trajectory-based anomaly detection techniques. Point-based methods are performed in such a way that they highlight individual anomalous AIS points, while trajectory-based methods receive as an input the whole trip trajectory instead of individual points and classifies these trajectories as anomalous or not [34].

The decision to solve the anomaly detection problem in a supervised, semi-supervised or unsupervised manner is based on the characteristics of the specific case that needs to be solved, the amount of labeled data, both for the normal case and the anomalous case, and the approach that needs to be taken, point or trajectory-based. It is mostly the case that there is a lack of labeled data for the anomalous case, so usually an unsupervised method is considered that extracts a normal representation of the data first, based on the assumption that most of the data or trajectories are normal [34]. Statistical models that generate maritime traffic normalcy from historical AIS data can be categorized in three groups: parametric methods, non-parametric methods, and clustering methods. A parametric method widely used in maritime anomaly detection is the so-called GMM (Gaussian Mixture Model). Generally, this approach clusters the data in a multi-dimensional feature space, the features usually being latitude, longitude, speed and orientation, and the goal is to approximate the multivariate probability density functions of normal maritime traffic patterns [34]. When it comes to the non-parametric approach, the usual choice is KDE (Kernel Density Estimation), used to create a non-parametric model of the normal maritime traffic behavior [34]. The most consistently used method in the maritime domain for anomaly detection is the clustering-based method. One of the most popular clustering algorithms used in many researches in this domain is K-Means, a distance-based algorithm, where distances are calculated in order to assign points to specific clusters. However, lately in the maritime domain, the popularity of DBSCAN (Density-based spatial clustering of applications with noise) has increased, mostly because the properties of maritime data are very convenient for density-based approaches, but also because that these approaches can create any number of clusters, which is not the case with distance-based techniques.

In the next few paragraphs we are going to discuss some of the most cited anomaly detection papers in the maritime domain and a recently published one where a Deep Learning approach is considered, without going too deep into the mathematical concepts. One of the most cited papers in this domain is [35] in which motion patterns were extracted from AIS data and then motion anomaly detectors were built from them. In this paper, the first step, which was considered as a preprocessing step, was to extract traffic motion patterns from the historic AIS data that are going to be used as training data. The motion pattern extraction is done using the usual data mining techniques. To perform anomaly detection, Kernel Density Estimation (KDE) is used, and it is applied sequentially to the new incoming data [35]. In order to simplify the problem of motion pattern extraction, the only kinematic information included as features are the position, latitude and longitude, and speed. When it comes to other static features, as usual, the vessel types and seasonality information are particularly useful. Then, after having all motion pattern trajectories available that would serve as a normalcy behavior training set, there are two possibilities to explore: one, determining anomalous trajectories sequentially, so checking if the trajectories of the test set lie within the normalcy boundaries, and two, motion prediction. In [35] it is mentioned that the classification problem the authors are trying to solve is defined as one-class classification and it is usually solved using Support Vector Machines (SVM), however, as mentioned previously, they tackled this problem using an adaptive Kernel Density Estimator (KDE) method instead. As a quantitative measure of performance, a function that measures the probability of false alarm was developed in order to evaluate performance of the detector. Firstly, the anomaly detector was tested using 100 artificially created vessel trajectories, and then as a final test, two tests were performed using trajectories constructed from real AIS data. One interesting aspect that caught the eye of the authors was that velocity turned out to be a very important feature when defining a motion pattern. At times even when the vessel's position was actually inside the boundary of the normal behavior, the anomaly detector raised a flag because the velocity vector was incompatible with the training data.

A very interesting paper published recently is [36] where the authors analyzed a

40 GB dataset containing more than 300 million messages collected in a period of one week on the USA coasts. The most interesting aspect of [36] is that the authors are trying to make a case for the usage of Deep Learning approaches to tackle anomaly detection problems in the maritime domain, which, even with the emergence of neural networks lately, doesn't seem to be that common in this domain. Before stepping to the Deep Learning part, some initial exploratory data analysis and classification steps were performed, using both supervised and unsupervised approaches. An unsupervised clustering approach was used to define groupings of AIS messages, and standard supervised machine learning classification approaches were used to classify ship types based on AIS features that could be extracted from the AIS messages. The goal was to use the quantified performance of the standard machine learning algorithms as baseline and compare them to the potential deep learning techniques. As a clustering model, WEKA's K-Means was implemented, the main factor in favor of it being that it is known to scale well to large datasets. Experiments were performed using two different distance functions: Euclidean and Manhattan. Different k values were experimented with, where values ranged from $k = 5$ up to $k = 50$. The results of the clustering step indicated natural groupings of AIS messages based on a set of basic features extracted from those AIS messages. These features were used as an input to the standard classification algorithms to predict per-vessel information, mainly vessel types. In order to acquire the labels of the vessels types, the Marine Traffic database was queried, where the associated IMO (International Maritime Organization) of each vessel was used as a unique identifier to merge the labels to the dataset. This resulted into the creation of seven vessel type categories: tanker, cargo, passenger, fishing, tug, sailing and the category "others". The standard classifiers that were used as a baseline were: Naive Bayes, Random Forests, K-Nearest Neighbors, Support Vector Machines and a simple version of the multilayer perceptron. The resulting performances of the classifiers seemed to surprise the authors of [36] because it turned out to be relatively poor, the best classification accuracy being 63% achieved by Random Forests. The authors argued that this relatively low accuracy is a result of the small dataset that they had available, but also the short period of from which the AIS messages were collected (one week). They argued that the sampling bias problem could be solved if a

bigger dataset covering longer periods of time was available. In order to develop a deep learning anomaly detection system, a Convolutional Neural Network (CNN) approach was considered. The authors set a goal for this system that once properly trained, it should be able to flag anomalous AIS messages within half an hour of their reception [36]. The CNN approach they considered was to obtain 2D images from historical AIS data partitioned into specific time-interval segments. The base stations would be represented by circular shapes, the vessels by box shapes, while the shape, size and color the trajectory would provide enough information to encode different features of the AIS data. However, disappointedly, it seems that the authors did not implement the deep learning approach yet.

Another interesting paper when it comes to anomaly detection in the maritime domain is TREAD [37], where an unsupervised and incremental learning approach for the extraction of maritime patterns is presented that acts as a basis for automatic anomaly detection. New research in this domain has also been performed in the context of the European Commission H2020 funded project BigDataOcean [38], where in the paper [39] an overview of the "Maritime Security and Anomaly Detection" pilot is discussed. In [39], unsupervised machine learning methods and behavioral analytics are utilized to automatically model shipping routes, construct vessel specific profiles, and detect deviations of normalcy patterns in real time.

2.3. Gap in the literature

As we can see from the literature review in this chapter, there have been many interesting machine learning based researches carried out in the maritime domain such as mapping maritime routes, mapping fishing activities, event based knowledge discovery, etc. Research has also been carried out as far as anomaly detection is concerned, such as detecting low-likelihood behavior, detecting spoofing or detecting AIS on/off switching. However, the detection of ship waste discharge, a topic that can be defined as an anomaly detection problem, seems to be an interesting but understudied problem, at least as far as machine learning methods are concerned. The gaps in literature that we are trying to fill or extend by performing this research can be defined in two points:

- Research the machine learning potential for detecting ship waste and residue discharge.
- Extend the maritime anomaly detection literature, methods and approaches as far as the North Sea is concerned.

3. BACKGROUND ON METHODS

3.1. Supervised classification methods

3.1.1. Decision Trees

Decision trees are a non-parametric supervised learning method used for classification and regression [40]. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. The decision tree consists of nodes that form a rooted tree, meaning it is a directed tree with a node called "root" that has no incoming edges. All other nodes have exactly one incoming edge. A node with outgoing edges is called an internal or test node. All other nodes are called leaves (also known as terminal or decision nodes). In a decision tree, each internal node splits the instance space into two or more sub-spaces according to a certain discrete function of the input attributes values [41]. In the simplest and most frequent case, each test considers a single attribute, such that the instance space is partitioned according to the attribute's value [41]. Each decision tree leaf is assigned to the class representing the most appropriate target value.

3.1.1.1. Random Forests. Just like Gradient Boosting Machines, Random Forests are an ensemble learning method consisting of many decisions trees used for a multitude of tasks such as classification, regression or other tasks [42]. Significant improvements in classification accuracy have resulted from growing an ensemble of trees and then letting them vote for the most popular class [42]. In order to grow these ensembles, often random vectors are generated that govern the growth of each tree in the ensemble, and this is usually done using the so-called "bagging" method. Bootstrap Aggregation, or bagging, is a general procedure that can be used to reduce the variance for those algorithms that have high variance. The general idea of the bagging method is that a combination of learning models increases the overall performance.

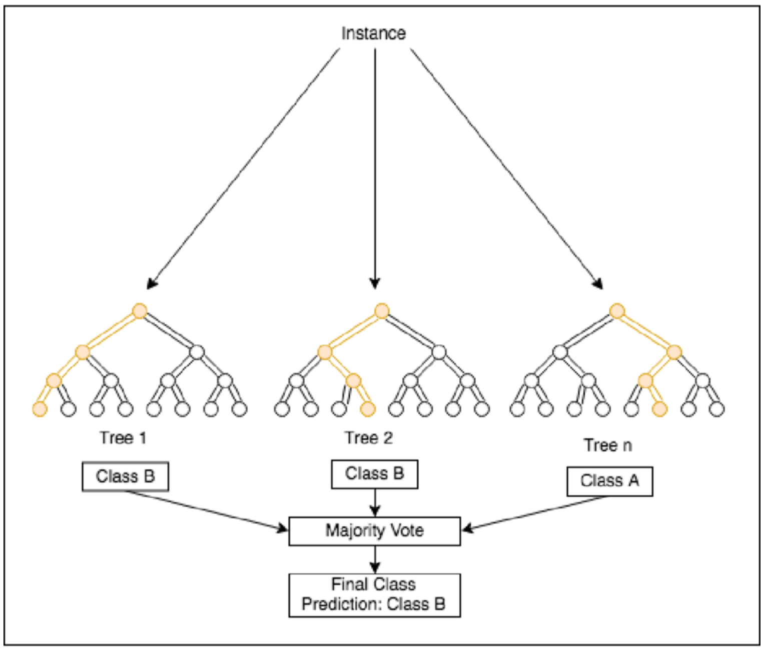

Figure 3.1: Random Forest diagram (Adapted from [1])

3.1.2. Gradient Boosting Machines

Boosting is one of the techniques that uses the concept of ensemble learning where a boosting algorithm combines multiple simple models, or weak learners, to generate the final output. In gradient boosting machines, or simply, GBMs, the learning procedure consecutively fits new models, usually decision trees, to provide a more accurate estimate of the response variable [43]. So, rather than training all the models in isolation of one another, Gradient Boosting Machines train models in succession, with each new model being trained to correct the errors made by the previous ones. Models are added sequentially until no further improvements can be made. In more simple terms, what happens is we look at all the observations that the machine learning algorithm is trained on, and we leave behind only the observations that the machine learning method classified correctly, stripping out the other observations. Then, a new weak learner is added and tested on the set of data that was poorly classified, where, again, just the examples that were successfully classified are kept. The final objective of the Gradient Boosting Machines is to minimize the loss of the model using a gradient descent like procedure. Gradient boosting machines internally solve a regression problem, however, their power comes from the fact that they are not only used on regression problems, but also on binary classification and multi-class classification problems, and at the same time they exhibit computationally fast boosting procedures [43].

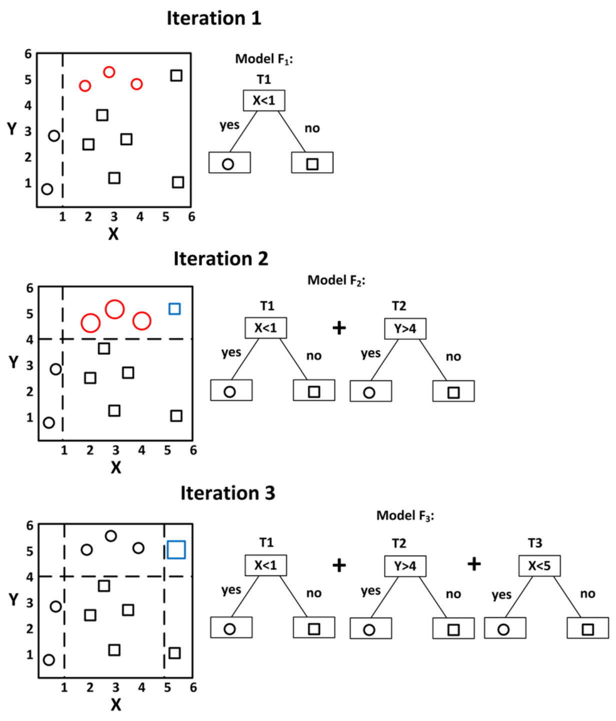

Figure 3.2: Gradient Boosting Machine simple example (Adapted from [2])

3.1.2.1. XGBoost. XGBoost is a refined and customized version of a gradient boosting decision tree system, created with performance and speed in mind. XGBoost actually stands for Extreme Gradient Boosting, and it refers to the fact that the algorithm has been customized to push the limit of what is possible for gradient boosting algorithms. The popularity of XGBoost is a result of it winning a considerable amount of Kaggle competitions where implementations of multiple algorithms and techniques are competing with each other to solve specific tasks. The most important factor behind the success of XGBoost is its scalability in all scenarios. The system runs more than ten times faster than existing popular solutions on a single machine and scales to billions of examples in distributed or memory-limited settings [44].

3.1.3. Support Vector Machines

A Support Vector Machine (SVM) is a linear discriminative classifier that constructs a hyperplane or set of hyperplanes in a high or infinite-dimensional space, which can be used for classification, regression, or other tasks like outlier detection [45]. The simplest formulation of Support Vector Machines is the linear one, where the hyperplane lies on the space of the input data $\mathbf{x}$ [46]. In this case the hypothesis space is a subset of all hyperplanes of the form:

$$f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + b \tag{3.1}$$

Given a training set of instance-label pairs $(\mathbf{x}_i, y_i)$, $i = 1, ..., l$ where $\mathbf{x}_i \in \mathbb{R}^n$ and $y \in \{1, -1\}^l$, the support vector machines require the solution of the following optimization problem [47]:

$$\underset{w,b,\xi}{min} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{l}\xi_i \tag{3.2}$$

$$\textit{subject to} \quad y_i(\mathbf{w}^T\phi(\mathbf{x}_i) + b) \geq 1 - \xi_i, \quad \xi_i \geq 0$$

Here training vectors $\mathbf{x}_i$ are mapped into a higher (maybe infinite) dimensional space by the function ϕ. Support vector machines find a linear separating hyperplane with the maximal in this higher dimensional space. $C > 0$ is the penalty parameter of the error term. Furthermore, $K(\mathbf{x}_i, \mathbf{x}_j) \equiv \phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j)^T$ is called the kernel function [47]. A kernel function scales the input vectors from dimension M into a higher dimensional space N. The idea of the mapping is to be able to create the hyperplane, which is linear in dimension N, and is non-linear if the hyperplane is transformed back to dimension M. There are four basic kernels: linear, polynomial, radial basis function (RBF) and sigmoid.

3.2. Dimensionality reduction

3.2.1. Principal Component Analysis (PCA)

Large datasets are increasingly widespread in many disciplines. In order to interpret such datasets, methods are required to drastically reduce their dimensionality, such that most of the information in the data is preserved [48]. Many techniques have been developed for this purpose, Principal Component Analysis (PCA) being one of them. Principal Component Analysis is a dimensionality reduction method that is often used to reduce the dimensionality of large data sets. This is done by transforming a large set of variables into a smaller one that still contains most of the information that the large set consists [49], so it preserves as much variability as possible. Reducing the number of features of a dataset might come at the expense of accuracy, or not, but the trick in dimensionality reduction is to trade a little accuracy for simplicity.

Usually the dataset is standardized before performing PCA. The reason why it is important to standardize the data prior to applying PCA, is that PCA is sensitive regarding the variances of the initial variables. This means that if there are large differences between the ranges of initial variables, those variables with larger ranges will dominate over those with small ranges and it leads to biased results. The next step is computing the covariance matrix of all variables. The covariance matrix is computed to understand how the variables of the dataset are varying from the mean with respect to each other, or in other words, to see if there is any relationship between them. Sometimes, variables are highly correlated with each other in such a way that they contain redundant information, and this means that the dataset can be reduced to smaller dimensions without losing too much information. The final step consists of computing the eigenvectors and eigenvalues of the covariance matrix to identify the principal components. The principal components are new variables that are constructed as linear combinations or mixtures of the initial variables [48]. These combinations are done in such a way that the new principal components are uncorrelated and most of the information within the initial variables is compressed into the first components [49].

3.3. Unsupervised learning methods

3.3.1. Cluster analysis (Clustering)

Clustering, considered as the most important question of unsupervised learning, deals with the data structure partition in unknown area and is the basis for further learning [50]. A classic definition for clustering is described as follows [51]:

- A cluster is a set of entities which are alike, and entities from different clusters are not alike.
- A cluster is an aggregation of points in the test space such that the distance between any two points in the cluster is less than the distance between any point in the cluster and any point not in it.
- Clusters may be described as connected regions of multi-dimensional space containing a relatively high density of points, separated from other such regions by a region containing a relatively low density of points.

Cluster analysis itself is not one specific algorithm, but the general task to be solved. It can be achieved by various algorithms that differ significantly in their understanding of what constitutes a cluster and how to efficiently find them. Popular notions of clusters include groups with small distances between cluster members, dense areas of the data space, intervals or particular statistical distributions [50].

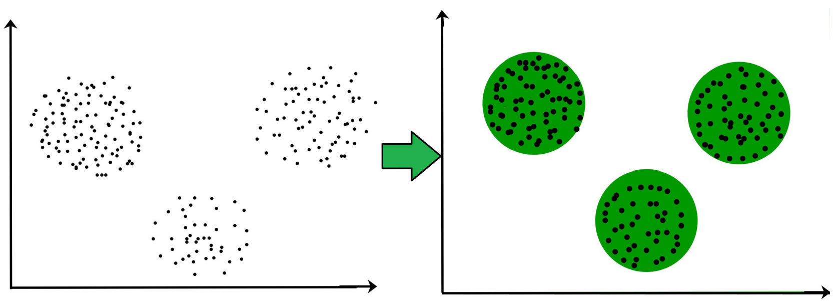
Figure 3.3: Simple clustering visualization (Adapted from [3])

3.3.1.1. Gaussian Mixture Models. Gaussian Mixture Models are a probabilistic model for representing normally distributed subpopulations within an overall population [52]. Mixture models in general don't require knowing which subpopulation a data point belongs to, allowing the model to identify the subpopulations autonomously. Since subpopulation assignment is not known, this constitutes a form of unsupervised learning.

The most commonly used distribution in modeling real-world unimodal data is the Gaussian distribution. Thus, modeling multimodal data as a mixture of many unimodal Gaussian distributions makes intuitive sense. Furthermore, GMMs maintain many of the theoretical and computational benefits of Gaussian models, making them practical for efficiently modeling very large datasets [53].

A Gaussian mixture model is parameterized by two types of values, the mixture component weights and the component means and variances/covariances. For a Gaussian mixture model with K components (clusters), the k^{th} component has a mean of μ_k and a variance of σ_k for the univariate case and a mean of $\vec{\mu}_k$ and covariance matrix of $\sum_k$ for the multivariate case. The mixture component weights are defined as ϕ_k for component C_k, with the constraint that $\sum_{i=1}^{K} \phi_i = 1$ so that the total probability distribution normalizes to 1.

Models are typically learned by using maximum likelihood estimation techniques, which seek to maximize the probability, or likelihood, of the observed data given the model parameters. Expectation maximization (EM) is an iterative algorithm and has the convenient property that the maximum likelihood of the data strictly increases with each subsequent iteration, meaning it is guaranteed to approach a local maximum or saddle point. Expectation maximization for mixture models consists of two steps [53]: The first step, known as the expectation step, consists of calculating the expectation of the component assignments C_k for each data point $x_i \in \mathbf{X}$ given the model parameters ϕ_k, μ_k, and σ_k. The second step, known as the maximization step, consists of maximizing the expectations calculated in the expectation step with respect to the model parameters. This step consists of updating the values ϕ_k, μ_k, and σ_k. The entire iterative process repeats until the algorithm converges, giving a maximum likelihood estimate. Once the EM algorithm has run to completion, the fitted model can be used to perform various forms of inference. The two most common forms of inference done on GMMs are density estimation and clustering.

3.4. Cross-validation

Cross-validation is a statistical method of evaluating and comparing learning algorithms by dividing data into two segments: one used to learn or train a model and the other used to validate the model [54]. In typical cross-validation, the training and validation sets must cross-over in successive rounds such that each data point has a chance of being validated against. The basic form of cross-validation is called k-fold cross-validation, where k refers to the number of groups that a given dataset is to be split into.

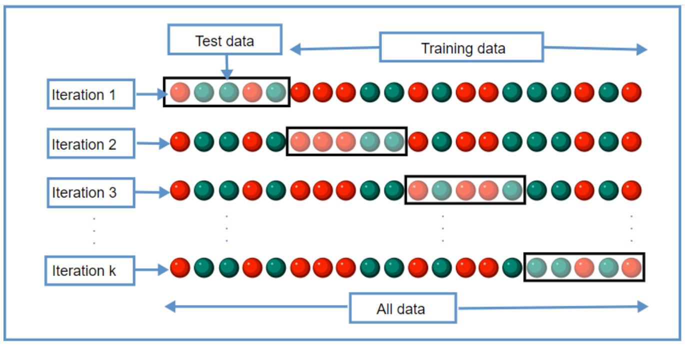

Figure 3.4: Diagram of k-fold cross-validation (Adapted from [4])

3.5. Majority voting ensemble

Majority voting learning is primarily used to improve the (classification, prediction, function approximation, etc.) performance of a model, or reduce the likelihood of a single unfortunate wrong prediction of an instance [55]. There are three versions of majority voting: (i) where the ensemble predicts the class which all classifiers agree with (unanimous voting); (ii) the ensemble predicts the class predicted by at least one more than half the number of classifiers (simple majority); or (iii), the ensemble predicts the class that receives the highest number of votes, whether or not the sum of those votes exceeds 50% (majority voting).

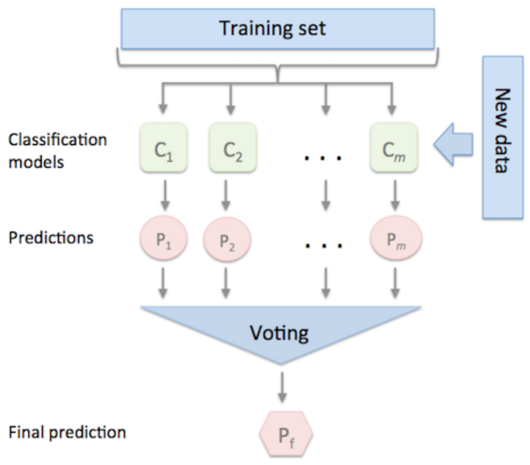

Figure 3.5: Majority voting depiction (Adapted from [5])

4. DATA UNDERSTANDING AND PREPARATION

4.1. General description

In this chapter, we are going to discuss the data available to us for this research, their source, their limitations, the labeled cases that we have, the exploratory data analysis part and some aspects of data preprocessing. There are two datasets available for us: the AIS dataset and the Portcall dataset. The merging of these two datasets is proposed so that we can use both dynamic information about a trajectory from the AIS dataset and static information from the Portcall datasets to learn when a particular trip begins and from which of the ports that we are interested in.

4.2. Automatic Identification System (AIS) dataset

The main source of the data is the AIS dataset. As we have mentioned previously in the AIS section of Chapter 1, the Automatic Identification System (AIS) is a mandatory location signal transmitted by all ships at sea with a high frequency (seconds to minutes). Our dataset is composed of AIS data collected by The Netherlands Coastguard, consisting per-vessel information such as latitude, longitude, speed, orientation, and so on. The AIS signals collectively represent the trajectories of the ships in a time series data representation. The two months of AIS data available to us have a time range starting from 04-01-2017 up to 05-31-2017 and contain over 64 million AIS messages. The range of the coordinates it contains after preprocessing is constrained to the North Sea, with the minimum latitude being 59.8 degrees, the maximum latitude being 62.0 degrees, the minimum longitude being -5.3 degrees, and the maximum longitude being 12.9 degrees. The correct bounding coordinates are taken from Marine Regions [56], which is a website that helps create a standard, relational list of geographic names containing information and maps of the geographic location of these features.

The AIS dataset originally contains 64 million records (AIS messages) and 34 features. Each of the AIS messages contain updates categorized as dynamic and static. The dynamic features of the AIS dataset contain kinematic and timing information of the vessel trips. Some of the most important dynamic features are: update time, latitude, longitude, orientation, rate of turn, length, breadth, navigational status, speed and heading. Update time contains the timestamp when the AIS message was updated and sent, and this is done with a high frequency usually in seconds or minutes. Latitude is a geographic coordinate that specifies the South-North position of a point on the Earth's surface. It is an angle which ranges from 0 at the Equator to 90 at the poles. Longitude is a geographic coordinate that specifies the East–West position of a point on the Earth's surface. The range of longitude is 360 degrees (180 degrees East and 180 degrees West), measured from the Greenwich meridian at which longitude is defined to be 0 degrees. Latitude and longitude together represent the exact position of the vessel at a particular point in time. Orientation (bearing) is the angle in degrees (clockwise) between North and the direction to the destination and it ranges from 0 to 360 degrees. The speed feature, or speed over ground, represents the speed of the vessel and it has a 0.1 knot (0.19 km/h) resolution ranging from 0 to 102 knots (189 km/h).

The static features of the AIS messages on the other hand contain data that are fixed, or need manual changes from the ship's operator. Some of the most important static features of the AIS dataset are the IMO (International Maritime Organization) number, the ship type and the cargo type. The IMO (International Maritime Organization) number is a seven-digit code which shows the ships unique identity. The ship type feature, obviously, tells what kind of vessels we are dealing with. It can be a cargo ship, a fishing ship, a military ship, a passenger ship, a pleasure/sailing ship, a tug tow, a tanker, or other. An important aspect that we have to reiterate about the static information of the AIS messages is that they are manually inserted by the vessel's operator and are prone to human error. This human error can simply be a mistake, or at times it can be deliberate disinformation. This means that this information might not be totally reliable, and if possible to do so, should be taken into consideration.

4.3. Portcall dataset

A port call is registered whenever a vessel enters/leaves the port area. The portcall data provide information that can be used to find the ports visited by specific vessels or to find all vessels that have visited specific ports. The portcall data of 2017 are available to us for this research and are provided by the European Maritime Safety Agency. We are only interested in the data belonging in the date range between 04-01-2017 and 05-31-2017 because the intention is to merge it with the AIS dataset. The portcall data contain important information such as the IMO numbers of the ships, when the port call was sent, actual times of arrival, actual times of departure, port names, previous ports, etc. It also contains a different ship type description feature than the AIS dataset, one that could turn out to be very useful when filtering ships based on what we need. The AIS and Portcall datasets can be linked through unique IMO numbers.

4.4. Limitations of the datasets

One of the limitations that might arise is the fact that we only have two months of available AIS data. At this point, we don't know if this amount of data will turn out to be enough for what we are trying to achieve. Another limitation is the small number of labeled cases, 20, which makes supervised learning very difficult. Another, not exactly a limitation, but it would have been potentially helpful, is that supplementary information such as meteorological and bathymetry data are not available. This could have helped to add more depth in order to understand the particular cases we are interested in by proving more contextual information.

4.5. Data exploration

The AIS dataset initially consists of 64 million rows and 34 features, however, not all of them are necessary or useful for our research project. When it comes to the number of records that we are going to use it will dramatically decrease when we focus on the types of ships that we are interested in and the ports that we are interested in. When it comes to the features of the AIS dataset, a number of them are empty of unrelated to this study.

Initially 13 features were dropped from the AIS dataset, namely: *t_sensors*, *t_atonoffpos*, *t_no_orientation*, *t_status_lost*, *t_status_not_stable*, *t_status_label_lost*, *t_vesseltype*, *p_sourcename*, *p_antposfront*, *p_antposleft*, *p_atontype*, *p_remark*, *p_vesseltype*. Most of these features were empty, while some of them contained unnecessary information such as the position of the antennas on the ship, or the number of sensors a ship contains.

After the initial feature selection step, we were left with 21 features that were potentially useful to us. It has to be mentioned that because of the very large size of the AIS dataset, it was not possible to load it completely in memory at the same time. So the initial analysis were performed by dividing the dataset into April and May 2017. However, the analysis results for both months were extremely similar. In Figure 4.1 we show a missing values bar chart of the AIS data of April 2017.

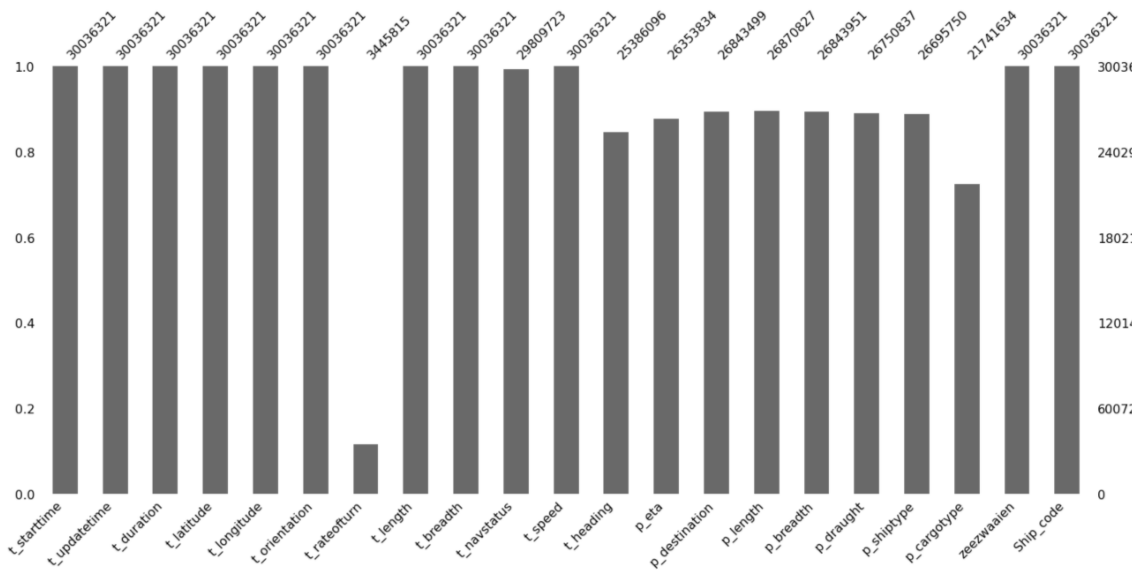

Figure 4.1: Amount of non-missing values for each AIS feature

The first thing that can be noticed is that most of the features, especially the most important ones such as *t_latitude*, *t_longitude*, *t_speed* and *t_orientation* have no missing values. The second thing that can be noticed is that the feature *t_rateofturn*, which indicates the instantaneous rate at which the ship is turning, contains around 90% of missing values and needs to be dropped. The third thing that can be noticed is that the feature *p_cargotype*, which contains information about what kind of cargo the ship contains, has around 30% of its values missing, and this could be a small disadvantage during the part where it would be useful to know what ships are actually carrying and if they contain chemicals that could be discharged at sea. Finally, just to make it clear, the *zeezwaaien* column contains the labels of the trips indicating whether they potentially performed waste discharge behavior. We are going to get into more details about the labeled cases in the next few sections.

In Figure 4.2 we can see the top 10 destinations specified on the AIS messages by the ship operators during April 2017.

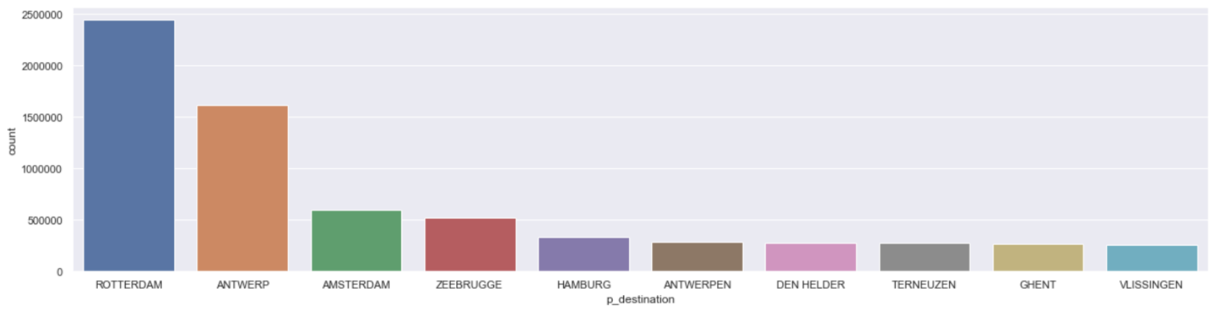

Figure 4.2: Top 10 Destinations - April 2017

The majority of the AIS messages have Rotterdam specified as the destination of the ship. This is good news because we are primarily interested in the port of Rotterdam and this signifies that we should have a lot of data to work with. Amsterdam is another port of interest for us, however, as we can see, it contains a much smaller amount of AIS messages related to it. Looking at the bar chart, we see that the port of Antwerp occurs twice, having values written both in Dutch and English. This was fixed, and they were mapped into one single value.

The Figure 4.3 shows the types of cargo indicated on the AIS messages and their frequency during April 2017.

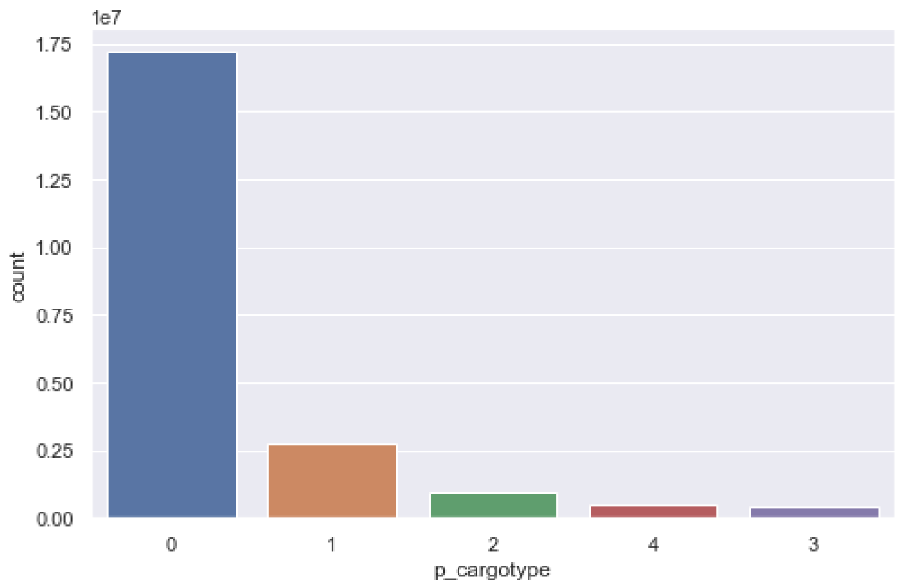

Figure 4.3: Cargo types and the number of occurrences - April 2017

The mapping of the cargo types to numerical values is done as following:

- 0 = Unspecified
- 1 = Major hazard
- 2 = Hazard
- 3 = Minor hazard
- 4 = Recognizable hazard

An observation that can be made from the Figure 4.3 is that along with a relatively high percentage of the cargo type values missing in the dataset, as seen previously, most of the ship operators don't manually input the real cargo type that the ship is carrying and simply leave it as unspecified. This is a big disadvantage considering that it is very important to know what a ship is actually carrying in order to determine if it could potentially be involved in waste discharge behavior. Also, in the case that we do find anomalous behavior, it would help give us extra contextual information. This disadvantage could be softened when merging the AIS dataset with the Porcall dataset. The Portcall dataset also contains a feature describing the ship type and indicating what that particular ship is carrying.

In Figure 4.4 we can see the number of tankers that have sent a port call in Europe during April-May 2017 and their ship type descriptions, a feature from the Portcall dataset.

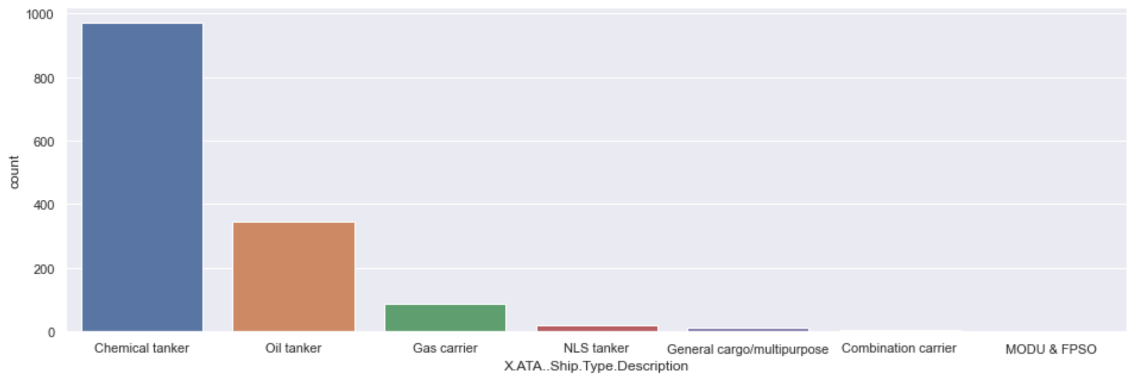

Figure 4.4: Portcall tanker ship type descriptions and the number of port calls during April-May 2017

Looking at Figure 4.4 we can see that there exist multiple sub-types of tankers, but, as mentioned previously we are not interested in all ship types because most of them are not associated with waste discharge behavior. The small number of labeled cases that we have and discussions with domain inspectors instructed us to focus on tankers, more specifically chemical tankers and NLS (Noxious Liquid Substances) tankers.

And finally, in the Figure 4.5 we can see the distributions of some features of the AIS dataset. From here we see a clearer picture on the ranges of these particular features, and we have our first insights at where we should look for some erroneous points that need to be removed. For example, we can see some small bumps at the tails of the *t_latitude*, *t_longitude* and *t_speed* features. That is a first indicator that there are some erroneous points with impossible values that should be looked into more carefully and maybe removed or simply be considered as anomalies.

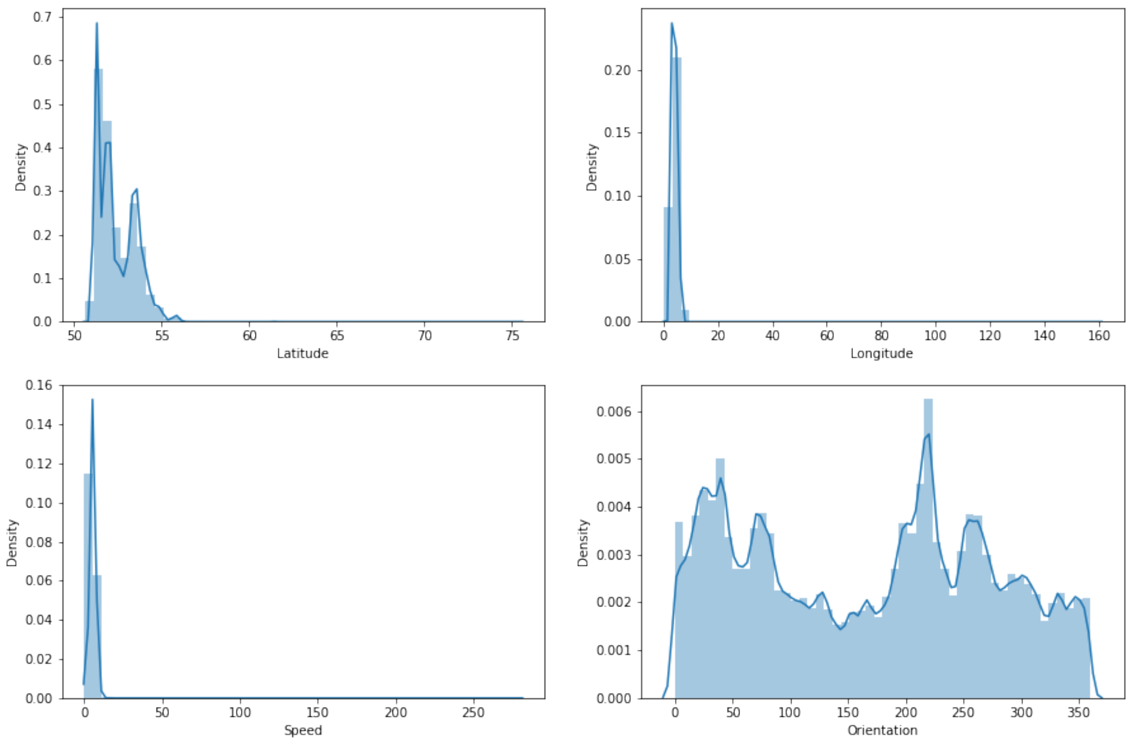

Figure 4.5: AIS feature distributions (Density plot) - April 2017

4.6. Labeled cases

In this section we are going to discuss the small number of labeled cases that we have and the process that was performed to acquire them. First, it is important to mention that the portcall data contain information about both the origin and the destination of ships, meaning that to extract the initial potentially anomalous cases it was not necessary to rely only on the self-reported destinations from the AIS dataset. This leads us to the labeled cases acquiring process: IDLab data scientists using the AIS and Portcall datasets filtered and merged ships that left the ports of Rotterdam or Amsterdam and returned to the same port within 48 hours without visiting another port. This so-called “looping” behavior is a good indicator that waste discharge behavior was performed. Then, with the help of domain inspectors, 20 cases were labeled into two categories. 11 of the cases were labeled as “1”, where the ships left the port of Rotterdam/Amsterdam and returned within 48 hours without visiting another port, but they were classified as trips not expected to have exhibited waste discharge behavior. Likewise, 9 of the cases were labeled as "2", where the ships left the port of Rotterdam/Amsterdam and returned within 48 hours without visiting another port, and the behavior was classified as being typical for waste discharge behavior. However, it is important to note that in none of these cases inspection was performed or ship waste discharge was confirmed.

One of the cases labeled "1", characterized as not expected to have exhibited waste discharge behavior, can be seen in Figure 4.6.

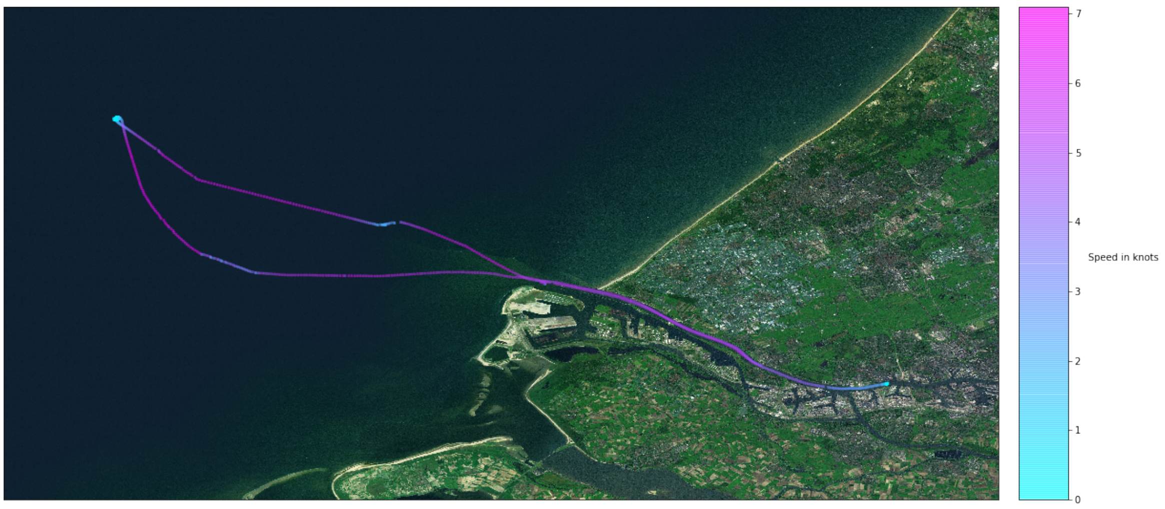

Figure 4.6: "Looping" trip labeled as "1" - not expected to have exhibited waste discharge behavior

Here, also paying attention to the speed color bar, we can see that a ship departs from the port of Rotterdam, goes some way into the North Sea, stays at an anchoring location for a while, and then without visiting another port goes back to the port of Rotterdam. This particular ship does perform the "looping" trajectory, but in this case it was labeled as not expected to have performed waste discharge because it is very common for ships to go to these anchoring locations, stay stationary for a while, not perform waste discharge, and then return to the port. One of the main reasons this happens is because sometimes it is cheaper to just stay at an anchoring location than to go to the port and pay for the port parking fee.

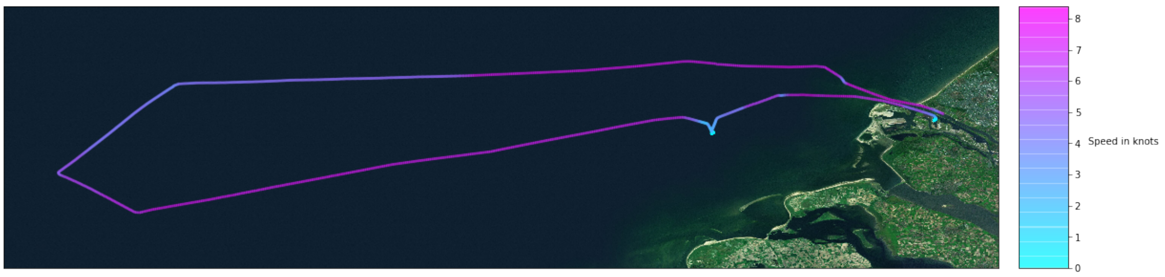

Figure 4.7: "Looping" trip labeled as "2" - typical for waste discharge behavior

Now, in Figure 4.7 we can see one of the cases labeled "2", so characterized as performing typical waste discharge behavior. We reiterate once again that none of the cases were inspected and verified. In this case we can see that the ship departed the port of Rotterdam, went some way into the North Sea, likely performed waste discharge, then went back without visiting another port, stayed stationary for a while at an anchoring location and then went back to the port of Rotterdam. This case and the other 8 cases similar to it are examples of what our targeted anomalous behavior actually is and what we are trying to detect using machine learning methods.

4.7. Preprocessing

4.7.1. Initial feature selection

Initially, 13 features were dropped from the AIS dataset, as described in Section 4.5. Most of these features were empty, while some of them contained unnecessary information such as the position of the antennas on the ship, or the number of sensors a ship contains. After the data exploration phase, $t\_duration$ was also dropped because of its irrelevance to our project, and $t\_rateofturn$ was dropped because of the high percentage of missing values. As far as the AIS dataset is concerned we are left with the following 20 features, the column containing the labels included: $t\_starttime$, *t_updatetime*, *t_latitutde*, *t_longitude*, *t_orientation*, *t_length*, *t_breadth*, *t_navstatus*, *t_speed*, *t_heading*, *p_eta*, *p_destination*, *p_length*, *p_breadth*, *p_draught*, *p_shiptype*, *p_cargotype*, *zeezwaaien*, *t_updatetime*, and *Ship_code*.

When it comes to the Portcall dataset, the features we were mostly interested in were: the actual times of arrival, the actual times of departure, the port names and the ship descriptions. The times of arrival/departure will help us pinpoint the starting time of each individual trajectory, the port names will give us information from which ports do the individual ships depart from, and the ship descriptions will give us another layer of information about what kind of tanker sub-types we are dealing with.

4.7.2. Merging the AIS and Portcall datasets

The first step of the preprocessing phase concerns merging the AIS and Portcall datasets, where the focus will be on tanker ships that have port calls registered in Rotterdam, Amsterdam or Antwerp during the period 04-01-2017 to 05-31-2017. The merging of these two datasets is possible because they both contain the IMO (International Maritime Organization) numbers of the ships, in hashed form to maintain anonymity, to act as a common key for correct merging.

Initially, the AIS dataset was filtered in such a way that we obtained a subset of the AIS data corresponding to tankers. Also, after discussions with domain inspectors, another tanker subset selection was necessary, where we filtered AIS messages that had values $ship\_type\_description = \{Chemical\_tanker, NLS\_tanker\}$, so the subtypes of tankers that could potentially be involved in waste discharge behavior. A large majority of the tankers that we are going to be further inspecting are chemical tankers while a very small number of those are NLS tankers.

When it comes to filtering the Portcall dataset, first we took a list of all the unique *Ship_codes* belonging to tankers in the AIS dataset, and we cross-referenced that list with the *Ship_codes* having made a port call in Rotterdam, Amsterdam or Antwerp during our time of interest. We ended up with 1147 port calls related to Rotterdam, 294 port calls related to Amsterdam, and 862 port calls related to Antwerp.

In order to perform the merge, Pandas *merge_asof*() [57] function was used, a function that does the merging similarly to a left join, however, in this case we match on nearest key rather than equal keys. The key of the AIS dataset is the feature *t_updatetime*, that is a *datetime* feature indicating the time of each AIS message update, while the key of the Portcall dataset is *ATD_LT*, the specified actual time of departure from the ports of interest. Both the datasets have to match on *Ship_code*, the unique ship identifier, before performing the merge operation. The tolerance was set to 48 hours, indicating that all AIS data points in the range starting from the actual time of departure from the ports of interest and up to 48 hours later, or 48 hours before up to the actual time of arrival at the ports of interest, would be merged to that particular trip, depending on which direction we are interested in. We performed two merges as seen in the last sentence: one merge consists of trips arriving at our ports of interest where all AIS points 48 hours prior to the arrival are included, and another merge consisting of trips departing from our ports of interest containing all AIS points up to 48 hours after departure. These values could easily be changed if different time ranges are necessary during experimentation.

Merging different datasets based on time stamps and time tolerances is not a trivial task in general, and this was the case for us too. Both datasets contain a lot of errors, inconsistencies, human errors, gaps and mismatches. They are also both individual datasets created for their own particular purpose and the necessity to merge them only arises because of what we are trying to detect, and therefore needing features and time information from both datasets combined. This leads to some trajectories containing missing points, or in some cases trajectories having some erroneous points that shouldn't be there. This happens because of the aforementioned timing mismatches while at the same time *Ship_codes* and the join keys both matching.

4.7.3. Data cleaning

The next step in preprocessing is data cleaning where erroneous data points such as data points not belonging to the North Sea, or data points containing impossible speed values, are removed.

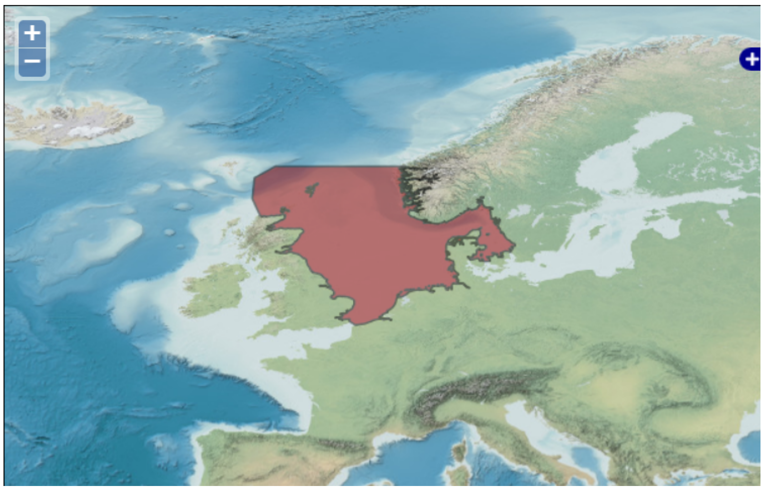

Figure 4.8: The North Sea bounding area - Red area

First, we create a bounding area containing The North Sea by defining the maximum and minimum longitude and latitude values. The visual form of this approach can be seen in Figure 4.8. The correct bounding coordinates are taken from Marine Regions [56] which is a website that helps create a standard, relational list of geographic names containing information and maps of the geographic location of these features. The North Sea is constrained in this way: the minimum latitude being 59.8 degrees, the maximum latitude being 62.0 degrees, the minimum longitude being -5.3 degrees, and the maximum longitude being 12.9 degrees. All data points not belonging inside this bounded area were removed.

The second part of data cleaning had to do with removing erroneous speed data points containing impossible values, such as speeds of 30-200 knots that were encountered. Each data point with $t\_speed > 30$ knots was removed, 30 knots being the maximum speed a tanker ship can achieve.

4.8. Maritime traffic

4.8.1. Maritime traffic visualizaton

First, our initial experiments are going to focus on ships departing from our points of interest, with a bigger focus on Rotterdam, because this is the scenario where most of our anomalous labeled data belong in. The number of trips departing from each port, before removing trips containing big trajectory gaps, can be seen in Table 4.1.

Port	**Number of departing trips**
Rotterdam	746
Amsterdam	203
Antwerp	522

Table 4.1: Number of departing ships from our ports of interest

All AIS data points of the departing tankers are plotted on a map in order to visualize the maritime traffic at the North Sea and have a clearer look on where the usual maritime traffic routes lie. The visualized maritime traffic of the ships departing Rotterdam can be seen in Figure 4.9. Important to note that in Figure 4.9, AIS points are plotted as individual points and are not connected to form trajectories at this point.

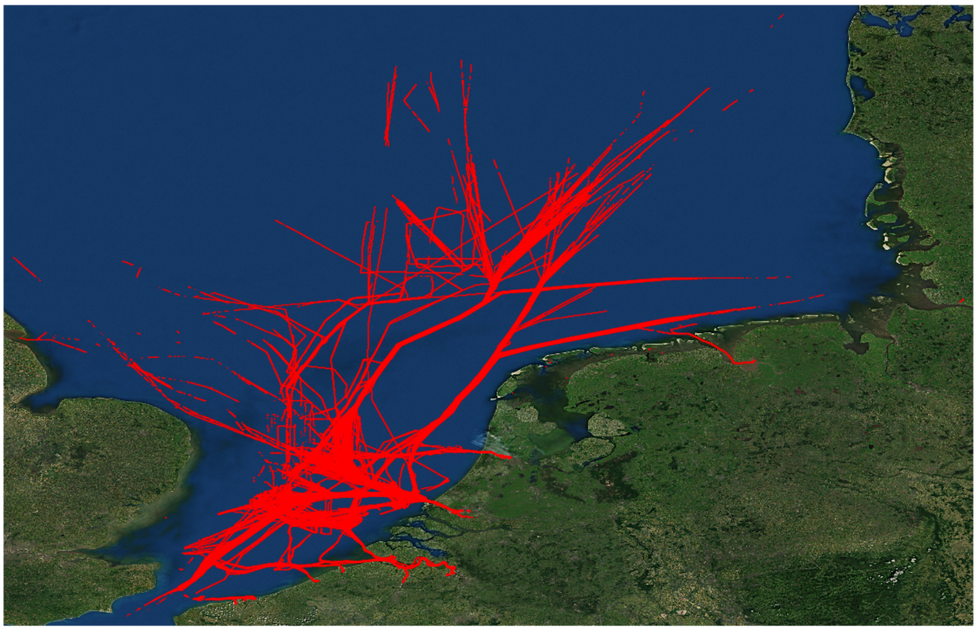

Figure 4.9: Maritime traffic - Departing ships from Rotterdam - 48 hour trajectories

4.8.2. Trajectory removal

As discussed before in Subsection 4.7.2, both AIS and Portcall datasets contain a lot of inaccuracies, trajectory gaps, human errors, mismatches, and so on. Before starting with machine learning experiments for anomaly detection one more trajectory removal step is necessary. This step is carried out in order to remove trajectories with very big gaps between AIS data points or to remove trajectories containing a very small number of AIS data points making them needless or even contaminating.

We are going to start with the port of Rotterdam and the 746 trips departing from it. To carry out the trajectory removal step several approaches were experimented with, ranging from manual to automated. The manual approach consisted of analyzing the trajectory plots of each individual trajectory. Based on the number of points an individual trajectory had and looking at the spatial gaps of a particular trajectory, it was decided if a trip should be left in the dataset or removed. Examples of trips that were removed from the dataset can be seen in Figure 4.10, removed because of very large gaps in the trajectory, and Figure 4.11, removed because it contains a very small number of AIS data points. Likewise, an example of a trip that was considered adequate can be seen in Figure 4.12.

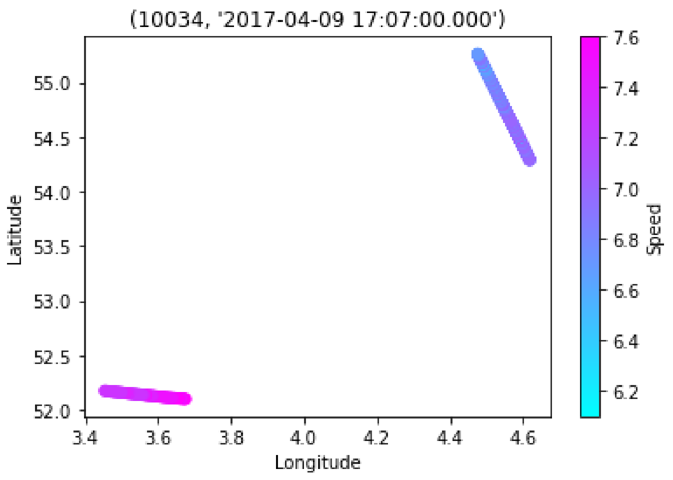

Figure 4.10: Removed trip because of very big gaps in the trajectory

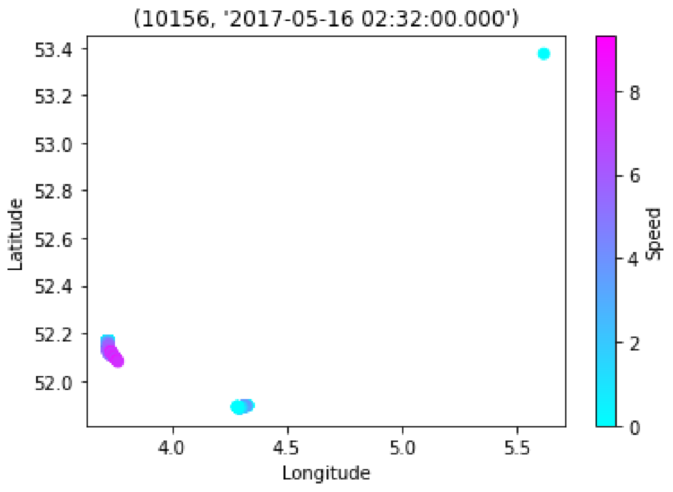

Figure 4.11: Removed trip because of very small number of AIS data points

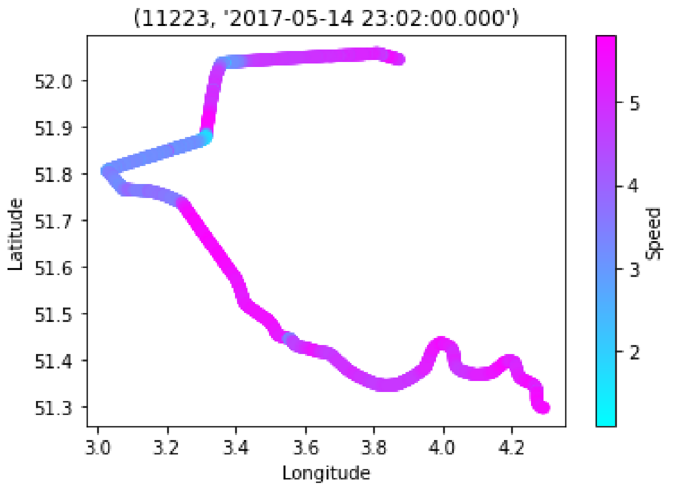

Figure 4.12: An example of a trip trajectory considered as adequate

The automated approach consisted of removing trips that fall below a certain threshold of AIS data points count, or removing trips that fall above a certain threshold of maximum time distance between consecutive points. These, let's say more automated approaches, ended up removing a very large chunk of the trips, even some trajectories that seemed useful at first look. Given that we already are low on data, we decided that for the time being we should stick to the manual approach. The trajectory removal step might need reconsideration as we further progress with our research.

Initially, we had 742 trips departing the port of Rotterdam. After the manual trajectory removal approach 72 trips were removed, meaning that we are left with 670 trips to work with.

4.8.3. Removal of points on land

After the removal of the 72 trajectories, we plotted the connected sequential AIS data points making up the 670 trajectories remaining. Looking at Figure 4.13, we see that a final preprocessing step is needed: removal of erroneous points on land. Erroneous points on land are also a product of the errors and imperfect quality of the AIS data.

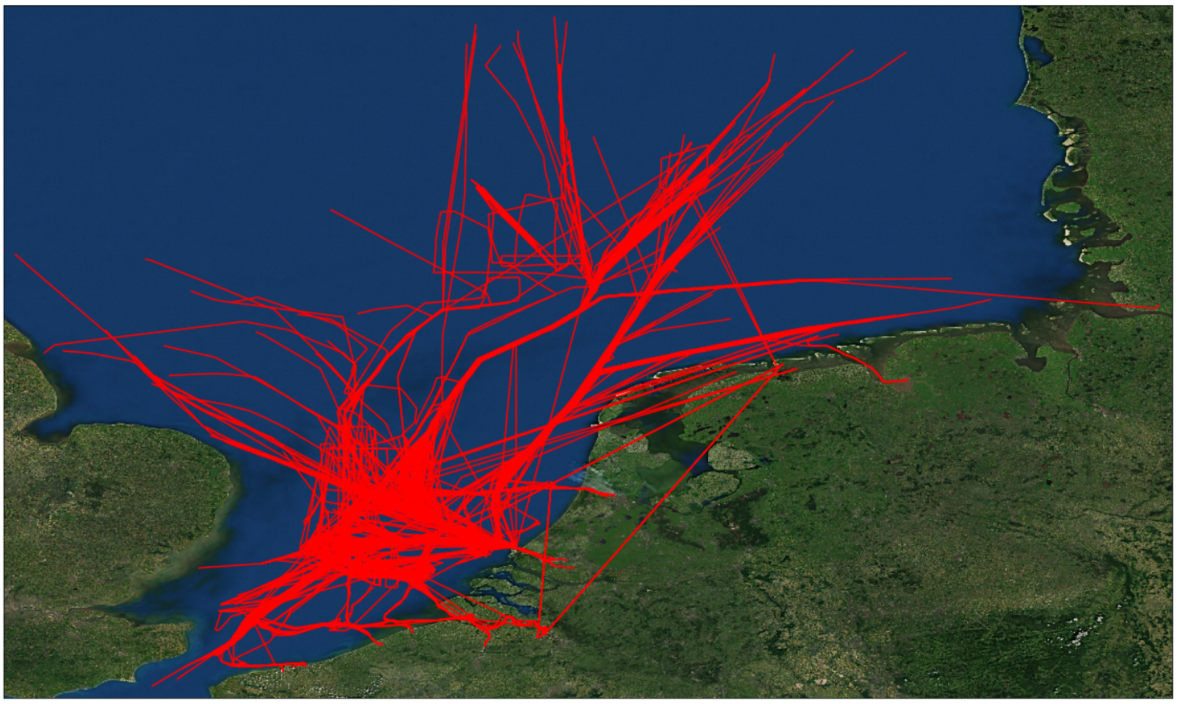

Figure 4.13: Maritime traffic - 670 trajectories departing Rotterdam - with erroneous points on land

This part turned out to be relatively straightforward, helped by the geographical position of The Netherlands. One way to avoid these points on land would have been to narrow the bounding area of the North Sea so that it doesn't come too close to land, however, that would have probably also removed some points at sea, which is undesirable. The approach that we took after analyzing the trajectories was to remove AIS data points for each trajectory where *t_latitude* surpassed the 97.5% quantile threshold value, which for almost all the cases turned out to be the points on land. Different quantile threshold values were tested and a balance was found in order to remove as many data points on land as possible but at the same time to not remove useful data points at sea. The final results can be seen in Figure 4.14 where it is clear that most of the points on land were removed, except for one apparently (AIS data point located in the Wadden Sea north of The Netherlands, an area too shallow to operate large ships).

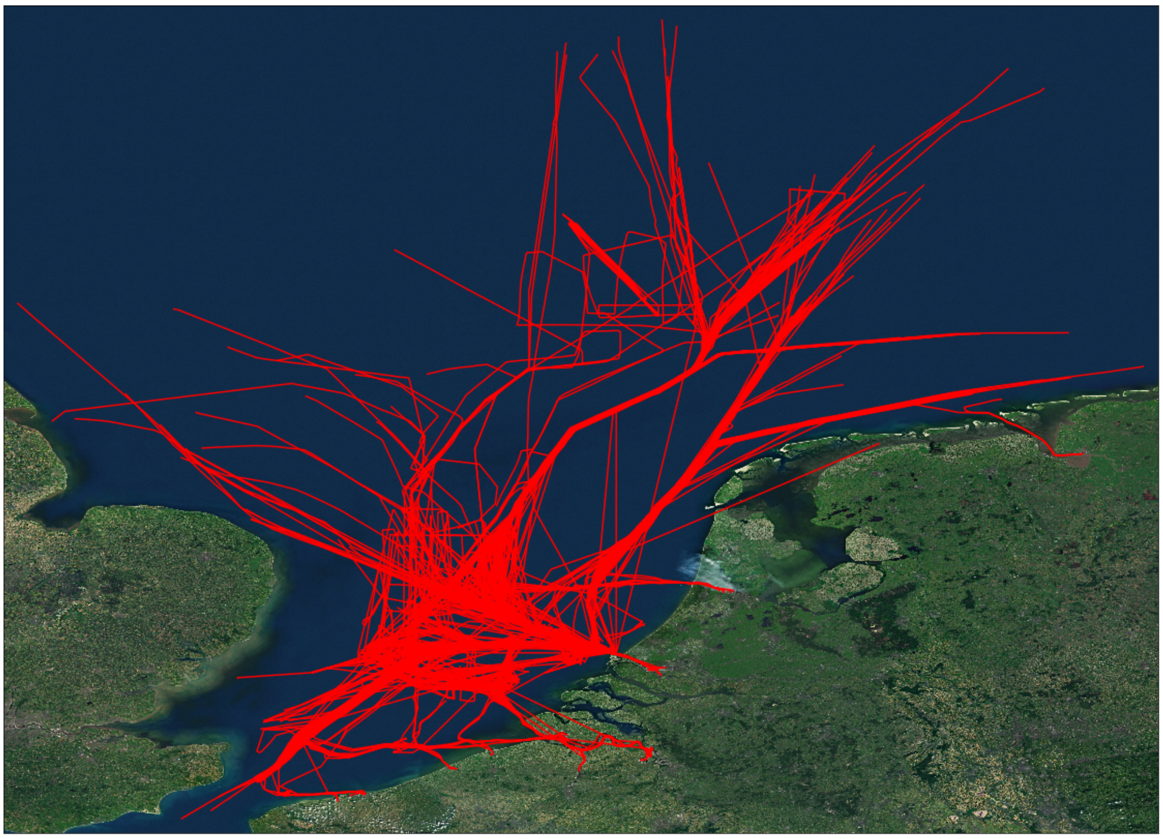

Figure 4.14: Maritime traffic - 670 trajectories departing Rotterdam - without erroneous points on land

4.9. Assumption about trips under 12 hours

After analyzing the anomalous trip trajectories and the information that comes with them, it was noticeable that all of them, except one, had a trip length of over 12 hours. Domain inspectors also confirmed that it is very unlikely for a ship to be involved in waste discharge behavior in such a short amount of time after leaving the port. They suggested that a more likely time interval for this specific behavior after departing from Rotterdam or Amsterdam is between 12 and 48 hours. Based on statistical analysis and expert opinion, it was decided that the AIS data belonging to trajectories with trip length of 12 hours or less will not be used to train our machine learning models.

5. METHODOLOGY

In this chapter we are going to discuss the methodological choices and pipelines that we have considered in order to answer our research questions. First, a quick reminder on what our main research question is:

Given the availability of the AIS dataset and the potential of supervised and unsupervised techniques to solve complex problems: Is it possible to develop machine learning models that detect ship waste and residue discharge at the North Sea effectively (with performance significantly higher than chance level) and efficiently (with minimal cost and preferably real-time)?

5.1. Time series supervised learning

In Chapter 4, we've established that the trajectories of the ships are represented as time series data. Initially, we are going to transform the time series problem into a general supervised learning structure. In order to do so, time series feature engineering is going to be a key part of the process pipeline, where sliding and expanding window transformations will be applied to a number of statistical functionals. The sliding and expanding window approaches take the data of previous time steps as input observations and use them to predict the next observation [58]. This type of data summarizing results in each time series sample being independent of other samples, but at the same time containing information about the preceding historical data, leading to a supervised learning structure representation of the time series problem [58].

5.2. Time ensemble classification pipeline

In Figure 5.1 we can see the complete time series supervised classification process pipeline, the implementation of which will be used to answer the following research questions (and the sub-research questions that they contain):

- What is the effect of summarizing the low-level descriptors that come with the AIS data (e.g. latitude, longitude, speed, orientation), over past records?
- Utilizing the AIS trajectory data, can we train models using previous sequential data points up to a specific point to detect our targeted behavior before occurring, or as early as possible after occurring?

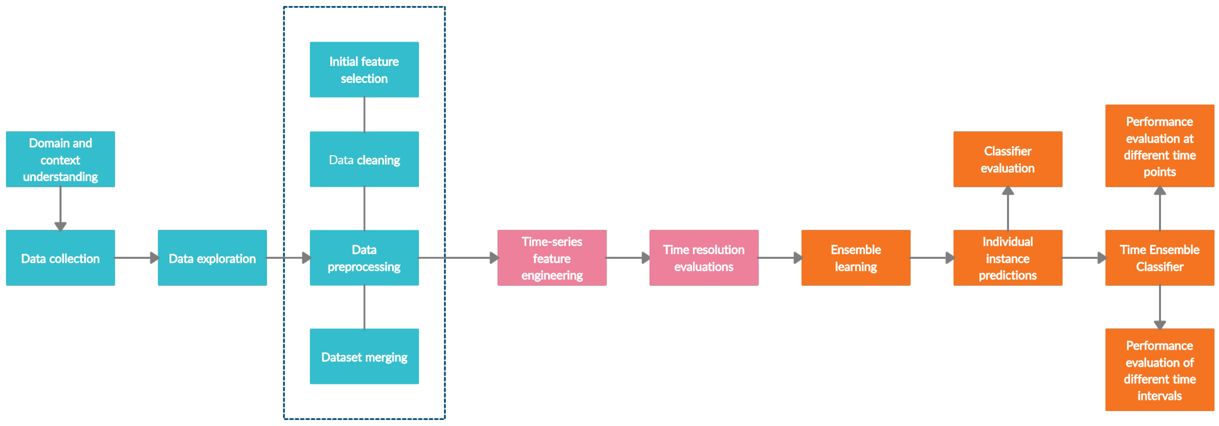
Domain and context understanding
Data collection
Data exploration
Initial feature selection
Data cleaning
Data preprocessing
Dataset merging
Time-series feature engineering
Time resolution evaluations
Ensemble learning
Classifier evaluation
Individual instance predictions
Performance evaluation at different time points
Time Ensemble Classifier
Performance evaluation of different time intervals

Figure 5.1: Time series supervised learning process pipeline

The initial stages of the pipeline seen in Figure 5.1, domain and context understanding, data collection and data preprocessing are discussed in Chapter 1 and Chapter 4. The forthcoming stages of the time series supervised learning pipeline described in Figure 5.1 will be discussed in more details in the subsections below.

5.2.1. Time series feature engineering

We mentioned that the trajectories in the AIS dataset are represented as time series, where the data is captured at specific intervals and each successive data point in the series depends on its past values. Most advanced machine learning algorithms that solve these challenges today are not time-aware. They typically carry out a row-based learning approach, where each row (data point) is an individual entity that doesn't contain any information about other data points. In order to use these methods for forecasting or classification, we need to derive informative features, based on past and present data in time. One of the main techniques utilized to engineer time series features is the sliding window transform approach, in statistics also known as the lag method [59]. This sliding window is the basis of how we turn any time series dataset into a general supervised learning problem [59].

The number of previous time steps that are taken into consideration is called the window size. The size of the sliding window is a very important parameter because when the window size is small it only contains a limited number of observations which might not be enough for a model to properly learn. On the other hand, if the window size is large it increases the risk for the model to learn from observations that may not have an impact on the target variable [58]. A final thing to be careful with is that increasing the sliding window size decreases the training data length, which in turn impacts model learnability. We are going to experiment with three different time resolutions and quantitatively evaluate which window size is the most useful for building accurate models.

5.2.1.1. Sliding window and expanding window. The concept of a sliding window calculation is most primarily used in signal processing and time series data. In very simple words, we take a window of size κ at a time and perform some desired mathematical operation on it. A window of size κ means κ consecutive values at a time. In a very simple case all the values belonging to that particular κ window are equally weighted. The sliding window is utilized to perform the statistical functionals mentioned in Subsection 5.2.1.2 in order to engineer time series features based on the last 10 and 60 minutes of the trajectory.

The expanding window is simply an advanced version of the sliding window technique. In the case of a sliding window, the size of the window is constant while the window slides as we move forward in time. Hence, we consider only the most recent values and ignore the past values. When it comes to the expanding window, with every step the size of the window increases by one as it takes into account every new succeeding value in the series. The expanding window is used to perform the statistical functionals mentioned in Subsection 5.2.1.2 in order to engineer time series features based on full trajectory time resolution, so taking the whole trip into account from start to finish.

5.2.1.2. Statistical functionals.

For each of the four low-level descriptors, *t_latitude*, *t_longitude*, *t_speed*, *t_orientation*, 10 different statistical functionals are going to be performed for three different time resolutions (sliding/expanding window sizes): the full trajectory using an expanding window, the last 10 minutes of the trajectory using a sliding window, and the last 60 minutes of the trajectory using a sliding window. The 10 statistical functionals to be performed on the low-level descriptors for the three time resolutions are:

- Mean - the average of the κ-sized window.
$$mean(x) = \overline{x} = \frac{\sum_{i=1}^{\kappa} x_i}{\kappa} \tag{5.1}$$
- Max - the maximum of the κ-sized window.
$$max(x) = \underset{i}{max}\ x_i \text{ ; where } i \leq \kappa \tag{5.2}$$
- Min - the minimum of the κ-sized window.
$$min(x) = \underset{i}{min}\ x_i \text{ ; where } i \leq \kappa \tag{5.3}$$
- Median - the median of the κ-sized window.
$$median(x) = \frac{o_{\lfloor \frac{\kappa}{2} \rfloor} + o_{\lceil \frac{\kappa+1}{2} \rceil}}{2} \text{ ; } o \text{ ordered list of } \kappa \text{ numbers} \tag{5.4}$$
- Std - the standard deviation of the κ sized window.
$$std(x) = \sqrt{\frac{1}{\kappa - 1} \sum_{i=1}^{\kappa} (x_i - \overline{x})^2} \tag{5.5}$$
- Range - the maximum of the κ-sized window - the minimum of the κ-sized window.
$$range(x) = \underset{i}{max}\ x - \underset{i}{min}\ x \text{ ; where } i \leq \kappa \tag{5.6}$$
- Slope - the slope of a linear polynomial of the κ-sized window.
$$x = a \cdot time\_index + b \text{ ; where } a \text{ is the slope of the line.} \tag{5.7}$$

- Curvature - the curvature of a quadratic polynomial of the κ-sized window.

$$x = d \cdot time\_index^2 + b \cdot time\_index + c \text{ ; where } d \text{ is the degree of curvature.} \tag{5.8}$$

- Relative location of the maximum - the relative location of the maximum value of the κ-sized window divided by the length of the κ-sized window.

$$relative\_location\_max(x) = \frac{argmax(x_i)}{\kappa} \text{ ; where } i \leq \kappa \tag{5.9}$$

- Relative location of the minimum - the relative location of the minimum value of the κ-sized window divided by the length of the κ-sized window.

$$relative\_location\_min(x) = \frac{argmin(x_i)}{\kappa} \text{ ; where } i \leq \kappa \tag{5.10}$$

Performing the statistical functionals on the four low-level descriptors for the three different time resolutions will lead to 121 new features, time elapsed included. This leads to engineering new features that contain historical information about the past data up to that particular point in time, on top of the information already available for each original data point.

5.2.2. Overcoming the class imbalance problem

In order to successfully implement our transformed time series supervised learning approach, we need to take care of the class imbalance problem. Class imbalance is a problem that does exist as far as our dataset is concerned, in which more than 98% of the data points are associated with the class describing the trajectories exhibiting normal behavior, while less than 2% of the data are associated with the class describing the trajectories exhibiting anomalous behavior.

In order to tackle the class imbalance problem several techniques are proposed:

(i) Undersampling the majority class data
   - Undersampling balances the dataset by reducing the size of the majority class. This method is used when the quantity of the data is sufficient, where

in our case, it is, at least for now. By keeping all samples from the minority class and randomly selecting an equal number of samples from the majority class, a balanced new dataset can be retrieved for further modelling [60].

(ii) Using ensemble methods

- The easiest way to successfully generalize a model is by using more data. Ensemble methods combine predictions of different classifiers where each individual classifier is trained with a different subset of the data. Ensemble methods are designed to reduce the bias towards the majority class by focusing on misclassified training patterns [60]. The resampling methods used are principally two, bagging and boosting, however, it is also possible to perform semi-manual resampling where we have more control on what goes in the training/testing sets.

(iii) Resampling with different ratios

- The ensemble method can be fine-tuned by changing the ratio between the majority and the minority class. Instead of training all the ensemble models using the same ratio, it is possible to experiment with different ratios and analyze the effect it has on performance [61]. So if, for example, 10 models are trained, it is worth experimenting with models that have a ratio of 2:1 (majority:minority), another one with 3:1, up to a ratio of 10:1.

(iv) Selecting appropriate evaluation measures

- The conventional model evaluation measures (mainly accuracy) do not accurately measure model performance when faced with imbalanced datasets. Standard classifier algorithms have a bias towards classes which have the highest number of instances. They tend to only predict the majority class data while the feature vectors of the minority classes are treated as noise and are often ignored. This means that there is a high probability of misclassification of the minority classes compared to the majority class. This leads to the selection of other alternative evaluation measures that can be applied such as Precision, Recall or F1 score [61].

5.2.3. Time resolution performance evaluation & feature set selection

Feature selection is one of the core concepts in machine learning which dramatically impacts the performance of the models [62]. In our time series supervised learning pipeline, the time resolution testing and feature selection stage is concerned with the following points:

- Testing the engineered time resolution feature sets individually and analyzing the effect they have on the performance on their own.
- Comparing the performance of the initial models when using only the low-level descriptors as input features versus the performance of the models when using all the engineered features as input. The difference in performance to be quantitatively evaluated.
- Selecting the optimal feature set to maximize model performance.
- Utilizing feature selection and dimensionality reduction techniques to train models efficiently.

The supervised learning algorithms that will be used to train the initial models will be the ensemble learning techniques discussed in Section 3.1, XGBoost and Random Forests.

5.2.4. Time Ensemble Classifier

In this subsection, we discuss one of the main methodological contributions in order to solve the problem of identifying waste dischargers at the North Sea. The first phase consists of performing k-fold cross-validation on our dataset in order to compare the initial performance of different classifiers, to test the generalizability of the models, and to select the optimal majority:minority training instance ratio that is going to be used to train the final models. The second phase consists of predicting each individual time instance of each trajectory using the best classifier and optimal ratio hyperparameter based on the results of the previous step. At this point, we have a prediction for each individual time instance of each trajectory without taking into consideration previous predictions up to that point. The third phase is concerned with experimenting by also taking into account different ranges of previous predictions and taking the majority vote in order to predict the next time instance. Finally, the last step is to measure the performance of the model at different time steps, the main goal being to reach acceptable performance as early as possible into the trajectories, which in itself, is an attempt to approach real time detection.

5.2.4.1. Cross-validation phase. Given that our dataset contains a very small amount of anomalous instances we are going to perform a manually configured stratified k-fold validation with a relatively small k value in order to keep the normal:anomalous test trajectory instances as close to the real dataset proportion as possible. The performance scores of a k-fold cross-validation on their own won't be too telling because we are restricted to using a small number of folds if we desire to approximate the real dataset proportion in our tests. That is why we are going to use this step to compare the initial performance of different classifiers, to test the generalizability of the models, and to select the optimal majority:minority training instance ratio that is going to be used to train the final models.

The classifiers that will be tested in this k-fold cross-validation phase will be the ensemble learning techniques discussed in Section 3.1, XGBoost and Random Forests. Support Vector Machines will also be tested in order to evaluate the performance of a linear separator and also compare its performance with the ensemble methods.

Lastly, an important hyperparameter that needs to be optimized in this phase is the majority:minority training instance ratio, a consequence of the extreme class imbalance. Training sets with different ratios will be tested (e.g. 2:1, 3:1, ..., 10:1) in order to find the optimal ratio that gives the highest performance.

5.2.4.2. Individual instance predictions. As discussed above, the previous step of the time ensemble classification involves selecting the classifier to be used for our main model and the optimal majority:minority training instance ratio for best performance. The selected classifier and the selected ratio will be used in order to train the supervised learning model and to predict every individual instance of every trajectory at each time point that they contain.

In order to further improve the performance, a majority voting learning approach will be implemented. Our version of the majority voting will include running N iterations of our selected classifier resulting in N predictions for each individual instance. The final predicted class will be selected using the simple majority voting approach.

5.2.4.3. Time series prediction smoothing and optimization. The next phase of the Time Ensemble Classifier pipeline involves an attempt to smooth the trajectory predictions (and potentially improve prediction accuracy) by optimizing the size of the previous predictions window that we take into account to predict the next time instance in the sequence. This approach is motivated from the field of time series forecasting, defined in [63] as the act of predicting the future by understanding the past. Forecasting systems usually are fed by some time series data points of the last several instances whereas the next time instance prediction is obtained at the system output, as shown in Equation 5.11:

$$Instance[t-n], Instance[t-n+1], ..., Instance[t-1], Instance[t] \rightarrow Instance[t+1] \tag{5.11}$$

This approach is usually used to forecast the value of the next time step based on the previous values, and not the prediction itself. However, in our modified approach, at this point we already have the individual instances assigned to the normal or anomalous class. Now, by taking into account different N-sized windows of previous predictions we perform another simple majority voting instance in order to potentially smooth and improve the initial predictions that we got from the previous pipeline steps. We aim to perform experiments with different N-sized windows, ranging from small N values to N containing all the previous prediction instances available for a particular trajectory.

5.2.4.4. Time Ensemble Classifier at different time points t. The final phase of the Time Ensemble Classifier pipeline is intended to give the final quantitative information of the performance of our models at different points in time. A quick reminder on what our main goal is in this supervised learning approach:

- Utilizing the AIS trajectory data, can we train models using previous sequential data points up to a specific point to detect our targeted behavior as early as possible after occurring?

At this point, we have predictions for each time instance of each trajectory, let those be the initial individual predictions or the predictions from the smoothing approach of the last step. In order to calculate the overall performance at a particular time point t (e.g. at 15 hours), from each trajectory we extract the prediction instance closest in time to that t and compute the performance based on the evaluation measures defined in Subsection 5.2.5, Macro Recall and Macro F1.

An important assumption that we made, discussed in Section 4.9, is that trips under 12 hours are very unlikely to be involved in waste discharge behavior. Based on that, it was decided that our baseline performance (naive classification) will be measured at a very early time t that is also under 12 hours. From that point on, the goal is to achieve an acceptable classification performance as early as possible meaning that the focus will be on the classification performance near the 12-hour mark.

Finally, it is also interesting to find out how good the performance of our model is after the trajectories have been completed, so computing the overall performance by extracting the final prediction of each trajectory. On top of that, it is interesting to see if we can find particular time intervals that are the most discriminatory when it comes to differentiating between normal and anomalous cases.

5.2.5. Evaluation measures

Different machine learning implementations require different evaluation measures based on what the specific task is and which area of correctness is more important. If we choose the wrong metric to evaluate our models, we are likely to choose a poor model, or in the worst case, be misled about the expected performance of our model. In our case, we are dealing with an extremely imbalanced dataset and applying an inappropriate evaluation metric for our model, such as accuracy, will lead to a very high accuracy score [61]. At first sight this seems to be a good result, but in reality that score is misleading because the high accuracy is only reflecting the underlying class distribution, which is imbalanced, and in turn this model doesn't provide any valuable information for us.

Some of the more suitable evaluation measures that can give more insight into our models as far as our case is concerned are: the confusion matrix, the precision, the recall and the F1 score.

<ins>5.2.5.1. Confusion matrix.</ins> A confusion matrix is a table that is often used to describe the performance of a classification model on a set of test data for which the true values are known. For a binary classification problem it is a table with 4 different combinations of predicted and actual values and an example can be seen in Figure 5.2. It can be just as easily implemented in a multi-class classification problem and it is extremely useful for computing the other important performance measures such as recall, precision, F1 score and accuracy.

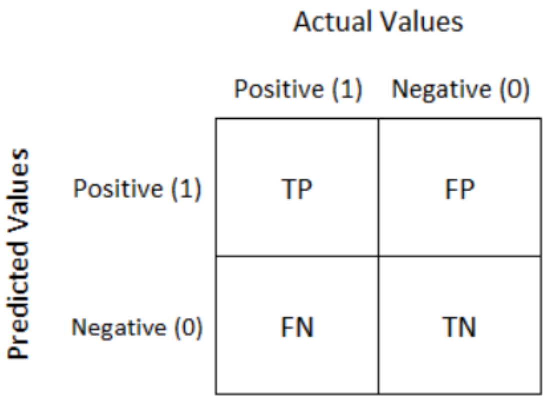

Figure 5.2: Confusion matrix (Adapted from [6])

- True positive (TN) — The sample's label is positive and it is classified as one.
- True negative (TN) — The sample's label is negative and it is classified as one.
- False positive (FP) — The sample's label is negative but it is classified as positive.
- False negative (FN) — The sample's label is positive but it is classified as negative.

5.2.5.2. Precision and Recall. Precision (Equation 5.12 for the binary case) summarizes the fraction of the examples assigned to the positive class that belong to the positive class. Recall (Equation 5.13 for the binary case) quantifies the number of positive class predictions made out of all positive examples in the dataset. Just like other evaluation measures, precision and recall are also not limited to binary classification problems.

$$Precision = \frac{TP}{TP + FP} \tag{5.12}$$

$$Recall = \frac{TP}{TP + FN} \tag{5.13}$$

When we are dealing with an imbalanced dataset in which we prioritize the correct classification of the minority class (anomalous class) we are interested in computing the so-called macro-average of the evaluation measures. Macro averaging calculates the corresponding metric for each of the existing classes, and then averages the results together. The equations for the macro averaged precision and recall can be seen in Equations 5.14 and 5.15.

$$Precision_{macro} = \frac{P_1 + P_2 + ... + P_c}{c} \text{ ; where } c \text{ is the number of classes.} \tag{5.14}$$

$$Recall_{macro} = \frac{R_1 + R_2 + ... + R_c}{c} \text{ ; where } c \text{ is the number of classes.} \tag{5.15}$$

5.2.5.3. F1 Score. The F1 score is a measure of a test's accuracy. The F1 score is defined as the weighted harmonic mean of the test's precision and recall. If we put it in another way, what the F1 score does is it conveys the balance between the precision and the recall. This score is calculated according to the Equation 5.16:

$$F1 = \left(\frac{2}{\text{recall}^{-1} + \text{precision}^{-1}}\right) = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{5.16}$$

Again, in order to account for the class imbalance we need to use the macro averaged F1 score, as it gives equal importance to each class. Macro F1 is the harmonic mean between precision and recall where the average is calculated per class and then averaged across all classes. The equation for the Macro F1 can be seen in Equation 5.17:

$$Macro_{F1} = \frac{1}{c}\sum_{i=1}^{c} \frac{2 \cdot Precision_i \cdot Recall_i}{Precision_i + Recall_i} \tag{5.17}$$

5.2.6. Statistical significance test

5.2.6.1. Test concerning two proportions. There are several different statistical hypothesis testing frameworks that are being used in practice to compare the performance of classification models, including conventional methods such as the difference of two proportions. In our case, the proportions are the estimated scores (Macro Recall and Macro F1) from the test set, for which we can construct 95% confidence intervals ($\alpha = 0.05$) based on the concept of the Normal Approximation to the Binomial distribution. The test concerning two proportions is defined as follows [64]: Let there be two treatments with population proportions p_1 and p_2 and sample sizes of n_1 and n_2 respectively. Samples associated with a certain event are x_1 and x_2 with the respective proportions.

$$\hat{p_1} = \frac{x_1}{n_1} \quad \text{and} \quad \hat{p_2} = \frac{x_2}{n_2}$$

$$H_0 : p_1 = p_2 \tag{5.18}$$

$$\textbf{TS : } Z = \frac{\hat{p_1} - \hat{p_2}}{\sqrt{\hat{p}(1-\hat{p})(\frac{1}{n_1} + \frac{1}{n_2})}} \qquad \hat{p} = \frac{x_1 + x_2}{n_1 + n_2}$$

Here, $\hat{p}$ is the pooled estimate of the proportions p_1 and p_2. The null hypothesis H_0 states that the proportions are the same. The alternative hypothesis, H_1, states that the proportions are different. The condition that needs to be satisfied in order for H_0 to be rejected is $|Z| \geq z_{\alpha/2}$, where z is a value from the standard normal table and α is the specified significance level.

5.3. Unsupervised learning

The unsupervised learning phase of our research is going to be relatively smaller and shorter compared to the time series supervised learning phase. This phase intends to find out if we can discover new cases in our dataset that deviate from the normal behavior, leading to potential waste discharge cases undetected until now by authorities.

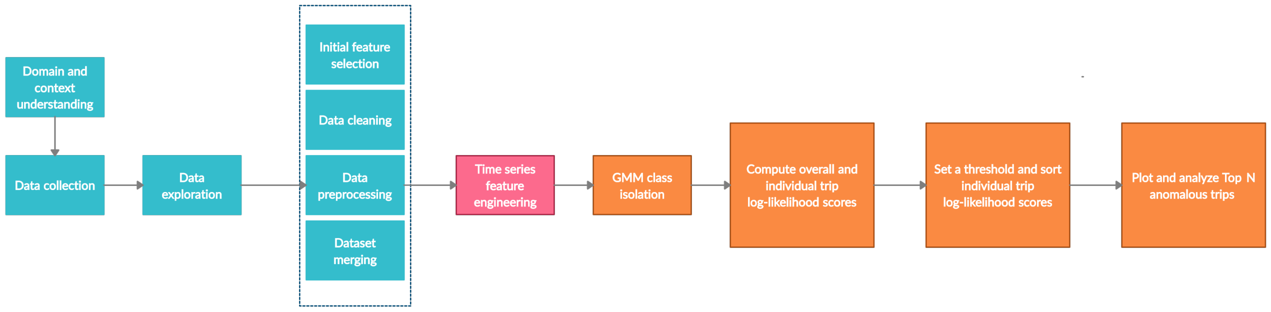
Domain and context understanding
Data collection
Data exploration
Initial feature selection
Data cleaning
Data preprocessing
Dataset merging
Time series feature engineering
GMM class isolation
Compute overall and individual trip log-likelihood scores
Set a threshold and sort individual trip log-likelihood scores
Plot and analyze Top N anomalous trips

Figure 5.3: Unsupervised learning process pipeline

The unsupervised learning process pipeline, seen in Figure 5.3, will be used to answer the following research question (and the sub-research questions that it contains):

- Based on the log-likelihood scores of the individual trips, are the tanker-specific fitted models able to detect new anomalous cases residing in the dataset?

5.3.1. Gaussian Mixture Model implementation

As seen in the research question above, in our unsupervised learning approach we intend to create tanker-specific fitted models and we aim to do that for each class separately (normal and anomalous). Gaussian Mixture Models, background information given in Subsection 3.3.1.1, are one of the most used density estimation or clustering techniques when it comes to modeling data by isolating individual class distributions.

For each fitted Gaussian Mixture Model of each class, the number of components will be selected based on the BIC score. In statistics, the Bayesian information criterion (BIC) is a criterion for model selection among a finite set of models where the model with the lowest BIC is preferred [65]. When fitting models, it is possible to increase the likelihood by adding parameters, but doing so may result in overfitting. BIC attempts to resolve this problem by introducing a penalty term for the number of parameters in the model [65].

In order to maximize the possibility of anomaly detection using this approach, we are going to use the knowledge acquired from the time series supervised learning phase. This means that as input to the models we will only use the data of the most discriminatory time interval instead of the complete trajectories.

After fitting the class-specific Gaussian Mixture Models and obtaining the overall log-likelihood scores, we are especially interested in the log-likelihood scores of individual trajectories in order to find out how good do they fit to each class. In order to have fair comparisons between individual trajectories, we are going to subsample each of them in a fixed-size representation of the same length L. All the trajectories will be tested in both class-specific GMM's and the log-likelihood output scores of both models will be analyzed, separately. Since each trajectory will be represented by L samples, for each individual sample we will compute the weighted log probabilities. The final log-likelihood score for each trajectory will be the summed score of individual samples. Finally, the summed trajectory scores will be sorted and compared to each other. Eventually, as a part of this research or as an extension of it, The Top N (or Top N percent) trajectories with the higher likelihood of being anomalous according to the models will be selected, plotted and further inspected, also with the help of the domain inspectors.

6. EXPERIMENTS AND RESULTS

In this chapter we are going to discuss the experimental configurations of both the supervised and unsupervised approaches where sections will be dedicated to each of them separately. In addition, in this chapter we will present the final results of this research.

6.1. Time series supervised learning experimental setup - Part I

As discussed in Chapter 5, the time series supervised learning approach is developed with two goals in mind: 1. Testing the predicting power of individual time resolution feature sets, and 2. Achieving acceptable performance in differentiating between normal and anomalous instances as early as possible with respect to time t.

The initial experimental tests are designed to give us information about the predicting power of individual time resolution feature sets, the difference in performance between using only the low-level descriptors and using all the engineered features, and to gain initial insight on how good our chosen classifiers perform and the differences between them. The experimental setup that we have devised in order to answer the research question contains the following configurations:

(i) Feature set/subset selection - In order to test the predicting power of the low-level descriptors and the different time resolution engineered features, five different sets of features were selected to be evaluated:
   - Low-level descriptors only: *t_latitude*, *t_longitude*, *t_speed*, *t_orientation*.
   - Full time resolution engineered features (40 features)
   - Last 10 minutes time resolution features (40 features)
   - Last 60 minutes time resolution features (40 features)
   - All features (124 features)

(ii) Train/test set selection - Only a subset of the data was used to acquire the initial results. In order to combat the class imbalance problem we experimented with different majority:minority class ratios, as it will be explained in the next bullet point. The testing set always contained the same 45 normal instances and the same 5 anomalous instances for fair comparisons.

(iii) Majority:minority class ratio configuration - One of the class imbalance overcoming techniques we discussed in Subsection 5.2.2 is the adjustment of the ratio between the majority and minority class training instances. Different models were trained with the following majority:minority ratios:
   - $k : 9$ where $k = 18, 27, 36, ..., 81$; $step\_size = 9$

(iv) Number of iterations per ratio configuration - In order to test the robustness of the models using different normal training instances as input we ran each model several times to get the average scores and their standard deviation. Another reason to perform more iterations of each classifier is to get an ensemble of predictions and then classify each instance according to the majority voting methodology discussed in 5.2.4.2.
   - Number of iterations for each ratio configuration = 50

(v) Dimensionality reduction configuration - PCA will be performed in such a way that it will automatically select the number of components it needs to explain at least 90% of the variance.

(vi) Classifiers evaluated - XGBoost and Random Forests.

(vii) Evaluation measures - As discussed in Section 5.2.5, we are interested in the performance scores that take the class imbalance problem into account. To quantify the performance of our models the following evaluation measures were used:
   - Macro Recall average (standard deviation)
   - Macro F1 average (standard deviation)

6.2. Evaluation of individual time resolutions

The initial time series supervised learning experiments were performed in order to find out if the engineered time resolution features discussed in Subsection 5.2.1 have additional predicting power compared to using only the original low-level descriptors (latitude, longitude, speed and orientation). Looking at the performance results of the individual time resolution feature sets, shown in Table 6.1, we see that the best performing feature set is the one labeled as "Last 60 minutes resolution", the time resolution that considers the window containing the data points of the last 60 minutes of the trajectory. The "Last 60 minutes resolution" achieves a Macro Recall score of 71%, that is 2% higher than the original low-level descriptors and 6% higher than the "Full time resolution" that considers all the preceding data points of a trajectory.

Individual time resolution performance			**XGBoost**			**Random Forest**	
Features	**Metric**		**k = 18**	**k = 81**		**k = 18**	**k = 81**
Low-level descriptors	**Macro Recall Avg (Std)**		0.69 (0.04)	0.54 (0.02)		0.67 (0.03)	0.60 (0.01)
	Macro F1 Avg (Std)		0.58 (0.03)	0.55 (0.03)		0.58 (0.03)	0.62 (0.02)
Full time resolution	**Macro Recall Avg (Std)**		0.65 (0.05)	0.52 (0.02)		0.63 (0.05)	0.51 (0.01)
	Macro F1 Avg (Std)		0.55 (0.03)	0.52 (0.03)		0.55 (0.03)	0.50 (0.02)
Last 10 minutes resolution	**Macro Recall Avg (Std)**		0.70 (0.03)	0.53 (0.02)		0.69 (0.03)	0.61 (0.01)
	Macro F1 Avg (Std)		0.58 (0.02)	0.54 (0.03)		0.59 (0.02)	0.64 (0.01)
Last 60 minutes resolution	**Macro Recall Avg (Std)**		0.71 (0.04)	0.54 (0.02)		0.70 (0.03)	0.60 (0.01)
	Macro F1 Avg (Std)		0.59 (0.03)	0.55 (0.03)		0.59 (0.02)	0.62 (0.02)

Table 6.1: Individual time resolution performance

In these preliminary results, one of the most important aspects that we can focus on is the difference in performance between using only the low-level descriptors, and using all the time resolution engineered features as input to the machine learning classifiers. The results shown in Table 6.2 tell us that when using all the features as input, XGBoost achieves a Macro Recall of 73%, that is 4% higher than when using only the low-level descriptors. This indicates that our approach of using statistical functionals to engineer time resolution based features does increase the predictive power of the machine learning models.

Performance using all features			**XGBoost**			**Random Forest**	
Features	**Metric**		**k = 18**	**k = 81**		**k = 18**	**k = 81**
All features	**Macro Recall Avg (Std)**		0.73 (0.04)	0.57 (0.03)		0.71 (0.04)	0.57 (0.02)
	Macro F1 Avg (Std)		0.58 (0.03)	0.59 (0.04)		0.59 (0.03)	0.59 (0.02)

Table 6.2: Performance using all features

An important configuration detail that we have to mention is that in order to obtain these preliminary results only a subset of the data was used, and as a consequence, the normal:anomalous instance ratio in the test set is not indicative of the real dataset proportion. Analyzing these results gave us the initial impressions on how good each individual feature set performs, if our approach of performing statistical functionals increases model performance, and on which feature sets (if not all) we should focus on when developing our final models.

6.3. Time series supervised learning experimental setup - Part II

The second and the main part of the time series supervised learning approach is concerned with clarifying one of the most important aspects of our research, that being: How early (and with acceptable performance) in the trajectories can we correctly differentiate between normal and anomalous cases?

The experimental setup that we have devised in order to help us answer that question contains the following configurations:

(i) Feature set selection - All features (124 features)

(ii) Majority:minority class ratio configurations - Same as in Part I (Section 6.1)

(iii) Cross-validation - A 3-fold cross-validation will be performed.
- A total of 670 trips in our dataset. 656 normal and 14 anomalous.
- In order to approximate the real dataset normal:anomalous instance ratio, in each test fold we will have around 218 normal instances and around 5 anomalous instances.
- The cross validation performance results will be used to select the optimal majority:minority training instance ratio that is going to be used to train the final models.

(iv) Final model & individual instance predictions
- The final model will be built with the ratio configuration seen as optimal from the previous step.

- In order to get an ensemble of 10 prediction for each time instance the classifier will be iterated 10 times. The number of iterations was reduced from 50 to 10 because of time and computational limitations.

(v) Dimensionality reduction configuration - Same as in Part I (Section 6.1)

(vi) Time series prediction smoothing and optimization - As discussed in 5.2.4.3, we are going to perform another optimizing step with the hopes of smoothing the trajectory predictions even further. Different N-sized windows will be experimented with:

- N = 3, 6, 9, 12, all previous instances.

(vii) Additional “expert system rule” - Trips under 12 hours will always be classified as normal.

(viii) Classifiers evaluated - XGBoost, Random Forests, Support Vector Machines.

(ix) Evaluation measures - The performance will be evaluated at different time points and for different time intervals.

- Overall Macro Recall & F1.
- Macro Recall & F1 at different time points.
- Macro Recall & F1 for different time intervals.

(x) Time Ensemble Classifier evaluation - In the final phase of the supervised learning approach we are going to evaluate the performance of our model using the evaluation measures described above. We are going to evaluate the following:

- Overall performance: [0h,48h].
- Performance at time points: t = 10h (Baseline), 12h, 14h, 16h, 18h, 20h, 25h, 30h, 35h, 40h, 48h.
- Performance for time intervals: [10h,16h], [15h,20h], [10h,20h], [15h,35h].

6.4. Evaluation of Time Ensemble Classifier

In the next paragraphs, we summarize the performance of our final models and discuss the performance achieved at particular time intervals, or at specific points in time. First, the cross-validation results shown in Table 6.3, indicate that, as far as performance is concerned, the choice of classifier (XGBoost, Random Forest or SVM), is not a very important aspect. XGBoost achieves a Macro Recall of 67%, Random Forest a Macro Recall of 68%, and SVM a Macro Recall of 66%. The cross-validation performance of the classifiers of our choice is almost identical, however, the runtime of the XGBoost ensemble classifiers was shorter by a considerable margin. Given that there isn't any dramatic difference in performance between the classifiers, XGBoost will be considered as the main classifier for the next part, the main reason being that it reaches almost identical performance as the others, but does so in a much shorter runtime. For this research, experiments were also performed using Random Forests and SVM, and the results will mostly be used for comparison purposes. The cross-validation phase was also used to experiment with different majority:minority instance ratios to train our models with, in order to maximize performance. The results showed that the optimal majority:minority ratio that maximized performance was at $k = 36$ (Ratio 4:1), meaning that 36 normal trajectories and 9 anomalous trajectories were used in the training sets of our final models.

3-Fold Cross-Validation			**XGBoost**		**Random Forest**		**SVM**
All features	**Macro Recall Avg**		0.67 (0.05)		0.68 (0.06)		0.66 (0.04)
	Macro F1 Avg		0.49 (0.02)		0.5 (0.03)		0.47 (0.02)

Table 6.3: 3-Fold Cross-Validation results (k = 36)

Looking at Figures 6.1 and 6.2, we depict the performance of our final models at different time points t, ranging from 10 hours (baseline) to 48 hours (end of trajectories). At $t = 10$ hours (baseline), the model achieves a Macro Recall of 64% and a Macro F1 of 53%. At $t = 18$ hours, the model achieves a Macro Recall of 80% and a Macro F1 of 60%, where the best results are achieved using $N = 12$ (the number of previous predictions taken into consideration to smooth the trajectory predictions). The very best results are achieved at $t = 40$ hours, where the model achieves a Macro Recall of 96% and a Macro F1 of 67%, however, we are not very concerned what happens this late into the trajectories because our goal is to reach acceptable performance as early as possible, and at $t = 40$ hours, it might be too late to potentially stop a waste discharge instance (in real time applications). That is why the main focus is analyzing the performance of the model between the time interval [10h,20h], and as discussed, the highest performance is achieved at $t = 18$ hours. Another important aspect that needs to be mentioned is that the Macro Recall scores might be boosted a bit from the fact that we always "expertly" classify trips under 12 hours as normal. This assumption, discussed in Section 4.9, is based on statistical analysis and domain expert opinion.

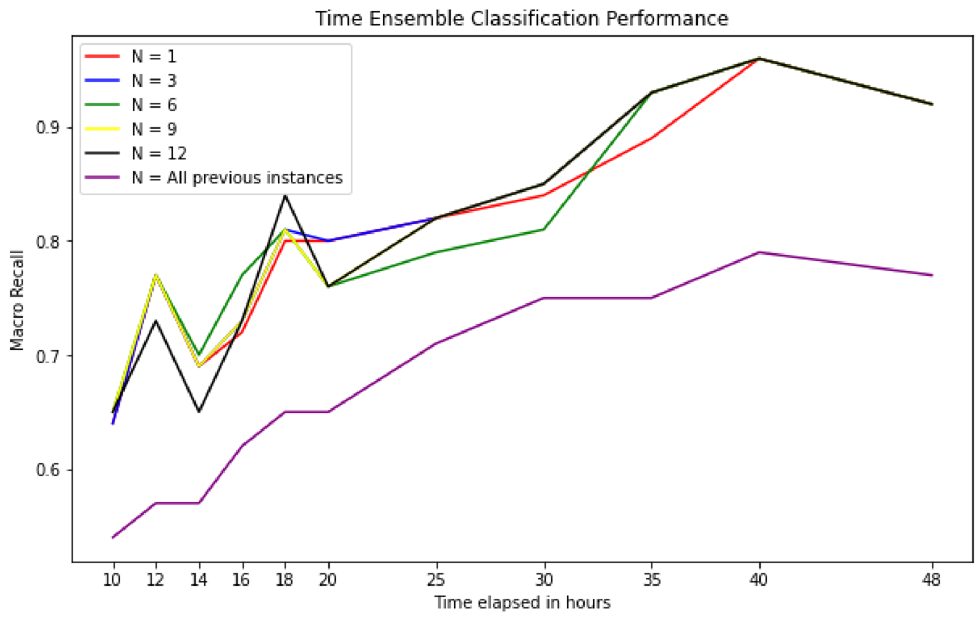

Figure 6.1: Time Ensemble Classification Performance - Recall (XGBoost)

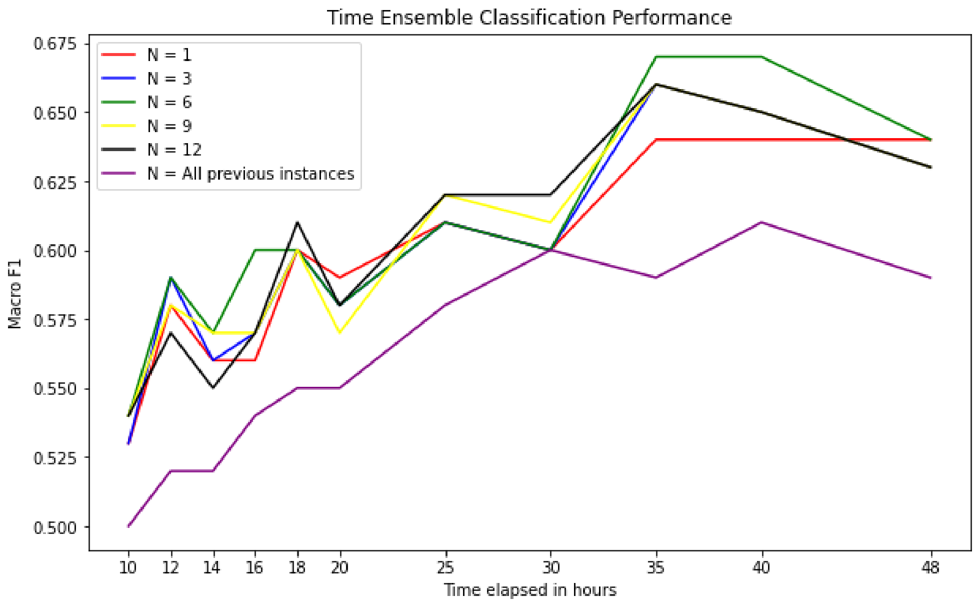

Figure 6.2: Time Ensemble Classification Performance - F1 Score (XGBoost)

Analyzing the results of Tables 6.4 and 6.5, we try to find out if there are particular time intervals where it is easier to discriminate between normal and anomalous behavior. In these performance analysis scenarios we consider the predictions of all instances at each time point and for each trajectory. In Table 6.4, we see that XGBoost achieves an Overall Macro Recall of 82% and an Overall Macro F1 of 62%. The overall performance is calculated by taking into account all predictions, at all times, and for each trajectory, so the full time interval [0h,48h]. The different time interval scenarios that we were guided to analyze by our findings are shown in Table 6.5. Looking at the results, we see that all three different classifiers (XGBoost, Random Forest, SVM) perform differently at different time intervals. This makes the time interval scenarios difficult to analyze and discuss, and maybe inconclusive. That being said, we can see that when the time interval [15h,35h] is considered, XGBoost achieves a Macro Recall of 84% and a Macro F1 of 63%, which is slightly better than the overall performance.

Overall Time Ensemble Performance (N=1)			XGBoost		Random Forest		SVM
All features	**Macro Recall Avg**		0.82		0.75		0.82
	Macro F1 Avg		0.62		0.59		0.61

Table 6.4: Overall Time Ensemble Performance (k = 36)

Time interval performance			XGBoost		Random Forest		SVM
10h-16h	**Macro Recall Avg**		0.78		0.7		0.81
	Macro F1 Avg		0.58		0.56		0.58
15h-20h	**Macro Recall Avg**		0.78		0.82		0.82
	Macro F1 Avg		0.59		0.6		0.6
10h-20h	**Macro Recall Avg**		0.78		0.82		0.81
	Macro F1 Avg		0.58		0.6		0.59
15h-35h	**Macro Recall Avg**		0.84		0.78		0.79
	Macro F1 Avg		0.63		0.62		0.6

Table 6.5: Time interval performance (k = 36)

6.5. Model statistical significance test (Best performance vs Baseline)

Test concerning two proportions (two-tailed):

- P_1: Best performance: $t = 18$ hours
- P_2: Baseline: $t = 10$ hours
- H_0: $P_1 = P_2$
- H_1: $P_1 \neq P_2$
- $\alpha = 0.05$
- $Z = 6.456$. Z is not in the 95% critical value accepted range: $[-1.9600 : 1.9600]$.
- $p < 0.00001$
- Since $p < \alpha$, H_0 is rejected. The difference between the proportion of the Best performance and Baseline populations is big enough to be statistically significant.

6.6. Unsupervised learning experimental setup

As discussed in Chapter 5, the unsupervised learning approach is developed in order to detect new potentially anomalous trajectories residing in our dataset. In order to do so, the following experimental setup and configurations have been devised:

(i) Gaussian Mixture Models - Two separate Gaussian Mixture Models will be fitted. One using only the data of the normal trajectories, one using only the data of the anomalous trajectories.

(ii) Number of components - The number of components (clusters) will be selected based on the BIC score discussed in Subsection 5.3.1. The lower the BIC score the more appropriate the number of components for the models.

- 14 mixture components for the GMM fitted with normal data.
- 4 mixture components for the GMM fitted with anomalous data.

(iii) Time interval data used for input - The results shown in Subsection 6.4 indicate that one of the most discriminative time intervals between normal and anomalous trajectories is between 10 and 20 hours. In order to fit the GMM's, only the data between this time interval will be used as input.

(iv) Fixed-size representation of trajectories - Each trajectory will be represented by exactly 100 sampled instances from the previously mentioned time interval. The sample size is fixed in order for the overall trajectory log-likelihood scores to be fairly compared to each other.

(v) Degree of anomaly measure - Each trajectory will be represented by 100 sampled instances, as stated above. First, each of the samples will have their log-probability computed individually. Then, for each trajectory the log-likelihood scores will be summed together to obtain the overall trajectory log-likelihood. The obtained scores will be sorted in such way that the potentially anomalous trajectories will be on top. Whether the sorting is in ascending or descending order, it depends on which fitted GMM we take into account.

(vi) Top N anomalous cases - The Top 20 anomalous cases according to the models will be plotted (only the Top 6 will be shown in the results subsection), visualized on maps and will be further inspected. Eventually, as part of this research, the visualized trajectories will also be shown to the domain inspectors, mirroring the initial process in which our already known anomalous cases were identified.

6.7. Evaluation of the unsupervised learning approach

In this section, we discuss the output of our Gaussian Mixture Model unsupervised learning approach. There are three things to discuss: the "looping" behavior that it detects, the detection of one trajectory instance leaving Rotterdam and returning to Rotterdam, and the potential of this approach if more anomalous data were available.

Analyzing the Figures 6.3 and 6.4, where in each figure the Top 6 ranked anomalous trajectories are visualized, the first thing to be noticed is that the Gaussian Mixture Model approach is able to detect the "looping" behavior, albeit, there is some confusion involved. We can see that most of the visualized trajectories have some kind of "loop", or let's say a U-turn involved, however, it can also be seen that it doesn't involve the same port. Because of the geographical position of the ports of Rotterdam, Amsterdam and Antwerp, in order to go from one point to another between these three ports, the trajectory is obviously expected to be in the form of a U-turn. In addition to that, because of the closeness of these ports, the dataset is expected to have a huge amount of data involving trips from one of the aforementioned ports to another. This results in the Gaussian Mixture Models confusing our targeted "looping" behavior with the previously discussed U-turn, and in doing so, models them as if they exhibit the same ship movements or behavior.

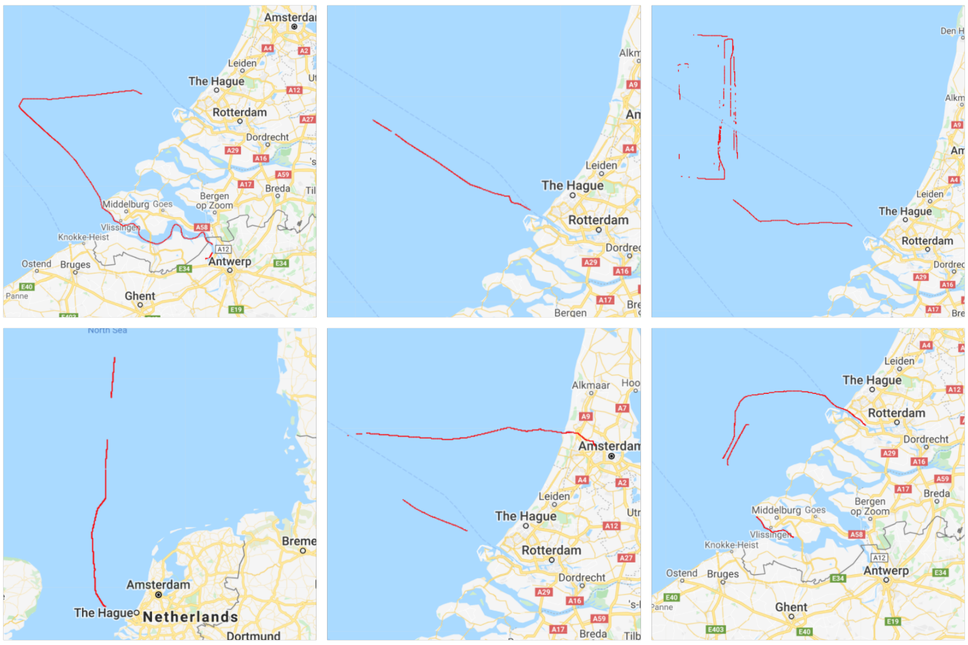

Figure 6.3: Top 6 anomalous trajectories according to the GMM fitted with normal data (Ground truth = Normal trajectory, for all six instances)

Trip ID	**Port**	**Destination**	**Cargo type**
253	Rotterdam	Antwerpen	NAN
80	Rotterdam	Grangemouth	NAN
336	Rotterdam	NL RTD 4 ORDERS	Recognizable hazard
609	Rotterdam	TJELDBERGODDEN^2CNOR	Hazard
473	Rotterdam	Amsterdam	Hazard
636	Rotterdam	FLUSHING	Unspecified

Table 6.6: Top 6 anomalous trips according to the GMM fitted with normal data (Destination name is the actual reported and unaltered text)

Looking at the visualization of trajectory number two in Figure 6.4, we can see the detection of one new and previously unseen instance of a ship departing Rotterdam and returning to Rotterdam without visiting another port. This instance was missed during the initial process of labeling anomalous instances because there exist two port destination features in the datasets: one belonging to the AIS dataset and another belonging to the Portcall dataset. This is one of the cases that highlights that the approach of merging the AIS and Portcall datasets will lead to mismatches and irregularities, also discussed in Subsection 4.7.2, because both datasets are collected with different usages in mind. That being said, in this case, with the existence of two different features describing the same thing in different datasets, we ended up with the detection of another potential anomalous trajectory.

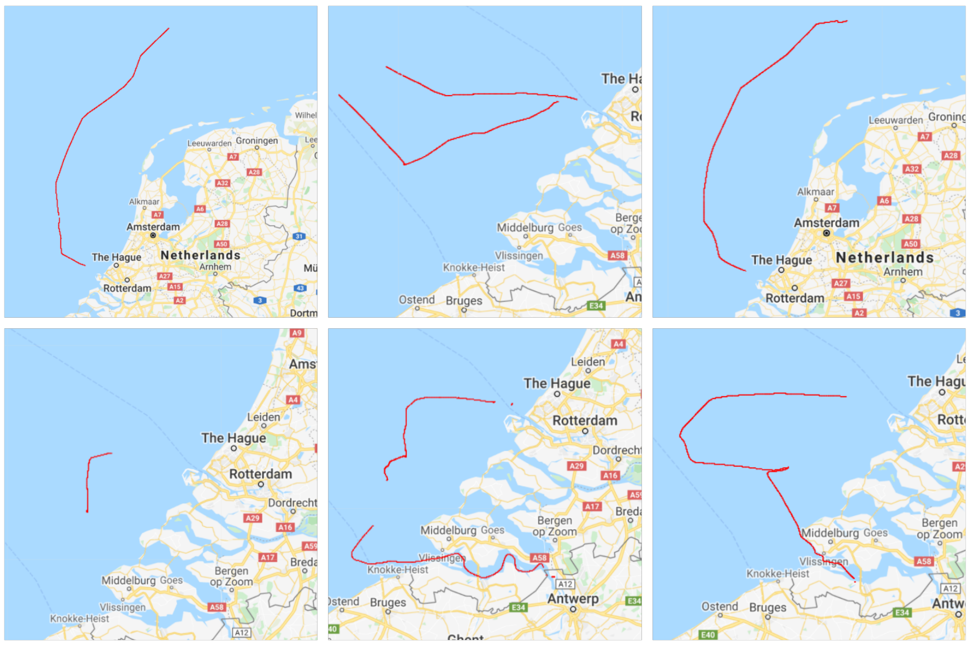

Figure 6.4: Top 6 anomalous trajectories according to the GMM fitted with anomalous data (Ground truth = Normal trajectory, for all six instances)

Trip ID	**Port**	**Destination**	**Cargo type**
222	Rotterdam	Gethenborg	NAN
297	Rotterdam	Rotterdam	NAN
423	Rotterdam	Hamburg	NAN
164	Rotterdam	STEENBANK ANCHORAGE	Unspecified
439	Rotterdam	Antwerpen	Hazard
36	Rotterdam	NL RTM>NL TNZ	NAN

Table 6.7: Top 6 anomalous trips according to the GMM fitted with anomalous data (Destination name is the actual reported and unaltered text)

It is also important to note that visualizing and analyzing the Top 6 anomalous instances according to the unsupervised learning models, we also see that they contain some trajectories that are perfectly normal, but are ranked high based on the anomaly score. This highlights the need for the acquisition of more anomalous instances that would be used to fit these kinds of unsupervised models. At the same time, the detection of the U-turn and the Rotterdam-Rotterdam trajectory gives us positive indicators that this approach can be additionally refined to be more efficient when it comes to anomaly detection.

7. CONCLUSIONS

In the final chapter of the thesis, we answer our research questions defined in Section 1.5 , reiterate the key points of this research, explain the relevance and significance of the work, and give recommendations for future work on this topic. Each sub-research problem will be answered and discussed individually, and as a collective, they contribute to answering the main research question.

The first research sub-problem was defined as follows:

- What is the effect of summarizing the low-level descriptors that come with the AIS data (e.g. latitude, longitude, speed, orientation), over past records?

The preliminary results obtained after performing the initial experiments, shown in Tables 6.1 and 6.2, indicate that our approach of applying statistical functionals to summarize the low-level descriptors increases the performance by 4%, achieving a Macro Recall of 73% when all engineered features are used as input. When it comes to comparing the individual time resolutions, the time resolution labeled as “Last 60 minutes resolution” performs the best, achieving a Macro Recall of 71%, that is a 2% increase compared to using only the low-level descriptors as input. Looking at the results discussed above, and also in Section 6.2, we can say that the approach of applying statistical functionals in order to engineer new features that contain historical information about the trajectories, does, in fact, boost the performance.

The second research sub-problem was defined as follows:

- Utilizing the AIS trajectory data, can we train models using previous sequential data points up to a specific point to detect our targeted behavior before occurring, or as early as possible after occurring?

One of the main goals defined for this research was to be able to distinguish between normal and anomalous instances as early as possible in the trajectories. At the baseline time point, defined at $t = 10$ hours, the model achieved a Macro Recall of 64% and a Macro F1 of 53%. As time progresses, the model's ability to discriminate between normal and anomalous instances increases. At $t = 18$ hours, the model reaches the best performance, time optimization also taken into account, achieving a Macro Recall of 80% and a Macro F1 of 60%. A statistical significance test was also performed (test concerning two proportions) in order to find out if the difference in performance between the model at $t = 10$ (Baseline) and at $t = 18$ (Best performance) is significant. The results showed that the difference between the proportion of the best performing model and the baseline model populations is big enough to be considered statistically significant. This means that we were able to achieve a significant improvement in performance, and did so in a relatively short time.

The third research sub-problem was defined as follows:

- Based on the log-likelihood scores of the individual trips, are the tanker-specific fitted models able to detect new anomalous cases residing in the dataset?

The unsupervised Gaussian Mixture Model (GMM) approach was devised with the goal of finding previously undetected anomalous cases residing in the AIS dataset of April-May 2017. The results showed that this approach has a lot of potential, however, a dataset with a higher amount of anomalous instances is necessary, as expected. The fitted GMMs are able to detect variations of the "looping" behavior, including the "U-turn" trajectory form, where ships travel between Rotterdam, Amsterdam and Antwerp. Because of the small amount of anomalous data instances, the model is not able to distinguish very well between our targeted "looping" trajectories and the "U-turn" trajectories, resulting in ranking them as anomalous very frequently. That being said, even with the small amount of anomalous data it already managed to detect one case of our targeted behavior. This means that with further tweaking, and fitted with a higher amount of quality data, this approach has the potential to be very useful.

The fourth research sub-problem was defined as follows:

- Is the amount and the quality of the data available to us sufficient for waste discharge anomaly detection?

This research question does not have a black or white resolution, understandably, because the more data that we have in our disposition, the greater our possibilities for successful implementations. However, there are certain aspects that we can discuss after getting more acquainted with these datasets during this research, concerning both the amount and the quality. First, it is clear that there is a lack of data labeled as anomalous, so, the data indicating our targeted behavior. The lack of anomalous data was an important limitation in our research because it restricted our methodological choices and approaches, meaning that even the techniques that we did manage to implement included substantial configurations and tweaks. Second, in terms of quality, the general limitations concerning AIS data, discussed in Section 1.3, also applied to our dataset. This included missing data, inconsistencies, mismatches, and very likely intentional reporting of incorrect information by ship operators.

And finally, the main research question was defined as follows:

Given the availability of the AIS dataset and the potential of supervised and unsupervised techniques to solve complex problems: Is it possible to develop machine learning models that detect ship waste and residue discharge at the North Sea effectively (with performance significantly higher than chance level) and efficiently (with minimal cost and preferably real-time)?

The discussions of the individual research sub-problems in the preceding paragraphs have already contributed to answering our main research question. To summarize it, we can conclude that, yes, it is possible to develop models that discriminate between normal and anomalous instances effectively and efficiently. In essence, our model achieves a Macro Recall of 80%, and does so in 18 hours time elapsed. That being said, the specific scenario of detecting ship waste discharge at the North Sea is a more complicated matter, even though there are many positive indicators. Because of the frequently mentioned limitation that is the lack of anomalous data, we had to combine the trajectory instances of “Class 1” (anomalous, but not expected to have exhibited waste discharge behavior), and “Class 2” (anomalous, typical waste discharge behavior), into one class. This means that our model learns the anomalous behavior as a combination of the previously mentioned “Class 1” and “Class 2”. That is not the scenario that we would aim for in perfect circumstances, however, it was a necessary step to carry out in order to obtain performance results that would have a meaningful interpretation. Finally, it is important to mention that at $t = 18$ hours, our model correctly classifies 9 out of 14 anomalous instances, a good indicator that it would be able to achieve similar, or very likely better performance, if we had a higher amount of anomalous “Class 2” instances to train and test our models.

Last but not the least, we will discuss what the results of our thesis signify, and give a final suggestion on where to go from here. First, we found out that it actually is possible to discriminate between normal and anomalous behavior relatively early in the trajectories. Second, based on the global and local time analysis plots, (Figures 6.1, 6.2, A.1, A.2, A.3, A.4, A.5), we found out that the time interval [10h,20h] is very informative when it comes to distinguishing between normal and anomalous instances. These findings align with our initial data exploratory analysis that indicated that 13 out of 14 anomalous trips have a trip length of over 12 hours. This also aligns with domain expert opinions that suggested that waste discharge behavior is likely to be exhibited after 12 hours or longer after departing the port of Rotterdam. And lastly, to conclude everything on a positive note, this newly acquired knowledge has given us positive indicators that this research can be used as a stepping stone to (near) real time detection of waste and residue discharge at the North Sea, and that we can further refine both the time series supervised approach and the unsupervised learning approach to achieve even higher performance.

REFERENCES

1. Murphy, D., *Using Random Forest Machine Learning Methods to Identify Spatiotemporal Patterns of Cheatgrass Invasion through Landsat Land Cover Classification in the Great Basin from 1984 - 2011*, Ph.D. Thesis, 04 2019.

2. Zhang, Z., G. Mayer, Y. Dauvilliers, G. Plazzi, F. Pizza, R. Fronczek, J. Santamaria, M. Partinen, S. Overeem, M. Peraita-Adrados, A. Silva, K. Sonka, R. Río, R. Heinzer, A. Wierzbicka, P. Young, B. Högl, C. Bassetti, M. Manconi and R. Khatami, "Exploring the clinical features of narcolepsy type 1 versus narcolepsy type 2 from European Narcolepsy Network database with machine learning", *Scientific Reports*, Vol. 8, 12 2018.

3. *Clustering in Machine Learning*, `https://www.geeksforgeeks.org/clustering-in-machine-learning/`, Accessed: 21.08.2020.

4. *Cross-validation (statistics)*, `https://en.wikipedia.org/wiki/Bayesian_information_criterion`, Accessed: 15.08.2020.

5. *EnsembleVoteClassifier*, `http://rasbt.github.io/mlxtend/user_guide/classifier/EnsembleVoteClassifier/`, Accessed: 16.08.2020.

6. *Understanding Confusion Matrix*, `https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62`, Accessed: 20.04.2020.

7. *North Sea*, `https://en.wikipedia.org/wiki/North_Sea`, Accessed: 04.03.2020.

8. Barry, M., I. Elema and P. van der Molen, "Governing the North Sea in the Netherlands", , 2006, in: Administering marine spaces : international issues.

Frederiksberg, International Federation of Surveyors (FIG), 2006. 176 p. ISBN: 87-90907-55-8. pp. 64-83.

9. *Port of Amsterdam*, `https://en.wikipedia.org/wiki/Port_of_Amsterdam`, Accessed: 04.03.2020.

10. *MARPOL*, `http://www.imo.org/en/About/Pages/Default.aspx`, Accessed: 02.06.2020.

11. *MARPOL Annex II*, `http://www.imo.org/en/OurWork/Environment/PollutionPrevention/ChemicalPollution/Pages/Default.aspx`, Accessed: 02.06.2020.

12. *Zeezwaaien wikipedia*, `https://nl.wikipedia.org/wiki/Zeezwaaien`, Accessed: 06.03.2020.

13. *Operational discharges of residues of NLS*, `http://www.marpoltraining.com/MMSKOREAN/MARPOL/Annex_II/r13.htm`, Accessed: 02.06.2020.

14. *What is the AIS?*, `https://help.marinetraffic.com/hc/en-us/articles/204581828-What-is-the-Automatic-Identification-System-AI`, Accessed: 10.03.2020.

15. Fournier, M., C. Hilliard, S. Rezaee and R. Pelot, "Past, present, and future of the satellite-based automatic identification system: areas of applications (2004–2016)", *WMU Journal of Maritime Affairs*, Vol. 17, 09 2018.

16. *How AIS works?*, `https://www.navcen.uscg.gov/?pageName=AISworks`, Accessed: 11.03.2020.

17. *Automatic Identification System*, `https://en.wikipedia.org/wiki/Automatic_identification_system/`,

Accessed: 12.03.2020.

18. *All about AIS*, `http://www.allaboutais.com/index.php/en/faqs/`, Accessed: 12.03.2020.

19. Katsilieris, F., P. Braca and S. Coraluppi, “Detection of Malicious AIS Position Spoofing by Exploiting Radar Information”, , 07 2013.

20. Claramunt, C., C. Ray, E. Camossi, A.-L. Jousselme, M. Hadzagic, G. Andrienko, N. Andrienko, Y. Theodoridis, G. Vouros and L. Salmon, “Maritime data integration and analysis: recent progress and research challenges”, , 03 2017.

21. Alessandrini, A., M. Alvarez, H. Greidanus, V. Gammieri, V. Fernandez Arguedas, F. Mazzarella, C. Santamaria, M. Stasolla, D. Tarchi and M. Vespe, “Anomaly detection and knowledge discovery using Vessel Tracking Data”, , 07 2016.

22. Grubbs, F. E., “Procedures for Detecting Outlying Observations in Samples”, , 1969.

23. Goldstein, M. and S. Uchida, “A Comparative Evaluation of Unsupervised Anomaly Detection Algorithms for Multivariate Data”, *PLoS ONE*, Vol. 11, 2016.

24. Chandola, V., A. Banerjee and V. Kumar, “Anomaly Detection: A Survey”, *ACM Comput. Surv.*, Vol. 41, 07 2009.

25. Aggarwal, C. C., *Outlier Analysis*, Springer Publishing Company, Incorporated, 2013.

26. Ruff, L., R. Vandermeulen, N. Görnitz, A. Binder, E. Müller, K.-R. Müller and M. Kloft, “Deep Semi-Supervised Anomaly Detection”, , 06 2019.

27. Chalapathy, R. and S. Chawla, “Deep Learning for Anomaly Detection: A

Survey", , 01 2019.

28. Vespe, M., M. Gibin, A. Alessandrini, F. Natale, F. Mazzarella and G. Osio, "Mapping EU fishing activities using ship tracking data", *Journal of Maps*, 06 2016.

29. Mazzarella, F., M. Vespe, D. Damalas and G. Osio, "Discovering vessel activities at sea using AIS data: Mapping of fishing footprints", , 07 2014.

30. Pham, K., J. Boy and M. Luengo-Oroz, "Data Fusion to Describe and Quantify Search and Rescue Operations in the Mediterranean Sea", pp. 514–523, 10 2018.

31. Palma, A., V. Bogorny, B. Kuijpers and L. Alvares, "A clustering-based approach for discovering interesting places in trajectories", pp. 863–868, 03 2008.

32. Horn, S., C. Eisler, P. Dobias and J. Collins, "Data Requirements for Anomaly Detection", , 07 2016.

33. Fernandez Arguedas, V., F. Mazzarella and M. Vespe, "Spatio-temporal data mining for maritime situational awareness", , 05 2015.

34. Riveiro, M., G. Pallotta and M. Vespe, "Maritime anomaly detection: A review", *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, Vol. 8, p. e1266, 05 2018.

35. Ristic, B., B. Scala, M. Morelande and N. Gordon, "Statistical Analysis of Motion Patterns in AIS Data: Anomaly Detection and Motion Prediction", pp. 1–7, 01 2008.

36. Blauwkamp, D., T. Nguyen and G. Xie, "Toward a Deep Learning Approach to Behavior-based AIS Traffic Anomaly Detection", , 12 2018.

37. Pallotta, G., M. Vespe and K. Bryan, "Vessel Pattern Knowledge Discovery from

AIS Data: A Framework for Anomaly Detection and Route Prediction", *Entropy*, Vol. 15, pp. 2218–2245, 06 2013.

38. *Big Data Ocean*, `https://www.bigdataocean.eu/`, Accessed: 13.04.2020.

39. Zissis, V. S. B., Chatzikokolakis, "A data driven approach to maritime anomaly detection", .

40. Kotsiantis, S., "Decision trees: a recent overview", *Artificial Intelligence Review*, Vol. 39, pp. 261–283, 2011.

41. Rokach, L. and O. Maimon, *Decision Trees*, Vol. 6, pp. 165–192, 01 2005.

42. Statistics, L. B. and L. Breiman, "Random Forests", *Machine Learning*, pp. 5–32, 2001.

43. Natekin, A. and A. Knoll, "Gradient Boosting Machines, A Tutorial", *Frontiers in neurorobotics*, Vol. 7, p. 21, 12 2013.

44. Chen, T. and C. Guestrin, "XGBoost: A Scalable Tree Boosting System", pp. 785–794, 08 2016.

45. Zhang, Y., "Support Vector Machine Classification Algorithm and Its Application", C. Liu, L. Wang and A. Yang (Editors), *Information Computing and Applications*, pp. 179–186, Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

46. Evgeniou, T. and M. Pontil, "Support Vector Machines: Theory and Applications", Vol. 2049, pp. 249–257, 01 2001.

47. Hsu, C.-W., C.-C. Chang and C. Lin, "A Practical Guide to Support Vector Classication", , 2008.

48. Jolliffe, I. and J. Cadima, “Principal component analysis: A review and recent developments”, *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, Vol. 374, p. 20150202, 04 2016.

49. *Principal Component Analysis*, `https://builtin.com/data-science/step-step-explanation-principal-component-analysis/`, Accessed: 22.04.2020.

50. Xu, D. and Y. Tian, “A Comprehensive Survey of Clustering Algorithms”, *Annals of Data Science*, Vol. 2, 08 2015.

51. Jain, A. K. and R. C. Dubes, *Algorithms for Clustering Data*, Prentice-Hall, Inc., USA, 1988.

52. Reynolds, D., *Gaussian Mixture Models*, pp. 659–663, Springer US, Boston, MA, 2009, `https://doi.org/10.1007/978-0-387-73003-5_196`.

53. *Gaussian Mixture Model*, `https://brilliant.org/wiki/gaussian-mixture-model/`, Accessed: 16.08.2020.

54. Refaeilzadeh, P., L. Tang and H. Liu, “Cross-Validation”, *Encyclopedia of Database Systems*, Vol. 532–538, pp. 532–538, 01 2009.

55. Dietterich, T., “Ensemble methods in machine learning”, pp. 1–15, 01 2000.

56. *Large Marine Ecosystems of the World*, `http://www.marineregions.org/gazetteer.php?p=details&id=8542`, Accessed: 20.04.2020.

57. *Pandas merge_asof()*, `https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.merge_asof.html`, Accessed: 08.04.2020.

58. *Lessons Learned While Solving Time-Series Forecasting as a Supervised ML*

Problem: Part 1, `https://dzone.com/articles/lessons-learnt-while-solving-time-series-forecasti`.

59. Brownlee, J., *Deep Learning for Time Series Forecasting: Predict the Future with MLPs, CNNs and LSTMs in Python*, Machine Learning Mastery, 2018, `https://books.google.nl/books?id=o5qnDwAAQBAJ`.

60. Nanni, L., C. Fantozzi and N. Lazzarini, "Coupling different methods for overcoming the class imbalance problem", *Neurocomputing*, Vol. 158, 02 2015.

61. *Techniques to handle imbalanced data*, `https://www.kdnuggets.com/2017/06/7-techniques-handle-imbalanced-data.html`, Accessed: 08.05.2020.

62. Guyon, I. and A. Elisseeff, "An Introduction to Variable and Feature Selection", *J. Mach. Learn. Res.*, Vol. 3, pp. 1157–1182, 2003.

63. Adhikari, R. and R. Agrawal, *An Introductory Study on Time series Modeling and Forecasting*, 01 2013.

64. Isaac, E., "Test of Hypothesis - Concise Formula Summary", , 10 2015.

65. Schwarz, G., "Estimating the Dimension of a Model", *Annals of Statistics*, Vol. 6, pp. 461–464, 1978.

APPENDIX A: Additional results

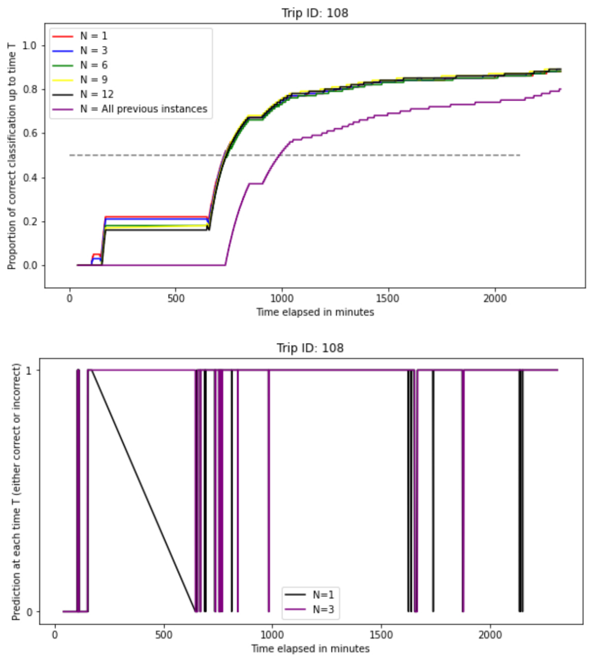

Figure A.1: Local plot analysis - Anomalous Trip ID = 108

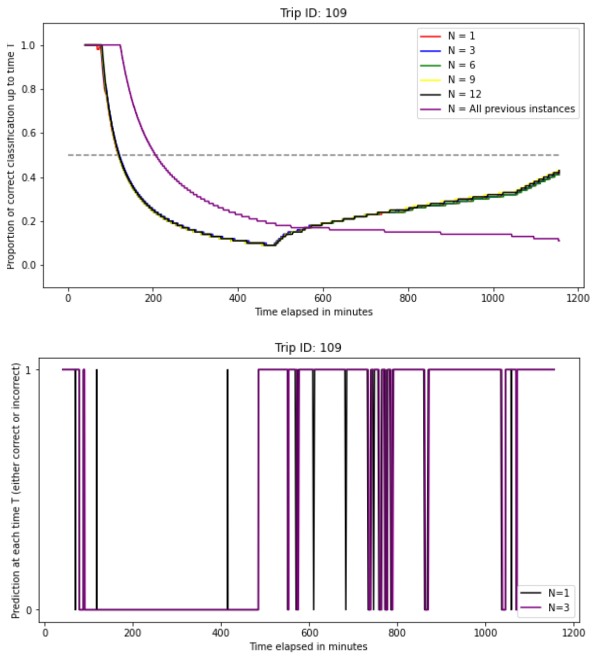

Figure A.2: Local plot analysis - Anomalous Trip ID = 109

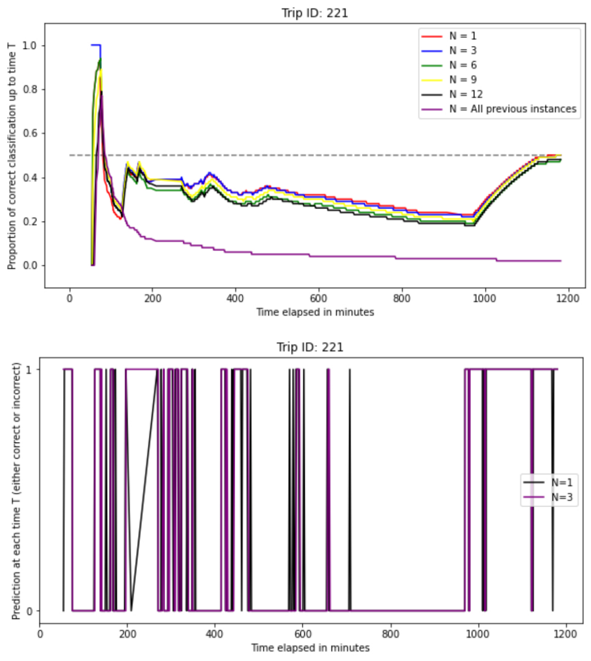

Figure A.3: Local plot analysis - Anomalous Trip ID = 221

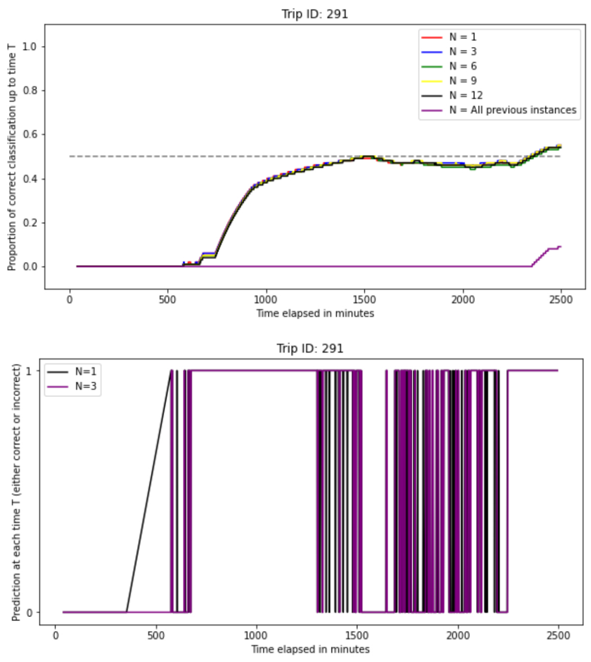

Figure A.4: Local plot analysis - Anomalous Trip ID = 291

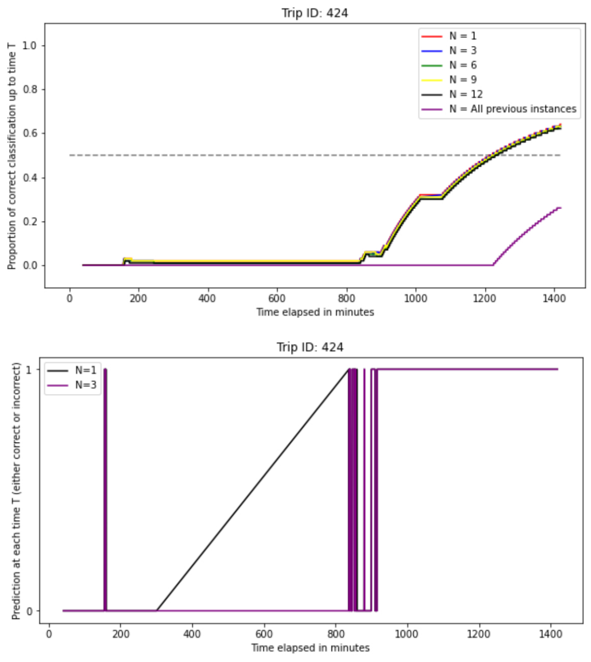

Figure A.5: Local plot analysis - Anomalous Trip ID = 424

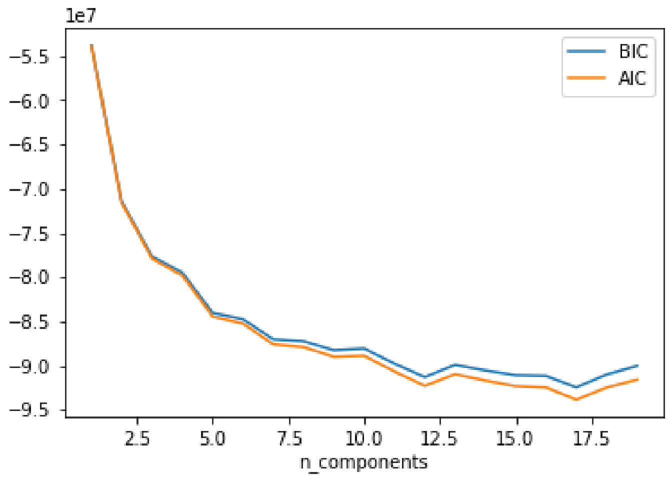

Figure A.6: Gaussian Mixture Model Selection - Input: Normal data

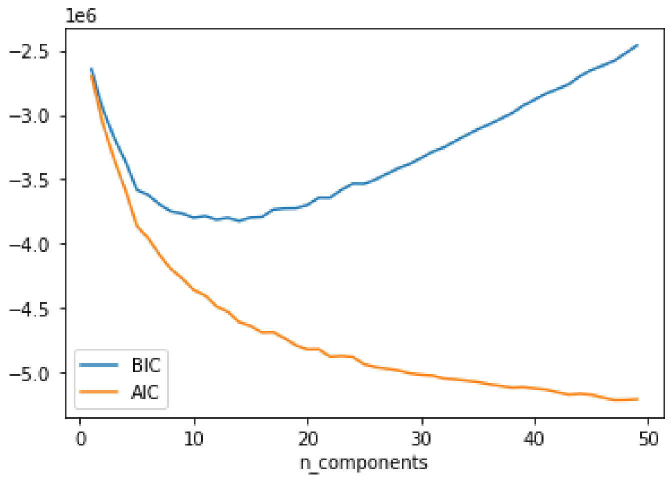

Figure A.7: Gaussian Mixture Model Selection - Input: Anomalous data